Package 'NeuDist'

January 16, 2026

Type Package

Title Univariate Continuous Distributions with Model Diagnostics

Version 1.0.1

Description Implements univariate continuous probability distributions and associated model diagnostics based on the Lindley, Logistic, Half-Cauchy, Half-Logistic, and Poisson families. Provides functions for probability density, cumulative distribution, quantile, and hazard evaluation, random variate generation, and diagnostic procedures including Q-Q and P-P plots, goodness-of-fit tests, and model selection criteria.

Maintainer Vijay Kumar <vkgkp@rediffmail.com>

License GPL-3

Encoding UTF-8

LazyData true

Imports stats, graphics, goftest

Author Vijay Kumar [aut, cre],
Laxmi Prasad Sapkota [aut],
Pankaj Kumar [aut],
Lal Babu Sah [aut]

RoxygenNote 7.3.3

Depends R (>= 3.5)

NeedsCompilation no

Repository CRAN

Date/Publication 2026-01-16 11:20:02 UTC

Contents

`NeuDist-package` *Univariate Continuous Distributions with Model Diagnostics*

Description

Tools for univariate continuous distributions with model diagnostics, based on the Lindley, Logistic, Half-Cauchy, Half-Logistic, and Poisson families, providing functions for probability density, distribution, quantile, and hazard evaluation, random variate generation, and generic diagnostic tools such as Q–Q and P–P plots, goodness-of-fit tests, and model selection criteria, with support for 58 distributions and 15 data sets.

Details

Distributions in the 'NeuDist' package:

`ChenExp`	`Chen-Exponential Distribution.`
`ExpoExpPower`	`Exponentiated Exponential Power Distribution.`
`ExpoInvChen`	`Exponentiated Inverse Chen Distribution.`
`GompertzExt`	`Gompertz Extension Distribution.`
`HCChen`	`Half-Cauchy Chen Distribution.`
`HCGenExp`	`Half-Cauchy Generalized Exponential Distribution.`

HCGenRayleigh	Half-Cauchy Generalized Rayleigh Distribution.
HCGompertz	Half-Cauchy Gompertz Distribution.
HCInvGPZ	Half-Cauchy Inverse Gompertz Distribution.
HCInvNHE	Half-Cauchy Inverse NHE Distribution.
HCNHE	Half-Cauchy exponential extension Distribution.
HLIW	Half Logistic Inverted Weibull Distribution.
HLNHE	Half Logistic NHE Distribution.
InvEEP	Inverse Exponentiated Exponential Poisson Distribution.
InvExpPower	Inverse Exponential Power Distribution.
InvGenGPZ	Inverse Generalized Gompertz Distribution.
InvPham	Inverse Pham Distribution.
InvPowerCauchy	Inverse Power Cauchy Distribution.
InvSGZ	Inverted Shifted Gompertz Distribution.
InvUBD	Inverse Upside Down Bathtub-Shaped Hazard Distribution.
LindleyChen	Lindley-Chen Distribution.
LindleyExpPower	Lindley Exponential Power Distribution.
LindleyGenInvExp	Lindley Generalized Inverted Exponential Distribution.
LindleyGompertz	Lindley Gompertz Distribution.
LindleyHC	New Lindley Half Cauchy Distribution.
LindleyInvExp	Lindley Inverse Exponential Distribution.
LindleyInvWeibull	Lindley inverse Weibull Distribution.
LindleyRayleigh	New Lindley-Rayleigh Distribution.
LogisChen	Logistic Chen Distribution Distribution.
LogisExpExt	Logistic Exponential Extension Distribution.
LogisExpPower	Logistic-Exponential Power Distribution.
LogisGompertz	Logistic Gompertz Distribution.
LogisInvExp	Logistic Inverse Exponential Distribution.
LogisInvLomax	Logistic Inverse Lomax Distribution.
LogisInvWeibull	Logistic Inverse Weibull Distribution.
LogisLomax	Logistic Lomax Distribution.
LogisModExp	Logistic-Modified Exponential Distribution.
LogisNHE	Logistic-NHE Distribution.
LogisRayleigh	Logistic-Rayleigh Distribution.
LogisWeib	Logistic-Weibull Distribution.
ModAtanExp	Modified Arctan Exponential Distribution.
ModGE	Modified Generalized Exponential Distribution.
ModInvGE	Modified Inverse Generalized Exponential Distribution.
ModInvLomax	Modified Inverse Lomax Distribution.
ModInvNHE	Modified Inverse NHE Distribution.
ModUbd	Modified Upside Down Bathtub Shaped Hazard Function
NewLindleyHC	New Lindley Half Cauchy Distribution.
Perks	Perks Distribution.
PoisInvWeib	Poisson Inverse Weibull Distribution.
PoissonChen	Poisson Chen Distribution.
PoissonExpPower	Poisson Exponential Power Distribution.
PoissonGenRayleigh	Poisson Generalized Rayleigh Distribution.
PoissonGPZ	Poisson Gompertz Distribution.
PoissonInvLomax	Poisson Inverted Lomax Distribution.

PoissonInvNHE	Poisson Inverse NHE Distribution.
PoissonInvSGZ	Poisson Inverse Shifted Gompertz Distribution.
PoissonNHE	Poisson NHE Distribution.
PoissonSGZ	Poisson Shifted Gompertz Distribution.

General functions:

gofic	Generic Goodness-of-Fit(GoF) and Model Diagnostics Function
pp.plot	Generic Probability-Probability(P-P) Plot Function
qq.plot	Generic Quantile-Quantile(Q-Q) Plot Function

Data:

bladder	Bladder Cancer Recurrence Times
conductors	Electromigration Failure Times of Microcircuit Conductors
fibers63	Strength of 63 Carbon Fibers at 10 mm Gauge Length
fibers65	Strength of 65 Carbon Fibers at 50 mm Gauge Length
fibers69	Tensile Strength of 69 Carbon Fibers at 20 mm Gauge Length
headneck44	Head and Neck Cancer Survival Times
rainfall	March Rainfall in Minneapolis/St. Paul
reactorpump	Failure Time Intervals of Secondary Reactor Pumps
relief	Relief Times of Patients Receiving an Analgesic
stress	Breaking Stress of Carbon Fibres
stress31	Fatigue Life of 6061-T6 Aluminum Coupons under 31,000 psi
stress66	Breaking Stress of 66 Carbon Fibers of Length 50 mm
survtimes	Survival Times of Guinea Pigs Infected with Tubercle Bacilli
waiting	Waiting Times of 100 Bank Customers
windshield	Service Times of Aircraft Windshields

Author(s)

Vijay Kumar <vkgkp@rediffmail.com>, Laxmi Prasad Sapkota <laxmi75@gmail.com>, Pankaj Kumar <pankajagadish@gmail.com>, Lal Babu Sah <lalbabu3131@gmail.com>

Maintainer: Vijay Kumar <vkgkp@rediffmail.com>

---

bladder *Bladder Cancer Recurrence Times*

Description

Recurrence times (in months) for bladder cancer patients, reported in Lee and Wang (2003). The dataset contains observed survival times without censoring information and is commonly used in survival analysis examples.

Usage

```
data(bladder)
```

Format

A numeric vector giving recurrence times (in months) for bladder cancer patients. A total of 128 observations are included.

Details

These recurrence times are widely used in demonstrations of survival analysis methods, including Kaplan–Meier estimation, hazard rate modelling, accelerated failure-time (AFT) models, and parametric distribution fitting. The dataset originally appears in Lee and Wang's *Statistical Methods for Survival Data Analysis* (3rd ed.), a standard reference text in biostatistics.

Note: The dataset provided here contains recurrence times only and does not include censoring indicators or covariates found in extended versions of the bladder cancer data.

Value

An object of class `"numeric"`.

The vector consists of 128 observed recurrence times (in months), each corresponding to a single bladder cancer patient. Each value represents the time from treatment or diagnosis to documented cancer recurrence. The dataset is commonly used in survival analysis and biostatistics to illustrate time-to-event modeling, including Kaplan–Meier estimation, hazard rate analysis, accelerated failure-time (AFT) models, and parametric survival distributions.

References

Lee, E. T., & Wang, J. W. (2003). *Statistical Methods for Survival Data Analysis* (3rd ed.). Wiley, New York.

Examples

```
data(bladder)

# Basic summary
summary(bladder)

# Histogram of recurrence times
hist(
  bladder,
  main = "Bladder Cancer Recurrence Times",
  xlab = "Time (months)"
)
```

`ChenExp` *Chen-Exponential Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Chen-Exponential distribution.

Usage

```
dchen.exp(x, alpha, beta, lambda, log = FALSE)
pchen.exp(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qchen.exp(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rchen.exp(n, alpha, beta, lambda)
hchen.exp(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Chen-Exponential distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Chen-Exponential distribution has CDF:

$$F(x;\, \alpha, \beta, \lambda) = 1 - \exp\left\{\lambda\left[1 - \exp\left\{\left(e^{\beta x} - 1\right)^{\alpha}\right\}\right]\right\}, \quad x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dchen.exp()` — Density function
- `pchen.exp()` — Distribution function
- `qchen.exp()` — Quantile function
- `rchen.exp()` — Random generation
- `hchen.exp()` — Hazard function

Value

- `dchen.exp`: numeric vector of (log-)densities
- `pchen.exp`: numeric vector of probabilities
- `qchen.exp`: numeric vector of quantiles
- `rchen.exp`: numeric vector of random variates
- `hchen.exp`: numeric vector of hazard values

References

Chen, Z. (2000). A new two-parameter lifetime distribution with bathtub shape or increasing failure rate function. *Statistics & Probability Letters*, **49**, 155–161.

Sapkota, L.P., & Kumar, V. (2023). Chen Exponential Distribution with Applications to Engineering Data. *International Journal of Statistics and Reliability Engineering*, **10(1)**, 33–47.

Sapkota, L.P., Alsahangiti, A.M., Kumar, V. Gemeay, A.M., Bakr, M.E., Balogun, O.S., & Muse, A.H. (2023). Arc-Tangent Exponential Distribution With Applications to Weather and Chemical Data Under Classical and Bayesian Approach, *IEEE Access*, **11**, 115462–115476. doi:10.1109/ACCESS.2023.3324293

Examples

```
x <- seq(0.1, 1, 0.1)
dchen.exp(x, 1.5, 0.8, 2)
pchen.exp(x, 1.5, 0.8, 2)
qchen.exp(0.5, 1.5, 0.8, 2)
rchen.exp(10, 1.5, 0.8, 2)
hchen.exp(x, 1.5, 0.8, 2)
#Data
x <- stress
#ML Estimates
params = list(alpha=2.5462, beta=0.0537, lambda=87.6028)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pchen.exp, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qchen.exp, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
# Display plot; numerical summary stored in 'out'
out <- gofic(x, params = params, dfun = dchen.exp,
             pfun = pchen.exp, plot=TRUE)
print.gofic(out)
```

---

conductors *Electromigration Failure Times of Microcircuit Conductors*

Description

Failure-time data from an accelerated life test involving 59 microcircuit conductors. Electromigration refers to the movement of atoms in conductors under high current density, leading to eventual failure. The dataset contains observed failure times (in hours), with no censored observations.

Usage

```
data(conductors)
```

Format

A numeric vector of length 59 giving failure times in hours.

Details

Electromigration is a major wear-out mechanism in thin-film microelectronic circuits. Because electric current accelerates atomic migration, accelerated life tests are widely used to study the reliability of conductors. This dataset has been used extensively in the reliability literature, including analyses involving Weibull, lognormal, and power-lognormal lifetime models.

Value

An object of class `"numeric"`.

The vector consists of 59 observed failure times (in hours), each corresponding to a single microcircuit conductor subjected to an accelerated life test. Each value represents the elapsed operating time until failure caused by electromigration. The dataset is commonly used in reliability engineering and lifetime data analysis to illustrate wear-out mechanisms and to fit and compare parametric lifetime models such as the Weibull, lognormal, and power-lognormal distributions.

References

Lawless, J. F. (2003). *Statistical Models and Methods for Lifetime Data*. John Wiley & Sons.

Nelson, W., & Doganaksoy, N. (1995). Statistical analysis of life or strength data from specimens of various sizes using the power-(log)normal model. *Recent Advances in Life-Testing and Reliability*, 377–408.

Examples

```
data(conductors)

# Summary statistics
summary(conductors)

# Histogram of failure times
hist(conductors)
```

ExpoExpPower *Exponentiated Exponential Power (EEP) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Exponentiated Exponential Power (EEP) distribution.

Usage

```
dgen.exp.power(x, alpha, lambda, theta, log = FALSE)
pgen.exp.power(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qgen.exp.power(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rgen.exp.power(n, alpha, lambda, theta)
hgen.exp.power(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The EEP distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Exponentiated Exponential Power (EEP) distribution has CDF:

$$F(x;\, \alpha,\, \lambda, \theta) = \left[1 - \exp\left\{1 - \exp\left(\lambda x^{\alpha}\right)\right\}\right]^{\theta} \quad ; \quad x > 0$$

where α, λ, and θ are the parameters.

The implementation includes the following functions:

- `dgen.exp.power()` — Density function
- `pgen.exp.power()` — Distribution function
- `qgen.exp.power()` — Quantile function
- `rgen.exp.power()` — Random generation
- `hgen.exp.power()` — Hazard function

Value

- `dgen.exp.power`: numeric vector of (log-)densities
- `pgen.exp.power`: numeric vector of probabilities
- `qgen.exp.power`: numeric vector of quantiles
- `rgen.exp.power`: numeric vector of random variates
- `hgen.exp.power`: numeric vector of hazard values

References

Sapkota, L.P., & Kumar, V.(2024). Bayesian Analysis of Exponentiated Exponential Power Distribution under Hamiltonian Monte Carlo Method, Statistics and Applications. *Statistics and Applications*, **22(2)**, 231–258.

Srivastava, A.K., & Kumar, V.(2011). Analysis of Software Reliability Data using Exponential Power Model. *International Journal of Advanced Computer Science and Applications*, **2(2)**, 38–45, doi:10.14569/IJACSA.2011.020208

Chen, Z.(1999). Statistical inference about the shape parameter of the exponential power distribution, *Statistical Papers*, **40**, 459–468.

Smith, R.M., & Bain, L.J. (1975). An exponential power life-test distribution. *IEEE Communications in Statistics*, **4**, 469–481.

Examples

```
x <- seq(0.1, 1, 0.1)
dgen.exp.power(x, 1.5, 0.8, 2)
pgen.exp.power(x, 1.5, 0.8, 2)
qgen.exp.power(0.5, 1.5, 0.8, 2)
rgen.exp.power(10, 1.5, 0.8, 2)
hgen.exp.power(x, 1.5, 0.8, 2)
#Data
x <- waiting
#ML Estimates
params = list(alpha=0.3407, lambda=0.6068, theta=7.6150)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pgen.exp.power, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qgen.exp.power, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
# Neither plot nor console output; results stored in 'out'
out <- gofic(x, params = params,
             dfun = dgen.exp.power, pfun = pgen.exp.power, plot=FALSE)
print.gofic(out)
```

ExpoInvChen *Exponentiated Inverse Chen Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Exponentiated Inverse Chen distribution.

Usage

```
dexpo.inv.chen(x, alpha, lambda, theta, log = FALSE)
pexpo.inv.chen(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qexpo.inv.chen(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rexpo.inv.chen(n, alpha, lambda, theta)
hexpo.inv.chen(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Exponentiated Inverse Chen distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Exponentiated Inverse Chen distribution has CDF:

$$F(x; \alpha, \lambda, \theta) = 1 - \left[1 - \exp\left(\lambda\left(1 - \exp(x^{-\alpha})\right)\right)\right]^{\theta}, \quad x > 0.$$

where α, λ, and θ are the parameters.

The functions available are listed below:

- `dexpo.inv.chen()` — Density function
- `pexpo.inv.chen()` — Distribution function
- `qexpo.inv.chen()` — Quantile function
- `rexpo.inv.chen()` — Random generation
- `hexpo.inv.chen()` — Hazard function

Value

- `dexpo.inv.chen`: numeric vector of (log-)densities
- `pexpo.inv.chen`: numeric vector of probabilities
- `qexpo.inv.chen`: numeric vector of quantiles
- `rexpo.inv.chen`: numeric vector of random variates
- `hexpo.inv.chen`: numeric vector of hazard values

References

Telee, L. B. S., & Kumar, V. (2023). Exponentiated Inverse Chen distribution: Properties and applications. *Journal of Nepalese Management Academia*, **1(1)**, 53–62. doi:10.3126/jnma.v1i1.62033

Srivastava, A.K., & Kumar, V.(2011). Markov Chain Monte Carlo Methods for Bayesian Inference of the Chen Model. *International Journal of Computer Information Systems*, **2(2)**, 7–14.

Examples

```
x <- seq(2, 5, 0.25)
dexpo.inv.chen(x, 0.5, 2.5, 1.5)
pexpo.inv.chen(x, 0.5, 2.5, 1.5)
qexpo.inv.chen(0.5, 0.5, 2.5, 1.5)
rexpo.inv.chen(10, 0.5, 2.5, 1.5)
hexpo.inv.chen(x, 0.5, 2.5, 1.5)

# Data
x <- headneck44
# ML estimates
params = list(alpha=0.3947, lambda=15.5330, theta=8.1726)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pexpo.inv.chen, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qexpo.inv.chen, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
# Display plot and print numerical summary
gofic(x, params = params,
      dfun = dexpo.inv.chen, pfun=pexpo.inv.chen, plot=TRUE, verbose = TRUE)
```

---

fibers63 *Strength of 63 Carbon Fibers at 10 mm Gauge Length*

Description

Measurements of tensile strength (in gigapascals, GPa) for 63 single carbon fibers tested at a gauge length of 10 mm. These data were originally reported by Bader and Priest (1982) in their study of fibre and bundle strength in hybrid composites.

Usage

```
fibers63
```

Format

A numeric vector of length 63 containing tensile strength measurements (in GPa).

Details

The dataset contains tensile strength values for individual carbon fibers cut to a gauge length of 10 mm. This dataset has been used extensively in materials science and reliability studies for modeling strength distributions and assessing variability in carbon fiber performance.

The data originate from the same experimental study that produced several related carbon-fiber datasets (e.g., `fibers65`, `fibers69`).

Value

An object of class `"numeric"`.

The vector consists of 63 observed tensile strength measurements (in gigapascals), each corresponding to an individual carbon fiber tested at a gauge length of 10 mm. Each value represents the breaking strength of a single fiber specimen. The dataset is commonly used in materials science and reliability engineering for modeling strength distributions, assessing variability, and fitting parametric lifetime or strength models.

References

Bader, M. G., & Priest, A. M. (1982). Statistical aspects of fibre and bundle strength in hybrid composites. *Progress in Science and Engineering of Composites*, 1129–1136.

Examples

```
data(fibers63)

summary(fibers63)

hist(
  fibers63,
  main = "Tensile Strength of Carbon Fibers (10 mm Gauge Length)",
  xlab = "Strength (GPa)"
)
```

---

fibers65 *Strength of 65 Carbon Fibers at 50 mm Gauge Length*

Description

Tensile strength measurements (in gigapascals, GPa) for 65 carbon fibers tested under tension at a gauge length of 50 mm. These data were originally reported by Bader and Priest (1982) in their foundational study on fibre and bundle strength in hybrid composites.

Usage

```
fibers65
```

Format

A numeric vector of length 65 containing tensile strength values (in GPa).

Details

The fibers were tested at a gauge length of 50 mm to study the variability of carbon fiber strength under controlled conditions. This dataset is frequently used in reliability analysis, composite material modeling, and strength distribution studies. It is one of several datasets originating from the Bader and Priest (1982) carbon-fiber experiments.

Value

An object of class `"numeric"`.

The vector contains 65 observed tensile strength measurements (in gigapascals) of individual carbon fibers tested at a gauge length of 50 mm. Each element represents the breaking strength of a single fiber specimen. The dataset is typically used as input for statistical modeling, reliability analysis, and lifetime or strength distribution studies in composite materials research.

References

Bader, M. G., & Priest, A. M. (1982). Statistical aspects of fibre and bundle strength in hybrid composites. *Progress in Science and Engineering of Composites*, 1129–1136.

Examples

```
data(fibers65)

summary(fibers65)

plot(
  fibers65,
  ylab = "Strength (GPa)",
  main = "Carbon Fiber Strength (50 mm Gauge Length)"
)

hist(
  fibers65,
  main = "Histogram of Carbon Fiber Strength",
  xlab = "Strength (GPa)"
)
```

---

fibers69 *Tensile Strength of 69 Carbon Fibers at 20 mm Gauge Length*

Description

Measurements of tensile strength (in gigapascals, GPa) for 69 carbon fibers tested under tension at a gauge length of 20 mm. These data were originally reported by Bader and Priest (1982) in their study of fibre and bundle strength in hybrid composites.

Usage

```
fibers69
```

Format

A numeric vector of length 69 containing tensile strength values (in GPa).

Details

This dataset has been widely used in composite-material and reliability studies, particularly for modeling strength distributions of carbon fibers. The original experiment measured the tensile strength of individual fibers at a gauge length of 20 mm, providing insight into the statistical behavior of fiber strength under tension.

Value

An object of class `"numeric"`.

The vector consists of 69 tensile strength measurements (in gigapascals) corresponding to individual carbon fiber specimens tested at a gauge length of 20 mm. Each value represents the breaking strength of a single fiber. The dataset is commonly used for statistical analysis of strength distributions, reliability modeling, and comparative studies of gauge-length effects in composite materials.

References

Bader, M. G., & Priest, A. M. (1982). Statistical aspects of fibre and bundle strength in hybrid composites. *Progress in Science and Engineering of Composites*, 1129–1136.

Examples

```
data(fibers69)

summary(fibers69)

hist(
  fibers69,
  main = "Tensile Strength of Carbon Fibers (20 mm Gauge Length)",
  xlab = "Strength (GPa)"
)
```

---

gofic *Generic Goodness-of-Fit(GoF) and Model Diagnostics Function*

Description

Computes log-likelihood, information criteria (AIC, BIC, AICC, HQIC) and classical goodness-of-fit statistics (Kolmogorov–Smirnov, Cramér–von Mises, Anderson–Darling) for a given numeric data vector and user-supplied density and distribution functions.

Usage

```
gofic(x, params, dfun, pfun, plot = TRUE, verbose = FALSE)
```

Arguments

x	Numeric vector of observed data. Must contain at least two values.
params	Named list of model parameters passed to dfun and pfun.
dfun	A probability density function with signature dfun(x, ...) returning numeric densities.
pfun	A cumulative distribution function with signature pfun(q, ...) returning cumulative probabilities.
plot	Logical; if TRUE, plots empirical vs theoretical CDF. Default is TRUE.
verbose	Logical; if TRUE, prints the output object. Default is FALSE.

Details

Optionally plots the empirical cumulative distribution function (ECDF) against the theoretical cumulative distribution function.

The supplied dfun and pfun must accept arguments x and q respectively, followed by named model parameters. Density values must be finite and positive; non-positive densities trigger a warning but computation proceeds.

Value

An object of class "gofic" containing:

- logLik Numeric; log-likelihood value.
- AIC Akaike Information Criterion.
- BIC Bayesian Information Criterion.
- AICC Corrected Akaike Information Criterion.
- HQIC Hannan–Quinn Information Criterion.
- KS Object returned by stats::ks.test().
- CVM Object returned by goftest::cvm.test().
- AD Object returned by goftest::ad.test().
- n Sample size.
- params Model parameters supplied.

The object is returned invisibly.

See Also

print.gofic, ks.test, cvm.test, ad.test

Examples

```
# Example 1 with built-in Weibull distribution

set.seed(123)
x <- rweibull(100, shape = 2, scale = 1)
out <- gofic(x, params = list(shape = 2, scale = 1),
      dfun = dweibull, pfun = pweibull, plot=FALSE)
out


# Example 2: For a user defined distribution
# Goodness-of-Fit(GoF) and Model Diagnostics for Chen-Exponential distribution
#Data
x <- stress
#ML Estimates
params = list(alpha=2.5462, beta=0.0537, lambda=87.6028)
# Display plot and print numerical summary
gofic(x, params = params,
        dfun = dchen.exp, pfun = pchen.exp, plot = TRUE, verbose = TRUE)

# Display plot only (no numerical summary)
gofic(x, params = params,
        dfun = dchen.exp, pfun = pchen.exp, plot = TRUE, verbose = FALSE)

# Print numerical summary only (no plot)
gofic(x, params = params,
        dfun = dchen.exp, pfun = pchen.exp, plot = FALSE, verbose = TRUE)

# Display plot; numerical summary stored in 'out'
out <- gofic(x, params = params,
              dfun = dchen.exp, pfun = pchen.exp, plot = TRUE, verbose = FALSE)
print.gofic(out)

# Neither plot nor console output; results stored in 'out'
out <- gofic(x, params = params,
               dfun = dchen.exp, pfun = pchen.exp, plot = FALSE, verbose = FALSE)
print.gofic(out)
```

`GompertzExt` *Gompertz Extension Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Gompertz Extension distribution.

Usage

```
dgompertz.ext(x, alpha, lambda, theta, log = FALSE)
pgompertz.ext(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qgompertz.ext(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rgompertz.ext(n, alpha, lambda, theta)
hgompertz.ext(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Gompertz Extension distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Gompertz Extension distribution has CDF:

$$F(x; \alpha, \lambda, \theta) = \quad 1 - \exp\left\{-\lambda \left(e^{\alpha x} - 1\right)^{\theta}\right\} \quad ; \; x > 0.$$

where α, λ, and θ are the parameters.

The functions available are listed below:

- `dgompertz.ext()` — Density function
- `pgompertz.ext()` — Distribution function
- `qgompertz.ext()` — Quantile function
- `rgompertz.ext()` — Random generation
- `hgompertz.ext()` — Hazard function

Value

- `dgompertz.ext`: numeric vector of (log-)densities
- `pgompertz.ext`: numeric vector of probabilities
- `qgompertz.ext`: numeric vector of quantiles
- `rgompertz.ext`: numeric vector of random variates
- `hgompertz.ext`: numeric vector of hazard values

References

Chaudhary, A.K., & Kumar, V. (2020). A Bayesian Estimation and Prediction of Gompertz Extension Distribution Using the MCMC Method. *Nepal Journal of Science and Technology(NJST)*, **19(1)**, 142–160. doi:10.3126/njst.v19i1.29795

Examples

```
x <- seq(1.0, 10, 0.25)
dgompertz.ext(x, 0.1, 5.0, 2.5)
pgompertz.ext(x, 0.1, 5.0, 2.5)
qgompertz.ext(0.5, 0.1, 5.0, 2.5)
rgompertz.ext(10, 0.1, 5.0, 2.5)
hgompertz.ext(x, 0.1, 5.0, 2.5)

# Data
x <- stress
# ML estimates
params = list(alpha=0.0678, lambda=44.34760, theta=2.5225)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pgompertz.ext, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qgompertz.ext, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dgompertz.ext, pfun=pgompertz.ext, plot=TRUE)
print.gofic(out)
```

HCChen *Half-Cauchy Chen Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Half-Cauchy Chen distribution.

Usage

```
dhc.chen(x, beta, lambda, theta, log = FALSE)
phc.chen(q, beta, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qhc.chen(p, beta, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rhc.chen(n, beta, lambda, theta)
hhc.chen(x, beta, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
beta	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Half-Cauchy Chen distribution is parameterized by the parameters $\beta > 0$, $\lambda > 0$, and $\theta > 0$.

The Half-Cauchy Chen distribution has CDF:

$$F(x;\beta,\lambda,\theta) = \quad \frac{2}{\pi}\arctan\left\{-\frac{\lambda}{\theta}(1-e^{x^{\beta}})\right\} \quad ; \; x > 0.$$

where β, λ, and θ are the parameters.

Included functions are:

- `dhc.chen()` — Density function
- `phc.chen()` — Distribution function
- `qhc.chen()` — Quantile function
- `rhc.chen()` — Random generation
- `hhc.chen()` — Hazard function

Value

- `dhc.chen`: numeric vector of (log-)densities
- `phc.chen`: numeric vector of probabilities
- `qhc.chen`: numeric vector of quantiles
- `rhc.chen`: numeric vector of random variates
- `hhc.chen`: numeric vector of hazard values

References

Chaudhary, A.K., Yadav, R.S., & Kumar, V.(2023). Half-Cauchy Chen Distribution with Theories and Applications. *Journal of Institute of Science and Technology*, **28(1)**, 45–55. doi:10.3126/jist.v28i1.56494

Polson, N.G., & Scott, J.G. (2012). On the half-Cauchy prior for a global scale parameter. *Bayesian Analysis*, **7(4)**, 887–902.

Telee, L.B.S., & Kumar, V.(2024). Arctan-Chen Distribution with Properties and Application. *International Journal of Statistics and Reliability Engineering*, **11(1)**, 93–100.

Examples

```
x <- seq(1.0, 5, 0.25)
dhc.chen(x, 2.0, 0.5, 2.5)
phc.chen(x, 2.0, 0.5, 2.5)
qhc.chen(0.5, 2.0, 0.5, 2.5)
rhc.chen(10, 2.0, 0.5, 2.5)
hhc.chen(x, 2.0, 0.5, 2.5)

# Data
x <- conductors
# ML estimates
params = list(beta=0.9753, lambda=0.0398, theta=29.0272)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = phc.chen, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qhc.chen, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
res <- gofic(x, params = params,
             dfun = dhc.chen, pfun=phc.chen, plot=FALSE)
print.gofic(res)
```

HCGenExp *Half-Cauchy Generalized Exponential(HCGE) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Half-Cauchy Generalized Exponential(HCGE) distribution.

Usage

```
dhc.gen.exp(x, alpha, lambda, theta, log = FALSE)
phc.gen.exp(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qhc.gen.exp(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rhc.gen.exp(n, alpha, lambda, theta)
hhc.gen.exp(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.

log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The HCGE distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The HCGE distribution has CDF:

$$F(x; \alpha, \lambda, \theta) = \quad 1 - \frac{2}{\pi} \arctan\left[-\frac{\alpha}{\theta} \ln\left(1 - e^{-\lambda x}\right)\right] \quad ; \; x > 0.$$

where α, λ, and θ are the parameters.

The implementation includes the following functions:

- `dhc.gen.exp()` — Density function
- `phc.gen.exp()` — Distribution function
- `qhc.gen.exp()` — Quantile function
- `rhc.gen.exp()` — Random generation
- `hhc.gen.exp()` — Hazard function

Value

- `dhc.gen.exp`: numeric vector of (log-)densities
- `phc.gen.exp`: numeric vector of probabilities
- `qhc.gen.exp`: numeric vector of quantiles
- `rhc.gen.exp`: numeric vector of random variates
- `hhc.gen.exp`: numeric vector of hazard values

References

Chaudhary, A.K., Sapkota, L.P. & Kumar, V. (2022). Half-Cauchy Generalized Exponential Distribution:Theory and Application. *Journal of Nepal Mathematical Society (JNMS)*, **5(2)**, 1–10. doi:10.3126/jnms.v5i2.50018

Gupta, R. D., & Kundu, D. (1999). Generalized exponential distributions. *Australian and New Zealand Journal of Statistics*, **41(2)**, 173–188.

Examples

```
x <- seq(0.1, 10, 0.2)
dhc.gen.exp(x, 2.0, 0.5, 0.1)
phc.gen.exp(x, 2.0, 0.5, 0.1)
qhc.gen.exp(0.5, 2.0, 0.5, 0.1)
rhc.gen.exp(10, 2.0, 0.5, 0.1)
hhc.gen.exp(x, 2.0, 0.5, 0.1)

# Data
```

```
x <- conductors
# ML estimates
params = list(alpha=6.6141, lambda=0.9352, theta=0.0103)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = phc.gen.exp, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qhc.gen.exp, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
res <- gofic(x, params = params,
             dfun = dhc.gen.exp, pfun=phc.gen.exp, plot=FALSE)
print.gofic(res)
```

HCGenRayleigh *Half-Cauchy Generalized Rayleigh Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Half-Cauchy Generalized Rayleigh distribution.

Usage

```
dhc.gen.rayleigh(x, alpha, lambda, theta, log = FALSE)
phc.gen.rayleigh(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qhc.gen.rayleigh(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rhc.gen.rayleigh(n, alpha, lambda, theta)
hhc.gen.rayleigh(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Half-Cauchy Generalized Rayleigh distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Half-Cauchy Generalized Rayleigh distribution has CDF:

$$F(x; \alpha, \lambda, \theta) = \quad 1 - \frac{2}{\pi} \arctan\left\{-\frac{\alpha}{\theta} \log\left\{1 - e^{-(\lambda x)^2}\right\}\right\} \quad ; \; x > 0.$$

where α, λ, and θ are the parameters.

The implementation includes the following functions:

- `dhc.gen.rayleigh()` — Density function
- `phc.gen.rayleigh()` — Distribution function
- `qhc.gen.rayleigh()` — Quantile function
- `rhc.gen.rayleigh()` — Random generation
- `hhc.gen.rayleigh()` — Hazard function

Value

- `dhc.gen.rayleigh`: numeric vector of (log-)densities
- `phc.gen.rayleigh`: numeric vector of probabilities
- `qhc.gen.rayleigh`: numeric vector of quantiles
- `rhc.gen.rayleigh`: numeric vector of random variates
- `hhc.gen.rayleigh`: numeric vector of hazard values

References

Sapkota, L.P., & Kumar, V. (2023). Half-Cauchy Generalized Rayleigh : Theory and Applications.*South East Asian J. Math. & Math. Sc.*, **19(1)**, 335–360. doi:10.56827/SEAJMMS.2023.1901.27

Shrestha, S.K., & Kumar, V. (2014). Bayesian Analysis for the Generalized Rayleigh Distribution. *International Journal of Statistika and Mathematika*, **9(3)**, 118–131.

Kundu, D., & Raqab, M.Z. (2005). Generalized Rayleigh Distribution: Different Methods of Estimation. *Computational Statistics and Data Analysis*, **49**, 187–200.

Examples

```
x <- seq(1.0, 5, 0.25)
dhc.gen.rayleigh(x, 2.0, 0.5, 0.1)
phc.gen.rayleigh(x, 2.0, 0.5, 0.1)
qhc.gen.rayleigh(0.5, 2.0, 0.5, 0.1)
rhc.gen.rayleigh(10, 2.0, 0.5, 0.1)
hhc.gen.rayleigh(x, 2.0, 0.5, 0.1)

# Data
x <- stress66
# ML estimates
params = list(alpha=1.4585, lambda=0.5300, theta=0.1655)
```

```
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = phc.gen.rayleigh, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qhc.gen.rayleigh, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dhc.gen.rayleigh, pfun=phc.gen.rayleigh, plot=FALSE)
print.gofic(out)
```

HCGompertz *Half-Cauchy Gompertz Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Half-Cauchy Gompertz distribution.

Usage

```
dhc.gpz(x, alpha, lambda, theta, log = FALSE)
phc.gpz(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qhc.gpz(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rhc.gpz(n, alpha, lambda, theta)
hhc.gpz(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Half-Cauchy Gompertz distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Half-Cauchy Gompertz distribution has CDF:

$$F(x; \alpha, \lambda, \theta) = \frac{2}{\pi} \arctan\left\{-\frac{\lambda}{\alpha\theta}\left(1 - e^{\alpha x}\right)\right\} \quad ; \; x > 0.$$

where α, λ, and θ are the parameters.

The implementation includes the following functions:

- `dhc.gpz()` — Density function
- `phc.gpz()` — Distribution function
- `qhc.gpz()` — Quantile function
- `rhc.gpz()` — Random generation
- `hhc.gpz()` — Hazard function

Value

- `dhc.gpz`: numeric vector of (log-)densities
- `phc.gpz`: numeric vector of probabilities
- `qhc.gpz`: numeric vector of quantiles
- `rhc.gpz`: numeric vector of random variates
- `hhc.gpz`: numeric vector of hazard values

References

Sah, L.B., & Kumar, V. (2019). Half-Cauchy Gompertz Distribution : Different Methods of Estimation, *Journal of National Academy of Mathematics*, **33**, 51–65.

Examples

```
x <- seq(1.0, 5, 0.25)
dhc.gpz(x, 2.0, 0.5, 2.5)
phc.gpz(x, 2.0, 0.5, 2.5)
qhc.gpz(0.5, 2.0, 0.5, 2.5)
rhc.gpz(10, 2.0, 0.5, 2.5)
hhc.gpz(x, 2.0, 0.5, 2.5)

# Data
x <- stress66
# ML estimates
params = list(alpha=1.6660, lambda=0.0328, theta=2.0578)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = phc.gpz, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
```

```
qq.plot(x, params = params, qfun = qhc.gpz, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params, dfun=dhc.gpz, pfun=phc.gpz, plot=TRUE)
print.gofic(out)
```

HCInvGPZ *Half-Cauchy Inverse Gompertz Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Half-Cauchy Inverse Gompertz distribution.

Usage

```
dhc.inv.gpz(x, alpha, lambda, theta, log = FALSE)
phc.inv.gpz(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qhc.inv.gpz(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rhc.inv.gpz(n, alpha, lambda, theta)
hhc.inv.gpz(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Half-Cauchy Inverse Gompertz distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Half-Cauchy Inverse Gompertz distribution has CDF:

$$F(x;\alpha,\lambda,\theta) = \quad 1 - \frac{2}{\pi}\arctan\left\{-\frac{\lambda}{\alpha\theta}\left(1-e^{\alpha/x}\right)\right\} \quad ;\ x > 0.$$

where α, λ, and θ are the parameters.

The implementation includes the following functions:

- `dhc.inv.gpz()` — Density function
- `phc.inv.gpz()` — Distribution function
- `qhc.inv.gpz()` — Quantile function
- `rhc.inv.gpz()` — Random generation
- `hhc.inv.gpz()` — Hazard function

Value

- `dhc.inv.gpz`: numeric vector of (log-)densities
- `phc.inv.gpz`: numeric vector of probabilities
- `qhc.inv.gpz`: numeric vector of quantiles
- `rhc.inv.gpz`: numeric vector of random variates
- `hhc.inv.gpz`: numeric vector of hazard values

References

Chaudhary, A. K., Yadav, R. S., & Kumar, V. (2022). Half-Cauchy Inverse Gompertz distribution: Theory and applications. *International Journal of Statistics and Applied Mathematics*, **7**(5), 94–102. doi:10.22271/maths.2022.v7.i5b.885

Examples

```
x <- seq(1.0, 10, 0.25)
dhc.inv.gpz(x, 2.0, 0.5, 2.5)
phc.inv.gpz(x, 2.0, 0.5, 2.5)
qhc.inv.gpz(0.5, 2.0, 0.5, 2.5)
rhc.inv.gpz(10, 2.0, 0.5, 2.5)
hhc.inv.gpz(x, 2.0, 0.5, 2.5)

# Data
x <- relief
# ML estimates
params = list(alpha=9.0830, lambda=0.8369, theta=17.9925)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = phc.inv.gpz, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qhc.inv.gpz, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dhc.inv.gpz, pfun=phc.inv.gpz, plot=TRUE)
print.gofic(out)
```

HCInvNHE *Half-Cauchy Inverse NHE Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Half-Cauchy Inverse NHE distribution.

Usage

```
dhc.inv.NHE(x, beta, lambda, theta, log = FALSE)
phc.inv.NHE(q, beta, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qhc.inv.NHE(p, beta, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rhc.inv.NHE(n, beta, lambda, theta)
hhc.inv.NHE(x, beta, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
beta	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Half-Cauchy Inverse NHE distribution is parameterized by the parameters $\beta > 0$, $\lambda > 0$, and $\theta > 0$.

The Half-Cauchy Inverse NHE distribution has CDF:

$$F(x;\beta,\lambda,\theta) = \quad 1 - \frac{2}{\pi}\arctan\left[-\frac{1}{\theta}\left\{1-\left(1+\frac{\lambda}{x}\right)^{\beta}\right\}\right] \quad ;\ x > 0.$$

where β, λ, and θ are the parameters.

Included functions are:

- dhc.inv.NHE() — Density function
- phc.inv.NHE() — Distribution function
- qhc.inv.NHE() — Quantile function
- rhc.inv.NHE() — Random generation
- hhc.inv.NHE() — Hazard function

Value

- dhc.inv.NHE: numeric vector of (log-)densities
- phc.inv.NHE: numeric vector of probabilities
- qhc.inv.NHE: numeric vector of quantiles
- rhc.inv.NHE: numeric vector of random variates
- hhc.inv.NHE: numeric vector of hazard values

References

Chaudhary, A.K., Telee, L.B.S. & Kumar,V. (2022). Half-Cauchy Inverse NHE Distribution: Properties and Applications. *Nepal Journal of Mathematical Sciences (NJMS)*, **3(2)**, 1–12. doi:10.3126/njmathsci.v3i2.49198

Chaudhary, A. K., Sapkota, L. P., & Kumar, V. (2022). Some properties and applications of half Cauchy extended exponential distribution. *Int. J. Stat. Appl. Math.*, **7(4)**, 226–235. doi:10.22271/maths.2022.v7.i4c.866

Chaudhary, A.K., & Kumar, V. (2022). Half Cauchy-Modified Exponential Distribution: Properties and Applications. *Nepal Journal of Mathematical Sciences (NJMS)*, **3(1)**, 47–58. doi:10.3126/njmathsci.v3i1.44125

Examples

```
x <- seq(1.0, 5, 0.25)
dhc.inv.NHE(x, 2.0, 0.5, 2.5)
phc.inv.NHE(x, 2.0, 0.5, 2.5)
qhc.inv.NHE(0.5, 2.0, 0.5, 2.5)
rhc.inv.NHE(10, 2.0, 0.5, 2.5)
hhc.inv.NHE(x, 2.0, 0.5, 2.5)

# Data
x <- relief
# ML estimates
params = list(beta=79.7799, lambda=0.1129, theta=154.1769)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = phc.inv.NHE, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qhc.inv.NHE, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
res <- gofic(x, params = params,
             dfun = dhc.inv.NHE, pfun=phc.inv.NHE, plot=FALSE)
print.gofic(res)
```

HCNHE *Half-Cauchy NHE Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Half-Cauchy NHE distribution.

Usage

```
dhc.NHE(x, beta, lambda, theta, log = FALSE)
phc.NHE(q, beta, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qhc.NHE(p, beta, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rhc.NHE(n, beta, lambda, theta)
hhc.NHE(x, beta, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
beta	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Half-Cauchy NHE distribution is parameterized by the parameters $\beta > 0$, $\lambda > 0$, and $\theta > 0$.

The Half-Cauchy NHE distribution has CDF:

$$F(x;\beta,\lambda,\theta) = \frac{2}{\pi}\arctan\left\{-\frac{1}{\theta}\left(1-(1+\lambda x)^{\beta}\right)\right\}, \quad x > 0.$$

where β, λ, and θ are the parameters.

The implementation includes the following functions:

- dhc.NHE() — Density function
- phc.NHE() — Distribution function
- qhc.NHE() — Quantile function
- rhc.NHE() — Random generation
- hhc.NHE() — Hazard function

Value

- dhc.NHE: numeric vector of (log-)densities
- phc.NHE: numeric vector of probabilities
- qhc.NHE: numeric vector of quantiles
- rhc.NHE: numeric vector of random variates
- hhc.NHE: numeric vector of hazard values

References

Chaudhary, A. K., & Kumar, V.(2021). Arctan Exponential Extension Distribution with Properties and Applications. *International Journal of Applied Research (IJAR)*, **7(1)**, 432–442. doi:10.22271/allresearch.2021.v7.i1f.8251

Telee, L. B. S., & Kumar, V. (2022). Some properties and applications of half-Cauchy exponential extension distribution. *Int. J. Stat. Appl. Math.*, **7(6)**, 91–101. doi:10.22271/maths.2022.v7.i6b.902

Kumar, V. (2010). Bayesian analysis of exponential extension model. *J. Nat. Acad. Math.*, **24**, 109-128.

Examples

```
x <- seq(1.0, 5, 0.25)
dhc.NHE(x, 2.0, 0.5, 2.5)
phc.NHE(x, 2.0, 0.5, 2.5)
qhc.NHE(0.5, 2.0, 0.5, 2.5)
rhc.NHE(10, 2.0, 0.5, 2.5)
hhc.NHE(x, 2.0, 0.5, 2.5)

# Data
x <- stress66
# ML estimates
params = list(beta=95.2115, lambda=0.0184, theta=118.0656)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = phc.NHE, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qhc.NHE, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dhc.NHE, pfun=phc.NHE, plot=TRUE)
print.gofic(out)
```

headneck44 *Head and Neck Cancer Survival Times*

Description

A dataset containing survival times (in days) of 44 patients with Head and Neck cancer who were treated using radiotherapy. The dataset was originally reported by Efron (1988) in his work on logistic regression, survival analysis, and Kaplan–Meier methods.

Usage

```
headneck44
```

Format

A numeric vector of length 44 containing survival times (in days).

Details

This dataset has been widely used in survival analysis literature, particularly for demonstrating Kaplan–Meier estimation and related nonparametric survival techniques. The patients in the study were treated with radiotherapy, and their survival times were recorded.

Value

An object of class `"numeric"`.

The vector consists of 44 observed survival times (in days), each corresponding to a single patient diagnosed with Head and Neck cancer and treated with radiotherapy. Each value represents the time from treatment initiation to death or last follow-up. The dataset is commonly used as input for illustrating and comparing nonparametric survival analysis methods, including Kaplan–Meier estimation.

References

Efron, B. (1988). Logistic regression, survival analysis and the Kaplan–Meier curve. *Journal of the American Statistical Association*, 83(402), 414–425.

Examples

```
summary(headneck44)

plot(
  headneck44,
  main = "Head and Neck Cancer Survival Times",
  ylab = "Days"
)
```

HLIW *Half-Logistic Inverted Weibull (HLIW) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Half-Logistic Inverted Weibull distribution.

Usage

```
dHL.inv.weib(x, alpha, beta, lambda, log = FALSE)
pHL.inv.weib(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qHL.inv.weib(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rHL.inv.weib(n, alpha, beta, lambda)
hHL.inv.weib(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric shape parameter
beta	positive numeric rate parameter
lambda	positive numeric shape parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The HLIW distribution is parameterized by shape parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Half-Logistic Inverted Weibull (HLIW) distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \frac{1 - \left\{1 - e^{-\alpha x^{-\beta}}\right\}^{\lambda}}{1 + \left\{1 - e^{-\alpha x^{-\beta}}\right\}^{\lambda}}; \quad x > 0.$$

where α, β, and λ are the parameters.

The implementation includes the following functions:

- `dHL.inv.weib()` — Density function
- `pHL.inv.weib()` — Distribution function
- `qHL.inv.weib()` — Quantile function
- `rHL.inv.weib()` — Random generation
- `hHL.inv.weib()` — Hazard function

Value

- dHL.inv.weib: numeric vector of (log-)densities
- pHL.inv.weib: numeric vector of probabilities
- qHL.inv.weib: numeric vector of quantiles
- rHL.inv.weib: numeric vector of random variates
- hHL.inv.weib: numeric vector of hazard values

References

Elgarhy, M., ul Haq, M.A. & Perveen, I. (2019). Type II Half Logistic Exponential Distribution with Applications. *Ann. Data. Sci.*, **6**, 245–257 doi:10.1007/s407450180175y

Chaudhary, A. K., & Kumar, V. (2020). Half Logistic Exponential Extension Distribution with Properties and Applications. *International Journal of Recent Technology and Engineering (IJRTE)*, **8(3)**, 506–512. doi:10.35940/ijrte.C4625.099320

Dhungana, G.P. & Kumar, V.(2022). Half Logistic Inverted Weibull Distribution: Properties and Applications. *J. Stat. Appl. Pro. Lett.*, **9(3)**, 161–178. doi:10.18576/jsapl/090306

Examples

```
x <- seq(0.1, 5, 0.1)
dHL.inv.weib(x, 1.5, 0.8, 2)
pHL.inv.weib(x, 1.5, 0.8, 2)
qHL.inv.weib(0.5, 1.5, 0.8, 2)
rHL.inv.weib(10, 1.5, 0.8, 2)
hHL.inv.weib(x, 1.5, 0.8, 2)

#Data
x <- survtimes
gofic(x,
      params = list(alpha=31.1650, beta=0.4213, lambda=45.5485),
      dfun = dHL.inv.weib, pfun = pHL.inv.weib, plot=TRUE, verbose = TRUE)

pp.plot(x,
        params = list(alpha=31.1650, beta=0.4213, lambda=45.5485),
        pfun = pHL.inv.weib, fit.line=TRUE)

qq.plot(x,
        params = list(alpha=31.1650, beta=0.4213, lambda=45.5485),
        qfun = qHL.inv.weib, fit.line=TRUE)
```

HLNHE *Half-Logistic NHE(Nadarajah-Haghighi Exponential) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Half-Logistic NHE distribution.

Usage

```
dHL.nhe(x, alpha, beta, lambda, log = FALSE)
pHL.nhe(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qHL.nhe(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rHL.nhe(n, alpha, beta, lambda)
hHL.nhe(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Half-Logistic NHE distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Half-Logistic NHE distribution has CDF:

$$F(x;\, \alpha, \beta, \lambda) = \frac{1 - \exp\left[\lambda\left\{1 - (1+\alpha x)^{\beta}\right\}\right]}{1 + \exp\left[\lambda\left\{1 - (1+\alpha x)^{\beta}\right\}\right]}; \quad x > 0.$$

where α, β, and λ are the parameters.

The functions available are listed below:

- `dHL.nhe()` — Density function
- `pHL.nhe()` — Distribution function
- `qHL.nhe()` — Quantile function
- `rHL.nhe()` — Random generation
- `hHL.nhe()` — Hazard function

Value

- `dHL.nhe`: numeric vector of (log-)densities
- `pHL.nhe`: numeric vector of probabilities
- `qHL.nhe`: numeric vector of quantiles
- `rHL.nhe`: numeric vector of random variates
- `hHL.nhe`: numeric vector of hazard values

References

Almarashi, A. M., Elgarhy, M., Elsehetry, M. M., Kibria, B. G., & Algarni, A. (2019). A new extension of exponential distribution with statistical properties and applications. *Journal of Nonlinear Sciences and Applications*, **12**, 135–145.

Chaudhary, A.K., & Kumar, V.(2020). Half Logistic Modified Exponential Distribution:Properties and Applications. *EPRA International Journal of Multidisciplinary Research (IJMR)*, **6(12)**,276–286. doi:10.36713/epra3291

Joshi, R. K., & Kumar, V. (2020). Half Logistic NHE: Properties and Application. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, **8(9)**, 742–753. doi:10.22214/ijraset.2020.31557

Nadarajah, S., & Haghighi, F. (2011). An extension of the exponential distribution. *Statistics*, **45(6)**, 543–558.

Examples

```
x <- seq(0.1, 1, 0.1)
dHL.nhe(x, 1.5, 0.8, 2)
pHL.nhe(x, 1.5, 0.8, 2)
qHL.nhe(0.5, 1.5, 0.8, 2)
rHL.nhe(10, 1.5, 0.8, 2)
hHL.nhe(x, 1.5, 0.8, 2)

#Data
x <- windshield
#ML Estimates
params = list(alpha =0.1649, beta=3.7152, lambda=0.5881)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pHL.nhe, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qHL.nhe, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dHL.nhe, pfun = pHL.nhe, plot=FALSE)
print.gofic(out)
```

`InvEEP` *Inverse Exponentiated Exponential Poisson (IEEP) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Inverse Exponentiated Exponential Poisson distribution.

Usage

```
dinv.expo.exp.pois(x, alpha, beta, lambda, log = FALSE)
pinv.expo.exp.pois(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qinv.expo.exp.pois(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rinv.expo.exp.pois(n, alpha, beta, lambda)
hinv.expo.exp.pois(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Inverse Exponentiated Exponential Poisson distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Inverse Exponentiated Exponential Poisson distribution has CDF:

$$F(x;\, \alpha, \beta, \lambda) = 1 - \frac{1}{(1 - e^{-\lambda})} \left[1 - \exp\left\{-\lambda\Big(1 - e^{-\beta/x}\Big)^{\alpha}\right\}\right] \;; \quad x > 0.$$

where α, β, and λ are the parameters.

The implementation includes the following functions:

- `dinv.expo.exp.pois()` — Density function
- `pinv.expo.exp.pois()` — Distribution function
- `qinv.expo.exp.pois()` — Quantile function
- `rinv.expo.exp.pois()` — Random generation
- `hinv.expo.exp.pois()` — Hazard function

Value

- `dinv.expo.exp.pois`: numeric vector of (log-)densities
- `pinv.expo.exp.pois`: numeric vector of probabilities
- `qinv.expo.exp.pois`: numeric vector of quantiles
- `rinv.expo.exp.pois`: numeric vector of random variates
- `hinv.expo.exp.pois`: numeric vector of hazard values

References

Ristic, M.M., & Nadarajah, S.(2014). A New Lifetime Distribution. *Journal of Statistical Computation and Simulation*, **84(1)**, 135–150. doi:10.1080/00949655.2012.697163

Telee, L. B. S., & Kumar, V. (2023). Inverse Exponentiated Exponential Poisson Distribution with Theory and Applications. *International Journal of Engineering Science Technologies*, **7(5)**, 17–36. doi:10.29121/IJOEST.v7.i5.2023.535

Examples

```
x <- seq(0.1, 1, 0.1)
dinv.expo.exp.pois(x, 1.5, 0.8, 2)
pinv.expo.exp.pois(x, 1.5, 0.8, 2)
qinv.expo.exp.pois(0.5, 1.5, 0.8, 2)
rinv.expo.exp.pois(10, 1.5, 0.8, 2)
hinv.expo.exp.pois(x, 1.5, 0.8, 2)

#Data
x <- conductors
#ML Estimates
params = list(alpha =40.5895, beta=22.7519, lambda=2.9979)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pinv.expo.exp.pois, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qinv.expo.exp.pois, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
res <- gofic(x, params = params, dfun = dinv.expo.exp.pois,
             pfun = pinv.expo.exp.pois, plot=FALSE)
print.gofic(res)
```

InvExpPower *Inverse Exponential Power Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Inverse Exponential Power distribution.

Usage

```
dinv.exp.power(x, alpha, lambda, log = FALSE)
pinv.exp.power(q, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
qinv.exp.power(p, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
rinv.exp.power(n, alpha, lambda)
hinv.exp.power(x, alpha, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Inverse Exponential Power distribution is parameterized by the parameters $\alpha > 0$ and $\lambda > 0$.

The Inverse Exponential Power distribution has CDF:

$$F(x; \alpha, \lambda) = \quad \exp\left\{1 - \exp\left(\frac{\lambda}{x}\right)^{\alpha}\right\}; \quad x > 0.$$

where α and λ are the parameters.

The implementation includes the following functions:

- `dinv.exp.power()` — Density function
- `pinv.exp.power()` — Distribution function
- `qinv.exp.power()` — Quantile function
- `rinv.exp.power()` — Random generation
- `hinv.exp.power()` — Hazard function

Value

- `dinv.exp.power`: numeric vector of (log-)densities
- `pinv.exp.power`: numeric vector of probabilities
- `qinv.exp.power`: numeric vector of quantiles
- `rinv.exp.power`: numeric vector of random variates
- `hinv.exp.power`: numeric vector of hazard values

References

Chaudhary, A.K., Sapkota,L.P. & Kumar, V.(2023). Inverse Exponential Power distribution: Theory and Applications. *International Journal of Mathematics, Statistics and Operations Research*, **3(1)**, 175–185.

Examples

```
x <- seq(1.0, 5.0, 0.2)
dinv.exp.power(x, 2.5, 0.5)
pinv.exp.power(x, 2.5, 0.5)
qinv.exp.power(0.5, 2.5, 0.5)
rinv.exp.power(10, 2.5, 0.5)
hinv.exp.power(x, 2.5, 0.5)

# Data
x <- relief
# ML estimates
params = list(alpha=2.8286, lambda=1.3346)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pinv.exp.power, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qinv.exp.power, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dinv.exp.power, pfun=pinv.exp.power, plot=FALSE)
print.gofic(out)
```

InvGenGPZ *Inverse Generalized Gompertz Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Inverse Generalized Gompertz distribution.

Usage

```
dinv.gen.gpz(x, alpha, lambda, theta, log = FALSE)
pinv.gen.gpz(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qinv.gen.gpz(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rinv.gen.gpz(n, alpha, lambda, theta)
hinv.gen.gpz(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density

lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Inverse Generalized Gompertz distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Inverse Generalized Gompertz distribution has CDF:

$$F(x; \alpha, \lambda, \theta) = 1 - \left[1 - \exp\left(\frac{\lambda}{\alpha}\left(1 - \exp(\alpha/x)\right)\right)\right]^{\theta}, \quad x > 0.$$

where α, λ, and θ are the parameters.

The implementation includes the following functions:

- dinv.gen.gpz() — Density function
- pinv.gen.gpz() — Distribution function
- qinv.gen.gpz() — Quantile function
- rinv.gen.gpz() — Random generation
- hinv.gen.gpz() — Hazard function

Value

- dinv.gen.gpz: numeric vector of (log-)densities
- pinv.gen.gpz: numeric vector of probabilities
- qinv.gen.gpz: numeric vector of quantiles
- rinv.gen.gpz: numeric vector of random variates
- hinv.gen.gpz: numeric vector of hazard values

References

Chaudhary, A.K., & Kumar, V. (2017). Inverse Generalized Gompertz Distribution with Properties and Applications. *Journal of National Academy of Mathematics*, **31**, 1–15.

Examples

```
x <- seq(2, 5, 0.25)
dinv.gen.gpz(x, 1.5, 2.5, 5.0)
pinv.gen.gpz(x, 1.5, 2.5, 5.0)
qinv.gen.gpz(0.5, 1.5, 2.5, 5.0)
rinv.gen.gpz(10, 1.5, 2.5, 5.0)
hinv.gen.gpz(x, 1.5, 2.5, 5.0)

# Data
```

```
x <- fibers63
# ML estimates
params = list(alpha=3.4106, lambda=5.4685, theta=20.9199)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pinv.gen.gpz, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qinv.gen.gpz, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dinv.gen.gpz, pfun=pinv.gen.gpz, plot=TRUE)
print.gofic(out)
```

`InvPham` *Inverse Pham Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Inverse Pham distribution.

Usage

```
dinv.pham(x, beta, delta, log = FALSE)
pinv.pham(q, beta, delta, lower.tail = TRUE, log.p = FALSE)
qinv.pham(p, beta, delta, lower.tail = TRUE, log.p = FALSE)
rinv.pham(n, beta, delta)
hinv.pham(x, beta, delta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
beta	positive numeric parameter
delta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities ($0 < p < 1$)
n	number of observations (integer > 0)

Details

The Inverse Pham distribution is parameterized by the parameters $\beta > 0$, and $\delta > 0$.

The Inverse Pham distribution has CDF:

$$F(x; \beta, \delta) = \exp\left(1 - \delta^{x^{-\beta}}\right) \quad ; \; x > 0.$$

whereβ and δ are the parameters.

The following functions are included:

- `dinv.pham()` — Density function
- `pinv.pham()` — Distribution function
- `qinv.pham()` — Quantile function
- `rinv.pham()` — Random generation
- `hinv.pham()` — Hazard function

Value

- `dinv.pham`: numeric vector of (log-)densities
- `pinv.pham`: numeric vector of probabilities
- `qinv.pham`: numeric vector of quantiles
- `rinv.pham`: numeric vector of random variates
- `hinv.pham`: numeric vector of hazard values

References

Elbatal, M., Araibi, M.I.A., Ocloo, S.K., Almetwally, E.M., Sapkota, L.P., & Gemeay, A.M. (2025). Classical and Bayesian Methodology for a New Inverse Statistical Model. *Engineering Reports*, **7(8)**, 1–33. doi:10.1002/eng2.70323

Srivastava, A.K., & Kumar, V. (2011). Analysis of Pham (Loglog) Reliability Model Using Bayesian Approach. *Computer Science Journal*, **1(2)**, 79–100.

Pham, H. (2002). A Vtub-Shaped Hazard Rate Function With Applications to System Safety. *International Journal of Reliability and Applications*, **3(1)**, 1–16.

Examples

```
x <- seq(1, 10, 0.5)
dinv.pham(x, 0.5, 1.5)
pinv.pham(x, 0.5, 1.5)
qinv.pham(0.5, 0.5, 1.5)
rinv.pham(10, 0.5, 1.5)
hinv.pham(x, 0.5, 1.5)

# Data
x <- relief
# ML estimates
params = list(beta=2.8287, delta=9.6044)
```

```
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pinv.pham, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qinv.pham, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dinv.pham, pfun=pinv.pham, plot=FALSE)
print.gofic(out)
```

`InvPowerCauchy` *Inverse Power Cauchy Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Inverse Power Cauchy distribution.

Usage

```
dinv.pow.cauchy(x, alpha, lambda, log = FALSE)
pinv.pow.cauchy(q, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
qinv.pow.cauchy(p, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
rinv.pow.cauchy(n, alpha, lambda)
hinv.pow.cauchy(x, alpha, lambda)
```

Arguments

`x, q`	numeric vector of quantiles (x, q)
`alpha`	positive numeric parameter
`lambda`	positive numeric parameter
`log`	logical; if TRUE, returns log-density
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
`log.p`	logical; if TRUE, probabilities are given as log(p)
`p`	numeric vector of probabilities (0 < p < 1)
`n`	number of observations (integer > 0)

Details

The Inverse Power Cauchy distribution is parameterized by the parameters $\alpha > 0$ and $\lambda > 0$.

The Inverse Power Cauchy distribution has CDF:

$$F(x;\alpha,\lambda) = \quad 1 - 2\pi^{-1}\tan^{-1}\left[\left(\frac{\lambda}{x}\right)^{\alpha}\right]\ ; \quad x > 0.$$

where α and λ are the parameters.

The following functions are included:

- `dinv.pow.cauchy()` — Density function
- `pinv.pow.cauchy()` — Distribution function
- `qinv.pow.cauchy()` — Quantile function
- `rinv.pow.cauchy()` — Random generation
- `hinv.pow.cauchy()` — Hazard function

Value

- `dinv.pow.cauchy`: numeric vector of (log-)densities
- `pinv.pow.cauchy`: numeric vector of probabilities
- `qinv.pow.cauchy`: numeric vector of quantiles
- `rinv.pow.cauchy`: numeric vector of random variates
- `hinv.pow.cauchy`: numeric vector of hazard values

References

Sapkota L. P., & Kumar V. (2023). Applications and Some Characteristics of Inverse Power Cauchy Distribution. *Reliability: Theory & Applications*. **18, 1(72)**, 301–315. doi:10.24412/19322321-2023172301315

Chaudhary, A.K., Sapkota, L.P., & Kumar, V. (2020). Truncated Cauchy Power–Inverse Exponential distribution: Theory and Applications. *IOSR Journal of Mathematics (IOSR-JM)*, **16(4)**, Ser.IV, 12–23.

Examples

```
x <- seq(0.1, 10, 0.2)
dinv.pow.cauchy(x, 2.0, 5.0)
pinv.pow.cauchy(x, 2.0, 5.0)
qinv.pow.cauchy(0.5, 2.0, 5.0)
rinv.pow.cauchy(10, 2.0, 5.0)
hinv.pow.cauchy(x, 2.0, 5.0)

# Data
x <- headneck44
# ML estimates
params = list(alpha=1.4271, lambda=123.5294)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pinv.pow.cauchy, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qinv.pow.cauchy, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
res <- gofic(x, params = params,
             dfun = dinv.pow.cauchy, pfun=pinv.pow.cauchy, plot=FALSE)
```

```
print.gofic(res)
```

InvSGZ *Inverted Shifted Gompertz (ISG) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Inverted Shifted Gompertz distribution.

Usage

```
dinv.sgz(x, alpha, theta, log = FALSE)
pinv.sgz(q, alpha, theta, lower.tail = TRUE, log.p = FALSE)
qinv.sgz(p, alpha, theta, lower.tail = TRUE, log.p = FALSE)
rinv.sgz(n, alpha, theta)
hinv.sgz(x, alpha, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Inverted Shifted Gompertz distribution is parameterized by the parameters $\alpha > 0$, and $\theta > 0$.

The Inverted Shifted Gompertz distribution has CDF:

$$F(x;\alpha,\theta) = 1 - \left(1 - e^{-\theta/x}\right) \exp\left(-\alpha e^{-\theta/x}\right) \quad ; \; x > 0.$$

whereα and θ are the parameters.

The following functions are included:

- `dinv.sgz()` — Density function
- `pinv.sgz()` — Distribution function
- `qinv.sgz()` — Quantile function
- `rinv.sgz()` — Random generation
- `hinv.sgz()` — Hazard function

Value

- `dinv.sgz`: numeric vector of (log-)densities
- `pinv.sgz`: numeric vector of probabilities
- `qinv.sgz`: numeric vector of quantiles
- `rinv.sgz`: numeric vector of random variates
- `hinv.sgz`: numeric vector of hazard values

References

Chaudhary, A.K., Sapkota, L.P., & Kumar, V. (2020). Inverted Shifted Gompertz Distribution with Theory and Applications. *Pravaha*, **26(1)**, 1–10. doi:10.3126/pravaha.v26i1.41645

Jimenez T.F. (2014). Estimation of the Parameters of the Shifted Gompertz Distribution, Using Least Squares, Maximum Likelihood and Moments Methods. *Journal of Computational and Applied Mathematics*, **255(1)** 867–877.

Examples

```
x <- seq(1.0, 5, 0.25)
dinv.sgz(x, 25, 10)
pinv.sgz(x, 25, 10)
qinv.sgz(0.5, 25, 10)
rinv.sgz(10, 25, 10)
hinv.sgz(x, 25, 10)

# Data
x <- fibers65
# ML estimates
params = list(alpha=215.8181, theta=12.7678)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pinv.sgz, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qinv.sgz, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dinv.sgz, pfun=pinv.sgz, plot=FALSE)
print.gofic(out)
```

InvUBD *Inverse Upside Down Bathtub-shaped Hazard Function Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Inverse Upside Down Bathtub-shaped Hazard Function distribution.

Usage

```
dinv.ubd(x, alpha, beta, lambda, log = FALSE)
pinv.ubd(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qinv.ubd(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rinv.ubd(n, alpha, beta, lambda)
hinv.ubd(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Inverse Upside Down Bathtub-shaped Hazard Function distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Inverse Upside Down Bathtub-shaped Hazard Function distribution has CDF:

$$F(x;\, \alpha, \beta, \lambda) = \exp\left[1 - \left(1 + \lambda x^{-\beta}\right)^{\alpha}\right], \quad x > 0.$$

where α, β, and λ are the parameters.

The functions available are listed below:

- `dinv.ubd()` — Density function
- `pinv.ubd()` — Distribution function
- `qinv.ubd()` — Quantile function
- `rinv.ubd()` — Random generation
- `hinv.ubd()` — Hazard function

Value

- `dinv.ubd`: numeric vector of (log-)densities
- `pinv.ubd`: numeric vector of probabilities
- `qinv.ubd`: numeric vector of quantiles
- `rinv.ubd`: numeric vector of random variates
- `hinv.ubd`: numeric vector of hazard values

References

Dimitrakopoulou, T., Adamidis, K., & Loukas, S.(2007). A liftime distribution with an upside down bathtub-shaped hazard function, *IEEE Trans. on Reliab.*, **56(2)**, 308–311.

Joshi, R.K., & Kumar, V. (2018). Inverse Upside Down Bathtub-Shaped Hazard Function distribution: Theory and Applications. *Journal of National Academy of Mathematics*, **32**, 6–20.

Examples

```
x <- seq(0.1, 1, 0.1)
dinv.ubd(x, 1.5, 0.8, 2)
pinv.ubd(x, 1.5, 0.8, 2)
qinv.ubd(0.5, 1.5, 0.8, 2)
rinv.ubd(10, 1.5, 0.8, 2)
hinv.ubd(x, 1.5, 0.8, 2)

#Data
x <- rainfall
#ML Estimates
params = list(alpha =0.1804, beta=4.3216, lambda=85.13)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pinv.ubd, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qinv.ubd, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dinv.ubd, pfun = pinv.ubd, plot=FALSE)
print.gofic(out)
```

`LindleyChen` *Lindley-Chen Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Lindley-Chen distribution.

Usage

```
dlindley.chen(x, alpha, lambda, theta, log = FALSE)
plindley.chen(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qlindley.chen(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rlindley.chen(n, alpha, lambda, theta)
hlindley.chen(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Lindley-Chen distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Lindley-Chen distribution has CDF:

$$F(x; \alpha, \lambda, \theta) = 1 - \left[1 - \lambda \left(\frac{\theta}{1+\theta}\right)\left(1 - e^{x^\alpha}\right)\right] \exp\left\{\lambda\theta\left(1 - e^{x^\alpha}\right)\right\}, \quad x > 0.$$

where α, λ, and θ are the parameters.

The functions available are listed below:

- `dlindley.chen()` — Density function
- `plindley.chen()` — Distribution function
- `qlindley.chen()` — Quantile function
- `rlindley.chen()` — Random generation
- `hlindley.chen()` — Hazard function

Value

- `dlindley.chen`: numeric vector of (log-)densities
- `plindley.chen`: numeric vector of probabilities
- `qlindley.chen`: numeric vector of quantiles
- `rlindley.chen`: numeric vector of random variates
- `hlindley.chen`: numeric vector of hazard values

References

Bhati, D., Malik, M. A., & Vaman, H. J. (2015). Lindley–Exponential distribution: properties and applications. *Metron*, **73(3)**, 335–357.

Joshi, R. K., & Kumar, V. (2020). Lindley-Chen Distribution with Applications. *International Journal of Engineering, Science & Mathematics (IJESM)*, **9(10)**, 12–22.

Examples

```
x <- seq(1.0, 3.0, 0.25)
dlindley.chen(x, 0.5, 2, 0.5)
plindley.chen(x, 0.5, 2, 0.5)
qlindley.chen(0.5, 0.5, 2, 0.5)
rlindley.chen(10, 0.5, 2, 0.5)
hlindley.chen(x, 0.5, 2, 0.5)

# Data
x <- fibers65
# ML estimates
params = list(alpha=1.26813, lambda=28.96389, theta=0.00355)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plindley.chen, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlindley.chen, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlindley.chen, pfun=plindley.chen, plot=FALSE)
print.gofic(out)
```

LindleyExpPower *Lindley-Exponential Power Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Lindley-Exponential Power distribution.

Usage

```
dlind.exp.pow(x, alpha, lambda, theta, log = FALSE)
plind.exp.pow(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qlind.exp.pow(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rlind.exp.pow(n, alpha, lambda, theta)
hlind.exp.pow(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density

`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
`log.p`	logical; if TRUE, probabilities are given as log(p)
`p`	numeric vector of probabilities (0 < p < 1)
`n`	number of observations (integer > 0)

Details

The Lindley-Exponential Power distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Lindley-Exponential Power distribution has CDF:

$$F(x; \alpha, \lambda, \theta) = \quad 1 - \left[1 - \left(\frac{\theta}{1+\theta}\right)\left(1 - e^{(\lambda x)^{\alpha}}\right)\right] \exp\left[\theta\left(1 - e^{(\lambda x)^{\alpha}}\right)\right] \quad ; \; x > 0.$$

where α, λ, and θ are the parameters.

The following functions are included:

- `dlind.exp.pow()` — Density function
- `plind.exp.pow()` — Distribution function
- `qlind.exp.pow()` — Quantile function
- `rlind.exp.pow()` — Random generation
- `hlind.exp.pow()` — Hazard function

Value

- `dlind.exp.pow`: numeric vector of (log-)densities
- `plind.exp.pow`: numeric vector of probabilities
- `qlind.exp.pow`: numeric vector of quantiles
- `rlind.exp.pow`: numeric vector of random variates
- `hlind.exp.pow`: numeric vector of hazard values

References

Joshi, R. K., & Kumar, V. (2020). Lindley exponential power distribution with Properties and Applications. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, **8(10)**, 22–30. doi:10.22214/ijraset.2020.31743

Joshi, R.K., & Kumar, V. (2016). Exponentiated Power Lindley Distribution : A Bayes Study using MCMC Approach. *J. Nat. Acad. Math.*, **30**, 80–102.

Examples

```
x <- seq(1.0, 5, 0.25)
dlind.exp.pow(x, 0.5, 0.2, 1.5)
plind.exp.pow(x, 0.5, 0.2, 1.5)
qlind.exp.pow(0.5, 0.5, 0.2, 1.5)
rlind.exp.pow(10, 0.5, 0.2, 1.5)
hlind.exp.pow(x, 0.5, 0.2, 1.5)

# Data
x <- windshield
# ML estimates
params = list(alpha=0.97722, lambda=0.39461, theta=0.96124)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plind.exp.pow, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlind.exp.pow, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlind.exp.pow, pfun=plind.exp.pow, plot=FALSE)
print.gofic(out)
```

LindleyGIE *Lindley Generalized Inverted Exponential(LGIE) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the LGIE distribution.

Usage

```
dlind.ginv.exp(x, alpha, lambda, theta, log = FALSE)
plind.ginv.exp(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qlind.ginv.exp(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rlind.ginv.exp(n, alpha, lambda, theta)
hlind.ginv.exp(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density

lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The LGIE distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The LGIE distribution has CDF:

$$F(x;\alpha,\lambda,\theta) = 1 - \left(1 - e^{-\lambda/x}\right)^{\alpha\theta}\left[1 - \left(\frac{\theta}{\theta+1}\right)\ln\left(1 - e^{-\lambda/x}\right)^{\alpha}\right] \quad ; \; x > 0.$$

where α, λ, and θ are the parameters.

The following functions are included:

- `dlind.ginv.exp()` — Density function
- `plind.ginv.exp()` — Distribution function
- `qlind.ginv.exp()` — Quantile function
- `rlind.ginv.exp()` — Random generation
- `hlind.ginv.exp()` — Hazard function

Value

- `dlind.ginv.exp`: numeric vector of (log-)densities
- `plind.ginv.exp`: numeric vector of probabilities
- `qlind.ginv.exp`: numeric vector of quantiles
- `rlind.ginv.exp`: numeric vector of random variates
- `hlind.ginv.exp`: numeric vector of hazard values

References

Telee, L. B. S., & Kumar, V. (2021). Lindley Generalized Inverted Exponential Distribution: Model and Applications. *Pravaha*, **27(1)**, 61–72. doi:10.3126/pravaha.v27i1.50616

Yadav, R.S., & Kumar, V. (2020). Arctan Generalized Inverted Exponential Distribution. *J. Nat. Acad. Math.*, **34**, 71–92.

Examples

```
x <- seq(5, 10, 0.2)
dlind.ginv.exp(x, 5.0, 1.5, 0.5)
plind.ginv.exp(x, 5.0, 1.5, 0.5)
qlind.ginv.exp(0.5, 5.0, 1.5, 0.5)
rlind.ginv.exp(10, 5.0, 1.5, 0.5)
hlind.ginv.exp(x, 5.0, 1.5, 0.5)
```

```
# Data
x <- conductors
# ML estimates
params = list(alpha=97.0105, lambda=29.9324, theta=0.9028)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plind.ginv.exp, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlind.ginv.exp, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlind.ginv.exp, pfun=plind.ginv.exp, plot=FALSE)
print.gofic(out)
```

`LindleyGompertz` *Lindley-Gompertz Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Lindley-Gompertz distribution.

Usage

```
dlindley.gpz(x, alpha, lambda, theta, log = FALSE)
plindley.gpz(q, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qlindley.gpz(p, alpha, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rlindley.gpz(n, alpha, lambda, theta)
hlindley.gpz(x, alpha, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Lindley-Gompertz distribution is parameterized by the parameters $\alpha > 0$, $\lambda > 0$, and $\theta > 0$.

The Lindley-Gompertz distribution has CDF:

$$F(x;\alpha,\lambda,\theta) = \left(1 - \exp\left\{\frac{\lambda}{\alpha}\left(1-\exp(\alpha x)\right)\right\}\right)^{\theta}\left[1 - \frac{\theta}{1+\theta}\log\left\{1-\exp\left(\frac{\lambda}{\alpha}\left(1-\exp(\alpha x)\right)\right)\right\}\right], \quad x > 0.$$

where α, λ, and θ are the parameters.

Value

- `dlindley.gpz`: numeric vector of (log-)densities
- `plindley.gpz`: numeric vector of probabilities
- `qlindley.gpz`: numeric vector of quantiles
- `rlindley.gpz`: numeric vector of random variates
- `hlindley.gpz`: numeric vector of hazard values

References

Joshi, R. K., & Kumar, V. (2020). Lindley Gompertz distribution with properties and application. *International Journal of Statistics and Applied Mathematics*, **5(6)**, 28–37. doi:10.22271/maths.2020.v5.i6a.610

Examples

```
x <- seq(1, 10, 0.5)
dlindley.gpz(x, 0.1, 0.5, 1.5)
plindley.gpz(x, 0.1, 0.5, 1.5)
qlindley.gpz(0.5, 0.1, 0.5, 1.5)
rlindley.gpz(10, 0.1, 0.5, 1.5)
hlindley.gpz(x, 0.1, 0.5, 1.5)

# Data
x <- conductors
# ML estimates
params = list(alpha=0.1765, lambda=0.2051, theta=11.4574)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plindley.gpz, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlindley.gpz, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlindley.gpz, pfun=plindley.gpz, plot=FALSE)
print.gofic(out)
```

LindleyHC *Lindley Half-Cauchy(LHC) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Lindley Half-Cauchy distribution.

Usage

```
dlindley.HC(x, lambda, theta, log = FALSE)
plindley.HC(q, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qlindley.HC(p, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rlindley.HC(n, lambda, theta)
hlindley.HC(x, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Lindley Half-Cauchy distribution is parameterized by the parameters $\lambda > 0$, and $\theta > 0$.

The Lindley Half-Cauchy distribution has CDF:

$$F(x;\lambda,\theta) = 1 - \left\{1 - \frac{2}{\pi}\tan^{-1}\left(\frac{x}{\lambda}\right)\right\}^{\theta}\left\{1 - \left(\frac{\theta}{1+\theta}\right)\ln\left[1 - \frac{2}{\pi}\tan^{-1}\left(\frac{x}{\lambda}\right)\right]\right\}\ ;\ x > 0.$$

whereλ and θ are the parameters.

Value

- dlindley.HC: numeric vector of (log-)densities
- plindley.HC: numeric vector of probabilities
- qlindley.HC: numeric vector of quantiles
- rlindley.HC: numeric vector of random variates
- hlindley.HC: numeric vector of hazard values

References

Chaudhary, A.K. & Kumar, V. (2020). Lindley Half Cauchy Distribution: Properties and Applications. *International Journal For Research in Applied Science & Engineering Technology (IJRASET)*, **8(9)**, 1232–1242. doi:10.22214/ijraset.2020.31745

Examples

```
x <- seq(1, 10, 0.5)
dlindley.HC(x, 0.5, 1.5)
plindley.HC(x, 0.5, 1.5)
qlindley.HC(0.5, 0.5, 1.5)
rlindley.HC(10, 0.5, 1.5)
hlindley.HC(x, 0.5, 1.5)

# Data
x <- reactorpump
# ML estimates
params = list(lambda=0.5479, theta=1.2766)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plindley.HC, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlindley.HC, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlindley.HC, pfun=plindley.HC, plot=FALSE)
print.gofic(out)
```

LindleyInvExp *Lindley Inverse Exponential Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Lindley Inverse Exponential distribution.

Usage

```
dlindley.inv.exp(x, lambda, theta, log = FALSE)
plindley.inv.exp(q, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qlindley.inv.exp(p, lambda, theta, lower.tail = TRUE, log.p = FALSE)
rlindley.inv.exp(n, lambda, theta)
hlindley.inv.exp(x, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Lindley Inverse Exponential distribution is parameterized by the parameters $\lambda > 0$, and $\theta > 0$.

The Lindley Inverse Exponential distribution has CDF:

$$F(x;\lambda,\theta) = 1 - \left(1 - e^{-\lambda/x}\right)^{\theta} \left\{1 - \left(\frac{\theta}{1+\theta}\right) \ln\left(1 - e^{-\lambda/x}\right)\right\} \quad ; \; x > 0.$$

whereλ and θ are the parameters.

The following functions are included:

- `dlindley.inv.exp()` — Density function
- `plindley.inv.exp()` — Distribution function
- `qlindley.inv.exp()` — Quantile function
- `rlindley.inv.exp()` — Random generation
- `hlindley.inv.exp()` — Hazard function

Value

- `dlindley.inv.exp`: numeric vector of (log-)densities
- `plindley.inv.exp`: numeric vector of probabilities
- `qlindley.inv.exp`: numeric vector of quantiles
- `rlindley.inv.exp`: numeric vector of random variates
- `hlindley.inv.exp`: numeric vector of hazard values

References

Chaudhary,A.K., & Kumar, V. (2020). Lindley Inverse Exponential Distribution With Properties and Applications. *Bulletin of Mathematics and Statistics Research (BOMSR)*, **8(4)**, 1–13.

Examples

```
x <- seq(5, 10, 0.5)
dlindley.inv.exp(x, 1.5, 5.0)
plindley.inv.exp(x, 1.5, 5.0)
qlindley.inv.exp(0.5, 1.5, 5.0)
rlindley.inv.exp(10, 1.5, 5.0)
hlindley.inv.exp(x, 1.5, 5.0)

# Data
x <- conductors
# ML estimates
params = list(lambda=33.8992, theta=96.0743)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plindley.inv.exp, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlindley.inv.exp, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlindley.inv.exp, pfun=plindley.inv.exp, plot=FALSE)
print.gofic(out)
```

`LindleyInvWeibull` *Lindley-Inverse Weibull Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Lindley-Inverse Weibull distribution.

Usage

```
dlindley.inv.weib(x, alpha, beta, theta, log = FALSE)
plindley.inv.weib(q, alpha, beta, theta, lower.tail = TRUE, log.p = FALSE)
qlindley.inv.weib(p, alpha, beta, theta, lower.tail = TRUE, log.p = FALSE)
rlindley.inv.weib(n, alpha, beta, theta)
hlindley.inv.weib(x, alpha, beta, theta)
```

Arguments

`x, q`	numeric vector of quantiles (x, q)
`alpha`	positive numeric parameter
`beta`	positive numeric parameter
`theta`	positive numeric parameter
`log`	logical; if TRUE, returns log-density

lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Lindley-Inverse Weibull distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\theta > 0$.

The Lindley-Inverse Weibull distribution has CDF:

$$F(x; \alpha, \beta, \theta) = \quad 1 - \left(1 - e^{-\alpha x^{-\beta}}\right)^{\theta} \left\{1 - \left(\frac{\theta}{\theta + 1}\right) \ln\left(1 - e^{-\alpha x^{-\beta}}\right)\right\}; \quad x > 0.$$

where α, β, and θ are the parameters.

The functions available are listed below:

- `dlindley.inv.weib()` — Density function
- `plindley.inv.weib()` — Distribution function
- `qlindley.inv.weib()` — Quantile function
- `rlindley.inv.weib()` — Random generation
- `hlindley.inv.weib()` — Hazard function

Value

- `dlindley.inv.weib`: numeric vector of (log-)densities
- `plindley.inv.weib`: numeric vector of probabilities
- `qlindley.inv.weib`: numeric vector of quantiles
- `rlindley.inv.weib`: numeric vector of random variates
- `hlindley.inv.weib`: numeric vector of hazard values

References

Joshi, R. K., & Kumar, V. (2020). Lindley inverse Weibull distribution: Theory and Applications. *Bull. Math. & Stat. Res.*, **8(3)**, 32–46.

Examples

```
x <- seq(0.1, 1, 0.1)
dlindley.inv.weib(x, 1.5, 2.0, 0.5)
plindley.inv.weib(x, 1.5, 2.0, 0.5)
qlindley.inv.weib(0.5, 2.0, 5.0, 0.1)
rlindley.inv.weib(10, 1.5, 2.0, 0.5)
hlindley.inv.weib(x, 1.5, 2.0, 0.5)

# Data
x <- waiting
```

```
# ML estimates
params = list(alpha=9.3340, beta=0.3010, theta=104.4248)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plindley.inv.weib, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlindley.inv.weib, fit.line=FALSE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlindley.inv.weib, pfun=plindley.inv.weib, plot=FALSE)
print.gofic(out)
```

`LindleyRayleigh` *Lindley-Rayleigh Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Lindley-Rayleigh distribution.

Usage

```
dlindley.rlh(x, alpha, theta, log = FALSE)
plindley.rlh(q, alpha, theta, lower.tail = TRUE, log.p = FALSE)
qlindley.rlh(p, alpha, theta, lower.tail = TRUE, log.p = FALSE)
rlindley.rlh(n, alpha, theta)
hlindley.rlh(x, alpha, theta)
```

Arguments

`x, q`	numeric vector of quantiles (x, q)
`alpha`	positive numeric parameter
`theta`	positive numeric parameter
`log`	logical; if TRUE, returns log-density
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
`log.p`	logical; if TRUE, probabilities are given as log(p)
`p`	numeric vector of probabilities (0 < p < 1)
`n`	number of observations (integer > 0)

Details

The Lindley-Rayleigh distribution is parameterized by the parameters $\alpha > 0$, and $\theta > 0$.

The Lindley-Rayleigh distribution has CDF:

$$F(x;\alpha,\theta) = \left[1 - e^{-\alpha x^2}\right]^{\theta} \left\{1 - \left(\frac{\theta}{1+\theta}\right) \ln\left(1 - e^{-\alpha x^2}\right)\right\} \quad ; \; x > 0.$$

whereα and θ are the parameters.

Included functions are:

- `dlindley.rlh()` — Density function
- `plindley.rlh()` — Distribution function
- `qlindley.rlh()` — Quantile function
- `rlindley.rlh()` — Random generation
- `hlindley.rlh()` — Hazard function

Value

- `dlindley.rlh`: numeric vector of (log-)densities
- `plindley.rlh`: numeric vector of probabilities
- `qlindley.rlh`: numeric vector of quantiles
- `rlindley.rlh`: numeric vector of random variates
- `hlindley.rlh`: numeric vector of hazard values

References

Joshi, R. K., & Kumar, V. (2020). New Lindley-Rayleigh Distribution with Statistical properties and Applications. *International Journal of Mathematics Trends and Technology (IJMTT)*, **66(9)**, 197–208. doi:10.14445/22315373/IJMTTV66I9P523

Examples

```
x <- seq(0.5, 5, 0.25)
dlindley.rlh(x, 0.25, 1.5)
plindley.rlh(x, 0.25, 1.5)
qlindley.rlh(0.75, 0.25, 1.5)
rlindley.rlh(10, 0.25, 1.5)
hlindley.rlh(x, 0.25, 1.5)

# Data
x <- rainfall
# ML estimates
params = list(alpha=0.2170, theta=1.2107)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plindley.rlh, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
```

```
qq.plot(x, params = params, qfun = qlindley.rlh, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlindley.rlh, pfun=plindley.rlh, plot=FALSE)
print.gofic(out)
```

LogisChen *Logistic-Chen Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic-Chen distribution.

Usage

```
dlogis.chen(x, alpha, beta, lambda, log = FALSE)
plogis.chen(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.chen(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.chen(n, alpha, beta, lambda)
hlogis.chen(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Logistic-Chen distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic-Chen distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad 1 - \frac{1}{1 + \left[\exp\left\{\lambda\left(e^{x^\beta} - 1\right)\right\} - 1\right]^\alpha}; \quad x \geq 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dlogis.chen()` — Density function
- `plogis.chen()` — Distribution function
- `qlogis.chen()` — Quantile function
- `rlogis.chen()` — Random generation
- `hlogis.chen()` — Hazard function

Value

- `dlogis.chen`: numeric vector of (log-)densities
- `plogis.chen`: numeric vector of probabilities
- `qlogis.chen`: numeric vector of quantiles
- `rlogis.chen`: numeric vector of random variates
- `hlogis.chen`: numeric vector of hazard values

References

Joshi, R.K., & Kumar, V. (2021). Logistic Chen Distribution with Properties and Applications. *International Journal of Mathematics Trends and Technology (IJMTT)*, **67(1)**, 141–151. doi:10.14445/22315373/IJMTTV67I1P519

Examples

```
x <- seq(0.1, 2.0, 0.1)
dlogis.chen(x, 1.5, 1.5, 0.1)
plogis.chen(x, 1.5, 1.5, 0.1)
qlogis.chen(0.5, 1.5, 1.5, 0.1)
rlogis.chen(10, 2.0, 5.0, 0.1)
hlogis.chen(x, 1.5, 1.5, 0.1)

# Data
x <- bladder
# ML estimates
params = list(alpha=4.46424, beta=0.15506, lambda=0.24904)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.chen, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.chen, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.chen, pfun=plogis.chen, plot=FALSE)
print.gofic(out)
```

`LogisExpExt` *Logistic-Exponential Extension Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic-Exponential Extension distribution.

Usage

```
dlogis.exp.ext(x, alpha, beta, lambda, log = FALSE)
plogis.exp.ext(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.exp.ext(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.exp.ext(n, alpha, beta, lambda)
hlogis.exp.ext(x, alpha, beta, lambda)
```

Arguments

`x, q`	numeric vector of quantiles (x, q)
`alpha`	positive numeric parameter
`beta`	positive numeric parameter
`lambda`	positive numeric parameter
`log`	logical; if TRUE, returns log-density
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
`log.p`	logical; if TRUE, probabilities are given as log(p)
`p`	numeric vector of probabilities (0 < p < 1)
`n`	number of observations (integer > 0)

Details

The Logistic-Exponential Extension distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic-Exponential Extension distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad 1 - \frac{1}{1 + \left[\exp\left\{-\lambda x e^{-\beta/x}\right\} - 1\right]^{\alpha}} \; ; \; x > 0.$$

where α, β, and λ are the parameters.

The implementation includes the following functions:

- `dlogis.exp.ext()` — Density function
- `plogis.exp.ext()` — Distribution function
- `qlogis.exp.ext()` — Quantile function
- `rlogis.exp.ext()` — Random generation
- `hlogis.exp.ext()` — Hazard function

Value

- `dlogis.exp.ext`: numeric vector of (log-)densities
- `plogis.exp.ext`: numeric vector of probabilities
- `qlogis.exp.ext`: numeric vector of quantiles
- `rlogis.exp.ext`: numeric vector of random variates
- `hlogis.exp.ext`: numeric vector of hazard values

References

Chaudhary,A.K., & Kumar, V.(2020). A Study on Properties and Real Data Applications of the Logistic Exponential Extension Distribution with Properties. *International Journal of Latest Trends In Engineering and Technology (IJLTET)*, **18(2)**, 20-30.

Examples

```
x <- seq(0.1, 10, 0.2)
dlogis.exp.ext(x, 2.0, 5.0, 0.1)
plogis.exp.ext(x, 2.0, 5.0, 0.1)
qlogis.exp.ext(0.5, 2.0, 5.0, 0.1)
rlogis.exp.ext(10, 2.0, 5.0, 0.1)
hlogis.exp.ext(x, 2.0, 5.0, 0.1)

# Data
x <- stress31
# ML estimates
params = list(alpha=1.7919, beta=418.0473, lambda=0.1211)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.exp.ext, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.exp.ext, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
res <- gofic(x, params = params,
             dfun = dlogis.exp.ext, pfun=plogis.exp.ext, plot=TRUE)
print.gofic(res)
```

LogisExpPower *Logistic-Exponential Power (LEP) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic-Exponential Power distribution.

Usage

```
dlogis.exp.power(x, alpha, beta, lambda, log = FALSE)
plogis.exp.power(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.exp.power(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.exp.power(n, alpha, beta, lambda)
hlogis.exp.power(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Logistic-Exponential Power distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic-Exponential Power distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad 1 - \frac{1}{1 + \left\{\exp\left(e^{\lambda x^{\beta}} - 1\right) - 1\right\}^{\alpha}}; x \geq 0.$$

where α, β, and λ are the parameters.

The implementation includes the following functions:

- `dlogis.exp.power()` — Density function
- `plogis.exp.power()` — Distribution function
- `qlogis.exp.power()` — Quantile function
- `rlogis.exp.power()` — Random generation
- `hlogis.exp.power()` — Hazard function

Value

- `dlogis.exp.power`: numeric vector of (log-)densities
- `plogis.exp.power`: numeric vector of probabilities
- `qlogis.exp.power`: numeric vector of quantiles
- `rlogis.exp.power`: numeric vector of random variates
- `hlogis.exp.power`: numeric vector of hazard values

References

Joshi, R. K., Sapkota, L.P., & Kumar, V. (2020). The Logistic-Exponential Power Distribution with Statistical Properties and Applications. *International Journal of Emerging Technologies and Innovative Research*, **7(12)**, 629–641.

Examples

```
x <- seq(0.1, 2.0, 0.1)
dlogis.exp.power(x, 1.5, 1.5, 0.1)
plogis.exp.power(x, 1.5, 1.5, 0.1)
qlogis.exp.power(0.5, 1.5, 1.5, 0.1)
rlogis.exp.power(10, 2.0, 5.0, 0.1)
hlogis.exp.power(x, 1.5, 1.5, 0.1)

# Data
x <- stress
# ML estimates
params = list(alpha=1.8940, beta=1.2276, lambda=0.1650)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.exp.power, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.exp.power, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.exp.power, pfun=plogis.exp.power, plot=FALSE)
print.gofic(out)
```

LogisGompertz *Logistic-Gompertz Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic-Gompertz distribution.

Usage

```
dlogis.gpz(x, alpha, beta, lambda, log = FALSE)
plogis.gpz(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.gpz(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.gpz(n, alpha, beta, lambda)
hlogis.gpz(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Logistic-Gompertz distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic-Gompertz distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad 1 - \frac{1}{1 + \left[\exp\left\{\frac{\lambda}{\beta}\left(e^{\beta x} - 1\right)\right\} - 1\right]^{\alpha}}; x \geq 0.$$

where α, β, and λ are the parameters.

The implementation includes the following functions:

- `dlogis.gpz()` — Density function
- `plogis.gpz()` — Distribution function
- `qlogis.gpz()` — Quantile function
- `rlogis.gpz()` — Random generation
- `hlogis.gpz()` — Hazard function

Value

- `dlogis.gpz`: numeric vector of (log-)densities
- `plogis.gpz`: numeric vector of probabilities
- `qlogis.gpz`: numeric vector of quantiles
- `rlogis.gpz`: numeric vector of random variates
- `hlogis.gpz`: numeric vector of hazard values

References

Joshi, R. K., & Kumar, V. (2020). The Logistic Gompertz Distribution with Properties and Applications. *Bull. Math. & Stat. Res.*, **8(4)**, 81–94.

Examples

```
x <- seq(0.1, 2.0, 0.2)
dlogis.gpz(x, 2.0, 1.5, 0.1)
plogis.gpz(x, 2.0, 1.5, 0.1)
qlogis.gpz(0.5, 2.0, 1.5, 0.1)
rlogis.gpz(10, 2.0, 1.5, 0.1)
hlogis.gpz(x, 2.0, 1.5, 0.1)

# Data
x <- stress
# ML estimates
params = list(alpha=2.09377, beta=0.30392, lambda=0.17763)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.gpz, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.gpz, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.gpz, pfun=plogis.gpz, plot=TRUE)
print.gofic(out)
```

LogisInvExp *Logistic Inverse Exponential Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic Inverse Exponential distribution.

Usage

```
dlogis.inv.exp(x, alpha, lambda, log = FALSE)
plogis.inv.exp(q, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.inv.exp(p, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.inv.exp(n, alpha, lambda)
hlogis.inv.exp(x, alpha, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)

p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Logistic Inverse Exponential distribution is parameterized by the parameters $\alpha > 0$ and $\lambda > 0$.

The Logistic Inverse Exponential distribution has CDF:

$$F(x; \alpha, \lambda) = \frac{1}{1 + [\exp\{\lambda/x\} - 1]^{\alpha}}; \quad x > 0.$$

where α and λ are the parameters.

Available functions are:

- `dlogis.inv.exp()` — Density function
- `plogis.inv.exp()` — Distribution function
- `qlogis.inv.exp()` — Quantile function
- `rlogis.inv.exp()` — Random generation
- `hlogis.inv.exp()` — Hazard function

Value

- `dlogis.inv.exp`: numeric vector of (log-)densities
- `plogis.inv.exp`: numeric vector of probabilities
- `qlogis.inv.exp`: numeric vector of quantiles
- `rlogis.inv.exp`: numeric vector of random variates
- `hlogis.inv.exp`: numeric vector of hazard values

References

Chaudhary, A.K., & Kumar, V. (2020). Logistic Inverse Exponential Distribution with Properties and Applications. *International Journal of Mathematics Trends and Technology*, **66(10)**, 151–162. doi:10.14445/22315373/IJMTTV66I10P518

Examples

```
x <- seq(0.1, 10, 0.5)
dlogis.inv.exp(x, 2.5, 1.5)
plogis.inv.exp(x, 2.5, 1.5)
qlogis.inv.exp(0.5, 2.5, 1.5)
rlogis.inv.exp(10, 2.5, 1.5)
hlogis.inv.exp(x, 2.5, 1.5)

# Data
x <- stress31
# ML estimates
params = list(alpha=7.6230, lambda=91.7136)
#P-P (probability-probability) plot
```

```
pp.plot(x, params = params, pfun = plogis.inv.exp, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.inv.exp, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.inv.exp, pfun=plogis.inv.exp, plot=FALSE)
print.gofic(out)
```

`LogisInvLomax` *Logistic Inverted Lomax Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic Inverted Lomax distribution.

Usage

```
dlogis.inv.lomax(x, alpha, beta, lambda, log = FALSE)
plogis.inv.lomax(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.inv.lomax(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.inv.lomax(n, alpha, beta, lambda)
hlogis.inv.lomax(x, alpha, beta, lambda)
```

Arguments

`x, q`	numeric vector of quantiles (x, q)
`alpha`	positive numeric parameter
`beta`	positive numeric parameter
`lambda`	positive numeric parameter
`log`	logical; if TRUE, returns log-density
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
`log.p`	logical; if TRUE, probabilities are given as log(p)
`p`	numeric vector of probabilities (0 < p < 1)
`n`	number of observations (integer > 0)

Details

The Logistic Inverted Lomax distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic Inverted Lomax distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad 1-\left[1+\left\{\left(1+\frac{\beta}{x}\right)^{\lambda}-1\right\}^{\alpha}\right]^{-1} \quad ; x>0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dlogis.inv.lomax()` — Density function
- `plogis.inv.lomax()` — Distribution function
- `qlogis.inv.lomax()` — Quantile function
- `rlogis.inv.lomax()` — Random generation
- `hlogis.inv.lomax()` — Hazard function

Value

- `dlogis.inv.lomax`: numeric vector of (log-)densities
- `plogis.inv.lomax`: numeric vector of probabilities
- `qlogis.inv.lomax`: numeric vector of quantiles
- `rlogis.inv.lomax`: numeric vector of random variates
- `hlogis.inv.lomax`: numeric vector of hazard values

References

Joshi, R. K., & Kumar, V. (2021). The Logistic Inverse Lomax Distribution with Properties and Applications. *International Journal of Mathematics and Computer Research*, **9(1)**, 2169–2177. doi:10.47191/ijmcr/v9i1.02

Lan, Y., & Leemis, L. M. (2008). The Logistic-Exponential Survival Distribution. *Naval Research Logistics*, **55**, 252–264.

Examples

```
x <- seq(0.1, 10, 0.2)
dlogis.inv.lomax(x, 2.0, 5.0, 0.2)
plogis.inv.lomax(x, 2.0, 5.0, 0.2)
qlogis.inv.lomax(0.5, 2.0, 5.0, 0.2)
rlogis.inv.lomax(10, 2.0, 5.0, 0.2)
hlogis.inv.lomax(x, 2.0, 5.0, 0.2)

# Data
x <- bladder
# ML estimates
params = list(alpha=2.87951, beta=38.51405, lambda=0.35313)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.inv.lomax, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.inv.lomax, fit.line=TRUE)
```

```
# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.inv.lomax, pfun=plogis.inv.lomax, plot=FALSE)
print.gofic(out)
```

LogisInvWeibull *Logistic Inverse Weibull Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic Inverse Weibull distribution.

Usage

```
dlogis.inv.weib(x, alpha, beta, lambda, log = FALSE)
plogis.inv.weib(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.inv.weib(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.inv.weib(n, alpha, beta, lambda)
hlogis.inv.weib(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Logistic Inverse Weibull distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic Inverse Weibull distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad \frac{1}{1 + \left(e^{\lambda x^{-\beta}} - 1\right)^{\alpha}} \; ; \; x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dlogis.inv.weib()` — Density function
- `plogis.inv.weib()` — Distribution function
- `qlogis.inv.weib()` — Quantile function
- `rlogis.inv.weib()` — Random generation
- `hlogis.inv.weib()` — Hazard function

Value

- `dlogis.inv.weib`: numeric vector of (log-)densities
- `plogis.inv.weib`: numeric vector of probabilities
- `qlogis.inv.weib`: numeric vector of quantiles
- `rlogis.inv.weib`: numeric vector of random variates
- `hlogis.inv.weib`: numeric vector of hazard values

References

Chaudhary,A.K., & Kumar, V.(2020). A Study on Properties and Goodness-of-Fit of The Logistic Inverse Weibull Distribution. *Global Journal of Pure and Applied Mathematics(GJPAM)*, **16(6)**,871–889. doi:10.37622/GJPAM/16.6.2020.871889

Examples

```
x <- seq(0.1, 2.0, 0.2)
dlogis.inv.weib(x, 2.5, 1.5, 0.1)
plogis.inv.weib(x, 2.5, 1.5, 0.1)
qlogis.inv.weib(0.5, 2.5, 1.5, 0.1)
rlogis.inv.weib(10, 2.5, 1.5, 0.1)
hlogis.inv.weib(x, 2.5, 1.5, 0.1)

# Data
x <- stress31
# ML estimates
params = list(alpha=22.20247, beta=0.34507, lambda=3.74216)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.inv.weib, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.inv.weib, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.inv.weib, pfun=plogis.inv.weib, plot=FALSE)
print.gofic(out)
```

`LogisLomax` *Logistic-Lomax Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic-Lomax distribution.

Usage

```
dlogis.lomax(x, alpha, beta, lambda, log = FALSE)
plogis.lomax(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.lomax(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.lomax(n, alpha, beta, lambda)
hlogis.lomax(x, alpha, beta, lambda)
```

Arguments

`x, q`	numeric vector of quantiles (x, q)
`alpha`	positive numeric parameter
`beta`	positive numeric parameter
`lambda`	positive numeric parameter
`log`	logical; if TRUE, returns log-density
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
`log.p`	logical; if TRUE, probabilities are given as log(p)
`p`	numeric vector of probabilities (0 < p < 1)
`n`	number of observations (integer > 0)

Details

The Logistic-Lomax distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic-Lomax distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad 1 - \frac{1}{1 + \left((1+\beta x)^{\lambda} - 1\right)^{\alpha}} \quad ; x \geqslant 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dlogis.lomax()` — Density function
- `plogis.lomax()` — Distribution function
- `qlogis.lomax()` — Quantile function
- `rlogis.lomax()` — Random generation
- `hlogis.lomax()` — Hazard function

Value

- `dlogis.lomax`: numeric vector of (log-)densities
- `plogis.lomax`: numeric vector of probabilities
- `qlogis.lomax`: numeric vector of quantiles
- `rlogis.lomax`: numeric vector of random variates
- `hlogis.lomax`: numeric vector of hazard values

References

Chaudhary, A.K., & Kumar, V.(2020). The Logistic Lomax Distribution with Properties and Applications. *International Journal of New Technology and Research*, **6(12)**, 74–80. doi:10.31871/IJNTR.6.12.21

Shrestha, S.K., & Kumar, V. (2014). Bayesian Analysis of Extended Lomax Distribution. *International Journal of Mathematical Trends and Technology (IJMTT)*, **7(1)**, 33–41. doi:10.14445/22315373/IJMTTV7P505

Examples

```
x <- seq(0.1, 10, 0.2)
dlogis.lomax(x, 1.5, 0.1, 2.0)
plogis.lomax(x, 1.5, 0.1, 2.0)
qlogis.lomax(0.5, 1.5, 0.1, 2.0)
rlogis.lomax(10, 1.5, 0.1, 2.0)
hlogis.lomax(x, 1.5, 0.1, 2.0)

# Data
x <- bladder
# ML estimates
params = list(alpha=1.38027, beta=0.04451, lambda=2.80412)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.lomax, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.lomax, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
      dfun = dlogis.lomax, pfun=plogis.lomax, plot=FALSE)
print.gofic(out)
```

LogisModExp *Logistic Modified Exponential Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic Modified Exponential distribution.

Usage

```
dlogis.mod.exp(x, alpha, beta, lambda, log = FALSE)
plogis.mod.exp(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.mod.exp(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.mod.exp(n, alpha, beta, lambda)
hlogis.mod.exp(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Logistic Modified Exponential distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic Modified Exponential distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad 1 - \frac{1}{1 + \left[\exp\left\{\lambda x e^{\beta x}\right\} - 1\right]^{\alpha}}; x \geq 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dlogis.mod.exp()` — Density function
- `plogis.mod.exp()` — Distribution function
- `qlogis.mod.exp()` — Quantile function
- `rlogis.mod.exp()` — Random generation
- `hlogis.mod.exp()` — Hazard function

Value

- `dlogis.mod.exp`: numeric vector of (log-)densities
- `plogis.mod.exp`: numeric vector of probabilities
- `qlogis.mod.exp`: numeric vector of quantiles
- `rlogis.mod.exp`: numeric vector of random variates
- `hlogis.mod.exp`: numeric vector of hazard values

References

Chaudhary, A.K., & Kumar, V.(2020). A Study on Properties and Applications of Logistic Modified Exponential Distribution. *International Journal of Latest Trends In Engineering and Technology (IJLTET)*,**18(1)**,19–29.

Examples

```
x <- seq(0.1, 2.0, 0.2)
dlogis.mod.exp(x, 1.5, 1.5, 0.2)
plogis.mod.exp(x, 1.5, 1.5, 0.2)
qlogis.mod.exp(0.5, 1.5, 1.5, 0.2)
rlogis.mod.exp(10, 1.5, 1.5, 0.2)
hlogis.mod.exp(x, 1.5, 1.5, 0.2)

# Data
x <- stress
# ML estimates
params = list(alpha=2.0354, beta=0.1891, lambda=0.1656)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.mod.exp, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.mod.exp, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.mod.exp, pfun=plogis.mod.exp, plot=TRUE)
print.gofic(out)
```

LogisNHE *Logistic-NHE Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic-NHE distribution.

Usage

```
dlogis.NHE(x, alpha, beta, lambda, log = FALSE)
plogis.NHE(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.NHE(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.NHE(n, alpha, beta, lambda)
hlogis.NHE(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Logistic-NHE distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic-NHE distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad 1 - \frac{1}{1 + \left[\exp\left\{(1+\lambda x)^{\beta} - 1\right\} - 1\right]^{\alpha}}; x \geq 0.$$

where α, β, and λ are the parameters.

Included functions are:

- `dlogis.NHE()` — Density function
- `plogis.NHE()` — Distribution function
- `qlogis.NHE()` — Quantile function
- `rlogis.NHE()` — Random generation
- `hlogis.NHE()` — Hazard function

Value

- `dlogis.NHE`: numeric vector of (log-)densities
- `plogis.NHE`: numeric vector of probabilities
- `qlogis.NHE`: numeric vector of quantiles
- `rlogis.NHE`: numeric vector of random variates
- `hlogis.NHE`: numeric vector of hazard values

References

Chaudhary,A.K., & Kumar, V.(2020). The Logistic NHE Distribution with Properties and Applications. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, **8(12)**,591–603. doi:10.22214/ijraset.2020.32565

Examples

```
x <- seq(0.1, 2.0, 0.2)
dlogis.NHE(x, 2.0, 5.0, 0.2)
plogis.NHE(x, 2.0, 5.0, 0.1)
qlogis.NHE(0.5, 2.0, 5.0, 0.1)
rlogis.NHE(10, 2.0, 5.0, 0.1)
hlogis.NHE(x, 2.0, 5.0, 0.1)

# Data
x <- conductors
# ML estimates
params = list(alpha=4.39078, beta=6.98955, lambda=0.01133)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.NHE, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.NHE, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.NHE, pfun=plogis.NHE, plot=TRUE)
print.gofic(out)
```

`LogisRayleigh` *Logistic-Rayleigh Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic-Rayleigh distribution.

Usage

```
dlogis.rayleigh(x, alpha, lambda, log = FALSE)
plogis.rayleigh(q, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.rayleigh(p, alpha, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.rayleigh(n, alpha, lambda)
hlogis.rayleigh(x, alpha, lambda)
```

Arguments

`x, q`	numeric vector of quantiles (x, q)
`alpha`	positive numeric parameter
`lambda`	positive numeric parameter
`log`	logical; if TRUE, returns log-density
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.

log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Logistic-Rayleigh distribution is parameterized by the parameters $\alpha > 0$ and $\lambda > 0$.

The Logistic-Rayleigh distribution has CDF:

$$F(x; \alpha, \lambda) = 1 - \frac{1}{1 + \left(e^{(\lambda x^2/2)} - 1\right)^{\alpha}}; \quad x \geq 0.$$

where α and λ are the parameters.

The following functions are included:

- `dlogis.rayleigh()` — Density function
- `plogis.rayleigh()` — Distribution function
- `qlogis.rayleigh()` — Quantile function
- `rlogis.rayleigh()` — Random generation
- `hlogis.rayleigh()` — Hazard function

Value

- `dlogis.rayleigh`: numeric vector of (log-)densities
- `plogis.rayleigh`: numeric vector of probabilities
- `qlogis.rayleigh`: numeric vector of quantiles
- `rlogis.rayleigh`: numeric vector of random variates
- `hlogis.rayleigh`: numeric vector of hazard values

References

Chaudhary, A.K., & Kumar, V. (2020). The Logistic - Rayleigh Distribution with Properties and Applications. *International Journal of Statistics and Applied Mathematics*, **5(6)**, 12–19. doi:10.22271/maths.2020.v5.i6a.603

Examples

```
x <- seq(0.1, 2.0, 0.2)
dlogis.rayleigh(x, 2.0, 5.0)
plogis.rayleigh(x, 2.0, 5.0)
qlogis.rayleigh(0.5, 2.0, 5.0)
rlogis.rayleigh(10, 2.0, 5.0)
hlogis.rayleigh(x, 2.0, 5.0)

# Data
x <- conductors
# ML estimates
```

```
params = list(alpha=2.6967, lambda=0.0291)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.rayleigh, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.rayleigh, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.rayleigh, pfun=plogis.rayleigh, plot=FALSE)
print.gofic(out)
```

LogisWeib *Logistic-Weibull Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Logistic-Weibull distribution.

Usage

```
dlogis.weib(x, alpha, beta, lambda, log = FALSE)
plogis.weib(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qlogis.weib(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rlogis.weib(n, alpha, beta, lambda)
hlogis.weib(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Logistic-Weibull distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Logistic-Weibull distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad 1 - \frac{1}{1 + (\exp(\lambda x^{\beta}) - 1)^{\alpha}} \quad ; x \geqslant 0.$$

where α, β, and λ are the parameters.

Included functions are:

- `dlogis.weib()` — Density function
- `plogis.weib()` — Distribution function
- `qlogis.weib()` — Quantile function
- `rlogis.weib()` — Random generation
- `hlogis.weib()` — Hazard function

Value

- `dlogis.weib`: numeric vector of (log-)densities
- `plogis.weib`: numeric vector of probabilities
- `qlogis.weib`: numeric vector of quantiles
- `rlogis.weib`: numeric vector of random variates
- `hlogis.weib`: numeric vector of hazard values

References

Chaudhary,A.K., & Kumar, V.(2021). The Logistic-Weibull distribution with Properties and Applications. *IOSR Journal of Mathematics (IOSR-JM)*, **17(1)**,Ser.1, 32–41.

Dhungana, G.P., & Kumar, V.(2021). Modified Half Logistic Weibull Distribution with Statistical Properties and Applications. *International Journal of Statistics and Reliability Engineering*, **8(1)**, 29-39.

Examples

```
x <- seq(0.1, 10, 0.2)
dlogis.weib(x, 2.0, 0.5, 0.2)
plogis.weib(x, 2.0, 0.5, 0.2)
qlogis.weib(0.5, 2.0, 0.5, 0.2)
rlogis.weib(10, 2.0, 0.5, 0.2)
hlogis.weib(x, 2.0, 0.5, 0.2)

# Data
x <- bladder
# ML estimates
params = list(alpha=2.4165, beta=0.5103, lambda=0.2711)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = plogis.weib, fit.line=TRUE)
```

```
#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qlogis.weib, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dlogis.weib, pfun=plogis.weib, plot=FALSE)
print.gofic(out)
```

ModAtanExp *Modified Atan Exponential Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Modified Atan Exponential distribution.

Usage

```
dmod.atan.exp(x, alpha, beta, lambda, log = FALSE)
pmod.atan.exp(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qmod.atan.exp(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rmod.atan.exp(n, alpha, beta, lambda)
hmod.atan.exp(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Modified Atan Exponential distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Modified Atan Exponential distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \left[\frac{2}{\pi} \arctan\left\{\beta x e^{\alpha x}\right\}\right]^{\lambda}; x \geq 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dmod.atan.exp()` — Density function
- `pmod.atan.exp()` — Distribution function
- `qmod.atan.exp()` — Quantile function
- `rmod.atan.exp()` — Random generation
- `hmod.atan.exp()` — Hazard function

Value

- `dmod.atan.exp`: numeric vector of (log-)densities
- `pmod.atan.exp`: numeric vector of probabilities
- `qmod.atan.exp`: numeric vector of quantiles
- `rmod.atan.exp`: numeric vector of random variates
- `hmod.atan.exp`: numeric vector of hazard values

References

Chaudhary, A.K., Telee, L.B.S., & Kumar, V.(2023). Modified Arctan Exponential Distribution with application to COVID-19 Second Wave data in Nepal. *Journal of Econometrics and Statistics*, **4(1)**, 63–78.

Examples

```
x <- seq(0.1, 10, 0.2)
dmod.atan.exp(x, 0.1, 0.2, 1.2)
pmod.atan.exp(x, 0.1, 0.2, 1.2)
qmod.atan.exp(0.5, 0.1, 0.2, 1.2)
rmod.atan.exp(10, 0.1, 0.2, 1.2)
hmod.atan.exp(x, 0.1, 0.2, 1.2)

# Data
x <- bladder
# ML estimates
params = list(alpha=0.04599, beta=0.14935, lambda=1.23266)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pmod.atan.exp, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qmod.atan.exp, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dmod.atan.exp, pfun=pmod.atan.exp, plot=FALSE)
print.gofic(out)
```

ModGE *Modified Generalized Exponential (MGE) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Modified Generalized Exponential distribution.

Usage

```
dmod.gen.exp(x, alpha, beta, lambda, log = FALSE)
pmod.gen.exp(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qmod.gen.exp(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rmod.gen.exp(n, alpha, beta, lambda)
hmod.gen.exp(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Modified Generalized Exponential distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Modified Generalized Exponential distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \left[1 - \exp\left\{1 - (\exp(\beta x))^{\alpha}\right\}\right]^{\lambda}, \quad x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dmod.gen.exp()` — Density function
- `pmod.gen.exp()` — Distribution function
- `qmod.gen.exp()` — Quantile function
- `rmod.gen.exp()` — Random generation
- `hmod.gen.exp()` — Hazard function

Value

- `dmod.gen.exp`: numeric vector of (log-)densities
- `pmod.gen.exp`: numeric vector of probabilities
- `qmod.gen.exp`: numeric vector of quantiles
- `rmod.gen.exp`: numeric vector of random variates
- `hmod.gen.exp`: numeric vector of hazard values

References

Telee, L. B. S., & Kumar, V. (2023). Modified Generalized Exponential Distribution. *Nepal Journal ofMathematical Sciences*, **4(1)**, 21–32. doi:10.3126/njmathsci.v4i1.53154

Chaudhary, A. K., Sapkota, L. P., & Kumar, V.(2021). Some Properties and Application of Arctan Generalized Exponential Distribution. *International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)*, **10(1)**,456–468.

Examples

```
x <- seq(0.1, 2.0, 0.2)
dmod.gen.exp(x, 2.0, 0.5, 0.2)
pmod.gen.exp(x, 2.0, 0.5, 0.2)
qmod.gen.exp(0.5, 2.0, 0.5, 0.2)
rmod.gen.exp(10, 2.0, 0.5, 0.2)
hmod.gen.exp(x, 2.0, 0.5, 0.2)

# Data
x <- stress
# ML estimates
params = list(alpha=3.1502, beta=0.2167, lambda=0.3636)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pmod.gen.exp, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qmod.gen.exp, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dmod.gen.exp, pfun=pmod.gen.exp, plot=FALSE)
print.gofic(out)
```

`ModInvGE` *Modified Inverse Generalized Exponential(MIGE) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the MIGE distribution.

Usage

```
dmod.inv.gen.exp(x, alpha, beta, lambda, log = FALSE)
pmod.inv.gen.exp(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qmod.inv.gen.exp(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rmod.inv.gen.exp(n, alpha, beta, lambda)
hmod.inv.gen.exp(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The MIGE distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Modified Inverse Generalized Exponential(MIGE) distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad 1 - \left[1 - \exp\left(-\lambda x^{-1} e^{-\beta x}\right)\right]^{\alpha} \quad ;\ x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- dmod.inv.gen.exp() — Density function
- pmod.inv.gen.exp() — Distribution function
- qmod.inv.gen.exp() — Quantile function
- rmod.inv.gen.exp() — Random generation
- hmod.inv.gen.exp() — Hazard function

Value

- dmod.inv.gen.exp: numeric vector of (log-)densities
- pmod.inv.gen.exp: numeric vector of probabilities
- qmod.inv.gen.exp: numeric vector of quantiles
- rmod.inv.gen.exp: numeric vector of random variates
- hmod.inv.gen.exp: numeric vector of hazard values

References

Krishna, H., & Kumar, K. (2013). Reliability estimation in generalized inverted exponential distribution with progressive type II censored sample. *Journal of Statistical Computation and Simulation*, **83(6)**, 1007–1019.

Telee, L. B. S., & Kumar, V. (2023). Modified Inverse Generalized Exponential Distribution : Model and Properties. *Int. J. Res. Granthaalayah*, **11(8)**, 96–111. doi:10.29121/granthaalayah.v11.i8.2023.5288

Examples

```
x <- seq(0.1, 10, 0.2)
dmod.inv.gen.exp(x, 2.0, 0.5, 0.2)
pmod.inv.gen.exp(x, 2.0, 0.5, 0.2)
qmod.inv.gen.exp(0.5, 2.0, 0.5, 0.2)
rmod.inv.gen.exp(10, 2.0, 0.5, 0.2)
hmod.inv.gen.exp(x, 2.0, 0.5, 0.2)

# Data
x <- fibers69
# ML estimates
params = list(alpha=30.7790, beta=0.1942, lambda=14.8297)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pmod.inv.gen.exp, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qmod.inv.gen.exp, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dmod.inv.gen.exp, pfun=pmod.inv.gen.exp, plot=TRUE)
print.gofic(out)
```

ModInvLomax *Modified Inverse Lomax (MIL) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Modified Inverse Lomax distribution.

Usage

```
dmod.inv.lomax(x, alpha, beta, lambda, log = FALSE)
pmod.inv.lomax(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qmod.inv.lomax(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rmod.inv.lomax(n, alpha, beta, lambda)
hmod.inv.lomax(x, alpha, beta, lambda)
```

Arguments

x	numeric vector of strictly positive quantiles.
alpha	positive shape parameter.
beta	positive scale parameter.
lambda	positive shape/scale parameter.
log	logical; if TRUE, returns the log-density.
q	numeric vector of strictly positive quantiles.
lower.tail	logical; if TRUE (default), probabilities are $P(X \leq x)$, otherwise $P(X > x)$.
log.p	logical; if TRUE, probabilities are returned as log(p).
p	numeric vector of probabilities with values in (0, 1).
n	number of observations (positive integer).

Details

The distribution is parameterized by shape parameters $\alpha > 0$, $\beta > 0$ and scale/shape parameter $\lambda > 0$.

The cumulative distribution function (CDF) of the MIL distribution is

$$F(x; \alpha, \beta, \lambda) = \left[1 + \left(\frac{\beta}{x}\right) e^{-\lambda x}\right]^{-\alpha}, \quad x > 0.$$

Value

- dmod.inv.lomax: numeric vector of (log) densities.
- pmod.inv.lomax: numeric vector of distribution function values.
- qmod.inv.lomax: numeric vector of quantiles.
- rmod.inv.lomax: numeric vector of random variates.
- hmod.inv.lomax: numeric vector of hazard rates.

References

Telee, L.B.S., Yadav, R.S., & Kumar V.(2023). Modified Inverse Lomax Distribution: Model and properties. *Discovery*, **59: e110d1352**. doi:10.54905/disssi.v59i333.e110d1352

Examples

```
x <- seq(0.1, 5, by = 0.1)
dmod.inv.lomax(x, alpha = 1.5, beta = 2, lambda = 0.5)
pmod.inv.lomax(x, alpha = 1.5, beta = 2, lambda = 0.5)
qmod.inv.lomax(0.5, alpha = 1.5, beta = 2, lambda = 0.5)
set.seed(123)
rmod.inv.lomax(5, alpha = 1.5, beta = 2, lambda = 0.5)
hmod.inv.lomax(x, alpha = 1.5, beta = 2, lambda = 0.5)

# Data
```

```
x <- windshield
# ML estimates
params = list(alpha=0.6661, beta=26.8875, lambda=1.0004)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pmod.inv.lomax, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qmod.inv.lomax, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dmod.inv.lomax, pfun=pmod.inv.lomax, plot=FALSE)
print.gofic(out)
```

ModInvNHE *Modified Inverse NHE Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Modified Inverse NHE distribution.

Usage

```
dmod.inv.NHE(x, alpha, beta, lambda, log = FALSE)
pmod.inv.NHE(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qmod.inv.NHE(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rmod.inv.NHE(n, alpha, beta, lambda)
hmod.inv.NHE(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Modified Inverse NHE distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Modified Inverse NHE distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad \exp\left\{1 - \left(1 + \frac{\lambda}{x} e^{-\beta x}\right)^{\alpha}\right\} \quad ; \; x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dmod.inv.NHE()` — Density function
- `pmod.inv.NHE()` — Distribution function
- `qmod.inv.NHE()` — Quantile function
- `rmod.inv.NHE()` — Random generation
- `hmod.inv.NHE()` — Hazard function

Value

- `dmod.inv.NHE`: numeric vector of (log-)densities
- `pmod.inv.NHE`: numeric vector of probabilities
- `qmod.inv.NHE`: numeric vector of quantiles
- `rmod.inv.NHE`: numeric vector of random variates
- `hmod.inv.NHE`: numeric vector of hazard values

References

Chaudhary, A. K., Sapkota, L. P., & Kumar, V. (2022). Modified Inverse NHE Distribution: Properties and Application. *Journal of Institute of Science and Technology*, **27(1)**, 125—133. doi:10.3126/jist.v27i1.46695

Examples

```
x <- seq(0.1, 10, 0.2)
dmod.inv.NHE(x, 2.0, 0.5, 0.2)
pmod.inv.NHE(x, 2.0, 0.5, 0.2)
qmod.inv.NHE(0.5, 2.0, 0.5, 0.2)
rmod.inv.NHE(10, 2.0, 0.5, 0.2)
hmod.inv.NHE(x, 2.0, 0.5, 0.2)

# Data
x <- waiting
# ML estimates
params = list(alpha=0.4858, beta=0.1099, lambda=37.5129)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pmod.inv.NHE, fit.line=TRUE)
```

```
#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qmod.inv.NHE, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dmod.inv.NHE, pfun=pmod.inv.NHE, plot=FALSE)
print.gofic(out)
```

ModUbd *Modified UBD (MUBD) Distribution*

Description

Density, distribution function, quantile function, random generation, and hazard rate function for the Modified UBD (MUBD) distribution.

Usage

```
dmod.ubd(x, alpha, beta, lambda, log = FALSE)
pmod.ubd(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qmod.ubd(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rmod.ubd(n, alpha, beta, lambda)
hmod.ubd(x, alpha, beta, lambda)
```

Arguments

x	Vector of positive quantiles.
alpha	Shape parameter ($\alpha > 0$).
beta	Shape parameter ($\beta > 0$).
lambda	Scale parameter ($\lambda > 0$).
log	Logical; if TRUE, returns the log-density.
q	Vector of positive quantiles.
lower.tail	Logical; if TRUE (default), returns $P(X \leq x)$.
log.p	Logical; if TRUE, probabilities are returned on the log scale.
p	Vector of probabilities.
n	Number of random observations. Must be a positive integer.

Details

The Modified UBD (MUBD) distribution is a flexible lifetime distribution with positive shape parameters $\alpha > 0$, $\beta > 0$ and scale parameter $\lambda > 0$.

The MUDB distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad 1 - \exp\left\{1 - \left(1 + x^{\beta} e^{-\lambda/x}\right)^{\alpha}\right\} \quad ;\ x > 0.$$

where α, β, and λ are the parameters.

Value

dmod.ubd returns the probability density function. pmod.ubd returns the cumulative distribution function. qmod.ubd returns the quantile function. rmod.ubd generates random variates. hmod.ubd returns the hazard rate function.

References

Chaudhary, A.K., Telee, L. B. S., & Kumar, V. (2023). Modified Upside Down Bathtub-Shaped Hazard Function Distribution: Properties and Applications. *Journal of Econometrics and Statistics*, **3(1)**, 107–120.

Examples

```
x <- seq(0.1, 1, by=0.1)
dmod.ubd(x, alpha = 1.5, beta = 1.2, lambda = 2)
pmod.ubd(x, alpha = 1.5, beta = 1.2, lambda = 2)
qmod.ubd(0.5, alpha = 1.5, beta = 1.2, lambda = 2)
rmod.ubd(10, alpha = 1.5, beta = 1.2, lambda = 2)
hmod.ubd(x, alpha = 1.5, beta = 1.2, lambda = 2)

# Data
x <- fibers69
# ML estimates
params = list(alpha=0.8559, beta=3.0133, lambda=7.0336)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pmod.ubd, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qmod.ubd, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dmod.ubd, pfun=pmod.ubd, plot=TRUE)
print.gofic(out)
```

NewLindleyHC *New Lindley Half-Cauchy Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the New Lindley Half-Cauchy distribution.

Usage

```
dNLindley.HC(x, lambda, theta, log = FALSE)
pNLindley.HC(q, lambda, theta, lower.tail = TRUE, log.p = FALSE)
qNLindley.HC(p, lambda, theta, lower.tail = TRUE, log.p = FALSE)
```

```
rNLindley.HC(n, lambda, theta)
hNLindley.HC(x, lambda, theta)
```

Arguments

x, q	numeric vector of quantiles (x, q)
lambda	positive numeric parameter
theta	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The New Lindley Half-Cauchy distribution is parameterized by the parameters $\lambda > 0$, and $\theta > 0$.

The New Lindley Half-Cauchy distribution has CDF:

$$F(x;\lambda,\theta) = \left\{\frac{2}{\pi}\tan^{-1}\left(\frac{x}{\lambda}\right)\right\}^{\theta}\left\{1-\left(\frac{\theta}{1+\theta}\right)\ln\left[\frac{2}{\pi}\tan^{-1}\left(\frac{x}{\lambda}\right)\right]\right\} \quad ; \; x > 0.$$

whereλ and θ are the parameters.

The following functions are included:

- `dNLindley.HC()` — Density function
- `pNLindley.HC()` — Distribution function
- `qNLindley.HC()` — Quantile function
- `rNLindley.HC()` — Random generation
- `hNLindley.HC()` — Hazard function

Value

- `dNLindley.HC`: numeric vector of (log-)densities
- `pNLindley.HC`: numeric vector of probabilities
- `qNLindley.HC`: numeric vector of quantiles
- `rNLindley.HC`: numeric vector of random variates
- `hNLindley.HC`: numeric vector of hazard values

References

Chaudhary, A.K. & Kumar, V. (2020). New Lindley Half Cauchy Distribution: Theory and Applications. *International Journal of Recent Technology and Engineering (IJRTE)*, **9(4)**, 1–7. doi:10.35940/ijrte.D4734.119420

Examples

```
x <- seq(1, 10, 0.5)
dNLindley.HC(x, 0.5, 1.5)
pNLindley.HC(x, 0.5, 1.5)
qNLindley.HC(0.5, 0.5, 1.5)
rNLindley.HC(10, 0.5, 1.5)
hNLindley.HC(x, 0.5, 1.5)

# Data
x <- reactorpump
# ML estimates
params = list(lambda=0.7743, theta=1.3829)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pNLindley.HC, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qNLindley.HC, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dNLindley.HC, pfun=pNLindley.HC, plot=TRUE)
print.gofic(out)
```

`Perks` *Perks Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Perks distribution.

Usage

```
dperks(x, alpha, beta, log = FALSE)
pperks(q, alpha, beta, lower.tail = TRUE, log.p = FALSE)
qperks(p, alpha, beta, lower.tail = TRUE, log.p = FALSE)
rperks(n, alpha, beta)
hperks(x, alpha, beta)

dperks(x, alpha, beta, log = FALSE)

pperks(q, alpha, beta, lower.tail = TRUE, log.p = FALSE)

qperks(p, alpha, beta, lower.tail = TRUE, log.p = FALSE)

rperks(n, alpha, beta)

hperks(x, alpha, beta)
```

Arguments

`x, q`	numeric vector of quantiles (x, q)
`alpha`	positive numeric parameter
`beta`	positive numeric parameter
`log`	logical; if TRUE, returns log-density
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
`log.p`	logical; if TRUE, probabilities are given as log(p)
`p`	numeric vector of probabilities (0 < p < 1)
`n`	number of observations (integer > 0)

Details

The Perks distribution is parameterized by the parameters $\alpha > 0$ and $\beta > 0$.

The Perks distribution has CDF:

$$F(x; \alpha, \beta) = \quad 1 - \left(\frac{1 + \alpha}{1 + \alpha e^{\beta x}} \right) ; \quad x \geq 0.$$

where α and β are the parameters.

The following functions are included:

- `dperks()` — Density function
- `pperks()` — Distribution function
- `qperks()` — Quantile function
- `rperks()` — Random generation
- `hperks()` — Hazard function

Value

- `dperks`: numeric vector of (log-)densities
- `pperks`: numeric vector of probabilities
- `qperks`: numeric vector of quantiles
- `rperks`: numeric vector of random variates
- `hperks`: numeric vector of hazard values

References

Richards, S.J. (2008). Applying survival models to pensioner mortality data. *Bra. Actuarial Journal*, **14**, 257–303.

Chaudhary, A.K., & Kumar, V. (2013). A Bayesian Analysis of Perks Distribution via Markov Chain Monte Carlo Simulation. *Nepal Journal of Science and Technology*, **14(1)**, 153–166. doi:10.3126/njst.v14i1.8936

Richards, S. J. (2012). A handbook of parametric survival models for actuarial use. *Scandinavian Actuarial Journal*, 1–25.

Examples

```
x <- seq(0.1, 2.0, 0.1)
dperks(x, 5.0, 1.5)
pperks(x, 5.0, 1.5)
qperks(0.5, 5.0, 1.5)
rperks(10, 5.0, 1.5)
hperks(x, 5.0, 1.5)

# Data
x <- conductors
# ML estimates
params = list(alpha=4.5967e-4, beta=1.1077)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = pperks, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qperks, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dperks, pfun=pperks, plot=TRUE)
print.gofic(out)
```

PoisInvWeib *Poisson Inverse Weibull Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Poisson Inverse Weibull distribution.

Usage

```
dpois.inv.weib(x, alpha, beta, lambda, log = FALSE)
ppois.inv.weib(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.inv.weib(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.inv.weib(n, alpha, beta, lambda)
hpois.inv.weib(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density

lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Poisson Inverse Weibull distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Poisson Inverse Weibull distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad \frac{1}{(1 - e^{-\lambda})} \left[1 - \exp\left\{-\lambda \exp\left(-(\alpha/x)^{\beta}\right)\right\}\right] \quad ; \; x > 0.$$

where α, β, and λ are the parameters.

Value

- dpois.inv.weib: numeric vector of (log-)densities
- ppois.inv.weib: numeric vector of probabilities
- qpois.inv.weib: numeric vector of quantiles
- rpois.inv.weib: numeric vector of random variates
- hpois.inv.weib: numeric vector of hazard values

References

Kus, C. (2007). A new lifetime distribution. *Computational Statistics and Data Analysis*, **51**, 4497–4509.

Joshi, R. K., & Kumar, V. (2021). Poisson Inverse Weibull Distribution with Theory and Applications. *International Journal of Statistics and Systems*, **16(1)**, 1–16.

Rodrigues, G.C., Louzada, F., & Ramos, P.L.(2018). Poisson–exponential distribution: different methods of estimation. *Journal of Applied Statistics*, **45(1)**, 128–144.

Examples

```
x <- seq(0.1, 10, 0.2)
dpois.inv.weib(x, 2.0, 0.5, 0.2)
ppois.inv.weib(x, 2.0, 0.5, 0.2)
qpois.inv.weib(0.5, 2.0, 0.5, 0.2)
rpois.inv.weib(10, 2.0, 0.5, 0.2)
hpois.inv.weib(x, 2.0, 0.5, 0.2)

# Data
x <- fibers63
# ML estimates
params = list(alpha=5.5146, beta=1.8811, lambda=16.2341)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.inv.weib, fit.line=TRUE)
```

```
#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.inv.weib, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dpois.inv.weib, pfun=ppois.inv.weib, plot=TRUE)
print.gofic(out)
```

PoissonChen *Poisson-Chen Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Poisson-Chen distribution.

Usage

```
dpois.chen(x, alpha, beta, lambda, log = FALSE)
ppois.chen(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.chen(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.chen(n, alpha, beta, lambda)
hpois.chen(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Poisson-Chen distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Poisson-Chen distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad 1 - \frac{1}{1-e^{-\lambda}}\left[1 - \exp\left\{-\lambda\ e^{\beta(1-e^{x^{\alpha}})}\right\}\right] \quad ;\ x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dpois.chen()` — Density function
- `ppois.chen()` — Distribution function
- `qpois.chen()` — Quantile function
- `rpois.chen()` — Random generation
- `hpois.chen()` — Hazard function

Value

- `dpois.chen`: numeric vector of (log-)densities
- `ppois.chen`: numeric vector of probabilities
- `qpois.chen`: numeric vector of quantiles
- `rpois.chen`: numeric vector of random variates
- `hpois.chen`: numeric vector of hazard values

References

Joshi, R. K., & Kumar, V. (2021). Poisson Chen Distribution: Properties and Application. *International Journal of Latest Trends in Engineering and Technology*, **18(4)**, 1–12.

Examples

```
x <- seq(0.1, 2.0, 0.2)
dpois.chen(x, 2.0, 0.5, 0.2)
ppois.chen(x, 2.0, 0.5, 0.2)
qpois.chen(0.5, 2.0, 0.5, 0.2)
rpois.chen(10, 2.0, 0.5, 0.2)
hpois.chen(x, 2.0, 0.5, 0.2)

# Data
x <- fibers63
# ML estimates
params = list(alpha=0.53679, beta=1.00238, lambda=108.22948)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.chen, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.chen, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dpois.chen, pfun=ppois.chen, plot=TRUE)
print.gofic(out)
```

`PoissonExpPower` *Poisson Exponential Power Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Poisson Exponential Power distribution.

Usage

```
dpois.exp.pow(x, alpha, beta, lambda, log = FALSE)
ppois.exp.pow(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.exp.pow(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.exp.pow(n, alpha, beta, lambda)
hpois.exp.pow(x, alpha, beta, lambda)
```

Arguments

`x, q`	numeric vector of quantiles (x, q)
`alpha`	positive numeric parameter
`beta`	positive numeric parameter
`lambda`	positive numeric parameter
`log`	logical; if TRUE, returns log-density
`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
`log.p`	logical; if TRUE, probabilities are given as log(p)
`p`	numeric vector of probabilities (0 < p < 1)
`n`	number of observations (integer > 0)

Details

The Poisson Exponential Power distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Poisson Exponential Power distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad 1 - \frac{1}{(1 - e^{-\lambda})} \left[1 - \exp\left\{-\lambda \exp\left(1 - e^{\beta x^{\alpha}}\right)\right\}\right] \quad ; \; x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dpois.exp.pow()` — Density function
- `ppois.exp.pow()` — Distribution function
- `qpois.exp.pow()` — Quantile function
- `rpois.exp.pow()` — Random generation
- `hpois.exp.pow()` — Hazard function

Value

- dpois.exp.pow: numeric vector of (log-)densities
- ppois.exp.pow: numeric vector of probabilities
- qpois.exp.pow: numeric vector of quantiles
- rpois.exp.pow: numeric vector of random variates
- hpois.exp.pow: numeric vector of hazard values

References

Joshi, R. K., & Kumar, V. (2020). Poisson Exponential Power distribution: Properties and Application. *International Journal of Mathematics & Computer Research*, **8(11)**, 2152–2158. doi:10.47191/ijmcr/v8i11.01

Examples

```
x <- seq(0.1, 2.0, 0.2)
dpois.exp.pow(x, 2.0, 0.5, 0.2)
ppois.exp.pow(x, 2.0, 0.5, 0.2)
qpois.exp.pow(0.5, 2.0, 0.5, 0.2)
rpois.exp.pow(10, 2.0, 0.5, 0.2)
hpois.exp.pow(x, 2.0, 0.5, 0.2)

# Data
x <- stress
# ML estimates
params = list(alpha=0.6976, beta=0.6395, lambda=7.8045)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.exp.pow, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.exp.pow, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dpois.exp.pow, pfun=ppois.exp.pow, plot=TRUE)
print.gofic(out)
```

PoissonGenRayleigh *Poisson Generalized Rayleigh (PGR) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the PGR distribution.

Usage

```
dpois.gen.rayleigh(x, alpha, beta, lambda, log = FALSE)
ppois.gen.rayleigh(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.gen.rayleigh(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.gen.rayleigh(n, alpha, beta, lambda)
hpois.gen.rayleigh(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The PGR distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The PGR distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad \frac{1}{(1-e^{-\lambda})}\left[1-\exp\left\{-\lambda\left(1-e^{-\beta x^2}\right)^{\alpha}\right\}\right] \quad ;\ x>0.$$

where α, β, and λ are the parameters.

The functions available are listed below:

- `dpois.gen.rayleigh()` — Density function
- `ppois.gen.rayleigh()` — Distribution function
- `qpois.gen.rayleigh()` — Quantile function
- `rpois.gen.rayleigh()` — Random generation
- `hpois.gen.rayleigh()` — Hazard function

Value

- `dpois.gen.rayleigh`: numeric vector of (log-)densities
- `ppois.gen.rayleigh`: numeric vector of probabilities
- `qpois.gen.rayleigh`: numeric vector of quantiles
- `rpois.gen.rayleigh`: numeric vector of random variates
- `hpois.gen.rayleigh`: numeric vector of hazard values

References

Joshi, R.K., & Kumar, V. (2021). Poisson Generalized Rayleigh Distribution with Properties and Application. *International Journal of Statistics and Applied Mathematics*, **6(1)**, 90–99. doi:10.22271/maths.2021.v6.i1b.637

Examples

```
x <- seq(0.1, 2.0, 0.2)
dpois.gen.rayleigh(x, 2.0, 0.5, 0.2)
ppois.gen.rayleigh(x, 2.0, 0.5, 0.2)
qpois.gen.rayleigh(0.5, 2.0, 0.5, 0.2)
rpois.gen.rayleigh(10, 2.0, 0.5, 0.2)
hpois.gen.rayleigh(x, 2.0, 0.5, 0.2)

# Data
x <- stress
# ML estimates
params = list(alpha=1.5466, beta=0.0211, lambda=16.4523)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.gen.rayleigh, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.gen.rayleigh, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
            dfun = dpois.gen.rayleigh, pfun=ppois.gen.rayleigh, plot=TRUE)
print.gofic(out)
```

PoissonGPZ *Poisson-Gompertz Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Poisson-Gompertz distribution.

Usage

```
dpois.gpz(x, alpha, beta, lambda, log = FALSE)
ppois.gpz(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.gpz(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.gpz(n, alpha, beta, lambda)
hpois.gpz(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Poisson-Gompertz distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Poisson-Gompertz distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad 1 - \frac{1}{(1-e^{-\lambda})}\left[1 - \exp\left\{-\lambda \exp\left(\frac{\beta}{\alpha}\left(1-e^{\alpha x}\right)\right)\right\}\right] \quad ; \; x > 0.$$

where α, β, and λ are the parameters.

The functions available are listed below:

- `dpois.gpz()` — Density function
- `ppois.gpz()` — Distribution function
- `qpois.gpz()` — Quantile function
- `rpois.gpz()` — Random generation
- `hpois.gpz()` — Hazard function

Value

- `dpois.gpz`: numeric vector of (log-)densities
- `ppois.gpz`: numeric vector of probabilities
- `qpois.gpz`: numeric vector of quantiles
- `rpois.gpz`: numeric vector of random variates
- `hpois.gpz`: numeric vector of hazard values

References

Chaudhary,A.K., Sapkota,L.P., & Kumar, V. (2021). Poisson Gompertz Distribution with Properties and Applications. *International Journal of Applied Engineering Research (IJEAR)*, **16(1)**,75–84. doi:10.37622/IJAER/16.1.2021.7584

Examples

```
x <- seq(0.1, 2.0, 0.2)
dpois.gpz(x, 2.0, 0.5, 0.2)
ppois.gpz(x, 2.0, 0.5, 0.2)
qpois.gpz(0.5, 2.0, 0.5, 0.2)
rpois.gpz(10, 2.0, 0.5, 0.2)
hpois.gpz(x, 2.0, 0.5, 0.2)

# Data
x <- stress
# ML estimates
params = list(alpha=0.2211, beta=0.6540, lambda=6.5456)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.gpz, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.gpz, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dpois.gpz, pfun=ppois.gpz, plot=FALSE)
print.gofic(out)
```

PoissonInvLomax *Poisson Inverse Lomax Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Poisson Inverse Lomax distribution.

Usage

```
dpois.inv.lomax(x, alpha, beta, lambda, log = FALSE)
ppois.inv.lomax(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.inv.lomax(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.inv.lomax(n, alpha, beta, lambda)
hpois.inv.lomax(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density

`lower.tail`	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
`log.p`	logical; if TRUE, probabilities are given as log(p)
`p`	numeric vector of probabilities (0 < p < 1)
`n`	number of observations (integer > 0)

Details

The Poisson Inverse Lomax distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Poisson Inverse Lomax distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \frac{1}{(1 - e^{-\lambda})} \left[1 - \exp\left\{-\lambda(1 + \beta/x)^{-\alpha}\right\}\right] \quad ; \; x > 0.$$

where α, β, and λ are the parameters.

The functions available are listed below:

- `dpois.inv.lomax()` — Density function
- `ppois.inv.lomax()` — Distribution function
- `qpois.inv.lomax()` — Quantile function
- `rpois.inv.lomax()` — Random generation
- `hpois.inv.lomax()` — Hazard function

Value

- `dpois.inv.lomax`: numeric vector of (log-)densities
- `ppois.inv.lomax`: numeric vector of probabilities
- `qpois.inv.lomax`: numeric vector of quantiles
- `rpois.inv.lomax`: numeric vector of random variates
- `hpois.inv.lomax`: numeric vector of hazard values

References

Joshi, R.K., & Kumar, V. (2021). Poisson Inverted Lomax Distribution: Properties and Applications. *International Journal of Research in Engineering and Science (IJRES)*, **9(1)**, 48–57.

Chaudhary, A. K., & Kumar, V.(2021). The ArcTan Lomax Distribution with Properties and Applications. *International Journal of Scientific Research in Science, Engineering and Technology(IJSRSET)*, **8(1)**, 117–125. doi:10.32628/IJSRSET218117

Examples

```
x <- seq(0.1, 10, 0.2)
dpois.inv.lomax(x, 2.0, 0.5, 0.2)
ppois.inv.lomax(x, 2.0, 0.5, 0.2)
qpois.inv.lomax(0.5, 2.0, 0.5, 0.2)
rpois.inv.lomax(10, 2.0, 0.5, 0.2)
hpois.inv.lomax(x, 2.0, 0.5, 0.2)

# Data
x <- stress
# ML estimates
params = list(alpha=4.1507, beta=5.4091, lambda=80.5762)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.inv.lomax, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.inv.lomax, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dpois.inv.lomax, pfun=ppois.inv.lomax, plot=FALSE)
print.gofic(out)
```

PoissonInvNHE *Poisson Inverse NHE Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Poisson Inverse NHE distribution.

Usage

```
dpois.inv.NHE(x, alpha, beta, lambda, log = FALSE)
ppois.inv.NHE(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.inv.NHE(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.inv.NHE(n, alpha, beta, lambda)
hpois.inv.NHE(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density

lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Poisson Inverse NHE distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Poisson Inverse NHE distribution has CDF:

$$F(x; \alpha, \beta, \lambda) = \quad \frac{1 - \exp\left[-\lambda \exp\left\{1 - (1 + \alpha/x)^{\beta}\right\}\right]}{1 - \exp(-\lambda)} \quad ; \; x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- dpois.inv.NHE() — Density function
- ppois.inv.NHE() — Distribution function
- qpois.inv.NHE() — Quantile function
- rpois.inv.NHE() — Random generation
- hpois.inv.NHE() — Hazard function

Value

- dpois.inv.NHE: numeric vector of (log-)densities
- ppois.inv.NHE: numeric vector of probabilities
- qpois.inv.NHE: numeric vector of quantiles
- rpois.inv.NHE: numeric vector of random variates
- hpois.inv.NHE: numeric vector of hazard values

References

Chaudhary,A.K.& Kumar, V.(2020). Poisson Inverse NHE Distribution. *International Journal of Science and Research(IJSR)*, **9(12)**, 1603–1610.

Examples

```
x <- seq(0.1, 10, 0.2)
dpois.inv.NHE(x, 2.0, 0.5, 0.2)
ppois.inv.NHE(x, 2.0, 0.5, 0.2)
qpois.inv.NHE(0.5, 2.0, 0.5, 0.2)
rpois.inv.NHE(10, 2.0, 0.5, 0.2)
hpois.inv.NHE(x, 2.0, 0.5, 0.2)

# Data
x <- fibers63
# ML estimates
```

```
params = list(alpha=1.0174, beta=5.1414, lambda=23.3476)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.inv.NHE, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.inv.NHE, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dpois.inv.NHE, pfun=ppois.inv.NHE, plot=FALSE)
print.gofic(out)
```

PoissonInvSGZ *Poisson Inverse Shifted Gompertz (PISG) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Poisson Inverse Shifted Gompertz distribution.

Usage

```
dpois.inv.sgz(x, alpha, beta, lambda, log = FALSE)
ppois.inv.sgz(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.inv.sgz(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.inv.sgz(n, alpha, beta, lambda)
hpois.inv.sgz(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Poisson Inverse Shifted Gompertz distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Poisson Inverse Shifted Gompertz distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = 1 - \frac{1}{(1-e^{-\lambda})}\left[1 - \exp\left\{-\lambda\left(1-e^{-\beta/x}\right)\exp\left(-\alpha e^{-\beta/x}\right)\right\}\right] \quad ; \; x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dpois.inv.sgz()` — Density function
- `ppois.inv.sgz()` — Distribution function
- `qpois.inv.sgz()` — Quantile function
- `rpois.inv.sgz()` — Random generation
- `hpois.inv.sgz()` — Hazard function

Value

- `dpois.inv.sgz`: numeric vector of (log-)densities
- `ppois.inv.sgz`: numeric vector of probabilities
- `qpois.inv.sgz`: numeric vector of quantiles
- `rpois.inv.sgz`: numeric vector of random variates
- `hpois.inv.sgz`: numeric vector of hazard values

References

Sapkota, L. P., Kumar, V., Tekle, G., Alrweili, H., Mustafa, M. S., & Yusuf, M. (2025). Fitting Real Data Sets by a New Version of Gompertz Distribution. *Modern Journal of Statistics*, **1(1)**, 25–48. doi:10.64389/mjs.2025.01109

Examples

```
x <- seq(0.1, 10, 0.2)
dpois.inv.sgz(x, 2.0, 0.5, 0.2)
ppois.inv.sgz(x, 2.0, 0.5, 0.2)
qpois.inv.sgz(0.5, 2.0, 0.5, 0.2)
rpois.inv.sgz(10, 2.0, 0.5, 0.2)
hpois.inv.sgz(x, 2.0, 0.5, 0.2)

# Data
x <- fibers69
# ML estimates
params = list(alpha=98.0893, beta=10.6326, lambda=2.1006)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.inv.sgz, fit.line=TRUE)
```

```
#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.inv.sgz, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dpois.inv.sgz, pfun=ppois.inv.sgz, plot=FALSE)
print.gofic(out)
```

PoissonNHE *Poisson-NHE Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Poisson-NHE distribution.

Usage

```
dpois.NHE(x, alpha, beta, lambda, log = FALSE)
ppois.NHE(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.NHE(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.NHE(n, alpha, beta, lambda)
hpois.NHE(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Poisson-NHE distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Poisson-NHE distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad 1 - \frac{1 - \exp\left(-\lambda \exp\left\{\left\{1-(1+\alpha x)^{\beta}\right\}\right\}\right)}{(1-e^{-\lambda})} \quad ; \; x > 0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dpois.NHE()` — Density function
- `ppois.NHE()` — Distribution function
- `qpois.NHE()` — Quantile function
- `rpois.NHE()` — Random generation
- `hpois.NHE()` — Hazard function

Value

- `dpois.NHE`: numeric vector of (log-)densities
- `ppois.NHE`: numeric vector of probabilities
- `qpois.NHE`: numeric vector of quantiles
- `rpois.NHE`: numeric vector of random variates
- `hpois.NHE`: numeric vector of hazard values

References

Chaudhary,A.K., & Kumar, V.(2020). Poisson NHE Distribution: Properties and Applications. *International Journal of Applied Research(IJAR)*, **6(12)**,399–409. doi:10.22271/allresearch.2020.v6.i12f.8143

Examples

```
x <- seq(0.1, 10, 0.2)
dpois.NHE(x, 2.0, 0.5, 0.2)
ppois.NHE(x, 2.0, 0.5, 0.2)
qpois.NHE(0.5, 2.0, 0.5, 0.2)
rpois.NHE(10, 2.0, 0.5, 0.2)
hpois.NHE(x, 2.0, 0.5, 0.2)

# Data
x <- fibers63
# ML estimates
params = list(alpha=0.5038, beta=1.8272, lambda=53.4573)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.NHE, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.NHE, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dpois.NHE, pfun=ppois.NHE, plot=FALSE)
print.gofic(out)
```

PoissonSGZ *Poisson Shifted Gompertz (PSG) Distribution*

Description

Provides density, distribution, quantile, random generation, and hazard functions for the Poisson Shifted Gompertz distribution.

Usage

```
dpois.shifted.gz(x, alpha, beta, lambda, log = FALSE)
ppois.shifted.gz(q, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
qpois.shifted.gz(p, alpha, beta, lambda, lower.tail = TRUE, log.p = FALSE)
rpois.shifted.gz(n, alpha, beta, lambda)
hpois.shifted.gz(x, alpha, beta, lambda)
```

Arguments

x, q	numeric vector of quantiles (x, q)
alpha	positive numeric parameter
beta	positive numeric parameter
lambda	positive numeric parameter
log	logical; if TRUE, returns log-density
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$.
log.p	logical; if TRUE, probabilities are given as log(p)
p	numeric vector of probabilities (0 < p < 1)
n	number of observations (integer > 0)

Details

The Poisson Shifted Gompertz distribution is parameterized by the parameters $\alpha > 0$, $\beta > 0$, and $\lambda > 0$.

The Poisson Shifted Gompertz distribution has CDF:

$$F(x;\alpha,\beta,\lambda) = \quad \frac{1}{(1-e^{-\lambda})}\left\{1-\exp\left[-\lambda\left\{1-\left(1-e^{-\beta x}\right)\exp\left(-\alpha e^{-\beta x}\right)\right\}\right]\right\} \quad ;\ x>0.$$

where α, β, and λ are the parameters.

The following functions are included:

- `dpois.shifted.gz()` — Density function
- `ppois.shifted.gz()` — Distribution function
- `qpois.shifted.gz()` — Quantile function
- `rpois.shifted.gz()` — Random generation
- `hpois.shifted.gz()` — Hazard function

Value

- dpois.shifted.gz: numeric vector of (log-)densities
- ppois.shifted.gz: numeric vector of probabilities
- qpois.shifted.gz: numeric vector of quantiles
- rpois.shifted.gz: numeric vector of random variates
- hpois.shifted.gz: numeric vector of hazard values

References

Chaudhary,A.K., & Kumar, V. (2021). Poisson Shifted Gompertz Distribution: Properties and Applications. *International Journal of Recent Technology and Engineering (IJRTE)* ,**9(5)**,202–208. doi:10.35940/ijrte.E5265.019521

Examples

```
x <- seq(0.1, 10, 0.2)
dpois.shifted.gz(x, 2.0, 0.5, 0.2)
ppois.shifted.gz(x, 2.0, 0.5, 0.2)
qpois.shifted.gz(0.5, 2.0, 0.5, 0.2)
rpois.shifted.gz(10, 2.0, 0.5, 0.2)
hpois.shifted.gz(x, 2.0, 0.5, 0.2)

# Data
x <- fibers63
# ML estimates
params = list(alpha=13.5877, beta=2.0139, lambda=18.8875)
#P-P (probability-probability) plot
pp.plot(x, params = params, pfun = ppois.shifted.gz, fit.line=TRUE)

#Q-Q (quantile-quantile) plot
qq.plot(x, params = params, qfun = qpois.shifted.gz, fit.line=TRUE)

# Goodness-of-Fit(GoF) and Model Diagnostics
out <- gofic(x, params = params,
             dfun = dpois.shifted.gz, pfun=ppois.shifted.gz, plot=FALSE)
print.gofic(out)
```

---

pp.plot *Generic Probability-Probability(P–P) Plot Function*

Description

Generates a P–P (probability–probability) plot for any custom or built-in probability distribution. The function compares the empirical probabilities of the sample data with the theoretical probabilities computed from a user-specified cumulative distribution function (CDF).

Usage

```
pp.plot(sample, pfun, params, fit.line = TRUE)
```

Arguments

sample	A numeric vector of sample observations.
pfun	A cumulative distribution function (CDF) corresponding to the theoretical distribution (e.g., pnorm, pexp, or a custom CDF).
params	A named list of distribution parameters (e.g., list(mean = 0, sd = 1) or list(alpha = 2, lambda = 1)).
fit.line	Logical; if TRUE (default), a red least-squares regression line is added to the plot. Also, displays the regression line equation and R^2 value on the plot.

Details

The P–P plot is used to assess how closely the empirical distribution of a dataset matches a specified theoretical distribution. The points should ideally fall along the 45° reference line if the model fits well.

Requires user-defined function 'pfun' for the CDF of the user-defined continuous distribution.

Missing values in the sample are automatically removed with a warning.

Value

This function returns no value; it produces a P–P plot.

Examples

```
# Example 1: Exponential distribution
set.seed(123)
x <- rexp(100, rate = 2)
pp.plot(x, pexp, list(rate = 2))

# Example 2: Customizing the fitted line
pp.plot(x, pexp, list(rate = 2),
              fit.line = TRUE)

# Example 3: Without regression line
pp.plot(x, pexp, list(rate = 2), fit.line = FALSE)

# Example 4: Display regression equation and R² value
pp.plot(x, pexp, list(rate = 2))

# Example 5: For a user defined distribution
# Exponentiated Exponential Power (EEP) Distribution
# Data
x <- waiting
pp.plot(x,
      params = list(alpha=0.3407, lambda=0.6068, theta=7.6150),
      pfun = pgen.exp.power, fit.line=TRUE)
```

print.gofic *Print Method for* gofic *Objects*

Description

Nicely formats and prints the results produced by gofic().

Usage

```
## S3 method for class 'gofic'
print(x, ...)
```

Arguments

x	An object of class "gofic" as returned by gofic().
...	Further arguments (currently unused).

Value

The input object x, returned invisibly. No value is returned for computational purposes; used for side effects.

See Also

gofic

qq.plot *Generic Quantile-Quantile(Q-Q) Plot Function*

Description

Generates a Q-Q (quantile–quantile) plot for any custom or built-in probability distribution. The function compares sample quantiles with theoretical quantiles computed using a user-specified quantile function.

Usage

```
qq.plot(sample, qfun, params, fit.line = FALSE)
```

Arguments

sample	A numeric vector of sample observations.
qfun	A quantile function corresponding to the theoretical distribution (e.g., qnorm, qexp, or a custom quantile function).
params	A named list of distribution parameters (e.g., list(mean = 0, sd = 1) or list(alpha = 2, lambda = 1)).
fit.line	Logical; if TRUE (default), a red least-squares regression line is added to the plot. Also, displays the regression line equation and R-square value on the plot.

Details

The function is general and can be used with any continuous distribution for which a quantile function is available. It overlays both a 45° reference line and (optionally) a fitted linear regression line through the points, enabling visual assessment of model fit. Also, displays the regression line equation and R^2 value on the plot.

Requires user-defined function `'qfun'` for the CDF of the user-defined continuous distribution.

Missing values in the sample are automatically removed with a warning.

Value

This function returns no value; it produces a Q-Q plot.

Examples

```
# Example 1: Exponential distribution
set.seed(123)
x <- rexp(100, rate = 2)
qq.plot(x, qexp, list(rate = 2))

# Example 2: Customizing the fitted line
qq.plot(x, qexp, list(rate = 2),
              fit.line = TRUE)

# Example 3: Without regression line
qq.plot(x, qexp, list(rate = 2), fit.line = FALSE)

# Example 4: Display regression equation and R-square value
qq.plot(x, qexp, list(rate = 2), fit.line = TRUE)

# Example 5: For a user defined distribution
# Exponentiated Exponential Power (EEP) Distribution
#Data
x <- waiting
qq.plot(x,
      params = list(alpha=0.3407, lambda=0.6068, theta=7.6150),
      qfun = qgen.exp.power, fit.line=TRUE)
```

---

rainfall *March Rainfall in Minneapolis/St. Paul*

Description

A dataset of thirty consecutive March precipitation values (in inches) recorded in Minneapolis/St. Paul. These data were originally presented by Hinkley (1977) in the context of power transformations and applied statistical analysis.

Usage

```
rainfall
```

Format

A numeric vector of length 30 containing March rainfall amounts in inches.

Details

Hinkley (1977) used this dataset to illustrate methods for selecting power transformations in statistical modeling. The dataset is frequently cited in regression diagnostics and transformation literature.

Value

An object of class `"numeric"`.

The vector consists of 30 observed precipitation amounts (in inches) recorded for the month of March in Minneapolis/St. Paul over consecutive years. Each value represents the total March rainfall for a single year. The dataset is commonly used to illustrate power transformations, regression diagnostics, and exploratory data analysis techniques in applied statistics.

References

Hinkley, D. (1977). On quick choice of power transformations. *Journal of the Royal Statistical Society, Series C (Applied Statistics)*, 26, 67–69.

Examples

```
summary(rainfall)

hist(
  rainfall,
  main = "March Rainfall Histogram",
  xlab = "Rainfall (inches)"
)

plot(
  rainfall,
  type = "o",
  main = "March Rainfall Series",
  ylab = "Inches",
  xlab = "Observation"
)
```

`reactorpump` *Failure Time Intervals of Secondary Reactor Pumps*

Description

This dataset contains the time intervals between failures (in thousands of hours) of secondary reactor pumps. The data were reported by Salman Suprawhardana, Prayoto, and Sangadji (1999) and later analyzed in Bebbington, Lai, and Zitikis (2007) in the context of flexible Weibull extensions.

Usage

```
reactorpump
```

Format

A numeric vector of length 23 containing time intervals between pump failures, measured in thousands of hours.

Details

These data are commonly used in the reliability engineering literature, particularly for assessing lifetime distributions, hazard shapes, and model flexibility in mechanical systems. The pump failure times originate from components of the RSG-GAS reactor.

Value

An object of class `"numeric"`.

The vector consists of 23 observed time intervals between successive failures of secondary reactor pumps, measured in thousands of operating hours. Each value represents the elapsed time between two consecutive failure events for a pump component. The dataset is commonly used in reliability engineering and survival analysis for modeling lifetime distributions, studying hazard rate shapes, and evaluating the flexibility of parametric failure-time models.

References

Bebbington, M., Lai, C.-D., & Zitikis, R. (2007). A flexible Weibull extension. *Reliability Engineering and System Safety*, 92, 719–726.

Salman Suprawhardana, M., Prayoto, & Sangadji (1999). Total time on test plot analysis for mechanical components of the RSG-GAS reactor. *Atom Indones*, 25(2).

Examples

```
summary(reactorpump)

plot(
  reactorpump,
  type = "b",
  main = "Reactor Pump Failure Intervals",
```

```
  ylab = "Thousands of Hours",
  xlab = "Observation"
)

hist(
  reactorpump,
  main = "Histogram of Failure Intervals",
  xlab = "Thousands of Hours"
)
```

`relief` *Relief Times of Patients Receiving an Analgesic*

Description

This dataset contains the relief times (in hours) of 20 patients who received an analgesic. The data were originally presented by Gross and Clark (1976) in their work on survival distributions and reliability applications in biomedical sciences.

Usage

```
relief
```

Format

A numeric vector of length 20 containing relief times in hours.

Details

The dataset is frequently used in survival analysis to illustrate basic distributional behavior, reliability concepts, and nonparametric survival estimation. It serves as a benchmark example in many survival analysis textbooks.

Value

An object of class `"numeric"`.

The vector consists of 20 observed relief times (in hours), each corresponding to a single patient who received an analgesic treatment. Each value represents the time elapsed from administration of the analgesic to the onset of pain relief. The dataset is commonly used in survival and reliability analysis to illustrate lifetime distributions, time-to-event modeling, and nonparametric estimation techniques.

References

Gross, A. J., & Clark, V. A. (1976). *Survival Distributions: Reliability Applications in the Biomedical Sciences*. Wiley, New York.

Examples

```
summary(relief)

hist(
  relief,
  main = "Relief Times Histogram",
  xlab = "Relief Time (hours)"
)

plot(
  relief,
  type = "b",
  main = "Relief Times",
  xlab = "Patient",
  ylab = "Time (hours)"
)
```

---

stress *Breaking Stress of Carbon Fibres*

Description

The dataset contains 100 observations on the breaking stress (in GPa) of carbon fibres. These measurements were originally reported in Nichols and Padgett (2006) in the context of bootstrap control charts for Weibull percentiles.

Usage

```
stress
```

Format

A numeric vector of length 100 giving observed breaking stress values (in GPa).

Details

The breaking stress of carbon fibres is an important characteristic in materials science and reliability engineering. The data have been widely used in studies involving Weibull distributions, reliability modelling, and bootstrap-based inference.

The dataset is frequently cited in literature dealing with Weibull percentiles and nonparametric control charts.

Value

An object of class `"numeric"`.

The vector consists of 100 observed breaking stress measurements (in gigapascals) for individual carbon fibre specimens. Each value represents the stress level at which a single fibre failed. The dataset is commonly used in reliability engineering and materials science for modeling strength

distributions, fitting Weibull models, and illustrating bootstrap-based inference and control chart methods.

References

Nichols, M. D., & Padgett, W. J. (2006). *A bootstrap control chart for Weibull percentiles*. Quality and Reliability Engineering International, 22, 141–151.

Examples

```
stress

# Summary statistics
summary(stress)

# Histogram
hist(
  stress,
  main = "Breaking Stress of Carbon Fibres",
  xlab = "Stress (GPa)"
)
```

---

stress31 *Fatigue Life of 6061-T6 Aluminum Coupons under 31,000 psi*

Description

Fatigue life measurements (in thousands of cycles) of 6061-T6 aluminum coupons cut parallel to the direction of rolling and oscillated at 18 cycles per second (cps). The dataset contains 101 observations and was originally analyzed by Birnbaum and Saunders (1969).

Usage

```
stress31
```

Format

A numeric vector of length 101 representing fatigue life measurements of aluminum coupons subjected to 31,000 psi maximum stress per cycle.

Details

This dataset corresponds to the well-known Birnbaum–Saunders fatigue life example. The data represent time-to-failure observations collected from aluminum coupons tested in a controlled experimental setup. These data have been widely used in the literature to illustrate lifetime modeling, particularly the Birnbaum–Saunders distribution.

Value

An object of class "numeric".

The vector consists of 101 observed fatigue life measurements, expressed in thousands of cycles to failure, for individual 6061-T6 aluminum coupons tested under a maximum cyclic stress of 31,000 psi. Each value represents the number of load cycles endured by a coupon before failure. The dataset is widely used in reliability engineering and survival analysis to illustrate lifetime modeling and inference based on the Birnbaum–Saunders fatigue life distribution.

References

Birnbaum, Z. W., & Saunders, S. C. (1969). Estimation for a family of life distributions with applications to fatigue. *Journal of Applied Probability*, 6, 328–347. doi:10.2307/3212004

Examples

```
data(stress31)

summary(stress31)

hist(
  stress31,
  main = "Fatigue Life at 31,000 psi",
  xlab = "Cycles to Failure (thousands)"
)
```

---

stress66 *Breaking Stress of 66 Carbon Fibers of Length 50 mm*

Description

This dataset contains the breaking stress (in GPa) of 66 carbon fibers of length 50 mm. The data were originally used by Nichols and Padgett (2006) in their study on bootstrap control charts for Weibull percentiles.

Usage

```
stress66
```

Format

A numeric vector of length 66 containing breaking stress values measured in gigapascals (GPa).

Details

The carbon fiber breaking stress dataset is commonly used in reliability analysis, survival models, and goodness-of-fit studies involving lifetime and strength distributions. Nichols and Padgett (2006) applied these data in developing bootstrap control charts based on Weibull percentiles.

Value

An object of class "numeric".

The vector consists of 66 observed breaking stress measurements (in gigapascals) for individual carbon fiber specimens of length 50 mm. Each value represents the stress level at which a single fiber failed. The dataset is commonly used in reliability analysis, survival modeling, and goodness-of-fit studies involving strength and lifetime distributions, particularly Weibull models and bootstrap-based control charts.

References

Nichols, M. D., & Padgett, W. J. (2006). A Bootstrap Control Chart for Weibull Percentiles. *Quality and Reliability Engineering International*, 22(2), 141–151.

Examples

```
summary(stress66)

plot(
  stress66,
  type = "h",
  main = "Breaking Stress Values",
  xlab = "Observation",
  ylab = "Stress (GPa)"
)

hist(
  stress66,
  main = "Histogram of Breaking Stress",
  xlab = "Stress (GPa)"
)
```

---

survtimes *Survival Times of Guinea Pigs Infected with Tubercle Bacilli*

Description

The survtimes data set contains the survival times (in days) of 72 guinea pigs infected with virulent tubercle bacilli. These data were originally reported by Bjerkedal (1960) in a study of the acquisition of resistance in guinea pigs subjected to varying doses of tubercle bacilli.

Usage

```
data(survtimes)
```

Format

A numeric vector of length 72 giving the survival times (in days).

Details

This dataset represents experimentally observed survival durations of guinea pigs infected with virulent tubercle bacilli. Survival analysis and lifetime modeling studies commonly use this dataset as an example for illustrating various statistical methodologies.

Value

An object of class `"numeric"`.

The vector consists of 72 observed survival times (in days), each corresponding to a single guinea pig experimentally infected with virulent tubercle bacilli. Each value represents the number of days from infection to death or end of observation. The dataset is commonly used in survival analysis and lifetime modeling to illustrate time-to-event data, hazard behavior, and comparative statistical methods.

References

Bjerkedal, T. (1960). *Acquisition of Resistance in Guinea Pigs Infected with Different Doses of Virulent Tubercle Bacilli*. American Journal of Hygiene, 72(1), 130–148.

Examples

```
data(survtimes)

# Basic summary
summary(survtimes)

# Plotting a simple histogram of survival times
hist(
  survtimes,
  main = "Survival Times of Guinea Pigs",
  xlab = "Days",
  col = "lightgray",
  border = "white"
)
```

---

waiting *Waiting Times of 100 Bank Customers*

Description

This dataset contains the waiting times (in minutes) of 100 bank customers, as originally analyzed in Ghitany, Atieh, and Nadarajah (2008) in their study on the Lindley distribution.

Usage

```
waiting
```

Format

A numeric vector of length 100 containing waiting times in minutes.

Details

These data were used to illustrate applications of the Lindley distribution in modeling waiting times. The dataset has been cited widely in reliability and lifetime distribution literature.

Value

An object of class `"numeric"`.

The vector consists of 100 observed waiting times (in minutes), each corresponding to a single bank customer. Each value represents the amount of time a customer waited before receiving service. The dataset is commonly used in reliability analysis and applied probability to illustrate lifetime and waiting-time distributions, particularly the Lindley distribution.

References

Ghitany, M. E., Atieh, B., & Nadarajah, S. (2008). Lindley distribution and its application. *Mathematics and Computers in Simulation*, 78, 493–506.

Examples

```
summary(waiting)

hist(
  waiting,
  main = "Histogram of Waiting Times",
  xlab = "Minutes"
)
```

---

windshield *Service Times of Aircraft Windshields*

Description

The `windshield` data set contains the service times (in years) of 63 aircraft windshields. These data have been widely used in the reliability literature, particularly for illustrating Weibull and related lifetime models.

Usage

```
data(windshield)
```

Format

A numeric vector of length 63 giving the service times of aircraft windshields.

Details

This dataset has been extensively analyzed in the context of reliability modeling, including Weibull models, compound lifetime models, and extended distributions such as the Weibull–Lomax distribution. The observations represent the time-to-failure of protective aircraft windshields and serve as a benchmark for demonstrating statistical methods for reliability and survival analysis.

Value

An object of class `"numeric"`.

The vector consists of 63 observed service times (in years), each corresponding to a single aircraft windshield. Each value represents the time elapsed from installation until failure or replacement of a windshield. The dataset is commonly used in reliability engineering and survival analysis to model time-to-failure behavior, study hazard rate shapes, and illustrate Weibull and extended lifetime distributions.

References

Murthy, D. N. P., Xie, M., & Jiang, R. (2004). *Weibull Models*. Wiley.

Blischke, W. R., & Murthy, D. N. P. (2000). *Reliability: Modeling, Prediction, and Optimization*. Wiley, New York.

Examples

```
data(windshield)

# Basic summary of the dataset
summary(windshield)

# Histogram of service times
hist(
  windshield,
  main = "Service Times of Aircraft Windshields",
  xlab = "Service Time (years)",
  col = "lightgray",
  border = "white"
)
```

Index