

# Package ‘RelDists’

May 7, 2026

**Title** Estimation for some Reliability Distributions

**Version** 1.0.1

**Description** Parameters estimation and linear regression models for Reliability distributions families reviewed by Almalki & Nadarajah (2014) <[doi:10.1016/j.res.2013.11.010](https://doi.org/10.1016/j.res.2013.11.010)> using Generalized Additive Models for Location, Scale and Shape, GAMLSS by Rigby & Stasinopoulos (2005) <[doi:10.1111/j.1467-9876.2005.00510.x](https://doi.org/10.1111/j.1467-9876.2005.00510.x)>.

**Depends** R (>= 3.5.0), survival, EstimationTools (>= 4.0.0)

**License** GPL-3

**URL** <https://fhernanb.github.io/RelDists/>

**BugReports** <https://github.com/fhernanb/RelDists/issues>

**Encoding** UTF-8

**RdMacros** Rdpack

**Imports** gamlss, gamlss.dist, Rdpack, zipfR, BBmisc, lamW, VGAM

**LazyData** true

**Suggests** knitr, rmarkdown, viridis, autoimage, gamlss.cens, V8, alr4

**RoxygenNote** 7.3.3

**VignetteBuilder** knitr

**NeedsCompilation** no

**Author** Freddy Hernandez-Barajas [aut, cre] (ORCID: <<https://orcid.org/0000-0001-7459-3329>>),  
Olga Usuga [aut] (ORCID: <<https://orcid.org/0000-0003-3062-1820>>),  
Carmen Patino [aut],  
Jaime Mosquera [aut] (ORCID: <<https://orcid.org/0000-0002-1684-4756>>)

**Maintainer** Freddy Hernandez-Barajas <[fhernanb@unal.edu.co](mailto:fhernanb@unal.edu.co)>

**Repository** CRAN

**Date/Publication** 2026-02-24 09:30:26 UTC

## Contents

AddW . . . . .	4
BGE . . . . .	5
BS . . . . .	7
BS2 . . . . .	10
BS3 . . . . .	12
CJ2 . . . . .	14
CS2e . . . . .	16
dAddW . . . . .	17
dBGE . . . . .	19
dBS . . . . .	21
dBS2 . . . . .	25
dBS3 . . . . .	28
dCJ2 . . . . .	30
dCS2e . . . . .	33
dEEG . . . . .	35
dEGG . . . . .	37
dEMWEx . . . . .	39
dEOFNH . . . . .	41
dEW . . . . .	43
dEXL . . . . .	44
dExW . . . . .	48
dExWALD . . . . .	50
dFWE . . . . .	52
dGammaW . . . . .	54
dGGD . . . . .	56
dGIW . . . . .	58
dGMW . . . . .	59
dGWF . . . . .	61
dIW . . . . .	63
dKumIW . . . . .	65
dLIN . . . . .	67
dLW . . . . .	69
dMOEIW . . . . .	71
dMOEW . . . . .	73
dMOK . . . . .	74
dMW . . . . .	76
dNEE . . . . .	78
dOW . . . . .	81
dPL . . . . .	83
dQXGP . . . . .	84
dRNMW . . . . .	86
dRW . . . . .	89
dSZMW . . . . .	91
dWALD . . . . .	92
dWG . . . . .	95
dWGEE . . . . .	97

dWP . . . . . 98

EEG . . . . . 100

EGG . . . . . 102

EMWEx . . . . . 104

EOFNH . . . . . 105

equipment . . . . . 107

EW . . . . . 108

EXL . . . . . 109

ExW . . . . . 112

ExWALD . . . . . 113

FWE . . . . . 115

GammaW . . . . . 117

GGD . . . . . 118

GIW . . . . . 120

GMW . . . . . 122

initValuesOW . . . . . 123

IW . . . . . 125

KumIW . . . . . 126

LIN . . . . . 128

LW . . . . . 129

mice . . . . . 131

MOEIW . . . . . 131

MOEW . . . . . 133

MOK . . . . . 135

MW . . . . . 136

myOW\_region . . . . . 138

NEE . . . . . 139

OW . . . . . 143

param.startOW . . . . . 144

PL . . . . . 146

QXGP . . . . . 147

RW . . . . . 149

summary.initValOW . . . . . 150

SZMW . . . . . 151

WALD . . . . . 153

WG . . . . . 154

WGEE . . . . . 156

WP . . . . . 157

AddW

*The Additive Weibull family***Description**

The Additive Weibull distribution

**Usage**

```
AddW(mu.link = "log", sigma.link = "log", nu.link = "log", tau.link = "log")
```

**Arguments**

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the mu parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the sigma.
<code>nu.link</code>	defines the <code>nu.link</code> , with "log" link as the default for the nu parameter.
<code>tau.link</code>	defines the <code>tau.link</code> , with "log" link as the default for the tau parameter.

**Details**

Additive Weibull distribution with parameters mu, sigma, nu and tau has density given by

$$f(x) = (\mu\nu x^{\nu-1} + \sigma\tau x^{\tau-1}) \exp(-\mu x^{\nu} - \sigma x^{\tau}),$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a AddW distribution in the `gamlss()` function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Almalki, S. J. (2018). A reduced new modified Weibull distribution. *Communications in Statistics-Theory and Methods*, 47(10), 2297-2313.

Xie, M., & Lai, C. D. (1996). Reliability analysis using an additive Weibull model with bathtub-shaped failure rate function. *Reliability Engineering & System Safety*, 52(1), 87-93.

**See Also**

[dAddW](#)

**Examples**

```

# Example 1
# Generating some random values with
# known mu, sigma, nu and tau
# Will not be run this example because high number is cycles
# is needed in order to get good estimates
## Not run:
y <- rAddW(n=100, mu=1.5, sigma=0.2, nu=3, tau=0.8)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, tau.fo=~1, family='AddW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma, nu and tau
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))
exp(coef(mod, what='tau'))

## End(Not run)

# Example 2
# Generating random values under some model
# Will not be run this example because high number is cycles
# is needed in order to get good estimates
## Not run:
n <- 200
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(1.67 + -3 * x1)
sigma <- exp(0.69 - 2 * x2)
nu <- 3
tau <- 0.8
x <- rAddW(n=n, mu, sigma, nu, tau)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, tau.fo=~1, family=AddW,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))
exp(coef(mod, what="tau"))

## End(Not run)

```

**Description**

The Beta Generalized Exponentiated family

**Usage**

```
BGE(mu.link = "log", sigma.link = "log", nu.link = "log", tau.link = "log")
```

**Arguments**

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the mu parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the sigma.
<code>nu.link</code>	defines the <code>nu.link</code> , with "log" link as the default for the nu parameter.
<code>tau.link</code>	defines the <code>tau.link</code> , with "log" link as the default for the tau parameter.

**Details**

The Beta Generalized Exponentiated distribution with parameters `mu`, `sigma`, `nu` and `tau` has density given by

$$f(x) = \frac{\nu\tau}{B(\mu,\sigma)} \exp(-\nu x)(1 - \exp(-\nu x))^{\tau\mu-1}(1 - (1 - \exp(-\nu x))^{\tau})^{\sigma-1},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu > 0$  and  $\tau > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a BGE distribution in the `gamlss()` function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

- Almalki, S. J. (2018). A reduced new modified Weibull distribution. *Communications in Statistics-Theory and Methods*, 47(10), 2297-2313.
- Barreto-Souza, W., Santos, A. H., & Cordeiro, G. M. (2010). The beta generalized exponential distribution. *Journal of statistical Computation and Simulation*, 80(2), 159-172.

**See Also**

[dBGE](#)

**Examples**

```
# Generating some random values with
# known mu, sigma, nu and tau
y <- rBGE(n=100, mu = 1.5, sigma =1.7, nu=1, tau=1)

# Fitting the model
require(gamlss)
```

```

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, tau.fo=~1, family=BGE,
             control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma, nu and tau
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))
exp(coef(mod, what='tau'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(0.5 - x1)
sigma <- exp(0.8 - x2)
nu <- 1
tau <- 1
x <- rBGE(n=n, mu, sigma, nu, tau)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, tau.fo=~1, family=BGE,
             control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))
exp(coef(mod, what="tau"))

```

---

BS

*The Birnbaum-Saunders family*


---

## Description

The function `BS()` defines The Birnbaum-Saunders, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

## Usage

```
BS(mu.link = "log", sigma.link = "log")
```

## Arguments

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the mu parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the sigma.

## Details

The Birnbaum-Saunders with parameters  $\mu$  and  $\sigma$  has density given by

$$f(x|\mu, \sigma) = \frac{x^{-3/2}(x+\mu)}{2\sigma\sqrt{2\pi\mu}} \exp\left(\frac{-1}{2\sigma^2}\left(\frac{x}{\mu} + \frac{\mu}{x} - 2\right)\right)$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$ . In this parameterization  $\mu$  is the median of  $X$ ,  $E(X) = \mu(1 + \sigma^2/2)$  and  $Var(X) = (\mu\sigma)^2(1 + 5\sigma^2/4)$ . The functions proposed here corresponds to the functions created by Roquim et al. (2021) with minor modifications to obtain correct log-likelihoods and random samples.

## Value

Returns a `gamlss.family` object which can be used to fit a BS distribution in the `gamlss()` function.

## References

Birnbaum, Z.W. and Saunders, S.C. (1969a). A new family of life distributions. *J. Appl. Prob.*, 6, 319–327.

Roquim, F. V., Ramires, T. G., Nakamura, L. R., Righetto, A. J., Lima, R. R., & Gomes, R. A. (2021). Building flexible regression models: including the Birnbaum-Saunders distribution in the `gamlss` package. *Semina: Ciências Exatas e Tecnológicas*, 42(2), 163-168.

## See Also

[dBS](#)

## Examples

```
# Example 1
# Generating some random values with
# known mu and sigma
y <- rBS(n=100, mu=0.75, sigma=1.3)

# Fitting the model
require(gamlss)
mod1 <- gamlss(y~1, sigma.fo=~1, family=BS)

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod1, what="mu"))
exp(coef(mod1, what="sigma"))

# Example 2
# Generating random values for a regression model

# A function to simulate a data set with Y as BS
gendat <- function(n) {
  x1 <- runif(n)
  x2 <- runif(n)
  mu <- exp(1.45 - 3 * x1)
  sigma <- exp(2 - 1.5 * x2)
```

```

y <- rBS(n=n, mu=mu, sigma=sigma)
data.frame(y=y, x1=x1, x2=x2)
}

set.seed(123)
dat <- gendat(n=300)

mod2 <- gamlss(y~x1, sigma.fo=~x2,
              family=BS, data=dat)

summary(mod2)

# Example 3
# Fatigue life (T) measures in cycles ( $\times 10^{-3}$ ) of n 101
# aluminum coupons (specimens) of type 6061-T6.
# Taken from Leiva et al. (2006) page 37.
# https://journal.r-project.org/articles/RN-2006-033/RN-2006-033.pdf

y <- c(70, 90, 96, 97, 99, 100, 103, 104,
      104, 105, 107, 108, 108, 108, 109, 109,
      112, 112, 113, 114, 114, 114, 116, 119,
      120, 120, 120, 121, 121, 123, 124, 124,
      124, 124, 124, 128, 128, 129, 129, 130,
      130, 130, 131, 131, 131, 131, 131, 132,
      132, 132, 133, 134, 134, 134, 134, 134,
      136, 136, 137, 138, 138, 138, 139, 139,
      141, 141, 142, 142, 142, 142, 142, 142,
      144, 144, 145, 146, 148, 148, 149, 151,
      151, 152, 155, 156, 157, 157, 157, 157,
      158, 159, 162, 163, 163, 164, 166, 166,
      168, 170, 174, 196, 212)

mod3 <- gamlss(y~1, sigma.fo=~1, family=BS)

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod3, what="mu"))
exp(coef(mod3, what="sigma"))

# Example 4
# Aggregate payments by the insurer
# in thousand Skr (Swedish currency).
# Taken from Balakrishnan and Kundu (2019) page 65.
# https://onlinelibrary.wiley.com/doi/abs/10.1002/asmb.2348

y <- c(5014, 5855, 6486, 6540, 6656, 6656, 7212, 7541, 7558,
      7797, 8546, 9345, 11762, 12478, 13624, 14451,
      14940, 14963, 15092, 16203, 16229, 16730, 18027,
      18343, 19365, 21782, 24248, 29069, 34267, 38993)

y <- y/10000

mod4 <- gamlss(y~1, sigma.fo=~1, family=BS)

```

```
# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod4, what="mu"))
exp(coef(mod4, what="sigma"))
```

---

 BS2

---

*The Birnbaum-Saunders family - Santos-Neto et al. (2014)*


---

### Description

The function `BS2()` defines The Birnbaum-Saunders, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

### Usage

```
BS2(mu.link = "log", sigma.link = "log")
```

### Arguments

`mu.link` defines the `mu.link`, with "log" link as the default for the mu parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the sigma.

### Details

The Birnbaum-Saunders with parameters `mu` and `sigma` has density given by

$$f(x|\mu, \sigma) = \frac{\exp(\sigma/2)\sqrt{\sigma+1}}{4\sqrt{\pi}\mu x^{3/2}} \left[ x + \frac{\mu\sigma}{\sigma+1} \right] \exp\left(\frac{-\sigma}{4} \left( \frac{x(\sigma+1)}{\mu\sigma} + \frac{\mu\sigma}{x(\sigma+1)} \right)\right)$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$ . In this parameterization  $E(X) = \mu$  and  $Var(X) = \frac{\mu^2(2\sigma+5)}{(\sigma+1)^2}$ .

### Value

Returns a `gamlss.family` object which can be used to fit a BS2 distribution in the `gamlss()` function.

### References

Santos-Neto, M., Cysneiros, F. J. A., Leiva, V., & Barros, M. (2014). A reparameterized Birnbaum–Saunders distribution and its moments, estimation and applications. *REVSTAT-Statistical Journal*, 12(3), 247-272.

### See Also

[dBS2](#).

**Examples**

```

# Example 1
# Generating some random values with
# known mu and sigma
y <- rBS2(n=50, mu=5, sigma=3)

# Fitting the model
require(gamlss)
mod1 <- gamlss(y~1, sigma.fo=~1, family=BS2)

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod1, what="mu"))
exp(coef(mod1, what="sigma"))

# Example 2
# Generating random values for a regression model

# A function to simulate a data set with  $Y \sim BS2$ 
gendat <- function(n) {
  x1 <- runif(n)
  x2 <- runif(n)
  mu <- exp(1.45 - 3 * x1)
  sigma <- exp(2 - 1.5 * x2)
  y <- rBS2(n=n, mu=mu, sigma=sigma)
  data.frame(y=y, x1=x1, x2=x2)
}

set.seed(123)
dat <- gendat(n=100)

mod2 <- gamlss(y~x1, sigma.fo=~x2,
              family=BS2, data=dat)

summary(mod2)

# Example 3
# Household expenditures for food in the United States (US) expressed
# in thousands of US dollars (M$)
# Santos-Neto et al. (2014) page 266.

y <- c(15.998, 16.652, 21.741, 7.431, 10.481, 13.548, 23.256, 17.976,
      14.161, 8.825, 14.184, 19.604, 13.728, 21.141, 17.446, 9.629,
      14.005, 9.160, 18.831, 7.641, 13.882, 9.670, 21.604, 10.866,
      28.980, 10.882, 18.561, 11.629, 18.067, 14.539, 19.192, 25.918,
      28.833, 15.869, 14.910, 9.550, 23.066, 14.751)

mod3 <- gamlss(y~1, sigma.fo=~1, family=BS2)

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod3, what="mu"))

```

```

exp(coef(mod3, what="sigma"))

# Example 4
# lifetimes of 6061-T6 aluminum coupons expressed in cycles (×10-3)
# at a maximum stress level of 3.1 psi (×104), until the failure to occur.
# Santos-Neto et al. (2014) page 267.

y <- c(70, 90, 96, 97, 99, 100, 103, 104, 104, 105, 107, 108, 108, 108, 109,
      109, 112, 112, 113, 114, 114, 114, 116, 119, 120, 120, 120, 121, 121,
      123, 124, 124, 124, 124, 124, 128, 128, 129, 129, 130, 130, 130, 131,
      131, 131, 131, 131, 132, 132, 132, 133, 134, 134, 134, 134, 134, 136,
      136, 137, 138, 138, 138, 139, 139, 141, 141, 142, 142, 142, 142, 142,
      142, 144, 144, 145, 146, 148, 148, 149, 151, 151, 152, 155, 156, 157,
      157, 157, 157, 158, 159, 162, 163, 163, 164, 166, 166, 168, 170, 174,
      196, 212)

mod4 <- gamlss(y~1, sigma.fo=~1, family=BS2)

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod4, what="mu"))
exp(coef(mod4, what="sigma"))

```

## Description

The function `BS3()` defines The Birnbaum-Saunders, a two parameter distribution, for a `gamlss` family object to be used in GAMLSS fitting using the function `gamlss()`.

## Usage

```
BS3(mu.link = "log", sigma.link = "logit")
```

## Arguments

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "logit" link as the default for the `sigma`.

## Details

The Birnbaum-Saunders with parameters `mu` and `sigma` has density given by

$$f(x|\mu, \sigma) = \frac{(1-\sigma)y+\mu}{2\sqrt{2\pi\mu\sigma(1-\sigma)}y^{3/2}} \exp\left[\frac{-1}{2\sigma}\left(\frac{(1-\sigma)y}{\mu} + \frac{\mu}{(1-\sigma)y} - 2\right)\right]$$

for  $x > 0$ ,  $\mu > 0$  and  $0 < \sigma < 1$ . In this parameterization  $Mode(X) = \mu$  and  $Var(X) = (\mu\sigma)^2(1 + 5\sigma^2/4)$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a BS3 distribution in the `gamlss()` function.

**References**

Bourguignon, M., & Gallardo, D. I. (2022). A new look at the Birnbaum–Saunders regression model. *Applied Stochastic Models in Business and Industry*, 38(6), 935-951.

**See Also**

[dBS3](#).

**Examples**

```
# Example 1
# Generating some random values with
# known mu and sigma
y <- rBS3(n=50, mu=2, sigma=0.2)

# Fitting the model
require(gamlss)
mod1 <- gamlss(y~1, sigma.fo=~1, family=BS3)

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod1, what="mu"))
exp(coef(mod1, what="sigma"))

# Example 2
# Generating random values for a regression model

# A function to simulate a data set with  $Y \sim BS3$ 
## Not run:
gendat <- function(n) {
  x1 <- runif(n)
  x2 <- runif(n)
  mu <- exp(1.45 - 3 * x1)
  inv_logit <- function(x) 1 / (1 + exp(-x))
  sigma <- inv_logit(2 - 1.5 * x2)
  y <- rBS3(n=n, mu=mu, sigma=sigma)
  data.frame(y=y, x1=x1, x2=x2)
}

set.seed(1234)
dat <- gendat(n=100)

mod2 <- gamlss(y~x1, sigma.fo=~x2,
              family=BS3, data=dat,
              control=gamlss.control(n.cyc=100))

summary(mod2)
```

```
## End(Not run)

# Example 3
# The response variable is the ratio between the average
# rent per acre planted with alfalfa and the corresponding
# average rent for other agricultural uses. The density of
# dairy cows (X2, number per square mile) is the explanatory variable.
library(alr4)
data("landrent")

landrent$ratio <- landrent$Y / landrent$X1

with(landrent, plot(x=X2, y=ratio))

mod3 <- gamlss(ratio~X2, sigma.fo=~X2,
               data=landrent, family=BS3)

summary(mod3)
logLik(mod3)
```

---

CJ2

*The two-parameter Chris-Jerry distribution family*


---

### Description

The function `CJ2()` defines The two-parameter Chris-Jerry distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

### Usage

```
CJ2(mu.link = "log", sigma.link = "log")
```

### Arguments

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.

### Details

The two-parameter Chris-Jerry distribution with parameters `mu` and `sigma` has density given by

$$f(x; \sigma, \mu) = \frac{\mu^2}{\sigma\mu+2} (\sigma + \mu x^2) e^{-\mu x}; \quad x > 0, \quad \mu > 0, \quad \sigma > 0$$

Note: In this implementation we changed the original parameters  $\theta$  for  $\mu$  and  $\lambda$  for  $\sigma$  we did it to implement this distribution within `gamlss` framework.

### Value

Returns a `gamlss.family` object which can be used to fit a CJ2 distribution in the `gamlss()` function.

**Author(s)**

Manuel Gutierrez Tangarife, <mgutierrez@unal.edu.co>

**References**

Chinedu, Eberchukwu Q., et al. "New lifetime distribution with applications to single acceptance sampling plan and scenarios of increasing hazard rates" *Symmetry* 15.10 (2023): 188.

**See Also**

[dCJ2](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu and sigma
y <- rCJ2(n=500, mu=1, sigma=1.5)

# Fitting the model
require(gamlss)

mod1 <- gamlss(y~1, sigma.fo=~1, family=CJ2,
               control=gamlss.control(n.cyc=5000, trace=TRUE))

# Extracting the fitted values for mu, sigma
# using the inverse link function
exp(coef(mod1, what="mu"))
exp(coef(mod1, what="sigma"))

# Example 2
# Generating random values under some model
gendat <- function(n) {
  x1 <- runif(n, min=0, max=5)
  x2 <- runif(n, min=0, max=5)
  mu <- exp(-0.2 + 1.5 * x1)
  sigma <- exp(1 - 0.7 * x2)
  y <- rCJ2(n=n, mu, sigma)
  data.frame(y=y, x1=x1, x2=x2)
}

set.seed(123)
datos <- gendat(n=500)

mod2 <- gamlss(y~x1, sigma.fo=~x2, family=CJ2, data=datos,
               control=gamlss.control(n.cyc=5000, trace=TRUE))

summary(mod2)
```

**Description**

The Cosine Sine Exponential family

**Usage**

```
CS2e(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.  
`nu.link` defines the `nu.link`, with "log" link as the default for the `nu` parameter.

**Details**

The Cosine Sine Exponential distribution with parameters `mu`, `sigma` and `nu` has density given by

$$f(x) = \frac{\pi \sigma \mu \exp\left(\frac{-x}{\nu}\right)}{2\nu[(\mu \sin\left(\frac{\pi}{2} \exp\left(\frac{-x}{\nu}\right)\right) + \sigma \cos\left(\frac{\pi}{2} \exp\left(\frac{-x}{\nu}\right)\right)]^2},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a CS2e distribution in the `gamlss()` function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

Chesneau, C., Bakouch, H. S., & Hussain, T. (2019). A new class of probability distributions via cosine and sine functions with applications. *Communications in Statistics-Simulation and Computation*, 48(8), 2287-2300.

**See Also**

[dCS2e](#)

**Examples**

```

# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rCS2e(n=100, mu = 0.1, sigma =1, nu=0.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='CS2e',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.45, max=0.55)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(0.2 - x1)
sigma <- exp(0.8 - x2)
nu <- 0.5
x <- rCS2e(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, family=CS2e,
              control=gamlss.control(n.cyc=50000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))

```

---

dAddW

*The Additive Weibull distribution*


---

**Description**

Density, distribution function, quantile function, random generation and hazard function for the Additive Weibull distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$ .

**Usage**

```
dAddW(x, mu, sigma, nu, tau, log = FALSE)
```

```
pAddW(q, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)
```

```
qAddW(p, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)
```

```
rAddW(n, mu, sigma, nu, tau)
```

```
hAddW(x, mu, sigma, nu, tau)
```

### Arguments

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
nu	shape parameter.
tau	shape parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

Additive Weibull Distribution with parameters mu, sigma, nu and tau has density given by

$$f(x) = (\mu\nu x^{\nu-1} + \sigma\tau x^{\tau-1}) \exp(-\mu x^{\nu} - \sigma x^{\tau}),$$

for  $x > 0$ .

### Value

dAddW gives the density, pAddW gives the distribution function, qAddW gives the quantile function, rAddW generates random deviates and hAddW gives the hazard function.

### Author(s)

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

### References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. Reliability Engineering & System Safety, 124, 32-55.

Xie, M., & Lai, C. D. (1996). Reliability analysis using an additive Weibull model with bathtub-shaped failure rate function. Reliability Engineering & System Safety, 52(1), 87-93.

### See Also

[AddW](#)

**Examples**

```

old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dAddW(x, mu=1.5, sigma=0.5, nu=3, tau=0.8), from=0.0001, to=2,
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pAddW(x, mu=1.5, sigma=0.5, nu=3, tau=0.8),
      from=0.0001, to=2, col="red", las=1, ylab="F(x)")
curve(pAddW(x, mu=1.5, sigma=0.5, nu=3, tau=0.8, lower.tail=FALSE),
      from=0.0001, to=2, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qAddW(p, mu=1.5, sigma=0.2, nu=3, tau=0.8), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pAddW(x, mu=1.5, sigma=0.2, nu=3, tau=0.8),
      from=0, add=TRUE, col="red")

## The random function
hist(rAddW(n=10000, mu=1.5, sigma=0.2, nu=3, tau=0.8), freq=FALSE,
     xlab="x", las=1, main="")
curve(dAddW(x, mu=1.5, sigma=0.2, nu=3, tau=0.8),
      from=0.09, to=5, add=TRUE, col="red")

## The Hazard function
curve(hAddW(x, mu=1.5, sigma=0.2, nu=3, tau=0.8), from=0.001, to=1,
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters

```

---

 dBGE

*The Beta Generalized Exponentiated distribution*


---

**Description**

Density, distribution function, quantile function, random generation and hazard function for the Beta Generalized Exponentiated distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$ .

**Usage**

```

dBGE(x, mu, sigma, nu, tau, log = FALSE)

pBGE(q, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

qBGE(p, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

```

rBGE(n, mu, sigma, nu, tau)

hBGE(x, mu, sigma, nu, tau)

### Arguments

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
nu	parameter.
tau	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

The Beta Generalized Exponentiated Distribution with parameters mu, sigma, nu and tau has density given by

$$f(x) = \frac{\nu\tau}{B(\mu,\sigma)} \exp(-\nu x)(1 - \exp(-\nu x))^{\tau\mu-1}(1 - (1 - \exp(-\nu x))^{\tau})^{\sigma-1},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu > 0$  and  $\tau > 0$ .

### Value

dBGE gives the density, pBGE gives the distribution function, qBGE gives the quantile function, rBGE generates random deviates and hBGE gives the hazard function.

### Author(s)

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

### References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. Reliability Engineering & System Safety, 124, 32-55.

Barreto-Souza, W., Santos, A. H., & Cordeiro, G. M. (2010). The beta generalized exponential distribution. Journal of statistical Computation and Simulation, 80(2), 159-172.

### See Also

[BGE](#)

**Examples**

```

old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dBGE(x, mu = 1.5, sigma =1.7, nu=1, tau=1), from = 0, to = 3,
      col = "red", las = 1, ylab = "f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pBGE(x, mu = 1.5, sigma =1.7, nu=1, tau=1), from = 0, to = 6,
      ylim = c(0, 1), col = "red", las = 1, ylab = "F(x)")
curve(pBGE(x, mu = 1.5, sigma =1.7, nu=1, tau=1, lower.tail = FALSE),
      from = 0, to = 6, ylim = c(0, 1), col = "red", las = 1, ylab = "R(x)")

## The quantile function
p <- seq(from = 0, to = 0.99999, length.out = 100)
plot(x = qBGE(p = p, mu = 1.5, sigma =1.7, nu=1, tau=1), y = p,
     xlab = "Quantile", las = 1, ylab = "Probability")
curve(pBGE(x, mu = (1/4), sigma =1, nu=1, tau=2), from = 0, add = TRUE,
     col = "red")

## The random function
hist(rBGE(1000, mu = 1.5, sigma =1.7, nu=1, tau=1), freq = FALSE, xlab = "x",
     ylim = c(0, 1), las = 1, main = "")
curve(dBGE(x, mu = 1.5, sigma =1.7, nu=1, tau=1), from = 0, add = TRUE,
     col = "red", ylim = c(0, 0.5))

## The Hazard function(
par(mfrow=c(1,1))
curve(hBGE(x, mu = 0.9, sigma =0.5, nu=1, tau=1), from = 0, to = 2,
     col = "red", ylab = "Hazard function", las = 1)

par(old_par) # restore previous graphical parameters

```

**Description**

Density, distribution function, quantile function, random generation and hazard function for the Birnbaum-Saunders distribution with parameters  $\mu$  and  $\sigma$ .

**Usage**

```

dBS(x, mu = 1, sigma = 1, log = FALSE)

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

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

```

rBS(n, mu = 1, sigma = 1)

hBS(x, mu, sigma)

### Arguments

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

The Birnbaum-Saunders with parameters mu and sigma has density given by

$$f(x|\mu, \sigma) = \frac{x^{-3/2}(x+\mu)}{2\sigma\sqrt{2\pi\mu}} \exp\left(\frac{-1}{2\sigma^2}\left(\frac{x}{\mu} + \frac{\mu}{x} - 2\right)\right)$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$ . In this parameterization  $\mu$  is the median of  $X$ ,  $E(X) = \mu(1 + \sigma^2/2)$  and  $Var(X) = (\mu\sigma)^2(1 + 5\sigma^2/4)$ . The functions proposed here corresponds to the functions created by Roquim et al. (2021) with minor modifications to obtain correct log-likelihoods and random samples.

### Value

dBS gives the density, pBS gives the distribution function, qBS gives the quantile function, rBS generates random deviates and hBS gives the hazard function.

### References

Birnbaum, Z.W. and Saunders, S.C. (1969a). A new family of life distributions. J. Appl. Prob., 6, 319–327.

Roquim, F. V., Ramires, T. G., Nakamura, L. R., Righetto, A. J., Lima, R. R., & Gomes, R. A. (2021). Building flexible regression models: including the Birnbaum-Saunders distribution in the gamlss package. Semina: Ciências Exatas e Tecnológicas, 42(2), 163-168.

### See Also

[BS](#).

**Examples**

```
#Example 1
#Plotting the mass function for different parameter values
curve(dBS(x, mu=0.5, sigma=0.5),
      from=0.001, to=5,
      ylim=c(0, 2),
      col="royalblue1", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dBS(x, mu=1, sigma=0.5),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(dBS(x, mu=1.5, sigma=0.5),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=0.5, sigma=0.5",
                           "mu=1.0, sigma=0.5",
                           "mu=1.5, sigma=0.5"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.6)

curve(dBS(x, mu=0.5, sigma=1),
      from=0.001, to=5,
      ylim=c(0, 1.5),
      col="royalblue1", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dBS(x, mu=1, sigma=1),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(dBS(x, mu=1.5, sigma=1),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=0.5, sigma=1",
                           "mu=1.0, sigma=1",
                           "mu=1.5, sigma=1"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.6)

curve(dBS(x, mu=0.5, sigma=1.5),
      from=0.001, to=8,
      ylim=c(0, 2),
      col="royalblue1", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dBS(x, mu=1, sigma=1.5),
      col="tomato",
      lwd=2,
```

```

    add=TRUE)
curve(dBS(x, mu=1.5, sigma=1.5),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=0.5, sigma=1.5",
                            "mu=1.0, sigma=1.5",
                            "mu=1.5, sigma=1.5"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.6)

# Example 2
# Checking if the cumulative curves converge to 1
curve(pBS(x, mu=0.5, sigma=0.5),
      from=0.001, to=5,
      ylim=c(0, 1),
      col="royalblue1", lwd=2,
      main="Cumulative Distribution Function",
      xlab="x", ylab="f(x)")
curve(pBS(x, mu=1, sigma=0.5),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(pBS(x, mu=1.5, sigma=0.5),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("bottomright", legend=c("mu=0.5, sigma=0.5",
                              "mu=1.0, sigma=0.5",
                              "mu=1.5, sigma=0.5"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.5)
curve(pBS(x, mu=0.5, sigma=0.5, lower.tail=FALSE),
      from=0.001, to=5,
      ylim=c(0, 1),
      col="royalblue1", lwd=2,
      main="Cumulative Distribution Function",
      xlab="x", ylab="f(x)")
curve(pBS(x, mu=1, sigma=0.5, lower.tail=FALSE),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(pBS(x, mu=1.5, sigma=0.5, lower.tail=FALSE),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=0.5, sigma=0.5",
                            "mu=1.0, sigma=0.5",
                            "mu=1.5, sigma=0.5"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.5)

#example 3
## The quantile function
p <- seq(from=0, to=0.999, length.out=100)
plot(x=qBS(p, mu=2.3, sigma=1.7), y=p, xlab="Quantile",

```

```

    las=1, ylab="Probability", main="Quantile function ")
curve(pBS(x, mu=2.3, sigma=1.7),
      from=0, add=TRUE, col="tomato", lwd=2.5)

#some values
p <- c(0.25, 0.5, 0.75)
quantile <- qBS(p=p, mu=2.3, sigma=1.7)
for(i in quantile){
  print(integrate(dBS, lower=0, upper=i, mu=2.3, sigma=1.7))
}

#example 4
## The random function
x <- rBS(n=10000, mu=20, sigma=0.5)
hist(x, freq=FALSE)
curve(dBS(x, mu=20, sigma=0.5), from=0, to=100,
      add=TRUE, col="tomato", lwd=2)

#example 5
## The Hazard function
curve(hBS(x, mu=20, sigma=0.5), from=0.001, to=100,
      col="tomato", ylab="Hazard function", las=1)

```

**Description**

Density, distribution function, quantile function, random generation and hazard function for the Birnbaum-Saunders distribution with parameters  $\mu$  and  $\sigma$ .

**Usage**

```

dBS2(x, mu = 1, sigma = 1, log = FALSE)

pBS2(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)

qBS2(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)

rBS2(n, mu = 1, sigma = 1)

hBS2(x, mu, sigma)

```

**Arguments**

x, q	vector of quantiles.
mu	parameter.

sigma	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

The Birnbaum-Saunders with parameters  $\mu$  and  $\sigma$  has density given by

$$f(x) = \frac{\exp(\sigma/2)\sqrt{\sigma+1}}{4\sqrt{\pi}\mu x^{3/2}} \left[ x + \frac{\mu\sigma}{\sigma+1} \right] \exp\left(\frac{-\sigma}{4} \left( \frac{x(\sigma+1)}{\mu\sigma} + \frac{\mu\sigma}{x(\sigma+1)} \right)\right)$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$ . In this parameterization  $E(X) = \mu$  and  $Var(X) = (\mu\sigma)^2(1+5\sigma^2/4)$ . The functions proposed here corresponds to the parameterization proposed by Santos-Neto et al. (2014).

### Value

dBS2 gives the density, pBS2 gives the distribution function, qBS2 gives the quantile function, rBS2 generates random deviates and hBS2 gives the hazard function.

### References

Santos-Neto, M., Cysneiros, F. J. A., Leiva, V., & Barros, M. (2014). A reparameterized Birnbaum–Saunders distribution and its moments, estimation and applications. *REVSTAT-Statistical Journal*, 12(3), 247-272.

### See Also

[BS2](#).

### Examples

```
#Example 1
#Plotting the mass function for different parameter values
curve(dBS2(x, mu=1.0, sigma=100),
      from=0.001, to=5,
      ylim=c(0, 3),
      col="royalblue1", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dBS2(x, mu=1.5, sigma=100),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(dBS2(x, mu=2.0, sigma=100),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=1.0, sigma=100",
                            "mu=1.5, sigma=100",
```

```

                                "mu=2.0, sigma=100"),
                                col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.6)

curve(dBS2(x, mu=1, sigma=2),
      from=0.001, to=2,
      ylim=c(0, 1.1),
      col="royalblue1", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dBS2(x, mu=1, sigma=5),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(dBS2(x, mu=1, sigma=10),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=1, sigma=2",
                            "mu=1, sigma=5",
                            "mu=1, sigma=10"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.6)

# Example 2
# Checking if the cumulative curves converge to 1
curve(pBS2(x, mu=0.5, sigma=0.5),
      from=0.001, to=15,
      ylim=c(0, 1),
      col="royalblue1", lwd=2,
      main="Cumulative Distribution Function",
      xlab="x", ylab="f(x)")
curve(pBS2(x, mu=1, sigma=0.5),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(pBS2(x, mu=1.5, sigma=0.5),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("bottomright", legend=c("mu=0.5, sigma=0.5",
                              "mu=1.0, sigma=0.5",
                              "mu=1.5, sigma=0.5"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.5)

# Example 3
# The quantile function
p <- seq(from=0, to=0.999, length.out=100)
plot(x=qBS2(p, mu=2.3, sigma=1.7), y=p, xlab="Quantile",
     las=1, ylab="Probability", main="Quantile function ")
curve(pBS2(x, mu=2.3, sigma=1.7),
      from=0, add=TRUE, col="tomato", lwd=2.5)

```

```

# Example 4
# The random function
x <- rBS2(n=10000, mu=2.5, sigma=100)
hist(x, freq=FALSE)
curve(dBS2(x, mu=2.5, sigma=100), from=0, to=10,
      add=TRUE, col="tomato", lwd=2)

# Example 5
# The Hazard function
curve(hBS2(x, mu=20, sigma=0.5), from=0.001, to=100,
      col="tomato", ylab="Hazard function", las=1)

```

---

dBS3

*The Birnbaum-Saunders distribution - Bourguignon & Gallardo (2022)*

---

### Description

Density, distribution function, quantile function, random generation and hazard function for the Birnbaum-Saunders distribution with parameters  $\mu$  and  $\sigma$ .

### Usage

```

dBS3(x, mu = 1, sigma = 0.5, log = FALSE)

pBS3(q, mu = 1, sigma = 0.5, lower.tail = TRUE, log.p = FALSE)

qBS3(p, mu = 1, sigma = 0.5, lower.tail = TRUE, log.p = FALSE)

rBS3(n, mu = 1, sigma = 0.5)

hBS3(x, mu, sigma)

```

### Arguments

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

## Details

The Birnbaum-Saunders with parameters  $\mu$  and  $\sigma$  has density given by

$$f(x) = \frac{\exp(\sigma/2)\sqrt{\sigma+1}}{4\sqrt{\pi}\mu x^{3/2}} \left[ x + \frac{\mu\sigma}{\sigma+1} \right] \exp\left(\frac{-\sigma}{4} \left( \frac{x(\sigma+1)}{\mu\sigma} + \frac{\mu\sigma}{x(\sigma+1)} \right)\right)$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$ . In this parameterization  $E(X) = \mu$  and  $Var(X) = (\mu\sigma)^2(1+5\sigma^2/4)$ . The functions proposed here corresponds to the parameterization proposed by Santos-Neto et al. (2014).

## Value

dBS3 gives the density, pBS3 gives the distribution function, qBS3 gives the quantile function, rBS3 generates random deviates and hBS3 gives the hazard function.

## References

Bourguignon, M., & Gallardo, D. I. (2022). A new look at the Birnbaum–Saunders regression model. *Applied Stochastic Models in Business and Industry*, 38(6), 935-951.

## See Also

[BS3](#).

## Examples

```
# Example 1
# Plotting the mass function for different parameter values
curve(dBS3(x, mu=2, sigma=0.2),
      from=0.001, to=10,
      ylim=c(0, 0.4),
      col="royalblue1", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dBS3(x, mu=2, sigma=0.4),
      col="tomato",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=2, sigma=0.2",
                           "mu=2, sigma=0.4"),
      col=c("royalblue1", "tomato"), lwd=2, cex=0.6)

# Example 2
# Checking if the cumulative curves converge to 1
curve(pBS3(x, mu=2, sigma=0.2),
      from=0.00001, to=10,
      ylim=c(0, 1),
      col="royalblue1", lwd=2,
      main="Cumulative Distribution Function",
      xlab="x", ylab="f(x)")
curve(pBS3(x, mu=2, sigma=0.4),
      col="tomato",
      lwd=2,
```

```

      add=TRUE)
legend("bottomright", legend=c("mu=2, sigma=0.2",
                               "mu=2, sigma=0.4"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.5)

# Example 3
# The quantile function
p <- seq(from=0, to=0.999, length.out=100)
plot(x=qBS3(p, mu=2, sigma=0.2), y=p, xlab="Quantile",
     las=1, ylab="Probability", main="Quantile function ")
curve(pBS3(x, mu=2, sigma=0.2),
     from=0, add=TRUE, col="tomato", lwd=2.5)

# Example 4
# The random function
x <- rBS3(n=10000, mu=2, sigma=0.2)
hist(x, freq=FALSE)
curve(dBS3(x, mu=2, sigma=0.2), from=0, to=10,
     add=TRUE, col="tomato", lwd=2)

# Example 5
# The Hazard function
curve(hBS3(x, mu=2, sigma=0.2), from=0.001, to=4,
     col="tomato", ylab="Hazard function", las=1)

```

---

dCJ2

*The two-parameter Chris-Jerry distribution*


---

### Description

Density, distribution function, quantile function, random generation and hazard function for the two-parameter Chris-Jerry distribution with parameters  $\mu$  and  $\sigma$ .

### Usage

```

dCJ2(x, mu, sigma, log = FALSE)

pCJ2(q, mu, sigma, log.p = FALSE, lower.tail = TRUE)

qCJ2(p, mu, sigma, lower.tail = TRUE, log.p = FALSE)

rCJ2(n, mu, sigma)

hCJ2(x, mu, sigma, log = FALSE)

```

### Arguments

$x, q$                     vector of quantiles.

mu	parameter.
sigma	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

The two-parameter Chris-Jerry distribution with parameters mu and sigma has density given by

$$f(x; \sigma, \mu) = \frac{\mu^2}{\sigma\mu+2}(\sigma + \mu x^2)e^{-\mu x}; \quad x > 0, \quad \mu > 0, \quad \sigma > 0$$

Note: In this implementation we changed the original parameters  $\theta$  for  $\mu$  and  $\lambda$  for  $\sigma$ , we did it to implement this distribution within gamlss framework.

### Value

dCJ2 gives the density, pCJ2 gives the distribution function, qCJ2 gives the quantile function, rCJ2 generates random deviates and hCJ2 gives the hazard function.

### Author(s)

Manuel Gutierrez Tangarife, <mgutierrez@unal.edu.co>

### References

Chinedu, Eberechukwu Q., et al. "New lifetime distribution with applications to single acceptance sampling plan and scenarios of increasing hazard rates" *Symmetry* 15.10 (2023): 188.

### See Also

[CJ2](#)

### Examples

```
# Example 1
# Plotting the density function for different parameter values
curve(dCJ2(x, mu=3.5, sigma=0.01),
      from=0.0001, to=5,
      ylim=c(0, 1),
      col="red", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dCJ2(x, mu=2, sigma=0.05),
      col="green",
      lwd=2,
      add=TRUE)
curve(dCJ2(x, mu=1.5, sigma=0.01),
      col="blue",
      lwd=2,
```

```

    add=TRUE)
curve(dCJ2(x, mu=2.5, sigma=0.01),
      col="lightblue",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=3.5, sigma=0.01",
                            "mu=2, sigma=0.05",
                            "mu=1.5, sigma=0.01",
                            "mu=2.5, sigma=0.1"),
      col=c("red", "green", "blue", "lightblue"), lwd=2, cex=0.6)

# Example 2
# Checking if the cumulative curves converge to 1
curve(pCJ2(x, mu=2.7, sigma=0.1),
      from=0.0001, to=5,
      ylim=c(0, 1),
      col="red", lwd=2,
      main="Cumulative function",
      xlab="x", ylab="f(x)")
curve(pCJ2(x, mu=2.3, sigma=0.5),
      col="green",
      lwd=2,
      add=TRUE)
curve(pCJ2(x, mu=2.8, sigma=0.2),
      col="blue",
      lwd=2,
      add=TRUE)
curve(pCJ2(x, mu=3.8, sigma=0.3),
      col="lightblue",
      lwd=2,
      add=TRUE)
legend("bottomright", legend=c("mu=2.75, sigma=0.1",
                              "mu=2.3, sigma=0.5",
                              "mu=2.8, sigma=0.2",
                              "mu=3.8, sigma=0.3"),
      col=c("red", "green", "blue", "lightblue"), lwd=2, cex=0.6)

# Example 3
# Checking the quantile function
p <- seq(from=0.0001, to=0.99999, length.out=100)
plot(x=qCJ2(p, mu=2.3, sigma=1.7), y=p, xlab="Quantile",
     las=1, ylab="Probability", main="Quantile function ")
curve(pCJ2(x, mu=2.3, sigma=1.7),
      from=0.0001, add=TRUE, col="red", lwd=2.5)

# Example 4
# Comparing the random generator output with
# the theoretical probabilities
x <- rCJ2(n=10000, mu=1.5, sigma=2.5)
hist(x, freq=FALSE)
curve(dCJ2(x, mu=1.5, sigma=2.5), from=0.001, to=8,
      add=TRUE, col="tomato", lwd=2)

```

```

# Example 5
# The Hazard function
curve(hCJ2(x, mu=0.85, sigma=0.15),
      from=0.0001, to=5,
      ylim=c(0, 1),
      col="red", lwd=2,
      main="Hazard function",
      xlab="x", ylab="f(x)")
curve(hCJ2(x, mu=1, sigma=0.05),
      col="green",
      lwd=2,
      add=TRUE)
curve(hCJ2(x, mu=0.9, sigma=0.1),
      col="blue",
      lwd=2,
      add=TRUE)
curve(hCJ2(x, mu=1.15, sigma=0.1),
      col="lightblue",
      lwd=2,
      add=TRUE)
legend("bottomright", legend=c("mu=0.85, sigma=0.15",
                              "mu=1, sigma=0.05",
                              "mu=0.9, sigma=0.1",
                              "mu=1.15, sigma=0.1"),
      col=c("red", "green", "blue", "lightblue"), lwd=2, cex=0.5)

```

---

dCS2e

*The Cosine Sine Exponential distribution*


---

### Description

Density, distribution function, quantile function, random generation and hazard function for the Cosine Sine Exponential distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

### Usage

```

dCS2e(x, mu, sigma, nu, log = FALSE)

pCS2e(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qCS2e(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rCS2e(n, mu, sigma, nu)

hCS2e(x, mu, sigma, nu)

```

### Arguments

$x, q$                 vector of quantiles.

mu	parameter.
sigma	parameter.
nu	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

The Cosine Sine Exponential Distribution with parameters mu, sigma and nu has density given by

$$f(x) = \frac{\pi \sigma \mu \exp\left(\frac{-x}{\nu}\right)}{2\nu[(\mu \sin\left(\frac{\pi}{2} \exp\left(\frac{-x}{\nu}\right)\right) + \sigma \cos\left(\frac{\pi}{2} \exp\left(\frac{-x}{\nu}\right)\right)]^2},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

### Value

dCS2e gives the density, pCS2e gives the distribution function, qCS2e gives the quantile function, rCS2e generates random deviates and hCS2e gives the hazard function.

### Author(s)

Juan Pablo Ramirez

### References

Chesneau, C., Bakouch, H. S., & Hussain, T. (2019). A new class of probability distributions via cosine and sine functions with applications. *Communications in Statistics-Simulation and Computation*, 48(8), 2287-2300.

### See Also

[CS2e](#)

### Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
par(mfrow=c(1,1))
curve(dCS2e(x, mu=1, sigma=0.1, nu =0.1), from=0, to=1,
      ylim=c(0, 3), col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pCS2e(x, mu=1, sigma=0.1, nu =0.1),
      from=0, to=1, col="red", las=1, ylab="F(x)")
curve(pCS2e(x, mu=1, sigma=0.1, nu =0.1, lower.tail=FALSE),
      from=0, to=1, col="red", las=1, ylab="R(x)")
```

```

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qCS2e(p, mu=0.1, sigma=1, nu=0.1), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pCS2e(x, mu=0.1, sigma=1, nu=0.1), from=0, add=TRUE, col="red")

## The random function
hist(rCS2e(n=10000, mu=0.1, sigma=1, nu=0.1), freq=FALSE,
     xlab="x", las=1, main="")
curve(dCS2e(x, mu=0.1, sigma=1, nu=0.1), from=0, add=TRUE, col="red")

## The Hazard function
par(mfrow=c(1,1))
curve(hCS2e(x, mu=1, sigma=0.1, nu =0.1), from=0, to=1, ylim=c(0, 10),
     col=2, ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters

```

---

dEEG

*The Extended Exponential Geometric distribution*


---

## Description

Density, distribution function, quantile function, random generation and hazard function for the Extended Exponential Geometric distribution with parameters  $\mu$  and  $\sigma$ .

## Usage

```

dEEG(x, mu, sigma, log = FALSE)

pEEG(q, mu, sigma, lower.tail = TRUE, log.p = FALSE)

qEEG(p, mu, sigma, lower.tail = TRUE, log.p = FALSE)

rEEG(n, mu, sigma)

hEEG(x, mu, sigma)

```

## Arguments

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

### Details

The Extended Exponential Geometric distribution with parameters  $\mu$ , and  $\sigma$  has density given by

$$f(x) = \mu\sigma \exp(-\mu x)(1 - (1 - \sigma) \exp(-\mu x))^{-2},$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$ .

### Value

dEEG gives the density, pEEG gives the distribution function, qEEG gives the quantile function, rEEG generates random deviates and hEEG gives the hazard function.

### Author(s)

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

### References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. Reliability Engineering & System Safety, 124, 32-55.

Adamidis, K., Dimitrakopoulou, T., & Loukas, S. (2005). On an extension of the exponential-geometric distribution. Statistics & probability letters, 73(3), 259-269.

### See Also

[EEG](#)

### Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
par(mfrow=c(1,1))
curve(dEEG(x, mu = 1, sigma =3), from = 0, to = 10,
      col = "red", las = 1, ylab = "f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pEEG(x, mu = 1, sigma =3), from = 0, to = 10,
      ylim = c(0, 1), col = "red", las = 1, ylab = "F(x)")
curve(pEEG(x, mu = 1, sigma =3, lower.tail = FALSE),
      from = 0, to = 6, ylim = c(0, 1), col = "red", las = 1, ylab = "R(x)")

## The quantile function
p <- seq(from = 0, to = 0.99999, length.out = 100)
plot(x = qEEG(p = p, mu = 1, sigma =0.5), y = p,
     xlab = "Quantile", las = 1, ylab = "Probability")
curve(pEEG(x, mu = 1, sigma =0.5), from = 0, add = TRUE,
     col = "red")

## The random function
```

```

hist(rEEG(1000, mu = 1, sigma =1), freq = FALSE, xlab = "x",
     ylim = c(0, 0.9), las = 1, main = "")
curve(dEEG(x, mu = 1, sigma =1), from = 0, add = TRUE,
      col = "red", ylim = c(0, 0.8))

## The Hazard function
par(mfrow=c(1,1))
curve(hEEG(x, mu = 1, sigma =0.5), from = 0, to = 2,
      col = "red", ylab = "Hazard function", las = 1)

par(old_par) # restore previous graphical parameters

```

---

dEGG

*The four parameter Exponentiated Generalized Gamma distribution*


---

### Description

Density, distribution function, quantile function, random generation and hazard function for the four parameter Exponentiated Generalized Gamma distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$ .

### Usage

```

dEGG(x, mu, sigma, nu, tau, log = FALSE)

pEGG(q, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

qEGG(p, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

rEGG(n, mu, sigma, nu, tau)

hEGG(x, mu, sigma, nu, tau)

```

### Arguments

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>nu</code>	parameter.
<code>tau</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

### Details

Four-Parameter Exponentiated Generalized Gamma distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$  has density given by

$$f(x) = \frac{\nu\sigma}{\mu\Gamma(\tau)} \left(\frac{x}{\mu}\right)^{\sigma\tau-1} \exp\left\{-\left(\frac{x}{\mu}\right)^\sigma\right\} \left\{\gamma_1\left(\tau, \left(\frac{x}{\mu}\right)^\sigma\right)\right\}^{\nu-1},$$

for  $x > 0$ .

### Value

dEGG gives the density, pEGG gives the distribution function, qEGG gives the quantile function, rEGG generates random deviates and hEGG gives the hazard function.

### Author(s)

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

### References

- Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.
- Cordeiro, G. M., Ortega, E. M., & Silva, G. O. (2011). The exponentiated generalized gamma distribution with application to lifetime data. *Journal of statistical computation and simulation*, 81(7), 827-842.

### See Also

[EGG](#)

### Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dEGG(x, mu=0.1, sigma=0.8, nu=10, tau=1.5), from=0.000001, to=1.5, ylim=c(0, 2.5),
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pEGG(x, mu=0.1, sigma=0.8, nu=10, tau=1.5),
      from=0.000001, to=1.5, col="red", las=1, ylab="F(x)")
curve(pEGG(x, mu=0.1, sigma=0.8, nu=10, tau=1.5, lower.tail=FALSE),
      from=0.000001, to=1.5, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qEGG(p, mu=0.1, sigma=0.8, nu=10, tau=1.5), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pEGG(x, mu=0.1, sigma=0.8, nu=10, tau=1.5),
      from=0.00001, add=TRUE, col="red")
```

```
## The random function
hist(rEGG(n=100, mu=0.1, sigma=0.8, nu=10, tau=1.5), freq=FALSE,
     xlab="x", las=1, main="")
curve(dEGG(x, mu=0.1, sigma=0.8, nu=10, tau=1.5),
      from=0.0001, to=2, add=TRUE, col="red")

## The Hazard function
curve(hEGG(x, mu=0.1, sigma=0.8, nu=10, tau=1.5), from=0.0001, to=1.5,
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters
```

---

dEMWEx

*The Exponentiated Modified Weibull Extension distribution*


---

## Description

Density, distribution function, quantile function, random generation and hazard function for the Exponentiated Modified Weibull Extension distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$ .

## Usage

```
dEMWEx(x, mu, sigma, nu, tau, log = FALSE)

pEMWEx(q, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

qEMWEx(p, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

rEMWEx(n, mu, sigma, nu, tau)

hEMWEx(x, mu, sigma, nu, tau)
```

## Arguments

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>nu</code>	parameter.
<code>tau</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

## Details

The Exponentiated Modified Weibull Extension Distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$  has density given by

$$f(x) = \nu\sigma\tau\left(\frac{x}{\mu}\right)^{\sigma-1} \exp\left(\left(\frac{x}{\mu}\right)^{\sigma} + \nu\mu(1 - \exp\left(\left(\frac{x}{\mu}\right)^{\sigma}\right))\right)(1 - \exp(\nu\mu(1 - \exp\left(\left(\frac{x}{\mu}\right)^{\sigma}\right))))^{\tau-1},$$

for  $x > 0$ ,  $\nu > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\tau > 0$ .

## Value

dEMWEx gives the density, pEMWEx gives the distribution function, qEMWEx gives the quantile function, rEMWEx generates random deviates and hEMWEx gives the hazard function.

## Author(s)

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

## References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. Reliability Engineering & System Safety, 124, 32-55.

Sarhan, A. M., & Apaloo, J. (2013). Exponentiated modified Weibull extension distribution. Reliability Engineering & System Safety, 112, 137-144.

## See Also

[EMWEx](#)

## Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dEMWEx(x, mu = 49.046, sigma = 3.148, nu=0.00005, tau=0.1), from=0, to=100,
      col = "red", las = 1, ylab = "f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pEMWEx(x, mu = (1/4), sigma = 1, nu=1, tau=2), from = 0, to = 1,
      ylim = c(0, 1), col = "red", las = 1, ylab = "F(x)")
curve(pEMWEx(x, mu = (1/4), sigma = 1, nu=1, tau=2, lower.tail = FALSE),
      from = 0, to = 1, ylim = c(0, 1), col = "red", las = 1, ylab = "R(x)")

## The quantile function
p <- seq(from = 0, to = 0.99999, length.out = 100)
plot(x = qEMWEx(p = p, mu = 49.046, sigma = 3.148, nu=0.00005, tau=0.1), y = p,
     xlab = "Quantile", las = 1, ylab = "Probability")
curve(pEMWEx(x, mu = 49.046, sigma = 3.148, nu=0.00005, tau=0.1), from = 0, add = TRUE,
     col = "red")

## The random function
hist(rEMWEx(1000, mu = (1/4), sigma = 1, nu=1, tau=2), freq = FALSE, xlab = "x",
```

```

    las = 1, main = "")
curve(dEMWEx(x, mu = (1/4), sigma =1, nu=1, tau=2), from = 0, add = TRUE,
      col = "red", ylim = c(0, 0.5))

## The Hazard function(
par(mfrow=c(1,1))
curve(hEMWEx(x, mu = 49.046, sigma =3.148, nu=0.00005, tau=0.1), from = 0, to = 80,
      col = "red", ylab = "Hazard function", las = 1)

par(old_par) # restore previous graphical parameters

```

dEOFNH

*The Extended Odd Frchet-Nadarajah-Haghighi***Description**

Density, distribution function, quantile function, random generation and hazard function for the Extended Odd Frchet-Nadarajah-Haghighi distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$ .

**Usage**

```

dEOFNH(x, mu, sigma, nu, tau, log = FALSE)

pEOFNH(q, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

qEOFNH(p, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

rEOFNH(n, mu, sigma, nu, tau)

hEOFNH(x, mu, sigma, nu, tau)

```

**Arguments**

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>nu</code>	parameter.
<code>tau</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

### Details

The Extended Odd Fréchet-Nadarajah-Haghighi  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$  has density given by

$$f(x) = \frac{\mu\sigma\nu\tau(1+\nu x)^{\sigma-1}e^{(1-(1+\nu x)^\sigma)}[1-(1-e^{(1-(1+\nu x)^\sigma)})^\mu]^{\tau-1}}{(1-e^{(1-(1+\nu x)^\sigma)})^{\mu\tau+1}}e^{-[(1-e^{(1-(1+\nu x)^\sigma)})^{-\mu}-1]^\tau},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu > 0$  and  $\tau > 0$ .

### Value

dEOFNH gives the density, pEOFNH gives the distribution function, qEOFNH gives the quantile function, rEOFNH generates random numbers and hEOFNH gives the hazard function.

### Author(s)

Helber Santiago Padilla, <hspadillar@unal.edu.co>

### References

Nasiru, S. (2018). Extended Odd Fréchet-G Family of Distributions Journal of Probability and Statistics, 2018(1), 2931326.

### See Also

[EOFNH](#)

### Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

##The probability density function
par(mfrow=c(1, 1))

curve(dEOFNH(x, mu=18.5, sigma=5.1, nu=0.1, tau=0.1),
      from=0, to=10, ylim=c(0, 0.25),
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pEOFNH(x,mu=18.5, sigma=5.1, nu=0.1, tau=0.1), from = 0, to = 10,
      ylim = c(0, 1), col = "red", las = 1, ylab = "F(x)")
curve(pEOFNH(x, mu=18.5, sigma=5.1, nu=0.1, tau=0.1, lower.tail = FALSE),
      from = 0, to = 10, ylim = c(0, 1), col = "red", las = 1, ylab = "R(x)")

##The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qEOFNH(p, mu=18.5, sigma=5.1, nu=0.1, tau=0.1), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pEOFNH(x, mu=18.5, sigma=5.1, nu=0.1, tau=0.1), from=0, add=TRUE, col="red")

##The random function
hist(rEOFNH(n=10000, mu=18.5, sigma=5.1, nu=0.1, tau=0.1), freq=FALSE,
     xlab="x", las=1, main="")
```

```

curve(dEOFNH(x, mu=18.5, sigma=5.1, nu=0.1, tau=0.1), from=0, add=TRUE, col="red", ylim=c(0,1.25))

##The Hazard function
par(mfrow=c(1,1))
curve(hEOFNH(x, mu=18.5, sigma=5.1, nu=0.1, tau=0.1), from=0, to=10, ylim=c(0, 1),
      col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters

```

dEW

*The Exponentiated Weibull distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the exponentiated Weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

```

dEW(x, mu, sigma, nu, log = FALSE)

pEW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qEW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rEW(n, mu, sigma, nu)

hEW(x, mu, sigma, nu)

```

**Arguments**

<code>x, q</code>	vector of quantiles.
<code>mu</code>	scale parameter.
<code>sigma, nu</code>	shape parameters.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

**Details**

The Exponentiated Weibull Distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = \nu\mu\sigma x^{\sigma-1} \exp(-\mu x^\sigma)(1 - \exp(-\mu x^\sigma))^{\nu-1},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

**Value**

dEW gives the density, pEW gives the distribution function, qEW gives the quantile function, rEW generates random deviates and hEW gives the hazard function.

**See Also**

[EW](#)

**Examples**

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dEW(x, mu=2, sigma=1.5, nu=0.5), from=0, to=2,
      ylim=c(0, 2.5), col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pEW(x, mu=2, sigma=1.5, nu=0.5),
      from=0, to=2, col="red", las=1, ylab="F(x)")
curve(pEW(x, mu=2, sigma=1.5, nu=0.5, lower.tail=FALSE),
      from=0, to=2, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qEW(p, mu=2, sigma=1.5, nu=0.5), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pEW(x, mu=2, sigma=1.5, nu=0.5), from=0, add=TRUE, col="red")

## The random function
hist(rEW(n=10000, mu=2, sigma=1.5, nu=0.5), freq=FALSE,
     xlab="x", las=1, main="")
curve(dEW(x, mu=2, sigma=1.5, nu=0.5), from=0, add=TRUE, col="red")

## The Hazard function
par(mfrow=c(1,1))
curve(hEW(x, mu=2, sigma=1.5, nu=0.5), from=0, to=2, ylim=c(0, 7),
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters
```

**Description**

Density, distribution function, quantile function, random generation and hazard function for the exponentiated XLindley distribution with parameters  $\mu$  and  $\sigma$ .

**Usage**

```
dEXL(x, mu, sigma, log = FALSE)
```

```
pEXL(q, mu, sigma, log.p = FALSE, lower.tail = TRUE)
```

```
qEXL(p, mu, sigma, lower.tail = TRUE, log.p = FALSE)
```

```
rEXL(n, mu, sigma)
```

```
hEXL(x, mu, sigma, log = FALSE)
```

**Arguments**

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

**Details**

The exponentiated XLindley with parameters  $\mu$  and  $\sigma$  has density given by

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

for  $x \geq 0$ ,  $\mu \geq 0$  and  $\sigma \geq 0$ .

Note: In this implementation we changed the original parameters  $\delta$  for  $\mu$  and  $\alpha$  for  $\sigma$ , we did it to implement this distribution within gamlss framework.

**Value**

dEXL gives the density, pEXL gives the distribution function, qEXL gives the quantile function, rEXL generates random deviates and hEXL gives the hazard function.

**Author(s)**

Manuel Gutierrez Tangarife, <mgutierrezta@unal.edu.co>

**References**

Alomair, A. M., Ahmed, M., Tariq, S., Ahsan-ul-Haq, M., & Talib, J. (2024). An exponentiated XLindley distribution with properties, inference and applications. *Heliyon*, 10(3).

**See Also**

[EXL](#).

**Examples**

```

# Example 1
# Plotting the mass function for different parameter values
curve(dEXL(x, mu=0.5, sigma=0.5),
      from=0, to=5,
      ylim=c(0, 1),
      col="royalblue1", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dEXL(x, mu=1, sigma=0.5),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(dEXL(x, mu=1.5, sigma=0.5),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=0.5, sigma=0.5",
                            "mu=1.0, sigma=0.5",
                            "mu=1.5, sigma=0.5"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.6)

curve(dEXL(x, mu=0.5, sigma=1),
      from=0, to=5,
      ylim=c(0, 1),
      col="royalblue1", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dEXL(x, mu=1, sigma=1),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(dEXL(x, mu=1.5, sigma=1),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=0.5, sigma=1",
                            "mu=1.0, sigma=1",
                            "mu=1.5, sigma=1"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.6)

curve(dEXL(x, mu=0.5, sigma=1.5),
      from=0., to=8,
      ylim=c(0, 1),
      col="royalblue1", lwd=2,
      main="Density function",
      xlab="x", ylab="f(x)")
curve(dEXL(x, mu=1, sigma=1.5),
      col="tomato",
      lwd=2,

```

```

      add=TRUE)
curve(dEXL(x, mu=1.5, sigma=1.5),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("topright", legend=c("mu=0.5, sigma=1.5",
                            "mu=1.0, sigma=1.5",
                            "mu=1.5, sigma=1.5"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.6)

# Example 2
# Checking if the cumulative curves converge to 1
curve(pEXL(x, mu=0.5, sigma=0.5),
      from=0, to=5,
      ylim=c(0, 1),
      col="royalblue1", lwd=2,
      main="Cumulative Distribution Function",
      xlab="x", ylab="f(x)")
curve(pEXL(x, mu=1, sigma=0.5),
      col="tomato",
      lwd=2,
      add=TRUE)
curve(pEXL(x, mu=1.5, sigma=0.5),
      col="seagreen",
      lwd=2,
      add=TRUE)
legend("bottomright", legend=c("mu=0.5, sigma=0.5",
                              "mu=1.0, sigma=0.5",
                              "mu=1.5, sigma=0.5"),
      col=c("royalblue1", "tomato", "seagreen"), lwd=2, cex=0.5)

# Example 3
# The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qEXL(p, mu=2.3, sigma=1.7), y=p, xlab="Quantile",
     las=1, ylab="Probability", main="Quantile function ")
curve(pEXL(x, mu=2.3, sigma=1.7),
      from=0, add=TRUE, col="tomato", lwd=2.5)

# Comparing quantile, density and cumulative
p <- c(0.25, 0.5, 0.75)
quantile <- qEXL(p=p, mu=2.3, sigma=1.7)

for(i in quantile){
  print(integrate(dEXL, lower=0, upper=i, mu=2.3, sigma=1.7))
}

pEXL(q=quantile, mu=2.3, sigma=1.7)

# Example 4
# The random function
x <- rEXL(n=10000, mu=1.5, sigma=2.5)

```

```

hist(x, freq=FALSE)
curve(dEXL(x, mu=1.5, sigma=2.5), from=0, to=20,
      add=TRUE, col="tomato", lwd=2)

# Example 5
# The Hazard function
curve(hEXL(x, mu=1.5, sigma=2), from=0.001, to=4,
      col="tomato", ylab="Hazard function", las=1)

```

---

dExW

*The Extended Weibull distribution*


---

### Description

Density, distribution function, quantile function, random generation and hazard function for the Extended Weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

### Usage

```

dExW(x, mu, sigma, nu, log = FALSE)

pExW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qExW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rExW(n, mu, sigma, nu)

hExW(x, mu, sigma, nu)

```

### Arguments

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>nu</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

### Details

The Extended Weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = \frac{\mu \sigma \nu x^{\sigma-1} \exp(-\mu x^\sigma)}{[1 - (1-\nu) \exp(-\mu x^\sigma)]^2},$$

for  $x > 0$ .

**Value**

dExW gives the density, pExW gives the distribution function, qExW gives the quantile function, rExW generates random deviates and hExW gives the hazard function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Zhang, T., & Xie, M. (2007). Failure data analysis with extended Weibull distribution. *Communications in Statistics—Simulation and Computation*, 36(3), 579-592.

**See Also**

[ExW](#)

**Examples**

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dExW(x, mu=0.3, sigma=2, nu=0.05), from=0.0001, to=2,
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pExW(x, mu=0.3, sigma=2, nu=0.05),
      from=0.0001, to=2, col="red", las=1, ylab="F(x)")
curve(pExW(x, mu=0.3, sigma=2, nu=0.05, lower.tail=FALSE),
      from=0.0001, to=2, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qExW(p, mu=0.3, sigma=2, nu=0.05), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pExW(x, mu=0.3, sigma=2, nu=0.05),
      from=0, add=TRUE, col="red")

## The random function
hist(rExW(n=10000, mu=0.3, sigma=2, nu=0.05), freq=FALSE,
     xlab="x", ylim=c(0, 2), las=1, main="")
curve(dExW(x, mu=0.3, sigma=2, nu=0.05),
      from=0.001, to=4, add=TRUE, col="red")

## The Hazard function
curve(hExW(x, mu=0.3, sigma=2, nu=0.05), from=0.001, to=4,
     col="red", ylab="Hazard function", las=1)
```

```
par(old_par) # restore previous graphical parameters
```

---

dExWALD

*The Ex-Wald distribution*


---

### Description

These functions define the density, distribution function, quantile function and random generation for the Ex-Wald distribution with parameter  $\mu$ ,  $\sigma$  and  $\nu$ .

### Usage

```
dExWALD(x, mu = 1.5, sigma = 1.5, nu = 2, log = FALSE)
```

```
pExWALD(q, mu = 1.5, sigma = 1.5, nu = 2, lower.tail = TRUE, log.p = FALSE)
```

```
qExWALD(p, mu = 1.5, sigma = 1.5, nu = 2)
```

```
rExWALD(n, mu = 1.5, sigma = 1.5, nu = 2)
```

### Arguments

x, q	vector of (non-negative integer) quantiles.
mu	vector of the mu parameter.
sigma	vector of the sigma parameter.
nu	vector of the nu parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of random values to return.

### Details

The Wald distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x|\mu, \sigma, \nu) = \frac{1}{\nu} \exp\left(\frac{-x}{\nu} + \sigma(\mu - k)\right) F_W(x|k, \sigma) \text{ for } k \geq 0$$

$$f(x|\mu, \sigma, \nu) = \frac{1}{\nu} \exp\left(\frac{-(\sigma - \mu)^2}{2x}\right) Re\left(w\left(k' \sqrt{x/2} + \frac{\sigma i}{\sqrt{2x}}\right)\right) \text{ for } k < 0$$

where  $k = \sqrt{\mu^2 - \frac{2}{\nu}}$ ,  $k' = \sqrt{\frac{2}{\nu} - \mu^2}$  and  $F_W$  corresponds to the cumulative function of the Wald distribution.

More details about those expressions can be found on page 680 from Heathcote (2004).

### Value

dExWALD gives the density, pExWALD gives the distribution function, qExWALD gives the quantile function, rExWALD generates random deviates.

**Author(s)**

Freddy Hernandez, <fhernanb@unal.edu.co>

**References**

Schwarz, W. (2001). The ex-Wald distribution as a descriptive model of response times. *Behavior Research Methods, Instruments, & Computers*, 33, 457-469.

Heathcote, A. (2004). Fitting Wald and ex-Wald distributions to response time data: An example using functions for the S-PLUS package. *Behavior Research Methods, Instruments, & Computers*, 36, 678-694.

**See Also**

[ExWALD](#)

**Examples**

```
# Example 1
# Plotting the mass function for different parameter values
curve(dExWALD(x, mu=0.15, sigma=52.5, nu=50), ylim=c(0, 0.005),
      from=0, to=1200, col="cadetblue3", las=1, ylab="f(x)")

curve(dExWALD(x, mu=0.20, sigma=70, nu=50),
      add=TRUE, col="purple")

curve(dExWALD(x, mu=0.25, sigma=87.5, nu=50),
      add=TRUE, col="goldenrod")

curve(dExWALD(x, mu=0.20, sigma=70, nu=115),
      add=TRUE, col="tomato")

curve(dExWALD(x, mu=0.20, sigma=70, nu=35),
      add=TRUE, col="blue")

legend("topright", col=c("cadetblue3", "purple", "goldenrod",
                        "tomato", "blue"),
      lty=1, bty="n",
      legend=c("mu=0.15, sigma=52.5, nu=50",
              "mu=0.20, sigma=70.0, nu=50",
              "mu=0.25, sigma=87.5, nu=50",
              "mu=0.20, sigma=70.0, nu=115",
              "mu=0.20, sigma=70.0, nu=35"))

# Example 2
# Checking if the cumulative curves converge to 1
curve(pExWALD(x, mu=0.15, sigma=52.5, nu=50), ylim=c(0, 1),
      from=0, to=1200, col="cadetblue3", las=1, ylab="F(x)")

curve(pExWALD(x, mu=0.20, sigma=70, nu=50),
      add=TRUE, col="purple")
```

```

curve(pExWALD(x, mu=0.25, sigma=87.5, nu=50),
      add=TRUE, col="goldenrod")

curve(pExWALD(x, mu=0.20, sigma=70, nu=115),
      add=TRUE, col="tomato")

curve(pExWALD(x, mu=0.20, sigma=70, nu=35),
      add=TRUE, col="blue")

legend("bottomright", col=c("cadetblue3", "purple", "goldenrod",
                           "tomato", "blue"),
      lty=1, bty="n",
      legend=c("mu=0.15, sigma=52.5, nu=50",
              "mu=0.20, sigma=70.0, nu=50",
              "mu=0.25, sigma=87.5, nu=50",
              "mu=0.20, sigma=70.0, nu=115",
              "mu=0.20, sigma=70.0, nu=35"))

# Example 3
# Checking the quantile function
mu <- 5
sigma <- 3
nu <- 2
p <- seq(from=0.1, to=0.99, length.out=100)
plot(x=qExWALD(p, mu=mu, sigma=sigma, nu=nu), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pExWALD(x, mu=mu, sigma=sigma, nu=nu), from=0, add=TRUE, col="red")

# Example 4
# Comparing the random generator output with
# the theoretical probabilities
mu <- 0.2
sigma <- 70
nu <- 35
x <- rExWALD(n=10000, mu=mu, sigma=sigma, nu=nu)
hist(x, freq=FALSE)
curve(dExWALD(x, mu=mu, sigma=sigma, nu=nu), col="tomato", add=TRUE)

```

**Description**

Density, distribution function, quantile function, random generation and hazard function for the Flexible Weibull Extension distribution with parameters  $\mu$  and  $\sigma$ .

**Usage**

```
dFWE(x, mu, sigma, log = FALSE)
```

```
pFWE(q, mu, sigma, lower.tail = TRUE, log.p = FALSE)
```

```
qFWE(p, mu, sigma, lower.tail = TRUE, log.p = FALSE)
```

```
rFWE(n, mu, sigma)
```

```
hFWE(x, mu, sigma)
```

### Arguments

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

The Flexible Weibull extension with parameters `mu` and `sigma` has density given by

$$f(x) = (\mu + \sigma/x^2) \exp(\mu x - \sigma/x) \exp(-\exp(\mu x - \sigma/x))$$

for  $x > 0$ .

### Value

dFWE gives the density, pFWE gives the distribution function, qFWE gives the quantile function, rFWE generates random deviates and hFWE gives the hazard function.

### See Also

[FWE](#)

### Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dFWE(x, mu=0.75, sigma=0.5), from=0, to=3,
      ylim=c(0, 1.7), col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pFWE(x, mu=0.75, sigma=0.5), from=0, to=3,
      col="red", las=1, ylab="F(x)")
curve(pFWE(x, mu=0.75, sigma=0.5, lower.tail=FALSE),
      from=0, to=3, col="red", las=1, ylab="R(x)")
```

```

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qFWE(p, mu=0.75, sigma=0.5), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pFWE(x, mu=0.75, sigma=0.5), from=0, add=TRUE, col="red")

## The random function
hist(rFWE(n=1000, mu=2, sigma=0.5), freq=FALSE, xlab="x",
     ylim=c(0, 2), las=1, main="")
curve(dFWE(x, mu=2, sigma=0.5), from=0, to=3, add=TRUE, col="red")

## The Hazard function
par(mfrow=c(1,1))
curve(hFWE(x, mu=0.75, sigma=0.5), from=0, to=2, ylim=c(0, 2.5),
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters

```

---

dGammaW

*The Gamma Weibull distribution*


---

### Description

Density, distribution function, quantile function, random generation and hazard function for the Gamma Weibull distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$ .

### Usage

```

dGammaW(x, mu, sigma, nu, log = FALSE)

pGammaW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qGammaW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rGammaW(n, mu, sigma, nu)

hGammaW(x, mu, sigma, nu)

```

### Arguments

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>nu</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

## Details

The Gamma Weibull Distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = \frac{\sigma \mu^\nu}{\Gamma(\nu)} x^{\nu\sigma-1} \exp(-\mu x^\sigma),$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma \geq 0$  and  $\nu > 0$ .

## Value

dGammaW gives the density, pGammaW gives the distribution function, qGammaW gives the quantile function, rGammaW generates random deviates and hGammaW gives the hazard function.

## Author(s)

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

## References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. Reliability Engineering & System Safety, 124, 32-55.

Stacy, E. W. (1962). A generalization of the gamma distribution. The Annals of mathematical statistics, 1187-1192.

## See Also

[GammaW](#)

## Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dGammaW(x, mu = 0.5, sigma = 2, nu=1), from = 0, to = 6,
      col = "red", las = 1, ylab = "f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pGammaW(x, mu = 0.5, sigma = 2, nu=1), from = 0, to = 3,
      ylim = c(0, 1), col = "red", las = 1, ylab = "F(x)")
curve(pGammaW(x, mu = 0.5, sigma = 2, nu=1, lower.tail = FALSE),
      from = 0, to = 3, ylim = c(0, 1), col = "red", las = 1, ylab = "R(x)")

## The quantile function
p <- seq(from = 0, to = 0.99999, length.out = 100)
plot(x = qGammaW(p = p, mu = 0.5, sigma = 2, nu=1), y = p,
     xlab = "Quantile", las = 1, ylab = "Probability")
curve(pGammaW(x, mu = 0.5, sigma = 2, nu=1), from = 0, add = TRUE,
     col = "red")

## The random function
hist(rGammaW(1000, mu = 0.5, sigma = 2, nu=1), freq = FALSE, xlab = "x",
     ylim = c(0, 1), las = 1, main = "")
```

```

curve(dGammaW(x, mu = 0.5, sigma = 2, nu=1), from = 0, add = TRUE,
col = "red", ylim = c(0, 1))

## The Hazard function
par(mfrow=c(1,1))
curve(hGammaW(x, mu = 0.5, sigma = 2, nu=1), from = 0, to = 2,
ylim = c(0, 1), col = "red", ylab = "Hazard function", las = 1)

par(old_par) # restore previous graphical parameters

```

---

dGGD

*The Generalized Gompertz distribution*


---

### Description

Density, distribution function, quantile function, random generation and hazard function for the generalized Gompertz distribution with parameters  $\mu$   $\sigma$  and  $\nu$ .

### Usage

```

dGGD(x, mu, sigma, nu, log = FALSE)

pGGD(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qGGD(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rGGD(n, mu, sigma, nu)

hGGD(x, mu, sigma, nu)

```

### Arguments

<code>x, q</code>	vector of quantiles.
<code>mu, nu</code>	scale parameter.
<code>sigma</code>	shape parameters.
<code>log, log.p</code>	logical; if TRUE, probabilities p are given as log(p).
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

### Details

The Generalized Gompertz Distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = \nu \mu \exp\left(-\frac{\mu}{\sigma}(\exp(\sigma x - 1))\right) \left(1 - \exp\left(-\frac{\mu}{\sigma}(\exp(\sigma x - 1))\right)\right)^{(\nu-1)},$$

for  $x \geq 0$ ,  $\mu > 0$ ,  $\sigma \geq 0$  and  $\nu > 0$ .

**Value**

dGGD gives the density, pGGD gives the distribution function, qGGD gives the quantile function, rGGD generates random deviates and hGGD gives the hazard function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

El-Gohary, A., Alshamrani, A., & Al-Otaibi, A. N. (2013). The generalized Gompertz distribution. Applied mathematical modelling, 37(1-2), 13-24.

**See Also**

[GGD](#)

**Examples**

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
par(mfrow = c(1, 1))
curve(dGGD(x, mu=1, sigma=0.3, nu=1.5), from = 0, to = 4,
      col = "red", las = 1, ylab = "f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pGGD(x, mu=1, sigma=0.3, nu=1.5), from = 0, to = 4,
      ylim = c(0, 1), col = "red", las = 1, ylab = "F(x)")
curve(pGGD(x, mu=1, sigma=0.3, nu=1.5, lower.tail = FALSE),
      from = 0, to = 4, ylim = c(0, 1), col = "red", las = 1, ylab = "R(x)")

## The quantile function
p <- seq(from = 0, to = 0.99999, length.out = 100)
plot(x = qGGD(p=p, mu=1, sigma=0.3, nu=1.5), y = p,
     xlab = "Quantile", las = 1, ylab = "Probability")
curve(pGGD(x, mu=1, sigma=0.3, nu=1.5), from = 0, add = TRUE,
     col = "red")

## The random function
hist(rGGD(1000, mu=1, sigma=0.3, nu=1.5), freq = FALSE, xlab = "x",
     las = 1, ylim = c(0, 0.7), main = "")
curve(dGGD(x,mu=1, sigma=0.3, nu=1.5), from = 0, to =8, add = TRUE,
     col = "red")

## The Hazard function
par(mfrow=c(1,1))
curve(hGGD(x, mu=1, sigma=0.3, nu=1.5), from = 0, to = 3, col = "red",
     ylab = "The hazard function", las = 1)

par(old_par) # restore previous graphical parameters
```

dGIW

*The Generalized Inverse Weibull distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the Generalized Inverse Weibull distribution with parameters mu, sigma and nu.

**Usage**

```
dGIW(x, mu, sigma, nu, log = FALSE)
```

```
pGIW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
```

```
qGIW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
```

```
rGIW(n, mu, sigma, nu)
```

```
hGIW(x, mu, sigma, nu)
```

**Arguments**

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
nu	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

**Details**

The Generalized Inverse Weibull distribution mu, sigma and nu has density given by

$$f(x) = \nu \sigma \mu^\sigma x^{-(\sigma+1)} \exp\left\{-\nu \left(\frac{\mu}{x}\right)^\sigma\right\},$$

for  $x > 0$ .

**Value**

dGIW gives the density, pGIW gives the distribution function, qGIW gives the quantile function, rGIW generates random deviates and hGIW gives the hazard function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

## References

- Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.
- De Gusmao, F. R., Ortega, E. M., & Cordeiro, G. M. (2011). The generalized inverse Weibull distribution. *Statistical Papers*, 52, 591-619.

## See Also

[GIW](#)

## Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dGIW(x, mu=3, sigma=5, nu=0.5), from=0.001, to=8,
      col="red", ylab="f(x)", las=1)

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pGIW(x, mu=3, sigma=5, nu=0.5),
      from=0.0001, to=14, col="red", las=1, ylab="F(x)")
curve(pGIW(x, mu=3, sigma=5, nu=0.5, lower.tail=FALSE),
      from=0.0001, to=14, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qGIW(p, mu=3, sigma=5, nu=0.5), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pGIW(x, mu=3, sigma=5, nu=0.5),
      from=0, add=TRUE, col="red")

## The random function
hist(rGIW(n=1000, mu=3, sigma=5, nu=0.5), freq=FALSE,
     xlab="x", ylim=c(0, 0.8), las=1, main="")
curve(dGIW(x, mu=3, sigma=5, nu=0.5),
      from=0.001, to=14, add=TRUE, col="red")

## The Hazard function
curve(hGIW(x, mu=3, sigma=5, nu=0.5), from=0.001, to=30,
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters
```

**Description**

Density, distribution function, quantile function, random generation and hazard function for the generalized modified weibull distribution with parameters mu, sigma, nu and tau.

**Usage**

```
dGMW(x, mu, sigma, nu, tau, log = FALSE)
pGMW(q, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)
qGMW(p, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)
rGMW(n, mu, sigma, nu, tau)
hGMW(x, mu, sigma, nu, tau, log = FALSE)
```

**Arguments**

x, q	vector of quantiles.
mu	scale parameter.
sigma	shape parameter.
nu	shape parameter.
tau	acceleration parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

**Details**

The generalized modified weibull with parameters mu, sigma, nu and tau has density given by

$$f(x) = \mu\sigma x^{\nu-1}(\nu + \tau x) \exp(\tau x - \mu x^{\nu} e^{\tau x}) [1 - \exp(-\mu x^{\nu} e^{\tau x})]^{\sigma-1},$$

for  $x > 0$ .

**Value**

dGMW gives the density, pGMW gives the distribution function, qGMW gives the quantile function, rGMW generates random deviates and hGMW gives the hazard function.

**See Also**

[GMW](#)

**Examples**

```

old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dGMW(x, mu=2, sigma=0.5, nu=2, tau=1.5), from=0, to=0.8,
      ylim=c(0, 2), col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pGMW(x, mu=2, sigma=0.5, nu=2, tau=1.5),
      from=0, to=1.2, col="red", las=1, ylab="F(x)")
curve(pGMW(x, mu=2, sigma=0.5, nu=2, tau=1.5, lower.tail=FALSE),
      from=0, to=1.2, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qGMW(p, mu=2, sigma=0.5, nu=2, tau=1.5), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pGMW(x, mu=2, sigma=0.5, nu=2, tau=1.5), from=0, add=TRUE, col="red")

## The random function
hist(rGMW(n=1000, mu=2, sigma=0.5, nu=2, tau=1.5), freq=FALSE,
     xlab="x", main="", las=1)
curve(dGMW(x, mu=2, sigma=0.5, nu=2, tau=1.5), from=0, add=TRUE, col="red")

## The Hazard function
par(mfrow=c(1,1))
curve(hGMW(x, mu=2, sigma=0.5, nu=2, tau=1.5), from=0, to=1, ylim=c(0, 16),
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters

```

---

dGWF

*Generalized Weibull distribution*


---

**Description**

Density, distribution function, quantile function, random generation and hazard function for the generalized Weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

```

dGWF(x, mu, sigma, nu, log = FALSE)

pGWF(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qGWF(p, mu, sigma, nu)

rGWF(n, mu, sigma, nu)

```

```
hGWF(x, mu, sigma, nu)
```

### Arguments

x, q	vector of quantiles.
mu	parameter one.
sigma	parameter two.
nu	parameter three.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

The generalized Weibull with parameters mu, sigma and nu has density given by

$$f(x) = \mu\sigma x^{\sigma-1} (1 - \mu\nu x^\sigma)^{\frac{1}{\nu}-1}$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $-\infty < \nu < \infty$ .

### Value

dGWF gives the density, pGWF gives the distribution function, qGWF gives the quantile function, rGWF generates random deviates and hGWF gives the hazard function.

### Author(s)

Jaime Mosquera, <jmosquerag@unal.edu.co>

### References

Mudholkar, G. S., & Kollia, G. D. (1994). Generalized Weibull family: a structural analysis. Communications in statistics-theory and methods, 23(4), 1149-1171.

### Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(
  dGWF(x, mu = 5, sigma = 2, nu = -0.2),
  from = 0, to = 5, col = "red", las = 1, ylab = "f(x)"
)

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
```

```

curve(
  pGWF(x, mu = 5, sigma = 2, nu = -0.2),
  from = 0, to = 5, ylim = c(0, 1),
  col = "red", las = 1, ylab = "F(x)"
)
curve(
  pGWF(
    x, mu = 5, sigma = 2, nu = -0.2,
    lower.tail = FALSE
  ),
  from = 0, to = 5, ylim = c(0, 1),
  col = "red", las = 1, ylab = "R(x)"
)

## The quantile function
p <- seq(from = 0, to = 0.999, length.out = 100)
plot(
  x = qGWF(p, mu = 5, sigma = 2, nu = -0.2),
  y = p, xlab = "Quantile", las = 1,
  ylab = "Probability"
)
curve(
  pGWF(x, mu = 5, sigma = 2, nu = -0.2),
  from = 0, add = TRUE, col = "red"
)

## The random function
hist(
  rGWF(n = 10000, mu = 5, sigma = 2, nu = -0.2),
  freq = FALSE, xlab = "x", las = 1, main = "", ylim = c(0, 2.0)
)
curve(dGWF(x, mu = 5, sigma = 2, nu = -0.2),
  from = 0, add = TRUE, col = "red"
)

## The Hazard function
par(mfrow = c(1, 1))
curve(
  hGWF(x, mu = 0.003, sigma = 5e-6, nu = 0.025),
  from = 0, to = 250, col = "red",
  ylab = "Hazard function", las = 1
)

par(old_par) # restore previous graphical parameters

```

**Description**

Density, distribution function, quantile function, random generation and hazard function for the inverse weibull distribution with parameters  $\mu$  and  $\sigma$ .

**Usage**

```
dIW(x, mu, sigma, log = FALSE)

pIW(q, mu, sigma, lower.tail = TRUE, log.p = FALSE)

qIW(p, mu, sigma, lower.tail = TRUE, log.p = FALSE)

rIW(n, mu, sigma)

hIW(x, mu, sigma)
```

**Arguments**

x, q	vector of quantiles.
mu	scale parameter.
sigma	shape parameters.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

**Details**

The inverse weibull distribution with parameters mu and sigma has density given by

$$f(x) = \mu\sigma x^{-\sigma-1} \exp(\mu x^{-\sigma})$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$

**Value**

dIW gives the density, pIW gives the distribution function, qIW gives the quantile function, rIW generates random deviates and hIW gives the hazard function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. Reliability Engineering & System Safety, 124, 32-55.

Drapella, A. (1993). The complementary Weibull distribution: unknown or just forgotten?. Quality and reliability engineering international, 9(4), 383-385.

**See Also**

[IW](#)

**Examples**

```

old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dIW(x, mu=5, sigma=2.5), from=0, to=10,
      ylim=c(0, 0.55), col="red", las=1, ylab="f(x)")
#'
## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pIW(x, mu=5, sigma=2.5),
      from=0, to=10, col="red", las=1, ylab="F(x)")
curve(pIW(x, mu=5, sigma=2.5, lower.tail=FALSE),
      from=0, to=10, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qIW(p, mu=5, sigma=2.5), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pIW(x, mu=5, sigma=2.5), from=0, add=TRUE, col="red")

## The random function
hist(rIW(n=10000, mu=5, sigma=2.5), freq=FALSE, xlim=c(0,60),
     xlab="x", las=1, main="")
curve(dIW(x, mu=5, sigma=2.5), from=0, add=TRUE, col="red")

## The Hazard function
par(mfrow=c(1,1))
curve(hIW(x, mu=5, sigma=2.5), from=0, to=15, ylim=c(0, 0.9),
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters

```

dKumIW

*The Kumaraswamy Inverse Weibull distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the Kumaraswamy Inverse Weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

```

dKumIW(x, mu, sigma, nu, log = FALSE)

pKumIW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qKumIW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rKumIW(n, mu, sigma, nu)

```

hKumIW(x, mu, sigma, nu)

### Arguments

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
nu	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

The Kumaraswamy Inverse Weibull Distribution with parameters mu, sigma and nu has density given by

$$f(x) = \mu\sigma\nu x^{-\mu-1} \exp -\sigma x^{-\mu} (1 - \exp -\sigma x^{-\mu})^{\nu-1},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

### Value

dKumIW gives the density, pKumIW gives the distribution function, qKumIW gives the quantile function, rKumIW generates random deviates and hKumIW gives the hazard function.

### Author(s)

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

### References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. Reliability Engineering & System Safety, 124, 32-55.

Shahbaz, M. Q., Shahbaz, S., & Butt, N. S. (2012). The Kumaraswamy Inverse Weibull Distribution. Pakistan journal of statistics and operation research, 479-489.

### See Also

[KumIW](#)

**Examples**

```

old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
par(mfrow = c(1, 1))
curve(dKumIW(x, mu = 1.5, sigma= 1.5, nu = 1), from = 0, to = 8.5,
      col = "red", las = 1, ylab = "f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pKumIW(x, mu = 1.5, sigma= 1.5, nu = 1), from = 0, to = 8.5,
      ylim = c(0, 1), col = "red", las = 1, ylab = "F(x)")
curve(pKumIW(x, mu = 1.5, sigma= 1.5, nu = 1, lower.tail = FALSE),
      from = 0, to = 6, ylim = c(0, 1), col = "red", las = 1, ylab = "R(x)")

## The quantile function
p <- seq(from = 0, to = 0.99999, length.out = 100)
plot(x = qKumIW(p=p, mu = 1.5, sigma= 1.5, nu = 10), y = p,
     xlab = "Quantile", las = 1, ylab = "Probability")
curve(pKumIW(x, mu = 1.5, sigma= 1.5, nu = 10), from = 0, add = TRUE,
     col = "red")

## The random function
hist(rKumIW(1000, mu = 1.5, sigma= 1.5, nu = 5), freq = FALSE, xlab = "x",
     las = 1, ylim = c(0, 1.5), main = "")
curve(dKumIW(x, mu = 1.5, sigma= 1.5, nu = 5), from = 0, to = 8, add = TRUE,
     col = "red")

## The Hazard function
par(mfrow=c(1,1))
curve(hKumIW(x, mu = 1.5, sigma= 1.5, nu = 1), from = 0, to = 3,
     ylim = c(0, 0.7), col = "red", ylab = "Hazard function", las = 1)

par(old_par) # restore previous graphical parameters

```

dLIN

*Lindley distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the Lindley distribution with parameter  $\mu$ .

**Usage**

```
dLIN(x, mu, log = FALSE)
```

```
pLIN(q, mu, lower.tail = TRUE, log.p = FALSE)
```

```
qLIN(p, mu, lower.tail = TRUE, log.p = FALSE)
```

```
rLIN(n, mu)
```

```
hLIN(x, mu, log = FALSE)
```

### Arguments

x, q	vector of quantiles.
mu	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

Lindley Distribution with parameter mu has density given by

$$f(x) = \frac{\mu^2}{\mu+1}(1+x)\exp(-\mu x),$$

for  $x > 0$  and  $\mu > 0$ . These function were taken form LindleyR package.

### Value

dLIN gives the density, pLIN gives the distribution function, qLIN gives the quantile function, rLIN generates random deviates and hLIN gives the hazard function.

### Author(s)

Freddy Hernandez, <fhernanb@unal.edu.co>

### References

Lindley, D. V. (1958). Fiducial distributions and Bayes' theorem. Journal of the Royal Statistical Society. Series B (Methodological), 102-107.

### See Also

[LIN](#)

### Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dLIN(x, mu=1.5), from=0.0001, to=10,
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
```

```

par(mfrow=c(1, 2))
curve(pLIN(x, mu=2), from=0.0001, to=10, col="red", las=1, ylab="F(x)")
curve(pLIN(x, mu=2, lower.tail=FALSE), from=0.0001,
      to=10, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qLIN(p, mu=2), y=p, xlab="Quantile", las=1, ylab="Probability")
curve(pLIN(x, mu=2), from=0, add=TRUE, col="red")

## The random function
hist(rLIN(n=10000, mu=2), freq=FALSE, xlab="x", las=1, main="")
curve(dLIN(x, mu=2), from=0.09, to=5, add=TRUE, col="red")

## The Hazard function
curve(hLIN(x, mu=2), from=0.001, to=10, col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters

```

dLW

*The Log-Weibull distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the Log-Weibull distribution with parameters  $\mu$  and  $\sigma$ .

**Usage**

```

dLW(x, mu, sigma, log = FALSE)

pLW(q, mu, sigma, lower.tail = TRUE, log.p = FALSE)

qLW(p, mu, sigma, lower.tail = TRUE, log.p = FALSE)

rLW(n, mu, sigma)

hLW(x, mu, sigma)

```

**Arguments**

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

**Details**

The Log-Weibull Distribution with parameters  $\mu$  and  $\sigma$  has density given by

$$f(y) = (1/\sigma)e^{((y-\mu)/\sigma)} \exp\{-e^{((y-\mu)/\sigma)}\},$$

for  $-\infty < y < \infty$ .

**Value**

dLW gives the density, pLW gives the distribution function, qLW gives the quantile function, rLW generates random deviates and hLW gives the hazard function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. Reliability Engineering & System Safety, 124, 32-55.

Gumbel, E. J. (1958). Statistics of extremes. Columbia university press.

**See Also**

[LW](#)

**Examples**

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dLW(x, mu=0, sigma=1.5), from=-8, to=5,
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pLW(x, mu=0, sigma=1.5),
      from=-8, to=5, col="red", las=1, ylab="F(x)")
curve(pLW(x, mu=0, sigma=1.5, lower.tail=FALSE),
      from=-8, to=5, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qLW(p, mu=0, sigma=1.5), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pLW(x, mu=0, sigma=1.5), from=-8, to=5, add=TRUE, col="red")

## The random function
hist(rLW(n=10000, mu=0, sigma=1.5), freq=FALSE,
     xlab="x", las=1, main="")
curve(dLW(x, mu=0, sigma=1.5), from=-15, to=6, add=TRUE, col="red")
```

```
## The Hazard function
par(mfrow=c(1,1))
curve(hLW(x, mu=0, sigma=1.5), from=-8, to=7,
      col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters
```

dMOEIW

*The Marshall-Olkin Extended Inverse Weibull distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the Marshall-Olkin Extended Inverse Weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

```
dMOEIW(x, mu, sigma, nu, log = FALSE)

pMOEIW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qMOEIW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rMOEIW(n, mu, sigma, nu)

hMOEIW(x, mu, sigma, nu)
```

**Arguments**

$x, q$	vector of quantiles.
$\mu$	parameter.
$\sigma$	parameter.
$\nu$	parameter.
$\log, \log.p$	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
$\text{lower.tail}$	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
$p$	vector of probabilities.
$n$	number of observations.

**Details**

The Marshall-Olkin Extended Inverse Weibull distribution  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = \frac{\mu \sigma \nu x^{-(\sigma+1)} \exp\{-\mu x^{-\sigma}\}}{\{\nu - (\nu-1) \exp\{-\mu x^{-\sigma}\}\}^2},$$

for  $x > 0$ .

**Value**

dMOEIW gives the density, pMOEIW gives the distribution function, qMOEIW gives the quantile function, rMOEIW generates random deviates and hMOEIW gives the hazard function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Okasha, H. M., El-Baz, A. H., Tarabia, A. M. K., & Basheer, A. M. (2017). Extended inverse Weibull distribution with reliability application. *Journal of the Egyptian Mathematical Society*, 25(3), 343-349.

**See Also**

[MOEIW](#)

**Examples**

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dMOEIW(x, mu=0.6, sigma=1.7, nu=0.3), from=0, to=2,
      col="red", ylab="f(x)", las=1)

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pMOEIW(x, mu=0.6, sigma=1.7, nu=0.3),
      from=0.0001, to=2, col="red", las=1, ylab="F(x)")
curve(pMOEIW(x, mu=0.6, sigma=1.7, nu=0.3, lower.tail=FALSE),
      from=0.0001, to=2, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qMOEIW(p, mu=0.6, sigma=1.7, nu=0.3), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pMOEIW(x, mu=0.6, sigma=1.7, nu=0.3),
      from=0, add=TRUE, col="red")

## The random function
hist(rMOEIW(n=1000, mu=0.6, sigma=1.7, nu=0.3), freq=FALSE,
     xlab="x", las=1, main="")
curve(dMOEIW(x, mu=0.6, sigma=1.7, nu=0.3),
      from=0.001, to=4, add=TRUE, col="red")

## The Hazard function
curve(hMOEIW(x, mu=0.5, sigma=0.7, nu=1), from=0.001, to=3,
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters
```

dMOEW

*The Marshall-Olkin Extended Weibull distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the Marshall-Olkin Extended Weibull distribution with parameters mu, sigma and nu.

**Usage**

```
dMOEW(x, mu, sigma, nu, log = FALSE)
```

```
pMOEW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
```

```
qMOEW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
```

```
rMOEW(n, mu, sigma, nu)
```

```
hMOEW(x, mu, sigma, nu)
```

**Arguments**

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
nu	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

**Details**

The Marshall-Olkin Extended Weibull distribution mu, sigma and nu has density given by

$$f(x) = \frac{\mu\sigma\nu(\nu x)^{\sigma-1} \exp\{-(\nu x)^\sigma\}}{\{1-(1-\mu)\exp\{-(\nu x)^\sigma\}\}^2},$$

for  $x > 0$ .

**Value**

dMOEW gives the density, pMOEW gives the distribution function, qMOEW gives the quantile function, rMOEW generates random deviates and hMOEW gives the hazard function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

## References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Ghitany, M. E., Al-Hussaini, E. K., & Al-Jarallah, R. A. (2005). Marshall–Olkin extended Weibull distribution and its application to censored data. *Journal of Applied Statistics*, 32(10), 1025-1034.

## See Also

[MOEW](#)

## Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dMOEW(x, mu=0.5, sigma=0.7, nu=1), from=0.001, to=1,
      col="red", ylab="f(x)", las=1)

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pMOEW(x, mu=0.5, sigma=0.7, nu=1),
      from=0.0001, to=2, col="red", las=1, ylab="F(x)")
curve(pMOEW(x, mu=0.5, sigma=0.7, nu=1, lower.tail=FALSE),
      from=0.0001, to=2, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qMOEW(p, mu=0.5, sigma=0.7, nu=1), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pMOEW(x, mu=0.5, sigma=0.7, nu=1),
      from=0, add=TRUE, col="red")

## The random function
hist(rMOEW(n=100, mu=0.5, sigma=0.7, nu=1), freq=FALSE,
     xlab="x", ylim=c(0, 1), las=1, main="")
curve(dMOEW(x, mu=0.5, sigma=0.7, nu=1),
      from=0.001, to=2, add=TRUE, col="red")

## The Hazard function
curve(hMOEW(x, mu=0.5, sigma=0.7, nu=1), from=0.001, to=3,
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters
```

**Description**

Density, distribution function, quantile function, random generation and hazard function for the Marshall-Olkin Kappa distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$ .

**Usage**

```
dMOK(x, mu, sigma, nu, tau, log = FALSE)

pMOK(q, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

qMOK(p, mu, sigma, nu, tau, lower.tail = TRUE, log.p = FALSE)

rMOK(n, mu, sigma, nu, tau)

hMOK(x, mu, sigma, nu, tau)
```

**Arguments**

$x, q$	vector of quantiles.
$\mu$	parameter.
$\sigma$	parameter.
$\nu$	parameter.
$\tau$	parameter.
$\log, \log.p$	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
$lower.tail$	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
$p$	vector of probabilities.
$n$	number of observations.

**Details**

The Marshall-Olkin Kappa distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$  has density given by:

$$f(x) = \frac{\tau \frac{\mu\nu}{\sigma} \left(\frac{x}{\sigma}\right)^{\nu-1} \left(\mu + \left(\frac{x}{\sigma}\right)^{\mu\nu}\right)^{-\frac{\mu+1}{\mu}}}{\left[\tau + (1-\tau) \left(\frac{\left(\frac{x}{\sigma}\right)^{\mu\nu}}{\mu + \left(\frac{x}{\sigma}\right)^{\mu\nu}}\right)^{\frac{1}{\mu}}\right]^2}$$

for  $x > 0$ .

**Value**

dMOK gives the density, pMOK gives the distribution function, qMOK gives the quantile function, rMOK generates random deviates and hMOK gives the hazard function.

**Author(s)**

Angel Muñoz,

## References

Javed, M., Nawaz, T., & Irfan, M. (2019). The Marshall-Olkin kappa distribution: properties and applications. *Journal of King Saud University-Science*, 31(4), 684-691.

## See Also

[MOK](#)

## Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
par(mfrow = c(1,1))
curve(dMOK(x = x, mu = 1, sigma = 3.5, nu = 3, tau = 2), from = 0, to = 15,
      ylab = 'f(x)', col = 2, las = 1)

## The cumulative distribution and the Reliability function

par(mfrow = c(1,2))
curve(pMOK(q = x, mu = 1, sigma = 2.5, nu = 3, tau = 2), from = 0, to = 10,
      col = 2, lwd = 2, las = 1, ylab = 'F(x)')
curve(pMOK(q = x, mu = 1, sigma = 2.5, nu = 3, tau = 2, lower.tail = FALSE), from = 0, to = 10,
      col = 2, lwd = 2, las = 1, ylab = 'R(x)')

## The quantile function
p <- seq(from = 0.00001, to = 0.99999, length.out = 100)
plot(x = qMOK(p = p, mu = 4, sigma = 2.5, nu = 3, tau = 2), y = p, xlab = 'Quantile',
     las = 1, ylab = 'Probability')
curve(pMOK(q = x, mu = 4, sigma = 2.5, nu = 3, tau = 2), from = 0, to = 15,
      add = TRUE, col = 2)

## The random function

hist(rMOK(n = 10000, mu = 1, sigma = 2.5, nu = 3, tau = 2), freq = FALSE,
     xlab = "x", las = 1, main = '', ylim = c(0,.3), xlim = c(0,20), breaks = 50)
curve(dMOK(x, mu = 1, sigma = 2.5, nu = 3, tau = 2), from = 0, to = 15, add = TRUE, col = 2)

## The Hazard function

par(mfrow = c(1,1))
curve(hMOK(x = x, mu = 1, sigma = 2.5, nu = 3, tau = 2), from = 0, to = 20,
      col = 2, ylab = 'Hazard function', las = 1)

par(old_par) # restore previous graphical parameters
```

**Description**

Density, distribution function, quantile function, random generation and hazard function for the modified weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

```
dMW(x, mu, sigma, nu, log = FALSE)
```

```
pMW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
```

```
qMW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
```

```
rMW(n, mu, sigma, nu)
```

```
hMW(x, mu, sigma, nu)
```

**Arguments**

<code>x, q</code>	vector of quantiles.
<code>mu</code>	shape parameter one.
<code>sigma</code>	parameter two.
<code>nu</code>	scale parameter three.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

**Details**

The modified weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = \mu(\sigma + \nu x)x^{\sigma-1} \exp(\nu x) \exp(-\mu x^\sigma \exp(\nu x))$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma \geq 0$  and  $\nu \geq 0$ .

**Value**

`dMW` gives the density, `pMW` gives the distribution function, `qMW` gives the quantile function, `rMW` generates random deviates and `hMW` gives the hazard function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

- Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.
- Lai, C. D., Xie, M., & Murthy, D. N. P. (2003). A modified Weibull distribution. *IEEE Transactions on reliability*, 52(1), 33-37.

**See Also**[MW](#)**Examples**

```

old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dMW(x, mu=2, sigma=1.5, nu=0.2), from=0, to=2,
      ylim=c(0, 1.5), col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pMW(x, mu=2, sigma=1.5, nu=0.2), from=0, to=2,
      col = "red", las=1, ylab="F(x)")
curve(pMW(x, mu=2, sigma=1.5, nu=0.2, lower.tail = FALSE),
      from=0, to=2, col="red", las=1, ylab = "R(x)")

## The quantile function
p <- seq(from=0, to=0.9999, length.out=100)
plot(x=qMW(p, mu=2, sigma=1.5, nu=0.2), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pMW(x, mu=2, sigma=1.5, nu=0.2), from=0, add=TRUE, col="red")

## The random function
hist(rMW(n=1000, mu=2, sigma=1.5, nu=0.2), freq=FALSE,
     xlab="x", las=1, main="")
curve(dMW(x, mu=2, sigma=1.5, nu=0.2), from=0, add=TRUE, col="red")

## The Hazard function
par(mfrow=c(1,1))
curve(hMW(x, mu=2, sigma=1.5, nu=0.2), from=0, to=1.5, ylim=c(0, 5),
     col="red", las=1, ylab="H(x)", las=1)

par(old_par) # restore previous graphical parameters

```

dNEE

*The New Exponentiated Exponential distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the two-parameter New Exponentiated Exponential with parameters  $\mu$  and  $\sigma$ .

**Usage**

```
dNEE(x, mu = 1, sigma = 1, log = FALSE)
```

```
pNEE(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
```

```
qNEE(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
```

```
rNEE(n = 1, mu = 1, sigma = 1)
```

```
hNEE(x, mu, sigma, log = FALSE)
```

### Arguments

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

### Details

The New Exponentiated Exponential distribution with parameters  $\mu$  and  $\sigma$  has density given by

$$f(x|\mu, \sigma) = \log(2^\sigma)\mu \exp(-\mu x)(1 - \exp(-\mu x))^{\sigma-1}2^{(1-\exp(-\mu x))^\sigma},$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$ .

Note: In this implementation we changed the original parameters  $\theta$  for  $\mu$  and  $\alpha$  for  $\sigma$ , we did it to implement this distribution within gamlss framework.

### Value

dNEE gives the density, pNEE gives the distribution function, qNEE gives the quantile function, rNEE generates random deviates and hNEE gives the hazard function.

### Author(s)

Juliana Garcia, <juliana.garciav@udea.edu.co>

### References

Hassan, Anwar, I. H. Dar, and M. A. Lone. "A New Class of Probability Distributions With An Application to Engineering Data." *Pakistan Journal of Statistics and Operation Research* 20.2 (2024): 217-231.

### See Also

[NEE](#)

**Examples**

```

# Example 1
# Plotting the mass function for different parameter values
curve(dNEE(x, mu=0.2, sigma=0.3),
      from=0, to=8, col="cadetblue3", las=1, ylab="f(x)")

curve(dNEE(x, mu=1, sigma=4),
      add=TRUE, col="purple")

curve(dNEE(x, mu=1.5, sigma=22),
      add=TRUE, col="goldenrod")

curve(dNEE(x, mu=0.5, sigma=2),
      add=TRUE, col="green3")

legend("topright", col=c("cadetblue3", "purple", "goldenrod", "green3"), lty=1, bty="n",
      legend=c("mu=0.2, sigma=0.3",
              "mu=1.0, sigma=4",
              "mu=1.5, sigma=22",
              "mu=0.5, sigma=2"))

# Example 2
# Checking if the cumulative curves converge to 1
curve(pNEE(x, mu=0.2, sigma=0.3), ylim=c(0, 1),
      from=0, to=8, col="cadetblue3", las=1, ylab="F(x)")

curve(pNEE(x, mu=1, sigma=4),
      add=TRUE, col="purple")

curve(pNEE(x, mu=1.5, sigma=22),
      add=TRUE, col="goldenrod")

curve(pNEE(x, mu=0.5, sigma=2),
      add=TRUE, col="green3")

legend("bottomright", col=c("cadetblue3", "purple", "goldenrod", "green3"), lty=1, bty="n",
      legend=c("mu=0.2, sigma=0.3",
              "mu=1.0, sigma=4",
              "mu=1.5, sigma=22",
              "mu=0.5, sigma=2"))

# Example 3
# Checking the quantile function
mu <- 0.5
sigma <- 2
p <- seq(from=0, to=0.999, length.out=100)
plot(x=qNEE(p, mu=mu, sigma=sigma), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pNEE(x, mu=mu, sigma=sigma), from=0, add=TRUE, col="red")

# Example 4
# Comparing the random generator output with

```

```
# the theoretical probabilities
mu <- 0.5
sigma <- 2
x <- rNEE(n=10000, mu=mu, sigma=sigma)
hist(x, freq=FALSE)
curve(dNEE(x, mu=mu, sigma=sigma), col="tomato", add=TRUE)
```

---

dOW

*The Odd Weibull Distribution*


---

### Description

Density, distribution function, quantile function, random generation and hazard function for the Odd Weibull distribution with parameters mu, sigma and nu.

### Usage

```
dOW(x, mu, sigma, nu, log = FALSE)

pOW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qOW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rOW(n, mu, sigma, nu)

hOW(x, mu, sigma, nu)
```

### Arguments

x, q	vector of quantiles.
mu	parameter one.
sigma	parameter two.
nu	parameter three.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[T <= t], otherwise, P[T > t].
p	vector of probabilities.
n	number of observations.

### Details

The Odd Weibull with parameters mu, sigma and nu has density given by

$$f(x) = \left(\frac{\sigma\nu}{x}\right) (\mu x)^\sigma e^{(\mu x)^\sigma} (e^{(\mu x)^\sigma} - 1)^{\nu-1} \left[1 + (e^{(\mu x)^\sigma} - 1)^\nu\right]^{-2}$$

for  $x > 0$ .

**Value**

dOW gives the density, pOW gives the distribution function, qOW gives the quantile function, rOW generates random deviates and hOW gives the hazard function.

**Author(s)**

Jaime Mosquera Gutiérrez <jmosquerag@unal.edu.co>

**References**

Cooray, K. (2006). Generalization of the Weibull distribution: the odd Weibull family. *Statistical Modelling*, 6(3), 265-277.

**See Also**

[OW](#)

**Examples**

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dOW(x, mu=2, sigma=3, nu=0.2), from=0, to=4, ylim=c(0, 2),
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pOW(x, mu=2, sigma=3, nu=0.2), from=0, to=4, ylim=c(0, 1),
      col="red", las=1, ylab="F(x)")
curve(pOW(x, mu=2, sigma=3, nu=0.2, lower.tail=FALSE), from=0,
      to=4, ylim=c(0, 1), col="red", las=1,
      ylab = "R(x)")

## The quantile function
p <- seq(from=0, to=0.998, length.out=100)
plot(x = qOW(p, mu=2, sigma=3, nu=0.2), y=p, xlab="Quantile", las=1,
     ylab="Probability")
curve(pOW(x, mu=2, sigma=3, nu=0.2), from=0, add=TRUE, col="red")

## The random function
hist(rOW(n=10000, mu=2, sigma = 3, nu = 0.2), freq=FALSE, ylim = c(0, 2),
     xlab = "x", las = 1, main = "")
curve(dOW(x, mu=2, sigma=3, nu=0.2), from=0, ylim=c(0, 2), add=TRUE,
     col = "red")

## The Hazard function
par(mfrow=c(1,1))
curve(hOW(x, mu=2, sigma=3, nu=0.2), from=0, to=2.5, ylim=c(0, 3),
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters
```

---

dPL *The Power Lindley distribution*

---

### Description

Density, distribution function, quantile function, random generation and hazard function for the Power Lindley distribution with parameters  $\mu$  and  $\sigma$ .

### Usage

dPL(x, mu, sigma, log = FALSE)

pPL(q, mu, sigma, lower.tail = TRUE, log.p = FALSE)

qPL(p, mu, sigma, lower.tail = TRUE, log.p = FALSE)

rPL(n, mu, sigma)

hPL(x, mu, sigma)

### Arguments

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
p	vector of probabilities.
n	number of observations.

### Details

The Power Lindley Distribution with parameters  $\mu$  and  $\sigma$  has density given by

$$f(x) = \frac{\mu\sigma^2}{\sigma+1}(1+x^\mu)x^{\mu-1}\exp(-\sigma x^\mu),$$

for  $x > 0$ .

### Value

dPL gives the density, pPL gives the distribution function, qPL gives the quantile function, rPL generates random deviates and hPL gives the hazard function.

### Author(s)

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

## References

- Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.
- Ghitany, M. E., Al-Mutairi, D. K., Balakrishnan, N., & Al-Enezi, L. J. (2013). Power Lindley distribution and associated inference. *Computational Statistics & Data Analysis*, 64, 20-33.

## See Also

[PL](#)

## Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dPL(x, mu=1.5, sigma=0.2), from=0.1, to=10,
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pPL(x, mu=1.5, sigma=0.2),
      from=0.1, to=10, col="red", las=1, ylab="F(x)")
curve(pPL(x, mu=1.5, sigma=0.2, lower.tail=FALSE),
      from=0.1, to=10, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qPL(p, mu=1.5, sigma=0.2), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pPL(x, mu=1.5, sigma=0.2), from=0.1, add=TRUE, col="red")

## The random function
hist(rPL(n=1000, mu=1.5, sigma=0.2), freq=FALSE,
     xlab="x", las=1, main="")
curve(dPL(x, mu=1.5, sigma=0.2), from=0.1, to=15, add=TRUE, col="red")

## The Hazard function
par(mfrow=c(1,1))
curve(hPL(x, mu=1.5, sigma=0.2), from=0.1, to=15,
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters
```

## Description

Density, distribution function, quantile function, random generation and hazard function for the Quasi XGamma Poisson distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

```
dQXGP(x, mu, sigma, nu, log = FALSE)

pQXGP(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qQXGP(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rQXGP(n, mu, sigma, nu)

hQXGP(x, mu, sigma, nu)
```

**Arguments**

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
nu	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

**Details**

The Quasi XGamma Poisson distribution with parameters mu, sigma and nu has density given by:

$$f(x) = K(\mu, \sigma, \nu) \left( \frac{\sigma^2 x^2}{2} + \mu \right) \exp\left( \frac{\nu \exp(-\sigma x) (1 + \mu + \sigma x + \frac{\sigma^2 x^2}{2})}{1 + \mu} - \sigma x \right),$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu > 1$ .

where

$$K(\mu, \sigma, \nu) = \frac{\nu \sigma}{(\exp(\nu) - 1)(1 + \mu)}$$

**Value**

dQXGP gives the density, pQXGP gives the distribution function, qQXGP gives the quantile function, rQXGP generates random deviates and hQXGP gives the hazard function.

**Author(s)**

Simon Zapata

**References**

Sen, S., Korkmaz, M. Ç., & Yousof, H. M. (2018). The quasi XGamma-Poisson distribution: properties and application. *Istatistik Journal of The Turkish Statistical Association*, 11(3), 65-76.

**See Also**[QXGP](#)**Examples**

```

old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dQXGP(x, mu=0.5, sigma=1, nu=1), from=0.1, to=8,
      ylim=c(0, 0.6), col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pQXGP(x, mu=0.5, sigma=1, nu=1),
      from=0.1, to=8, col="red", las=1, ylab="F(x)")
curve(pQXGP(x, mu=0.5, sigma=1, nu=1, lower.tail=FALSE),
      from=0.1, to=8, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qQXGP(p, mu=0.5, sigma=1, nu=1), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pQXGP(x, mu=0.5, sigma=1, nu=1),
      from=0.1, add=TRUE, col="red")

## The random function
hist(rQXGP(n=1000, mu=0.5, sigma=1, nu=1), freq=FALSE,
     xlab="x", ylim=c(0, 0.4), las=1, main="", xlim=c(0, 15))
curve(dQXGP(x, mu=0.5, sigma=1, nu=1),
      from=0.001, to=500, add=TRUE, col="red")

## The Hazard function
curve(hQXGP(x, mu=0.5, sigma=1, nu=1), from=0.01, to=3,
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters

```

dRNMW

*The Reduced New Modified Weibull distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the reduced new modified Weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

```
dRNMW(x, mu, sigma, nu, log = FALSE)
```

```
pRNMW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
```

```
qRNMW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
```

```
rRNMW(n, mu, sigma, nu)
```

```
hRNMW(x, mu, sigma, nu)
```

### Arguments

x, q	vector of quantiles.
mu	parameter one.
sigma	parameter two.
nu	parameter three.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[T <= t], otherwise, P[T > t].
p	vector of probabilities.
n	number of observations.

### Details

The reduced new modified Weibull with parameters mu, sigma and nu has density given by

$$f(x) = \frac{1}{2\sqrt{x}} (\mu + \sigma(1 + 2\nu x)e^{\nu x}) e^{-\mu\sqrt{x} - \sigma\sqrt{x}e^{\nu x}}$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

### Value

dRNMW gives the density, pRNMW gives the distribution function, qRNMW gives the quantile function, rRNMW generates random deviates and hRNMW gives the hazard function.

### Author(s)

Jaime Mosquera, <jmosquera@unal.edu.co>

### References

Almalki, S. J. (2018). A reduced new modified Weibull distribution. Communications in Statistics-Theory and Methods, 47(10), 2297-2313.

**Examples**

```

old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(
  dRNMW(x, mu = 0.05, sigma = 0.00025, nu = 2.2),
  from = 0, to = 5, col = "red", las = 1, ylab = "f(x)"
)

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(
  pRNMW(x, mu = 0.05, sigma = 0.00025, nu = 2.2),
  from = 0, to = 5, ylim = c(0, 1),
  col = "red", las = 1, ylab = "F(x)"
)
curve(
  pRNMW(
    x, mu = 0.05, sigma = 0.00025, nu = 2.2,
    lower.tail = FALSE
  ),
  from = 0, to = 5, ylim = c(0, 1),
  col = "red", las = 1, ylab = "R(x)"
)

## The quantile function
p <- seq(from = 0, to = 0.999, length.out = 100)
plot(
  x = qRNMW(p, mu = 0.05, sigma = 0.00025, nu = 2.2),
  y = p, xlab = "Quantile", las = 1,
  ylab = "Probability"
)
curve(
  pRNMW(x, mu = 0.05, sigma = 0.00025, nu = 2.2),
  from = 0, add = TRUE, col = "red"
)

## The random function
hist(
  rRNMW(n = 10000, mu = 0.05, sigma = 0.00025, nu = 2.2),
  freq = FALSE, xlab = "x", las = 1, main = "", ylim = c(0,0.8)
)
curve(dRNMW(x, mu = 0.05, sigma = 0.00025, nu = 2.2),
  from = 0, add = TRUE, col = "red"
)

## The Hazard function
par(mfrow = c(1, 1))
curve(
  hRNMW(x, mu = 0.003, sigma = 5e-6, nu = 0.025),
  from = 0, to = 250, col = "red",
  ylab = "Hazard function", las = 1
)

```

```
)
par(old_par) # restore previous graphical parameters
```

---

dRW *The Reflected Weibull distribution*

---

### Description

Density, distribution function, quantile function, random generation and hazard function for the Reflected Weibull Distribution with parameters  $\mu$  and  $\sigma$ .

### Usage

```
dRW(x, mu, sigma, log = FALSE)
pRW(q, mu, sigma, lower.tail = TRUE, log.p = FALSE)
qRW(p, mu, sigma, lower.tail = TRUE, log.p = FALSE)
rRW(n, mu, sigma)
hRW(x, mu, sigma)
```

### Arguments

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

### Details

The Reflected Weibull Distribution with parameters  $\mu$  and  $\sigma$  has density given by

$$f(x) = \mu\sigma(-x)^{\sigma-1}e^{-\mu(-x)^\sigma},$$

for  $x < 0$ .

### Value

`dRW` gives the density, `pRW` gives the distribution function, `qRW` gives the quantile function, `rRW` generates random deviates and `hRW` gives the hazard function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. Reliability Engineering & System Safety, 124, 32-55.

Cohen, A. C. (1973). The reflected Weibull distribution. Technometrics, 15(4), 867-873.

**See Also**

[RW](#)

**Examples**

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dRW(x, mu=1, sigma=1), from=-5, to=-0.01,
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pRW(x, mu=1, sigma=1),
      from=-5, to=-0.01, col="red", las=1, ylab="F(x)")
curve(pRW(x, mu=1, sigma=1, lower.tail=FALSE),
      from=-5, to=-0.01, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qRW(p, mu=1, sigma=1), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pRW(x, mu=1, sigma=1), from=-5, add=TRUE, col="red")

## The random function
hist(rRW(n=10000, mu=1, sigma=1), freq=FALSE,
     xlab="x", las=1, main="")
curve(dRW(x, mu=1, sigma=1), from=-5, to=-0.01, add=TRUE, col="red")

## The Hazard function
par(mfrow=c(1,1))
curve(hRW(x, mu=1, sigma=1), from=-0.3, to=-0.01,
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters
```

**Description**

Density, distribution function, quantile function, random generation and hazard function for Sarhan and Zaindin's modified weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

```
dSZMW(x, mu, sigma, nu, log = FALSE)
pSZMW(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
qSZMW(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
rSZMW(n, mu, sigma, nu)
hSZMW(x, mu, sigma, nu)
```

**Arguments**

<code>x, q</code>	vector of quantiles.
<code>mu</code>	scale parameter.
<code>sigma</code>	shape parameter.
<code>nu</code>	shape parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

**Details**

The Sarhan and Zaindin's modified weibull with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = (\mu + \sigma \nu x^{\nu-1}) \exp(-\mu x - \sigma x^\nu)$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

**Value**

dSZMW gives the density, pSZMW gives the distribution function, qSZMW gives the quantile function, rSZMW generates random deviates and hSZMW gives the hazard function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

## References

- Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.
- Sarhan, A. M., & Zaindin, M. (2009). Modified Weibull distribution. *APPS. Applied Sciences*, 11, 123-136.

## See Also

[SZMW](#)

## Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dSZMW(x, mu = 2, sigma = 1.5, nu = 0.2), from = 0, to = 2,
      ylim = c(0, 1.7), col = "red", las = 1, ylab = "f(x)")
## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pSZMW(x, mu = 2, sigma = 1.5, nu = 0.2), from = 0, to = 2, ylim = c(0, 1),
      col = "red", las = 1, ylab = "F(x)")
curve(pSZMW(x, mu = 2, sigma = 1.5, nu = 0.2, lower.tail = FALSE), from = 0,
      to = 2, ylim = c(0, 1), col = "red", las = 1, ylab = "R(x)")

## The quantile function
p <- seq(from = 0, to = 0.99999, length.out = 100)
plot(x = qSZMW(p = p, mu = 2, sigma = 1.5, nu = 0.2), y = p, xlab = "Quantile",
     las = 1, ylab = "Probability")
curve(pSZMW(x, mu = 2, sigma = 1.5, nu = 0.2), from = 0, add = TRUE, col = "red")

## The random function
hist(rSZMW(n = 1000, mu = 2, sigma = 1.5, nu = 0.2), freq = FALSE, xlab = "x",
     las = 1, main = "")
curve(dSZMW(x, mu = 2, sigma = 1.5, nu = 0.2), from = 0, add = TRUE, col = "red")

## The Hazard function
par(mfrow=c(1,1))
curve(hSZMW(x, mu = 2, sigma = 1.5, nu = 0.2), from = 0, to = 3, ylim = c(0, 8),
     col = "red", ylab = "Hazard function", las = 1)

par(old_par) # restore previous graphical parameters
```

---

dWALD

*The Wald distribution*

---

## Description

These functions define the density, distribution function, quantile function and random generation for the Wald distribution with parameter  $\mu$  and  $\sigma$ .

**Usage**

dWALD(x, mu, sigma, log = FALSE)

pWALD(q, mu, sigma, lower.tail = TRUE, log.p = FALSE)

qWALD(p, mu, sigma, lower.tail = TRUE, log.p = FALSE)

rWALD(n, mu, sigma)

**Arguments**

x, q	vector of (non-negative integer) quantiles.
mu	vector of the mu parameter.
sigma	vector of the sigma parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of random values to return.

**Details**

The Wald distribution with parameters  $\mu$  and  $\sigma$  has density given by

$$f(x|\mu, \sigma) = \frac{\sigma}{\sqrt{2\pi x^3}} \exp\left[-\frac{(\sigma - \mu x)^2}{2x}\right],$$

for  $x < 0$ .

**Value**

dWALD gives the density, pWALD gives the distribution function, qWALD gives the quantile function, rWALD generates random deviates.

**Author(s)**

Sofia Cuartas, <scuartas@unal.edu.co>

**References**

Heathcote, A. (2004). Fitting Wald and ex-Wald distributions to response time data: An example using functions for the S-PLUS package. Behavior Research Methods, Instruments, & Computers, 36, 678-694.

**See Also**

[WALD](#)

[WALD](#).

**Examples**

```

# Example 1
# Plotting the mass function for different parameter values
curve(dWALD(x, mu=1, sigma=1),
      from=0, to=3, col="cadetblue3", las=1, ylab="f(x)")

curve(dWALD(x, mu=1, sigma=2),
      add=TRUE, col= "purple")

curve(dWALD(x, mu=2, sigma=4),
      add=TRUE, col="goldenrod")

legend("topright", col=c("cadetblue3", "purple", "goldenrod"),
      lty=1, bty="n",
      legend=c("mu=1, sigma=1",
              "mu=1, sigma=2",
              "mu=2, sigma=4"))

# Example 2
# Checking if the cumulative curves converge to 1
curve(pWALD(x, mu=1, sigma=1), ylim=c(0, 1),
      from=0, to=5, col="cadetblue3", las=1, ylab="F(x)")

curve(pWALD(x, mu=1, sigma=2),
      add=TRUE, col= "purple")

curve(pWALD(x, mu=2, sigma=4),
      add=TRUE, col="goldenrod")

legend("bottomright", col=c("cadetblue3", "purple", "goldenrod"),
      lty=1, bty="n",
      legend=c("mu=1, sigma=1",
              "mu=1, sigma=2",
              "mu=2, sigma=4"))

# Example 3
# Checking the quantile function
mu <- 1
sigma <- 2
p <- seq(from=0, to=0.999, length.out=100)
plot(x=qWALD(p, mu=mu, sigma=sigma), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pWALD(x, mu=mu, sigma=sigma), from=0, add=TRUE, col="red")

# Example 4
# Comparing the random generator output with
# the theoretical probabilities
mu <- 1
sigma <- 20
x <- rWALD(n=10000, mu=mu, sigma=sigma)
hist(x, freq=FALSE)
curve(dWALD(x, mu=mu, sigma=sigma), col="tomato", add=TRUE)

```

---

dWG

*The Weibull Geometric distribution*


---

### Description

Density, distribution function, quantile function, random generation and hazard function for the weibull geometric distribution with parameters mu, sigma and nu.

### Usage

```
dWG(x, mu, sigma, nu, log = FALSE)
```

```
pWG(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
```

```
qWG(p, sigma, mu, nu, lower.tail = TRUE, log.p = FALSE)
```

```
rWG(n, mu, sigma, nu)
```

```
hWG(x, mu, sigma, nu)
```

### Arguments

x, q	vector of quantiles.
mu	scale parameter.
sigma	shape parameter.
nu	parameter of geometric random variable.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

### Details

The Weibull geometric distribution with parameters mu, sigma and nu has density given by

$$f(x) = (\sigma\mu^\sigma(1-\nu)x^{\sigma-1}\exp(-(\mu x)^\sigma))(1-\nu\exp(-(\mu x)^\sigma))^{-2},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $0 < \nu < 1$ .

### Value

dWG gives the density, pWG gives the distribution function, qWG gives the quantile function, rWG generates random deviates and hWG gives the hazard function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

Barreto-Souza, W., de Morais, A. L., & Cordeiro, G. M. (2011). The Weibull-geometric distribution. *Journal of Statistical Computation and Simulation*, 81(5), 645-657.

**See Also**

[WG](#)

**Examples**

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dWG(x, mu = 0.9, sigma = 2, nu = 0.5), from = 0, to = 3,
      ylim = c(0, 1.1), col = "red", las = 1, ylab = "f(x)")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pWG(x, mu = 0.9, sigma = 2, nu = 0.5), from = 0, to = 3,
      ylim = c(0, 1), col = "red", las = 1, ylab = "F(x)")
curve(pWG(x, mu = 0.9, sigma = 2, nu = 0.5, lower.tail = FALSE),
      from = 0, to = 3, ylim = c(0, 1), col = "red", las = 1, ylab = "R(x)")

## The quantile function
p <- seq(from = 0, to = 0.99999, length.out = 100)
plot(x = qWG(p = p, mu = 0.9, sigma = 2, nu = 0.5), y = p,
     xlab = "Quantile", las = 1, ylab = "Probability")
curve(pWG(x, mu = 0.9, sigma = 2, nu = 0.5), from = 0, add = TRUE,
      col = "red")

## The random function
hist(rWG(1000, mu = 0.9, sigma = 2, nu = 0.5), freq = FALSE, xlab = "x",
     ylim = c(0, 1.8), las = 1, main = "")
curve(dWG(x, mu = 0.9, sigma = 2, nu = 0.5), from = 0, add = TRUE,
      col = "red", ylim = c(0, 1.8))

## The Hazard function(
par(mfrow=c(1,1))
curve(hWG(x, mu = 0.9, sigma = 2, nu = 0.5), from = 0, to = 8,
      ylim = c(0, 12), col = "red", ylab = "Hazard function", las = 1)

par(old_par) # restore previous graphical parameters
```

dWGEE

*The Weighted Generalized Exponential-Exponential distribution***Description**

Density, distribution function, quantile function, random generation and hazard function for the Weighted Generalized Exponential-Exponential distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

```
dWGEE(x, mu, sigma, nu, log = FALSE)
pWGEE(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
qWGEE(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)
rWGEE(n, mu, sigma, nu)
hWGEE(x, mu, sigma, nu)
```

**Arguments**

<code>x, q</code>	vector of quantiles.
<code>mu</code>	parameter.
<code>sigma</code>	parameter.
<code>nu</code>	parameter.
<code>log, log.p</code>	logical; if TRUE, probabilities $p$ are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$ .
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations.

**Details**

The Weighted Generalized Exponential-Exponential Distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = \sigma\nu \exp(-\nu x)(1 - \exp(-\nu x))^{\sigma-1}(1 - \exp(-\mu\nu x))/1 - \sigma B(\mu + 1, \sigma),$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

**Value**

dWGEE gives the density, pWGEE gives the distribution function, qWGEE gives the quantile function, rWGEE generates random deviates and hWGEE gives the hazard function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

## References

Mahdavi, A. (2015). Two weighted distributions generated by exponential distribution. *Journal of Mathematical Extension*, 9, 1-12.

## See Also

[WGEE](#)

## Examples

```
old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dWGEE(x, mu = 5, sigma = 0.5, nu = 1), from = 0, to = 6,
      ylim = c(0, 1), col = "red", las = 1, ylab = "The probability density function")

## The cumulative distribution and the Reliability function
par(mfrow = c(1, 2))
curve(pWGEE(x, mu = 5, sigma = 0.5, nu = 1), from = 0, to = 6,
      ylim = c(0, 1), col = "red", las = 1, ylab = "The cumulative distribution function")
curve(pWGEE(x, mu = 5, sigma = 0.5, nu = 1, lower.tail = FALSE),
      from = 0, to = 6, ylim = c(0, 1), col = "red", las = 1, ylab = "The Reliability function")

## The quantile function
p <- seq(from = 0, to = 0.99999, length.out = 100)
plot(x = qWGEE(p = p, mu = 5, sigma = 0.5, nu = 1), y = p,
     xlab = "Quantile", las = 1, ylab = "Probability")
curve(pWGEE(x, mu = 5, sigma = 0.5, nu = 1), from = 0, add = TRUE,
      col = "red")

## The random function
hist(rWGEE(1000, mu = 5, sigma = 0.5, nu = 1), freq = FALSE, xlab = "x",
     ylim = c(0, 1), las = 1, main = "")
curve(dWGEE(x, mu = 5, sigma = 0.5, nu = 1), from = 0, add = TRUE,
      col = "red", ylim = c(0, 1))

## The Hazard function(
par(mfrow=c(1,1))
curve(hWGEE(x, mu = 5, sigma = 0.5, nu = 1), from = 0, to = 6,
      ylim = c(0, 1.4), col = "red", ylab = "The hazard function", las = 1)

par(old_par) # restore previous graphical parameters
```

## Description

Density, distribution function, quantile function, random generation and hazard function for the Weibull Poisson distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$ .

**Usage**

dWP(x, mu, sigma, nu, log = FALSE)

pWP(q, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

qWP(p, mu, sigma, nu, lower.tail = TRUE, log.p = FALSE)

rWP(n, mu, sigma, nu)

hWP(x, mu, sigma, nu)

**Arguments**

x, q	vector of quantiles.
mu	parameter.
sigma	parameter.
nu	parameter.
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x].
p	vector of probabilities.
n	number of observations.

**Details**

The Weibull Poisson distribution with parameters mu, sigma and nu has density given by

$$f(x) = \frac{\mu\sigma\nu e^{-\nu}}{1-e^{-\nu}} x^{\mu-1} \exp(-\sigma x^{\mu} + \nu \exp(-\sigma x^{\mu})),$$

for  $x > 0$ .

**Value**

dWP gives the density, pWP gives the distribution function, qWP gives the quantile function, rWP generates random deviates and hWP gives the hazard function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Lu, Wanbo, and Daimin Shi. "A new compounding life distribution: the Weibull–Poisson distribution." *Journal of applied statistics* 39.1 (2012): 21-38.

**See Also**

[WP](#)

**Examples**

```

old_par <- par(mfrow = c(1, 1)) # save previous graphical parameters

## The probability density function
curve(dWP(x, mu=1.5, sigma=0.5, nu=10), from=0.0001, to=2,
      col="red", las=1, ylab="f(x)")

## The cumulative distribution and the Reliability function
par(mfrow=c(1, 2))
curve(pWP(x, mu=1.5, sigma=0.5, nu=10),
      from=0.0001, to=2, col="red", las=1, ylab="F(x)")
curve(pWP(x, mu=1.5, sigma=0.5, nu=10, lower.tail=FALSE),
      from=0.0001, to=2, col="red", las=1, ylab="R(x)")

## The quantile function
p <- seq(from=0, to=0.99999, length.out=100)
plot(x=qWP(p, mu=1.5, sigma=0.5, nu=10), y=p, xlab="Quantile",
     las=1, ylab="Probability")
curve(pWP(x, mu=1.5, sigma=0.5, nu=10),
      from=0, add=TRUE, col="red")

## The random function
hist(rWP(n=10000, mu=1.5, sigma=0.5, nu=10), freq=FALSE,
     xlab="x", ylim=c(0, 2.2), las=1, main="")
curve(dWP(x, mu=1.5, sigma=0.5, nu=10),
      from=0.001, to=4, add=TRUE, col="red")

## The Hazard function
curve(hWP(x, mu=1.5, sigma=0.5, nu=10), from=0.001, to=5,
     col="red", ylab="Hazard function", las=1)

par(old_par) # restore previous graphical parameters

```

---

 EEG

*The Extended Exponential Geometric family*


---

**Description**

The Extended Exponential Geometric family

**Usage**

```
EEG(mu.link = "log", sigma.link = "log")
```

**Arguments**

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.

**Details**

The Extended Exponential Geometric distribution with parameters  $\mu$  and  $\sigma$  has density given by

$$f(x) = \mu\sigma \exp(-\mu x)(1 - (1 - \sigma) \exp(-\mu x))^{-2},$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a EEG distribution in the `gamlss()` function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Adamidis, K., Dimitrakopoulou, T., & Loukas, S. (2005). On an extension of the exponential-geometric distribution. *Statistics & probability letters*, 73(3), 259-269.

**See Also**

[dEEG](#)

**Examples**

```
# Generating some random values with
# known mu, sigma, nu and tau
y <- rEEG(n=100, mu = 1, sigma =1.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, family=EEG,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma, nu and tau
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.1, max=0.2)
x2 <- runif(n, min=0.1, max=0.15)
mu <- exp(0.75 - x1)
sigma <- exp(0.5 - x2)
x <- rEEG(n=n, mu, sigma)
```

```

mod <- gamlss(x~x1, sigma.fo=~x2, family=EGG,
             control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")

```

EGG

*The four parameter Exponentiated Generalized Gamma family***Description**

The four parameter Exponentiated Generalized Gamma distribution

**Usage**

```
EGG(mu.link = "log", sigma.link = "log", nu.link = "log", tau.link = "log")
```

**Arguments**

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the mu parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the sigma.
<code>nu.link</code>	defines the <code>nu.link</code> , with "log" link as the default for the nu parameter.
<code>tau.link</code>	defines the <code>tau.link</code> , with "log" link as the default for the tau parameter.

**Details**

Four parameter Exponentiated Generalized Gamma distribution with parameters  $\mu$ ,  $\sigma$ ,  $\nu$  and  $\tau$  has density given by

$$f(x) = \frac{\nu\sigma}{\mu\Gamma(\tau)} \left(\frac{x}{\mu}\right)^{\sigma\tau-1} \exp\left\{-\left(\frac{x}{\mu}\right)^\sigma\right\} \left\{\gamma_1\left(\tau, \left(\frac{x}{\mu}\right)^\sigma\right)\right\}^{\nu-1},$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a EGG distribution in the `gamlss()` function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

- Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.
- Cordeiro, G. M., Ortega, E. M., & Silva, G. O. (2011). The exponentiated generalized gamma distribution with application to lifetime data. *Journal of statistical computation and simulation*, 81(7), 827-842.

**See Also**[dEGG](#)**Examples**

```

# Example 1
# Generating some random values with
# known mu, sigma, nu and tau

set.seed(123456)
y <- rEGG(n=100, mu=0.1, sigma=0.8, nu=10, tau=1.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, tau.fo=~1,
              family=EGG,
              control=gamlss.control(n.cyc=500, trace=FALSE))

# Extracting the fitted values for mu, sigma, nu and tau
# using the inverse link function
exp(coef(mod, what="mu"))
exp(coef(mod, what="sigma"))
exp(coef(mod, what="nu"))
exp(coef(mod, what="tau"))

# Example 2
# Generating random values under some model

# A function to simulate a data set with  $Y \sim \text{EGG}$ 
gendat <- function(n) {
  x1 <- runif(n)
  x2 <- runif(n)
  mu   <- exp(-0.8 + -3 * x1)
  sigma <- exp(0.77 - 2 * x2)
  nu   <- 10
  tau  <- 1.5
  y <- rEGG(n=n, mu=mu, sigma=sigma, nu=nu, tau)
  data.frame(y=y, x1=x1, x2=x2)
}

set.seed(12345)
dat <- gendat(n=200)

mod <- gamlss(y~x1, sigma.fo=~x2, nu.fo=~1, tau.fo=~1,
              family=EGG, data=dat,
              control=gamlss.control(n.cyc=500, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")

```

```
exp(coef(mod, what="nu"))
exp(coef(mod, what="tau"))
```

EMWEx

*The Exponentiated Modified Weibull Extension family***Description**

The Exponentiated Modified Weibull Extension family

**Usage**

```
EMWEx(mu.link = "log", sigma.link = "log", nu.link = "log", tau.link = "log")
```

**Arguments**

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the mu parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the sigma.
<code>nu.link</code>	defines the <code>nu.link</code> , with "log" link as the default for the nu parameter.
<code>tau.link</code>	defines the <code>tau.link</code> , with "log" link as the default for the tau parameter.

**Details**

The Beta-Weibull distribution with parameters `mu`, `sigma`, `nu` and `tau` has density given by

$$f(x) = \nu\sigma\tau\left(\frac{x}{\mu}\right)^{\sigma-1} \exp\left(\left(\frac{x}{\mu}\right)^{\sigma} + \nu\mu(1 - \exp\left(\left(\frac{x}{\mu}\right)^{\sigma}\right))\right)(1 - \exp(\nu\mu(1 - \exp\left(\left(\frac{x}{\mu}\right)^{\sigma}\right))))^{\tau-1},$$

for  $x > 0$ ,  $\nu > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\tau > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a EMWEx distribution in the `gamlss()` function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Sarhan, A. M., & Apaloo, J. (2013). Exponentiated modified Weibull extension distribution. *Reliability Engineering & System Safety*, 112, 137-144.

**See Also**

[dEMWEx](#)

**Examples**

```

# Example 1
# Generating some random values with
# known mu, sigma, nu and tau
y <- rEMWEx(n=100, mu = 1, sigma =1.21, nu=1, tau=2)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, tau.fo=~1, family=EMWEx,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma, nu and tau
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))
exp(coef(mod, what='tau'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(0.75 - x1)
sigma <- exp(0.5 - x2)
nu <- 1
tau <- 2
x <- rEMWEx(n=n, mu, sigma, nu, tau)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, tau.fo=~1, family=EMWEx,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))
exp(coef(mod, what="tau"))

```

**Description**

The Extended Odd Frechet-Nadarjad-Hanhighi family

**Usage**

```
EOFNH(mu.link = "log", sigma.link = "log", nu.link = "log", tau.link = "log")
```

**Arguments**

mu.link	defines the mu.link, with "log" link as the default for the mu parameter.
sigma.link	defines the sigma.link, with "log" link as the default for the sigma.
nu.link	defines the nu.link, with "log" link as the default for the nu parameter.
tau.link	defines the tau.link, with "log" link as the default for the tau parameter.

**Details**

The Extended Odd Fréchet-Nadarajah-Haghighi distribution with parameters mu, sigma, nu and tau has density given by

$$f(x) = \frac{\mu\sigma\nu\tau(1+\nu x)^{\sigma-1}e^{-(1+(1+\nu x)^\sigma)}[1-(1-e^{-(1+(1+\nu x)^\sigma)})^\mu]^{\tau-1}}{(1-e^{-(1+(1+\nu x)^\sigma)})^{\mu\tau+1}}e^{-[(1-e^{-(1+(1+\nu x)^\sigma)})^{-\mu}-1]^\tau},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu > 0$  and  $\tau > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a EOFNH distribution in the `gamlss()` function.

**Author(s)**

Helber Santiago Padilla, <hspadillar@unal.edu.co>

**References**

Nasiru, S. (2018). Extended Odd Fréchet-G Family of Distributions Journal of Probability and Statistics, 2018(1), 2931326.

**See Also**

[dEOFNH](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma, nu and tau
set.seed(123)
y <- rEOFNH(n=100, mu=1, sigma=2.1, nu=0.8, tau=1)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, tau.fo=~1, family=EOFNH,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma, nu and tau
# using the inverse link function
exp(coef(mod, what="mu"))
exp(coef(mod, what="sigma"))
```

```
exp(coef(mod, what="nu"))
exp(coef(mod, what="tau"))

# Example 2
# Generating random values under the model
n <- 100
x1 <- runif(n)
x2 <- runif(n)
mu <- exp(0.5 - 1.2 * x1)
sigma <- 2.1
nu <- 0.8
tau <- 1
y <- rEOFNH(n=n, mu, sigma, nu, tau)

mod <- gamlss(y~x1, sigma.fo=~1, nu.fo=~1, tau.fo=~1, family=EOFNH,
             control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
exp(coef(mod, what="sigma"))
exp(coef(mod, what="nu"))
exp(coef(mod, what="tau"))
```

---

equipment

*Electronic equipment data*

---

### **Description**

Time to failure in hours of 18 units of the same electronic device.

### **Usage**

```
data(equipment)
```

### **Format**

A vector with 18 observations.

### **Examples**

```
data(equipment)
hist(equipment, main="", xlab="Time (h)")
```

EW

*The Exponentiated Weibull family***Description**

The Exponentiated Weibull distribution

**Usage**

```
EW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> .
<code>nu.link</code>	defines the <code>nu.link</code> , with "log" link as the default for the <code>nu</code> parameter.

**Details**

The Exponentiated Weibull Distribution with parameters `mu`, `sigma` and `nu` has density given by

$$f(x) = \nu\mu\sigma x^{\sigma-1} \exp(-\mu x^\sigma)(1 - \exp(-\mu x^\sigma))^{\nu-1},$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a EW distribution in the `gamlss()` function.

**See Also**

[dEW](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
# Will not be run this example because high number is cycles
# is needed in order to get good estimates
## Not run:
y <- rEW(n=100, mu=2, sigma=1.5, nu=0.5)

# Fitting the model
require(gamlss)
mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='EW',
             control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
```

```

exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

## End(Not run)

# Example 2
# Generating random values under some model
# Will not be run this example because high number is cycles
# is needed in order to get good estimates
## Not run:
n <- 200
x1 <- rpois(n, lambda=2)
x2 <- runif(n)
mu <- exp(2 + -3 * x1)
sigma <- exp(3 - 2 * x2)
nu <- 2
x <- rEW(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, family=EW,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))

## End(Not run)

```

---

EXL

---

*The exponentiated XLindley family*


---

### Description

The function `EXL()` defines The exponentiated XLindley, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

### Usage

```
EXL(mu.link = "log", sigma.link = "log")
```

### Arguments

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> .

**Details**

The exponentiated XLindley with parameters  $\mu$  and  $\sigma$  has density given by

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

for  $x \geq 0$ ,  $\mu \geq 0$  and  $\sigma \geq 0$ .

Note: In this implementation we changed the original parameters  $\delta$  for  $\mu$  and  $\alpha$  for  $\sigma$  we did it to implement this distribution within gamlss framework.

**Value**

Returns a gamlss.family object which can be used to fit a EXL distribution in the gamlss() function.

**Author(s)**

Manuel Gutierrez Tangarife, <mgutierrez@unal.edu.co>

**References**

Alomair, A. M., Ahmed, M., Tariq, S., Ahsan-ul-Haq, M., & Talib, J. (2024). An exponentiated XLindley distribution with properties, inference and applications. *Heliyon*, 10(3).

**See Also**

[EXL](#).

**Examples**

```
# Example 1
# Generating some random values with
# known mu and sigma
y <- rEXL(n=300, mu=0.75, sigma=1.3)

# Fitting the model
require(gamlss)
mod1 <- gamlss(y~1, sigma.fo=~1, family=EXL,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod1, what="mu"))
exp(coef(mod1, what="sigma"))

# Example 2
# Generating random values for a regression model

# A function to simulate a data set with Y ~ EXL
gendat <- function(n) {
  x1 <- runif(n)
```

```

x2 <- runif(n)
mu <- exp(1.45 - 3 * x1)
sigma <- exp(2 - 1.5 * x2)
y <- rEXL(n=n, mu=mu, sigma=sigma)
data.frame(y=y, x1=x1, x2=x2)
}

set.seed(1234)
dat <- gendat(n=100)

mod2 <- gamlss(y~x1, sigma.fo=~x2,
              family=EXL, data=dat,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

summary(mod2)

# Example 3
# Mortality rate due to COVID-19 for 30 days (31st March to April 30, 2020)
# recorded for the Netherlands.
# Taken from Alomair et al. (2024) page 12.

x <- c(14.918, 10.656, 12.274, 10.289, 10.832, 7.099, 5.928, 13.211,
       7.968, 7.584, 5.555, 6.027, 4.097, 3.611, 4.960, 7.498, 6.940,
       5.307, 5.048, 2.857, 2.254, 5.431, 4.462, 3.883,
       3.461, 3.647, 1.974, 1.273, 1.416, 4.235)

mod3 <- gamlss(x~1, sigma.fo=~1, family=EXL,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod3, what="mu"))
exp(coef(mod3, what="sigma"))

# Replicating figure 4 from Alomair et al. (2024)
# Hist and estimated pdf
hist(x, freq=FALSE)
curve(dEXL(x, mu=0.4089915, sigma=2.710467), add=TRUE,
      col="tomato", lwd=2)
# Empirical cdf and estimated ecdf
plot(ecdf(x))
curve(pEXL(x, mu=0.4089915, sigma=2.710467), add=TRUE,
      col="tomato", lwd=2)
# QQplot
qqplot(x, rEXL(n=30, mu=0.4089915, sigma=2.710467),
       col="tomato")
qqline(x, distribution=function(p) qEXL(p, mu=0.4089915, sigma=2.710467))

# Example 4
# Precipitation in inches
# Taken from Alomair et al. (2024) page 13.

```

# Manuel

---

ExW

*The Extended Weibull family*

---

## Description

The Extended Weibull family

## Usage

```
ExW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

## Arguments

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> .
<code>nu.link</code>	defines the <code>nu.link</code> , with "log" link as the default for the <code>nu</code> parameter.

## Details

The Extended Weibull distribution with parameters `mu`, `sigma` and `nu` has density given by

$$f(x) = \frac{\mu\sigma\nu x^{\sigma-1} \exp(-\mu x^\sigma)}{[1-(1-\nu)\exp(-\mu x^\sigma)]^2},$$

for  $x > 0$ .

## Value

Returns a `gamlss.family` object which can be used to fit a ExW distribution in the `gamlss()` function.

## Author(s)

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

## References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Zhang, T., & Xie, M. (2007). Failure data analysis with extended Weibull distribution. *Communications in Statistics—Simulation and Computation*, 36(3), 579-592.

## See Also

[dExW](#)

**Examples**

```

# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rExW(n=200, mu=0.3, sigma=2, nu=0.05)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='ExW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 500
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(-2 + 3 * x1)
sigma <- exp(1.3 - 2 * x2)
nu <- 0.05
x <- rExW(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, family=ExW,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))

```

---

ExWALD

*The Ex-Wald family*


---

**Description**

The function `ExWALD()` defines the Ex-wALD distribution, three-parameter continuous distribution for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

**Usage**

```
ExWALD(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter.
<code>nu.link</code>	defines the <code>nu.link</code> , with "log" link as the default for the <code>nu</code> parameter.

**Details**

The Ex-Wald distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x|\mu, \sigma, \nu) = \frac{1}{\nu} \exp\left(\frac{-x}{\nu} + \sigma(\mu - k)\right) F_W(x|k, \sigma) \text{ for } k \geq 0$$

$$f(x|\mu, \sigma, \nu) = \frac{1}{\nu} \exp\left(\frac{-(\sigma - \mu)^2}{2x}\right) Re\left(w(k' \sqrt{x/2} + \frac{\sigma_i}{\sqrt{2x}})\right) \text{ for } k < 0$$

where  $k = \sqrt{\mu^2 - \frac{2}{\nu}}$ ,  $k' = \sqrt{\frac{2}{\nu} - \mu^2}$  and  $F_W$  corresponds to the cumulative function of the Wald distribution.

More details about those expressions can be found on page 680 from Heathcote (2004).

**Value**

Returns a `gamlss.family` object which can be used to fit a Ex-WALD distribution in the `gamlss()` function.

**Author(s)**

Freddy Hernandez, <fhernanb@unal.edu.co>

**References**

Schwarz, W. (2001). The ex-Wald distribution as a descriptive model of response times. *Behavior Research Methods, Instruments, & Computers*, 33, 457-469.

Heathcote, A. (2004). Fitting Wald and ex-Wald distributions to response time data: An example using functions for the S-PLUS package. *Behavior Research Methods, Instruments, & Computers*, 36, 678-694.

**See Also**

[dExWALD](#).

**Examples**

```
# Example 1
# Generating random values with
# known mu, sigma and nu

mu <- 0.20
sigma <- 70
nu <- 115

set.seed(123)
y <- rExWALD(n=100, mu, sigma, nu)
```

```

library(gamlss)
mod1 <- gamlss(y~1, family=ExWALD,
              control=gamlss.control(n.cyc=1000, trace=TRUE))

exp(coef(mod1, what="mu"))
exp(coef(mod1, what="sigma"))
exp(coef(mod1, what="nu"))

# Example 2
# Generating random values under some model

# A function to simulate a data set with Y ~ ExWALD
gendat <- function(n) {
  x1 <- runif(n)
  x2 <- runif(n)
  mu   <- exp(-1 + 2.8 * x1) # 1.5 approximately
  sigma <- exp( 1 - 1.2 * x2) # 1.5 approximately
  nu   <- 2
  y <- rExWALD(n=n, mu=mu, sigma=sigma, nu=nu)
  data.frame(y=y, x1=x1, x2=x2)
}

set.seed(1234)
dat <- gendat(n=200)

# Fitting the model
mod2 <- gamlss(y ~ x1,
              sigma.fo = ~ x2,
              nu.fo = ~ 1,
              family = ExWALD,
              data = dat,
              control = gamlss.control(n.cyc=1000,
                                      trace=TRUE))

summary(mod2)

```

### Description

The function `FWE()` defines the Flexible Weibull distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

### Usage

```
FWE(mu.link = "log", sigma.link = "log")
```

**Arguments**

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.

**Details**

The Flexible Weibull extension with parameters `mu` and `sigma` has density given by

$$f(x) = (\mu + \sigma/x^2) \exp(\mu x - \sigma/x) \exp(-\exp(\mu x - \sigma/x))$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a FWE distribution in the `gamlss()` function.

**Examples**

```
# Example 1
# Generating some random values with
# known mu and sigma
y <- rFWE(n=100, mu=0.75, sigma=1.3)

# Fitting the model
require(gamlss)
mod <- gamlss(y~1, sigma.fo=~1, family='FWE',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n)
x2 <- runif(n)
mu <- exp(1.21 - 3 * x1)
sigma <- exp(1.26 - 2 * x2)
x <- rFWE(n=n, mu, sigma)

mod <- gamlss(x~x1, sigma.fo=~x2, family=FWE,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
```

---

GammaW

*The Gamma Weibull family*

---

## Description

The Gamma Weibull family

## Usage

```
GammaW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

## Arguments

`mu.link` defines the `mu.link`, with "log" link as the default for the mu parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the sigma.  
`nu.link` defines the `nu.link`, with "log" link as the default for the nu parameter.

## Details

The Gamma Weibull distribution with parameters `mu`, `sigma` and `nu` has density given by

$$f(x) = \frac{\sigma \mu^\nu}{\Gamma(\nu)} x^{\nu\sigma-1} \exp(-\mu x^\sigma),$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma \geq 0$  and  $\nu > 0$ .

## Value

Returns a `gamlss.family` object which can be used to fit a GammaW distribution in the `gamlss()` function.

## Author(s)

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

## References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Stacy, E. W. (1962). A generalization of the gamma distribution. *The Annals of mathematical statistics*, 1187-1192.

## See Also

[dGammaW](#)

**Examples**

```

# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rGammaW(n=100, mu = 0.5, sigma = 2, nu=1)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='GammaW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n)
x2 <- runif(n)
mu <- exp(-1.6 * x1)
sigma <- exp(1.1 - 1 * x2)
nu <- 1
x <- rGammaW(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, mu.fo=~x1, sigma.fo=~x2, nu.fo=~1, family=GammaW,
              control=gamlss.control(n.cyc=50000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
coef(mod, what='nu')

```

GGD

*The Generalized Gompertz family***Description**

The Generalized Gompertz family

**Usage**

```
GGD(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.

sigma.link defines the sigma.link, with "log" link as the default for the sigma.  
 nu.link defines the nu.link, with "log" link as the default for the nu parameter.

### Details

The Generalized Gompertz Distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = \nu\mu \exp\left(-\frac{\mu}{\sigma}(\exp(\sigma x - 1))\right) \left(1 - \exp\left(-\frac{\mu}{\sigma}(\exp(\sigma x - 1))\right)\right)^{(\nu-1)},$$

for  $x \geq 0$ ,  $\mu > 0$ ,  $\sigma \geq 0$  and  $\nu > 0$

### Value

Returns a `gamlss.family` object which can be used to fit a GGD distribution in the `gamlss()` function.

### Author(s)

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

### References

# El-Gohary, A., Alshamrani, A., & Al-Otaibi, A. N. (2013). The generalized Gompertz distribution. *Applied mathematical modelling*, 37(1-2), 13-24.

### See Also

[dGGD](#)

### Examples

```
#Example 1
# Generating some random values with
# known mu, sigma, nu and tau
y <- rGGD(n=1000, mu=1, sigma=0.3, nu=1.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='GGD',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
```

```

mu <- exp(0.5 - x1)
sigma <- exp(-1 - x2)
nu <- 1.5
x <- rGGD(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, family=GGD,
             control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))

```

GIW

*The Generalized Inverse Weibull family***Description**

The Generalized Inverse Weibull family

**Usage**

```
GIW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.  
`nu.link` defines the `nu.link`, with "log" link as the default for the `nu` parameter.

**Details**

The Generalized Inverse Weibull distribution with parameters `mu`, `sigma` and `nu` has density given by

$$f(x) = \nu \sigma \mu^\sigma x^{-(\sigma+1)} \exp\left\{-\nu \left(\frac{\mu}{x}\right)^\sigma\right\},$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a GIW distribution in the `gamlss()` function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

## References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

De Gusmao, F. R., Ortega, E. M., & Cordeiro, G. M. (2011). The generalized inverse Weibull distribution. *Statistical Papers*, 52, 591-619.

## See Also

[dGIW](#)

## Examples

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rGIW(n=200, mu=3, sigma=5, nu=0.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='GIW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 500
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(-1.02 + 3 * x1)
sigma <- exp(1.69 - 2 * x2)
nu <- 0.5
x <- rGIW(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, family=GIW,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))
```

**Description**

The Generalized modified Weibull distribution

**Usage**

```
GMW(mu.link = "log", sigma.link = "log", nu.link = "sqrt", tau.link = "sqrt")
```

**Arguments**

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the mu parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the sigma.
<code>nu.link</code>	defines the <code>nu.link</code> , with "sqrt" link as the default for the nu parameter.
<code>tau.link</code>	defines the <code>tau.link</code> , with "sqrt" link as the default for the tau parameter.

**Details**

The Generalized modified Weibull distribution with parameters mu, sigma, nu and tau has density given by

$$f(x) = \mu\sigma x^{\nu-1}(\nu + \tau x) \exp(\tau x - \mu x^{\nu} e^{\tau x}) [1 - \exp(-\mu x^{\nu} e^{\tau x})]^{\sigma-1},$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a GMW distribution in the `gamlss()` function.

**See Also**

[dGMW](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma, nu and tau
y <- rGMW(n=100, mu=2, sigma=0.5, nu=2, tau=1.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, tau.fo=~ 1, family='GMW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))
```

```

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
(coef(mod, what='nu'))^2
(coef(mod, what='tau'))^2

# Example 2
# Generating random values under some model
## Not run:
n <- 1000
x1 <- runif(n)
x2 <- runif(n)
mu <- exp(2 + -3 * x1)
sigma <- exp(3 - 2 * x2)
nu <- 2
tau <- 1.5
x <- rGMW(n=n, mu, sigma, nu, tau)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, tau.fo=~ 1, family="GMW",
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
coef(mod, what="nu")^2
coef(mod, what="tau")^2

## End(Not run)

```

---

initValuesOW

*Initial values and search region for Odd Weibull distribution*


---

## Description

This function can be used so as to get suggestions about initial values and the search region for parameter estimation in OW distribution.

## Usage

```

initValuesOW(
  formula,
  data = NULL,
  local_reg = loess.options(),
  interpolation = interp.options(),
  ...
)

```

**Arguments**

formula	an object of class <code>formula</code> with the response on the left of an operator $\sim$ . The right side must be 1.
data	an optional data frame containing the response variables. If data is not specified, the variables are taken from the environment from which <code>initValuesOW</code> is called.
local_reg	a list of control parameters for LOESS. See <code>loess.options</code> .
interpolation	a list of control parameters for interpolation function. See <code>interp.options</code> .
...	further arguments passed to <code>TTTE_Analytical</code> .

**Details**

This function performs a non-parametric estimation of the empirical total time on test (TTT) plot. Then, this estimated curve can be used so as to get suggestions about initial values and the search region for parameters based on hazard shape associated to the shape of empirical TTT plot.

**Value**

Returns an object of class `c("initValOW", "HazardShape")` containing:

- `sigma.start` value for *sigma* parameter of OW distribution.
- `nu.start` value for *nu* parameter of OW distribution.
- `sigma.valid` search region for *sigma* parameter of OW distribution.
- `nu.valid` search region for *nu* parameter of OW distribution.
- `TTTplot` Total Time on Test transform computed from the data.
- `hazard_type` shape of the hazard function determined from the TTT plot.

**Author(s)**

Jaime Mosquera Gutiérrez <jmosquerag@unal.edu.co>

**Examples**

```
# Example 1
# Bathtub hazard and its corresponding TTT plot
y1 <- rOW(n = 1000, mu = 0.1, sigma = 7, nu = 0.08)
my_initial_guess1 <- initValuesOW(formula=y1~1)
summary(my_initial_guess1)
plot(my_initial_guess1, par_plot=list(mar=c(3.7,3.7,1,2.5),
                                     mgp=c(2.5,1,0)))

curve(hOW(x, mu = 0.022, sigma = 8, nu = 0.01), from = 0,
      to = 80, ylim = c(0, 0.04), col = "red",
      ylab = "Hazard function", las = 1)

# Example 2
# Bathtub hazard and its corresponding TTT plot with right censored data
```

```

y2 <- rOW(n = 1000, mu = 0.1, sigma = 7, nu = 0.08)
status <- c(rep(1, 980), rep(0, 20))
my_initial_guess2 <- initialValuesOW(formula=Surv(y2, status)~1)
summary(my_initial_guess2)
plot(my_initial_guess2, par_plot=list(mar=c(3.7,3.7,1,2.5),
                                     mgp=c(2.5,1,0)))

curve(hOW(x, mu = 0.022, sigma = 8, nu = 0.01), from = 0,
      to = 80, ylim = c(0, 0.04), col = "red",
      ylab = "Hazard function", las = 1)

```

---

 IW

*The Inverse Weibull family*


---

## Description

The Inverse Weibull distribution

## Usage

```
IW(mu.link = "log", sigma.link = "log")
```

## Arguments

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.

## Details

The Inverse Weibull distribution with parameters `mu`, `sigma` has density given by

$$f(x) = \mu\sigma x^{-\sigma-1} \exp(\mu x^{-\sigma})$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$

## Value

Returns a `gamlss.family` object which can be used to fit a IW distribution in the `gamlss()` function.

## Author(s)

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

## References

- Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.
- Drapella, A. (1993). The complementary Weibull distribution: unknown or just forgotten?. *Quality and reliability engineering international*, 9(4), 383-385.

**See Also**[dIW](#)**Examples**

```

# Example 1
# Generating some random values with
# known mu and sigma
y <- rIW(n=100, mu=5, sigma=2.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, mu.fo=~1, sigma.fo=~1, family='IW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- rpois(n, lambda=2)
x2 <- runif(n)
mu <- exp(2 + -1 * x1)
sigma <- exp(2 - 2 * x2)
x <- rIW(n=n, mu, sigma)

mod <- gamlss(x~x1, mu.fo=~1, sigma.fo=~x2, family=IW,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")

```

**Description**

The Kumaraswamy Inverse Weibull family

**Usage**

```
KumIW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the mu parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the sigma.
<code>nu.link</code>	defines the <code>nu.link</code> , with "log" link as the default for the nu parameter.

**Details**

The Kumaraswamy Inverse Weibull Distribution with parameters `mu`, `sigma` and `nu` has density given by

$$f(x) = \mu\sigma\nu x^{-\mu-1} \exp -\sigma x^{-\mu} (1 - \exp -\sigma x^{-\mu})^{\nu-1},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a KumIW distribution in the `gamlss()` function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

- Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.
- Shahbaz, M. Q., Shahbaz, S., & Butt, N. S. (2012). The Kumaraswamy Inverse Weibull Distribution. *Pakistan journal of statistics and operation research*, 479-489.

**See Also**

[dKumIW](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma, nu and tau
y <- rKumIW(n=100, mu = 1.5, sigma= 1.5, nu = 5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family="KumIW",
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what="mu"))
exp(coef(mod, what="sigma"))
```

```

exp(coef(mod, what="nu"))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(1 - x1)
sigma <- exp(1 - x2)
nu <- 5
x <- rKumIW(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, family=KumIW,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))

```

---

 LIN

*The Lindley family*


---

### Description

The function `LIN()` defines the Lindley distribution with only one parameter for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

### Usage

```
LIN(mu.link = "log")
```

### Arguments

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.

### Details

The Lindley with parameter  $\mu$  has density given by

$$f(x) = \frac{\mu^2}{\mu+1}(1+x)\exp(-\mu x),$$

for  $x > 0$  and  $\mu > 0$ .

### Value

Returns a `gamlss.family` object which can be used to fit a LIN distribution in the `gamlss()` function.

### Author(s)

Freddy Hernandez <fhernanb@unal.edu.co>

## References

Lindley, D. V. (1958). Fiducial distributions and Bayes' theorem. *Journal of the Royal Statistical Society. Series B (Methodological)*, 102-107.

## Examples

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rLIN(n=200, mu=2)

# Fitting the model
require(gamlss)
mod <- gamlss(y ~ 1, family="LIN")

# Extracting the fitted values for mu
# using the inverse link function
exp(coef(mod, what='mu'))

# Example 2
# Generating random values under some model
n <- 100
x1 <- runif(n=n)
x2 <- runif(n=n)
eta <- 1 + 3 * x1 - 2 * x2
mu <- exp(eta)
y <- rLIN(n=n, mu=mu)

mod <- gamlss(y ~ x1 + x2, family=LIN)

coef(mod, what='mu')
```

---

LW

*The Log-Weibull family*


---

## Description

The Log-Weibull distribution

## Usage

```
LW(mu.link = "identity", sigma.link = "log")
```

## Arguments

mu.link	defines the mu.link, with "log" link as the default for the mu parameter.
sigma.link	defines the sigma.link, with "log" link as the default for the sigma.

**Details**

The Log-Weibull Distribution with parameters  $\mu$  and  $\sigma$  has density given by

$$f(y) = (1/\sigma)e^{((y-\mu)/\sigma)} \exp\{-e^{((y-\mu)/\sigma)}\},$$

for  $-\infty < y < \infty$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a LW distribution in the `gamlss()` function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Gumbel, E. J. (1958). *Statistics of extremes*. Columbia university press.

**See Also**

[dLW](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu and sigma
y <- rLW(n=100, mu=0, sigma=1.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, family= 'LW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu and sigma
# using the inverse link function
coef(mod, 'mu')
exp(coef(mod, 'sigma'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- 1.5 - 3 * x1
sigma <- exp(1.4 - 2 * x2)
x <- rLW(n=n, mu, sigma)
```

```
mod <- gamlss(x~x1, sigma.fo=~x2, family=LW,
             control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
```

---

mice	<i>Mice mortality data</i>
------	----------------------------

---

### Description

The ages at death in weeks for male mice exposed to 240r of gamma radiation.

### Usage

```
data(mice)
```

### Format

A vector with 208 data points.

### Examples

```
data(mice)
hist(mice, main="", xlab="Time (weeks)", freq=FALSE)
lines(density(mice), col="blue", lwd=2)
```

---

MOEIW	<i>The Marshall-Olkin Extended Inverse Weibull family</i>
-------	---

---

### Description

The Marshall-Olkin Extended Inverse Weibull family

### Usage

```
MOEIW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

### Arguments

mu.link	defines the mu.link, with "log" link as the default for the mu parameter.
sigma.link	defines the sigma.link, with "log" link as the default for the sigma.
nu.link	defines the nu.link, with "log" link as the default for the nu parameter.

**Details**

The Marshall-Olkin Extended Inverse Weibull distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = \frac{\mu\sigma\nu x^{-(\sigma+1)} \exp\{-\mu x^{-\sigma}\}}{\{\nu - (\nu-1)\exp\{-\mu x^{-\sigma}\}\}^2},$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a MOEIW distribution in the `gamlss()` function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Okasha, H. M., El-Baz, A. H., Tarabia, A. M. K., & Basheer, A. M. (2017). Extended inverse Weibull distribution with reliability application. *Journal of the Egyptian Mathematical Society*, 25(3), 343-349.

**See Also**

[dMOEIW](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
set.seed(123456)
y <- rMOEIW(n=100, mu=0.6, sigma=1.7, nu=0.3)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family="MOEIW",
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what="mu"))
exp(coef(mod, what="sigma"))
exp(coef(mod, what="nu"))

# Example 2
# Generating random values under some model

# A function to simulate a data set with Y ~ MOEIW
gendat <- function(n) {
```

```

x1 <- runif(n)
x2 <- runif(n)
mu   <- exp(-2.02 + 3 * x1) # 0.60 approximately
sigma <- exp(2.23 - 2 * x2) # 3.42 approximately
nu   <- 2
y <- rMOEIW(n=n, mu=mu, sigma=sigma, nu=nu)
data.frame(y=y, x1=x1, x2=x2)
}

set.seed(123)
dat <- gendat(n=100)

mod <- gamlss(y~x1, sigma.fo=~x2, nu.fo=~1,
              family=MOEIW, data=dat,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))

```

MOEW

*The Marshall-Olkin Extended Weibull family***Description**

The Marshall-Olkin Extended Weibull family

**Usage**

```
MOEW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

mu.link defines the mu.link, with "log" link as the default for the mu parameter.  
sigma.link defines the sigma.link, with "log" link as the default for the sigma.  
nu.link defines the nu.link, with "log" link as the default for the nu parameter.

**Details**

The Marshall-Olkin Extended Weibull distribution with parameters mu, sigma and nu has density given by

$$f(x) = \frac{\mu\sigma\nu(\nu x)^{\sigma-1} \exp\{-(\nu x)^\sigma\}}{\{1-(1-\mu)\exp\{-(\nu x)^\sigma\}\}^2},$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a MOEW distribution in the `gamlss()` function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

- Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.
- Ghitany, M. E., Al-Hussaini, E. K., & Al-Jarallah, R. A. (2005). Marshall–Olkin extended Weibull distribution and its application to censored data. *Journal of Applied Statistics*, 32(10), 1025-1034.

**See Also**

[dMOEW](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rMOEW(n=400, mu=0.5, sigma=0.7, nu=1)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='MOEW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 500
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(-1.20 + 3 * x1)
sigma <- exp(0.84 - 2 * x2)
nu <- 1
x <- rMOEW(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, family=MOEW,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))
```

MOK

*The Marshall-Olkin Kappa family***Description**

The Marshall-Olkin Kappa family

**Usage**

```
MOK(mu.link = "log", sigma.link = "log", nu.link = "log", tau.link = "log")
```

**Arguments**

mu.link	defines the mu.link, with "log" link as the default for the mu parameter.
sigma.link	defines the sigma.link, with "log" link as the default for the sigma.
nu.link	defines the nu.link, with "log" link as the default for the nu parameter.
tau.link	defines the tau.link, with "log" link as the default for the tau parameter.

**Details**

The Marshall-Olkin Kappa distribution with parameters mu, sigma, nu and tau has density given by

$$f(x) = \frac{\tau \frac{\mu\nu}{\sigma} \left(\frac{x}{\sigma}\right)^{\nu-1} \left(\mu + \left(\frac{x}{\sigma}\right)^{\mu\nu}\right)^{-\frac{\mu+1}{\mu}}}{\left(\tau + (1-\tau) \left(\frac{\left(\frac{x}{\sigma}\right)^{\mu\nu}}{\mu + \left(\frac{x}{\sigma}\right)^{\mu\nu}}\right)^{\frac{1}{\mu}}\right)^2}$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a MOK distribution in the `gamlss()` function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

Javed, M., Nawaz, T., & Irfan, M. (2019). The Marshall-Olkin kappa distribution: properties and applications. *Journal of King Saud University-Science*, 31(4), 684-691.

**See Also**

[dMOK](#)

**Examples**

```

# Example 1
# Generating some random values with
# known mu, sigma, nu and tau
y <- rMOK(n=100, mu = 1, sigma = 3.5, nu = 3, tau = 2)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, tau.fo=~1, family=MOK,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma, nu and tau
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))
exp(coef(mod, what='tau'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(0.5 + x1)
sigma <- exp(0.8 + x2)
nu <- 1
tau <- 0.5
x <- rMOK(n=n, mu, sigma, nu, tau)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, tau.fo=~1, family=MOK,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))
exp(coef(mod, what="tau"))

```

---

 MW

*The Modified Weibull family*


---

**Description**

```
#' The Modified Weibull distribution
```

**Usage**

```
MW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

<code>mu.link</code>	defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter.
<code>sigma.link</code>	defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> .
<code>nu.link</code>	defines the <code>nu.link</code> , with "log" link as the default for the <code>nu</code> parameter.

**Details**

The Modified Weibull distribution with parameters `mu`, `sigma` and `nu` has density given by

$$f(x) = \mu(\sigma + \nu x)x^{\sigma-1} \exp(\nu x) \exp(-\mu x^\sigma \exp(\nu x)),$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma \geq 0$  and  $\nu \geq 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a MW distribution in the `gamlss()` function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Lai, C. D., Xie, M., & Murthy, D. N. P. (2003). A modified Weibull distribution. *IEEE Transactions on reliability*, 52(1), 33-37.

**See Also**

[dMW](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rMW(n=100, mu = 2, sigma = 1.5, nu = 0.2)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family= 'MW',
             control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))
```

```

# Example 2
# Generating random values under some model
n      <- 200
x1     <- rpois(n, lambda=2)
x2     <- runif(n)
mu     <- exp(3 - 1 * x1)
sigma  <- exp(2 - 2 * x2)
nu     <- 0.2
x      <- rMW(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, mu.fo=~x1, sigma.fo=~x2, nu.fo=~1, family=MW,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
coef(mod, what='nu')

```

---

myOW\_region

*Customized region search for odd Weibull distribution*


---

### Description

This function can be used to modify OW `gamlss.family` object in order to set a customized region search for `gamlss()` function.

### Usage

```
myOW_region(family = OW, valid.values = "auto", initVal)
```

### Arguments

family	The OW family. This arguments allows the user to modify input arguments of the family, like the link functions.
valid.values	a list of character elements specifying the region for sigma and/or nu. See <b>Details</b> and <b>Examples</b> section to learn about its use.
initVal	An <code>initValOW</code> object generated with <code>initValuesOW</code> function.

### Details

This function was created to help users to fit OW distribution easily bounding the parametric space for sigma and nu.

The `valid.values` must be defined as a list of characters containing a call of the `all` function.

### Value

Returns a `gamlss.family` object which can be used to fit an OW distribution in the `gamlss()` function.

**Author(s)**

Jaime Mosquera Gutiérrez <jmosquerag@unal.edu.co>

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rOW(n=200, mu=0.2, sigma=4, nu=0.05)

# Custom search region
myvalues <- list(sigma="all(sigma > 1)",
                 nu="all(nu < 1) & all(nu < 1)")

my_initial_guess <- initialValuesOW(formula=y~1)
summary(my_initial_guess)

# OW family modified with 'myOW_region'
require(gamlss)
myOW <- myOW_region(valid.values=myvalues, initVal=my_initial_guess)
mod1 <- gamlss(y~1, sigma.fo=~1, nu.fo=~1,
              sigma.start=param.startOW('sigma', my_initial_guess),
              nu.start=param.startOW('nu', my_initial_guess),
              control=gamlss.control(n.cyc=300, trace=FALSE),
              family=myOW)

exp(coef(mod1, what='mu'))
exp(coef(mod1, what='sigma'))
exp(coef(mod1, what='nu'))

# Example 2
# Same example using another link function and using 'myOW_region'
# in the argument 'family'
mod2 <- gamlss(y~1, sigma.fo=~1, nu.fo=~1,
              sigma.start=2, nu.start=0.1,
              control=gamlss.control(n.cyc=300, trace=FALSE),
              family=myOW_region(family=OW(sigma.link='identity'),
                                valid.values=myvalues,
                                initVal=my_initial_guess))

exp(coef(mod2, what='mu'))
coef(mod2, what='sigma')
exp(coef(mod2, what='nu'))
```

---

 NEE

*New Exponentiated Exponential family*


---

**Description**

The function `NEE()` defines the New Exponentiated Exponential distribution, a two parameter distribution, for a `gamlss.family` object to be used in `GAMLSS` fitting using the function `gamlss()`.

**Usage**

```
NEE(mu.link = "log", sigma.link = "log")
```

**Arguments**

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "logit" link as the default for the `sigma`.

**Details**

The New Exponentiated Exponential distribution with parameters `mu` and `sigma` has density given by

$$f(x|\mu, \sigma) = \log(2^\sigma) \mu \exp(-\mu x) (1 - \exp(-\mu x))^{\sigma-1} 2^{(1-\exp(-\mu x))^\sigma},$$

for  $x > 0$ ,  $\mu > 0$  and  $\sigma > 0$ .

Note: In this implementation we changed the original parameters  $\theta$  for  $\mu$  and  $\alpha$  for  $\sigma$ , we did it to implement this distribution within `gamlss` framework.

**Value**

Returns a `gamlss.family` object which can be used to fit a NEE distribution in the `gamlss()` function.

**References**

Hassan, Anwar, I. H. Dar, and M. A. Lone. "A New Class of Probability Distributions With An Application to Engineering Data." *Pakistan Journal of Statistics and Operation Research* 20.2 (2024): 217-231.

**See Also**

[dNEE](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu and sigma
y <- rNEE(n=500, mu=2.5, sigma=3.5)

# Fitting the model
require(gamlss)

mod1 <- gamlss(y~1, sigma.fo=~1, family=NEE,
               control=gamlss.control(n.cyc=5000, trace=TRUE))

# Extracting the fitted values for mu, sigma
# using the inverse link function
exp(coef(mod1, what="mu"))
exp(coef(mod1, what="sigma"))
```

```

# Example 2
# Generating random values under some model
gendat <- function(n) {
  x1 <- runif(n)
  x2 <- runif(n)
  mu <- exp(-0.2 + 1.5 * x1)
  sigma <- exp(1 - 0.7 * x2)
  y <- rNEE(n=n, mu, sigma)
  data.frame(y=y, x1=x1, x2=x2)
}

set.seed(123)
datos <- gendat(n=500)

mod2 <- gamlss(y~x1, sigma.fo=~x2, family=NEE, data=datos,
               control=gamlss.control(n.cyc=5000, trace=TRUE))

summary(mod2)

# Example 3 -----
# Obtained from Hassan (2024) page 226
# The data set consists of 63 observations of the gauge lengths of 10mm.

y <- c(1.901, 2.132, 2.203, 2.228, 2.257, 2.350, 2.361, 2.396, 2.397,
       2.445, 2.454, 2.474, 2.518, 2.522, 2.525, 2.532, 2.575, 2.614,
       2.616, 2.618, 2.624, 2.659, 2.675, 2.738, 2.740, 2.856, 2.917,
       2.928, 2.937, 2.937, 2.977, 2.996, 3.030, 3.125, 3.139, 3.145,
       3.220, 3.223, 3.235, 3.243, 3.264, 3.272, 3.294, 3.332, 3.346,
       3.377, 3.408, 3.435, 3.493, 3.501, 3.537, 3.554, 3.562, 3.628,
       3.852, 3.871, 3.886, 3.971, 4.024, 4.027, 4.225, 4.395, 5.020)

mod3 <- gamlss(y~1, family=NEE)

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod3, what="mu"))
exp(coef(mod3, what="sigma"))

# Hist and estimated pdf
hist(y, freq=FALSE, ylim=c(0, 0.7))
curve(dNEE(x, mu=2.076862, sigma=255.2289),
      add=TRUE, col="tomato", lwd=2)

# Empirical cdf and estimated ecdf
plot(ecdf(y))
curve(pNEE(x, mu=2.076862, sigma=255.2289),
      add=TRUE, col="tomato", lwd=2)

# QQplot
qqplot(y, rNEE(n=length(y), mu=2.076862, sigma=255.2289), col="tomato")
qqline(y, distribution=function(p) qNEE(p, mu=2.076862, sigma=255.2289))

# Example 4 -----
# Obtained from Hassan (2024) page 226

```

```

# The dataset was reported by Bader and Priest (1982) on failure
# stresses (in GPa) of 65 single carbon fibers of lengths 50 mm

y <- c(0.564, 0.729, 0.802, 0.95, 1.053, 1.111, 1.115, 1.194, 1.208,
      1.216, 1.247, 1.256, 1.271, 1.277, 1.305, 1.313, 1.348,
      1.39, 1.429, 1.474, 1.49, 1.503, 1.52, 1.522, 1.524, 1.551,
      1.551, 1.609, 1.632, 1.632, 1.676, 1.684, 1.685, 1.728, 1.74,
      1.761, 1.764, 1.785, 1.804, 1.816, 1.824, 1.836, 1.879, 1.883,
      1.892, 1.898, 1.934, 1.947, 1.976, 2.02, 2.023, 2.05, 2.059,
      2.068, 2.071, 2.098, 2.13, 2.204, 2.317, 2.334, 2.34, 2.346,
      2.378, 2.483, 2.269)

mod4 <- gamlss(y~1, family=NEE)

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod4, what="mu"))
exp(coef(mod4, what="sigma"))

hist(y, freq=FALSE)
curve(dNEE(x, mu=2.400515, sigma=25.15236),
      add=TRUE, col="tomato", lwd=2)

# Empirical cdf and estimated ecdf
plot(ecdf(y))
curve(pNEE(x, mu=2.400515, sigma=25.15236),
      add=TRUE, col="tomato", lwd=2)

# QQplot
qqplot(y, rNEE(n=length(y), mu=2.400515, sigma=25.15236), col="tomato")
qqline(y, distribution=function(p) qNEE(p, mu=2.400515, sigma=25.15236))

# Example 5 -----
# 69 Observations of the gauge lengths of 20m.
y <- c(1.312,1.314,1.479,1.552,1.700,1.803,1.861,1.865,1.944,1.958,1.966,1.997,
      2.006,2.021,2.027,2.055, 2.063,2.098,2.140,2.179,2.224,2.240,2.253,2.270,
      2.272,2.274,2.301,2.301,2.359,2.382,2.382,2.426, 2.434,2.435,2.478,2.490,
      2.511,2.514,2.535,2.554,2.566,2.570,2.586,2.629,2.633,2.642,2.648,2.684,
      2.697,2.726,2.770,2.773,2.800,2.809,2.818,2.821,2.848,2.880,2.954,3.012,
      3.067,3.084,3.090,3.096, 3.128,3.233,3.433,3.585,3.585)

mod5 <- gamlss(y~1, sigma.fo=~1, family = NEE)

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod5, what="mu"))
exp(coef(mod5, what="sigma"))

hist(y, freq=FALSE)
curve(dNEE(x, mu=2.197771, sigma=100.8888), add=TRUE,
      col="tomato", lwd=2)
# Empirical cdf and estimated ecdf
plot(ecdf(y))
curve(pNEE(x, mu=2.197771, sigma=100.8888), add=TRUE,

```

```

      col="tomato", lwd=2)
# QQplot
qqplot(y, rNEE(n=length(y), mu=2.197771, sigma=100.8888), col="tomato")
qqline(y, distribution=function(p) qNEE(p, mu=2.197771, sigma=100.8888))

```

OW

*The Odd Weibull family***Description**

The function `OW()` defines the Odd Weibull distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

**Usage**

```
OW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.  
`nu.link` defines the `nu.link`, with "log" link as the default for the `nu`.

**Details**

The odd Weibull with parameters `mu`, `sigma` and `nu` has density given by

$$f(t) = \left(\frac{\sigma\nu}{t}\right) (\mu t)^\sigma e^{(\mu t)^\sigma} (e^{(\mu t)^\sigma} - 1)^{\nu-1} \left[1 + (e^{(\mu t)^\sigma} - 1)^\nu\right]^{-2}$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a OW distribution in the `gamlss()` function.

**Author(s)**

Jaime Mosquera Gutiérrez <jmosquerag@unal.edu.co>

**References**

Cooray, K. (2006). Generalization of the Weibull distribution: the odd Weibull family. *Statistical Modelling*, 6(3), 265-277.

**Examples**

```

# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rOW(n=200, mu=0.1, sigma=7, nu = 1.1)

# Fitting the model
require(gamlss)
mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family="OW",
              control=gamlss.control(n.cyc=500, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what="mu"))
exp(coef(mod, what="sigma"))
exp(coef(mod, what="nu"))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n)
x2 <- runif(n)
x3 <- runif(n)
mu <- exp(1.2 + 2 * x1)
sigma <- 2.12 + 3 * x2
nu <- exp(0.2 - x3)
y <- rOW(n=n, mu, sigma, nu)

mod <- gamlss(y~x1, sigma.fo=~x2, nu.fo=~x3,
              family=OW(sigma.link="identity"),
              control=gamlss.control(n.cyc=300, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
coef(mod, what="nu")

```

---

param.startOW

*Initial values extraction for Odd Weibull distribution*


---

**Description**

This function can be used to extract initial values found with empirical time on test transform (TTT) through [initValuesOW](#) function. This is used for parameter estimation in OW distribution.

**Usage**

```
param.startOW(param, initValOW)
```

**Arguments**

param	a character used to specify the parameter required. It can take the values "sigma" or "nu".
initValOW	an initValOW object generated with <code>initValuesOW</code> function.

**Details**

This function just gets initial values computed with `initValuesOW` for OW family. It must be called in `sigma.start` and `nu.start` arguments from `gamlss` function. This function is useful only if user want to set start values automatically with TTT plot. See example for an illustration.

**Value**

A length-one vector numeric value corresponding to the initial value of the parameter specified in `param` extracted from a `initValuesOW` object specified in the `initValOW` input argument.

**Author(s)**

Jaime Mosquera Gutiérrez <jmosquerag@unal.edu.co>

**Examples**

```
# Random data generation (OW distributed)
y <- rOW(n=500, mu=0.05, sigma=0.6, nu=2)

# Initial values with TTT plot
iv <- initValuesOW(formula = y ~ 1)
summary(iv)

# This data is from unimodal hazard
# See TTT estimate from sample
plot(iv, legend_options=list(pos=1.03))

# See the true hazard
curve(hOW(x, mu=0.05, sigma=0.6, nu=2), to=100, lwd=3, ylab="h(x)")

# Finally, we fit the model
require(gamlss)
con.out <- gamlss.control(n.cyc = 300, trace = FALSE)
con.in <- glim.control(cyc = 300)

(sigma.start <- param.startOW("sigma", iv))
(nu.start <- param.startOW("nu", iv))

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, control=con.out, i.control=con.in,
             family=myOW_region(OW(sigma.link="identity", nu.link="identity"),
                                valid.values="auto", iv),
             sigma.start=sigma.start, nu.start=nu.start)

# Estimates are close to actual values
(mu <- exp(coef(mod, what = "mu")))
```

```
(sigma <- coef(mod, what = "sigma"))
(nu <- coef(mod, what = "nu"))
```

---

 PL

---

*The Power Lindley family*


---

### Description

Power Lindley distribution

### Usage

```
PL(mu.link = "log", sigma.link = "log")
```

### Arguments

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.

### Details

The Power Lindley Distribution with parameters `mu` and `sigma` has density given by

$$f(x) = \frac{\mu\sigma^2}{\sigma+1}(1+x^\mu)x^{\mu-1}\exp(-\sigma x^\mu),$$

for  $x > 0$ .

### Value

Returns a `gamlss.family` object which can be used to fit a PL distribution in the `gamlss()` function.

### Author(s)

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

### References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Ghitany, M. E., Al-Mutairi, D. K., Balakrishnan, N., & Al-Enezi, L. J. (2013). Power Lindley distribution and associated inference. *Computational Statistics & Data Analysis*, 64, 20-33.

### See Also

[dPL](#)

**Examples**

```

# Example 1
# Generating some random values with
# known mu and sigma
y <- rPL(n=100, mu=1.5, sigma=0.2)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, family= 'PL',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod, 'mu'))
exp(coef(mod, 'sigma'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(1.2 - 2 * x1)
sigma <- exp(0.8 - 3 * x2)
x <- rPL(n=n, mu, sigma)

mod <- gamlss(x~x1, sigma.fo=~x2, family=PL,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")

```

---

QXGP

*The Quasi XGamma Poisson family*


---

**Description**

The Quasi XGamma Poisson family

**Usage**

```
QXGP(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

mu.link	defines the mu.link, with "log" link as the default for the mu parameter.
sigma.link	defines the sigma.link, with "log" link as the default for the sigma.
nu.link	defines the nu.link, with "log" link as the default for the nu parameter.

**Details**

The Quasi XGamma Poisson distribution with parameters  $\mu$ ,  $\sigma$  and  $\nu$  has density given by

$$f(x) = K(\mu, \sigma, \nu) \left( \frac{\sigma^2 x^2}{2} + \mu \right) \exp\left( \frac{\nu \exp(-\sigma x) (1 + \mu + \sigma x + \frac{\sigma^2 x^2}{2})}{1 + \mu} - \sigma x \right),$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu > 1$ .

where

$$K(\mu, \sigma, \nu) = \frac{\nu \sigma}{(\exp(\nu) - 1)(1 + \mu)}$$

**Value**

Returns a `gamlss.family` object which can be used to fit a QXGP distribution in the `gamlss()` function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Sen, S., Korkmaz, M. Ç., & Yousof, H. M. (2018). The quasi XGamma-Poisson distribution: properties and application. *Istatistik Journal of The Turkish Statistical Association*, 11(3), 65-76.

**See Also**

[dQXGP](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rQXGP(n=200, mu=4, sigma=2, nu=3)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='QXGP',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 2000
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
```

```

mu <- exp(-2.19 + 3 * x1)
sigma <- exp(1 - 2 * x2)
nu <- 1
x <- rQXGP(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, family=QXGP,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))

```

RW

*The Reflected Weibull family***Description**

Reflected Weibull distribution

**Usage**

```
RW(mu.link = "log", sigma.link = "log")
```

**Arguments**

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.

**Details**

The Reflected Weibull Distribution with parameters `mu` and `sigma` has density given by

$$f(y) = \mu\sigma(-y)^{\sigma-1}e^{-\mu(-y)^\sigma},$$

for  $y < 0$

**Value**

Returns a `gamlss.family` object which can be used to fit a RW distribution in the `gamlss()` function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.  
 Cohen, A. C. (1973). The reflected Weibull distribution. *Technometrics*, 15(4), 867-873.

**See Also**[dRW](#)**Examples**

```

# Example 1
# Generating some random values with
# known mu and sigma
y <- rRW(n=100, mu=1, sigma=1)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, family= 'RW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu and sigma
# using the inverse link function
exp(coef(mod, 'mu'))
exp(coef(mod, 'sigma'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(1.5 - 1.5 * x1)
sigma <- exp(2 - 2 * x2)
x <- rRW(n=n, mu, sigma)

mod <- gamlss(x~x1, sigma.fo=~x2, family=RW,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")

```

---

summary.initValOW

*Summary of initValOW objects*


---

**Description**

This summary method displays initial values and search regions for [OW](#) family.

**Usage**

```

## S3 method for class 'initValOW'
summary(object, ...)

```

**Arguments**

object            an object of class `initVal`, generated with `initValuesOW`.  
 ...                extra arguments

**Value**

No return value, it just prints out in the console the initial values and the search regions for *sigma* and *nu* from OW distribution (see `dOW`).

**Author(s)**

Jaime Mosquera Gutiérrez <jmosquerag@unal.edu.co>

---

 SZMW

*The Sarhan and Zaindin's Modified Weibull family*

---

**Description**

The Sarhan and Zaindin's Modified Weibull distribution

**Usage**

```
SZMW(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

`mu.link`            defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link`        defines the `sigma.link`, with "log" link as the default for the `sigma`.  
`nu.link`            defines the `nu.link`, with "log" link as the default for the `nu` parameter.

**Details**

The Sarhan and Zaindin's Modified Weibull distribution with parameters `mu`, `sigma` and `nu` has density given by

$$f(x) = (\mu + \sigma \nu x^{\nu-1}) \exp(-\mu x - \sigma x^\nu),$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a SZMW distribution in the `gamlss()` function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

## References

Almalki, S. J., & Nadarajah, S. (2014). Modifications of the Weibull distribution: A review. *Reliability Engineering & System Safety*, 124, 32-55.

Sarhan, A. M., & Zaindin, M. (2009). Modified Weibull distribution. *APPS. Applied Sciences*, 11, 123-136.

## See Also

[dSZMW](#)

## Examples

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rSZMW(n=100, mu = 1, sigma = 1, nu = 1.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='SZMW',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 200
x1 <- runif(n)
x2 <- runif(n)
mu <- exp(-1.6 * x1)
sigma <- exp(0.9 - 1 * x2)
nu <- 1.5
x <- rSZMW(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, mu.fo=~x1, sigma.fo=~x2, nu.fo=~1, family=SZMW,
              control=gamlss.control(n.cyc=50000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
coef(mod, what='nu')
```

---

WALD

*The Wald family*

---

### Description

The function `WALD()` defines the `wALD` distribution, two-parameter continuous distribution for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

### Usage

```
WALD(mu.link = "log", sigma.link = "log")
```

### Arguments

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma` parameter.

### Details

The Wald distribution with parameters  $\mu$  and  $\sigma$  has density given by

$$f(x|\mu, \sigma) = \frac{\sigma}{\sqrt{2\pi x^3}} \exp\left[-\frac{(\sigma - \mu x)^2}{2x}\right], x > 0$$

### Value

Returns a `gamlss.family` object which can be used to fit a WALD distribution in the `gamlss()` function.

### Author(s)

Sofia Cuartas García, <[scuartasg@unal.edu.co](mailto:scuartasg@unal.edu.co)>

### References

Heathcote, A. (2004). Fitting Wald and ex-Wald distributions to response time data: An example using functions for the S-PLUS package. *Behavior Research Methods, Instruments, & Computers*, 36, 678-694.

### See Also

[dWALD](#).

**Examples**

```

# Example 1
# Generating random values with
# known mu and sigma
require(gamlss)
mu <- 1.5
sigma <- 4.0

y <- rWALD(10000, mu, sigma)

mod1 <- gamlss(y~1, sigma.fo=~1, family="WALD",
               control=gamlss.control(n.cyc=5000, trace=TRUE))

exp(coef(mod1, what="mu"))
exp(coef(mod1, what="sigma"))

# Example 2
# Generating random values under some model

# A function to simulate a data set with  $Y \sim \text{WALD}$ 
gendat <- function(n) {
  x1 <- runif(n)
  x2 <- runif(n)
  mu <- exp(0.75 - 0.69 * x1) # Approx 1.5
  sigma <- exp(0.5 - 0.64 * x2) # Approx 1.20
  y <- rWALD(n, mu, sigma)
  data.frame(y=y, x1=x1, x2=x2)
}

dat <- gendat(n=200)

mod2 <- gamlss(y~x1, sigma.fo=~x2, family=WALD, data=dat,
               control=gamlss.control(n.cyc=5000, trace=TRUE))

summary(mod2)

```

**Description**

The Weibull Geometric distribution

**Usage**

```
WG(mu.link = "log", sigma.link = "log", nu.link = "logit")
```

**Arguments**

mu.link defines the mu.link, with "log" link as the default for the mu parameter.  
 sigma.link defines the sigma.link, with "log" link as the default for the sigma.  
 nu.link defines the nu.link, with "log" link as the default for the nu parameter.

**Details**

The weibull geometric distribution with parameters mu, sigma and nu has density given by

$$f(x) = (\sigma\mu^\sigma(1-\nu)x^{\sigma-1}\exp(-(\mu x)^\sigma))(1-\nu\exp(-(\mu x)^\sigma))^{-2},$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $0 < \nu < 1$ .

**Value**

Returns a gamlss.family object which can be used to fit a WG distribution in the gamlss() function.

**Author(s)**

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

**References**

Barreto-Souza, W., de Morais, A. L., & Cordeiro, G. M. (2011). The Weibull-geometric distribution. Journal of Statistical Computation and Simulation, 81(5), 645-657.

**See Also**

[dWG](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rWG(n=100, mu = 0.9, sigma = 2, nu = 0.5)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='WG',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 200
```

```

x1 <- runif(n)
x2 <- runif(n)
mu <- exp(- 0.2 * x1)
sigma <- exp(1.2 - 1 * x2)
nu <- 0.5
x <- rWG(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, mu.fo=~x1, sigma.fo=~x2, nu.fo=~1, family=WG,
             control=gamlss.control(n.cyc=50000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
coef(mod, what='nu')

```

---

WGEE

*The Weighted Generalized Exponential-Exponential family*


---

### Description

The Weighted Generalized Exponential-Exponential family

### Usage

```
WGEE(mu.link = "log", sigma.link = "log", nu.link = "log")
```

### Arguments

mu.link defines the mu.link, with "log" link as the default for the mu parameter.  
sigma.link defines the sigma.link, with "log" link as the default for the sigma.  
nu.link defines the nu.link, with "log" link as the default for the nu parameter.

### Details

The Weighted Generalized Exponential-Exponential distribution with parameters mu, sigma and nu has density given by

$$f(x) = \sigma\nu \exp(-\nu x)(1 - \exp(-\nu x))^{\sigma-1}(1 - \exp(-\mu\nu x))/1 - \sigma B(\mu + 1, \sigma),$$

for  $x > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

### Value

Returns a gamlss.family object which can be used to fit a WGEE distribution in the gamlss() function.

### Author(s)

Johan David Marin Benjumea, <johand.marin@udea.edu.co>

## References

Mahdavi, A. (2015). Two weighted distributions generated by exponential distribution. *Journal of Mathematical Extension*, 9, 1-12.

## See Also

[dWGEE](#)

## Examples

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rWGEE(n=1000, mu = 5, sigma = 0.5, nu = 1)

# Fitting the model
require(gamlss)

mod <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family='WGEE',
              control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod, what='mu'))
exp(coef(mod, what='sigma'))
exp(coef(mod, what='nu'))

# Example 2
# Generating random values under some model
n <- 500
x1 <- runif(n, min=0.4, max=0.6)
x2 <- runif(n, min=0.4, max=0.6)
mu <- exp(2 - x1)
sigma <- exp(1 - 3*x2)
nu <- 1
x <- rWGEE(n=n, mu, sigma, nu)

mod <- gamlss(x~x1, sigma.fo=~x2, nu.fo=~1, family=WGEE,
              control=gamlss.control(n.cyc=50000, trace=FALSE))

coef(mod, what="mu")
coef(mod, what="sigma")
exp(coef(mod, what="nu"))
```

## Description

The Weibull Poisson family

**Usage**

```
WP(mu.link = "log", sigma.link = "log", nu.link = "log")
```

**Arguments**

`mu.link` defines the `mu.link`, with "log" link as the default for the `mu` parameter.  
`sigma.link` defines the `sigma.link`, with "log" link as the default for the `sigma`.  
`nu.link` defines the `nu.link`, with "log" link as the default for the `nu` parameter.

**Details**

The Weibull Poisson distribution with parameters `mu`, `sigma` and `nu` has density given by

$$f(x) = \frac{\mu\sigma\nu e^{-\nu}}{1-e^{-\nu}} x^{\mu-1} \exp(-\sigma x^{\mu} + \nu \exp(-\sigma x^{\mu})),$$

for  $x > 0$ .

**Value**

Returns a `gamlss.family` object which can be used to fit a WP distribution in the `gamlss()` function.

**Author(s)**

Amylkar Urrea Montoya, <amylkar.urrea@udea.edu.co>

**References**

Lu, Wanbo, and Daimin Shi. "A new compounding life distribution: the Weibull–Poisson distribution." *Journal of applied statistics* 39.1 (2012): 21-38.

**See Also**

[dWP](#)

**Examples**

```
# Example 1
# Generating some random values with
# known mu, sigma and nu
y <- rWP(n=3000, mu=1.5, sigma=0.5, nu=0.5)

# Fitting the model
require(gamlss)

mod1 <- gamlss(y~1, sigma.fo=~1, nu.fo=~1, family=WP,
               control=gamlss.control(n.cyc=5000, trace=FALSE))

# Extracting the fitted values for mu, sigma and nu
# using the inverse link function
exp(coef(mod1, what="mu"))
exp(coef(mod1, what="sigma"))
```

```
exp(coef(mod1, what="nu"))

# Example 2
# Generating random values for a regression model

# A function to simulate a data set with  $Y \sim WP$ 
gendat <- function(n) {
  x1 <- runif(n)
  x2 <- runif(n)
  mu <- exp(-1.3 + 3.1 * x1)
  sigma <- exp(0.9 - 3.2 * x2)
  nu <- 0.5
  y <- rWP(n=n, mu, sigma, nu)
  data.frame(y=y, x1=x1, x2)
}

set.seed(1234)
dat <- gendat(n=100)

# Fitting the model
mod2 <- NULL
mod2 <- gamlss(y~x1, sigma.fo=~x2, nu.fo=~1,
              family=WP, data=dat,
              control=gamlss.control(n.cyc=5000, trace=FALSE))

coef(mod2, what="mu")
coef(mod2, what="sigma")
exp(coef(mod2, what="nu"))
```

# Index

- \* **datasets**
  - equipment, [107](#)
  - mice, [131](#)
- \* **initValOW**
  - initValuesOW, [123](#)
- AddW, [4, 18](#)
- all, [138](#)
- BGE, [5, 20](#)
- BS, [7, 22](#)
- BS2, [10, 26](#)
- BS3, [12, 29](#)
- CJ2, [14, 31](#)
- CS2e, [16, 34](#)
- dAddW, [4, 17](#)
- dBGE, [6, 19](#)
- dBS, [8, 21](#)
- dBS2, [10, 25](#)
- dBS3, [13, 28](#)
- dCJ2, [15, 30](#)
- dCS2e, [16, 33](#)
- dEEG, [35, 101](#)
- dEGG, [37, 103](#)
- dEMWEx, [39, 104](#)
- dEOFNH, [41, 106](#)
- dEW, [43, 108](#)
- dEXL, [44](#)
- dExW, [48, 112](#)
- dExWALD, [50, 114](#)
- dFWE, [52](#)
- dGammaW, [54, 117](#)
- dGGD, [56, 119](#)
- dGIW, [58, 121](#)
- dGMW, [59, 122](#)
- dGWF, [61](#)
- dIW, [63, 126](#)
- dKumIW, [65, 127](#)
- dLIN, [67](#)
- dLW, [69, 130](#)
- dMOEIW, [71, 132](#)
- dMOEW, [73, 134](#)
- dMOK, [74, 135](#)
- dMW, [76, 137](#)
- dNEE, [78, 140](#)
- dOW, [81, 151](#)
- dPL, [83, 146](#)
- dQXGP, [84, 148](#)
- dRNMW, [86](#)
- dRW, [89, 150](#)
- dSZMW, [91, 152](#)
- dWALD, [92, 153](#)
- dWG, [95, 155](#)
- dWGEE, [97, 157](#)
- dWP, [98, 158](#)
- EEG, [36, 100](#)
- EGG, [38, 102](#)
- EMWEx, [40, 104](#)
- EOFNH, [42, 105](#)
- equipment, [107](#)
- EW, [44, 108](#)
- EXL, [45, 109, 110](#)
- ExW, [49, 112](#)
- ExWALD, [51, 113](#)
- formula, [124](#)
- FWE, [53, 115](#)
- gamlss, [145](#)
- GammaW, [55, 117](#)
- GGD, [57, 118](#)
- GIW, [59, 120](#)
- GMW, [60, 122](#)
- hAddW (dAddW), [17](#)
- hBGE (dBGE), [19](#)
- hBS (dBS), [21](#)

- hBS2 (dBS2), 25
- hBS3 (dBS3), 28
- hCJ2 (dCJ2), 30
- hCS2e (dCS2e), 33
- hEEG (dEEG), 35
- hEGG (dEGG), 37
- hEMWEx (dEMWEx), 39
- hEOFNH (dEOFNH), 41
- hEW (dEW), 43
- hEXL (dEXL), 44
- hExW (dExW), 48
- hFWE (dFWE), 52
- hGammaW (dGammaW), 54
- hGGD (dGGD), 56
- hGIW (dGIW), 58
- hGMW (dGMW), 59
- hGWF (dGWF), 61
- hIW (dIW), 63
- hKumIW (dKumIW), 65
- hLIN (dLIN), 67
- hLW (dLW), 69
- hMOEIW (dMOEIW), 71
- hMOEW (dMOEW), 73
- hMOK (dMOK), 74
- hMW (dMW), 76
- hNEE (dNEE), 78
- hOW (dOW), 81
- hPL (dPL), 83
- hQXGP (dQXGP), 84
- hRNMW (dRNMW), 86
- hRW (dRW), 89
- hSZMW (dSZMW), 91
- hWG (dWG), 95
- hWGEE (dWGEE), 97
- hWP (dWP), 98
  
- initValuesOW, 123, 138, 144, 145, 151
- interp.options, 124
- IW, 64, 125
  
- KumIW, 66, 126
  
- LIN, 68, 128
- loess.options, 124
- LW, 70, 129
  
- mice, 131
- MOEIW, 72, 131
- MOEW, 74, 133
  
- MOK, 76, 135
- MW, 78, 136
- myOW\_region, 138
  
- NEE, 79, 139
  
- OW, 82, 138, 143, 150
  
- pAddW (dAddW), 17
- param.startOW, 144
- pBGE (dBGE), 19
- pBS (dBS), 21
- pBS2 (dBS2), 25
- pBS3 (dBS3), 28
- pCJ2 (dCJ2), 30
- pCS2e (dCS2e), 33
- pEEG (dEEG), 35
- pEGG (dEGG), 37
- pEMWEx (dEMWEx), 39
- pEOFNH (dEOFNH), 41
- pEW (dEW), 43
- pEXL (dEXL), 44
- pExW (dExW), 48
- pExWALD (dExWALD), 50
- pFWE (dFWE), 52
- pGammaW (dGammaW), 54
- pGGD (dGGD), 56
- pGIW (dGIW), 58
- pGMW (dGMW), 59
- pGWF (dGWF), 61
- pIW (dIW), 63
- pKumIW (dKumIW), 65
- PL, 84, 146
- pLIN (dLIN), 67
- pLW (dLW), 69
- pMOEIW (dMOEIW), 71
- pMOEW (dMOEW), 73
- pMOK (dMOK), 74
- pMW (dMW), 76
- pNEE (dNEE), 78
- pOW (dOW), 81
- pPL (dPL), 83
- pQXGP (dQXGP), 84
- pRNMW (dRNMW), 86
- pRW (dRW), 89
- pSZMW (dSZMW), 91
- pWALD (dWALD), 92
- pWG (dWG), 95
- pWGEE (dWGEE), 97

- pWP (dWP), 98  
 qAddW (dAddW), 17  
 qBGE (dBGE), 19  
 qBS (dBS), 21  
 qBS2 (dBS2), 25  
 qBS3 (dBS3), 28  
 qCJ2 (dCJ2), 30  
 qCS2e (dCS2e), 33  
 qEEG (dEEG), 35  
 qEGG (dEGG), 37  
 qEMWEx (dEMWEx), 39  
 qEOFNH (dEOFNH), 41  
 qEW (dEW), 43  
 qEXL (dEXL), 44  
 qExW (dExW), 48  
 qExWALD (dExWALD), 50  
 qFWE (dFWE), 52  
 qGammaW (dGammaW), 54  
 qGGD (dGGD), 56  
 qGIW (dGIW), 58  
 qGMW (dGMW), 59  
 qGWF (dGWF), 61  
 qIW (dIW), 63  
 qKumIW (dKumIW), 65  
 qLIN (dLIN), 67  
 qLW (dLW), 69  
 qMOEIW (dMOEIW), 71  
 qMOEW (dMOEW), 73  
 qMOK (dMOK), 74  
 qMW (dMW), 76  
 qNEE (dNEE), 78  
 qOW (dOW), 81  
 qPL (dPL), 83  
 qQXGP (dQXGP), 84  
 qRNMW (dRNMW), 86  
 qRW (dRW), 89  
 qSZMW (dSZMW), 91  
 qWALD (dWALD), 92  
 qWG (dWG), 95  
 qWGEE (dWGEE), 97  
 qWP (dWP), 98  
 QXGP, 86, 147  
 rAddW (dAddW), 17  
 rBGE (dBGE), 19  
 rBS (dBS), 21  
 rBS2 (dBS2), 25  
 rBS3 (dBS3), 28  
 rCJ2 (dCJ2), 30  
 rCS2e (dCS2e), 33  
 rEEG (dEEG), 35  
 rEGG (dEGG), 37  
 rEMWEx (dEMWEx), 39  
 rEOFNH (dEOFNH), 41  
 rEW (dEW), 43  
 rEXL (dEXL), 44  
 rExW (dExW), 48  
 rExWALD (dExWALD), 50  
 rFWE (dFWE), 52  
 rGammaW (dGammaW), 54  
 rGGD (dGGD), 56  
 rGIW (dGIW), 58  
 rGMW (dGMW), 59  
 rGWF (dGWF), 61  
 rIW (dIW), 63  
 rKumIW (dKumIW), 65  
 rLIN (dLIN), 67  
 rLW (dLW), 69  
 rMOEIW (dMOEIW), 71  
 rMOEW (dMOEW), 73  
 rMOK (dMOK), 74  
 rMW (dMW), 76  
 rNEE (dNEE), 78  
 rOW (dOW), 81  
 rPL (dPL), 83  
 rQXGP (dQXGP), 84  
 rRNMW (dRNMW), 86  
 rRW (dRW), 89  
 rSZMW (dSZMW), 91  
 RW, 90, 149  
 rWALD (dWALD), 92  
 rWG (dWG), 95  
 rWGEE (dWGEE), 97  
 rWP (dWP), 98  
 summary.initValOW, 150  
 SZMW, 92, 151  
 TTTE\_Analytical, 124  
 WALD, 93, 153  
 WG, 96, 154  
 WGEE, 98, 156  
 WP, 99, 157