

# Package ‘gamlss.dist’

October 6, 2018

**Type** Package

**Title** Distributions for Generalized Additive Models for Location Scale and Shape

**Version** 5.1-0

**Date** 2018-10-06

**Description** A set of distributions which can be used for modelling the response variables in Generalized Additive Models for Location Scale and Shape, Rigby and Stasinopoulos (2005), <doi:10.1111/j.1467-9876.2005.00510.x>. The distributions can be continuous, discrete or mixed distributions. Extra distributions can be created, by transforming, any continuous distribution defined on the real line, to a distribution defined on ranges 0 to infinity or 0 to 1, by using a "log" or a "logit" transformation respectively.

**License** GPL-2 | GPL-3

**URL** <http://www.gamlss.org/>

**Depends** R (>= 2.15.0), MASS, graphics, stats, methods

**Repository** CRAN

**Author** Mikis Stasinopoulos [aut, cre, cph],  
Robert Rigby [aut],  
Calliope Akantziliotou [ctb],  
Vlasios Voudouris [ctb],  
Gillian Heller [ctb],  
Fernanda De Bastiani [ctb],  
Raydonal Ospina [ctb],  
Nicoletta Motpan [ctb],  
Fiona McElduff [ctb],  
Majid Djennad [ctb],  
Marco Enea [ctb],  
Alexios Ghalanos [ctb],  
Christos Argyropoulos [ctb],  
Almond Stocker [ctb],  
Jens Lichter [ctb],  
Stanislaus Stadlmann [ctb]

**Maintainer** Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>

**NeedsCompilation** yes

Date/Publication 2018-10-06 11:30:02 UTC

## R topics documented:

gamlss.dist-package . . . . .	3
BB . . . . .	8
BCCG . . . . .	10
BCPE . . . . .	13
BCT . . . . .	15
BE . . . . .	18
BEINF . . . . .	20
BEOI . . . . .	24
BEZI . . . . .	26
BI . . . . .	29
BNB . . . . .	31
checklink . . . . .	33
DBI . . . . .	34
DBURR12 . . . . .	36
DEL . . . . .	38
DPO . . . . .	40
EGB2 . . . . .	42
exGAUS . . . . .	44
EXP . . . . .	46
flexDist . . . . .	48
GA . . . . .	49
gamlss.family . . . . .	51
GB1 . . . . .	55
GB2 . . . . .	57
gen.Family . . . . .	59
GEOM . . . . .	61
GG . . . . .	63
GIG . . . . .	65
GPO . . . . .	67
GT . . . . .	69
GU . . . . .	71
hazardFun . . . . .	72
IG . . . . .	74
IGAMMA . . . . .	75
JSU . . . . .	77
JSUo . . . . .	79
LG . . . . .	82
LNO . . . . .	84
LO . . . . .	87
LOGITNO . . . . .	88
LQNO . . . . .	90
make.link.gamlss . . . . .	92
MN3 . . . . .	96
NBF . . . . .	98

NBI . . . . .	100
NBII . . . . .	102
NET . . . . .	104
NO . . . . .	106
NO2 . . . . .	108
NOF . . . . .	110
PARETO2 . . . . .	112
PE . . . . .	114
PIG . . . . .	116
PO . . . . .	118
RG . . . . .	120
RGE . . . . .	122
SEP . . . . .	124
SEP1 . . . . .	127
SHASH . . . . .	129
SI . . . . .	133
SICHEL . . . . .	135
SIMPLEX . . . . .	138
SN1 . . . . .	139
SN2 . . . . .	141
ST1 . . . . .	143
TF . . . . .	146
WARING . . . . .	148
WEI . . . . .	150
WEI2 . . . . .	152
WEI3 . . . . .	154
YULE . . . . .	156
ZABB . . . . .	157
ZABI . . . . .	159
ZAGA . . . . .	161
ZAIG . . . . .	163
ZANBI . . . . .	166
ZAP . . . . .	168
ZIP . . . . .	169
ZIP2 . . . . .	171
ZIPF . . . . .	173

**Index****176**

## Description

A set of distributions which can be used for modelling the response variables in Generalized Additive Models for Location Scale and Shape, Rigby and Stasinopoulos (2005), <doi:10.1111/j.1467-9876.2005.00510.x>. The distributions can be continuous, discrete or mixed distributions. Extra distributions can be created, by transforming, any continuous distribution defined on the real line, to a distribution defined on ranges 0 to infinity or 0 to 1, by using a "log" or a "logit" transformation respectively.

## Details

The DESCRIPTION file:

```
Package:      gamlss.dist
Type:         Package
Title:        Distributions for Generalized Additive Models for Location Scale and Shape
Version:      5.1-0
Date:         2018-10-06
Authors@R:    c(person("Mikis", "Stasinopoulos", role = c("aut", "cre", "cph"), email = "d.stasinopoulos@londonmet.ac.uk")
Description:  A set of distributions which can be used for modelling the response variables in Generalized Additive Models
License:      GPL-2 | GPL-3
URL:          http://www.gamlss.org/
Depends:      R (>= 2.15.0), MASS, graphics, stats, methods
Repository:   CRAN
Author:       Mikis Stasinopoulos [aut, cre, cph], Robert Rigby [aut], Calliope Akantziliotou [ctb], Vlasios Voudouris [ctb],
Maintainer:   Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>
```

Index of help topics:

BB	Beta Binomial Distribution For Fitting a GAMLSS Model
BCCG	Box-Cox Cole and Green distribution (or Box-Cox normal) for fitting a GAMLSS
BCPE	Box-Cox Power Exponential distribution for fitting a GAMLSS
BCT	Box-Cox t distribution for fitting a GAMLSS
BE	The beta distribution for fitting a GAMLSS
BEINF	The beta inflated distribution for fitting a GAMLSS
BEOI	The one-inflated beta distribution for fitting a GAMLSS
BEZI	The zero-inflated beta distribution for fitting a GAMLSS
BI	Binomial distribution for fitting a GAMLSS
BNB	Beta Negative Binomial distribution for fitting a GAMLSS
DBI	The Double binomial distribution
DBURR12	The Discrete Burr type XII distribution for

	fitting a GAMLSS model
DEL	The Delaporte distribution for fitting a GAMLSS model
DPO	The Double Poisson distribution
EGB2	The exponential generalized Beta type 2 distribution for fitting a GAMLSS
EXP	Exponential distribution for fitting a GAMLSS
GA	Gamma distribution for fitting a GAMLSS
GB1	The generalized Beta type 1 distribution for fitting a GAMLSS
GB2	The generalized Beta type 2 and generalized Pareto distributions for fitting a GAMLSS
GEOM	Geometric distribution for fitting a GAMLSS model
GG	Generalized Gamma distribution for fitting a GAMLSS
GIG	Generalized Inverse Gaussian distribution for fitting a GAMLSS
GPO	The generalised Poisson distribution
GT	The generalized t distribution for fitting a GAMLSS
GU	The Gumbel distribution for fitting a GAMLSS
IG	Inverse Gaussian distribution for fitting a GAMLSS
IGAMMA	Inverse Gamma distribution for fitting a GAMLSS
JSU	The Johnson's Su distribution for fitting a GAMLSS
JSUo	The original Johnson's Su distribution for fitting a GAMLSS
LG	Logarithmic and zero adjusted logarithmic distributions for fitting a GAMLSS model
LNO	Log Normal distribution for fitting in GAMLSS
LO	Logistic distribution for fitting a GAMLSS
LOGITNO	Logit Normal distribution for fitting in GAMLSS
LQNO	Normal distribution with a specific mean and variance relationship for fitting a GAMLSS model
MN3	Multinomial distribution in GAMLSS
NBF	Negative Binomial Family distribution for fitting a GAMLSS
NBI	Negative Binomial type I distribution for fitting a GAMLSS
NBII	Negative Binomial type II distribution for fitting a GAMLSS
NET	Normal Exponential t distribution (NET) for fitting a GAMLSS
NO	Normal distribution for fitting a GAMLSS
NO2	Normal distribution (with variance as sigma

	parameter) for fitting a GAMLSS
NOF	Normal distribution family for fitting a GAMLSS
PARETO2	Pareto Type 2 distribution for fitting a GAMLSS
PE	Power Exponential distribution for fitting a GAMLSS
PIG	The Poisson-inverse Gaussian distribution for fitting a GAMLSS model
PO	Poisson distribution for fitting a GAMLSS model
RG	The Reverse Gumbel distribution for fitting a GAMLSS
RGE	Reverse generalized extreme family distribution for fitting a GAMLSS
SEP	The Skew Power exponential (SEP) distribution for fitting a GAMLSS
SEP1	The Skew Power exponential type 1-4 distribution for fitting a GAMLSS
SHASH	The Sinh-Arcsinh (SHASH) distribution for fitting a GAMLSS
SI	The Sichel distribution for fitting a GAMLSS model
SICHEL	The Sichel distribution for fitting a GAMLSS model
SIMPLEX	The simplex distribution for fitting a GAMLSS
SN1	Skew Normal Type 1 distribution for fitting a GAMLSS
SN2	Skew Normal Type 2 distribution for fitting a GAMLSS
ST1	The skew t distributions, type 1 to 5
TF	t family distribution for fitting a GAMLSS
WARING	Waring distribution for fitting a GAMLSS model
WEI	Weibull distribution for fitting a GAMLSS
WEI2	A specific parameterization of the Weibull distribution for fitting a GAMLSS
WEI3	A specific parameterization of the Weibull distribution for fitting a GAMLSS
YULE	Yule distribution for fitting a GAMLSS model
ZABB	Zero inflated and zero adjusted Binomial distribution for fitting in GAMLSS
ZABI	Zero inflated and zero adjusted Binomial distribution for fitting in GAMLSS
ZAGA	The zero adjusted Gamma distribution for fitting a GAMLSS model
ZAIG	The zero adjusted Inverse Gaussian distribution for fitting a GAMLSS model
ZANBI	Zero inflated and zero adjusted negative binomial distributions for fitting a GAMLSS model
ZAP	Zero adjusted poisson distribution for fitting

	a GAMLSS model
ZIP	Zero inflated poisson distribution for fitting a GAMLSS model
ZIP2	Zero inflated poisson distribution for fitting a GAMLSS model
ZIPF	The zipf and zero adjusted zipf distributions for fitting a GAMLSS model
checklink	Set the Right Link Function for Specified Parameter and Distribution
exGAUS	The ex-Gaussian distribution
flexDist	Non-parametric pdf from limited information data
gamlss.dist-package	Distributions for Generalized Additive Models for Location Scale and Shape
gamlss.family	Family Objects for fitting a GAMLSS model
gen.Family	Functions to generate log and logit distributions from existing continuous gamlss.family distributions
hazardFun	Hazard functions for gamlss.family distributions
make.link.gamlss	Create a Link for GAMLSS families

### Author(s)

Mikis Stasinopoulos [aut, cre, cph], Robert Rigby [aut], Calliope Akantziliotou [ctb], Vlasios Voudouris [ctb], Gillian Heller [ctb], Fernanda De Bastiani [ctb], Raydonal Ospina [ctb], Nicoletta Motpan [ctb], Fiona McElduff [ctb], Majid Djennad [ctb], Marco Enea [ctb], Alexios Ghalanos [ctb], Christos Argyropoulos [ctb], Almond Stocker [ctb], Jens Lichter [ctb], Stanislaus Stadlmann [ctb]

Maintainer: Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>

### References

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#)

### Examples

```
# pdf plot
plot(function(y) dSICHEL(y, mu=10, sigma = 0.1 , nu=1 ),
```

```

                                from=0, to=30, n=30+1, type="h")
# cdf plot
PPP <- par(mfrow=c(2,1))
plot(function(y) pSICHEL(y, mu=10, sigma =0.1, nu=1 ),
      from=0, to=30, n=30+1, type="h") # cdf
cdf<-pSICHEL(0:30, mu=10, sigma=0.1, nu=1)
sfun1 <- stepfun(1:30, cdf, f = 0)
plot(sfun1, xlim=c(0,30), main="cdf(x)")
par(PPP)

```

BB

---

*Beta Binomial Distribution For Fitting a GAMLSS Model*


---

**Description**

This function defines the beta binomial distribution, a two parameter distribution, for a `gamlss.family` object to be used in a GAMLSS fitting using the function `gamlss()`

**Usage**

```

BB(mu.link = "logit", sigma.link = "log")
dBB(x, mu = 0.5, sigma = 1, bd = 10, log = FALSE)
pBB(q, mu = 0.5, sigma = 1, bd = 10, lower.tail = TRUE,
    log.p = FALSE)
qBB(p, mu = 0.5, sigma = 1, bd = 10, lower.tail = TRUE,
    log.p = FALSE, fast = FALSE)
rBB(n, mu = 0.5, sigma = 1, bd = 10, fast = FALSE)

```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "logit" link as the default for the mu parameter. Other links are "probit" and "cloglog"(complementary log-log)
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the sigma parameter. Other links are "inverse", "identity" and "sqrt"
<code>mu</code>	vector of positive probabilities
<code>sigma</code>	the dispersion parameter
<code>bd</code>	vector of binomial denominators
<code>p</code>	vector of probabilities
<code>x, q</code>	vector of quantiles
<code>n</code>	number of random values to return
<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>fast</code>	a logical variable if <code>fast=TRUE</code> the <code>dBB</code> function is used in the calculation of the inverse c.d.f function. This is faster to the default <code>fast=FALSE</code> , where the <code>pBB{}</code> is used, but not always consistent with the results obtained from <code>pBB()</code> , for example if <code>p &lt;- pBB(c(0,1,2,3,4,5), mu=.5, sigma=1, bd=5)</code> do not ensure that <code>qBB(p, mu=.5, sigma=1, bd=5)</code> will be <code>c(0,1,2,3,4,5)</code>



**Details**

Definition file for beta binomial distribution.

$$f(y|\mu, \sigma) = \frac{\Gamma(n+1)}{\Gamma(y+1)\Gamma(n-y+1)} \frac{\Gamma(\frac{1}{\sigma})\Gamma(y+\frac{\mu}{\sigma})\Gamma[n+\frac{(1-\mu)}{\sigma}-y]}{\Gamma(n+\frac{1}{\sigma})\Gamma(\frac{\mu}{\sigma})\Gamma(\frac{1-\mu}{\sigma})}$$

for  $y = 0, 1, 2, \dots, n$ ,  $0 < \mu < 1$  and  $\sigma > 0$ . For  $\mu = 0.5$  and  $\sigma = 0.5$  the distribution is uniform.

**Value**

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

**Warning**

The functions `pBB` and `qBB` are calculated using a laborious procedure so they are relatively slow.

**Note**

The response variable should be a matrix containing two columns, the first with the count of successes and the second with the count of failures. The parameter `mu` represents a probability parameter with limits  $0 < \mu < 1$ .  $n\mu$  is the mean of the distribution where  $n$  is the binomial denominator.  $\{n\mu(1-\mu)[1+(n-1)\sigma/(\sigma+1)]\}^{0.5}$  is the standard deviation of the Beta Binomial distribution. Hence  $\sigma$  is a dispersion type parameter

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby and Kalliope Akantziliotou

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [BI](#),

## Examples

```
# BB()# gives information about the default links for the Beta Binomial distribution
#plot the pdf
plot(function(y) dBB(y, mu = .5, sigma = 1, bd =40), from=0, to=40, n=40+1, type="h")
#calculate the cdf and plotting it
ppBB <- pBB(seq(from=0, to=40), mu=.2 , sigma=3, bd=40)
plot(0:40,ppBB, type="h")
#calculating quantiles and plotting them
qqBB <- qBB(ppBB, mu=.2 , sigma=3, bd=40)
plot(qqBB~ ppBB)
# when the argument fast is useful
p <- pBB(c(0,1,2,3,4,5), mu=.01 , sigma=1, bd=5)
qBB(p, mu=.01 , sigma=1, bd=5, fast=TRUE)
# 0 1 1 2 3 5
qBB(p, mu=.01 , sigma=1, bd=5, fast=FALSE)
# 0 1 2 3 4 5
# generate random sample
tN <- table(Ni <- rBB(1000, mu=.2, sigma=1, bd=20))
r <- barplot(tN, col='lightblue')
# fitting a model
# library(gamlss)
#data(aep)
# fits a Beta-Binomial model
#h<-gamlss(y~ward+loglos+year, sigma.formula=~year+ward, family=BB, data=aep)
```

BCCG

*Box-Cox Cole and Green distribution (or Box-Cox normal) for fitting a GAMLSS*

## Description

The function BCCG defines the Box-Cox Cole and Green distribution (Box-Cox normal), a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dBCCG`, `pBCCG`, `qBCCG` and `rBCCG` define the density, distribution function, quantile function and random generation for the specific parameterization of the Box-Cox Cole and Green distribution. [The function `BCCGuntr()` is the original version of the function suitable only for the untruncated Box-Cox Cole and Green distribution See Cole and Green (1992) and Rigby and Stasinopoulos (2003a,2003b) for details. The function `BCCGo` is identical to `BCCG` but with log link for  $\mu$ .

## Usage

```
BCCG(mu.link = "identity", sigma.link = "log", nu.link = "identity")
BCCGo(mu.link = "log", sigma.link = "log", nu.link = "identity")
BCCGuntr(mu.link = "identity", sigma.link = "log", nu.link = "identity")
dBCCG(x, mu = 1, sigma = 0.1, nu = 1, log = FALSE)
pBCCG(q, mu = 1, sigma = 0.1, nu = 1, lower.tail = TRUE, log.p = FALSE)
qBCCG(p, mu = 1, sigma = 0.1, nu = 1, lower.tail = TRUE, log.p = FALSE)
```

```

rBCCG(n, mu = 1, sigma = 0.1, nu = 1)
dBCCG(x, mu = 1, sigma = 0.1, nu = 1, log = FALSE)
pBCCG(q, mu = 1, sigma = 0.1, nu = 1, lower.tail = TRUE, log.p = FALSE)
qBCCG(p, mu = 1, sigma = 0.1, nu = 1, lower.tail = TRUE, log.p = FALSE)
rBCCG(n, mu = 1, sigma = 0.1, nu = 1)

```

### Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" link as the default for the <code>mu</code> parameter, other links are "inverse", "log" and "own"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter, other links are "inverse", "identity" and "own"
<code>nu.link</code>	Defines the <code>nu.link</code> , with "identity" link as the default for the <code>nu</code> parameter, other links are "inverse", "log" and "own"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of skewness parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

### Details

The probability distribution function of the untruncated Box-Cox Cole and Green distribution, `BCCGuntr`, is defined as

$$f(y|\mu, \sigma, \nu) = \frac{1}{\sqrt{2\pi}\sigma} \frac{y^{\nu-1}}{\mu^\nu} \exp\left(-\frac{z^2}{2}\right)$$

where if  $\nu \neq 0$  then  $z = [(y/\mu)^\nu - 1]/(\nu\sigma)$  else  $z = \log(y/\mu)/\sigma$ , for  $y > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu = (-\infty, +\infty)$ .

The Box-Cox Cole and Green distribution, `BCCG`, adjusts the above density  $f(y|\mu, \sigma, \nu)$  for the truncation resulting from the condition  $y > 0$ . See Rigby and Stasinopoulos (2003a, 2003b) for details.

### Value

`BCCG()` returns a `gamLss.family` object which can be used to fit a Cole and Green distribution in the `gamLss()` function. `dBCCG()` gives the density, `pBCCG()` gives the distribution function, `qBCCG()` gives the quantile function, and `rBCCG()` generates random deviates.

**Warning**

The BCCG distribution may be unsuitable for some combinations of the parameters (mainly for large  $\sigma$ ) where the integrating constant is less than 0.99. A warning will be given if this is the case. The BCCG distribution is suitable for all combinations of the distributional parameters within their range [i.e.  $\mu > 0, \sigma > 0, \nu = (-\infty, +\infty)$ ]

**Note**

$\mu$  is the median of the distribution  $\sigma$  is approximately the coefficient of variation (for small values of  $\sigma$ ), and  $\nu$  controls the skewness.

The BCCG distribution is suitable for all combinations of the parameters within their ranges [i.e.  $\mu > 0, \sigma > 0$ , and  $\nu = (-\infty, \infty)$  ]

**Author(s)**

Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Kalliope Akantziliotou

**References**

- Cole, T. J. and Green, P. J. (1992) Smoothing reference centile curves: the LMS method and penalized likelihood, *Statist. Med.* **11**, 1305–1319
- Rigby, R. A. and Stasinopoulos, D. M. (2004). Smooth centile curves for skew and kurtotic data modelled using the Box-Cox Power Exponential distribution. *Statistics in Medicine*, **23**: 3053-3076.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape, (with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Rigby, R.A. Stasinopoulos, D.M. (2006). Using the Box-Cox  $t$  distribution in GAMLSS to model skewness and kurtosis. to appear in *Statistical Modelling*.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [BCPE](#), [BCT](#)

**Examples**

```
BCCG() # gives information about the default links for the Cole and Green distribution
# library(gamlss)
# data(abdom)
# h<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=BCCG, data=abdom)
```

```
#plot(h)
plot(function(x) dBCCG(x, mu=5,sigma=.5,nu=-1), 0.0, 20,
      main = "The BCCG density mu=5,sigma=.5,nu=-1")
plot(function(x) pBCCG(x, mu=5,sigma=.5,nu=-1), 0.0, 20,
      main = "The BCCG cdf mu=5, sigma=.5, nu=-1")
```

---

BCPE

*Box-Cox Power Exponential distribution for fitting a GAMLSS*


---

### Description

This function defines the Box-Cox Power Exponential distribution, a four parameter distribution, for a `gamlss.family` object to be used for a GAMLSS fitting using the function `gamlss()`. The functions `dBCE`, `pBCE`, `qBCE` and `rBCE` define the density, distribution function, quantile function and random generation for the Box-Cox Power Exponential distribution. The function `checkBCE` can be used, typically when a BCE model is fitted, to check whether there exit a turning point of the distribution close to zero. It give the number of values of the response below their minimum turning point and also the maximum probability of the lower tail below minimum turning point. [The function `Biventer()` is the original version of the function suitable only for the untruncated BCE distribution.] See Rigby and Stasinopoulos (2003) for details. The function `BCEo` is identical to BCE but with log link for mu.

### Usage

```
BCPE(mu.link = "identity", sigma.link = "log", nu.link = "identity",
      tau.link = "log")
BCPEo(mu.link = "log", sigma.link = "log", nu.link = "identity",
       tau.link = "log")
BCPEuntr(mu.link = "identity", sigma.link = "log", nu.link = "identity",
         tau.link = "log")
dBCE(x, mu = 5, sigma = 0.1, nu = 1, tau = 2, log = FALSE)
pBCE(q, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
qBCE(p, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
rBCE(n, mu = 5, sigma = 0.1, nu = 1, tau = 2)
dBCEo(x, mu = 5, sigma = 0.1, nu = 1, tau = 2, log = FALSE)
pBCEo(q, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE,
      log.p = FALSE)
qBCEo(p, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE,
      log.p = FALSE)
rBCEo(n, mu = 5, sigma = 0.1, nu = 1, tau = 2)
checkBCE(obj = NULL, mu = 10, sigma = 0.1, nu = 0.5, tau = 2,...)
```

### Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" link as the default for the mu parameter. Other links are "inverse", "log" and "own"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the sigma parameter. Other links are "inverse", "identity" and "own"

<code>nu.link</code>	Defines the <code>nu.link</code> , with "identity" link as the default for the <code>nu</code> parameter. Other links are "inverse", "log" and "own"
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the <code>tau</code> parameter. Other links are "logshifted", "identity" and "own"
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of <code>nu</code> parameter values
<code>tau</code>	vector of <code>tau</code> parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required
<code>obj</code>	a <code>gamlss</code> BCPE family object
<code>...</code>	for extra arguments

### Details

The probability density function of the untruncated Box Cox Power Exponential distribution, (`BCPE.untr`), is defined as

$$f(y|\mu, \sigma, \nu, \tau) = \frac{y^{\nu-1} \tau \exp[-\frac{1}{2} | \frac{z}{c} |^{\tau}]}{\mu^{\nu} \sigma c 2^{(1+1/\tau)} \Gamma(\frac{1}{\tau})}$$

where  $c = [2^{(-2/\tau)} \Gamma(1/\tau) / \Gamma(3/\tau)]^{0.5}$ , where if  $\nu \neq 0$  then  $z = [(y/\mu)^{\nu} - 1] / (\nu\sigma)$  else  $z = \log(y/\mu) / \sigma$ , for  $y > 0$ ,  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu = (-\infty, +\infty)$  and  $\tau > 0$ .

The Box-Cox Power Exponential, BCPE, adjusts the above density  $f(y|\mu, \sigma, \nu, \tau)$  for the truncation resulting from the condition  $y > 0$ . See Rigby and Stasinopoulos (2003) for details.

### Value

`BCPE()` returns a `gamlss.family` object which can be used to fit a Box Cox Power Exponential distribution in the `gamlss()` function. `dBCPE()` gives the density, `pBCPE()` gives the distribution function, `qBCPE()` gives the quantile function, and `rBCPE()` generates random deviates.

### Warning

The `BCPE.untr` distribution may be unsuitable for some combinations of the parameters (mainly for large  $\sigma$ ) where the integrating constant is less than 0.99. A warning will be given if this is the case.

The BCPE distribution is suitable for all combinations of the parameters within their ranges [i.e.  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu = (-\infty, \infty)$  and  $\tau > 0$ ]

**Note**

$\mu$ , is the median of the distribution,  $\sigma$  is approximately the coefficient of variation (for small  $\sigma$  and moderate  $\nu > 0$ ),  $\nu$  controls the skewness and  $\tau$  the kurtosis of the distribution

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Rigby, R. A. and Stasinopoulos, D. M. (2004). Smooth centile curves for skew and kurtotic data modelled using the Box-Cox Power Exponential distribution. *Statistics in Medicine*, **23**: 3053-3076.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), BCT

**Examples**

```
# BCPE() #
# library(gamlss)
# data(abdom)
#h<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=BCPE, data=abdom)
#plot(h)
plot(function(x)dBCPE(x, mu=5,sigma=.5,nu=1, tau=3), 0.0, 15,
      main = "The BCPE density mu=5,sigma=.5,nu=1, tau=3")
plot(function(x) pBCPE(x, mu=5,sigma=.5,nu=1, tau=3), 0.0, 15,
      main = "The BCPE cdf mu=5, sigma=.5, nu=1, tau=3")
```

## Description

The function `BCT()` defines the Box-Cox t distribution, a four parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dBCT`, `pBCT`, `qBCT` and `rBCT` define the density, distribution function, quantile function and random generation for the Box-Cox t distribution. [The function `BCTuntr()` is the original version of the function suitable only for the untruncated BCT distribution]. See Rigby and Stasinopoulos (2003) for details. The function `BCT` is identical to `BCT` but with log link for  $\mu$ .

## Usage

```
BCT(mu.link = "identity", sigma.link = "log", nu.link = "identity",
    tau.link = "log")
BCTo(mu.link = "log", sigma.link = "log", nu.link = "identity",
    tau.link = "log")
BCTuntr(mu.link = "identity", sigma.link = "log", nu.link = "identity",
    tau.link = "log")
dBCT(x, mu = 5, sigma = 0.1, nu = 1, tau = 2, log = FALSE)
pBCT(q, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
qBCT(p, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
rBCT(n, mu = 5, sigma = 0.1, nu = 1, tau = 2)
dBCTo(x, mu = 5, sigma = 0.1, nu = 1, tau = 2, log = FALSE)
pBCTo(q, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
qBCTo(p, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
rBCTo(n, mu = 5, sigma = 0.1, nu = 1, tau = 2)
```

## Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" link as the default for the $\mu$ parameter. Other links are "inverse", "log" and "own"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the $\sigma$ parameter. Other links are "inverse", "identity", "own"
<code>nu.link</code>	Defines the <code>nu.link</code> , with "identity" link as the default for the $\nu$ parameter. Other links are "inverse", "log", "own"
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the $\tau$ parameter. Other links are "inverse", "identity" and "own"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of $\nu$ parameter values
<code>tau</code>	vector of $\tau$ parameter values
<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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required



## Details

The probability density function of the untruncated Box-Cox  $t$  distribution, `BCTuntr`, is given by

$$f(y|\mu, \sigma, \nu, \tau) = \frac{y^{\nu-1}}{\mu^\nu \sigma} \frac{\Gamma[(\tau+1)/2]}{\Gamma(1/2)\Gamma(\tau/2)\tau^{0.5}} [1 + (1/\tau)z^2]^{-(\tau+1)/2}$$

where if  $\nu \neq 0$  then  $z = [(y/\mu)^\nu - 1]/(\nu\sigma)$  else  $z = \log(y/\mu)/\sigma$ , for  $y > 0$ ,  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu = (-\infty, +\infty)$  and  $\tau > 0$ .

The Box-Cox  $t$  distribution, `BCT`, adjusts the above density  $f(y|\mu, \sigma, \nu, \tau)$  for the truncation resulting from the condition  $y > 0$ . See Rigby and Stasinopoulos (2003) for details.

## Value

`BCT()` returns a `gamlss.family` object which can be used to fit a Box-Cox- $t$  distribution in the `gamlss()` function. `dBCT()` gives the density, `pBCT()` gives the distribution function, `qBCT()` gives the quantile function, and `rBCT()` generates random deviates.

## Warning

The use `BCTuntr` distribution may be unsuitable for some combinations of the parameters (mainly for large  $\sigma$ ) where the integrating constant is less than 0.99. A warning will be given if this is the case.

The `BCT` distribution is suitable for all combinations of the parameters within their ranges [i.e.  $\mu > 0$ ,  $\sigma > 0$ ,  $\nu = (-\infty, \infty)$  and  $\tau > 0$ ]

## Note

$\mu$  is the median of the distribution,  $\sigma(\frac{\tau}{\tau-2})^{0.5}$  is approximate the coefficient of variation (for small  $\sigma$  and moderate  $\nu > 0$  and moderate or large  $\tau$ ),  $\nu$  controls the skewness and  $\tau$  the kurtosis of the distribution

## Author(s)

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

## References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape, (with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Rigby, R.A. Stasinopoulos, D.M. (2006). Using the Box-Cox  $t$  distribution in GAMLSS to model skewness and kurtosis. to appear in *Statistical Modelling*.
- Stasinopoulos, D. M. Rigby, R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [BCPE](#), [BCCG](#)

**Examples**

```
BCT() # gives information about the default links for the Box Cox t distribution
# library(gamlss)
#data(abdom)
#h<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=BCT, data=abdom) #
#plot(h)
plot(function(x)dBCT(x, mu=5,sigma=.5,nu=1, tau=2), 0.0, 20,
      main = "The BCT density mu=5,sigma=.5,nu=1, tau=2")
plot(function(x) pBCT(x, mu=5,sigma=.5,nu=1, tau=2), 0.0, 20,
      main = "The BCT cdf mu=5, sigma=.5, nu=1, tau=2")
```

---

 BE

*The beta distribution for fitting a GAMLSS*


---

**Description**

The functions `BE()` and `BEo()` define the beta distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. `BE()` has mean equal to the parameter `mu` and `sigma` as scale parameter, see below. `BEo()` is the original parameterizations of the beta distribution as in `dbeta()` with `shape1=mu` and `shape2=sigma`. The functions `dBE` and `dBEo`, `pBE` and `pBEo`, `qBE` and `qBEo` and finally `rBE` and `rBEo` define the density, distribution function, quantile function and random generation for the `BE` and `BEo` parameterizations respectively of the beta distribution.

**Usage**

```
BE(mu.link = "logit", sigma.link = "logit")
dBE(x, mu = 0.5, sigma = 0.2, log = FALSE)
pBE(q, mu = 0.5, sigma = 0.2, lower.tail = TRUE, log.p = FALSE)
qBE(p, mu = 0.5, sigma = 0.2, lower.tail = TRUE, log.p = FALSE)
rBE(n, mu = 0.5, sigma = 0.2)
BEo(mu.link = "log", sigma.link = "log")
dBEo(x, mu = 0.5, sigma = 0.2, log = FALSE)
pBEo(q, mu = 0.5, sigma = 0.2, lower.tail = TRUE, log.p = FALSE)
qBEo(p, mu = 0.5, sigma = 0.2, lower.tail = TRUE, log.p = FALSE)
```

**Arguments**

<code>mu.link</code>	the mu link function with default <code>logit</code>
<code>sigma.link</code>	the sigma link function with default <code>logit</code>
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values

sigma	vector of scale parameter values
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required

### Details

The original beta distribution is given as

$$f(y|\alpha, \beta) = \frac{1}{B(\alpha, \beta)} y^{\alpha-1} (1-y)^{\beta-1}$$

for  $y \in (0, 1)$ ,  $\alpha > 0$  and  $\beta > 0$ . In the `gamlss` implementation of `BEo`  $\alpha = \mu$  and  $\beta = \sigma$ . The reparametrization in the function `BE()` is  $\mu = \frac{\alpha}{\alpha+\beta}$  and  $\sigma = \frac{1}{\alpha+\beta+1}$  for  $\mu \in (0, 1)$  and  $\sigma \in (0, 1)$ . The expected value of  $y$  is  $\mu$  and the variance is  $\sigma^2 \mu * (1 - \mu)$ .

### Value

`BE()` and `BEo()` return a `gamlss.family` object which can be used to fit a beta distribution in the `gamlss()` function.

### Note

Note that for `BE`, `mu` is the mean and `sigma` a scale parameter contributing to the variance of  $y$

### Author(s)

Bob Rigby and Mikis Stasinopoulos

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [BE](#), [LOGITNO](#), [GB1](#), [BEINF](#)

## Examples

```
BE()# gives information about the default links for the beta distribution
dat1<-rBE(100, mu=.3, sigma=.5)
hist(dat1)
#library(gamlss)
# mod1<-gamlss(dat1~1,family=BE) # fits a constant for mu and sigma
#fitted(mod1)[1]
#fitted(mod1,"sigma")[1]
plot(function(y) dBE(y, mu=.1 ,sigma=.5), 0.001, .999)
plot(function(y) pBE(y, mu=.1 ,sigma=.5), 0.001, 0.999)
plot(function(y) qBE(y, mu=.1 ,sigma=.5), 0.001, 0.999)
plot(function(y) qBE(y, mu=.1 ,sigma=.5, lower.tail=FALSE), 0.001, .999)
dat2<-rBEo(100, mu=1, sigma=2)
#mod2<-gamlss(dat2~1,family=BEo) # fits a constant for mu and sigma
#fitted(mod2)[1]
#fitted(mod2,"sigma")[1]
```

---

BEINF

*The beta inflated distribution for fitting a GAMLSS*

---

## Description

The function `BEINF()` defines the beta inflated distribution, a four parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The beta inflated is similar to the beta but allows zeros and ones as values for the response variable. The two extra parameters model the probabilities at zero and one.

The functions `BEINF0()` and `BEINF1()` are three parameter beta inflated distributions allowing zeros or ones only at the response respectively. `BEINF0()` and `BEINF1()` are re-parameterize versions of the distributions `BEZI` and `BEOI` contributed to `gamlss` by Raydonal Ospina (see Ospina and Ferrari (2010)).

The functions `dBEINF`, `pBEINF`, `qBEINF` and `rBEINF` define the density, distribution function, quantile function and random generation for the `BEINF` parametrization of the beta inflated distribution.

The functions `dBEINF0`, `pBEINF0`, `qBEINF0` and `rBEINF0` define the density, distribution function, quantile function and random generation for the `BEINF0` parametrization of the beta inflated at zero distribution.

The functions `dBEINF1`, `pBEINF1`, `qBEINF1` and `rBEINF1` define the density, distribution function, quantile function and random generation for the `BEINF1` parametrization of the beta inflated at one distribution.

`plotBEINF`, `plotBEINF0` and `plotBEINF1` can be used to plot the distributions. `meanBEINF`, `meanBEINF0` and `meanBEINF1` calculates the expected value of the response for a fitted model.

## Usage

```
BEINF(mu.link = "logit", sigma.link = "logit", nu.link = "log",
      tau.link = "log")
BEINF0(mu.link = "logit", sigma.link = "logit", nu.link = "log")
```

```

BEINF1(mu.link = "logit", sigma.link = "logit", nu.link = "log")

dBEINF(x, mu = 0.5, sigma = 0.1, nu = 0.1, tau = 0.1,
       log = FALSE)
dBEINF0(x, mu = 0.5, sigma = 0.1, nu = 0.1, log = FALSE)
dBEINF1(x, mu = 0.5, sigma = 0.1, nu = 0.1, log = FALSE)

pBEINF(q, mu = 0.5, sigma = 0.1, nu = 0.1, tau = 0.1,
       lower.tail = TRUE, log.p = FALSE)
pBEINF0(q, mu = 0.5, sigma = 0.1, nu = 0.1,
       lower.tail = TRUE, log.p = FALSE)
pBEINF1(q, mu = 0.5, sigma = 0.1, nu = 0.1,
       lower.tail = TRUE, log.p = FALSE)

qBEINF(p, mu = 0.5, sigma = 0.1, nu = 0.1, tau = 0.1,
       lower.tail = TRUE, log.p = FALSE)
qBEINF0(p, mu = 0.5, sigma = 0.1, nu = 0.1, tau = 0.1,
       lower.tail = TRUE, log.p = FALSE)
qBEINF1(p, mu = 0.5, sigma = 0.1, nu = 0.1,
       lower.tail = TRUE, log.p = FALSE)

rBEINF(n, mu = 0.5, sigma = 0.1, nu = 0.1, tau = 0.1)
rBEINF0(n, mu = 0.5, sigma = 0.1, nu = 0.1)
rBEINF1(n, mu = 0.5, sigma = 0.1, nu = 0.1)

plotBEINF(mu = 0.5, sigma = 0.5, nu = 0.5, tau = 0.5,
         from = 0.001, to = 0.999, n = 101, ...)
plotBEINF0(mu = 0.5, sigma = 0.5, nu = 0.5,
         from = 1e-04, to = 0.9999, n = 101, ...)
plotBEINF1(mu = 0.5, sigma = 0.5, nu = 0.5,
         from = 1e-04, to = 0.9999, n = 101, ...)

meanBEINF(obj)
meanBEINF0(obj)
meanBEINF1(obj)

```

### Arguments

<code>mu.link</code>	the mu link function with default logit
<code>sigma.link</code>	the sigma link function with default logit
<code>nu.link</code>	the nu link function with default log
<code>tau.link</code>	the tau link function with default log
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of parameter values modelling the probability at zero

tau	vector of parameter values modelling the probability at one
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required
from	where to start plotting the distribution from
to	up to where to plot the distribution
obj	a fitted BEINF object
...	other graphical parameters for plotting

### Details

The beta inflated distribution is given as

$$f(y) = p_0$$

if (y=0)

$$f(y) = p_1$$

if (y=1)

$$f(y|\alpha, \beta) = \frac{1}{B(\alpha, \beta)} y^{\alpha-1} (1-y)^{\beta-1}$$

otherwise

for  $y = (0, 1)$ ,  $\alpha > 0$  and  $\beta > 0$ . The parametrization in the function BEINF() is  $\mu = \frac{\alpha}{\alpha+\beta}$  and  $\sigma = \frac{1}{\alpha+\beta+1}$  for  $\mu = (0, 1)$  and  $\sigma = (0, 1)$  and  $\nu = \frac{p_0}{p_2}$ ,  $\tau = \frac{p_1}{p_2}$  where  $p_2 = 1 - p_0 - p_1$ .

### Value

returns a `gamlss.family` object which can be used to fit a beta inflated distribution in the `gamlss()` function. ...

### Author(s)

Bob Rigby and Mikis Stasinopoulos

### References

- Ospina R. and Ferrari S. L. P. (2010) Inflated beta distributions, *Statistical Papers*, **23**, 111-126.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [BE](#), [BEo](#), [BEZI](#), [BEOI](#)

### Examples

```
BEINF()# gives information about the default links for the beta inflated distribution
BEINF0()
BEINF1()
# plotting the distributions
op<-par(mfrow=c(2,2))
plotBEINF( mu =.5 , sigma=.5, nu = 0.5, tau = 0.5, from = 0, to=1, n = 101)
plotBEINF0( mu =.5 , sigma=.5, nu = 0.5, from = 0, to=1, n = 101)
plotBEINF1( mu =.5 , sigma=.5, nu = 0.5, from = 0.001, to=1, n = 101)
curve(dBE(x, mu =.5, sigma=.5), 0.01, 0.999)
par(op)
# plotting the cdf
op<-par(mfrow=c(2,2))
plotBEINF( mu =.5 , sigma=.5, nu = 0.5, tau = 0.5, from = 0, to=1, n = 101, main="BEINF")
plotBEINF0( mu =.5 , sigma=.5, nu = 0.5, from = 0, to=1, n = 101, main="BEINF0")
plotBEINF1( mu =.5 , sigma=.5, nu = 0.5, from = 0.001, to=1, n = 101, main="BEINF1")
curve(dBE(x, mu =.5, sigma=.5), 0.01, 0.999, main="BE")
par(op)
#-----
op<-par(mfrow=c(2,2))
plotBEINF( mu =.5 , sigma=.5, nu = 0.5, tau = 0.5, from = 0, to=1, n = 101, main="BEINF")
plotBEINF0( mu =.5 , sigma=.5, nu = 0.5, from = 0, to=1, n = 101, main="BEINF0")
plotBEINF1( mu =.5 , sigma=.5, nu = 0.5, from = 0.001, to=1, n = 101, main="BEINF1")
curve(dBE(x, mu =.5, sigma=.5), 0.01, 0.999, main="BE")
par(op)
#-----
op<-par(mfrow=c(2,2))
curve( pBEINF(x, mu=.5 ,sigma=.5, nu = 0.5, tau = 0.5, ), 0, 1, ylim=c(0,1), main="BEINF" )
curve(pBEINF0(x, mu=.5 ,sigma=.5, nu = 0.5), 0, 1, ylim=c(0,1), main="BEINF0")
curve(pBEINF1(x, mu=.5 ,sigma=.5, nu = 0.5), 0, 1, ylim=c(0,1), main="BEINF1")
curve( pBE(x, mu=.5 ,sigma=.5), .001, .99, ylim=c(0,1), main="BE")
par(op)
#-----
op<-par(mfrow=c(2,2))
curve(qBEINF(x, mu=.5 ,sigma=.5, nu = 0.5, tau = 0.5), .01, .99, main="BEINF" )
curve(qBEINF0(x, mu=.5 ,sigma=.5, nu = 0.5), .01, .99, main="BEINF0" )
curve(qBEINF1(x, mu=.5 ,sigma=.5, nu = 0.5), .01, .99, main="BEINF1" )
curve(qBE(x, mu=.5 ,sigma=.5), .01, .99 , main="BE")
par(op)
#-----
op<-par(mfrow=c(2,2))
hist(rBEINF(200, mu=.5 ,sigma=.5, nu = 0.5, tau = 0.5))
hist(rBEINF0(200, mu=.5 ,sigma=.5, nu = 0.5))
hist(rBEINF1(200, mu=.5 ,sigma=.5, nu = 0.5))
```

```

hist(rBE(200, mu=.5 ,sigma=.5))
par(op)
# fit a model to the data
# library(gamlss)
#m1<-gamlss(dat~1,family=BEINF)
#meanBEINF(m1)[1]

```

---

BEOI

*The one-inflated beta distribution for fitting a GAMLSS*


---

### Description

The function `BEOI()` defines the one-inflated beta distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The one-inflated beta is similar to the beta distribution but allows ones as y values. This distribution is an extension of the beta distribution using a parameterization of the beta law that is indexed by mean and precision parameters (Ferrari and Cribari-Neto, 2004). The extra parameter models the probability at one. The functions `dBEOI`, `pBEOI`, `qBEOI` and `rBEOI` define the density, distribution function, quantile function and random generation for the BEOI parameterization of the one-inflated beta distribution. `plotBEOI` can be used to plot the distribution. `meanBEOI` calculates the expected value of the response for a fitted model.

### Usage

```

BEOI(mu.link = "logit", sigma.link = "log", nu.link = "logit")

dBEOI(x, mu = 0.5, sigma = 1, nu = 0.1, log = FALSE)

pBEOI(q, mu = 0.5, sigma = 1, nu = 0.1, lower.tail = TRUE, log.p = FALSE)

qBEOI(p, mu = 0.5, sigma = 1, nu = 0.1, lower.tail = TRUE,
      log.p = FALSE)

rBEOI(n, mu = 0.5, sigma = 1, nu = 0.1)

plotBEOI(mu = .5, sigma = 1, nu = 0.1, from = 0.001, to = 1, n = 101,
        ...)

meanBEOI(obj)

```

### Arguments

<code>mu.link</code>	the mu link function with default logit
<code>sigma.link</code>	the sigma link function with default log
<code>nu.link</code>	the nu link function with default logit
<code>x,q</code>	vector of quantiles



<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of precision parameter values
<code>nu</code>	vector of parameter values modelling the probability at one
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required
<code>from</code>	where to start plotting the distribution from
<code>to</code>	up to where to plot the distribution
<code>obj</code>	a fitted BEOI object
<code>...</code>	other graphical parameters for plotting

### Details

The one-inflated beta distribution is given as

$$f(y) = \nu$$

if  $(y = 1)$

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

if  $y = (0, 1)$ . The parameters satisfy  $0 < \mu < 1$ ,  $\sigma > 0$  and  $0 < \nu < 1$ .

Here  $E(y) = \nu + (1 - \nu)\mu$  and  $Var(y) = (1 - \nu) \frac{\mu(1 - \mu)}{\sigma + 1} + \nu(1 - \nu)(1 - \mu)^2$ .

### Value

returns a `gamlss.family` object which can be used to fit a one-inflated beta distribution in the `gamlss()` function.

### Note

This work is part of my PhD project at the University of Sao Paulo under the supervision of Professor Silvia Ferrari. My thesis is concerned with regression modelling of rates and proportions with excess of zeros and/or ones

### Author(s)

Raydonal Ospina, Department of Statistics, University of Sao Paulo, Brazil.

<rospina@ime.usp.br>

## References

- Ferrari, S.L.P., Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31** (1), 799-815.
- Ospina R. and Ferrari S. L. P. (2010) Inflated beta distributions, *Statistical Papers*, **23**, 111-126.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape (with discussion). *Applied Statistics*, **54** (3), 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006). Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family](#), [BEOI](#)

## Examples

```
BEOI()# gives information about the default links for the BEOI distribution
# plotting the distribution
plotBEOI( mu =0.5 , sigma=5, nu = 0.1, from = 0.001, to=1, n = 101)
# plotting the cdf
plot(function(y) pBEOI(y, mu=.5 ,sigma=5, nu=0.1), 0.001, 0.999)
# plotting the inverse cdf
plot(function(y) qBEOI(y, mu=.5 ,sigma=5, nu=0.1), 0.001, 0.999)
# generate random numbers
dat<-rBEOI(100, mu=.5, sigma=5, nu=0.1)
# fit a model to the data.
# library(gamlss)
#mod1<-gamlss(dat~1,sigma.formula=~1, nu.formula=~1, family=BEOI)
#fitted(mod1)[1]
#summary(mod1)
#fitted(mod1,"mu")[1]      #fitted mu
#fitted(mod1,"sigma")[1]  #fitted sigma
#fitted(mod1,"nu")[1]     #fitted nu
#meanBEOI(mod1)[1] # expected value of the response
```

## Description

The function `BEZI()` defines the zero-inflated beta distribution, a three parameter distribution, for a `gamlss.family` object to be used in `GAMLSS` fitting using the function `gamlss()`. The zero-inflated beta is similar to the beta distribution but allows zeros as `y` values. This distribution is an extension of the beta distribution using a parameterization of the beta law that is indexed by mean and precision parameters (Ferrari and Cribari-Neto, 2004). The extra parameter models the probability at zero. The functions `dBEZI`, `pBEZI`, `qBEZI` and `rBEZI` define the density, distribution function, quantile function and random generation for the BEZI parameterization of the zero-inflated beta distribution. `plotBEZI` can be used to plot the distribution. `meanBEZI` calculates the expected value of the response for a fitted model.

## Usage

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

dBEZI(x, mu = 0.5, sigma = 1, nu = 0.1, log = FALSE)

pBEZI(q, mu = 0.5, sigma = 1, nu = 0.1, lower.tail = TRUE, log.p = FALSE)

qBEZI(p, mu = 0.5, sigma = 1, nu = 0.1, lower.tail = TRUE,
      log.p = FALSE)

rBEZI(n, mu = 0.5, sigma = 1, nu = 0.1)

plotBEZI(mu = .5, sigma = 1, nu = 0.1, from = 0, to = 0.999, n = 101,
         ...)

meanBEZI(obj)
```

## Arguments

<code>mu.link</code>	the mu link function with default <code>logit</code>
<code>sigma.link</code>	the sigma link function with default <code>log</code>
<code>nu.link</code>	the nu link function with default <code>logit</code>
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of precision parameter values
<code>nu</code>	vector of parameter values modelling the probability at zero
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code> .
<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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required
<code>from</code>	where to start plotting the distribution from
<code>to</code>	up to where to plot the distribution

obj                    a fitted BEZI object  
 ...                    other graphical parameters for plotting

### Details

The zero-inflated beta distribution is given as

$$f(y) = \nu$$

if  $(y = 0)$

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

if  $y = (0, 1)$ . The parameters satisfy  $0 < \mu < 1$ ,  $\sigma > 0$  and  $0 < \nu < 1$ .

Here  $E(y) = (1 - \nu)\mu$  and  $Var(y) = (1 - \nu) \frac{\mu(1 - \mu)}{\sigma + 1} + \nu(1 - \nu)\mu^2$ .

### Value

returns a `gamlss.family` object which can be used to fit a zero-inflated beta distribution in the `gamlss()` function.

### Note

This work is part of my PhD project at the University of Sao Paulo under the supervision of Professor Silvia Ferrari. My thesis is concerned with regression modelling of rates and proportions with excess of zeros and/or ones

### Author(s)

Raydonal Ospina, Department of Statistics, University of Sao Paulo, Brazil.  
 <rospina@ime.usp.br>

### References

- Ferrari, S.L.P., Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, **31** (1), 799-815.
- Ospina R. and Ferrari S. L. P. (2010) Inflated beta distributions, *Statistical Papers*, **23**, 111-126.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape (with discussion). *Applied Statistics*, **54** (3), 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006). Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [BEZI](#)

**Examples**

```
BEZI()# gives information about the default links for the BEZI distribution
# plotting the distribution
plotBEZI( mu =0.5 , sigma=5, nu = 0.1, from = 0, to=0.99, n = 101)
# plotting the cdf
plot(function(y) pBEZI(y, mu=.5 ,sigma=5, nu=0.1), 0, 0.999)
# plotting the inverse cdf
plot(function(y) qBEZI(y, mu=.5 ,sigma=5, nu=0.1), 0, 0.999)
# generate random numbers
dat<-rBEZI(100, mu=.5, sigma=5, nu=0.1)
# fit a model to the data. Tits a constant for mu, sigma and nu
# library(gamlss)
#mod1<-gamlss(dat~1,sigma.formula=~1, nu.formula=~1, family=BEZI)
#fitted(mod1)[1]
#summary(mod1)
#fitted(mod1,"mu")[1]          #fitted mu
#fitted(mod1,"sigma")[1]      #fitted sigma
#fitted(mod1,"nu")[1]         #fitted nu
#meanBEZI(mod1)[1] # expected value of the response
```

---

BI

*Binomial distribution for fitting a GAMLSS*


---

**Description**

The `BI()` function defines the binomial distribution, a one parameter family distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dBI`, `pBI`, `qBI` and `rBI` define the density, distribution function, quantile function and random generation for the binomial, `BI()`, distribution.

**Usage**

```
BI(mu.link = "logit")
dBI(x, bd = 1, mu = 0.5, log = FALSE)
pBI(q, bd = 1, mu = 0.5, lower.tail = TRUE, log.p = FALSE)
qBI(p, bd = 1, mu = 0.5, lower.tail = TRUE, log.p = FALSE)
rBI(n, bd = 1, mu = 0.5)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "logit" link as the default for the <code>mu</code> parameter. Other links are "probit" and "cloglog"(complementary log-log)
<code>x</code>	vector of (non-negative integer) quantiles

<code>mu</code>	vector of positive probabilities
<code>bd</code>	vector of binomial denominators
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$

### Details

Definition file for binomial distribution.

$$f(y|\mu) = \frac{\Gamma(n+1)}{\Gamma(y+1)\Gamma(n-y+1)} \mu^y (1-\mu)^{(n-y)}$$

for  $y = 0, 1, 2, \dots, n$  and  $0 < \mu < 1$ .

### Value

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

### Note

The response variable should be a matrix containing two columns, the first with the count of successes and the second with the count of failures. The parameter `mu` represents a probability parameter with limits  $0 < \mu < 1$ .  $n\mu$  is the mean of the distribution where `n` is the binomial denominator.

### Author(s)

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [ZABI](#), [ZIBI](#)

## Examples

```

BI()# gives information about the default links for the Binomial distribution
# data(aep)
# library(gamlss)
# h<-gamlss(y~ward+loglos+year, family=BI, data=aep)
# plot of the binomial distribution
curve(dBI(x, mu = .5, bd=10), from=0, to=10, n=10+1, type="h")
tN <- table(Ni <- rBI(1000, mu=.2, bd=10))
r <- barplot(tN, col='lightblue')

```

---

 BNB

*Beta Negative Binomial distribution for fitting a GAMLSS*


---

## Description

The `BNB()` function defines the beta negative binomial distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`.

The functions `dBNB`, `pBNB`, `qBNB` and `rBNB` define the density, distribution function, quantile function and random generation for the beta negative binomial distribution, `BNB()`.

The functions `ZABNB()` and `ZIBNB()` are the zero adjusted (hurdle) and zero inflated versions of the beta negative binomial distribution, respectively. That is four parameter distributions.

The functions `dZABNB`, `dZIBNB`, `pZABNB`, `pZIBNB`, `qZABNB`, `qZIBNB`, `rZABNB` and `rZIBNB` define the probability, cumulative, quantile and random generation functions for the zero adjusted and zero inflated beta negative binomial distributions, `ZABNB()`, `ZIBNB()`, respectively.

## Usage

```

BNB(mu.link = "log", sigma.link = "log", nu.link = "log")
dBNB(x, mu = 1, sigma = 1, nu = 1, log = FALSE)
pBNB(q, mu = 1, sigma = 1, nu = 1, lower.tail = TRUE, log.p = FALSE)
qBNB(p, mu = 1, sigma = 1, nu = 1, lower.tail = TRUE, log.p = FALSE,
     max.value = 10000)
rBNB(n, mu = 1, sigma = 1, nu = 1, max.value = 10000)

ZABNB(mu.link = "log", sigma.link = "log", nu.link = "log",
      tau.link = "logit")
dZABNB(x, mu = 1, sigma = 1, nu = 1, tau = 0.1, log = FALSE)
pZABNB(q, mu = 1, sigma = 1, nu = 1, tau = 0.1, lower.tail = TRUE,
      log.p = FALSE)
qZABNB(p, mu = 1, sigma = 1, nu = 1, tau = 0.1, lower.tail = TRUE,
      log.p = FALSE, max.value = 10000)
rZABNB(n, mu = 1, sigma = 1, nu = 1, tau = 0.1, max.value = 10000)

ZIBNB(mu.link = "log", sigma.link = "log", nu.link = "log",
      tau.link = "logit")
dZIBNB(x, mu = 1, sigma = 1, nu = 1, tau = 0.1, log = FALSE)

```

```

pZIBNB(q, mu = 1, sigma = 1, nu = 1, tau = 0.1, lower.tail = TRUE,
       log.p = FALSE)
qZIBNB(p, mu = 1, sigma = 1, nu = 1, tau = 0.1, lower.tail = TRUE,
       log.p = FALSE, max.value = 10000)
rZIBNB(n, mu = 1, sigma = 1, nu = 1, tau = 0.1, max.value = 10000)

```

### Arguments

<code>mu.link</code>	The link function for mu
<code>sigma.link</code>	The link function for sigma
<code>nu.link</code>	The link function for nu
<code>tau.link</code>	The link function for tau
<code>x</code>	vector of (non-negative integer)
<code>mu</code>	vector of positive means
<code>sigma</code>	vector of positive dispersion parameter
<code>nu</code>	vector of a positive parameter
<code>tau</code>	vector of probabilities
<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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>max.value</code>	a constant, set to the default value of 10000 for how far the algorithm should look for q

### Details

The probability function of the BNB is

$$P(Y = y|\mu, \sigma, \nu) = \frac{\Gamma(y + \nu^{-1})}{\Gamma(y + 1)} \frac{B(y + \mu\sigma^{-1}\nu, \sigma^{-1} + \nu^{-1} + 1)}{\Gamma(\nu^{-1}) B(\mu\sigma^{-1}\nu, \sigma^{-1} + 1)}$$

for  $y = 0, 1, 2, 3, \dots$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

The distribution has mean  $\mu$ .

### Value

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

### Author(s)

Bob Rigby and Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>



## References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[NBI](#), [NBII](#)

## Examples

```
BNB() # gives information about the default links for the beta negative binomial
# plotting the distribution
plot(function(y) dBNB(y, mu = 10, sigma = 0.5, nu=2), from=0, to=40, n=40+1, type="h")
# creating random variables and plot them
tN <- table(Ni <- rBNB(1000, mu=5, sigma=0.5, nu=2))
r <- barplot(tN, col='lightblue')

ZABNB()
ZIBNB()
# plotting the distribution
plot(function(y) dZABNB(y, mu = 10, sigma = 0.5, nu=2, tau=.1),
      from=0, to=40, n=40+1, type="h")
plot(function(y) dZIBNB(y, mu = 10, sigma = 0.5, nu=2, tau=.1),
      from=0, to=40, n=40+1, type="h")
## Not run:
library(gamlss)
data(species)
species <- transform(species, x=log(lake))
m6 <- gamlss(fish~ pb(x), sigma.fo=~1, data=species, family=BNB)

## End(Not run)
```

---

checklink

*Set the Right Link Function for Specified Parameter and Distribution*

---

## Description

This function is used within the distribution family specification of a GAMLSS model to define the right link for each of the parameters of the distribution. This function should not be called by the user unless he/she specify a new distribution family or wishes to change existing link functions in the parameters.

**Usage**

```
checklink(which.link = NULL, which.dist = NULL, link = NULL, link.List = NULL)
```

**Arguments**

<code>which.link</code>	which parameter link e.g. <code>which.link="mu.link"</code>
<code>which.dist</code>	which distribution family e.g. <code>which.dist="Cole.Green"</code>
<code>link</code>	a repetition of <code>which.link</code> e.g. <code>link=substitute(mu.link)</code>
<code>link.List</code>	what link function are required e.g. <code>link.List=c("inverse", "log", "identity")</code>

**Value**

Defines the right link for each parameter

**Author(s)**

Calliope Akantziliotou

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#)

---

DBI

*The Double binomial distribution*

---

**Description**

The function `DBI()` defines the double binomial distribution, a two parameters distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dDBI`, `pDBI`, `qDBI` and `rDBI` define the density, distribution function, quantile function and random generation for the double binomial, `DBI()`, distribution. The function `GetBI_C` calculates numerically the constant of proportionality needed for the pdf to sum up to 1.

**Usage**

```

DBI(mu.link = "logit", sigma.link = "log")
dDBI(x, mu = 0.5, sigma = 1, bd = 2, log = FALSE)
pDBI(q, mu = 0.5, sigma = 1, bd = 2, lower.tail = TRUE,
      log.p = FALSE)
qDBI(p, mu = 0.5, sigma = 1, bd = 2, lower.tail = TRUE,
      log.p = FALSE)
rDBI(n, mu = 0.5, sigma = 1, bd = 2)
GetBI_C(mu, sigma, bd)

```

**Arguments**

<code>mu.link</code>	the link function for mu with default log
<code>sigma.link</code>	the link function for sigma with default log
<code>x, q</code>	vector of (non-negative integer) quantiles
<code>bd</code>	vector of binomial denominator
<code>p</code>	vector of probabilities
<code>mu</code>	the mu parameter
<code>sigma</code>	the sigma parameter
<code>lower.tail</code>	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
<code>log, log.p</code>	logical; if TRUE, probabilities p are given as log(p)
<code>n</code>	how many random values to generate

**Details**

The definition for the Double Poisson distribution first introduced by Efron (1986) is:

$$p_Y(y|n, \mu, \sigma) = \text{frac1}C(n, \mu, \sigma) \frac{\Gamma(n+1)}{\Gamma(y+1)\Gamma(n-y+1)} \frac{y^y (n-y)^{n-y}}{n^n} \frac{n^{n/\sigma} \mu^{y/\sigma} (1-\mu)^{(n-y)/\sigma}}{y^{y/\sigma} (n-y)^{(n-y)/\sigma}}$$

for  $y = 0, 1, 2, \dots, \infty$ ,  $\mu > 0$  and  $\sigma > 0$  where  $C$  is the constant of proportionality which is calculated numerically using the function `GetBI_C()`.

**Value**

The function `DBI` returns a `gamlss.family` object which can be used to fit a double binomial distribution in the `gamlss()` function.

**Author(s)**

Mikis Stasinopoulos, Bob Rigby, Marco Enea and Fernanda de Bastiani

## References

- Efron, B., 1986. Double exponential families and their use in generalized linear Regression. *Journal of the American Statistical Association* 81 (395), 709-721.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[BI, BB](#)

## Examples

```
DBI()
x <- 0:20
# underdispersed DBI
plot(x, dDBI(x, mu=.5, sigma=.2, bd=20), type="h", col="green", lwd=2)
# binomial
lines(x+0.1, dDBI(x, mu=.5, sigma=1, bd=20), type="h", col="black", lwd=2)
# overdispersed DBI
lines(x+.2, dDBI(x, mu=.5, sigma=2, bd=20), type="h", col="red", lwd=2)
```

---

DBURR12

*The Discrete Burr type XII distribution for fitting a GAMLSS model*

---

## Description

The `DBURR12()` function defines the discrete Burr type XII distribution, a three parameter discrete distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dDBURR12()`, `pDBURR12()`, `qDBURR12()` and `rDBURR12()` define the density, distribution function, quantile function and random generation for the discrete Burr type XII `DBURR12()`, distribution.

## Usage

```
DBURR12(mu.link = "log", sigma.link = "log", nu.link = "log")
dDBURR12(x, mu = 5, sigma = 2, nu = 2, log = FALSE)
pDBURR12(q, mu = 5, sigma = 2, nu = 2, lower.tail = TRUE,
         log.p = FALSE)
qDBURR12(p, mu = 5, sigma = 2, nu = 2, lower.tail = TRUE,
         log.p = FALSE)
rDBURR12(n, mu = 5, sigma = 2, nu = 2)
```

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

**Details**

The probability function of the discrete Burr XII distribution is given by

$$f(y|\mu, \sigma, \nu) = (1 + (y/\mu)^\sigma)^\nu - (1 + ((y+1)/\mu)^\sigma)^\nu$$

for  $y = 0, 1, 2, \dots, \infty$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu > 0$ .

Note that the above parametrization is different from Para and Jan (2016).

**Value**

The function `DBURR12()` Returns a `gamlss.family` object which can be used to fit a discrete Burr XII distribution in the `gamlss()` function.

**Note**

The parameters of the distributions are highly correlated so the argument of `gamlss method=mixed(10,100)` may have to be used.

The distribution can be under/over dispersed and also with long tails.

**Author(s)**

Rigby, R. A., Stasinopoulos D. M., Fernanda De Bastiani.

**References**

Para, B. A. and Jan, T. R. (2016). On discrete three parameter Burr type XII and discrete Lomax distributions and their applications to model count data from medical science. *Biometrics and Biostatistics International Journal*, **54**, part 3, pp 507-554.

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **4**, pp 1-15.

Stasinopoulos D. M., Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [DPO](#)

### Examples

```
DBURR12()#
#plot the pdf using plot
plot(function(y) dDBURR12(y, mu=10, sigma=1, nu=1), from=0, to=100, n=100+1, type="h") # pdf
# plot the cdf
plot(seq(from=0,to=100),pDBURR12(seq(from=0,to=100), mu=10, sigma=1, nu=1), type="h") # cdf
# generate random sample
tN <- table(Ni <- rDBURR12(100, mu=5, sigma=1, nu=1))
r <- barplot(tN, col='lightblue')
```

---

DEL

*The Delaporte distribution for fitting a GAMLSS model*

---

### Description

The `DEL()` function defines the Delaporte distribution, a three parameter discrete distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dDEL`, `pDEL`, `qDEL` and `rDEL` define the density, distribution function, quantile function and random generation for the Delaporte `DEL()`, distribution.

### Usage

```
DEL(mu.link = "log", sigma.link = "log", nu.link = "logit")
dDEL(x, mu=1, sigma=1, nu=0.5, log=FALSE)
pDEL(q, mu=1, sigma=1, nu=0.5, lower.tail = TRUE,
      log.p = FALSE)
qDEL(p, mu=1, sigma=1, nu=0.5, lower.tail = TRUE,
      log.p = FALSE, max.value = 10000)
rDEL(n, mu=1, sigma=1, nu=0.5, max.value = 10000)
```

### 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 parameter
<code>nu.link</code>	Defines the <code>nu.link</code> , with "logit" link as the default for the nu parameter
<code>x</code>	vector of (non-negative integer) quantiles

mu	vector of positive mu
sigma	vector of positive dispersion parameter
nu	vector of nu
p	vector of probabilities
q	vector of quantiles
n	number of random values to return
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]
max.value	a constant, set to the default value of 10000 for how far the algorithm should look for q

### Details

The probability function of the Delaporte distribution is given by

$$f(y|\mu, \sigma, \nu) = \frac{e^{-\mu\nu}}{\Gamma(1/\sigma)} [1 + \mu\sigma(1 - \nu)]^{-1/\sigma} S$$

where

$$S = \sum_{j=0}^y \binom{y}{j} \frac{\mu^y \nu^{y-j}}{y!} \left[ \mu + \frac{1}{\sigma(1 - \nu)} \right]^{-j} \Gamma\left(\frac{1}{\sigma} + j\right)$$

for  $y = 0, 1, 2, \dots, \infty$  where  $\mu > 0$ ,  $\sigma > 0$  and  $0 < \nu < 1$ . This distribution is a parametrization of the distribution given by Wimmer and Altmann (1999) p 515-516 where  $\alpha = \mu\nu$ ,  $k = 1/\sigma$  and  $\rho = [1 + \mu\sigma(1 - \nu)]^{-1}$

### Value

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

### Note

The mean of  $Y$  is given by  $E(Y) = \mu$  and the variance by  $V(Y) = \mu + \mu^2\sigma(1 - \nu)^2$ .

### Author(s)

Rigby, R. A., Stasinopoulos D. M. and Marco Enea

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Rigby, R. A., Stasinopoulos D. M. and Akantziliotou, C. (2006) Modelling the parameters of a family of mixed Poisson distributions including the Sichel and Delaportte. Submitted for publication.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2003) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

Wimmer, G. and Altmann, G (1999). *Thesaurus of univariate discrete probability distributions* . Stamm Verlag, Essen, Germany

### See Also

[gamlss.family](#), [SI](#), [SICHEL](#)

### Examples

```
DEL()# gives information about the default links for the Delaporte distribution
#plot the pdf using plot
plot(function(y) dDEL(y, mu=10, sigma=1, nu=.5), from=0, to=100, n=100+1, type="h") # pdf
# plot the cdf
plot(seq(from=0,to=100),pDEL(seq(from=0,to=100), mu=10, sigma=1, nu=0.5), type="h") # cdf
# generate random sample
tN <- table(Ni <- rDEL(100, mu=10, sigma=1, nu=0.5))
r <- barplot(tN, col='lightblue')
# fit a model to the data
# library(gamlss)
# gamlss(Ni~1,family=DEL, control=gamlss.control(n.cyc=50))
```

---

DPO

*The Double Poisson distribution*

---

### Description

The function `DPO()` defines the double Poisson distribution, a two parameters distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dDPO`, `pDPO`, `qDPO` and `rDPO` define the density, distribution function, quantile function and random generation for the double Poisson, `DPO()`, distribution. The function `get_C()` calculates numerically the constant of proportionality needed for the pdf to sum up to 1.

### Usage

```
DPO(mu.link = "log", sigma.link = "log")
dDPO(x, mu = 1, sigma = 1, log = FALSE)
pDPO(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qDPO(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE,
     max.value = 10000)
rDPO(n, mu = 1, sigma = 1, max.value = 10000)
get_C(x, mu, sigma)
```



**Arguments**

<code>mu.link</code>	the link function for mu with default log
<code>sigma.link</code>	the link function for sigma with default log
<code>x, q</code>	vector of (non-negative integer) quantiles
<code>p</code>	vector of probabilities
<code>mu</code>	the mu parameter
<code>sigma</code>	the sigma parameter
<code>lower.tail</code>	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
<code>log, log.p</code>	logical; if TRUE, probabilities p are given as log(p)
<code>max.value</code>	a constant, set to the default value of 10000 for how far the algorithm should look for q
<code>n</code>	how many random values to generate

**Details**

The definition for the Double Poisson distribution first introduced by Efron (1986) is:

$$f(y|\mu, \sigma) = \left(\frac{1}{\sigma}\right)^{1/2} e^{-\mu/\sigma} \left(\frac{e^{-y} y^y}{y!}\right) \left(\frac{e\mu}{y}\right)^{y/\sigma} C$$

for  $y = 0, 1, 2, \dots, \infty$ ,  $\mu > 0$  and  $\sigma > 0$  where  $C$  is the constant of proportionality which is calculated numerically using the function `get_C`.

**Value**

The function `DPO` returns a `gamlss.family` object which can be used to fit a double Poisson distribution in the `gamlss()` function.

**Note**

The distributons calculates the constant of proportionality numerically therefore it can be slow for large data

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Marco Enea

**References**

- Efron, B., 1986. Double exponential families and their use in generalized linear Regression. *Journal of the American Statistical Association* 81 (395), 709-721.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

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

```

DPO()
# overdispersed DPO
x <- 0:20
plot(x, dDPO(x, mu=5, sigma=3), type="h", col="red")
# underdispersed DPO
plot(x, dDPO(x, mu=5, sigma=.3), type="h", col="red")
# generate random sample
Y <- rDPO(100,5,.5)
plot(table(Y))
points(0:20, 100*dDPO(0:20, mu=5, sigma=.5)+0.2, col="red")
# fit a model to the data
# library(gamlss)
# gamlss(Y~1,family=DPO)

```

EGB2

*The exponential generalized Beta type 2 distribution for fitting a GAMLSS*

**Description**

This function defines the generalized t distribution, a four parameter distribution. The response variable is in the range from minus infinity to plus infinity. The functions `dEGB2`, `pEGB2`, `qEGB2` and `rEGB2` define the density, distribution function, quantile function and random generation for the generalized beta type 2 distribution.

**Usage**

```

EGB2(mu.link = "identity", sigma.link = "log", nu.link = "log",
      tau.link = "log")
dEGB2(x, mu = 0, sigma = 1, nu = 1, tau = 0.5, log = FALSE)
pEGB2(q, mu = 0, sigma = 1, nu = 1, tau = 0.5, lower.tail = TRUE,
      log.p = FALSE)
qEGB2(p, mu = 0, sigma = 1, nu = 1, tau = 0.5, lower.tail = TRUE,
      log.p = FALSE)
rEGB2(n, mu = 0, sigma = 1, nu = 1, tau = 0.5)

```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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.
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the <code>tau</code> parameter.

x, q	vector of quantiles
mu	vector of location parameter values
sigma	vector of scale parameter values
nu	vector of skewness nu parameter values
tau	vector of kurtosis tau parameter values
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required

### Details

The probability density function of the Generalized Beta type 2, (GB2), is defined as

$$f(y|\mu, \sigma, \nu, \tau) = e^{\nu z} \{|\sigma| B(\nu, \tau) [1 + e^z]^{\nu+\tau}\}^{-1}$$

for  $-\infty < y < \infty$ , where  $z = (y - \mu)/\sigma$  and  $-\infty < \mu < \infty$ ,  $-\infty < \sigma < \infty$ ,  $\nu > 0$  and  $\tau > 0$ , McDonald and Xu (1995).

### Value

EGB2() returns a `gamlss.family` object which can be used to fit the EGB2 distribution in the `gamlss()` function. `dEGB2()` gives the density, `pEGB2()` gives the distribution function, `qEGB2()` gives the quantile function, and `rEGB2()` generates random deviates.

### Author(s)

Bob Rigby and Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [JSU](#), [BCT](#)

## Examples

```
EGB2() #
y<- rEGB2(200, mu=5, sigma=2, nu=1, tau=4)
library(MASS)
truehist(y)
fx<-dEGB2(seq(min(y), 20, length=200), mu=5, sigma=2, nu=1, tau=4)
lines(seq(min(y),20,length=200),fx)
# something funny here
# library(gamlss)
# histDist(y, family=EGB2, n.cyc=60)
integrate(function(x) x*dEGB2(x=x, mu=5, sigma=2, nu=1, tau=4), -Inf, Inf)
curve(dEGB2(x, mu=5, sigma=2, nu=1, tau=4), -10, 10, main = "The EGB2 density
      mu=5, sigma=2, nu=1, tau=4")
```

---

exGAUS

*The ex-Gaussian distribution*

---

## Description

The ex-Gaussian distribution is often used by psychologists to model response time (RT). It is defined by adding two random variables, one from a normal distribution and the other from an exponential. The parameters  $\mu$  and  $\sigma$  are the mean and standard deviation from the normal distribution variable while the parameter  $\nu$  is the mean of the exponential variable. The functions `dexGAUS`, `pexGAUS`, `qexGAUS` and `rexGAUS` define the density, distribution function, quantile function and random generation for the ex-Gaussian distribution.

## Usage

```
exGAUS(mu.link = "identity", sigma.link = "log", nu.link = "log")
dexGAUS(x, mu = 5, sigma = 1, nu = 1, log = FALSE)
pexGAUS(q, mu = 5, sigma = 1, nu = 1, lower.tail = TRUE, log.p = FALSE)
qexGAUS(p, mu = 5, sigma = 1, nu = 1, lower.tail = TRUE, log.p = FALSE)
rexGAUS(n, mu = 5, sigma = 1, nu = 1, ...)
```

## Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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. Other links are "inverse", "identity", "logshifted" (shifted from one) and "own"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of <code>mu</code> parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of <code>nu</code> parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code> .

lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required
...	for extra arguments

### Details

The probability density function of the ex-Gaussian distribution, (exGAUS), is defined as

$$f(y|\mu, \sigma, \nu) = \frac{1}{\nu} e^{\frac{\mu-y}{\nu} + \frac{\sigma^2}{2\nu^2}} \Phi\left(\frac{y-\mu}{\sigma} - \frac{\sigma}{\nu}\right)$$

where  $\Phi$  is the cdf of the standard normal distribution, for  $-\infty < y < \infty$ ,  $-\infty < \mu < \infty$ ,  $\sigma > 0$  and  $\nu > 0$ .

### Value

exGAUS() returns a `gamlss.family` object which can be used to fit ex-Gaussian distribution in the `gamlss()` function. `dexGAUS()` gives the density, `pexGAUS()` gives the distribution function, `qexGAUS()` gives the quantile function, and `rexGAUS()` generates random deviates.

### Note

The mean of the ex-Gaussian is  $\mu + \nu$  and the variance is  $\sigma^2 + \nu^2$ .

### Author(s)

Mikis Stasinopoulos and Bob Rigby

### References

- Cousineau, D. Brown, S. and Heathcote A. (2004) Fitting distributions using maximum likelihood: Methods and packages, *Behavior Research Methods, Instruments and Computers*, **46**, 742-756.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [BCCG](#), [GA](#), [IG LNO](#)

## Examples

```

exGAUS() #
y<- rexGAUS(100, mu=300, nu=100, sigma=35)
hist(y)
# library(gamlss)
# m1<-gamlss(y~1, family=exGAUS)
# plot(m1)
curve(dexGAUS(x, mu=300 ,sigma=35,nu=100), 100, 600,
      main = "The ex-GAUS density mu=300 ,sigma=35,nu=100")
plot(function(x) pexGAUS(x, mu=300,sigma=35,nu=100), 100, 600,
      main = "The ex-GAUS cdf mu=300, sigma=35, nu=100")

```

---

 EXP

*Exponential distribution for fitting a GAMLSS*


---

## Description

The function EXP defines the exponential distribution, a one parameter distribution for a `gamlss` family object to be used in GAMLSS fitting using the function `gamlss()`. The `mu` parameter represents the mean of the distribution. The functions `dEXP`, `pEXP`, `qEXP` and `rEXP` define the density, distribution function, quantile function and random generation for the specific parameterization of the exponential distribution defined by function EXP.

## Usage

```

EXP(mu.link = "log")
dEXP(x, mu = 1, log = FALSE)
pEXP(q, mu = 1, lower.tail = TRUE, log.p = FALSE)
qEXP(p, mu = 1, lower.tail = TRUE, log.p = FALSE)
rEXP(n, mu = 1)

```

## Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter, other links are "inverse" and "identity"
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

The specific parameterization of the exponential distribution used in EXP is

$$f(y|\mu) = \frac{1}{\mu} \exp \left\{ -\frac{y}{\mu} \right\}$$

, for  $y > 0, \mu > 0$ .

**Value**

EXP() returns a `gamlss.family` object which can be used to fit an exponential distribution in the `gamlss()` function. `dEXP()` gives the density, `pEXP()` gives the distribution function, `qEXP()` gives the quantile function, and `rEXP()` generates random deviates.

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby and Nicoleta Motpan

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#)

**Examples**

```
y<-rEXP(1000,mu=1) # generates 1000 random observations
hist(y)
# library(gamlss)
# histDist(y, family=EXP)
```

flexDist

*Non-parametric pdf from limited information data***Description**

This is an attempt to create a distribution function if the only existing information is the quantiles or expectiles of the distribution.

**Usage**

```
flexDist(quantiles = list(values=c(-1.96,0,1.96), prob=c(0.05, .50, 0.95)),
        expectiles = list(), lambda = 10,
        kappa = 10, delta = 1e-07, order = 3, n.iter = 200,
        plot = TRUE, no.inter = 100, lower = NULL,
        upper = NULL, perc.quant = 0.3, ...)
```

**Arguments**

quantiles	a list with components values and prob
expectiles	a list with components values and prob
lambda	smoothing parameter for the log-pdf
kappa	smoothing parameter for log concavity
delta	smoothing parameter for ridge penalty
order	the order of the penalty for log-pdf
n.iter	maximum number of iterations
plot	whether to plot the result
no.inter	How many discrete probabilities to evaluate
lower	the lower value of the x
upper	the upper value of the x
perc.quant	how far from the quantile should go out to define the limit of x if not set by lower or upper
...	additional arguments

**Value**

Returns a list with components

pdf	the heights of the fitted pdf, the sum of it multiplied by the Dx should add up to 1 i.e. $\text{sum}(\text{object}\$pdf * \text{diff}(\text{object}\$x)[1])$
cdf	the fitted cdf
x	the values of x where the discretise distribution is defined
pFun	the cdf of the fitted non-parametric distribution
qFun	the inverse cdf function of the fitted non-parametric distribution
rFun	a function to generate a random sample from the fitted non-parametric distribution



**Author(s)**

Mikis Stasinopoulos, Paul Eilers, Bob Rigby and Vlasios Voudouris

**References**

Eilers, P. H. C., Voudouris, V., Rigby R. A., Stasinopoulos D. M. (2012) Estimation of nonparametric density from sparse summary information, under review.

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), Appl. Statist., 54, part 3, pp 507-554.

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. Journal of Statistical Software, Vol. 23, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[histSmo](#)

**Examples**

```
# Normal
r1<-flexDist(quantiles=list(values=qN0(c(0.05, 0.25, 0.5,0.75, 0.95), mu=0,
      sigma=1), prob=c( 0.05, 0.25, 0.5,0.75,0.95 )),
      no.inter=200, lambda=10, kappa=10, perc.quant=0.3)

# GAMMA
r1<-flexDist(quantiles=list(values=qGA(c(0.05,0.25, 0.5,0.75,0.95), mu=1,
      sigma=.8), prob=c(0.05,0.25, 0.5,0.75,0.95)),
      exptiles=list(values=1, prob=0.5), lambda=10,
      kappa=10, lower=0, upper=5)#
```

---

 GA

*Gamma distribution for fitting a GAMLSS*


---

**Description**

The function GA defines the gamma distribution, a two parameter distribution, for a `gamlss` family object to be used in GAMLSS fitting using the function `gamlss()`. The parameterization used has the mean of the distribution equal to  $\mu$  and the variance equal to  $\sigma^2\mu^2$ . The functions `dGA`, `pGA`, `qGA` and `rGA` define the density, distribution function, quantile function and random generation for the specific parameterization of the gamma distribution defined by function GA.

**Usage**

```
GA(mu.link = "log", sigma.link = "log")
dGA(x, mu = 1, sigma = 1, log = FALSE)
pGA(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qGA(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rGA(n, mu = 1, sigma = 1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter, other links are "inverse", "identity" and "own"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter, other link is the "inverse", "identity" and "own"
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

The specific parameterization of the gamma distribution used in GA is

$$f(y|\mu, \sigma) = \frac{y^{(1/\sigma^2)-1} \exp[-y/(\sigma^2\mu)]}{(\sigma^2\mu)^{(1/\sigma^2)}\Gamma(1/\sigma^2)}$$

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

**Value**

`GA()` returns a `gamlss.family` object which can be used to fit a gamma distribution in the `gamlss()` function. `dGA()` gives the density, `pGA()` gives the distribution function, `qGA()` gives the quantile function, and `rGA()` generates random deviates. The latest functions are based on the equivalent R functions for gamma distribution.

**Note**

$\mu$  is the mean of the distribution in GA. In the function GA,  $\sigma$  is the square root of the usual dispersion parameter for a GLM gamma model. Hence  $\sigma\mu$  is the standard deviation of the distribution defined in GA.

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby and Calliope Akantziliotou

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziotiou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#)

### Examples

```
GA()# gives information about the default links for the gamma distribution
# dat<-rgamma(100, shape=1, scale=10) # generates 100 random observations
# fit a gamlss model
# gamlss(dat~1,family=GA)
# fits a constant for each parameter mu and sigma of the gamma distribution
newdata<-rGA(1000,mu=1,sigma=1) # generates 1000 random observations
hist(newdata)
rm(dat,newdata)
```

---

gamlss.family

*Family Objects for fitting a GAMLSS model*

---

### Description

GAMLSS families are the current available distributions that can be fitted using the `gamlss()` function.

### Usage

```
gamlss.family(object,...)
as.gamlss.family(object)
as.family(object)
## S3 method for class 'gamlss.family'
print(x,...)
```

### Arguments

object	a gamlss family object e.g. BCT
x	a gamlss family object e.g. BCT
...	further arguments passed to or from other methods.

## Details

There are several distributions available for the response variable in the `gamlss` function. The following table display their names and their abbreviations in R. Note that the different distributions can be fitted using their R abbreviations (and optionally excluding the brackets) i.e. `family=BI()`, `family=BI` are equivalent.

Distributions	R names	No of parameters
Beta	<code>BE()</code>	2
Beta Binomial	<code>BB()</code>	2
Beta negative binomial	<code>BNB()</code>	3
Beta one inflated	<code>BEOI()</code>	3
Beta zero inflated	<code>BEZI()</code>	3
Beta inflated	<code>BEINF()</code>	4
Binomial	<code>BI()</code>	1
Box-Cox Cole and Green	<code>BCCG()</code>	3
Box-Cox Power Exponential	<code>BCPE()</code>	4
Box-Cox-t	<code>BCT()</code>	4
Delaport	<code>DEL()</code>	3
Discrete Burr XII	<code>DBURR12()</code>	3
Double Poisson	<code>DPO()</code>	2
Double binomial	<code>DBI()</code>	2
Exponential	<code>EXP()</code>	1
Exponential Gaussian	<code>exGAUS()</code>	3
Exponential generalized Beta type 2	<code>EGB2()</code>	4
Gamma	<code>GA()</code>	2
Generalized Beta type 1	<code>GB1()</code>	4
Generalized Beta type 2	<code>GB2()</code>	4
Generalized Gamma	<code>GG()</code>	3
Generalized Inverse Gaussian	<code>GIG()</code>	3
Generalized t	<code>GT()</code>	4
Geometric	<code>GEOM()</code>	1
Geometric (original)	<code>GEOMo()</code>	1
Gumbel	<code>GU()</code>	2
Inverse Gamma	<code>IGAMMA()</code>	2
Inverse Gaussian	<code>IG()</code>	2
Johnson's SU	<code>JSU()</code>	4
Logarithmic	<code>LG()</code>	1
Logistic	<code>LO()</code>	2
Logit-Normal	<code>LOGITNO()</code>	2
log-Normal	<code>LOGNO()</code>	2
log-Normal (Box-Cox)	<code>LNO()</code>	3 (1 fixed)
Negative Binomial type I	<code>NBI()</code>	2
Negative Binomial type II	<code>NBII()</code>	2
Negative Binomial family	<code>NBF()</code>	3
Normal Exponential $t$	<code>NET()</code>	4 (2 fixed)
Normal	<code>NO()</code>	2
Normal Family	<code>NOF()</code>	3 (1 fixed)
Normal Linear Quadratic	<code>LQNO()</code>	2

Pareto type 2	PARETO2()	2
Pareto type 2 original	PARETO2o()	2
Power Exponential	PE()	3
Power Exponential type 2	PE2()	3
Poisson	PO()	1
Poisson inverse Gaussian	PIG()	2
Reverse generalized extreme	RGE()	3
Reverse Gumbel	RG()	2
Skew Power Exponential type 1	SEP1()	4
Skew Power Exponential type 2	SEP2()	4
Skew Power Exponential type 3	SEP3()	4
Skew Power Exponential type 4	SEP4()	4
Shash	SHASH()	4
Shash original	SHASHo()	4
Shash original 2	SHASH()	4
Sichel (original)	SI()	3
Sichel (mu as the maen)	SICHEL()	3
Simplex	SIMPLEX()	2
Skew t type 1	ST1()	3
Skew t type 2	ST2()	3
Skew t type 3	ST3()	3
Skew t type 4	ST4()	3
Skew t type 5	ST5()	3
t-distribution	TF()	3
Waring	WARING()	1
Weibull	WEI()	2
Weibull(PH parameterization)	WEI2()	2
Weibull (mu as mean)	WEI3()	2
Yule	YULE()	1
Zero adjusted binomial	ZABI()	2
Zero adjusted beta neg. bin.	ZABNB()	4
Zero adjusted IG	ZAIG()	2
Zero adjusted logarithmic	ZALG()	2
Zero adjusted neg. bin.	ZANBI()	3
Zero adjusted poisson	ZAP()	2
Zero adjusted Sichel	ZASICHEL()	4
Zero adjusted Zipf	ZAZIPF()	2
Zero inflated binomial	ZIBI()	2
Zero inflated beta neg. bin.	ZIBNB()	4
Zero inflated neg. bin.	ZINBI()	3
Zero inflated poisson	ZIP()	2
Zero inf. poiss.(mu as mean)	ZIP2()	2
Zero inflated PIG	ZIPIG()	3
Zero inflated Sichel	ZISICHEL()	4
Zipf	ZIPF()	1

Note that some of the distributions are in the package `gamlss.dist`. The parameters of the distribu-

tions are in order,  $\mu$  for location,  $\sigma$  for scale (or dispersion), and  $\nu$  and  $\tau$  for shape. More specifically for the BCCG family  $\mu$  is the median,  $\sigma$  approximately the coefficient of variation, and  $\nu$  the skewness parameter. The parameters for BCPE distribution have the same interpretation with the extra fourth parameter  $\tau$  modelling the kurtosis of the distribution. The parameters for BCT have the same interpretation except that  $\sigma[(\tau/(\tau - 2))^{0.5}]$  is approximately the coefficient of variation.

All of the distribution in the above list are also provided with the corresponding d, p, q and r functions for density (pdf), distribution function (cdf), quantile function and random generation function respectively, (see individual distribution for details).

### Value

The above GAMLSS families return an object which is of type `gamlss.family`. This object is used to define the family in the `gamlss()` fit.

### Note

More distributions will be documented in later GAMLSS releases. Further user defined distributions can be incorporate relatively easy, see, for example, the help documentation accompanying the `gamlss` library.

### Author(s)

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby and Calliope Akantziliotou

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[BE](#),[BB](#),[BEINF](#),[BI](#),[LNO](#),[BCT](#), [BCPE](#),[BCCG](#), [GA](#),[GU](#),[JSU](#),[IG](#),[LO](#), [NBI](#),[NBII](#),[NO](#),[PE](#),[PO](#), [RG](#),[PIG](#),[TF](#),[WEI](#),[WEI2](#), [ZIP](#)

### Examples

```
normal<-N0(mu.link="log", sigma.link="log")
normal
```

**Description**

This function defines the generalized beta type 1 distribution, a four parameter distribution. The function GB1 creates a `gamlss.family` object which can be used to fit the distribution using the function `gamlss()`. Note the range of the response variable is from zero to one. The functions `dGB1`, `GB1`, `qGB1` and `rGB1` define the density, distribution function, quantile function and random generation for the generalized beta type 1 distribution.

**Usage**

```
GB1(mu.link = "logit", sigma.link = "logit", nu.link = "log",
    tau.link = "log")
dGB1(x, mu = 0.5, sigma = 0.4, nu = 1, tau = 1, log = FALSE)
pGB1(q, mu = 0.5, sigma = 0.4, nu = 1, tau = 1, lower.tail = TRUE,
    log.p = FALSE)
qGB1(p, mu = 0.5, sigma = 0.4, nu = 1, tau = 1, lower.tail = TRUE,
    log.p = FALSE)
rGB1(n, mu = 0.5, sigma = 0.4, nu = 1, tau = 1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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.
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the <code>tau</code> parameter.
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of skewness <code>nu</code> parameter values
<code>tau</code>	vector of kurtosis <code>tau</code> parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

### Details

The probability density function of the Generalized Beta type 1, (GB1), is defined as

$$f(y|\mu, \sigma, \nu, \tau) = \frac{\tau \nu^\beta y^{\tau\alpha-1} (1-y^\tau)^{\beta-1}}{B(\alpha, \beta) [\nu + (1-\nu)y^\tau]^{\alpha+\beta}}$$

where  $0 < y < 1$ ,  $\alpha = \mu(1 - \sigma^2)/\sigma^2$  and  $\beta = (1 - \mu)(1 - \sigma^2)/\sigma^2$ , and  $\alpha > 0$ ,  $\beta > 0$ . Note the  $\mu = \alpha/(\alpha + \beta)$ ,  $\sigma = (\alpha + \beta + 1)^{-1/2}$ .

### Value

GB1() returns a `gamlss.family` object which can be used to fit the GB1 distribution in the `gamlss()` function. `dGB1()` gives the density, `pGB1()` gives the distribution function, `qGB1()` gives the quantile function, and `rGB1()` generates random deviates.

### Warning

The `qSHASH` and `rSHASH` are slow since they are relying on golden section for finding the quantiles

### Author(s)

Bob Rigby and Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [JSU](#), [BCT](#)

### Examples

```
GB1() #
y<- rGB1(200, mu=.1, sigma=.6, nu=1, tau=4)
hist(y)
# library(gamlss)
# histDist(y, family=GB1, n.cyc=60)
```



```
curve(dGB1(x, mu=.1 ,sigma=.6, nu=1, tau=4), 0.01, 0.99, main = "The GB1
      density mu=0.1, sigma=.6, nu=1, tau=4")
```

---

GB2 *The generalized Beta type 2 and generalized Pareto distributions for fitting a GAMLSS*

---

## Description

This function defines the generalized beta type 2 distribution, a four parameter distribution. The function GB2 creates a `gamlss.family` object which can be used to fit the distribution using the function `gamlss()`. The response variable is in the range from zero to infinity. The functions `dGB2`, `GB2`, `qGB2` and `rGB2` define the density, distribution function, quantile function and random generation for the generalized beta type 2 distribution. The generalised Pareto GP distribution is defined by setting the parameters `sigma` and `nu` of the GB2 distribution to 1.

## Usage

```
GB2(mu.link = "log", sigma.link = "log", nu.link = "log",
     tau.link = "log")
dGB2(x, mu = 1, sigma = 1, nu = 1, tau = 0.5, log = FALSE)
pGB2(q, mu = 1, sigma = 1, nu = 1, tau = 0.5, lower.tail = TRUE,
     log.p = FALSE)
qGB2(p, mu = 1, sigma = 1, nu = 1, tau = 0.5, lower.tail = TRUE,
     log.p = FALSE)
rGB2(n, mu = 1, sigma = 1, nu = 1, tau = 0.5)

GP(mu.link = "log", sigma.link = "log")
dGP(x, mu = 1, sigma = 1, log = FALSE)
pGP(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qGP(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rGP(n, mu = 1, sigma = 1)
```

## Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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.
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the <code>tau</code> parameter.
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of skewness <code>nu</code> parameter values

<code>tau</code>	vector of kurtosis tau parameter values
<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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

### Details

The probability density function of the Generalized Beta type 2, (GB2), is defined as

$$f(y|\mu, \sigma, \nu, \tau) = |\sigma| y^{\sigma\nu-1} \{ \mu^{\sigma\nu} B(\nu, \tau) [1 + (y/\mu)^\sigma]^{\nu+\tau} \}^{-1}$$

where  $y > 0$ ,  $\mu > 0$ ,  $-\infty < \sigma < \infty$ ,  $\nu > 0$  and  $\tau > 0$ .

### Value

`GB2()` returns a `gamlss.family` object which can be used to fit the GB2 distribution in the `gamlss()` function. `dGB2()` gives the density, `pGB2()` gives the distribution function, `qGB2()` gives the quantile function, and `rGB2()` generates random deviates.

### Warning

The `qSHASH` and `rSHASH` are slow since they are relying on golden section for finding the quantiles

### Author(s)

Bob Rigby and Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape, (with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [JSU](#), [BCT](#)

**Examples**

```

GB2() #
y<- rGB2(200, mu=5, sigma=2, nu=1, tau=1)
library(MASS)
truehist(y)
fx<-dGB2(seq(0.01, 20, length=200), mu=5 ,sigma=2, nu=1, tau=1)
lines(seq(0.01,20,length=200),fx)
integrate(function(x) x*dGB2(x=x, mu=5, sigma=2, nu=1, tau=1), 0, Inf)
mean(y)
curve(dGB2(x, mu=5 ,sigma=2, nu=1, tau=1), 0.01, 20,
      main = "The GB2 density mu=5, sigma=2, nu=1, tau=4")

```

gen.Family

*Functions to generate log and logit distributions from existing continuous gamlss.family distributions*

**Description**

There are five functions here. Only the functions Family and gen.Family should be used (see details).

**Usage**

```

Family.d(family = "NO", type = c("log", "logit"), ...)
Family.p(family = "NO", type = c("log", "logit"), ...)
Family.q(family = "NO", type = c("log", "logit"), ...)
Family.r(family = "NO", type = c("log", "logit"), ...)
Family(family = "NO", type = c("log", "logit"), local = TRUE, ...)
gen.Family(family = "NO", type = c("log", "logit"), ...)

```

**Arguments**

family	a continuous gamlss.family distribution
type	the type of transformation only "log" and "logit" are allowed
local	It is TRUE if is called within gamlss() otherwise is FALSE
...	for passing extra arguments

**Details**

The function gen.Family creates the standard d,p,q,r functions for the distribution plus the fitting gamlss.family. For example gen.Family("NO", "logit") will generate the functions dlogitNO(), plogitNO(), qlongitNO(), rlogitNO() and dlogitNO(). The latest function can be used in family argument of gamlss() to fit a logic-Normal distribution i.e. family=logitNO. The same fitting can be achieved by using family=Family("NO", "logit"). Here the required dlogitNO(), plogitNO() and logitNO() functions are generated locally within the gamlss() environment.

**Value**

The function `gen.Family` returns the `d`, `p`, `q` `r` functions plus the fitting function.

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)> and Bob Rigby

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**Examples**

```
# generating a log t distribution
gen.Family("TF")
# plotting the d, p, q, and r functions
op<-par(mfrow=c(2,2))
curve(dlogTF(x, mu=0), 0, 10)
curve(plogTF(x, mu=0), 0, 10)
curve(qlogTF(x, mu=0), 0, 1)
Y<- rlogTF(200)
hist(Y)
par(op)

# different mu
curve(dlogTF(x, mu=-1, sigma=1, nu=10), 0, 5, ylim=c(0,1))
curve(dlogTF(x, mu=0, sigma=1, nu=10), 0, 5, add=TRUE, col="red", lty=2)
curve(dlogTF(x, mu=1, sigma=1, nu=10), 0, 5, add=TRUE, col="blue", lty=3)

# different sigma
curve(dlogTF(x, mu=0, sigma=.5, nu=10), 0, 5, ylim=c(0,1))
curve(dlogTF(x, mu=0, sigma=1, nu=10), 0, 5, add=TRUE, col="red", lty=2)
curve(dlogTF(x, mu=0, sigma=2, nu=10), 0, 5, add=TRUE, col="blue", lty=3)

# different degrees of freedom nu
curve(dlogTF(x, mu=0, sigma=1, nu=1), 0, 5, ylim=c(0,.8), n = 1001)
curve(dlogTF(x, mu=0, sigma=1, nu=2), 0, 5, add=TRUE, col="red", lty=2)
curve(dlogTF(x, mu=0, sigma=1, nu=5), 0, 5, add=TRUE, col="blue", lty=3)
```

```

# generating a logit t distribution
gen.Family("TF", "logit")
# plotting the d, p, q, and r functions
op<-par(mfrow=c(2,2))
curve(dlogitTF(x, mu=0), 0, 1)
curve(plogitTF(x, mu=0), 0, 1)
curve(qlogitTF(x, mu=0), 0, 1)
abline(v=1)
Y<- rlogitTF(200)
hist(Y)
par(op)

# different mu
curve(dlogitTF(x, mu=-2, sigma=1, nu=10), 0, 1, ylim=c(0,5))
curve(dlogitTF(x, mu=0, sigma=1, nu=10), 0, 1, add=TRUE, col="red", lty=2)
curve(dlogitTF(x, mu=2, sigma=1, nu=10), 0, 1, add=TRUE, col="blue", lty=3)

# different sigma
curve(dlogitTF(x, mu=0, sigma=1, nu=10), 0, 1, ylim=c(0,2.5))
curve(dlogitTF(x, mu=0, sigma=2, nu=10), 0, 1, add=TRUE, col="red", lty=2)
curve(dlogitTF(x, mu=0, sigma=.7, nu=10), 0, 1, add=TRUE, col="blue", lty=3)

# different degrees of freedom nu
curve(dlogitTF(x, mu=0, sigma=1, nu=1), 0, 1, ylim=c(0,1.6))
curve(dlogitTF(x, mu=0, sigma=1, nu=2), 0, 1, add=TRUE, col="red", lty=2)
curve(dlogitTF(x, mu=0, sigma=1, nu=5), 0, 1, add=TRUE, col="blue", lty=3)

```

---

 GEOM

*Geometric distribution for fitting a GAMLSS model*


---

## Description

The functions `GEOMo()` and `GEOM()` define two parametrizations of the geometric distribution. The geometric distribution is a one parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The mean of `GEOM()` is equal to the parameter `mu`. The functions `dGEOM`, `pGEOM`, `qGEOM` and `rGEOM` define the density, distribution function, quantile function and random generation for the GEOM parameterization of the Geometric distribution.

## Usage

```

GEOM(mu.link = "log")
dGEOM(x, mu = 2, log = FALSE)
pGEOM(q, mu = 2, lower.tail = TRUE, log.p = FALSE)
qGEOM(p, mu = 2, lower.tail = TRUE, log.p = FALSE)
rGEOM(n, mu = 2)
GEOMo(mu.link = "logit")
dGEOMo(x, mu = 0.5, log = FALSE)

```

```
pGEMO(q, mu = 0.5, lower.tail = TRUE, log.p = FALSE)
qGEMO(p, mu = 0.5, lower.tail = TRUE, log.p = FALSE)
rGEMO(n, mu = 0.5)
```

### Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with <code>log</code> link as the default for the <code>mu</code> parameter
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code>
<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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

### Details

The parameterization of the original geometric distribution in the function `GE` is

$$f(y|\mu) = (1 - \mu)^y \mu$$

for  $y \geq 0$  and  $\mu > 0$ .

The parameterization of the geometric distribution in the function `GEOM` is

$$f(y|\mu) = \mu^y / (\mu + 1)^{y+1}$$

where for  $y \geq 0$  and  $\mu > 0$ .

### Value

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

### Author(s)

Fiona McElduff, Bob Rigby and Mikis Stasinopoulos.

### References

- Johnson, N. L., Kemp, A. W., and Kotz, S. (2005). *Univariate discrete distributions*. Wiley.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, 54, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. 23, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**[gamlss.family](#)**Examples**

```

par(mfrow=c(2,2))
y<-seq(0,20,1)
plot(y, dGEOM(y), type="h")
q <- seq(0, 20, 1)
plot(q, pGEOM(q), type="h")
p<-seq(0.0001,0.999,0.05)
plot(p , qGEOM(p), type="s")
dat <- rGEOM(100)
hist(dat)
#summary(gamlss(dat~1, family=GEOM))
par(mfrow=c(2,2))
y<-seq(0,20,1)
plot(y, dGEOMo(y), type="h")
q <- seq(0, 20, 1)
plot(q, pGEOMo(q), type="h")
p<-seq(0.0001,0.999,0.05)
plot(p , qGEOMo(p), type="s")
dat <- rGEOMo(100)
hist(dat)
#summary(gamlss(dat~1, family="GE"))

```

**Description**

The function GG defines the generalized gamma distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The parameterization used has the mean of the distribution equal to  $\mu$  and the variance equal to  $(\sigma^2)(\mu^2)$ . The functions `dGG`, `pGG`, `qGG` and `rGG` define the density, distribution function, quantile function and random generation for the specific parameterization of the generalized gamma distribution defined by function GG.

**Usage**

```

GG(mu.link = "log", sigma.link = "log",
   nu.link = "identity")
dGG(x, mu=1, sigma=0.5, nu=1,
     log = FALSE)
pGG(q, mu=1, sigma=0.5, nu=1, lower.tail = TRUE,
     log.p = FALSE)
qGG(p, mu=1, sigma=0.5, nu=1, lower.tail = TRUE,
     log.p = FALSE )
rGG(n, mu=1, sigma=0.5, nu=1)

```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter, other links are "inverse" and "identity"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter, other links are "inverse" and "identity"
<code>nu.link</code>	Defines the <code>nu.link</code> , with "identity" link as the default for the <code>sigma</code> parameter, other links are $1/nu^2$ and "log"
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of shape parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

The specific parameterization of the generalized gamma distribution used in GG is

$$f(y|\mu, \sigma, \nu) = \frac{\theta^\theta z^\theta \nu e^{-\theta z}}{(\Gamma(\theta) y)}$$

where  $z = (y/\mu)^\nu$ ,  $\theta = 1/(\sigma^2 \nu^2)$  for  $y > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $-\infty < \nu < +\infty$ . Note that for  $\nu = 0$  the distribution is log normal.

**Value**

GG() returns a `gamlss.family` object which can be used to fit a generalized gamma distribution in the `gamlss()` function. `dGG()` gives the density, `pGG()` gives the distribution function, `qGG()` gives the quantile function, and `rGG()` generates random deviates.

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Nicoleta Motpan

**References**

- Lopatzidis, A. and Green, P. J. (2000), Nonparametric quantile regression using the gamma distribution, unpublished.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).



Stasinopoulos D. M., Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [GA](#)

### Examples

```
y<-rGG(100,mu=1,sigma=0.1, nu=-.5) # generates 100 random observations
hist(y)
# library(gamlss)
#histDist(y, family=GG)
#m1 <-gamlss(y~1,family=GG)
#prof.dev(m1, "nu", min=-2, max=2, step=0.2)
```

---

GIG

*Generalized Inverse Gaussian distribution for fitting a GAMLSS*


---

### Description

The function GIG defines the generalized inverse gaussian distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions DIG, pGIG, GIG and rGIG define the density, distribution function, quantile function and random generation for the specific parameterization of the generalized inverse gaussian distribution defined by function GIG.

### Usage

```
GIG(mu.link = "log", sigma.link = "log",
     nu.link = "identity")
dGIG(x, mu=1, sigma=1, nu=1,
     log = FALSE)
pGIG(q, mu=1, sigma=1, nu=1, lower.tail = TRUE,
     log.p = FALSE)
qGIG(p, mu=1, sigma=1, nu=1, lower.tail = TRUE,
     log.p = FALSE)
rGIG(n, mu=1, sigma=1, nu=1, ...)
```

### Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter, other links are "inverse" and "identity"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter, other links are "inverse" and "identity"

nu.link	Defines the nu.link, with "identity" link as the default for the nu parameter, other links are "inverse" and "log"
x,q	vector of quantiles
mu	vector of location parameter values
sigma	vector of scale parameter values
nu	vector of shape parameter values
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
p	vector of probabilities
n	number of observations. If length(n) > 1, the length is taken to be the number required
...	for extra arguments

### Details

The specific parameterization of the generalized inverse gaussian distribution used in GIG is  $f(y|\mu, \sigma, \nu) = \left(\frac{c}{\mu}\right)^\nu \left(\frac{y^{\nu-1}}{2K\left(\frac{1}{\sigma}, \nu\right)}\right) \left(\exp\left(\left(\frac{-1}{2\sigma}\right)\left(\frac{cy}{\mu} + \frac{\mu}{cy}\right)\right)\right)$  where  $c = \frac{K\left(\frac{1}{\sigma}, \nu+1\right)}{K\left(\frac{1}{\sigma}, \nu\right)}$ , for  $y > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $-\infty < \nu < +\infty$ .

### Value

GIG() returns a gamlss.family object which can be used to fit a generalized inverse gaussian distribution in the gamlss() function. DIG() gives the density, pGIG() gives the distribution function, GIG() gives the quantile function, and rGIG() generates random deviates.

### Author(s)

Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Nicoleta Motpan

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.
- Jorgensen B. (1982) Statistical properties of the generalized inverse Gaussian distribution, Series: Lecture notes in statistics; 9, New York : Springer-Verlag.

**See Also**

[gamlss.family, IG](#)

**Examples**

```
y<-rGIG(100,mu=1,sigma=1, nu=-0.5) # generates 1000 random observations
hist(y)
# library(gamlss)
# histDist(y, family=GIG)
```

---

GPO

*The generalised Poisson distribution*


---

**Description**

The GPO() function defines the generalised Poisson distribution, a two parameter discrete distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dGPO`, `pGPO`, `qGPO` and `rGPO` define the density, distribution function, quantile function and random generation for the Delaporte GPO(), distribution.

**Usage**

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

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

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

qGPO(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE, max.value = 10000)

rGPO(n, mu = 1, sigma = 1, max.value = 10000)
```

**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>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive <code>mu</code>
<code>sigma</code>	vector of positive dispersion parameter <code>sigma</code>
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code>

lower.tail	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$
max.value	a constant, set to the default value of 10000 for how far the algorithm should look for q

### Details

The probability function of the Generalised Poisson distribution is given by

$$P(Y = y | \mu, \sigma) = \left( \frac{\mu}{1 + \sigma\mu} \right)^y \frac{(1 + \sigma y)^{y-1}}{y!} \exp \left[ \frac{-\mu(1 + \sigma y)}{1 + \sigma\mu} \right]$$

for  $y = 0, 1, 2, \dots, \infty$  where  $\mu > 0$  and  $\sigma > 0$ .

### Value

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

### Author(s)

Rigby, R. A., Stasinopoulos D. M.

### References

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

(see also <http://www.gamlss.org/>).

### See Also

[gamlss.family](#), [PO](#), [DPO](#)

### Examples

```
GPO()# gives information about the default links for the
#plot the pdf using plot
plot(function(y) dGPO(y, mu=10, sigma=1 ), from=0, to=100, n=100+1, type="h") # pdf
# plot the cdf
plot(seq(from=0,to=100),pGPO(seq(from=0,to=100), mu=10, sigma=1), type="h") # cdf
# generate random sample
tN <- table(Ni <- rGPO(100, mu=5, sigma=1))
r <- barplot(tN, col='lightblue')
```

GT

*The generalized t distribution for fitting a GAMLSS***Description**

This function defines the generalized t distribution, a four parameter distribution, for a `gamlss.family` object to be used for a GAMLSS fitting using the function `gamlss()`. The functions `dGT`, `pGT`, `qGT` and `rGT` define the density, distribution function, quantile function and random generation for the generalized t distribution.

**Usage**

```
GT(mu.link = "identity", sigma.link = "log", nu.link = "log",
   tau.link = "log")
dGT(x, mu = 0, sigma = 1, nu = 3, tau = 1.5, log = FALSE)
pGT(q, mu = 0, sigma = 1, nu = 3, tau = 1.5, lower.tail = TRUE,
    log.p = FALSE)
qGT(p, mu = 0, sigma = 1, nu = 3, tau = 1.5, lower.tail = TRUE,
    log.p = FALSE)
rGT(n, mu = 0, sigma = 1, nu = 3, tau = 1.5)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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.
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the <code>tau</code> parameter.
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of skewness <code>nu</code> parameter values
<code>tau</code>	vector of kurtosis <code>tau</code> parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code> .
<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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

**Details**

The probability density function of the generalized t distribution, (GT), is defined as

$$f(y|\mu, \sigma, \nu, \tau) = \tau \left\{ 2\sigma\nu^{1/\tau} B\left(\frac{1}{\tau}, \nu\right) [1 + |z|^\tau/\nu]^{\nu+1/\tau} \right\}^{-1}$$

where  $-\infty < y < \infty$ ,  $z = (y - \mu)/\sigma$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$ ,  $\nu > 0$  and  $\tau > 0$ .

**Value**

GT() returns a `gamlss.family` object which can be used to fit the GT distribution in the `gamlss()` function. `dGT()` gives the density, `pGT()` gives the distribution function, `qGT()` gives the quantile function, and `rGT()` generates random deviates.

**Warning**

The `qGT` and `rGT` are slow since they are relying on optimization for finding the quantiles

**Author(s)**

Bob Rigby and Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [JSU](#), [BCT](#)

**Examples**

```
GT() #
y<- rGT(200, mu=5, sigma=1, nu=1, tau=4)
hist(y)
curve(dGT(x, mu=5 ,sigma=2,nu=1, tau=4), -2, 11,
      main = "The GT density mu=5 ,sigma=1, nu=1, tau=4")
# library(gamlss)
# m1<-gamlss(y~1, family=GT)
```

**Description**

The function GU defines the Gumbel distribution, a two parameter distribution, for a `gamlss` family object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dGU`, `pGU`, `qGU` and `rGU` define the density, distribution function, quantile function and random generation for the specific parameterization of the Gumbel distribution.

**Usage**

```
GU(mu.link = "identity", sigma.link = "log")
dGU(x, mu = 0, sigma = 1, log = FALSE)
pGU(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qGU(p, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rGU(n, mu = 0, sigma = 1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" link as the default for the mu parameter. other available link is "inverse", "log" and "own")
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the sigma parameter, other links are the "inverse", "identity" and "own"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

**Details**

The specific parameterization of the Gumbel distribution used in GU is

$$f(y|\mu, \sigma) = \frac{1}{\sigma} \exp \left\{ \left( \frac{y - \mu}{\sigma} \right) - \exp \left( \frac{y - \mu}{\sigma} \right) \right\}$$

for  $y = (-\infty, \infty)$ ,  $\mu = (-\infty, +\infty)$  and  $\sigma > 0$ .

**Value**

`GU()` returns a `gamlss` family object which can be used to fit a Gumbel distribution in the `gamlss()` function. `dGU()` gives the density, `pGU()` gives the distribution function, `qGU()` gives the quantile function, and `rGU()` generates random deviates.

**Note**

The mean of the distribution is  $\mu - 0.57722\sigma$  and the variance is  $\pi^2\sigma^2/6$ .

**Author(s)**

Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Calliope Akantziliotou

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [RG](#)

**Examples**

```
plot(function(x) dGU(x, mu=0,sigma=1), -6, 3,
      main = "{Gumbel density mu=0,sigma=1}")
GU()# gives information about the default links for the Gumbel distribution
dat<-rGU(100, mu=10, sigma=2) # generates 100 random observations
hist(dat)
# library(gamlss)
# gamlss(dat~1,family=GU) # fits a constant for each parameter mu and sigma
```

---

hazardFun

*Hazard functions for gamlss.family distributions*

---

**Description**

The function hazardFun() takes as an argument a gamlss.family object and creates the hazard function for it. The function gen.hazard() generates a hazard function called hNAME where NAME is a gamlss.family i.e. hGA().



**Usage**

```
hazardFun(family = "NO", ...)  
gen.hazard(family = "NO", ...)
```

**Arguments**

family	a <code>gamlss.family</code> object
...	for passing extra arguments

**Value**

A hazard function.

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby and Vlasios Voudouris

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#)

**Examples**

```
gen.hazard("WEI2")  
y<-seq(0,10,by=0.01)  
plot(hWEI2(y, mu=1, sigma=1)~y, type="l", col="black", ylab="h(y)", ylim=c(0,2.5))  
lines(hWEI2(y, mu=1, sigma=1.2)~y, col="red",lt=2,lw=2)  
lines(hWEI2(y, mu=1, sigma=.5)~y, col="blue",lt=3,lw=2)
```

**Description**

The function `IG()`, or equivalently `Inverse.Gaussian()`, defines the inverse Gaussian distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dIG`, `pIG`, `qIG` and `rIG` define the density, distribution function, quantile function and random generation for the specific parameterization of the Inverse Gaussian distribution defined by function `IG`.

**Usage**

```
IG(mu.link = "log", sigma.link = "log")
dIG(x, mu = 1, sigma = 1, log = FALSE)
pIG(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qIG(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rIG(n, mu = 1, sigma = 1, ...)
```

**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 parameter
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations. If <code>length(n) &gt; 1</code> , the length is taken to be the number required
<code>...</code>	<code>...</code> can be used to pass the <code>uppr.limit</code> argument to <code>qIG</code>

**Details**

Definition file for inverse Gaussian distribution.

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

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

**Value**

returns a `gamlss.family` object which can be used to fit a inverse Gaussian distribution in the `gamlss()` function.

**Note**

$\mu$  is the mean and  $\sigma^2\mu^3$  is the variance of the inverse Gaussian

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>)
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [GA](#), [GIG](#)

**Examples**

```
IG()# gives information about the default links for the normal distribution
# library(gamlss)
# data(rent)
# gamlss(R~cs(F1),family=IG, data=rent) #
plot(function(x)dIG(x, mu=1,sigma=.5), 0.01, 6,
      main = "{Inverse Gaussian density mu=1,sigma=0.5}")
plot(function(x)pIG(x, mu=1,sigma=.5), 0.01, 6,
      main = "{Inverse Gaussian cdf mu=1,sigma=0.5}")
```

**Description**

The function `IGAMMA()` defines the Inverse Gamma distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`, with parameters `mu` (the mode) and `sigma`. The functions `dIGAMMA`, `pIGAMMA`, `qIGAMMA` and `rIGAMMA` define the density, distribution function, quantile function and random generation for the IGAMMA parameterization of the Inverse Gamma distribution.

**Usage**

```

IGAMMA(mu.link = "log", sigma.link="log")
dIGAMMA(x, mu = 1, sigma = .5, log = FALSE)
pIGAMMA(q, mu = 1, sigma = .5, lower.tail = TRUE, log.p = FALSE)
qIGAMMA(p, mu = 1, sigma = .5, lower.tail = TRUE, log.p = FALSE)
rIGAMMA(n, mu = 1, sigma = .5)

```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with <code>log link</code> as the default for the <code>mu</code> parameter
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with <code>log</code> as the default for the <code>sigma</code> parameter
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code>
<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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

**Details**

The parameterization of the Inverse Gamma distribution in the function `IGAMMA` is

$$f(y|\mu, \sigma) = \frac{[\mu(\alpha + 1)]^\alpha}{\Gamma(\alpha)} y^{-(\alpha+1)} \exp\left[-\frac{\mu(\alpha + 1)}{y}\right]$$

where  $\alpha = 1/(\sigma^2)$  for  $y > 0$ ,  $\mu > 0$  and  $\sigma > 0$ .

**Value**

returns a `gamlss.family` object which can be used to fit an Inverse Gamma distribution in the `gamlss()` function.

**Note**

For the function `IGAMMA()`, `mu` is the mode of the Inverse Gamma distribution.

**Author(s)**

Fiona McElduff, Bob Rigby and Mikis Stasinopoulos.

## References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, 54, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. 23, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family, GA](#)

## Examples

```
par(mfrow=c(2,2))
y<-seq(0.2,20,0.2)
plot(y, dIGAMMA(y), type="l")
q <- seq(0.2, 20, 0.2)
plot(q, pIGAMMA(q), type="l")
p<-seq(0.0001,0.999,0.05)
plot(p , qIGAMMA(p), type="l")
dat <- rIGAMMA(50)
hist(dat)
#summary(gamlss(dat~1, family="IGAMMA"))
```

## Description

This function defines the , a four parameter distribution, for a `gamlss.family` object to be used for a GAMLSS fitting using the function `gamlss()`. The functions `dJSU`, `pJSU`, `qJSU` and `rJSU` define the density, distribution function, quantile function and random generation for the the Johnson's Su distribution.

## Usage

```
JSU(mu.link = "identity", sigma.link = "log", nu.link = "identity", tau.link = "log")
dJSU(x, mu = 0, sigma = 1, nu = 1, tau = 1, log = FALSE)
pJSU(q, mu = 0, sigma = 1, nu = 1, tau = 1, lower.tail = TRUE, log.p = FALSE)
qJSU(p, mu = 0, sigma = 1, nu = 1, tau = 1, lower.tail = TRUE, log.p = FALSE)
rJSU(n, mu = 0, sigma = 1, nu = 1, tau = 1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" link as the default for the <code>mu</code> parameter. Other links are "inverse" "log" and "own"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter. Other links are "inverse", "identity" and "own"
<code>nu.link</code>	Defines the <code>nu.link</code> , with "identity" link as the default for the <code>nu</code> parameter. Other links are "onverse", "log" and "own"
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the <code>tau</code> parameter. Other links are "onverse", "identity" and "own"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of skewness <code>nu</code> parameter values
<code>tau</code>	vector of kurtosis <code>tau</code> parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

The probability density function of the Johnson's SU distribution, (JSU), is defined as

$$f(y|n, \mu, \sigma, \nu, \tau) = \frac{1}{c\sigma} \frac{1}{\tau(z^2 + 1)^{\frac{1}{2}}} \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}r^2\right]$$

for  $-\infty < y < \infty$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$ ,  $\nu = (-\infty, +\infty)$  and  $\tau > 0$ . where  $r = -\nu + \frac{1}{\tau} \sinh^{-1}(z)$ ,  $z = \frac{y - (\mu + c\sigma w^{\frac{1}{2}} \sinh \Omega)}{c\sigma}$ ,  $c = [\frac{1}{2}(w - 1)(w \cosh 2\Omega + 1)]^{\frac{1}{2}}$ ,  $w = e^{\tau^2}$  and  $\Omega = -\nu\tau$ .

This is a reparameterization of the original Johnson Su distribution, Johnson (1954), so the parameters `mu` and `sigma` are the mean and the standard deviation of the distribution. The parameter `nu` determines the skewness of the distribution with `nu>0` indicating positive skewness and `nu<0` negative. The parameter `tau` determines the kurtosis of the distribution. `tau` should be positive and most likely in the region from zero to 1. As `tau` goes to 0 (and for `nu=0`) the distribution approaches the the Normal density function. The distribution is appropriate for leptokurtic data that is data with kurtosis larger than the Normal distribution one.

**Value**

`JSU()` returns a `gamlss.family` object which can be used to fit a Johnson's Su distribution in the `gamlss()` function. `dJSU()` gives the density, `pJSU()` gives the distribution function, `qJSU()` gives the quantile function, and `rJSU()` generates random deviates.

**Warning**

The function JSU uses first derivatives square in the fitting procedure so standard errors should be interpreted with caution

**Author(s)**

Bob Rigby and Mikis Stasinopoulos

**References**

- Johnson, N. L. (1954). Systems of frequency curves derived from the first law of Laplace., *Trabajos de Estadística*, **5**, 283-291.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [JSUo](#), [BCT](#)

**Examples**

```
JSU()
plot(function(x)dJSU(x, mu=0,sigma=1,nu=-1, tau=.5), -4, 4,
      main = "The JSU density mu=0,sigma=1,nu=-1, tau=.5")
plot(function(x) pJSU(x, mu=0,sigma=1,nu=-1, tau=.5), -4, 4,
      main = "The JSU cdf mu=0, sigma=1, nu=-1, tau=.5")
# library(gamlss)
# data(abdom)
# h<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=JSU, data=abdom)
```

**Description**

This function defines the , a four parameter distribution, for a `gamlss.family` object to be used for a GAMLSS fitting using the function `gamlss()`. The functions `dJSUo`, `pJSUo`, `qJSUo` and `rJSUo` define the density, distribution function, quantile function and random generation for the the Johnson's Su distribution.

**Usage**

```

JSUo(mu.link = "identity", sigma.link = "log", nu.link = "identity", tau.link = "log")
dJSUo(x, mu = 0, sigma = 1, nu = 0, tau = 1, log = FALSE)
pJSUo(q, mu = 0, sigma = 1, nu = 0, tau = 1, lower.tail = TRUE, log.p = FALSE)
qJSUo(p, mu = 0, sigma = 1, nu = 0, tau = 1, lower.tail = TRUE, log.p = FALSE)
rJSUo(n, mu = 0, sigma = 1, nu = 0, tau = 1)

```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" link as the default for the <code>mu</code> parameter. Other links are "inverse", "log" and "own"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter. Other links are "inverse", "identity" and "own"
<code>nu.link</code>	Defines the <code>nu.link</code> , with "identity" link as the default for the <code>nu</code> parameter. Other links are "inverse", "log" and "own"
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the <code>tau</code> parameter. Other links are "inverse", "identity" and "own"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of skewness <code>nu</code> parameter values
<code>tau</code>	vector of kurtosis <code>tau</code> parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code> .
<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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

**Details**

The probability density function of the original Johnson's SU distribution, (JSU), is defined as

$$f(y|n, \mu, \sigma, \nu, \tau) = \frac{\tau}{\sigma} \frac{1}{(z^2 + 1)^{\frac{1}{2}}} \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}r^2\right]$$

for  $-\infty < y < \infty$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$ ,  $\nu = (-\infty, +\infty)$  and  $\tau > 0$ . where  $z = \frac{(y-\mu)}{\sigma}$ ,  $r = \nu + \tau \sinh^{-1}(z)$ .

**Value**

`JSUo()` returns a `gamlss.family` object which can be used to fit a Johnson's Su distribution in the `gamlss()` function. `dJSUo()` gives the density, `pJSUo()` gives the distribution function, `qJSUo()` gives the quantile function, and `rJSUo()` generates random deviates.



**Warning**

The function JSU uses first derivatives square in the fitting procedure so standard errors should be interpreted with caution. It is recomented to be used only with `method=mixed(2,20)`

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)> and Bob Rigby

**References**

Johnson, N. L. (1954). Systems of frequency curves derived from the first law of Laplace., *Trabajos de Estadistica*, **5**, 283-291.

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [JSU](#), [BCT](#)

**Examples**

```
JSU()
plot(function(x)dJSUo(x, mu=0,sigma=1,nu=-1, tau=.5), -4, 15,
      main = "The JSUo density mu=0,sigma=1,nu=-1, tau=.5")
plot(function(x) pJSUo(x, mu=0,sigma=1,nu=-1, tau=.5), -4, 15,
      main = "The JSUo cdf mu=0, sigma=1, nu=-1, tau=.5")
# library(gamlss)
# data(abdom)
# h<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=JSUo,
#          data=abdom, method=mixed(2,20))
# plot(h)
```

---

LG *Logarithmic and zero adjusted logarithmic distributions for fitting a GAMLSS model*

---

## Description

The function LG defines the logarithmic distribution, a one parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dLG`, `pLG`, `qLG` and `rLG` define the density, distribution function, quantile function and random generation for the logarithmic, `LG()`, distribution.

The function ZALG defines the zero adjusted logarithmic distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZALG`, `pZALG`, `qZALG` and `rZALG` define the density, distribution function, quantile function and random generation for the inflated logarithmic, `ZALG()`, distribution.

## Usage

```
LG(mu.link = "logit")
dLG(x, mu = 0.5, log = FALSE)
pLG(q, mu = 0.5, lower.tail = TRUE, log.p = FALSE)
qLG(p, mu = 0.5, lower.tail = TRUE, log.p = FALSE, max.value = 10000)
rLG(n, mu = 0.5)
ZALG(mu.link = "logit", sigma.link = "logit")
dZALG(x, mu = 0.5, sigma = 0.1, log = FALSE)
pZALG(q, mu = 0.5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
qZALG(p, mu = 0.5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
rZALG(n, mu = 0.5, sigma = 0.1)
```

## Arguments

<code>mu.link</code>	defines the <code>mu.link</code> , with <code>logit</code> link as the default for the <code>mu</code> parameter
<code>sigma.link</code>	defines the <code>sigma.link</code> , with <code>logit</code> link as the default for the <code>sigma</code> parameter which in this case is the probability at zero.
<code>x</code>	vector of (non-negative integer)
<code>mu</code>	vector of positive means
<code>sigma</code>	vector of probabilities at zero
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$
<code>max.value</code>	valued needed for the numerical calculation of the q-function

**Details**

For the definition of the distributions see Rigby and Stasinopoulos (2010) below.

The parameterization of the logarithmic distribution in the function LM is

$$f(y|\mu) = \alpha\mu^y/y$$

where for  $y \geq 1$  and  $\mu > 0$  and

$$\alpha = -[\log(1 - \mu)]^{-1}$$

**Value**

The function LG and ZALG return a `gamlss.family` object which can be used to fit a logarithmic and a zero inflated logarithmic distributions respectively in the `gamlss()` function.

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby

**References**

Johnson, Norman Lloyd; Kemp, Adrienne W; Kotz, Samuel (2005). "Chapter 7: Logarithmic and Lagrangian distributions". Univariate discrete distributions (3 ed.). John Wiley & Sons. ISBN 9780471272465.

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.com/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Rigby, R. A. and Stasinopoulos D. M. (2010) The `gamlss.family` distributions, (distributed with this package or see <http://www.gamlss.org/>)

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [P0](#), [ZAP](#)

**Examples**

```
LG()
ZAP()
# creating data and plotting them
dat <- rLG(1000, mu=.3)
r <- barplot(table(dat), col='lightblue')
dat1 <- rZALG(1000, mu=.3, sigma=.1)
r1 <- barplot(table(dat1), col='lightblue')
```

## Description

The functions LOGNO and LOGNO2 define a `gamlss.family` distribution to fits the log-Normal distribution. The difference between them is that while LOGNO retains the original parametrization for  $\mu$ , (identical to the normal distribution NO) and therefore  $\mu = (-\infty, +\infty)$ , the function LOGNO2 use  $\mu$  as the median, so  $\mu = (0, +\infty)$ .

The function LNO is more general and can fit a Box-Cox transformation to data using the `gamlss()` function. In the LOGNO (and LOGNO2) there are two parameters involved  $\mu$   $\sigma$ , while in the LNO there are three parameters  $\mu$   $\sigma$ , and the transformation parameter  $\nu$ . The transformation parameter  $\nu$  in LNO is a 'fixed' parameter (not estimated) and it has its default value equal to zero allowing the fitting of the log-normal distribution as in LOGNO. See the example below on how to fix  $\nu$  to be a particular value. In order to estimate (or model) the parameter  $\nu$ , use the `gamlss.family` `BCCG` distribution which uses a reparameterized version of the the Box-Cox transformation. The functions `dLOGNO`, `pLOGNO`, `qLOGNO` and `rLOGNO` define the density, distribution function, quantile function and random generation for the specific parameterization of the log-normal distribution.

The functions `dLOGNO2`, `pLOGNO2`, `qLOGNO2` and `rLOGNO2` define the density, distribution function, quantile function and random generation when  $\mu$  is the median of the log-normal distribution.

The functions `dLNO`, `pLNO`, `qLNO` and `rLNO` define the density, distribution function, quantile function and random generation for the specific parameterization of the log-normal distribution and more generally a Box-Cox transformation.

## Usage

```
LNO(mu.link = "identity", sigma.link = "log")
LOGNO(mu.link = "identity", sigma.link = "log")
LOGNO2(mu.link = "log", sigma.link = "log")
dLNO(x, mu = 1, sigma = 0.1, nu = 0, log = FALSE)
dLOGNO(x, mu = 0, sigma = 1, log = FALSE)
dLOGNO2(x, mu = 1, sigma = 1, log = FALSE)
pLNO(q, mu = 1, sigma = 0.1, nu = 0, lower.tail = TRUE, log.p = FALSE)
pLOGNO(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
pLOGNO2(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qLNO(p, mu = 1, sigma = 0.1, nu = 0, lower.tail = TRUE, log.p = FALSE)
qLOGNO(p, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qLOGNO2(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rLNO(n, mu = 1, sigma = 0.1, nu = 0)
rLOGNO(n, mu = 0, sigma = 1)
rLOGNO2(n, mu = 1, sigma = 1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" or "log" link depending on the parametrization
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the sigma parameter. Other links are "inverse", "identity" and "own"
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of shape parameter values
<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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

**Details**

The probability density function in LOGNO is defined as

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

for  $y > 0$ ,  $\mu = (-\infty, +\infty)$  and  $\sigma > 0$ .

The probability density function in LNO is defined as

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

where if  $\nu \neq 0$   $z = (y^\nu - 1)/\nu$  else  $z = \log(y)$  and  $z \sim N(0, \sigma^2)$ , for  $y > 0$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $\nu = (-\infty, +\infty)$ .

**Value**

`LNO()` returns a `gamlss.family` object which can be used to fit a log-normal distribution in the `gamlss()` function. `dLNO()` gives the density, `pLNO()` gives the distribution function, `qLNO()` gives the quantile function, and `rLNO()` generates random deviates.

**Warning**

This is a two parameter fit for  $\mu$  and  $\sigma$  while  $\nu$  is fixed. If you wish to model  $\nu$  use the `gamlss` family `BCCG`.

**Note**

$\mu$  is the mean of  $z$  (and also the median of  $y$ ), the Box-Cox transformed variable and  $\sigma$  is the standard deviation of  $z$  and approximate the coefficient of variation of  $y$

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

**References**

- Box, G. E. P. and Cox, D. R. (1964) An analysis of transformations (with discussion), *J. R. Statist. Soc. B.*, **26**, 211–252
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [BCCG](#)

**Examples**

```
LOGNO()# gives information about the default links for the log normal distribution
LOGNO2()
LNO()# gives information about the default links for the Box Cox distribution

# plotting the d, p, q, and r functions
op<-par(mfrow=c(2,2))
curve(dLOGNO(x, mu=0), 0, 10)
curve(pLOGNO(x, mu=0), 0, 10)
curve(qLOGNO(x, mu=0), 0, 1)
Y<- rLOGNO(200)
hist(Y)
par(op)

# plotting the d, p, q, and r functions
op<-par(mfrow=c(2,2))
curve(dLOGNO2(x, mu=1), 0, 10)
curve(pLOGNO2(x, mu=1), 0, 10)
curve(qLOGNO2(x, mu=1), 0, 1)
Y<- rLOGNO(200)
hist(Y)
par(op)

# library(gamlss)
# data(abdom)
# h1<-gamlss(y~cs(x), family=LOGNO, data=abdom)#fits the log-Normal distribution
```

```
# h2<-gamlss(y~cs(x), family=LNO, data=abdom) #should be identical to the one above
# to change to square root transformation, i.e. fix nu=0.5
# h3<-gamlss(y~cs(x), family=LNO, data=abdom, nu.fix=TRUE, nu.start=0.5)
```

LO

*Logistic distribution for fitting a GAMLSS***Description**

The function `LO()`, or equivalently `Logistic()`, defines the logistic distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`

**Usage**

```
LO(mu.link = "identity", sigma.link = "log")
dLO(x, mu = 0, sigma = 1, log = FALSE)
pLO(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qLO(p, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rLO(n, mu = 0, sigma = 1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

Definition file for Logistic distribution.

$$f(y|\mu, \sigma) = \frac{1}{\sigma} e^{-\frac{y-\mu}{\sigma}} [1 + e^{-\frac{y-\mu}{\sigma}}]^{-2}$$

for  $y = (-\infty, \infty)$ ,  $\mu = (-\infty, \infty)$  and  $\sigma > 0$ .

**Value**

`LO()` returns a `gamlss.family` object which can be used to fit a logistic distribution in the `gamlss()` function. `dLO()` gives the density, `pLO()` gives the distribution function, `qLO()` gives the quantile function, and `rLO()` generates random deviates for the logistic distribution. The latest functions are based on the equivalent R functions for logistic distribution.

**Note**

$\mu$  is the mean and  $\sigma\pi/\sqrt{3}$  is the standard deviation for the logistic distribution

**Author(s)**

Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Calliope Akantziliotou

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), NO, TF

**Examples**

```
L0()# gives information about the default links for the Logistic distribution
plot(function(y) dL0(y, mu=10 ,sigma=2), 0, 20)
plot(function(y) pL0(y, mu=10 ,sigma=2), 0, 20)
plot(function(y) qL0(y, mu=10 ,sigma=2), 0, 1)
# library(gamlss)
# data(abdom)
# h<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=L0, data=abdom) # fits
# plot(h)
```

---

LOGITNO

*Logit Normal distribution for fitting in GAMLSS*

---

**Description**

The functions dLOGITNO, pLOGITNO, qLOGITNO and rLOGITNO define the density, distribution function, quantile function and random generation for the logit-normal distribution. The function LOGITNO can be used for fitting the distribution in `gamlss()`.



**Usage**

```
LOGITNO(mu.link = "logit", sigma.link = "log")
dLOGITNO(x, mu = 0.5, sigma = 1, log = FALSE)
pLOGITNO(q, mu = 0.5, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qLOGITNO(p, mu = 0.5, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rLOGITNO(n, mu = 0.5, sigma = 1)
```

**Arguments**

<code>mu.link</code>	the link function for mu
<code>sigma.link</code>	the link function for sigma
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations. If length(n) > 1, the length is taken to be the number required

**Details**

The probability density function in LOGITNO is defined as

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

for  $0 < y < 1$ ,  $\mu \in (0, 1)$  and  $\sigma > 0$ .

**Value**

LOGITNO() returns a `gamlss.family` object which can be used to fit a logit-normal distribution in the `gamlss()` function.

**Author(s)**

Mikis Stasinopoulos, Bob Rigby

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape, (with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M., Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family](#), LOGNO

## Examples

```
# plotting the d, p, q, and r functions
op<-par(mfrow=c(2,2))
curve(dLOGITNO(x), 0, 1)
curve(pLOGITNO(x), 0, 1)
curve(qLOGITNO(x), 0, 1)
Y<- rLOGITNO(200)
hist(Y)
par(op)
```

```
# plotting the d, p, q, and r functions
# sigma 3
op<-par(mfrow=c(2,2))
curve(dLOGITNO(x, sigma=3), 0, 1)
curve(pLOGITNO(x, sigma=3), 0, 1)
curve(qLOGITNO(x, sigma=3), 0, 1)
Y<- rLOGITNO(200, sigma=3)
hist(Y)
par(op)
```

---

LQNO

*Normal distribution with a specific mean and variance relationship for fitting a GAMLSS model*

---

## Description

The function `LQNO()` defines a normal distribution family, which has a specific mean and variance relationship. The distribution can be used in a GAMLSS fitting using the function `gamlss()`. The mean of `LQNO` is equal to  $\mu$ . The variance is equal to  $\mu \cdot (1 + \sigma \cdot \mu)$  so the standard deviation is  $\sqrt{\mu \cdot (1 + \sigma \cdot \mu)}$ . The function is found useful in modelling small RNA sequencing experiments. The functions `dLQNO`, `pLQNO`, `qLQNO` and `rLQNO` define the density, distribution function, quantile function (inverse cdf) and random generation for the `LQNO()` parametrization of the normal distribution.

**Usage**

```
LQNO(mu.link = "log", sigma.link = "log")
dLQNO(x, mu = 1, sigma = 1, log = FALSE)
pLQNO(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qLQNO(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rLQNO(n, mu = 1, sigma = 1)
```

**Arguments**

<code>mu.link</code>	mu link function with "log" as default
<code>sigma.link</code>	mu link function with "log" as default
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities p are given as log(p)
<code>lower.tail</code>	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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

**Details**

LQNO stands for Linear Quadratic Normal Family, in which the variance is a linear quadratic function of the mean:  $\text{Var}(Y) = \mu \cdot (1 + \sigma \cdot \mu)$ . This is created to facilitate the analysis of data coming from small RNA sequencing experiments, basically counts of short RNAs that one isolates from cells or biofluids such as urine, plasma or cerebrospinal fluid. Argyropoulos *et al.* (2017) showing that the LQNO distribution (and the Negative Binomial which implements the same mean- variance relationship) are highly accurate approximations to the generative models of the signals in these experiments

**Value**

The function LQNO returns a `gamlss.family` object which can be used to fit this specific form of the normal distribution family in the `gamlss()` function.

**Note**

The mu parameters must be positive so for the relationship  $\text{Var}(Y) = \mu \cdot (1 + \sigma \cdot \mu)$  to be valid.

**Author(s)**

Christos Argyropoulos

**References**

Argyropoulos C, Etheridge A, Sakhanenko N, Galas D. (2017) Modeling bias and variation in the stochastic processes of small RNA sequencing. *Nucleic Acids Res.* 2017 Mar 27. doi: 10.1093/nar/gkx199. [Epub ahead of print] PubMed PMID: 28369495.

**See Also**[NO,NO2, NOF](#)**Examples**

```
LQNO()# gives information about the default links for the normal distribution
# a comparison of different Normal models
#m1 <- gamlss(y~pb(x), sigma.fo=~pb(x), data=abdom, family=NO(mu.link="log"))
#m2 <- gamlss(y~pb(x), sigma.fo=~pb(x), data=abdom, family=LQNO)
#m3 <- gamlss(y~pb(x), sigma.fo=~pb(x), data=abdom, family=NOF(mu.link="log"))
#AIC(m1,m2,m3)
```

make.link.gamlss

*Create a Link for GAMLSS families***Description**

The function `make.link.gamlss()` is used with `gamlss.family` distributions in package **gamlss()**. Given a link, it returns a link function, an inverse link function, the derivative `dpar/deta` where 'par' is the appropriate distribution parameter and a function for checking the domain. It differs from the usual `make.link` of `glm()` by having extra links as the `logshift1`, and the `own`. For the use of the `own` link see the example bellow. `show.link` provides a way in which the user can identify the link functions available for each `gamlss` distribution. If your required link function is not available for any of the `gamlss` distributions you can add it in.

**Usage**

```
make.link.gamlss(link)
show.link(family = "NO")
```

**Arguments**

link	character or numeric; one of "logit", "probit", "cloglog", "identity", "log", "sqrt", "1/mu^2", "inverse", "logshifted", "logitshifted", or number, say lambda resulting in power link $\mu^\lambda$ .
family	a <code>gamlss</code> distribution family

**Details**

The `own` link function is added to allow the user greater flexibility. In order to used the `own` link function for any of the parameters of the distribution the `own` link should appear in the available links for this parameter. You can check this using the function `show.link`. If the `own` do not appear in the list you can create a new function for the distribution in which `own` is added in the list. For example the first line of the code of the binomial distribution, `BI`, has change from

```
"mstats <- checklink("mu.link", "Binomial", substitute(mu.link), c("logit", "probit", "cloglog", "log")),
in version 1.0-0 of gamlss, to
```

```
"mstats <- checklink("mu.link", "Binomial", substitute(mu.link), c("logit", "probit", "cloglog", "log",
"own"))
```

in version 1.0-1. Given that the parameter has own as an option the user needs also to define the following four new functions in order to used an own link.

i) own.linkfun

ii) own.linkinv

iii) own.mu.eta and

iv) own.valideta.

An example is given below.

Only one parameter of the distribution at a time is allowed to have its own link, (unless the same four own functions above are suitable for more that one parameter of the distribution).

Note that from **gamlss** version 1.9-0 the user can introduce its own link function by define an appropriate function, (see the example below).

### Value

For the `make.link.gamlss` a list with components

`linkfun`: Link function `function(parameter)`

`linkinv`: Inverse link function `function(eta)`

`mu.eta`: Derivative function `function(eta) dparameter/deta`

`valideta`: `function(eta)` TRUE if all of eta is in the domain of `linkinv`.

For the `show.link` a list with components the available links for the distribution parameters

### Note

For the links involving parameters as in `logshifted` and `logitshifted` the parameters can be passed in the definition of the distribution by calling the `checklink` function, for example in the definition of the tau parameter in BCPE distribution the following call is made: `tstats <- checklink("tau.link", "Bo`

### Author(s)

Mikis Stasinopoulos and Bob Rigby

### References

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**[gamlss.family](#)**Examples**

```

str(make.link.gamlss("logshiftt01"))
l2<-make.link.gamlss("logshiftt01")
l2$linkfun(2) # should close to zero (Note that 0.00001 is added)
l2$linkfun(1-0.00001) # should be -Inf but it is large negative
#-----
# now use the own link function
# first if the distribution allows you
show.link(BI)
# seems OK now define the four own functions
# First try the probit link using the own link function
# 1: the linkfun function
own.linkfun <- function(mu) { qNO(p=mu)}
# 2: the inverse link function
own.linkinv <- function(eta) {
  thresh <- -qNO(.Machine$double.eps)
  eta <- pmin(thresh, pmax(eta, -thresh))
  pNO(eta)}
# 3: the dmu/deta function
own.mu.eta <- function(eta) pmax(dNO(eta), .Machine$double.eps)
# 4: the valideta function
own.valideta <- function(eta) TRUE

## bring the data
# library(gamlss)
#data(aep)
# fitting the model using "own"
# h1<-gamlss(y~ward+loglos+year, family=BI(mu.link="own"), data=aep)
# model h1 should be identical to the probit
# h2<-gamlss(y~ward+loglos+year, family=BI(mu.link="probit"), data=aep)
# now using a function instead of "own"
probittest <- function()
{
linkfun <- function(mu) { qNO(p=mu)}
linkinv <- function(eta)
  {
  thresh <- -qNO(.Machine$double.eps)
  eta <- pmin(thresh, pmax(eta, -thresh))
  pNO(eta)
  }
mu.eta <- function(eta) pmax(dNO(eta), .Machine$double.eps)
valideta <- function(eta) TRUE
link <- "probitTest"
structure(list(linkfun = linkfun, linkinv = linkinv, mu.eta = mu.eta,
  valideta = valideta, name = link), class = "link-gamlss")
}
# h3<-gamlss(y~ward+loglos+year, family=BI(mu.link=probittest()), data=aep)
# Second try the complementary log-log

```

```

# using the Gumbel distribution
own.linkfun <- function(mu) { qGU(p=mu)}
own.linkinv <- function(eta) {
  thresh <- -qGU(.Machine$double.eps)
  eta <- pmin(thresh, pmax(eta, -thresh))
  pGU(eta)}
own.mu.eta <- function(eta) pmax(dGU(eta), .Machine$double.eps)
own.valideta <- function(eta) TRUE
# h1 and h2 should be identical to cloglog
# h1<-gamlss(y~ward+loglos+year, family=BI(mu.link="own"), data=aep)
# h2<-gamlss(y~ward+loglos+year, family=BI(mu.link="cloglog"), data=aep)
# note that the Gumbel distribution is negatively skew
# for a positively skew link function we can use the Reverse Gumbel
revloglog <- function()
{
linkfun <- function(mu) { qRG(p=mu)}
linkinv <- function(eta) {
  thresh <- -qRG(.Machine$double.eps)
  eta <- pmin(thresh, pmax(eta, -thresh))
  pRG(eta)}
mu.eta <- function(eta) pmax(dRG(eta), .Machine$double.eps)
valideta <- function(eta) TRUE
link <- "revloglog"
structure(list(linkfun = linkfun, linkinv = linkinv, mu.eta = mu.eta,
  valideta = valideta, name = link), class = "link-gamlss")
}
# h1<-gamlss(y~ward+loglos+year, family=BI(mu.link=revloglog()), data=aep)
# a considerable improvement in the deviance
# try a shifted logit link function from -1, 1
own.linkfun <- function(mu)
  { shift = c(-1,1)
  log((mu-shift[1])/(shift[2]-mu))
  }
own.linkinv <- function(eta)
  {
  shift = c(-1,1)
  thresh <- -log(.Machine$double.eps)
  eta <- pmin(thresh, pmax(eta, -thresh))
  shift[2]-(shift[2]-shift[1])/(1 + exp(eta))
  }
own.mu.eta <- function(eta)
  {
  shift = c(-1,1)
  thresh <- -log(.Machine$double.eps)
  res <- rep(.Machine$double.eps, length(eta))
  res[abs(eta) < thresh] <- ((shift[2]-shift[1])*exp(eta)/(1 +
  exp(eta))^2)[abs(eta) < thresh]
  res
  }
own.valideta <- function(eta) TRUE
#-----
str(make.link.gamlss("own"))
l2<-make.link.gamlss("own")

```

```

l2$linkfun(0) # should be zero
l2$linkfun(1) # should be Inf
l2$linkinv(-5:5)

```

MN3

*Multinomial distribution in GAMLSS***Description**

The set of function presented here is useful for fitting multinomial regression within gamlss.

**Usage**

```

MN3(mu.link = "log", sigma.link = "log")
MN4(mu.link = "log", sigma.link = "log", nu.link = "log")
MN5(mu.link = "log", sigma.link = "log", nu.link = "log", tau.link = "log")
MULTIN(type = "3")
fittedMN(model)

dMN3(x, mu = 1, sigma = 1, log = FALSE)
dMN4(x, mu = 1, sigma = 1, nu = 1, log = FALSE)
dMN5(x, mu = 1, sigma = 1, nu = 1, tau = 1, log = FALSE)

pMN3(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
pMN4(q, mu = 1, sigma = 1, nu = 1, lower.tail = TRUE, log.p = FALSE)
pMN5(q, mu = 1, sigma = 1, nu = 1, tau = 1, lower.tail = TRUE, log.p = FALSE)

qMN3(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qMN4(p, mu = 1, sigma = 1, nu = 1, lower.tail = TRUE, log.p = FALSE)
qMN5(p, mu = 1, sigma = 1, nu = 1, tau = 1, lower.tail = TRUE, log.p = FALSE)

rMN3(n, mu = 1, sigma = 1)
rMN4(n, mu = 1, sigma = 1, nu = 1)
rMN5(n, mu = 1, sigma = 1, nu = 1, tau = 1)

```

**Arguments**

<code>mu.link</code>	the link function for mu
<code>sigma.link</code>	the link function for sigma
<code>nu.link</code>	the link function for nu
<code>tau.link</code>	the link function for tau
<code>x</code>	the x variable
<code>q</code>	vector of quantiles
<code>p</code>	vector of probabilities
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ otherwise, $P[X > x]$ .



log.p	logical; if TRUE, probabilities p are given as log(p).
log	logical; if TRUE, probabilities p are given as log(p).
n	the number of observations
mu	the mu parameter
sigma	the sigma parameter
nu	the nu parameter
tau	the tau parameter
type	permitted values are 2 (Binomial), 3, 4, and 5
model	a gamlss multinomial fitted model

### Details

GAMLSS is in general not suitable for multinomial regression. Nevertheless multinomial regression can be fitted within GAMLSS if the response variable  $y$  has less than five categories. The function here provide the facilities to do so. The functions `MN3()`, `MN4()` and `MN5()` fit multinomial responses with 3, 4 and 5 categories respectively. The function `MULTIN()` can be used instead of `codeMN3()`, `MN4()` and `MN5()` by specifying the number of levels of the response. Note that `MULTIN(2)` will produce a binomial fit.

### Value

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

### Author(s)

Mikis Stasinopoulos, Bob Rigby and Vlasios Voudouris

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [BI](#)

**Examples**

```

dMN3(3)
pMN3(2)
qMN3(.6)
rMN3(10)

```

---

NBF

*Negative Binomial Family distribution for fitting a GAMLSS*


---

**Description**

The `NBF()` function defines the Negative Binomial family distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dNBF`, `pNBF`, `qNBF` and `rNBF` define the density, distribution function, quantile function and random generation for the negative binomial family, `NBF()`, distribution.

The functions `dZINBF`, `pZINBF`, `qZINBF` and `rZINBF` define the density, distribution function, quantile function and random generation for the zero inflated negative binomial family, `ZINBF()`, distribution a four parameter distribution.

**Usage**

```

NBF(mu.link = "log", sigma.link = "log", nu.link = "log")

dNBF(x, mu = 1, sigma = 1, nu = 2, log = FALSE)

pNBF(q, mu = 1, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)

qNBF(p, mu = 1, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)

rNBF(n, mu = 1, sigma = 1, nu = 2)

ZINBF(mu.link = "log", sigma.link = "log", nu.link = "log",
      tau.link = "logit")

dZINBF(x, mu = 1, sigma = 1, nu = 2, tau = 0.1, log = FALSE)

pZINBF(q, mu = 1, sigma = 1, nu = 2, tau = 0.1, lower.tail = TRUE,
      log.p = FALSE)

qZINBF(p, mu = 1, sigma = 1, nu = 2, tau = 0.1, lower.tail = TRUE,
      log.p = FALSE)

rZINBF(n, mu = 1, sigma = 1, nu = 2, tau = 0.1)

```

**Arguments**

<code>mu.link</code>	The link function for mu
<code>sigma.link</code>	The link function for sigma
<code>nu.link</code>	The link function for nu
<code>tau.link</code>	The link function for tau
<code>x</code>	vector of (non-negative integer)
<code>mu</code>	vector of positive means
<code>sigma</code>	vector of positive dispersion parameter
<code>nu</code>	vector of power parameter
<code>tau</code>	vector of inflation 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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return

**Details**

The definition for Negative Binomial Family distribution, NBF, is similar to the Negative Binomial type I. The probability function of the NBF can be obtained by replacing  $\sigma$  with  $\sigma\mu^{\nu-2}$  where  $\nu$  is a power parameter. The distribution has mean  $\mu$  and variance  $\mu + \sigma\mu^{\nu}$ .

**Value**

returns a `gamlss.family` object which can be used to fit a Negative Binomial Family distribution in the `gamlss()` function.

**Author(s)**

Bob Rigby and Mikis Stasinopoulos

**References**

- Anscombe, F. J. (1950) Sampling theory of the negative binomial and logarithmic distributions, *Biometrika*, **37**, 358-382.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[NBI](#), [NBII](#)

### Examples

```
NBF() # default link functions for the Negative Binomial Family
# plotting the distribution
plot(function(y) dNBF(y, mu = 10, sigma = 0.5, nu=2 ), from=0,
      to=40, n=40+1, type="h")
# creating random variables and plot them
tN <- table(Ni <- rNBF(1000, mu=5, sigma=0.5, nu=2))
r <- barplot(tN, col='lightblue')
# zero inflated NBF
ZINBF() # default link functions for the zero inflated NBF
# plotting the distribution
plot(function(y) dZINBF(y, mu = 10, sigma = 0.5, nu=2, tau=.1 ),
      from=0, to=40, n=40+1, type="h")
# creating random variables and plot them
tN <- table(Ni <- rZINBF(1000, mu=5, sigma=0.5, nu=2, tau=0.1))
r <- barplot(tN, col='lightblue')
## Not run:
library(gamlss)
data(species)
species <- transform(species, x=log(lake))
m6 <- gamlss(fish~poly(x,2), sigma.fo=~1, data=species, family=NBF,
            n.cyc=200)
fitted(m6, "nu")[1]

## End(Not run)
```

---

NBI

*Negative Binomial type I distribution for fitting a GAMLSS*

---

### Description

The `NBI()` function defines the Negative Binomial type I distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dNBI`, `pNBI`, `qNBI` and `rNBI` define the density, distribution function, quantile function and random generation for the Negative Binomial type I, `NBI()`, distribution.

**Usage**

```

NBI(mu.link = "log", sigma.link = "log")
dNBI(x, mu = 1, sigma = 1, log = FALSE)
pNBI(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qNBI(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rNBI(n, mu = 1, sigma = 1)

```

**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 parameter
<code>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive means
<code>sigma</code>	vector of positive dispersion parameter
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<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 <= x], otherwise, P[X > x]

**Details**

Definition file for Negative Binomial type I distribution.

$$P(Y = y|\mu, \sigma) = \frac{\Gamma(y + \frac{1}{\sigma})}{\Gamma(\frac{1}{\sigma})\Gamma(y + 1)} \left(\frac{\sigma\mu}{1 + \sigma\mu}\right)^y \left(\frac{1}{1 + \sigma\mu}\right)^{1/\sigma}$$

for  $y = 0, 1, 2, \dots, \infty$ ,  $\mu > 0$  and  $\sigma > 0$ . This parameterization is equivalent to that used by Anscombe (1950) except he used  $\alpha = 1/\sigma$  instead of  $\sigma$ .

**Value**

returns a `gamlss.family` object which can be used to fit a Negative Binomial type I distribution in the `gamlss()` function.

**Warning**

For values of  $\sigma < 0.0001$  the `d,p,q,r` functions switch to the Poisson distribution

**Note**

$\mu$  is the mean and  $(\mu + \sigma\mu^2)^{0.5}$  is the standard deviation of the Negative Binomial type I distribution (so  $\sigma$  is the dispersion parameter in the usual GLM for the negative binomial type I distribution)

**Author(s)**

Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Calliope Akantzi-Iotou

## References

- Anscombe, F. J. (1950) Sampling theory of the negative binomial and logarithmic distributiona, *Biometrika*, **37**, 358-382.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family](#), [NBII](#), [PIG](#), [SI](#)

## Examples

```
NBI() # gives information about the default links for the Negative Binomial type I distribution
# plotting the distribution
plot(function(y) dNBI(y, mu = 10, sigma = 0.5 ), from=0, to=40, n=40+1, type="h")
# creating random variables and plot them
tN <- table(Ni <- rNBI(1000, mu=5, sigma=0.5))
r <- barplot(tN, col='lightblue')
# library(gamlss)
# data(aids)
# h<-gamlss(y~cs(x,df=7)+qrt, family=NBII, data=aids) # fits the model
# plot(h)
# pdf.plot(family=NBII, mu=10, sigma=0.5, min=0, max=40, step=1)
```

---

NBII

*Negative Binomial type II distribution for fitting a GAMLSS*

---

## Description

The NBII() function defines the Negative Binomial type II distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dNBII`, `pNBII`, `qNBII` and `rNBII` define the density, distribution function, quantile function and random generation for the Negative Binomial type II, NBII(), distribution.

**Usage**

```
NBII(mu.link = "log", sigma.link = "log")
dNBII(x, mu = 1, sigma = 1, log = FALSE)
pNBII(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qNBII(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rNBII(n, mu = 1, sigma = 1)
```

**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>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive means
<code>sigma</code>	vector of positive dispersion parameter
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$

**Details**

Definition file for Negative Binomial type II distribution.

$$P(Y = y | \mu, \sigma) = \frac{\Gamma(y + \frac{\mu}{\sigma}) \sigma^y}{\Gamma(\frac{\mu}{\sigma}) \Gamma(y + 1) (1 + \sigma)^{y + \mu/\sigma}}$$

for  $y = 0, 1, 2, \dots, \infty$ ,  $\mu > 0$  and  $\sigma > 0$ . This parameterization was used by Evans (1953) and also by Johnson *et al.* (1993) p 200.

**Value**

returns a `gamlss.family` object which can be used to fit a Negative Binomial type II distribution in the `gamlss()` function.

**Note**

$\mu$  is the mean and  $[(1 + \sigma)\mu]^{0.5}$  is the standard deviation of the Negative Binomial type II distribution, so  $\sigma$  is a dispersion parameter

**Author(s)**

Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Calliope Akantzi-Iotou

## References

- Evans, D. A. (1953). Experimental evidence concerning contagious distributions in ecology. *Biometrika*, **40**: 186-211.
- Johnson, N. L., Kotz, S. and Kemp, A. W. (1993). *Univariate Discrete Distributions*, 2nd edn. Wiley, New York.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family](#), [NBI](#), [PIG](#), [SI](#)

## Examples

```
NBII() # gives information about the default links for the Negative Binomial type II distribution
# plotting the distribution
plot(function(y) dNBII(y, mu = 10, sigma = 0.5 ), from=0, to=40, n=40+1, type="h")
# creating random variables and plot them
tN <- table(Ni <- rNBII(1000, mu=5, sigma=0.5))
r <- barplot(tN, col='lightblue')
# library(gamlss)
# data(aids)
# h<-gamlss(y~cs(x,df=7)+qrt, family=NBII, data=aids) # fits a model
# plot(h)
# pdf.plot(family=NBII, mu=10, sigma=0.5, min=0, max=40, step=1)
```

---

NET

*Normal Exponential t distribution (NET) for fitting a GAMLSS*

---

## Description

This function defines the Power Exponential t distribution (NET), a four parameter distribution, for a `gamlss.family` object to be used for a GAMLSS fitting using the function `gamlss()`. The functions `dNET`, `pNET` define the density and distribution function the NET distribution.



**Usage**

```
NET(mu.link = "identity", sigma.link = "log", nu.link = "identity",
    tau.link = "identity")
pNET(q, mu=0, sigma=1, nu=1.5, tau=2, lower.tail = TRUE, log.p = FALSE)
dNET(x, mu=0, sigma=1, nu=1.5, tau=2, log=FALSE)
qNET(p, mu=0, sigma=1, nu=1.5, tau=2, lower.tail = TRUE, log.p = FALSE)
rNET(n, mu=0, sigma=1, nu=1.5, tau=2)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" link as the default for the <code>mu</code> parameter. Other links are "inverse", "log" and "own"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter. Other links are "inverse", "identity" and "own"
<code>nu.link</code>	Defines the <code>nu.link</code> , and because <code>nu</code> is fixed we use "identity" link
<code>tau.link</code>	Defines the <code>tau.link</code> , and because <code>tau</code> is fixed we use "identity" link
<code>x, q</code>	vector of quantiles
<code>p</code>	vector of probabilities
<code>n</code>	number of observations.
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of <code>nu</code> parameter values
<code>tau</code>	vector of <code>tau</code> parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$

**Details**

The NET distribution was introduced by Rigby and Stasinopoulos (1994) as a robust distribution for a response variable with heavier tails than the normal. The NET distribution is the abbreviation of the Normal Exponential Student t distribution. The NET distribution is a four parameter continuous distribution, although in the GAMLSS implementation only the two parameters, `mu` and `sigma`, of the distribution are modelled with `nu` and `tau` fixed. The distribution takes its names because it is normal up to `nu`, Exponential from `nu` to `tau` (hence  $\text{abs}(\text{nu}) \leq \text{abs}(\text{tau})$ ) and Student-t with  $\text{nu} \times \text{tau} - 1$  degrees of freedom after `tau`. Maximum likelihood estimator of the third and fourth parameter can be obtained, using the GAMLSS functions, `find.hyper` or `prof.dev`.

**Value**

`NET()` returns a `gamlss.family` object which can be used to fit a Box Cox Power Exponential distribution in the `gamlss()` function. `dNET()` gives the density, `pNET()` gives the distribution function.

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

**References**

- Rigby, R. A. and Stasinopoulos, D. M. (1994), Robust fitting of an additive model for variance heterogeneity, *COMPSTAT : Proceedings in Computational Statistics*, editors:R. Dutter and W. Grossmann, pp 263-268, Physica, Heidelberg.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family, BCPE](#)

**Examples**

```
NET() #
data(abdom)
plot(function(x)dNET(x, mu=0,sigma=1,nu=2, tau=3), -5, 5)
plot(function(x)pNET(x, mu=0,sigma=1,nu=2, tau=3), -5, 5)
# fit NET with nu=1 and tau=3
# library(gamlss)
#h<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=NET,
#         data=abdom, nu.start=2, tau.start=3)
#plot(h)
```

---

NO

*Normal distribution for fitting a GAMLSS*


---

**Description**

The function `NO()` defines the normal distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`, with mean equal to the parameter `mu` and `sigma` equal the standard deviation. The functions `dNO`, `pNO`, `qNO` and `rNO` define the density, distribution function, quantile function and random generation for the NO parameterization of the normal distribution. [A alternative parameterization with `sigma` equal to the variance is given in the function `NO2()`]

**Usage**

```

NO(mu.link = "identity", sigma.link = "log")
dNO(x, mu = 0, sigma = 1, log = FALSE)
pNO(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qNO(p, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rNO(n, mu = 0, sigma = 1)

```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

The parametrization of the normal distribution given in the function `NO()` is

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

for  $y = (-\infty, \infty)$ ,  $\mu = (-\infty, +\infty)$  and  $\sigma > 0$ .

**Value**

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

**Note**

For the function `NO()`,  $\mu$  is the mean and  $\sigma$  is the standard deviation (not the variance) of the normal distribution.

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

## References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family](#), [NO2](#)

## Examples

```
NO()# gives information about the default links for the normal distribution
plot(function(y) dNO(y, mu=10 ,sigma=2), 0, 20)
plot(function(y) pNO(y, mu=10 ,sigma=2), 0, 20)
plot(function(y) qNO(y, mu=10 ,sigma=2), 0, 1)
dat<-rNO(100)
hist(dat)
# library(gamlss)
# gamlss(dat~1,family=NO) # fits a constant for mu and sigma
```

---

NO2

*Normal distribution (with variance as sigma parameter) for fitting a GAMLSS*

---

## Description

The function NO2() defines the normal distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()` with mean equal to `mu` and variance equal to `sigma`. The functions `dNO2`, `pNO2`, `qNO2` and `rNO2` define the density, distribution function, quantile function and random generation for this specific parameterization of the normal distribution.

[A alternative parameterization with `sigma` as the standard deviation is given in the function `NO()`]

## Usage

```
NO2(mu.link = "identity", sigma.link = "log")
dNO2(x, mu = 0, sigma = 1, log = FALSE)
pNO2(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qNO2(p, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rNO2(n, mu = 0, sigma = 1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

The parametrization of the normal distribution given in the function `NO2()` is

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

for  $y = (-\infty, \infty)$ ,  $\mu = (-\infty, +\infty)$  and  $\sigma > 0$ .

**Value**

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

**Note**

For the function `NO()`,  $\mu$  is the mean and  $\sigma$  is the standard deviation (not the variance) of the normal distribution. [The function `NO2()` defines the normal distribution with  $\sigma$  as the variance.]

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [NO](#)

**Examples**

```
NO()# gives information about the default links for the normal distribution
dat<-rNO(100)
hist(dat)
plot(function(y) dNO(y, mu=10 ,sigma=2), 0, 20)
plot(function(y) pNO(y, mu=10 ,sigma=2), 0, 20)
plot(function(y) qNO(y, mu=10 ,sigma=2), 0, 1)
# library(gamlss)
# gamlss(dat~1,family=NO) # fits a constant for mu and sigma
```

---

NOF

*Normal distribution family for fitting a GAMLSS*


---

**Description**

The function `NOF()` defines a normal distribution family, which has three parameters. The distribution can be used in a GAMLSS fitting using the function `gamlss()`. The mean of NOF is equal to  $\mu$ . The variance is equal to  $\sigma^2 \mu^\nu$  so the standard deviation is  $\sigma \mu^{\nu/2}$ . The function is design for cases where the variance is proportional to a power of the mean. The functions `dNOF`, `pNOF`, `qNOF` and `rNOF` define the density, distribution function, quantile function and random generation for the NOF parametrization of the normal distribution family.

**Usage**

```
NOF(mu.link = "identity", sigma.link = "log", nu.link = "identity")
dNOF(x, mu = 0, sigma = 1, nu = 0, log = FALSE)
pNOF(q, mu = 0, sigma = 1, nu = 0, lower.tail = TRUE, log.p = FALSE)
qNOF(p, mu = 0, sigma = 1, nu = 0, lower.tail = TRUE, log.p = FALSE)
rNOF(n, mu = 0, sigma = 1, nu = 0)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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 "identity" link as the default for the <code>nu</code> parameter
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of power parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$ .
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$

p	vector of probabilities.
n	number of observations. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

### Details

The parametrization of the normal distribution given in the function `NOF()` is

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

for  $y = (-\infty, \infty)$ ,  $\mu = (-\infty, \infty)$ ,  $\sigma > 0$  and  $\nu = (-\infty, +\infty)$ .

### Value

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

### Note

For the function `NOF()`,  $\mu$  is the mean and  $\sigma\mu^{\nu/2}$  is the standard deviation of the normal distribution family. The NOF is design for fitting regression type models where the variance is proportional to a power of the mean. Models of this type are related to the "pseudo likelihood" models of Carroll and Rubert (1987) but here a proper likelihood is maximised.

Note that because the high correlation between the `sigma` and the `nu` parameter the `mixed()` method should be used in the fitting.

### Author(s)

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

### References

- Davidian, M. and Carroll, R. J. (1987), Variance Function Estimation, *Journal of the American Statistical Association*, Vol. **82**, pp. 1079-1091
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [NO](#), [NO2](#)

### Examples

```
NOF()# gives information about the default links for the normal distribution family
# library(gamlss)
#data(abdom)
## the normal distribution fit with constant sigma
#m1<-gamlss(y~poly(x,2), sigma.fo=~1, family=NO, data=abdom)
## the normal family fit with variance proportional to mu
#m2<-gamlss(y~poly(x,2), sigma.fo=~1, family=NOF, data=abdom, method=mixed(1,20))
## a normal distribution fit with variance as a function of x
#m3 <-gamlss(y~poly(x,2), sigma.fo=~x, family=NO, data=abdom, method=mixed(1,20))
#GAIC(m1,m2,m3)
```

---

PARETO2

*Pareto Type 2 distribution for fitting a GAMLSS*


---

### Description

The functions PARETO2() and PARETO2o() define the Pareto Type 2 distribution, a two parameter distribution, for a gamlss.family object to be used in GAMLSS fitting using the function gamlss(). The parameters are mu and sigma in both functions but the parameterisation different. The mu is identical for both PARETO2() and PARETO2o(). The sigma in PARETO2o() is the inverse of the sigma in codePARETO2() and correspond to the usual parameter alpha of the Pareto distribution. The functions dPARETO2, pPARETO2, qPARETO2 and rPARETO2 define the density, distribution function, quantile function and random generation for the PARETO2 parameterization of the Pareto type 2 distribution while the functions dPARETO2o, pPARETO2o, qPARETO2o and rPARETO2o define the density, distribution function, quantile function and random generation for the original PARETO2o parameterization of the Pareto type 2 distribution

### Usage

```
PARETO2(mu.link = "log", sigma.link = "log")
dPARETO2(x, mu = 1, sigma = 0.5, log = FALSE)
pPARETO2(q, mu = 1, sigma = 0.5, lower.tail = TRUE, log.p = FALSE)
qPARETO2(p, mu = 1, sigma = 0.5, lower.tail = TRUE, log.p = FALSE)
rPARETO2(n, mu = 1, sigma = 0.5)
PARETO2o(mu.link = "log", sigma.link = "log")
dPARETO2o(x, mu = 1, sigma = 0.5, log = FALSE)
pPARETO2o(q, mu = 1, sigma = 0.5, lower.tail = TRUE, log.p = FALSE)
qPARETO2o(p, mu = 1, sigma = 0.5, lower.tail = TRUE, log.p = FALSE)
rPARETO2o(n, mu = 1, sigma = 0.5)
```

### Arguments

mu.link	Defines the mu.link, with "" link as the default for the mu parameter
sigma.link	Defines the sigma.link, with "log" as the default for the sigma parameter
x, q	vector of quantiles
mu	vector of location parameter values



<code>sigma</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code>
<code>lower.tail</code>	logical; if TRUE (default), probabilities are <code>P[X &lt;= x]</code> , otherwise <code>P[X &gt; x]</code>
<code>p</code>	vector of probabilities
<code>n</code>	number of observations. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

### Details

The parameterization of the Pareto Type 2 distribution in the function `PA2` is:

$$f(y|\mu, \sigma) = \frac{1}{\sigma} \mu^{\frac{1}{\sigma}} (y + \mu)^{-\frac{1}{\sigma+1}}$$

for  $y \geq 0$ ,  $\mu > 0$  and  $\sigma > 0$ .

### Value

returns a `gamlss.family` object which can be used to fit a Pareto type 2 distribution in the `gamlss()` function.

### Author(s)

Fiona McElduff, Bob Rigby and Mikis Stasinopoulos

### References

- Johnson, N., Kotz, S., and Balakrishnan, N. (1997). *Discrete Multivariate Distributions*. Wiley-Interscience, NY, USA.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#)

### Examples

```

par(mfrow=c(2,2))
y<-seq(0.2,20,0.2)
plot(y, dPARETO2(y), type="l" , lwd=2)
q<-seq(0,20,0.2)
plot(q, pPARETO2(q), ylim=c(0,1), type="l", lwd=2)
p<-seq(0.0001,0.999,0.05)
plot(p, qPARETO2(p), type="l", lwd=2)
dat <- rPARETO2(100)
hist(rPARETO2(100), nclass=30)
#summary(gamlss(a~1, family="PARETO2"))

```

---

 PE

*Power Exponential distribution for fitting a GAMLSS*


---

### Description

The functions define the Power Exponential distribution, a three parameter distribution, for a `gamlss` family object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dPE`, `pPE`, `qPE` and `rPE` define the density, distribution function, quantile function and random generation for the specific parameterization of the power exponential distribution showing below. The functions `dPE2`, `pPE2`, `qPE2` and `rPE2` define the density, distribution function, quantile function and random generation of a standard parameterization of the power exponential distribution.

### Usage

```

PE(mu.link = "identity", sigma.link = "log", nu.link = "log")
dPE(x, mu = 0, sigma = 1, nu = 2, log = FALSE)
pPE(q, mu = 0, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)
qPE(p, mu = 0, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)
rPE(n, mu = 0, sigma = 1, nu = 2)
PE2(mu.link = "identity", sigma.link = "log", nu.link = "log")
dPE2(x, mu = 0, sigma = 1, nu = 2, log = FALSE)
pPE2(q, mu = 0, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)
qPE2(p, mu = 0, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)
rPE2(n, mu = 0, sigma = 1, nu = 2)

```

### Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of kurtosis 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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations. If length(n) > 1, the length is taken to be the number required

### Details

Power Exponential distribution (PE) is defined as

$$f(y|\mu, \sigma, \nu) = \frac{\nu \exp[-(\frac{1}{2})|\frac{z}{c}|^\nu]}{\sigma c 2^{(1+1/\nu)} \Gamma(\frac{1}{\nu})}$$

where  $c = [2^{-2/\nu} \Gamma(1/\nu) / \Gamma(3/\nu)]^{0.5}$ , for  $y = (-\infty, +\infty)$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$  and  $\nu > 0$ . This parametrization was used by Nelson (1991) and ensures  $\mu$  is the mean and  $\sigma$  is the standard deviation of y (for all parameter values of  $\mu$ ,  $\sigma$  and  $\nu$  within the ranges above)

The Power Exponential distribution (PE2) is defined as

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

### Value

returns a `gamlss.family` object which can be used to fit a Power Exponential distribution in the `gamlss()` function.

### Note

$\mu$  is the mean and  $\sigma$  is the standard deviation of the Power Exponential distribution

### Author(s)

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby

### References

- Nelson, D.B. (1991) Conditional heteroskedasticity in asset returns: a new approach. *Econometrica*, **57**, 347-370.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [BCPE](#)

**Examples**

```
PE()# gives information about the default links for the Power Exponential distribution
# library(gamlss)
# data(abdom)
# h1<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=PE, data=abdom) # fit
# h2<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=PE2, data=abdom) # fit
# plot(h1)
# plot(h2)
# leptokurtotic
plot(function(x) dPE(x, mu=10,sigma=2,nu=1), 0.0, 20,
      main = "The PE density mu=10,sigma=2,nu=1")
# platykurtotic
plot(function(x) dPE(x, mu=10,sigma=2,nu=4), 0.0, 20,
      main = "The PE density mu=10,sigma=2,nu=4")
```

---

 PIG

---

*The Poisson-inverse Gaussian distribution for fitting a GAMLSS model*


---

**Description**

The `PIG()` function defines the Poisson-inverse Gaussian distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dPIG`, `pPIG`, `qPIG` and `rPIG` define the density, distribution function, quantile function and random generation for the Poisson-inverse Gaussian `PIG()`, distribution.

The functions `ZAPIG()` and `ZIPIG()` are the zero adjusted (hurdle) and zero inflated versions of the Poisson-inverse Gaussian distribution, respectively. That is three parameter distributions.

The functions `dZAPIG`, `dZIPIG`, `pZAPIG`, `pZIPIG`, `qZAPIG`, `qZIPIG`, `rZAPIG` and `rZIPIG` define the probability, cumulative, quantile and random generation functions for the zero adjusted and zero inflated beta negative binomial distributions, `ZAPIG()`, `ZIPIG()`, respectively.

**Usage**

```
PIG(mu.link = "log", sigma.link = "log")
dPIG(x, mu = 1, sigma = 1, log = FALSE)
pPIG(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qPIG(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE,
     max.value = 10000)
rPIG(n, mu = 1, sigma = 1, max.value = 10000)

ZIPIG(mu.link = "log", sigma.link = "log", nu.link = "logit")
dZIPIG(x, mu = 1, sigma = 1, nu = 0.3, log = FALSE)
pZIPIG(q, mu = 1, sigma = 1, nu = 0.3, lower.tail = TRUE, log.p = FALSE)
qZIPIG(p, mu = 1, sigma = 1, nu = 0.3, lower.tail = TRUE, log.p = FALSE,
```

```

max.value = 10000)
rZIPIG(n, mu = 1, sigma = 1, nu = 0.3, max.value = 10000)

ZAPIG(mu.link = "log", sigma.link = "log", nu.link = "logit")
dZAPIG(x, mu = 1, sigma = 1, nu = 0.3, log = FALSE)
pZAPIG(q, mu = 1, sigma = 1, nu = 0.3, lower.tail = TRUE, log.p = FALSE)
qZAPIG(p, mu = 1, sigma = 1, nu = 0.3, lower.tail = TRUE, log.p = FALSE,
max.value = 10000)
rZAPIG(n, mu = 1, sigma = 1, nu = 0.3, max.value = 10000)

```

### 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
nu.link	Defines the mu.link, with "logit" link as the default for the nu parameter
x	vector of (non-negative integer) quantiles
mu	vector of positive means
sigma	vector of positive dispersion parameter
nu	vector of zero probability parameter
p	vector of probabilities
q	vector of quantiles
n	number of random values to return
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]
max.value	a constant, set to the default value of 10000 for how far the algorithm should look for q

### Details

The probability function of the Poisson-inverse Gaussian distribution, is given by

$$f(y|\mu, \sigma) = \left( \frac{2\alpha^{1/2}}{\pi} \right) \frac{\mu^y e^{\frac{1}{\sigma}} K_{y-\frac{1}{2}}(\alpha)}{(\alpha\sigma)^y y!}$$

where  $\alpha^2 = \frac{1}{\sigma^2} + \frac{2\mu}{\sigma}$ , for  $y = 0, 1, 2, \dots, \infty$  where  $\mu > 0$  and  $\sigma > 0$  and  $K_\lambda(t) = \frac{1}{2} \int_0^\infty x^{\lambda-1} \exp\{-\frac{1}{2}t(x+x^{-1})\} dx$  is the modified Bessel function of the third kind. [Note that the above parameterization was used by Dean, Lawless and Willmot(1989). It is also a special case of the Sichel distribution SI() when  $\nu = -\frac{1}{2}$ .]

### Value

Returns a `gamlss.family` object which can be used to fit a Poisson-inverse Gaussian distribution in the `gamlss()` function.

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Marco Enea

**References**

- Dean, C., Lawless, J. F. and Willmot, G. E., A mixed poisson-inverse-Gaussian regression model, *Canadian J. Statist.*, **17**, 2, pp 171-181
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [NBI](#), [NBII](#), [SI](#), [SICHEL](#)

**Examples**

```
PIG()# gives information about the default links for the Poisson-inverse Gaussian distribution
#plot the pdf using plot
plot(function(y) dPIG(y, mu=10, sigma = 1 ), from=0, to=50, n=50+1, type="h") # pdf
# plot the cdf
plot(seq(from=0,to=50),pPIG(seq(from=0,to=50), mu=10, sigma=1), type="h") # cdf
# generate random sample
tN <- table(Ni <- rPIG(100, mu=5, sigma=1))
r <- barplot(tN, col='lightblue')
# fit a model to the data
# library(gamlss)
# gamlss(Ni~1,family=PIG)
ZIPIG()
ZAPIG()
```

---

PO

*Poisson distribution for fitting a GAMLSS model*

---

**Description**

This function PO defines the Poisson distribution, an one parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dPO`, `pPO`, `qPO` and `rPO` define the density, distribution function, quantile function and random generation for the Poisson, `PO()`, distribution.

**Usage**

```
PO(mu.link = "log")
dPO(x, mu = 1, log = FALSE)
pPO(q, mu = 1, lower.tail = TRUE, log.p = FALSE)
qPO(p, mu = 1, lower.tail = TRUE, log.p = FALSE)
rPO(n, mu = 1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter
<code>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive means
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$

**Details**

Definition file for Poisson distribution.

$$f(y|\mu) = \frac{e^{-\mu} \mu^y}{\Gamma(y+1)}$$

for  $y = 0, 1, 2, \dots$  and  $\mu > 0$ .

**Value**

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

**Note**

$\mu$  is the mean of the Poisson distribution

**Author(s)**

Bob Rigby, Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, and Kalliope Akantziliotou

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [NBI](#), [NBII](#), [SI](#), [SICHEL](#)

### Examples

```
P0()# gives information about the default links for the Poisson distribution
# fitting data using P0()

# plotting the distribution
plot(function(y) dP0(y, mu=10 ), from=0, to=20, n=20+1, type="h")
# creating random variables and plot them
tN <- table(Ni <- rP0(1000, mu=5))
r <- barplot(tN, col='lightblue')
# library(gamlss)
# data(aids)
# h<-gamlss(y~cs(x,df=7)+qrt, family=P0, data=aids) # fits the constant+x+qrt model
# plot(h)
# pdf.plot(family=P0, mu=10, min=0, max=20, step=1)
```

---

 RG

*The Reverse Gumbel distribution for fitting a GAMLSS*

---

### Description

The function RG defines the reverse Gumbel distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dRG`, `pRG`, `qRG` and `rRG` define the density, distribution function, quantile function and random generation for the specific parameterization of the reverse Gumbel distribution.

### Usage

```
RG(mu.link = "identity", sigma.link = "log")
dRG(x, mu = 0, sigma = 1, log = FALSE)
pRG(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qRG(p, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rRG(n, mu = 0, sigma = 1)
```



**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" link as the default for the <code>mu</code> parameter. other available link is "inverse", "log" and "own"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter, other links are the "inverse", "identity" and "own"
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

The specific parameterization of the reverse Gumbel distribution used in RG is

$$f(y|\mu, \sigma) = \frac{1}{\sigma} \exp \left\{ - \left( \frac{y - \mu}{\sigma} \right) - \exp \left[ - \frac{(y - \mu)}{\sigma} \right] \right\}$$

for  $y = (-\infty, \infty)$ ,  $\mu = (-\infty, +\infty)$  and  $\sigma > 0$ .

**Value**

`RG()` returns a `gamlss.family` object which can be used to fit a Gumbel distribution in the `gamlss()` function. `dRG()` gives the density, `pGU()` gives the distribution function, `qRG()` gives the quantile function, and `rRG()` generates random deviates.

**Note**

The mean of the distribution is  $\mu + 0.57722\sigma$  and the variance is  $\pi^2\sigma^2/6$ .

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby and Calliope Akantziliotou

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M., Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#)

### Examples

```
plot(function(x) dRG(x, mu=0,sigma=1), -3, 6,
      main = "{Reverse Gumbel density mu=0,sigma=1}")
RG()# gives information about the default links for the Gumbel distribution
dat<-rRG(100, mu=10, sigma=2) # generates 100 random observations
# library(gamlss)
# gamlss(dat~1,family=RG) # fits a constant for each parameter mu and sigma
```

---

RGE

*Reverse generalized extreme family distribution for fitting a GAMLSS*

---

### Description

The function RGE defines the reverse generalized extreme family distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dRGE`, `pRGE`, `qRGE` and `rRGE` define the density, distribution function, quantile function and random generation for the specific parameterization of the reverse generalized extreme distribution given in details below.

### Usage

```
RGE(mu.link = "identity", sigma.link = "log", nu.link = "log")
dRGE(x, mu = 1, sigma = 0.1, nu = 1, log = FALSE)
pRGE(q, mu = 1, sigma = 0.1, nu = 1, lower.tail = TRUE, log.p = FALSE)
qRGE(p, mu = 1, sigma = 0.1, nu = 1, lower.tail = TRUE, log.p = FALSE)
rRGE(n, mu = 1, sigma = 0.1, nu = 1)
```

### Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values

sigma	vector of scale parameter values
nu	vector of the shape parameter values
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required

### Details

Definition file for reverse generalized extreme family distribution.

The probability density function of the generalized extreme value distribution is obtained from Johnson *et al.* (1995), Volume 2, p76, equation (22.184) [where  $(\xi, \theta, \gamma) \rightarrow (\mu, \sigma, \nu)$ ].

The probability density function of the reverse generalized extreme value distribution is then obtained by replacing  $y$  by  $-y$  and  $\mu$  by  $-\mu$ .

Hence the probability density function of the reverse generalized extreme value distribution with  $\nu > 0$  is given by

$$f(y|\mu, \sigma, \nu) = \frac{1}{\sigma} \left[ 1 + \frac{\nu(y - \mu)}{\sigma} \right]^{\frac{1}{\nu} - 1} S_1(y|\mu, \sigma, \nu)$$

for

$$\mu - \frac{\sigma}{\nu} < y < \infty$$

where

$$S_1(y|\mu, \sigma, \nu) = \exp \left\{ - \left[ 1 + \frac{\nu(y - \mu)}{\sigma} \right]^{\frac{1}{\nu}} \right\}$$

and where  $-\infty < \mu < y + \frac{\sigma}{\nu}$ ,  $\sigma > 0$  and  $\nu > 0$ . Note that only the case  $\nu > 0$  is allowed here. The reverse generalized extreme value distribution is denoted as RGE( $\mu, \sigma, \nu$ ) or as Reverse Generalized.Extreme.Family( $\mu, \sigma, \nu$ ).

Note the the above distribution is a reparameterization of the three parameter Weibull distribution given by

$$f(y|\alpha_1, \alpha_2, \alpha_3) = \frac{\alpha_3}{\alpha_2} \left[ \frac{y - \alpha_1}{\alpha_2} \right]^{\alpha_3 - 1} \exp \left[ - \left( \frac{y - \alpha_1}{\alpha_2} \right)^{\alpha_3} \right]$$

given by setting  $\alpha_1 = \mu - \sigma/\nu$ ,  $\alpha_2 = \sigma/\nu$ ,  $\alpha_3 = 1/\nu$ .

### Value

RGE() returns a gamlss.family object which can be used to fit a reverse generalized extreme distribution in the gamlss() function. dRGE() gives the density, pRGE() gives the distribution function, qRGE() gives the quantile function, and rRGE() generates random deviates.

**Note**

This distribution is very difficult to fit because the y values depends on the parameter values. The RS() and CG() algorithms are not appropriate for this type of problem.

**Author(s)**

Bob Rigby, Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org> and Kalliope Akantziliotou

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#)

**Examples**

```
RGE()# default links for the reverse generalized extreme family distribution
newdata<-rRGE(100,mu=0,sigma=1,nu=5) # generates 100 random observations
# library(gamlss)
# gamlss(newdata~1, family=RGE, method=mixed(5,50)) # difficult to converse
```

---

 SEP

---

*The Skew Power exponential (SEP) distribution for fitting a GAMLSS*


---

**Description**

This function defines the Skew Power exponential (SEP) distribution, a four parameter distribution, for a `gamlss.family` object to be used for a GAMLSS fitting using the function `gamlss()`. The functions `dSEP`, `pSEP`, `qSEP` and `rSEP` define the density, distribution function, quantile function and random generation for the Skew Power exponential (SEP) distribution.

**Usage**

```
SEP(mu.link = "identity", sigma.link = "log", nu.link = "identity",
    tau.link = "log")
dSEP(x, mu = 0, sigma = 1, nu = 0, tau = 2, log = FALSE)
pSEP(q, mu = 0, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE,
     log.p = FALSE)
qSEP(p, mu = 0, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE,
     log.p = FALSE, lower.limit = mu - 5 * sigma,
     upper.limit = mu + 5 * sigma)
rSEP(n, mu = 0, sigma = 1, nu = 0, tau = 2)
```

**Arguments**

mu.link	Defines the mu.link, with "identity" link as the default for the mu parameter. Other links are "1/mu <sup>2</sup> " and "log"
sigma.link	Defines the sigma.link, with "log" link as the default for the sigma parameter. Other links are "inverse" and "identity"
nu.link	Defines the nu.link, with "identity" link as the default for the nu parameter. Other links are "1/nu <sup>2</sup> " and "log"
tau.link	Defines the tau.link, with "log" link as the default for the tau parameter. Other links are "1/tau <sup>2</sup> ", and "identity"
x, q	vector of quantiles
mu	vector of location parameter values
sigma	vector of scale parameter values
nu	vector of skewness nu parameter values
tau	vector of kurtosis tau parameter values
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required
lower.limit	lower limit for the golden search to find quantiles from probabilities
upper.limit	upper limit for the golden search to find quantiles from probabilities

**Details**

The probability density function of the Skew Power exponential distribution, (SEP), is defined as

$$f(y|n, \mu, \sigma, \nu, \tau) == \frac{z}{\sigma} \Phi(\omega) f_{EP}(z, 0, 1, \tau)$$

for  $-\infty < y < \infty$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$ ,  $\nu = (-\infty, +\infty)$  and  $\tau > 0$ . where  $z = \frac{y-\mu}{\sigma}$ ,  $\omega = \text{sign}(z)|z|^{\tau/2}\nu\sqrt{2/\tau}$  and  $f_{EP}(z, 0, 1, \tau)$  is the pdf of an Exponential Power distribution.

**Value**

SEP() returns a `gamlss.family` object which can be used to fit the SEP distribution in the `gamlss()` function. `dSEP()` gives the density, `pSEP()` gives the distribution function, `qSEP()` gives the quantile function, and `rSEP()` generates random deviates.

**Warning**

The `qSEP` and `rSEP` are slow since they are relying on golden section for finding the quantiles

**Author(s)**

Bob Rigby and Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>

**References**

- Diciccio, T. J. and Monti A. C. (2004). Inferential Aspects of the Skew Exponential Power distribution., *JASA*, **99**, 439-450.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [JSU](#), [BCT](#)

**Examples**

```
SEP() #
plot(function(x)dSEP(x, mu=0,sigma=1, nu=1, tau=2), -5, 5,
      main = "The SEP density mu=0,sigma=1,nu=1, tau=2")
plot(function(x) pSEP(x, mu=0,sigma=1,nu=1, tau=2), -5, 5,
      main = "The BCPE cdf mu=0, sigma=1, nu=1, tau=2")
dat <- rSEP(100,mu=10,sigma=1,nu=-1,tau=1.5)
# library(gamlss)
# gamlss(dat~1,family=SEP, control=gamlss.control(n.cyc=30))
```

---

SEPI	<i>The Skew Power exponential type 1-4 distribution for fitting a GAMLSS</i>
------	--

---

### Description

These functions define the Skew Power exponential type 1 to 4 distributions. All of them are four parameter distributions and can be used to fit a GAMLSS model. The functions dSEP1, dSEP2, dSEP3 and dSEP4 define the probability distribution functions, the functions pSEP1, pSEP2, pSEP3 and pSEP4 define the cumulative distribution functions the functions qSEP1, qSEP2, qSEP3 and qSEP4 define the inverse cumulative distribution functions and the functions rSEP1, rSEP2, rSEP3 and rSEP4 define the random generation for the Skew exponential power distributions.

### Usage

```
SEP1(mu.link = "identity", sigma.link = "log", nu.link = "identity",
     tau.link = "log")
dSEP1(x, mu = 0, sigma = 1, nu = 0, tau = 2, log = FALSE)
pSEP1(q, mu = 0, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE,
     log.p = FALSE)
qSEP1(p, mu = 0, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE,
     log.p = FALSE)
rSEP1(n, mu = 0, sigma = 1, nu = 0, tau = 2)

SEP2(mu.link = "identity", sigma.link = "log", nu.link = "identity",
     tau.link = "log")
dSEP2(x, mu = 0, sigma = 1, nu = 0, tau = 2, log = FALSE)
pSEP2(q, mu = 0, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE,
     log.p = FALSE)
qSEP2(p, mu = 0, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE,
     log.p = FALSE)
rSEP2(n, mu = 0, sigma = 1, nu = 0, tau = 2)

SEP3(mu.link = "identity", sigma.link = "log", nu.link = "log",
     tau.link = "log")
dSEP3(x, mu = 0, sigma = 1, nu = 2, tau = 2, log = FALSE)
pSEP3(q, mu = 0, sigma = 1, nu = 2, tau = 2, lower.tail = TRUE,
     log.p = FALSE)
qSEP3(p, mu = 0, sigma = 1, nu = 2, tau = 2, lower.tail = TRUE,
     log.p = FALSE)

SEP4(mu.link = "identity", sigma.link = "log", nu.link = "log",
     tau.link = "log")
dSEP4(x, mu = 0, sigma = 1, nu = 2, tau = 2, log = FALSE)
pSEP4(q, mu = 0, sigma = 1, nu = 2, tau = 2, lower.tail = TRUE,
     log.p = FALSE)
qSEP4(p, mu = 0, sigma = 1, nu = 2, tau = 2, lower.tail = TRUE,
```

```
log.p = FALSE)
rSEP4(n, mu = 0, sigma = 1, nu = 2, tau = 2)
```

### Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" link as the default for the <code>mu</code> parameter. Other links are "inverse" and "log"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the <code>sigma</code> parameter. Other links are "inverse" and "identity"
<code>nu.link</code>	Defines the <code>nu.link</code> , with "log" link as the default for the <code>nu</code> parameter. Other links are "identity" and "inverse"
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the <code>tau</code> parameter. Other links are "inverse", and "identity"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of skewness <code>nu</code> parameter values
<code>tau</code>	vector of kurtosis <code>tau</code> parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

### Details

The probability density function of the Skew Power exponential distribution type 2, (SEP2), is defined as

$$f_Y(y|\mu, \sigma, \nu, \tau) = \frac{\nu}{\sigma(1 + \nu^2)^{1/\tau} \Gamma(1 + 1/\tau)} \left\{ \exp\left(-\frac{1}{2} \left| \frac{\nu(y - \mu)}{\sigma} \right|^\tau\right) I(y < \mu) + \exp\left(-\frac{1}{2} \left| \frac{(y - \mu)}{\sigma\nu} \right|^\tau\right) I(y \geq \mu) \right\}$$

for  $-\infty < y < \infty$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$ ,  $\nu > 0$  and  $\tau > 0$ .

### Value

`SEP2()` returns a `gamlss.family` object which can be used to fit the SEP2 distribution in the `gamlss()` function. `dSEP2()` gives the density, `pSEP2()` gives the distribution function, `qSEP2()` gives the quantile function, and `rSEP2()` generates random deviates.

### Author(s)

Bob Rigby and Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>



## References

- Fernandez C., Osiewalski J. and Steel M.F.J.(1995) Modelling and inference with v-spherical distributions. *JASA*, **90**, pp 1331-1340.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family](#), [SEP](#)

## Examples

```
SEP1()
curve(dSEP4(x, mu=5 ,sigma=1, nu=2, tau=1.5), -2, 10,
      main = "The SEP4 density mu=5 ,sigma=1, nu=1, tau=1.5")
# library(gamlss)
#y<- rSEP4(100, mu=5, sigma=1, nu=2, tau=1.5);hist(y)
#m1<-gamlss(y~1, family=SEP1, n.cyc=50)
#m2<-gamlss(y~1, family=SEP2, n.cyc=50)
#m3<-gamlss(y~1, family=SEP3, n.cyc=50)
#m4<-gamlss(y~1, family=SEP4, n.cyc=50)
#GAIC(m1,m2,m3,m4)
```

## Description

The Sinh-Arcsinh (SHASH) distribution is a four parameter distribution, for a `gamlss.family` object to be used for a GAMLSS fitting using the function `gamlss()`. The functions `dSHASH`, `pSHASH`, `qSHASH` and `rSHASH` define the density, distribution function, quantile function and random generation for the Sinh-Arcsinh (SHASH) distribution.

There are 3 different SHASH distributions implemented in GAMLSS.

**Usage**

```

SHASH(mu.link = "identity", sigma.link = "log", nu.link = "log",
      tau.link = "log")
dSHASH(x, mu = 0, sigma = 1, nu = 0.5, tau = 0.5, log = FALSE)
pSHASH(q, mu = 0, sigma = 1, nu = 0.5, tau = 0.5, lower.tail = TRUE,
      log.p = FALSE)
qSHASH(p, mu = 0, sigma = 1, nu = 0.5, tau = 0.5, lower.tail = TRUE,
      log.p = FALSE)
rSHASH(n, mu = 0, sigma = 1, nu = 0.5, tau = 0.5)

SHASHo(mu.link = "identity", sigma.link = "log", nu.link = "identity",
      tau.link = "log")
dSHASHo(x, mu = 0, sigma = 1, nu = 0, tau = 1, log = FALSE)
pSHASHo(q, mu = 0, sigma = 1, nu = 0, tau = 1, lower.tail = TRUE,
      log.p = FALSE)
qSHASHo(p, mu = 0, sigma = 1, nu = 0, tau = 1, lower.tail = TRUE,
      log.p = FALSE)
rSHASHo(n, mu = 0, sigma = 1, nu = 0, tau = 1)

SHASHo2(mu.link = "identity", sigma.link = "log", nu.link = "identity",
      tau.link = "log")
dSHASHo2(x, mu = 0, sigma = 1, nu = 0, tau = 1, log = FALSE)
pSHASHo2(q, mu = 0, sigma = 1, nu = 0, tau = 1, lower.tail = TRUE,
      log.p = FALSE)
qSHASHo2(p, mu = 0, sigma = 1, nu = 0, tau = 1, lower.tail = TRUE,
      log.p = FALSE)
rSHASHo2(n, mu = 0, sigma = 1, nu = 0, tau = 1)

```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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.
<code>tau.link</code>	Defines the <code>tau.link</code> , with "log" link as the default for the <code>tau</code> parameter.
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of skewness <code>nu</code> parameter values
<code>tau</code>	vector of kurtosis <code>tau</code> parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

The probability density function of the Sinh-Arcsinh distribution, (SHASH), Jones(2005), is defined as

$$f(y|\mu, \sigma, \nu, \tau) = \frac{c}{\sqrt{2\pi}\sigma(1+z^2)^{1/2}} e^{-r^2/2}$$

where

$$r = \frac{1}{2} \{ \exp [\tau \sinh^{-1}(z)] - \exp [-\nu \sinh^{-1}(z)] \}$$

and

$$c = \frac{1}{2} \{ \tau \exp [\tau \sinh^{-1}(z)] + \nu \exp [-\nu \sinh^{-1}(z)] \}$$

and  $z = (y - \mu)/\sigma$  for  $-\infty < y < \infty$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$ ,  $\nu > 0$  and  $\tau > 0$ .

The parameters  $\mu$  and  $\sigma$  are the location and scale of the distribution. The parameter  $\nu$  determines the left hand tail of the distribution with  $\nu > 1$  indicating a lighter tail than the normal and  $\nu < 1$  heavier tail than the normal. The parameter  $\tau$  determines the right hand tail of the distribution in the same way.

The second form of the Sinh-Arcsinh distribution can be found in Jones and Pewsey (2009, p.2) denoted by SHASHo and the probability density function is defined as,

$$f(y|\mu, \sigma, \nu, \tau) = \frac{\tau}{\sigma} \frac{c}{\sqrt{2\pi}} \frac{1}{2\sqrt{1+z^2}} \exp\left(-\frac{r^2}{2}\right)$$

where

$$r = \sinh(\tau \arcsin(z) - \nu)$$

and

$$c = \cosh(\tau \arcsin(z) - \nu)$$

and  $z = (y - \mu)/\sigma$  for  $-\infty < y < \infty$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$ ,  $\nu = (-\infty, +\infty)$  and  $\tau > 0$ .

The third form of the Sinh-Arcsinh distribution (Jones and Pewsey, 2009, p.8) divides the distribution by sigma for the density of the unstandardized variable. This distribution is denoted by SHASHo2 and has pdf

$$f(y|\mu, \sigma, \nu, \tau) = \frac{c}{\sigma} \frac{\tau}{\sqrt{2\pi}} \frac{1}{\sqrt{1+z^2}} - \exp -\frac{r^2}{2}$$

where  $z = (y - \mu)/(\sigma\tau)$ , with  $r$  and  $c$  as for the pdf of the SHASHo distribution, for  $-\infty < y < \infty$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$ ,  $\nu = (-\infty, +\infty)$  and  $\tau > 0$ .

**Value**

SHASH() returns a `gamlss.family` object which can be used to fit the SHASH distribution in the `gamlss()` function. `dSHASH()` gives the density, `pSHASH()` gives the distribution function, `qSHASH()` gives the quantile function, and `rSHASH()` generates random deviates.

**Warning**

The `qSHASH` and `rSHASH` are slow since they are relying on golden section for finding the quantiles

**Author(s)**

Bob Rigby, Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)> and Fiona McElduff

**References**

- Jones, M. C. (2006) p 546-547 in the discussion of Rigby, R. A. and Stasinopoulos D. M. (2005) *Appl. Statist.*, **54**, part 3.
- Jones and Pewsey (2009) Sinh-arcsinh distributions. *Biometrika*. **96**(4), pp. 761-780.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [JSU](#), [BCT](#)

**Examples**

```
SHASH() #
plot(function(x)dSHASH(x, mu=0,sigma=1, nu=1, tau=2), -5, 5,
      main = "The SHASH density mu=0,sigma=1,nu=1, tau=2")
plot(function(x) pSHASH(x, mu=0,sigma=1,nu=1, tau=2), -5, 5,
      main = "The BCPE cdf mu=0, sigma=1, nu=1, tau=2")
dat<-rSHASH(100,mu=10,sigma=1,nu=1,tau=1.5)
hist(dat)
# library(gamlss)
# gamlss(dat~1,family=SHASH, control=gamlss.control(n.cyc=30))
```

**Description**

The `SI()` function defines the Sichel distribution, a three parameter discrete distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dSI`, `pSI`, `qSI` and `rSI` define the density, distribution function, quantile function and random generation for the Sichel `SI()`, distribution.

**Usage**

```
SI(mu.link = "log", sigma.link = "log", nu.link = "identity")
dSI(x, mu = 0.5, sigma = 0.02, nu = -0.5, log = FALSE)
pSI(q, mu = 0.5, sigma = 0.02, nu = -0.5, lower.tail = TRUE,
    log.p = FALSE)
qSI(p, mu = 0.5, sigma = 0.02, nu = -0.5, lower.tail = TRUE,
    log.p = FALSE, max.value = 10000)
rSI(n, mu = 0.5, sigma = 0.02, nu = -0.5)
tofyS(y, mu, sigma, nu, what = 1)
```

**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 parameter
<code>nu.link</code>	Defines the <code>nu.link</code> , with "identity" link as the default for the nu parameter
<code>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive mu
<code>sigma</code>	vector of positive dispersion parameter
<code>nu</code>	vector of nu
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<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>max.value</code>	a constant, set to the default value of 10000 for how far the algorithm should look for q
<code>y</code>	the y variable. The function <code>tofyS()</code> should be not used on its own.
<code>what</code>	take values 1 or 2, for function <code>tofyS()</code> .

## Details

The probability function of the Sichel distribution is given by

$$f(y|\mu, \sigma, \nu) = \frac{\mu^y K_{y+\nu}(\alpha)}{(\alpha\sigma)^{y+\nu} y! K_\nu(\frac{1}{\sigma})}$$

where  $\alpha^2 = \frac{1}{\sigma^2} + \frac{2\mu}{\sigma}$ , for  $y = 0, 1, 2, \dots, \infty$  where  $\mu > 0$ ,  $\sigma > 0$  and  $-\infty < \nu < \infty$  and  $K_\lambda(t) = \frac{1}{2} \int_0^\infty x^{\lambda-1} \exp\{-\frac{1}{2}t(x + x^{-1})\} dx$  is the modified Bessel function of the third kind. Note that the above parameterization is different from Stein, Zucchini and Juritz (1988) who use the above probability function but treat  $\mu$ ,  $\alpha$  and  $\nu$  as the parameters. Note that  $\sigma = [(\mu^2 + \alpha^2)^{\frac{1}{2}} - \mu]^{-1}$ .

## Value

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

## Author(s)

Akantziliotou C., Rigby, R. A., Stasinopoulos D. M. and Marco Enea

## References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape, (with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2003) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.
- Stein, G. Z., Zucchini, W. and Juritz, J. M. (1987). Parameter Estimation of the Sichel Distribution and its Multivariate Extension. *Journal of American Statistical Association*, **82**, 938-944.

## See Also

[gamlss.family](#), [PIG](#), [NBI](#), [NBII](#)

## Examples

```
SI()# gives information about the default links for the Sichel distribution
#plot the pdf using plot
plot(function(y) dSI(y, mu=10, sigma=1, nu=1), from=0, to=100, n=100+1, type="h") # pdf
# plot the cdf
plot(seq(from=0,to=100),pSI(seq(from=0,to=100), mu=10, sigma=1, nu=1), type="h") # cdf
# generate random sample
tN <- table(Ni <- rSI(100, mu=5, sigma=1, nu=1))
r <- barplot(tN, col='lightblue')
```

```
# fit a model to the data
# library(gamlss)
# gamlss(Ni~1,family=SI, control=gamlss.control(n.cyc=50))
```

SICHEL

*The Sichel distribution for fitting a GAMLSS model***Description**

The SICHEL() function defines the Sichel distribution, a three parameter discrete distribution, for a gamlss.family object to be used in GAMLSS fitting using the function gamlss(). The functions dSICHEL, pSICHEL, qSICHEL and rSICHEL define the density, distribution function, quantile function and random generation for the Sichel SICHEL(), distribution. The function VSICHEL gives the variance of a fitted Sichel model.

The functions ZASICHEL() and ZISICHEL() are the zero adjusted (hurdle) and zero inflated versions of the Sichel distribution, respectively. That is four parameter distributions.

The functions dZASICHEL, dZISICHEL, pZASICHEL, pZISICHEL, qZASICHEL, qZISICHEL, rZASICHEL and rZISICHEL define the probability, cumulative, quantile and random generation functions for the zero adjusted and zero inflated Sichel distributions, ZASICHEL(), ZISICHEL(), respectively.

**Usage**

```
SICHEL(mu.link = "log", sigma.link = "log", nu.link = "identity")
dSICHEL(x, mu=1, sigma=1, nu=-0.5, log=FALSE)
pSICHEL(q, mu=1, sigma=1, nu=-0.5, lower.tail = TRUE,
        log.p = FALSE)
qSICHEL(p, mu=1, sigma=1, nu=-0.5, lower.tail = TRUE,
        log.p = FALSE, max.value = 10000)
rSICHEL(n, mu=1, sigma=1, nu=-0.5, max.value = 10000)
VSICHEL(obj)
tofySICHEL(y, mu, sigma, nu)

ZASICHEL(mu.link = "log", sigma.link = "log", nu.link = "identity",
        tau.link = "logit")
dZASICHEL(x, mu = 1, sigma = 1, nu = -0.5, tau = 0.1, log = FALSE)
pZASICHEL(q, mu = 1, sigma = 1, nu = -0.5, tau = 0.1,
        lower.tail = TRUE, log.p = FALSE)
qZASICHEL(p, mu = 1, sigma = 1, nu = -0.5, tau = 0.1,
        lower.tail = TRUE, log.p = FALSE, max.value = 10000)
rZASICHEL(n, mu = 1, sigma = 1, nu = -0.5, tau = 0.1,
        max.value = 10000)

ZISICHEL(mu.link = "log", sigma.link = "log", nu.link = "identity",
        tau.link = "logit")
dZISICHEL(x, mu = 1, sigma = 1, nu = -0.5, tau = 0.1, log = FALSE)
pZISICHEL(q, mu = 1, sigma = 1, nu = -0.5, tau = 0.1,
        lower.tail = TRUE, log.p = FALSE)
```

```

qZISICHEL(p, mu = 1, sigma = 1, nu = -0.5, tau = 0.1,
          lower.tail = TRUE, log.p = FALSE, max.value = 10000)
rZISICHEL(n, mu = 1, sigma = 1, nu = -0.5, tau = 0.1,
          max.value = 10000)

```

### 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 "identity" link as the default for the <code>nu</code> parameter
<code>tau.link</code>	Defines the <code>tau.link</code> , with "logit" link as the default for the <code>tau</code> parameter
<code>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive <code>mu</code>
<code>sigma</code>	vector of positive dispersion parameter <code>sigma</code>
<code>nu</code>	vector of <code>nu</code>
<code>tau</code>	vector of probabilities <code>tau</code>
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$
<code>max.value</code>	a constant, set to the default value of 10000 for how far the algorithm should look for <code>q</code>
<code>obj</code>	a fitted Sichel <code>gamlss</code> model
<code>y</code>	the <code>y</code> variable, the <code>tofySICHEL()</code> should not be used on its own.

### Details

The probability function of the Sichel distribution is given by

$$f(y|\mu, \sigma, \nu) = \frac{\mu^y K_{y+\nu}(\alpha)}{y!(\alpha\sigma)^{y+\nu} K_\nu(\frac{1}{\sigma})}$$

for  $y = 0, 1, 2, \dots, \infty$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $-\infty < \nu < \infty$  where

$$\alpha^2 = \frac{1}{\sigma^2} + \frac{2\mu}{\sigma}$$

$$c = K_{\nu+1}(1/\sigma)/K_\nu(1/\sigma)$$

and  $K_\lambda(t)$  is the modified Bessel function of the third kind. Note that the above parametrization is different from Stein, Zucchini and Juritz (1988) who use the above probability function but treat  $\mu$ ,  $\alpha$  and  $\nu$  as the parameters.



**Value**

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

**Note**

The mean of the above Sichel distribution is  $\mu$  and the variance is  $\mu^2 \left[ \frac{2\sigma(\nu+1)}{c} + \frac{1}{c^2} - 1 \right]$

**Author(s)**

Rigby, R. A., Stasinopoulos D. M., Akantziliotou C and Marco Enea.

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Rigby, R. A., Stasinopoulos D. M. and Akantziliotou, C. (2006) Modelling the parameters of a family of mixed Poisson distributions including the Sichel and Delaptorte. Submitted for publication.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2003) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.
- Stein, G. Z., Zucchini, W. and Juritz, J. M. (1987). Parameter Estimation of the Sichel Distribution and its Multivariate Extension. *Journal of American Statistical Association*, **82**, 938-944.

**See Also**

[gamlss.family](#), [PIG](#), [SI](#)

**Examples**

```
SICHEL()# gives information about the default links for the Sichel distribution
#plot the pdf using plot
plot(function(y) dSICHEL(y, mu=10, sigma=1, nu=1), from=0, to=100, n=100+1, type="h") # pdf
# plot the cdf
plot(seq(from=0,to=100),pSICHEL(seq(from=0,to=100), mu=10, sigma=1, nu=1), type="h") # cdf
# generate random sample
tN <- table(Ni <- rSICHEL(100, mu=5, sigma=1, nu=1))
r <- barplot(tN, col='lightblue')
# fit a model to the data
# library(gamlss)
# gamlss(Ni~1,family=SICHEL, control=gamlss.control(n.cyc=50))
```

SIMPLEX

*The simplex distribution for fitting a GAMLSS***Description**

The functions SIMPLEX() define the simplex distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. SIMPLEX() has mean equal to the parameter `mu` and `sigma` as scale parameter, see below. The functions `dSIMPLEX`, `pSIMPLEX`, `qSIMPLEX` and `rSIMPLEX` define the density, cumulative distribution function, quantile function and random generation for the simplex distribution.

**Usage**

```
SIMPLEX(mu.link = "logit", sigma.link = "log")
dSIMPLEX(x, mu = 0.5, sigma = 1, log = FALSE)
pSIMPLEX(q, mu = 0.5, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qSIMPLEX(p, mu = 0.5, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rSIMPLEX(n = 1, mu = 0.5, sigma = 1)
```

**Arguments**

<code>mu.link</code>	the mu link function with default <code>logit</code>
<code>sigma.link</code>	the sigma link function with default <code>log</code>
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code> .
<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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

**Details**

The simplex distribution is given as

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

for  $y$  in  $(0,1)$ ,  $0 < \mu < 1$  and  $\sigma > 0$ .

**Value**

SIMPLEX() returns a `gamlss.family` object which can be used to fit a simplex distribution in the `gamlss()` function.

**Author(s)**

Bob Rigby, Mikis Stasinopoulos and Fernanda De Bastiani

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**Examples**

```
SIMPLEX()# default links for the simplex distribution
plot(function(y) dSIMPLEX(y, mu=.5 ,sigma=1), 0.001, .999)
plot(function(y) pSIMPLEX(y, mu=.5 ,sigma=1), 0.001, 0.999)
plot(function(y) qSIMPLEX(y, mu=.5 ,sigma=1), 0.001, 0.999)
plot(function(y) qSIMPLEX(y, mu=.5 ,sigma=1, lower.tail=FALSE), 0.001, .999)
```

---

 SN1

---

*Skew Normal Type 1 distribution for fitting a GAMLSS*


---

**Description**

The function SN1() defines the Skew Normal Type 1 distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`, with parameters `mu`, `sigma` and `nu`. The functions `dSN1`, `pSN1`, `qSN1` and `rSN1` define the density, distribution function, quantile function and random generation for the SN1 parameterization of the Skew Normal Type 1 distribution.

**Usage**

```
SN1(mu.link = "identity", sigma.link = "log", nu.link="identity")
dSN1(x, mu = 0, sigma = 1, nu = 0, log = FALSE)
pSN1(q, mu = 0, sigma = 1, nu = 0, lower.tail = TRUE, log.p = FALSE)
qSN1(p, mu = 0, sigma = 1, nu = 0, lower.tail = TRUE, log.p = FALSE)
rSN1(n, mu = 0, sigma = 1, nu = 0)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" links the default for the <code>mu</code> parameter
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" as the default for the <code>sigma</code> parameter
<code>nu.link</code>	Defines the <code>nu.link</code> , with "identity" as the default for the <code>nu</code> parameter

<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of scale parameter values
<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. If $\text{length}(n) > 1$ , the length is taken to be the number required

### Details

The parameterization of the Skew Normal Type 1 distribution in the function SN1 is ...

### Value

returns a `gamlss.family` object which can be used to fit a Skew Normal Type 1 distribution in the `gamlss()` function.

### Note

This is a special case of the Skew Exponential Power type 1 distribution (SEP1) where  $\tau=2$ .

### Author(s)

Mikis Stasinopoulos, Bob Rigby and Fiona McElduff

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape, (with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#)

**Examples**

```

par(mfrow=c(2,2))
y<-seq(-3,3,0.2)
plot(y, dSN1(y), type="l" , lwd=2)
q<-seq(-3,3,0.2)
plot(q, pSN1(q), ylim=c(0,1), type="l", lwd=2)
p<-seq(0.0001,0.999,0.05)
plot(p, qSN1(p), type="l", lwd=2)
dat <- rSN1(100)
hist(rSN1(100), nclass=30)

```

SN2

*Skew Normal Type 2 distribution for fitting a GAMLSS***Description**

The function `SN2()` defines the Skew Normal Type 2 distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`, with parameters `mu`, `sigma` and `nu`. The functions `dSN2`, `pSN2`, `qSN2` and `rSN2` define the density, distribution function, quantile function and random generation for the SN2 parameterization of the Skew Normal Type 2 distribution.

**Usage**

```

SN2(mu.link = "identity", sigma.link = "log", nu.link = "log")
dSN2(x, mu = 0, sigma = 1, nu = 2, log = FALSE)
pSN2(q, mu = 0, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)
qSN2(p, mu = 0, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)
rSN2(n, mu = 0, sigma = 1, nu = 2)

```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" links the default for the <code>mu</code> parameter
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" as the default for the <code>sigma</code> parameter
<code>nu.link</code>	Defines the <code>nu.link</code> , with "log" as the default for the <code>nu</code> parameter
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of scale parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

**Details**

The parameterization of the Skew Normal Type 2 distribution in the function SN2 is ...

**Value**

returns a `gamlss.family` object which can be used to fit a Skew Normal Type 2 distribution in the `gamlss()` function.

**Note**

This is a special case of the Skew Exponential Power type 3 distribution (SEP3) where  $\tau=2$ .

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Fiona McElduff.

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape, (with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#)

**Examples**

```
par(mfrow=c(2,2))
y<-seq(-3,3,0.2)
plot(y, dSN2(y), type="l" , lwd=2)
q<-seq(-3,3,0.2)
plot(q, pSN2(q), ylim=c(0,1), type="l", lwd=2)
p<-seq(0.0001,0.999,0.05)
plot(p, qSN2(p), type="l", lwd=2)
dat <- rSN2(100)
hist(rSN2(100), nclass=30)
```

**Description**

There are 5 different skew t distributions implemented in GAMLSS.

The Skew t type 1 distribution, ST1, is based on Azzalini (1986).

The skew t type 2 distribution, ST2, is based on Azzalini and Capitanio (2003).

The skew t type 3 , ST3 and ST3C, distribution is based Fernande and Steel (1998). The difference between the ST3 and ST3C is that the first is written entirely in R while the second is in C.

The skew t type 4 distribution , ST4, is a spliced-shape distribution.

The skew t type 5 distribution , ST5, is Jones and Faddy (2003).

The SST is a reparametrised version of dST3 where sigma is the standard deviation of the distribution.

**Usage**

```
ST1(mu.link = "identity", sigma.link = "log", nu.link = "identity", tau.link="log")
dST1(x, mu = 0, sigma = 1, nu = 0, tau = 2, log = FALSE)
pST1(q, mu = 0, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE, log.p = FALSE)
qST1(p, mu = 0, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE, log.p = FALSE)
rST1(n, mu = 0, sigma = 1, nu = 0, tau = 2)
```

```
ST2(mu.link = "identity", sigma.link = "log", nu.link = "identity", tau.link = "log")
dST2(x, mu = 0, sigma = 1, nu = 0, tau = 2, log = FALSE)
pST2(q, mu = 0, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE, log.p = FALSE)
qST2(p, mu = 1, sigma = 1, nu = 0, tau = 2, lower.tail = TRUE, log.p = FALSE)
rST2(n, mu = 0, sigma = 1, nu = 0, tau = 2)
```

```
ST3(mu.link = "identity", sigma.link = "log", nu.link = "log", tau.link = "log")
dST3(x, mu = 0, sigma = 1, nu = 1, tau = 10, log = FALSE)
pST3(q, mu = 0, sigma = 1, nu = 1, tau = 10, lower.tail = TRUE, log.p = FALSE)
qST3(p, mu = 0, sigma = 1, nu = 1, tau = 10, lower.tail = TRUE, log.p = FALSE)
rST3(n, mu = 0, sigma = 1, nu = 1, tau = 10)
```

```
ST3C(mu.link = "identity", sigma.link = "log", nu.link = "log", tau.link = "log")
dST3C(x, mu = 0, sigma = 1, nu = 1, tau = 10, log = FALSE)
pST3C(q, mu = 0, sigma = 1, nu = 1, tau = 10, lower.tail = TRUE, log.p = FALSE)
qST3C(p, mu = 0, sigma = 1, nu = 1, tau = 10, lower.tail = TRUE, log.p = FALSE)
rST3C(n, mu = 0, sigma = 1, nu = 1, tau = 10)
```

```
SST(mu.link = "identity", sigma.link = "log", nu.link = "log",
    tau.link = "logshiftto2")
dSST(x, mu = 0, sigma = 1, nu = 0.8, tau = 7, log = FALSE)
pSST(q, mu = 0, sigma = 1, nu = 0.8, tau = 7, lower.tail = TRUE, log.p = FALSE)
```

```
qSST(p, mu = 0, sigma = 1, nu = 0.8, tau = 7, lower.tail = TRUE, log.p = FALSE)
rSST(n, mu = 0, sigma = 1, nu = 0.8, tau = 7)
```

```
ST4(mu.link = "identity", sigma.link = "log", nu.link = "log", tau.link = "log")
dST4(x, mu = 0, sigma = 1, nu = 1, tau = 10, log = FALSE)
pST4(q, mu = 0, sigma = 1, nu = 1, tau = 10, lower.tail = TRUE, log.p = FALSE)
qST4(p, mu = 0, sigma = 1, nu = 1, tau = 10, lower.tail = TRUE, log.p = FALSE)
rST4(n, mu = 0, sigma = 1, nu = 1, tau = 10)
```

```
ST5(mu.link = "identity", sigma.link = "log", nu.link = "identity", tau.link = "log")
dST5(x, mu = 0, sigma = 1, nu = 0, tau = 1, log = FALSE)
pST5(q, mu = 0, sigma = 1, nu = 0, tau = 1, lower.tail = TRUE, log.p = FALSE)
qST5(p, mu = 0, sigma = 1, nu = 0, tau = 1, lower.tail = TRUE, log.p = FALSE)
rST5(n, mu = 0, sigma = 1, nu = 0, tau = 1)
```

### Arguments

mu.link	Defines the mu.link, with "identity" link as the default for the mu parameter. Other links are "1/mu <sup>2</sup> " and "log"
sigma.link	Defines the sigma.link, with "log" link as the default for the sigma parameter. Other links are "inverse" and "identity"
nu.link	Defines the nu.link, with "identity" link as the default for the nu parameter. Other links are "1/mu <sup>2</sup> " and "log"
tau.link	Defines the nu.link, with "log" link as the default for the nu parameter. Other links are "inverse", "identity"
x, q	vector of quantiles
mu	vector of mu parameter values
sigma	vector of scale parameter values
nu	vector of nu parameter values
tau	vector of tau parameter values
log, log.p	logical; if TRUE, probabilities p are given as log(p).
lower.tail	logical; if TRUE (default), probabilities are P[X <= x], otherwise, P[X > x]
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required
...	for extra arguments

### Details

$$f(y|\mu, \sigma, \nu, \frac{z}{\sigma} f_{z_1}(z) F_{z_2}(w)\tau) =$$

for  $-\infty < y < \infty$ , where  $z = (y - \mu)/\sigma$ ,  $w = \nu\lambda^{1/2}z$ ,  $\lambda = (\tau + 1)/(\tau + z^2)$  and  $z_1 \sim TF(0, 1, \tau)$  and  $z_2 \sim TF(0, 1, \tau + 1)$ .



The probability density function of the skew t distribution type q, (ST3), is defined in Chapter 10 of the GAMLSS manual.

The probability density function of the skew t distribution type q, (ST4), is defined in Chapter of the GAMLSS manual.

The probability density function of the skew t distribution type 5, (ST5), is defined as

$$f(y|\mu, \sigma, \nu, \tau) = \frac{1}{c} \left[ 1 + \frac{z}{(a+b+z^2)^{1/2}} \right]^{a+1/2} \left[ 1 - \frac{z}{(a+b+z^2)^{1/2}} \right]^{b+1/2}$$

where  $c = 2^{a+b-1}(a+b)^{1/2}B(a,b)$ , and  $B(a,b) = \Gamma(a)\Gamma(b)/\Gamma(a+b)$  and  $z = (y - \mu)/\sigma$  and  $\nu = (a - b)/[ab(a+b)]^{1/2}$  and  $\tau = 2/(a+b)$  for  $-\infty < y < \infty$ ,  $-\infty < \mu < \infty$ ,  $\sigma > 0$ ,  $-\infty < \nu < \infty$  and  $\tau > 0$ .

### Value

ST1(), ST2(), ST3(), ST4() and ST5() return a `gamlss.family` object which can be used to fit the skew t type 1-5 distribution in the `gamlss()` function. `dST1()`, `dST2()`, `dST3()`, `dST4()` and `dST5()` give the density functions, `pST1()`, `pST2()`, `pST3()`, `pST4()` and `pST5()` give the cumulative distribution functions, `qST1()`, `qST2()`, `qST3()`, `qST4()` and `qST5()` give the quantile function, and `rST1()`, `rST2()`, `rST3()`, `rST4()` and `rST5()` generates random deviates.

### Note

The mean of the ex-Gaussian is  $\mu + \nu$  and the variance is  $\sigma^2 + \nu^2$ .

### Author(s)

Bob Rigby and Mikis Stasinopoulos

### References

- Azzalini A. (1986) Further results on a class of distributions which includes the normal ones, *Statistica*, **46**, pp. 199-208.
- Azzalini A. and Capitanio, A. Distributions generated by perturbation of symmetry with emphasis on a multivariate skew t-distribution, *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **65**, pp. 367-389.
- Jones, M.C. and Faddy, M. J. (2003) A skew extension of the t distribution, with applications. *Journal of the Royal Statistical Society, Series B*, **65**, pp 159-174.
- Fernandez, C. and Steel, M. F. (1998) On Bayesian modeling of fat tails and skewness. *Journal of the American Statistical Association*, **93**, pp. 359-371.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R. A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

**See Also**

[gamlss.family](#), [SEP1](#), [SHASH](#)

**Examples**

```
y<- rST5(200, mu=5, sigma=1, nu=.1)
hist(y)
curve(dST5(x, mu=30 ,sigma=5,nu=-1), -50, 50, main = "The ST5 density mu=30 ,sigma=5,nu=1")
# library(gamlss)
# m1<-gamlss(y~1, family=ST1)
# m2<-gamlss(y~1, family=ST2)
# m3<-gamlss(y~1, family=ST3)
# m4<-gamlss(y~1, family=ST4)
# m5<-gamlss(y~1, family=ST5)
# GAIC(m1,m2,m3,m4,m5)
```

TF

*t family distribution for fitting a GAMLSS***Description**

The function TF defines the t-family distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dTF`, `pTF`, `qTF` and `rTF` define the density, distribution function, quantile function and random generation for the specific parameterization of the t distribution given in details below, with mean equal to  $\mu$  and standard deviation equal to  $\sigma(\frac{\nu}{\nu-2})^{0.5}$  with the degrees of freedom  $\nu$ . The function TF2 is a different parametrization where `sigma` is the standard deviation.

**Usage**

```
TF(mu.link = "identity", sigma.link = "log", nu.link = "log")
dTF(x, mu = 0, sigma = 1, nu = 10, log = FALSE)
pTF(q, mu = 0, sigma = 1, nu = 10, lower.tail = TRUE, log.p = FALSE)
qTF(p, mu = 0, sigma = 1, nu = 10, lower.tail = TRUE, log.p = FALSE)
rTF(n, mu = 0, sigma = 1, nu = 10)

TF2(mu.link = "identity", sigma.link = "log", nu.link = "logshiftto2")
dTF2(x, mu = 0, sigma = 1, nu = 10, log = FALSE)
pTF2(q, mu = 0, sigma = 1, nu = 10, lower.tail = TRUE, log.p = FALSE)
qTF2(p, mu = 0, sigma = 1, nu = 10, lower.tail = TRUE, log.p = FALSE)
rTF2(n, mu = 0, sigma = 1, nu = 10)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "identity" 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
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of the degrees of freedom parameter values
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> 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. If $\text{length}(n) > 1$ , the length is taken to be the number required

### Details

Definition file for t family distribution.

$$f(y|\mu, \sigma, \nu) = \frac{\Gamma((\nu + 1)/2)}{\sigma\Gamma(1/2)\Gamma(\nu/2)\nu^{0.5}} \left[ 1 + \frac{(y - \mu)^2}{\nu\sigma^2} \right]^{-(\nu+1)/2}$$

$y = (-\infty, +\infty)$ ,  $\mu = (-\infty, +\infty)$ ,  $\sigma > 0$  and  $\nu > 0$ . Note that  $z = (y - \mu)/\sigma$  has a standard t distribution with degrees of freedom  $\nu$ .

### Value

`TF()` returns a `gamlss.family` object which can be used to fit a t distribution in the `gamlss()` function. `dTF()` gives the density, `pTF()` gives the distribution function, `qTF()` gives the quantile function, and `rTF()` generates random deviates. The latest functions are based on the equivalent R functions for gamma distribution.

### Note

$\mu$  is the mean and  $\sigma[\nu/(\nu - 2)]^{0.5}$  is the standard deviation of the t family distribution.  $\nu > 0$  is a positive real valued parameter.

### Author(s)

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby and Kalliope Akantziliotou

### References

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M., Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#)

### Examples

```
TF()# gives information about the default links for the t-family distribution
# library(gamlss)
#data(abdom)
#h<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=TF, data=abdom) # fits
#plot(h)
newdata<-rTF(1000,mu=0,sigma=1,nu=5) # generates 1000 random observations
hist(newdata)
```

---

WARING

*Waring distribution for fitting a GAMLSS model*

---

### Description

The function `WARING()` defines the Waring distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`, with mean equal to the parameter `mu` and scale parameter `sigma`. The functions `dWARING`, `pWARING`, `qWARING` and `rWARING` define the density, distribution function, quantile function and random generation for the `WARING` parameterization of the Waring distribution.

### Usage

```
WARING(mu.link = "log", sigma.link = "log")
dWARING(x, mu = 2, sigma = 2, log = FALSE)
pWARING(q, mu = 2, sigma = 2, lower.tail = TRUE, log.p = FALSE)
qWARING(p, mu = 2, sigma = 2, lower.tail = TRUE, log.p = FALSE,
        max.value = 10000)
rWARING(n, mu = 2, sigma = 2)
```

### 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>x</code>	vector of (non-negative integer) quantiles.
<code>q</code>	vector of quantiles.
<code>p</code>	vector of probabilities.

n	number of random values to return.
mu	vector of positive mu values.
sigma	vector of positive sigma values.
lower.tail	logical; if TRUE (default) probabilities are $P[Y \leq y]$ , otherwise, $P[Y > y]$ .
log, log.p	logical; if TRUE probabilities p are given as $\log(p)$ .
max.value	constant; generates a sequence of values for the cdf function.

### Details

The Waring distribution has density,

$$f(y|\mu, \sigma) = \frac{(1 + \sigma) \Gamma(y + \frac{\mu}{\sigma}) \Gamma(\frac{\mu + \sigma + 1}{\sigma})}{\sigma \Gamma(y + \frac{\mu + 1}{\sigma} + 2) \Gamma(\frac{\mu}{\sigma})}$$

for  $y = 0, 1, 2, \dots$ ,  $\mu > 0$  and  $\sigma > 0$ .

### Value

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

### Author(s)

Fiona McElduff, Bob Rigby and Mikis Stasinopoulos. <f.mcelduff@ich.ucl.ac.uk>

### References

Wimmer, G. and Altmann, G. (1999) *Thesaurus of univariate discrete probability distributions*. Stamm.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#)

### Examples

```
par(mfrow=c(2,2))
y<-seq(0,20,1)
plot(y, dWARING(y), type="h")
q <- seq(0, 20, 1)
plot(q, pWARING(q), type="h")
p<-seq(0.0001,0.999,0.05)
plot(p , qWARING(p), type="s")
dat <- rWARING(100)
hist(dat)
#summary(gamlss(dat~1, family=WARING))
```

WEI

*Weibull distribution for fitting a GAMLSS***Description**

The function WEI can be used to define the Weibull distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. [Note that the GAMLSS function WEI2 uses a different parameterization for fitting the Weibull distribution.] The functions `dWEI`, `pWEI`, `qWEI` and `rWEI` define the density, distribution function, quantile function and random generation for the specific parameterization of the Weibull distribution.

**Usage**

```
WEI(mu.link = "log", sigma.link = "log")
dWEI(x, mu = 1, sigma = 1, log = FALSE)
pWEI(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qWEI(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rWEI(n, mu = 1, sigma = 1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "log" link as the default for the mu parameter, other links are "inverse", "identity" and "own"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the sigma parameter, other link is the "inverse", "identity" and "own"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of the mu parameter
<code>sigma</code>	vector of sigma 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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

**Details**

The parameterization of the function WEI is given by

$$f(y|\mu, \sigma) = \frac{\sigma y^{\sigma-1}}{\mu^\sigma} \exp \left[ - \left( \frac{y}{\mu} \right)^\sigma \right]$$

for  $y > 0$ ,  $\mu > 0$  and  $\sigma > 0$ . The GAMLSS functions `dWEI`, `pWEI`, `qWEI`, and `rWEI` can be used to provide the pdf, the cdf, the quantiles and random generated numbers for the Weibull distribution with argument `mu`, and `sigma`. [See the GAMLSS function WEI2 for a different parameterization of the Weibull.]

**Value**

WEI() returns a `gamlss.family` object which can be used to fit a Weibull distribution in the `gamlss()` function. `dWEI()` gives the density, `pWEI()` gives the distribution function, `qWEI()` gives the quantile function, and `rWEI()` generates random deviates. The latest functions are based on the equivalent R functions for Weibull distribution.

**Note**

The mean in WEI is given by  $\mu\Gamma(\frac{1}{\sigma} + 1)$  and the variance  $\mu^2 [\Gamma(\frac{2}{\sigma} + 1) - (\Gamma(\frac{1}{\sigma} + 1))^2]$

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby and Calliope Akantziliotou

**References**

Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.

Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [WEI2](#), [WEI3](#)

**Examples**

```
WEI()
dat<-rWEI(100, mu=10, sigma=2)
# library(gamlss)
# gamlss(dat~1, family=WEI)
```

---

WEI2	<i>A specific parameterization of the Weibull distribution for fitting a GAMLSS</i>
------	---

---

### Description

The function WEI2 can be used to define the Weibull distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. This is the parameterization of the Weibull distribution usually used in proportional hazard models and is defined in details below. [Note that the GAMLSS function WEI uses a different parameterization for fitting the Weibull distribution.] The functions `dWEI2`, `pWEI2`, `qWEI2` and `rWEI2` define the density, distribution function, quantile function and random generation for the specific parameterization of the Weibull distribution.

### Usage

```
WEI2(mu.link = "log", sigma.link = "log")
dWEI2(x, mu = 1, sigma = 1, log = FALSE)
pWEI2(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qWEI2(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rWEI2(n, mu = 1, sigma = 1)
```

### Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "log" link as the default for the mu parameter, other links are "inverse" and "identity"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the sigma parameter, other link is the "inverse" and "identity"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of the mu parameter values
<code>sigma</code>	vector of sigma parameter values
<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. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

### Details

The parameterization of the function WEI2 is given by

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

for  $y > 0$ ,  $\mu > 0$  and  $\sigma > 0$ . The GAMLSS functions `dWEI2`, `pWEI2`, `qWEI2`, and `rWEI2` can be used to provide the pdf, the cdf, the quantiles and random generated numbers for the Weibull distribution with argument `mu`, and `sigma`. [See the GAMLSS function WEI for a different parameterization of the Weibull.]



**Value**

WEI2() returns a `gamlss.family` object which can be used to fit a Weibull distribution in the `gamlss()` function. `dWEI2()` gives the density, `pWEI2()` gives the distribution function, `qWEI2()` gives the quantile function, and `rWEI2()` generates random deviates. The latest functions are based on the equivalent R functions for Weibull distribution.

**Warning**

In WEI2 the estimated parameters `mu` and `sigma` can be highly correlated so it is advisable to use the `CG()` method for fitting [as the `RS()` method can be very slow in this situation.]

**Note**

The mean in WEI2 is given by  $\mu^{-1/\sigma}\Gamma(\frac{1}{\sigma} + 1)$  and the variance  $\mu^{-2/\sigma}(\Gamma(\frac{2}{\sigma} + 1) - [\Gamma(\frac{1}{\sigma} + 1)]^2)$

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby and Calliope Akantziliotou

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [WEI](#), [WEI3](#),

**Examples**

```
WEI2()
dat<-rWEI(100, mu=.1, sigma=2)
hist(dat)
# library(gamlss)
# gamlss(dat~1, family=WEI2, method=CG())
```

---

WEI3 *A specific parameterization of the Weibull distribution for fitting a GAMLSS*

---

### Description

The function WEI3 can be used to define the Weibull distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. This is a parameterization of the Weibull distribution where  $\mu$  is the mean of the distribution. [Note that the GAMLSS functions WEI and WEI2 use different parameterizations for fitting the Weibull distribution.] The functions dWEI3, pWEI3, qWEI3 and rWEI3 define the density, distribution function, quantile function and random generation for the specific parameterization of the Weibull distribution.

### Usage

```
WEI3(mu.link = "log", sigma.link = "log")
dWEI3(x, mu = 1, sigma = 1, log = FALSE)
pWEI3(q, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qWEI3(p, mu = 1, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rWEI3(n, mu = 1, sigma = 1)
```

### Arguments

<code>mu.link</code>	Defines the <code>mu.link</code> , with "log" link as the default for the mu parameter, other links are "inverse" and "identity"
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "log" link as the default for the sigma parameter, other link is the "inverse" and "identity"
<code>x, q</code>	vector of quantiles
<code>mu</code>	vector of the mu parameter values
<code>sigma</code>	vector of sigma parameter values
<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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations. If <code>length(n) &gt; 1</code> , the length is taken to be the number required

### Details

The parameterization of the function WEI3 is given by

$$f(y|\mu, \sigma) = \frac{\sigma}{\beta} \left(\frac{y}{\beta}\right)^{\sigma-1} e^{-\left(\frac{y}{\beta}\right)^\sigma}$$

where  $\beta = \frac{\mu}{\Gamma((1/\sigma)+1)}$  for  $y > 0$ ,  $\mu > 0$  and  $\sigma > 0$ . The GAMLSS functions dWEI3, pWEI3, qWEI3, and rWEI3 can be used to provide the pdf, the cdf, the quantiles and random generated numbers

for the Weibull distribution with argument `mu`, and `sigma`. [See the GAMLSS function `WEI` for a different parameterization of the Weibull.]

### Value

`WEI3()` returns a `gamlss.family` object which can be used to fit a Weibull distribution in the `gamlss()` function. `dWEI3()` gives the density, `pWEI3()` gives the distribution function, `qWEI3()` gives the quantile function, and `rWEI3()` generates random deviates. The latest functions are based on the equivalent R functions for Weibull distribution.

### Warning

In `WEI3` the estimated parameters `mu` and `sigma` can be highly correlated so it is advisable to use the `CG()` method for fitting [as the `RS()` method can be very slow in this situation.]

### Note

The mean in `WEI3` is given by  $\mu$  and the variance  $\mu^2 \left\{ \frac{\Gamma(2/\sigma + 1)}{[\Gamma(1/\sigma + 1)]^2} - 1 \right\}$

### Author(s)

Bob Rigby and Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [WEI](#), [WEI2](#)

### Examples

```
WEI3()
dat<-rWEI(100, mu=.1, sigma=2)
# library(gamlss)
# gamlss(dat~1, family=WEI3, method=CG())
```

---

YULE

*Yule distribution for fitting a GAMLSS model*


---

**Description**

The function YULE defines the Yule distribution, a one parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`, with mean equal to the parameter `mu`. The functions `dYULE`, `pYULE`, `qYULE` and `rYULE` define the density, distribution function, quantile function and random generation for the YULE parameterization of the Yule distribution.

**Usage**

```
YULE(mu.link = "log")
dYULE(x, mu = 2, log = FALSE)
pYULE(q, mu = 2, lower.tail = TRUE, log.p = FALSE)
qYULE(p, mu = 2, lower.tail = TRUE, log.p = FALSE,
      max.value = 10000)
rYULE(n, mu = 2)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "log" link as the default for the <code>mu</code> parameter
<code>x</code>	vector of (non-negative integer) quantiles.
<code>q</code>	vector of quantiles.
<code>p</code>	vector of probabilities.
<code>n</code>	number of random values to return.
<code>mu</code>	vector of positive <code>mu</code> values.
<code>lower.tail</code>	logical; if TRUE (default) probabilities are $P[Y \leq y]$ , otherwise, $P[Y > y]$ .
<code>log, log.p</code>	logical; if TRUE probabilities <code>p</code> are given as $\log(p)$ .
<code>max.value</code>	constant; generates a sequence of values for the cdf function.

**Details**

The Yule distribution has density

$$P(Y = y|\mu) = (\mu^{-1} + 1)B(y + 1, \mu^{-1} + 2)$$

for  $y = 0, 1, 2, \dots$  and  $mu > 0$ .

**Value**

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

**Author(s)**

Fiona McElduff, Bob Rigby and Mikis Stasinopoulos.

**References**

Wimmer, G. and Altmann, G. (1999) *Thesaurus of univariate discrete probability distributions*. Stamm.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#)

**Examples**

```
par(mfrow=c(2,2))
y<-seq(0,20,1)
plot(y, dYULE(y), type="h")
q <- seq(0, 20, 1)
plot(q, pYULE(q), type="h")
p<-seq(0.0001,0.999,0.05)
plot(p , qYULE(p), type="s")
dat <- rYULE(100)
hist(dat)
#summary(gamlss(dat~1, family=YULE))
```

---

ZABB

*Zero inflated and zero adjusted Binomial distribution for fitting in GAMLSS*

---

**Description**

The function ZIBB defines the zero inflated beta binomial distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZIBB`, `pZIBB`, `qZIBB` and `rZINN` define the density, distribution function, quantile function and random generation for the zero inflated beta binomial, ZIBB, distribution.

The function ZABB defines the zero adjusted beta binomial distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZABB`, `pZABB`, `qZABB` and `rZABB` define the density, distribution function, quantile function and random generation for the zero inflated beta binomial, ZABB(), distribution.

**Usage**

```

ZABB(mu.link = "logit", sigma.link = "log", nu.link = "logit")
ZIBB(mu.link = "logit", sigma.link = "log", nu.link = "logit")

dZIBB(x, mu = 0.5, sigma = 0.5, nu = 0.1, bd = 1, log = FALSE)
dZABB(x, mu = 0.5, sigma = 0.1, nu = 0.1, bd = 1, log = FALSE)

pZIBB(q, mu = 0.5, sigma = 0.5, nu = 0.1, bd = 1, lower.tail = TRUE, log.p = FALSE)
pZABB(q, mu = 0.5, sigma = 0.1, nu = 0.1, bd = 1, lower.tail = TRUE, log.p = FALSE)

qZIBB(p, mu = 0.5, sigma = 0.5, nu = 0.1, bd = 1, lower.tail = TRUE, log.p = FALSE)
qZABB(p, mu = 0.5, sigma = 0.1, nu = 0.1, bd = 1, lower.tail = TRUE, log.p = FALSE)

rZIBB(n, mu = 0.5, sigma = 0.5, nu = 0.1, bd = 1)
rZABB(n, mu = 0.5, sigma = 0.1, nu = 0.1, bd = 1)

```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "logit" link as the default for the <code>mu</code> parameter. Other links are "probit" and "cloglog" (complementary log-log)
<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 "logit" link as the default for the <code>nu</code> parameter. Other links are "probit" and "cloglog" (complementary log-log)
<code>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive probabilities
<code>sigma</code>	vector of positive dispersion parameter
<code>nu</code>	vector of positive probabilities
<code>bd</code>	vector of binomial denominators
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$

**Details**

For the definition of the distributions see Rigby and Stasinopoulos (2010) below.

**Value**

The functions ZIBB and ZABB return a `gamlss.family` object which can be used to fit a zero inflated or zero adjusted beta binomial distribution respectively in the `gamlss()` function.

**Author(s)**

Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby

## References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Rigby, R. A. and Stasinopoulos D. M. (2010) The gamlss.family distributions, (distributed with this package or see <http://www.gamlss.org/>)
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family](#), [NBI](#), [NBII](#)

## Examples

```
ZIBB()
ZABB()
# creating data and plotting them
dat <- rZIBB(1000, mu=.5, sigma=.5, nu=0.1, bd=10)
r <- barplot(table(dat), col='lightblue')
dat1 <- rZABB(1000, mu=.5, sigma=.2, nu=0.1, bd=10)
r1 <- barplot(table(dat1), col='lightblue')
```

---

ZABI

*Zero inflated and zero adjusted Binomial distribution for fitting in GAMLSS*

---

## Description

The ZABI() function defines the zero adjusted binomial distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZABI`, `pZABI`, `qZABI` and `rZABI` define the density, distribution function, quantile function and random generation for the zero adjusted binomial, ZABI(), distribution.

The ZIBI() function defines the zero inflated binomial distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZIBI`, `pZIBI`, `qZIBI` and `rZIBI` define the density, distribution function, quantile function and random generation for the zero inflated binomial, ZIBI(), distribution.

**Usage**

```
ZABI(mu.link = "logit", sigma.link = "logit")
dZABI(x, bd = 1, mu = 0.5, sigma = 0.1, log = FALSE)
pZABI(q, bd = 1, mu = 0.5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
qZABI(p, bd = 1, mu = 0.5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
rZABI(n, bd = 1, mu = 0.5, sigma = 0.1)

ZIBI(mu.link = "logit", sigma.link = "logit")
dZIBI(x, bd = 1, mu = 0.5, sigma = 0.1, log = FALSE)
pZIBI(q, bd = 1, mu = 0.5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
qZIBI(p, bd = 1, mu = 0.5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
rZIBI(n, bd = 1, mu = 0.5, sigma = 0.1)
```

**Arguments**

<code>mu.link</code>	Defines the <code>mu.link</code> , with "logit" link as the default for the mu parameter. Other links are "probit" and "cloglog" (complementary log-log)
<code>sigma.link</code>	Defines the <code>sigma.link</code> , with "logit" link as the default for the mu parameter. Other links are "probit" and "cloglog" (complementary log-log)
<code>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive probabilities
<code>sigma</code>	vector of positive probabilities
<code>bd</code>	vector of binomial denominators
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<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, $P[X > x]$

**Details**

For the definition of the distributions see Rigby and Stasinopoulos (2010) below.

**Value**

The functions ZABI and ZIBI return a `gamlss.family` object which can be used to fit a binomial distribution in the `gamlss()` function.

**Note**

The response variable should be a matrix containing two columns, the first with the count of successes and the second with the count of failures.

**Author(s)**

Mikis Stasinopoulos, Bob Rigby



## References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Rigby, R. A. and Stasinopoulos D. M. (2010) The `gamlss.family` distributions, (distributed with this package or see <http://www.gamlss.org/>)
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family](#), [BI](#)

## Examples

```
ZABI()
curve(dZABI(x, mu = .5, bd=10), from=0, to=10, n=10+1, type="h")
tN <- table(Ni <- rZABI(1000, mu=.2, sigma=.3, bd=10))
r <- barplot(tN, col='lightblue')
```

```
ZIBI()
curve(dZIBI(x, mu = .5, bd=10), from=0, to=10, n=10+1, type="h")
tN <- table(Ni <- rZIBI(1000, mu=.2, sigma=.3, bd=10))
r <- barplot(tN, col='lightblue')
```

## Description

The function `ZAGA()` defines the zero adjusted Gamma distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The zero adjusted Gamma distribution is similar to the Gamma distribution but allows zeros as y values. The extra parameter `nu` models the probabilities at zero. The functions `dZAGA`, `pZAGA`, `qZAGA` and `rZAGA` define the density, distribution function, quartile function and random generation for the ZAGA parameterization of the zero adjusted Gamma distribution. `plotZAGA` can be used to plot the distribution. `meanZAGA` calculates the expected value of the response for a fitted model.

**Usage**

```
ZAGA(mu.link = "log", sigma.link = "log", nu.link = "logit")
dZAGA(x, mu = 1, sigma = 1, nu = 0.1, log = FALSE)
pZAGA(q, mu = 1, sigma = 1, nu = 0.1, lower.tail = TRUE,
      log.p = FALSE)
qZAGA(p, mu = 1, sigma = 1, nu = 0.1, lower.tail = TRUE,
      log.p = FALSE)
rZAGA(n, mu = 1, sigma = 1, nu = 0.1, ...)
plotZAGA(mu = 5, sigma = 1, nu = 0.1, from = 0, to = 10,
         n = 101, main=NULL, ...)
meanZAGA(obj)
```

**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 parameter
<code>nu.link</code>	Defines the <code>nu.link</code> , with "logit" link as the default for the sigma parameter
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of probability at zero parameter values
<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 <= x], otherwise, P[X > x]
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations. If <code>length(n) &gt; 1</code> , the length is taken to be the number required
<code>from</code>	where to start plotting the distribution from
<code>to</code>	up to where to plot the distribution
<code>obj</code>	a fitted <code>gamlss</code> object
<code>main</code>	for title in the plot
<code>...</code>	... can be used to pass the <code>uppr.limit</code> argument to <code>qIG</code>

**Details**

The Zero adjusted GA distribution is given as

$$f(y|\mu, \sigma, \nu) = \nu$$

if  $(y=0)$

$$f(y|\mu, \sigma, \nu) = (1 - \nu) \left[ \frac{1}{(\sigma^2 \mu)^{1/\sigma^2}} \frac{y^{\frac{1}{\sigma^2} - 1} e^{-y/(\sigma^2 \mu)}}{\Gamma(1/\sigma^2)} \right]$$

otherwise

for  $y = (0, \infty)$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $0 < \nu < 1$ .  $E(y) = (1 - \nu)\mu$  and  $Var(y) = (1 - \nu)\mu^2(\nu + \sigma^2)$ .

**Value**

The function ZAGA returns a `gamlss.family` object which can be used to fit a zero adjusted Gamma distribution in the `gamlss()` function.

**Author(s)**

Bob Rigby, Mikis Stasinopoulos and Almond Stocker

**References**

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [GA](#), [ZAIG](#)

**Examples**

```
ZAGA()# gives information about the default links for the ZAGA distribution
# plotting the function
PPP <- par(mfrow=c(2,2))
plotZAGA(mu=1, sigma=.5, nu=.2, from=0,to=3)
#curve(dZAGA(x,mu=1, sigma=.5, nu=.2), 0,3) # pdf
curve(pZAGA(x,mu=1, sigma=.5, nu=.2), 0,3, ylim=c(0,1)) # cdf
curve(qZAGA(x,mu=1, sigma=.5, nu=.2), 0,.99) # inverse cdf
y<-rZAGA(100, mu=1, sigma=.5, nu=.2) # randomly generated values
hist(y)
par(PPP)
# check that the positive part sums up to .8 (since nu=0.2)
integrate(function(x) dZAGA(x,mu=1, sigma=.5, nu=.2), 0,Inf)
```

## Description

The function `ZAIG()` defines the zero adjusted Inverse Gaussian distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The zero adjusted Inverse Gaussian distribution is similar to the Inverse Gaussian distribution but allows zeros as y values. The extra parameter models the probabilities at zero. The functions `dZAIG`, `pZAIG`, `qZAIG` and `rZAIG` define the density, distribution function, quantile function and random generation for the ZAIG parameterization of the zero adjusted Inverse Gaussian distribution. `plotZAIG` can be used to plot the distribution. `meanZAIG` calculates the expected value of the response for a fitted model.

## Usage

```
ZAIG(mu.link = "log", sigma.link = "log", nu.link = "logit")
dZAIG(x, mu = 1, sigma = 1, nu = 0.1, log = FALSE)
pZAIG(q, mu = 1, sigma = 1, nu = 0.1, lower.tail = TRUE, log.p = FALSE)
qZAIG(p, mu = 1, sigma = 1, nu = 0.1, lower.tail = TRUE, log.p = FALSE,
      upper.limit = mu + 10 * sqrt(sigma^2 * mu^3))
rZAIG(n, mu = 1, sigma = 1, nu = 0.1, ...)
plotZAIG(mu = 5, sigma = 1, nu = 0.1, from = 0, to = 10, n = 101, ...)
meanZAIG(obj)
```

## 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 parameter
<code>nu.link</code>	Defines the <code>nu.link</code> , with "logit" link as the default for the sigma parameter
<code>x,q</code>	vector of quantiles
<code>mu</code>	vector of location parameter values
<code>sigma</code>	vector of scale parameter values
<code>nu</code>	vector of probability at zero parameter values
<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>upper.limit</code>	the argument <code>upper.limit</code> sets the upper limit in the golden section search for q, the default is 10 time its standard deviation
<code>p</code>	vector of probabilities.
<code>n</code>	number of observations. If <code>length(n) &gt; 1</code> , the length is taken to be the number required
<code>from</code>	where to start plotting the distribution from
<code>to</code>	up to where to plot the distribution
<code>obj</code>	a fitted BEINF object
<code>...</code>	... can be used to pass the <code>uppr.limit</code> argument to <code>qIG</code>

**Details**

The Zero adjusted IG distribution is given as

$$f(y|\mu, \sigma, \nu) = \nu$$

if  $(y=0)$

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

otherwise

for  $y \in (0, \infty)$ ,  $\mu > 0$ ,  $\sigma > 0$  and  $0 < \nu < 1$ .  $E(y) = (1 - \nu)\mu$  and  $Var(y) = (1 - \nu)\mu^2(\nu + \mu\sigma^2)$ .

**Value**

returns a `gamlss` family object which can be used to fit a zero adjusted inverse Gaussian distribution in the `gamlss()` function.

**Author(s)**

Bob Rigby and Mikis Stasinopoulos

**References**

- Heller, G. Stasinopoulos M and Rigby R.A. (2006) The zero-adjusted Inverse Gaussian distribution as a model for insurance claims. in *Proceedings of the 21th International Workshop on Statistical Modelling*, eds J. Hinde, J. Einbeck and J. Newell, pp 226-233, Galway, Ireland.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family, IG](#)

**Examples**

```
ZAIG()# gives information about the default links for the ZAIG distribution
# plotting the distribution
plotZAIG( mu =10 , sigma=.5, nu = 0.1, from = 0, to=10, n = 101)
# plotting the cdf
plot(function(y) pZAIG(y, mu=10 ,sigma=.5, nu = 0.1 ), 0, 1)
# plotting the inverse cdf
```

```

plot(function(y) qZAIG(y, mu=10 ,sigma=.5, nu = 0.1 ), 0.001, .99)
# generate random numbers
dat <- rZAIG(100,mu=10,sigma=.5, nu=.1)
# fit a model to the data
# library(gamlss)
# m1<-gamlss(dat~1,family=ZAIG)
# meanZAIG(m1)[1]

```

ZANBI

*Zero inflated and zero adjusted negative binomial distributions for fitting a GAMLSS model*

## Description

The function ZINBI defines the zero inflated negative binomial distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZINBI`, `pZINBI`, `qZINBI` and `rZINBI` define the density, distribution function, quantile function and random generation for the zero inflated negative binomial, ZINBI(), distribution.

The function ZANBI defines the zero adjusted negative binomial distribution, a three parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZANBI`, `pZANBI`, `qZANBI` and `rZANBI` define the density, distribution function, quantile function and random generation for the zero inflated negative binomial, ZANBI(), distribution.

## Usage

```

ZINBI(mu.link = "log", sigma.link = "log", nu.link = "logit")
dZINBI(x, mu = 1, sigma = 1, nu = 0.3, log = FALSE)
pZINBI(q, mu = 1, sigma = 1, nu = 0.3, lower.tail = TRUE, log.p = FALSE)
qZINBI(p, mu = 1, sigma = 1, nu = 0.3, lower.tail = TRUE, log.p = FALSE)
rZINBI(n, mu = 1, sigma = 1, nu = 0.3)
ZANBI(mu.link = "log", sigma.link = "log", nu.link = "logit")
dZANBI(x, mu = 1, sigma = 1, nu = 0.3, log = FALSE)
pZANBI(q, mu = 1, sigma = 1, nu = 0.3, lower.tail = TRUE, log.p = FALSE)
qZANBI(p, mu = 1, sigma = 1, nu = 0.3, lower.tail = TRUE, log.p = FALSE)
rZANBI(n, mu = 1, sigma = 1, nu = 0.3)

```

## 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 parameter
<code>nu.link</code>	Defines the <code>nu.link</code> , with "logit" link as the default for the nu parameter
<code>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive means
<code>sigma</code>	vector of positive dispersion parameter
<code>nu</code>	vector of zero probability parameter

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

### Details

The definition for the zero inflated Negative Binomial type I distribution and for the zero adjusted Negative Binomial type I distribution is given in Rigby and Stasinopoulos (2010) below

### Value

The functions ZINBI and ZANBI return a `gamlss.family` object which can be used to fit a zero inflated or zero adjusted Negative Binomial type I distribution respectively in the `gamlss()` function.

### Author(s)

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby

### References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Rigby, R. A. and Stasinopoulos D. M. (2010) The `gamlss.family` distributions, (distributed with this package or see <http://www.gamlss.org/>)
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [NBI](#), [NBII](#)

### Examples

```
ZINBI()
ZANBI()
# creating data and plotting them
dat <- rZINBI(1000, mu=5, sigma=.5, nu=0.1)
r <- barplot(table(dat), col='lightblue')
dat1 <- rZANBI(1000, mu=5, sigma=.5, nu=0.1)
r1 <- barplot(table(dat1), col='lightblue')
```

ZAP

*Zero adjusted poisson distribution for fitting a GAMLSS model***Description**

The function ZAP defines the zero adjusted Poisson distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZAP`, `pZAP`, `qZAP` and `rZAP` define the density, distribution function, quantile function and random generation for the inflated poisson, `ZAP()`, distribution.

**Usage**

```
ZAP(mu.link = "log", sigma.link = "logit")
dZAP(x, mu = 5, sigma = 0.1, log = FALSE)
pZAP(q, mu = 5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
qZAP(p, mu = 5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
rZAP(n, mu = 5, sigma = 0.1)
```

**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 "logit" link as the default for the <code>sigma</code> parameter which in this case is the probability at zero. Other links are "probit" and "cloglog"(complementary log-log)
<code>x</code>	vector of (non-negative integer)
<code>mu</code>	vector of positive means
<code>sigma</code>	vector of probabilities at zero
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$

**Details**

For the definition of the distribution see Rigby and Stasinopoulos (2010) below.

**Value**

The function ZAP returns a `gamlss.family` object which can be used to fit a zero inflated poisson distribution in the `gamlss()` function.

**Author(s)**

Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby



## References

- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Rigby, R. A. and Stasinopoulos D. M. (2010) The gamlss.family distributions, (distributed with this package or see <http://www.gamlss.org/>)
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[gamlss.family](#), [P0](#), [ZIP](#), [ZIP2](#), [ZALG](#)

## Examples

```
ZAP()
# creating data and plotting them
dat<-rZAP(1000, mu=5, sigma=.1)
r <- barplot(table(dat), col='lightblue')
```

---

ZIP

*Zero inflated poisson distribution for fitting a GAMLSS model*

---

## Description

The function ZIP defines the zero inflated Poisson distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZIP`, `pZIP`, `qZIP` and `rZIP` define the density, distribution function, quantile function and random generation for the inflated poisson, `ZIP()`, distribution.

## Usage

```
ZIP(mu.link = "log", sigma.link = "logit")
dZIP(x, mu = 5, sigma = 0.1, log = FALSE)
pZIP(q, mu = 5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
qZIP(p, mu = 5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
rZIP(n, mu = 5, sigma = 0.1)
```

**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 "logit" link as the default for the <code>sigma</code> parameter which in this case is the probability at zero. Other links are "probit" and "cloglog"(complementary log-log)
<code>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive means
<code>sigma</code>	vector of probabilities at zero
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$

**Details**

Let  $Y = 0$  with probability  $\sigma$  and  $Y \sim Po(\mu)$  with probability  $(1 - \sigma)$  the  $Y$  has a Zero inflated Poisson Distribution given by

$$f(y) = \sigma + (1 - \sigma)e^{-\mu}$$

if ( $y=0$ )

$$f(y) = (1 - \sigma) \frac{e^{-\mu} \mu^y}{y!}$$

if ( $y>0$ ) for  $y = 0, 1, \dots$ ,

**Value**

returns a `gamlss.family` object which can be used to fit a zero inflated poisson distribution in the `gamlss()` function.

**Author(s)**

Mikis Stasinopoulos <[mikis.stasinopoulos@gamlss.org](mailto:mikis.stasinopoulos@gamlss.org)>, Bob Rigby

**References**

- Lambert, D. (1992), Zero-inflated Poisson Regression with an application to defects in Manufacturing, *Technometrics*, **34**, pp 1-14.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).

Stasinopoulos D. M., Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.

Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

### See Also

[gamlss.family](#), [P0](#), [ZIP2](#)

### Examples

```
ZIP()# gives information about the default links for the normal distribution
# creating data and plotting them
dat<-rZIP(1000, mu=5, sigma=.1)
r <- barplot(table(dat), col='lightblue')
# library(gamlss)
# fit the distribution
# mod1<-gamlss(dat~1, family=ZIP)# fits a constant for mu and sigma
# fitted(mod1)[1]
# fitted(mod1,"sigma")[1]
```

---

ZIP2

*Zero inflated poisson distribution for fitting a GAMLSS model*

---

### Description

The function ZIP2 defines the zero inflated Poisson type 2 distribution, a two parameter distribution, for a `gamlss.family` object to be used in GAMLSS fitting using the function `gamlss()`. The functions `dZIP2`, `pZIP2`, `qZIP2` and `rZIP2` define the density, distribution function, quantile function and random generation for the inflated poisson, `ZIP2()`, distribution. The ZIP2 is a different parameterization of the ZIP distribution. In the ZIP2 the `mu` is the mean of the distribution.

### Usage

```
ZIP2(mu.link = "log", sigma.link = "logit")
dZIP2(x, mu = 5, sigma = 0.1, log = FALSE)
pZIP2(q, mu = 5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
qZIP2(p, mu = 5, sigma = 0.1, lower.tail = TRUE, log.p = FALSE)
rZIP2(n, mu = 5, sigma = 0.1)
```

### 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 "logit" link as the default for the <code>sigma</code> parameter which in this case is the probability at zero. Other links are "probit" and "cloglog"(complementary log-log)

<code>x</code>	vector of (non-negative integer) quantiles
<code>mu</code>	vector of positive means
<code>sigma</code>	vector of probabilities at zero
<code>p</code>	vector of probabilities
<code>q</code>	vector of quantiles
<code>n</code>	number of random values to return
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as $\log(p)$
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$

### Details

Let  $Y = 0$  with probability  $\sigma$  and  $Y \sim Po(\mu/[1 - \sigma])$  with probability  $(1 - \sigma)$  then  $Y$  has a Zero inflated Poisson type 2 distribution given by

$$f(y|\mu, \sigma) = \sigma + (1 - \sigma)e^{-\mu/(1-\sigma)} \quad \text{if } y = 0$$

$$f(y|\mu, \sigma) = (1 - \sigma) \frac{e^{-\mu/(1-\sigma)} [\mu/(1 - \sigma)]^y}{y!} \quad \text{if } y = 1, 2, 3, \dots$$

The mean of the distribution in this parameterization is  $\mu$ .

### Value

returns a `gamlss.family` object which can be used to fit a zero inflated poisson distribution in the `gamlss()` function.

### Author(s)

Bob Rigby, Gillian Heller and Mikis Stasinopoulos

### References

- Lambert, D. (1992), Zero-inflated Poisson Regression with an application to defects in Manufacturing, *Technometrics*, **34**, pp 1-14.
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M., Rigby R.A. and Akantziliotou C. (2006) Instructions on how to use the GAMLSS package in R. Accompanying documentation in the current GAMLSS help files, (see also <http://www.gamlss.org/>).
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

**See Also**

[gamlss.family](#), [ZIP](#)

**Examples**

```
ZIP2()# gives information about the default links for the normal distribution
# creating data and plotting them
dat<-rZIP2(1000, mu=5, sigma=.1)
r <- barplot(table(dat), col='lightblue')
# fit the distribution
# library(gamlss)
# mod1<-gamlss(dat~1, family=ZIP2)# fits a constant for mu and sigma
# fitted(mod1)[1]
# fitted(mod1,"sigma")[1]
```

---

ZIPF

*The zipf and zero adjusted zipf distributions for fitting a GAMLSS model*

---

**Description**

This function `ZIPF()` defines the zipf distribution, Johnson et. al., (2005), sections 11.2.20, p 527-528. The zipf distribution is an one parameter distribution with long tails (a discrete version of the Pareto distribution). The function `ZIPF()` creates a `gamlss.family` object to be used in GAMLSS fitting. The functions `dZIPF`, `pZIPF`, `qZIPF` and `rZIPF` define the density, distribution function, quantile function and random generation for the zipf, `ZIPF()`, distribution. The function `zetaP()` defines the zeta function and it is based on the zeta function defined on the VGAM package of Thomas Yee, see Yee (2017).

The distribution zipf is defined on  $y = 1, 2, 3, \dots, \infty$ , the zero adjusted zipf permits values on  $y = 0, 1, 2, \dots, \infty$ . The function `ZAZIPF()` defines the zero adjusted zipf distribution. The function `ZAZIPF()` creates a `gamlss.family` object to be used in GAMLSS fitting. The functions `dZAZIPF`, `pZAZIPF`, `qZAZIPF` and `rZAZIPF` define the density, distribution function, quantile function and random generation for the zero adjusted zipf, `ZAZIPF()`, distribution.

**Usage**

```
ZIPF(mu.link = "log")
dZIPF(x, mu = 1, log = FALSE)
pZIPF(q, mu = 1, lower.tail = TRUE, log.p = FALSE)
qZIPF(p, mu = 1, lower.tail = TRUE, log.p = FALSE,
      max.value = 10000)
rZIPF(n, mu = 1, max.value = 10000)
zetaP(x)
ZAZIPF(mu.link = "log", sigma.link = "logit")
dZAZIPF(x, mu = 0.5, sigma = 0.1, log = FALSE)
pZAZIPF(q, mu = 0.5, sigma = 0.1, lower.tail = TRUE,
        log.p = FALSE)
```

```

qZAZIPF(p, mu = 0.5, sigma = 0.1, lower.tail = TRUE,
        log.p = FALSE, max.value = 10000)
rZAZIPF(n, mu = 0.5, sigma = 0.1, max.value = 10000)

```

### Arguments

<code>mu.link</code>	the link function for the parameter <code>mu</code> with default <code>log</code>
<code>x, q</code>	vectors of (non-negative integer) quantiles
<code>p</code>	vector of probabilities
<code>mu</code>	vector of positive parameter
<code>log, log.p</code>	logical; if TRUE, probabilities <code>p</code> are given as <code>log(p)</code>
<code>lower.tail</code>	logical; if TRUE (default), probabilities are $P[X \leq x]$ , otherwise, $P[X > x]$
<code>n</code>	number of random values to return
<code>max.value</code>	a constant, set to the default value of 10000, It is used in the <code>q</code> function which numerically calculates how far the algorithm should look for <code>q</code> . Maybe for zipf data the values has to increase at a considerable computational cost.
<code>sigma.link</code>	the link function for the parameter <code>sigma</code> with default <code>logit</code>
<code>sigma</code>	a vector of probabilities of zero

### Details

The probability density for the zipf distribution, ZIPF, is:

$$f(y|\mu) = \frac{y^{-(\mu+1)}}{\zeta(\mu+1)}$$

for  $y = 1, 2, \dots, \infty$ ,  $\mu > 0$  and where  $\zeta()$  is the (Reimann) zeta function.

The distribution has mean  $\zeta(\mu)/\zeta(\mu+1)$  and variance  $\zeta(\mu+1)\zeta(\mu-1) - [\zeta(\mu)]^2/[\zeta(\mu+1)]^2$ .

### Value

The function `ZIPF()` returns a `gamlss.family` object which can be used to fit a zipf distribution in the `gamlss()` function.

### Note

Because the zipf distribution has very long tails the `max.value` in the `q` and `r`, may need to increase.

### Author(s)

Mikis Stasinopoulos and Bob Rigby

## References

- N. L. Johnson, A. W. Kemp, and S. Kotz. (2005) *Univariate Discrete Distributions*. Wiley, New York, 3rd edition.
- Thomas W. Yee (2017). VGAM: Vector Generalized Linear and Additive Models. R package version 1.0-3. <https://CRAN.R-project.org/package=VGAM>
- Rigby, R. A. and Stasinopoulos D. M. (2005). Generalized additive models for location, scale and shape,(with discussion), *Appl. Statist.*, **54**, part 3, pp 507-554.
- Stasinopoulos D. M. Rigby R.A. (2007) Generalized additive models for location scale and shape (GAMLSS) in R. *Journal of Statistical Software*, Vol. **23**, Issue 7, Dec 2007, <http://www.jstatsoft.org/v23/i07>.
- Stasinopoulos D. M., Rigby R.A., Heller G., Voudouris V., and De Bastiani F., (2017) *Flexible Regression and Smoothing: Using GAMLSS in R*, Chapman and Hall/CRC.

## See Also

[PO](#), [LG](#), [GEOM](#), [YULE](#)

## Examples

```
# ZIPF
par(mfrow=c(2,2))
y<-seq(1,20,1)
plot(y, dZIPF(y), type="h")
q <- seq(1, 20, 1)
plot(q, pZIPF(q), type="h")
p<-seq(0.0001,0.999,0.05)
plot(p , qZIPF(p), type="s")
dat <- rZIPF(100)
hist(dat)
# ZAZIPF
y<-seq(0,20,1)
plot(y, dZAZIPF(y, mu=.9, sigma=.1), type="h")
q <- seq(1, 20, 1)
plot(q, pZAZIPF(q, mu=.9, sigma=.1), type="h")
p<-seq(0.0001,0.999,0.05)
plot(p, qZAZIPF(p, mu=.9, sigma=.1), type="s")
dat <- rZAZIPF(100, mu=.9, sigma=.1)
hist(dat)
```

# Index

## \*Topic **distribution**

BB, 8  
BCCG, 10  
BCPE, 13  
BCT, 15  
BE, 18  
BEINF, 20  
BEOI, 24  
BEZI, 26  
BI, 29  
BNB, 31  
DBI, 34  
DBURR12, 36  
DEL, 38  
DPO, 40  
EGB2, 42  
exGAUS, 44  
EXP, 46  
flexDist, 48  
GA, 49  
GB1, 55  
GB2, 57  
gen.Family, 59  
GEOM, 61  
GG, 63  
GIG, 65  
GPO, 67  
GT, 69  
GU, 71  
hazardFun, 72  
IG, 74  
IGAMMA, 75  
JSU, 77  
JSUo, 79  
LG, 82  
LNO, 84  
LO, 87  
LOGITNO, 88  
LQNO, 90

MN3, 96  
NBF, 98  
NBI, 100  
NBII, 102  
NET, 104  
NO, 106  
NO2, 108  
NOF, 110  
PARETO2, 112  
PE, 114  
PIG, 116  
PO, 118  
RG, 120  
RGE, 122  
SEP, 124  
SEP1, 127  
SHASH, 129  
SI, 133  
SICHEL, 135  
SIMPLEX, 138  
SN1, 139  
SN2, 141  
ST1, 143  
TF, 146  
WARING, 148  
WEI, 150  
WEI2, 152  
WEI3, 154  
YULE, 156  
ZABB, 157  
ZABI, 159  
ZAGA, 161  
ZAIG, 163  
ZANBI, 166  
ZAP, 168  
ZIP, 169  
ZIP2, 171  
ZIPF, 173

## \*Topic **package**



- gamlss.dist-package, 3
- \*Topic **regression**
  - BB, 8
  - BCCG, 10
  - BCPE, 13
  - BCT, 15
  - BE, 18
  - BEINF, 20
  - BEOI, 24
  - BEZI, 26
  - BI, 29
  - BNB, 31
  - checklink, 33
  - DBI, 34
  - DBURR12, 36
  - DEL, 38
  - DPO, 40
  - EGB2, 42
  - exGAUS, 44
  - EXP, 46
  - flexDist, 48
  - GA, 49
  - gamlss.family, 51
  - GB1, 55
  - GB2, 57
  - gen.Family, 59
  - GEOM, 61
  - GG, 63
  - GIG, 65
  - GPO, 67
  - GT, 69
  - GU, 71
  - hazardFun, 72
  - IG, 74
  - IGAMMA, 75
  - JSU, 77
  - JSUo, 79
  - LG, 82
  - LNO, 84
  - LO, 87
  - LOGITNO, 88
  - LQNO, 90
  - make.link.gamlss, 92
  - MN3, 96
  - NBF, 98
  - NBI, 100
  - NBII, 102
  - NET, 104
  - NO, 106
  - NO2, 108
  - NOF, 110
  - PARETO2, 112
  - PE, 114
  - PIG, 116
  - PO, 118
  - RG, 120
  - RGE, 122
  - SEP, 124
  - SEP1, 127
  - SHASH, 129
  - SI, 133
  - SICHEL, 135
  - SIMPLEX, 138
  - SN1, 139
  - SN2, 141
  - ST1, 143
  - TF, 146
  - WARING, 148
  - WEI, 150
  - WEI2, 152
  - WEI3, 154
  - YULE, 156
  - ZABB, 157
  - ZABI, 159
  - ZAGA, 161
  - ZAIG, 163
  - ZANBI, 166
  - ZAP, 168
  - ZIP, 169
  - ZIP2, 171
  - ZIPF, 173
- as.family (gamlss.family), 51
- as.gamlss.family (gamlss.family), 51
- BB, 8, 36, 52, 54
- BCCG, 10, 18, 45, 52, 54, 84, 86
- BCCGo (BCCG), 10
- BCCGuntr (BCCG), 10
- BCPE, 12, 13, 18, 52, 54, 106, 116
- BCPEo (BCPE), 13
- BCPEuntr (BCPE), 13
- BCT, 12, 15, 15, 43, 52, 54, 56, 58, 70, 79, 81, 126, 132
- BCTo (BCT), 15
- BCTuntr (BCT), 15
- BE, 18, 19, 23, 52, 54

- BEINF, [19](#), [20](#), [52](#), [54](#)  
 BEINF0 (BEINF), [20](#)  
 BEINF1 (BEINF), [20](#)  
 BEo, [23](#)  
 BEo (BE), [18](#)  
 BEOI, [20](#), [23](#), [24](#), [26](#), [52](#)  
 BEZI, [20](#), [23](#), [26](#), [29](#), [52](#)  
 BI, [9](#), [29](#), [36](#), [52](#), [54](#), [97](#), [161](#)  
 BNB, [31](#), [52](#)
- checkBCPE (BCPE), [13](#)  
 checklink, [33](#)
- dBB (BB), [8](#)  
 dBCCG (BCCG), [10](#)  
 dBCCGo (BCCG), [10](#)  
 dBCPE (BCPE), [13](#)  
 dBCPEo (BCPE), [13](#)  
 dBCT (BCT), [15](#)  
 dBCTo (BCT), [15](#)  
 dBE (BE), [18](#)  
 dBEINF (BEINF), [20](#)  
 dBEINF0 (BEINF), [20](#)  
 dBEINF1 (BEINF), [20](#)  
 dBEo (BE), [18](#)  
 dBEOI (BEOI), [24](#)  
 dBEZI (BEZI), [26](#)  
 DBI, [34](#), [52](#)  
 dBI (BI), [29](#)  
 dBNB (BNB), [31](#)  
 DBURR12, [36](#), [52](#)  
 dDBI (DBI), [34](#)  
 dDBURR12 (DBURR12), [36](#)  
 dDEL (DEL), [38](#)  
 dDPO (DPO), [40](#)  
 dEGB2 (EGB2), [42](#)  
 DEL, [38](#), [52](#)  
 dexGAUS (exGAUS), [44](#)  
 dEXP (EXP), [46](#)  
 dGA (GA), [49](#)  
 dGB1 (GB1), [55](#)  
 dGB2 (GB2), [57](#)  
 dGEOM (GEOM), [61](#)  
 dGEOMo (GEOM), [61](#)  
 dGG (GG), [63](#)  
 dGIG (GIG), [65](#)  
 dGP (GB2), [57](#)  
 dGPO (GPO), [67](#)  
 dGT (GT), [69](#)
- dGU (GU), [71](#)  
 dIG (IG), [74](#)  
 dIGAMMA (IGAMMA), [75](#)  
 dJSU (JSU), [77](#)  
 dJSUo (JSUo), [79](#)  
 dLG (LG), [82](#)  
 dLNO (LNO), [84](#)  
 dLO (LO), [87](#)  
 dLOGITNO (LOGITNO), [88](#)  
 dLOGNO (LNO), [84](#)  
 dLOGNO2 (LNO), [84](#)  
 dLQNO (LQNO), [90](#)  
 dMN3 (MN3), [96](#)  
 dMN4 (MN3), [96](#)  
 dMN5 (MN3), [96](#)  
 dNBF (NBF), [98](#)  
 dNBI (NBI), [100](#)  
 dNBII (NBII), [102](#)  
 dNET (NET), [104](#)  
 dNO (NO), [106](#)  
 dNO2 (NO2), [108](#)  
 dNOF (NOF), [110](#)  
 dPARETO2 (PARETO2), [112](#)  
 dPARETO2o (PARETO2), [112](#)  
 dPE (PE), [114](#)  
 dPE2 (PE), [114](#)  
 dPIG (PIG), [116](#)  
 DPO, [38](#), [40](#), [52](#), [68](#)  
 dPO (PO), [118](#)  
 dRG (RG), [120](#)  
 dRGE (RGE), [122](#)  
 dSEP (SEP), [124](#)  
 dSEP1 (SEP1), [127](#)  
 dSEP2 (SEP1), [127](#)  
 dSEP3 (SEP1), [127](#)  
 dSEP4 (SEP1), [127](#)  
 dSHASH (SHASH), [129](#)  
 dSHASHo (SHASH), [129](#)  
 dSHASHo2 (SHASH), [129](#)  
 dSI (SI), [133](#)  
 dSICHEL (SICHEL), [135](#)  
 dSIMPLEX (SIMPLEX), [138](#)  
 dSN1 (SN1), [139](#)  
 dSN2 (SN2), [141](#)  
 dSST (ST1), [143](#)  
 dST1 (ST1), [143](#)  
 dST2 (ST1), [143](#)  
 dST3 (ST1), [143](#)

- dST3C (ST1), 143  
 dST4 (ST1), 143  
 dST5 (ST1), 143  
 dTF (TF), 146  
 dTF2 (TF), 146  
 dWARING (WARING), 148  
 dWEI (WEI), 150  
 dWEI2 (WEI2), 152  
 dWEI3 (WEI3), 154  
 dYULE (YULE), 156  
 dZABB (ZABB), 157  
 dZABI (ZABI), 159  
 dZABNB (BNB), 31  
 dZAGA (ZAGA), 161  
 dZAIG (ZAIG), 163  
 dZALG (LG), 82  
 dZANBI (ZANBI), 166  
 dZAP (ZAP), 168  
 dZAPIG (PIG), 116  
 dZASICHEL (SICHEL), 135  
 dZAZIPF (ZIPF), 173  
 dZIBB (ZABB), 157  
 dZIBI (ZABI), 159  
 dZIBNB (BNB), 31  
 dZINBF (NBF), 98  
 dZINBI (ZANBI), 166  
 dZIP (ZIP), 169  
 dZIP2 (ZIP2), 171  
 dZIPF (ZIPF), 173  
 dZIPIG (PIG), 116  
 dZISICHEL (SICHEL), 135
- EGB2, 42, 52  
 exGAUS, 44, 52  
 EXP, 46, 52
- Family (gen.Family), 59  
 fittedMN (MN3), 96  
 flexDist, 48
- GA, 45, 49, 52, 54, 65, 75, 77, 163  
 gamlss.dist (gamlss.dist-package), 3  
 gamlss.dist-package, 3  
 gamlss.family, 7, 9, 12, 15, 18, 19, 23, 26,  
     29, 30, 34, 38, 40, 43, 45, 47, 51, 51,  
     56, 58, 63, 65, 67, 68, 70, 72, 73, 75,  
     77, 79, 81, 83, 84, 86, 88, 90, 94, 97,  
     102, 104, 106, 108, 110, 111, 113,  
     116, 118, 120, 122, 124, 126, 129,  
     132, 134, 137, 140, 142, 146, 148,  
     149, 151, 153–155, 157, 159, 161,  
     163, 165, 167, 169, 171, 173
- GB1, 19, 52, 55  
 GB2, 52, 57  
 gen.Family, 59  
 gen.hazard (hazardFun), 72  
 GEOM, 52, 61, 175  
 GEOMo, 52  
 GEOMo (GEOM), 61  
 get\_C (DPO), 40  
 GetBI\_C (DBI), 34  
 GG, 52, 63  
 GIG, 52, 65, 75  
 GP (GB2), 57  
 GPO, 67  
 GT, 52, 69  
 GU, 52, 54, 71
- hazardFun, 72  
 histSmo, 49
- IG, 45, 52, 54, 67, 74, 165  
 IGAMMA, 52, 75
- JSU, 43, 52, 54, 56, 58, 70, 77, 81, 126, 132  
 JSUo, 79, 79
- LG, 52, 82, 175  
 LNO, 45, 52, 54, 84  
 LO, 52, 54, 87  
 LOGITNO, 19, 52, 88  
 LOGNO, 52, 90  
 LOGNO (LNO), 84  
 LOGNO2 (LNO), 84  
 LQNO, 52, 90
- make.link.gamlss, 92  
 meanBEINF (BEINF), 20  
 meanBEINF0 (BEINF), 20  
 meanBEINF1 (BEINF), 20  
 meanBEOI (BEOI), 24  
 meanBEZI (BEZI), 26  
 meanZAGA (ZAGA), 161  
 meanZAIG (ZAIG), 163  
 MN3, 96  
 MN4 (MN3), 96  
 MN5 (MN3), 96  
 MULTIN (MN3), 96

- NBF, [52](#), [98](#)  
 NBI, [33](#), [52](#), [54](#), [100](#), [100](#), [104](#), [118](#), [120](#), [134](#),  
     [159](#), [167](#)  
 NBII, [33](#), [52](#), [54](#), [100](#), [102](#), [102](#), [118](#), [120](#), [134](#),  
     [159](#), [167](#)  
 NET, [52](#), [104](#)  
 NO, [52](#), [54](#), [88](#), [92](#), [106](#), [110](#), [111](#)  
 NO2, [92](#), [108](#), [108](#), [111](#)  
 NOF, [52](#), [92](#), [110](#)
- own.linkfun (make.link.gamlss), [92](#)  
 own.linkinv (make.link.gamlss), [92](#)  
 own.mu.eta (make.link.gamlss), [92](#)  
 own.valideta (make.link.gamlss), [92](#)
- PARETO2, [53](#), [112](#)  
 PARETO2o, [53](#)  
 PARETO2o (PARETO2), [112](#)  
 pBB (BB), [8](#)  
 pBCCG (BCCG), [10](#)  
 pBCCGo (BCCG), [10](#)  
 pBCPE (BCPE), [13](#)  
 pBCPEo (BCPE), [13](#)  
 pBCT (BCT), [15](#)  
 pBCTo (BCT), [15](#)  
 pBE (BE), [18](#)  
 pBEINF (BEINF), [20](#)  
 pBEINF0 (BEINF), [20](#)  
 pBEINF1 (BEINF), [20](#)  
 pBEo (BE), [18](#)  
 pBEOI (BEOI), [24](#)  
 pBEZI (BEZI), [26](#)  
 pBI (BI), [29](#)  
 pBNB (BNB), [31](#)  
 pDBI (DBI), [34](#)  
 pDBURR12 (DBURR12), [36](#)  
 pDEL (DEL), [38](#)  
 pDPO (DPO), [40](#)  
 PE, [53](#), [54](#), [114](#)  
 PE2, [53](#)  
 PE2 (PE), [114](#)  
 pEGB2 (EGB2), [42](#)  
 pexGAUS (exGAUS), [44](#)  
 pEXP (EXP), [46](#)  
 pGA (GA), [49](#)  
 pGB1 (GB1), [55](#)  
 pGB2 (GB2), [57](#)  
 pGEOM (GEOM), [61](#)  
 pGEOMo (GEOM), [61](#)
- pGG (GG), [63](#)  
 pGIG (GIG), [65](#)  
 pGP (GB2), [57](#)  
 pGPO (GPO), [67](#)  
 pGT (GT), [69](#)  
 pGU (GU), [71](#)  
 PIG, [53](#), [54](#), [102](#), [104](#), [116](#), [134](#), [137](#)  
 pIG (IG), [74](#)  
 pIGAMMA (IGAMMA), [75](#)  
 pJSU (JSU), [77](#)  
 pJSUo (JSUo), [79](#)  
 pLG (LG), [82](#)  
 pLNO (LNO), [84](#)  
 pLO (LO), [87](#)  
 pLOGITNO (LOGITNO), [88](#)  
 pLOGNO (LNO), [84](#)  
 pLOGNO2 (LNO), [84](#)  
 plotBEINF (BEINF), [20](#)  
 plotBEINF0 (BEINF), [20](#)  
 plotBEINF1 (BEINF), [20](#)  
 plotBEOI (BEOI), [24](#)  
 plotBEZI (BEZI), [26](#)  
 plotZAGA (ZAGA), [161](#)  
 plotZAIG (ZAIG), [163](#)  
 pLQNO (LQNO), [90](#)  
 pMN3 (MN3), [96](#)  
 pMN4 (MN3), [96](#)  
 pMN5 (MN3), [96](#)  
 pNBF (NBF), [98](#)  
 pNBI (NBI), [100](#)  
 pNBII (NBII), [102](#)  
 pNET (NET), [104](#)  
 pNO (NO), [106](#)  
 pNO2 (NO2), [108](#)  
 pNOF (NOF), [110](#)  
 PO, [42](#), [53](#), [54](#), [68](#), [83](#), [118](#), [169](#), [171](#), [175](#)  
 pPARETO2 (PARETO2), [112](#)  
 pPARETO2o (PARETO2), [112](#)  
 pPE (PE), [114](#)  
 pPE2 (PE), [114](#)  
 pPIG (PIG), [116](#)  
 pPO (PO), [118](#)  
 pRG (RG), [120](#)  
 pRGE (RGE), [122](#)  
 print.gamlss.family (gamlss.family), [51](#)  
 pSEP (SEP), [124](#)  
 pSEP1 (SEP1), [127](#)  
 pSEP2 (SEP1), [127](#)

- pSEP3 (SEP1), 127  
 pSEP4 (SEP1), 127  
 pSHASH (SHASH), 129  
 pSHASHo (SHASH), 129  
 pSHASHo2 (SHASH), 129  
 pSI (SI), 133  
 pSICHEL (SICHEL), 135  
 pSIMPLEX (SIMPLEX), 138  
 pSN1 (SN1), 139  
 pSN2 (SN2), 141  
 pSST (ST1), 143  
 pST1 (ST1), 143  
 pST2 (ST1), 143  
 pST3 (ST1), 143  
 pST3C (ST1), 143  
 pST4 (ST1), 143  
 pST5 (ST1), 143  
 pTF (TF), 146  
 pTF2 (TF), 146  
 pWARING (WARING), 148  
 pWEI (WEI), 150  
 pWEI2 (WEI2), 152  
 pWEI3 (WEI3), 154  
 pYULE (YULE), 156  
 pZABB (ZABB), 157  
 pZABI (ZABI), 159  
 pZABNB (BNB), 31  
 pZAGA (ZAGA), 161  
 pZAIG (ZAIG), 163  
 pZALG (LG), 82  
 pZANBI (ZANBI), 166  
 pZAP (ZAP), 168  
 pZAPIG (PIG), 116  
 pZASICHEL (SICHEL), 135  
 pZAZIPF (ZIPF), 173  
 pZIBB (ZABB), 157  
 pZIBI (ZABI), 159  
 pZIBNB (BNB), 31  
 pZINBF (NBF), 98  
 pZINBI (ZANBI), 166  
 pZIP (ZIP), 169  
 pZIP2 (ZIP2), 171  
 pZIPF (ZIPF), 173  
 pZIPIG (PIG), 116  
 pZISICHEL (SICHEL), 135  
  
 qBB (BB), 8  
 qBCCG (BCCG), 10  
 qBCCGo (BCCG), 10  
  
 qBCPE (BCPE), 13  
 qBCPEo (BCPE), 13  
 qBCT (BCT), 15  
 qBCTo (BCT), 15  
 qBE (BE), 18  
 qBEINF (BEINF), 20  
 qBEINF0 (BEINF), 20  
 qBEINF1 (BEINF), 20  
 qBEo (BE), 18  
 qBEOI (BEOI), 24  
 qBEZI (BEZI), 26  
 qBI (BI), 29  
 qBNB (BNB), 31  
 qDBI (DBI), 34  
 qDBURR12 (DBURR12), 36  
 qDEL (DEL), 38  
 qDPO (DPO), 40  
 qEGB2 (EGB2), 42  
 qexGAUS (exGAUS), 44  
 qEXP (EXP), 46  
 qGA (GA), 49  
 qGB1 (GB1), 55  
 qGB2 (GB2), 57  
 qGEOM (GEOM), 61  
 qGEOMo (GEOM), 61  
 qGG (GG), 63  
 qGIG (GIG), 65  
 qGP (GB2), 57  
 qGPO (GPO), 67  
 qGT (GT), 69  
 qGU (GU), 71  
 qIG (IG), 74  
 qIGAMMA (IGAMMA), 75  
 qJSU (JSU), 77  
 qJSUo (JSUo), 79  
 qLG (LG), 82  
 qLNO (LNO), 84  
 qLO (LO), 87  
 qLOGITNO (LOGITNO), 88  
 qLOGNO (LNO), 84  
 qLOGNO2 (LNO), 84  
 qLQNO (LQNO), 90  
 qMN3 (MN3), 96  
 qMN4 (MN3), 96  
 qMN5 (MN3), 96  
 qNBF (NBF), 98  
 qNBI (NBI), 100  
 qNBII (NBII), 102

- qNET (NET), 104
- qNO (NO), 106
- qNO2 (NO2), 108
- qNOF (NOF), 110
- qPARETO2 (PARETO2), 112
- qPARETO2o (PARETO2), 112
- qPE (PE), 114
- qPE2 (PE), 114
- qPIG (PIG), 116
- qPO (PO), 118
- qRG (RG), 120
- qRGE (RGE), 122
- qSEP (SEP), 124
- qSEP1 (SEP1), 127
- qSEP2 (SEP1), 127
- qSEP3 (SEP1), 127
- qSEP4 (SEP1), 127
- qSHASH (SHASH), 129
- qSHASHo (SHASH), 129
- qSHASHo2 (SHASH), 129
- qSI (SI), 133
- qSICHEL (SICHEL), 135
- qSIMPLEX (SIMPLEX), 138
- qSN1 (SN1), 139
- qSN2 (SN2), 141
- qSST (ST1), 143
- qST1 (ST1), 143
- qST2 (ST1), 143
- qST3 (ST1), 143
- qST3C (ST1), 143
- qST4 (ST1), 143
- qST5 (ST1), 143
- qTF (TF), 146
- qTF2 (TF), 146
- qWARING (WARING), 148
- qWEI (WEI), 150
- qWEI2 (WEI2), 152
- qWEI3 (WEI3), 154
- qYULE (YULE), 156
- qZABB (ZABB), 157
- qZABI (ZABI), 159
- qZABNB (BNB), 31
- qZAGA (ZAGA), 161
- qZAIG (ZAIG), 163
- qZALG (LG), 82
- qZANBI (ZANBI), 166
- qZAP (ZAP), 168
- qZAPIG (PIG), 116
- qZASICHEL (SICHEL), 135
- qZAZIPF (ZIPF), 173
- qZIBB (ZABB), 157
- qZIBI (ZABI), 159
- qZIBNB (BNB), 31
- qZINBF (NBF), 98
- qZINBI (ZANBI), 166
- qZIP (ZIP), 169
- qZIP2 (ZIP2), 171
- qZIPF (ZIPF), 173
- qZIPIG (PIG), 116
- qZISICHEL (SICHEL), 135
- rBB (BB), 8
- rBCCG (BCCG), 10
- rBCCGo (BCCG), 10
- rBCPE (BCPE), 13
- rBCPEo (BCPE), 13
- rBCT (BCT), 15
- rBCTo (BCT), 15
- rBE (BE), 18
- rBEINF (BEINF), 20
- rBEINF0 (BEINF), 20
- rBEINF1 (BEINF), 20
- rBEo (BE), 18
- rBEOI (BEOI), 24
- rBEZI (BEZI), 26
- rBI (BI), 29
- rBNB (BNB), 31
- rDBI (DBI), 34
- rDBURR12 (DBURR12), 36
- rDEL (DEL), 38
- rDPO (DPO), 40
- rEGB2 (EGB2), 42
- rexGAUS (exGAUS), 44
- rEXP (EXP), 46
- RG, 53, 54, 72, 120
- rGA (GA), 49
- rGB1 (GB1), 55
- rGB2 (GB2), 57
- RGE, 53, 122
- rGEOM (GEOM), 61
- rGEOMo (GEOM), 61
- rGG (GG), 63
- rGIG (GIG), 65
- rGP (GB2), 57
- rGPO (GPO), 67
- rGT (GT), 69
- rGU (GU), 71

- rIG (IG), 74
- rIGAMMA (IGAMMA), 75
- rJSU (JSU), 77
- rJSUo (JSUo), 79
- rLG (LG), 82
- rLNO (LNO), 84
- rLO (LO), 87
- rLOGITNO (LOGITNO), 88
- rLOGNO (LNO), 84
- rLOGNO2 (LNO), 84
- rLQNO (LQNO), 90
- rMN3 (MN3), 96
- rMN4 (MN3), 96
- rMN5 (MN3), 96
- rNBF (NBF), 98
- rNBI (NBI), 100
- rNBII (NBII), 102
- rNET (NET), 104
- rNO (NO), 106
- rNO2 (NO2), 108
- rNOF (NOF), 110
- rPARETO2 (PARETO2), 112
- rPARETO2o (PARETO2), 112
- rPE (PE), 114
- rPE2 (PE), 114
- rPIG (PIG), 116
- rPO (PO), 118
- rRG (RG), 120
- rRGE (RGE), 122
- rSEP (SEP), 124
- rSEP1 (SEP1), 127
- rSEP2 (SEP1), 127
- rSEP3 (SEP1), 127
- rSEP4 (SEP1), 127
- rSHASH (SHASH), 129
- rSHASHo (SHASH), 129
- rSHASHo2 (SHASH), 129
- rSI (SI), 133
- rSICHEL (SICHEL), 135
- rSIMPLEX (SIMPLEX), 138
- rSN1 (SN1), 139
- rSN2 (SN2), 141
- rSST (ST1), 143
- rST1 (ST1), 143
- rST2 (ST1), 143
- rST3 (ST1), 143
- rST3C (ST1), 143
- rST4 (ST1), 143
- rST5 (ST1), 143
- rTF (TF), 146
- rTF2 (TF), 146
- rWARING (WARING), 148
- rWEI (WEI), 150
- rWEI2 (WEI2), 152
- rWEI3 (WEI3), 154
- rYULE (YULE), 156
- rZABB (ZABB), 157
- rZABI (ZABI), 159
- rZABNB (BNB), 31
- rZAGA (ZAGA), 161
- rZAIG (ZAIG), 163
- rZALG (LG), 82
- rZANBI (ZANBI), 166
- rZAP (ZAP), 168
- rZAPIG (PIG), 116
- rZASICHEL (SICHEL), 135
- rZAZIPF (ZIPF), 173
- rZIBB (ZABB), 157
- rZIBI (ZABI), 159
- rZIBNB (BNB), 31
- rZINBF (NBF), 98
- rZINBI (ZANBI), 166
- rZIP (ZIP), 169
- rZIP2 (ZIP2), 171
- rZIPF (ZIPF), 173
- rZIPIG (PIG), 116
- rZISICHEL (SICHEL), 135
- SEP, 124, 129
- SEP1, 53, 127, 146
- SEP2, 53
- SEP2 (SEP1), 127
- SEP3, 53
- SEP3 (SEP1), 127
- SEP4, 53
- SEP4 (SEP1), 127
- SHASH, 53, 129, 146
- SHASHo, 53
- SHASHo (SHASH), 129
- SHASHo2 (SHASH), 129
- show.link (make.link.gamlss), 92
- SI, 40, 53, 102, 104, 118, 120, 133, 137
- SICHEL, 40, 53, 118, 120, 135
- SIMPLEX, 53, 138
- SN1, 139
- SN2, 141
- SST (ST1), 143

- ST1, [53](#), [143](#)  
 ST2, [53](#)  
 ST2 (ST1), [143](#)  
 ST3, [53](#)  
 ST3 (ST1), [143](#)  
 ST3C (ST1), [143](#)  
 ST4, [53](#)  
 ST4 (ST1), [143](#)  
 ST5, [53](#)  
 ST5 (ST1), [143](#)
- TF, [53](#), [54](#), [88](#), [146](#)  
 TF2 (TF), [146](#)  
 tofyS (SI), [133](#)  
 tofySICHEL (SICHEL), [135](#)
- VSICHEL (SICHEL), [135](#)
- WARING, [53](#), [148](#)  
 WEI, [53](#), [54](#), [150](#), [153–155](#)  
 WEI2, [53](#), [54](#), [151](#), [152](#), [154](#), [155](#)  
 WEI3, [53](#), [151](#), [153](#), [154](#)
- YULE, [53](#), [156](#), [175](#)
- ZABB, [157](#)  
 ZABI, [30](#), [53](#), [159](#)  
 ZABNB, [53](#)  
 ZABNB (BNB), [31](#)  
 ZAGA, [161](#)  
 ZAIG, [53](#), [163](#), [163](#)  
 ZALG, [53](#), [169](#)  
 ZALG (LG), [82](#)  
 ZANBI, [53](#), [166](#)  
 ZAP, [53](#), [83](#), [168](#)  
 ZAPIG (PIG), [116](#)  
 ZASICHEL, [53](#)  
 ZASICHEL (SICHEL), [135](#)  
 ZAZIPF, [53](#)  
 ZAZIPF (ZIPF), [173](#)  
 zetaP (ZIPF), [173](#)  
 ZIBB (ZABB), [157](#)  
 ZIBI, [30](#), [53](#)  
 ZIBI (ZABI), [159](#)  
 ZIBNB, [53](#)  
 ZIBNB (BNB), [31](#)  
 ZINBF (NBF), [98](#)  
 ZINBI, [53](#)  
 ZINBI (ZANBI), [166](#)
- ZIP, [53](#), [54](#), [169](#), [169](#), [173](#)  
 ZIP2, [53](#), [169](#), [171](#), [171](#)  
 ZIPF, [53](#), [173](#)  
 ZIPIG, [53](#)  
 ZIPIG (PIG), [116](#)  
 ZISICHEL, [53](#)  
 ZISICHEL (SICHEL), [135](#)