Package ‘gamlss.dist’

November 13, 2021

Type Package
Title Distributions for Generalized Additive Models for Location Scale and Shape
Version 6.0-1
Date 2021-11-13
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 https://www.gamlss.com/
Depends R (>= 2.15.0), MASS, graphics, stats, methods, grDevices
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 2021-11-13 12:30:02 UTC

R topics documented:

gamlss.dist-package .............................. 4
BB ............................................. 8
BCCG .......................................... 11
BCPE .......................................... 13
BCT ........................................... 16
BE ............................................. 18
BEINF ......................................... 20
BEOI ........................................... 24
BEZI ........................................... 27
BI .............................................. 30
BNB ........................................... 31
checklink ...................................... 34
count_1_31 ..................................... 35
DBI ............................................. 39
DBURR12 ...................................... 41
DEL ............................................ 43
DPO ............................................ 45
EGB2 ........................................... 47
exGAUS ........................................ 49
EXP ........................................... 51
flexDist ......................................... 53
GA ............................................. 54
GAF ........................................... 56
gamlss.family ................................... 58
GB1 ............................................. 62
GB2 ............................................. 64
gen.Family ....................................... 66
GEOM .......................................... 68
GG .............................................. 71
GIG ............................................ 73
GPO ............................................ 75
GT ................................................ 76
GU ............................................. 78
hazardFun ....................................... 80
IG .............................................. 81
IGAMMA ........................................ 83
JSU ............................................ 85
JSUo ........................................... 87
LG .............................................. 89
LNO ............................................ 91
LO ............................................. 94
LOGITNO ...................................... 96
LQNO .......................................... 98
make.link.gamlss ................................. 100
### Topics Documented

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>MN3</td>
<td>104</td>
</tr>
<tr>
<td>momentSK</td>
<td>106</td>
</tr>
<tr>
<td>NBF</td>
<td>109</td>
</tr>
<tr>
<td>NBI</td>
<td>112</td>
</tr>
<tr>
<td>NBII</td>
<td>114</td>
</tr>
<tr>
<td>NET</td>
<td>116</td>
</tr>
<tr>
<td>NO</td>
<td>118</td>
</tr>
<tr>
<td>NO2</td>
<td>119</td>
</tr>
<tr>
<td>NOF</td>
<td>121</td>
</tr>
<tr>
<td>PARETO2</td>
<td>124</td>
</tr>
<tr>
<td>PE</td>
<td>126</td>
</tr>
<tr>
<td>PIG</td>
<td>129</td>
</tr>
<tr>
<td>PO</td>
<td>131</td>
</tr>
<tr>
<td>RG</td>
<td>133</td>
</tr>
<tr>
<td>RGE</td>
<td>135</td>
</tr>
<tr>
<td>SEP</td>
<td>137</td>
</tr>
<tr>
<td>SEP1</td>
<td>140</td>
</tr>
<tr>
<td>SHASH</td>
<td>142</td>
</tr>
<tr>
<td>SI</td>
<td>146</td>
</tr>
<tr>
<td>SICHEL</td>
<td>148</td>
</tr>
<tr>
<td>SIMPLEX</td>
<td>151</td>
</tr>
<tr>
<td>SN1</td>
<td>152</td>
</tr>
<tr>
<td>SN2</td>
<td>154</td>
</tr>
<tr>
<td>ST1</td>
<td>156</td>
</tr>
<tr>
<td>TF</td>
<td>159</td>
</tr>
<tr>
<td>WARING</td>
<td>161</td>
</tr>
<tr>
<td>WEI</td>
<td>163</td>
</tr>
<tr>
<td>WEI2</td>
<td>165</td>
</tr>
<tr>
<td>WEI3</td>
<td>167</td>
</tr>
<tr>
<td>YULE</td>
<td>169</td>
</tr>
<tr>
<td>ZABB</td>
<td>170</td>
</tr>
<tr>
<td>ZABI</td>
<td>172</td>
</tr>
<tr>
<td>ZAGA</td>
<td>175</td>
</tr>
<tr>
<td>ZAIG</td>
<td>177</td>
</tr>
<tr>
<td>ZANBI</td>
<td>179</td>
</tr>
<tr>
<td>ZAP</td>
<td>181</td>
</tr>
<tr>
<td>ZIP</td>
<td>183</td>
</tr>
<tr>
<td>ZIP2</td>
<td>185</td>
</tr>
<tr>
<td>ZIPF</td>
<td>187</td>
</tr>
</tbody>
</table>

**Index** 190
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: 6.0-1
Date: 2021-11-13
Authors@R: c(person("Mikis", "Stasinopoulos", role = c("aut", "cre", "cph"), email = "d.stasinopoulos@londonmet.ac.uk"), ... "Stocker", role = "ctb"), person("Jens", "Lichter", role = "ctb"), person("Stanislaus", "Stadlmann", role = "ctb")
Description: A set of distributions which can be used for modelling the response variables in Generalized Additive Models for Location Scale and Shape. 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: https://www.gamlss.com/
Depends: R (>= 2.15.0), MASS, graphics, stats, methods, grDevices
Repository: CRAN
Author: Mikis Stasinopoulos [aut, cre, cph], Robert Rigby [aut], Calliope Akantziliotou [ctb], Vlasios Voudouris [ctb], Christos Argyropoulos [ctb], Almond Stocker [ctb], Jens Lichter [ctb], Stanislaus Stadlmann [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
BEIO
- The one-inflated beta distribution for fitting a GAMLSS
BEZI
- The zero-inflated beta distribution for fitting
<table>
<thead>
<tr>
<th>Distribution</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BI</td>
<td>Binomial distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>BNB</td>
<td>Beta Negative Binomial distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>DBI</td>
<td>The Double binomial distribution</td>
</tr>
<tr>
<td>DBURR12</td>
<td>The Discrete Burr type XII distribution for fitting a GAMLSS model</td>
</tr>
<tr>
<td>DEL</td>
<td>The Delaporte distribution for fitting a GAMLSS model</td>
</tr>
<tr>
<td>DPO</td>
<td>The Double Poisson distribution</td>
</tr>
<tr>
<td>EGB2</td>
<td>The exponential generalized Beta type 2 distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>EXP</td>
<td>Exponential distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>GA</td>
<td>Gamma distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>GAF</td>
<td>The Gamma distribution family</td>
</tr>
<tr>
<td>GB1</td>
<td>The generalized Beta type 1 distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>GB2</td>
<td>The generalized Beta type 2 and generalized Pareto distributions for fitting a GAMLSS</td>
</tr>
<tr>
<td>GEOM</td>
<td>Geometric distribution for fitting a GAMLSS model</td>
</tr>
<tr>
<td>GG</td>
<td>Generalized Gamma distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>GIG</td>
<td>Generalized Inverse Gaussian distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>GPO</td>
<td>The generalised Poisson distribution</td>
</tr>
<tr>
<td>GT</td>
<td>The generalized t distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>GU</td>
<td>The Gumbel distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>IG</td>
<td>Inverse Gaussian distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>IGAMMA</td>
<td>Inverse Gamma distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>JSU</td>
<td>The Johnson's Su distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>JSUo</td>
<td>The original Johnson's Su distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>LG</td>
<td>Logarithmic and zero adjusted logarithmic distributions for fitting a GAMLSS model</td>
</tr>
<tr>
<td>LNO</td>
<td>Log Normal distribution for fitting in GAMLSS</td>
</tr>
<tr>
<td>L0</td>
<td>Logistic distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>LOGITNO</td>
<td>Logit Normal distribution for fitting in GAMLSS</td>
</tr>
<tr>
<td>LQNO</td>
<td>Normal distribution with a specific mean and variance relationship for fitting a GAMLSS model</td>
</tr>
<tr>
<td>MN3</td>
<td>Multinomial distribution in GAMLSS</td>
</tr>
<tr>
<td>NBF</td>
<td>Negative Binomial Family distribution for fitting a GAMLSS</td>
</tr>
<tr>
<td>NBI</td>
<td>Negative Binomial type I distribution for</td>
</tr>
</tbody>
</table>
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 distributions for fitting in 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
gamlss.dist-package

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

count_1_31 A set of functions to plot gamlss.family distributions

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

momentSK Sample and theoretical Moment and Centile Skewness and Kurtosis Functions

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


Beta Binomial Distribution For Fitting a GAMLSS Model

Description

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

Usage

\begin{verbatim}
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)
\end{verbatim}
Arguments

mu.link  Defines the mu.link, with "logit" link as the default for the mu parameter. Other links are "probit" and "cloglog" (complementary log-log)

sigma.link  Defines the sigma.link, with "log" link as the default for the sigma parameter. Other links are "inverse", "identity" and "sqrt"

mu  vector of positive probabilities

sigma  the dispersion parameter

bd  vector of binomial denominators

p  vector of probabilities

x, 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]

fast  a logical variable if fast=TRUE the dB function is used in the calculation of the inverse c.d.f function. This is faster to the default fast=FALSE, where the pBB() is used, but not always consistent with the results obtained from pBB(), for example if p <- pBB(c(0,1,2,3,4,5), mu=.5, sigma=1, bd=5) do not ensure that qBB(p, mu=.5, sigma=1, bd=5) will be c(0,1,2,3,4,5)

Details

Definition file for beta binomial distribution.

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

for \(y = 0, 1, 2, \ldots, n, 0 < \mu < 1 \) 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, Bob Rigby and Kalliope Akantziliotou

References


(see also https://www.gamlss.com/).

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(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 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))
print(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)
dBCCGo(x, mu = 1, sigma = 0.1, nu = 1, log = FALSE)
pBCCGo(q, mu = 1, sigma = 0.1, nu = 1, lower.tail = TRUE, log.p = FALSE)
qBCCGo(p, mu = 1, sigma = 0.1, nu = 1, lower.tail = TRUE, log.p = FALSE)
rBCCGo(n, mu = 1, sigma = 0.1, nu = 1)

Arguments

mu.link Defines the mu.link, with "identity" link as the default for the mu parameter, other links are "inverse", "log" and "own"
sigma.link Defines the sigma.link, with "log" link as the default for the sigma parameter, other links are "inverse", "identity" and "own"
uu.link Defines the nu.link, with "identity" link as the default for the nu parameter, other links are "inverse", "log" and "own"
x, q vector of quantiles
mu vector of location parameter values
sigma vector of scale parameter values
nu vector of skewness 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 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 \( \text{gamlss.family} \) object which can be used to fit a Cole and Green distribution in the \( \text{gamlss}() \) function. \( dBCCG() \) gives the density, \( pBCCG() \) gives the distribution function, \( qBCCG() \) gives the quantile function, and \( rBCCG() \) generates random deviates.

Warning

The \( BCCGuntr \) 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 \([\mu > 0, \sigma > 0, \nu = (-\infty, +\infty)]\)

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 \([\mu > 0, \sigma > 0, \text{and} \nu = (-\infty, +\infty)]\)

Author(s)

Mikis Stasinopoulos, Bob Rigby and Kalliope Akantziliotou

References


Rigby, R.A. Stasinopoulos, D.M. (2006). Using the Box-Cox \( t \) distribution in \( \text{GAMLSS} \) to model skewness and and kurtosis. \( \text{Statistical Modelling,} \ 6(3) :209. \ doi: 10.1191/1471082X06st122oa \)


See Also
gamlss.family, BCPE, BCT

Examples

```r
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, 20,
     main = "The BCCG density mu=5, sigma=.5, nu=-1")
plot(function(x) pBCCG(x, mu = 5, sigma = .5, nu = -1), 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 dBPE, pBCPE, qBCPE and rBCPE define the density, distribution function, quantile function and random generation for the Box-Cox Power Exponential distribution. The function checkBCPE can be used, typically when a BCPE 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 BCPE distribution.] See Rigby and Stasinopoulos (2003) for details. The function BCPEo is identical to BCPE but with log link for mu.

**Usage**

```r
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")
```
tau.link = "log")
BCPEuntr(mu.link = "identity", sigma.link = "log", nu.link = "identity",
tau.link = "log")
dBCPE(x, mu = 5, sigma = 0.1, nu = 1, tau = 2, log = FALSE)
pBCPE(q, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
qBCPE(p, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
rBCPE(n, mu = 5, sigma = 0.1, nu = 1, tau = 2)
dBCPEo(x, mu = 5, sigma = 0.1, nu = 1, tau = 2, log = FALSE)
pBCPEo(q, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
qBCPEo(p, mu = 5, sigma = 0.1, nu = 1, tau = 2, lower.tail = TRUE, log.p = FALSE)
rBCPEo(n, mu = 5, sigma = 0.1, nu = 1, tau = 2)
checkBCPE(obj = NULL, mu = 10, sigma = 0.1, nu = 0.5, tau = 2,...)

Arguments
mu.link    Defines the mu.link, with "identity" link as the default for the mu parameter. Other links are "inverse", "log" and "own"
sigma.link Defines the sigma.link, with "log" link as the default for the sigma parameter. Other links are "inverse", "identity" and "own"
u.link     Defines the nu.link, with "identity" link as the default for the nu parameter. Other links are "inverse", "log" and "own"
tau.link   Defines the tau.link, with "log" link as the default for the tau parameter. Other links are "logshifted", "identity" and "own"
x,q        vector of quantiles
mu          vector of location 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
obj         a gamlss BCPE family object
...         for extra arguments

Details
The probability density function of the untruncated Box Cox Power Exponential distribution, (BCPE.untr), is defined as
\[
f(y|\mu, \sigma, \nu, \tau) = \frac{y^{\nu-1}\tau \exp[-\frac{1}{2}|\tau\tilde{z}|^\tau]}{\mu^{\nu}\sigma c^{(1+1/\tau)}\Gamma(\frac{1}{\tau})}
\]
where \( c = \left[2^{(-2/\tau)}\Gamma(1/\tau)/\Gamma(3/\tau)\right]^{0.5} \), 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 \( \nu \)), \( \nu \) controls the skewness and \( \tau \) the kurtosis of the distribution

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

**References**


**See Also**

gamlss.family, BCT
### Examples

```r
# BCPE() #
# library(gamlss)
# data(abdom)
# h<-gamlss(y~cs(x,df=3), sigmaformula=~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")
```

---

**BCT**  
*Box-Cox t distribution for fitting a GAMLSS*

---

**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**

```r
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**

- `mu.link`  
  Defines the `mu.link`, with "identity" link as the default for the mu parameter. Other links are "inverse", "log" and "own"

- `sigma.link`  
  Defines the `sigma.link`, with "log" link as the default for the sigma parameter. Other links are "inverse", "identity", "own"
nu.link  Defines the nu.link, with "identity" link as the default for the nu parameter. Other links are "inverse", "log", "own".

tau.link  Defines the tau.link, with "log" link as the default for the tau parameter. Other links are "inverse", "identity" and "own".

x, q  vector of quantiles
mu  vector of location 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

Details

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

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

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

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 dB, 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.
**BE**

**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**

mu.link the mu link function with default logit

sigma.link the sigma link function with default logit

x,q vector of quantiles

mu 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 \( y = (0,1), \alpha > 0 \) 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 \( \sigma = \left(\frac{1}{\alpha+\beta+1}\right)^{1/2} \) for \( \mu = (0,1) \) and \( \sigma = (0,1) \). The expected value of y is \( \mu \) and the variance is \( \sigma^2 \mu \ast (1 - \mu) \).

**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
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, 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
mu.link     the mu link function with default logit
sigma.link  the sigma link function with default logit
nu.link     the nu link function with default log
tau.link    the tau link function with default log
x,q         vector of quantiles
mu          vector of location parameter values
sigma       vector of scale parameter values
nu           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

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")
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")
```
\textbf{Arguments}

\begin{itemize}
  \item \texttt{mu.link} \quad the mu link function with default \texttt{logit}
  \item \texttt{sigma.link} \quad the sigma link function with default \texttt{log}
  \item \texttt{nu.link} \quad the nu link function with default \texttt{logit}
  \item \texttt{x,q} \quad vector of quantiles
  \item \texttt{mu} \quad vector of location parameter values
  \item \texttt{sigma} \quad vector of precision parameter values
  \item \texttt{nu} \quad vector of parameter values modelling the probability at one
  \item \texttt{log, log.p} \quad logical; if TRUE, probabilities p are given as \texttt{log}\texttt{(p)}.
  \item \texttt{lower.tail} \quad logical; if TRUE (default), probabilities are \texttt{P}[X \leq x], otherwise, \texttt{P}[X > x]
  \item \texttt{p} \quad vector of probabilities.
  \item \texttt{n} \quad number of observations. If \texttt{length(n) > 1}, the length is taken to be the number required
  \item \texttt{from} \quad where to start plotting the distribution from
  \item \texttt{to} \quad up to where to plot the distribution
  \item \texttt{obj} \quad a fitted BEOI object
  \item \texttt{...} \quad other graphical parameters for plotting
\end{itemize}

\textbf{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 \(0 < \mu < 0, \sigma > 0\) and \(0 < \nu < 1\).

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


See Also

`gamlss.family`, `BEOI`

Examples

```r
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)
```
BEZI

The zero-inflated beta distribution for fitting a GAMLSS

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

mu.link the mu link function with default logit
sigma.link the sigma link function with default log
nu.link: the nu link function with default logit

x, q: vector of quantiles

mu: vector of location parameter values

sigma: vector of precision parameter values

nu: vector of parameter values modelling the probability at zero

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

default: 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 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 \( 0 < \mu < 0, \sigma > 0 \) 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>
BEZI

References


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 = 0.5, sigma = 5, nu = 0.1), 0, 0.999)
# plotting the inverse cdf
plot(function(y) qBEZI(y, mu = 0.5, sigma = 5, nu = 0.1), 0, 0.999)
# generate random numbers
dat <- rBEZI(100, mu = 0.5, sigma = 5, nu = 0.1)
# fit a model to the data. Fits 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
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

mu.link Defines the mu.link, with "logit" link as the default for the mu parameter. Other links are "probit" and "cloglog" (complementary log-log)
x vector of (non-negative integer) quantiles
mu vector of positive probabilities
bd vector of binomial denominators
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]

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 \( y = 0, 1, 2, ..., n \) 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


See Also

`gamlss.family, ZABI, ZIBI`

Examples

```r
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')
```
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

mu.link The link function for mu
sigma.link The link function for sigma
nu.link The link function for nu
tau.link The link function for tau
x vector of (non-negative integer)
um vector of positive means
**sigma**
vector of positive dispersion parameter

**nu**
vector of a positive parameter

**tau**
vector of probabilities

**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

**q**
vector of quantiles

**n**
number of random values to return

**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 BNB is

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

for \( y = 0, 1, 2, 3, ..., \mu > 0, \sigma > 0 \) 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 <d.stasinopoulos@londonmrt.ac.uk>

**References**


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
rN <- 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

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

Defines the right link for each parameter

Author(s)

Calliope Akantziliotou

References


See Also

gamlss.family

Description

Those functions are used in the distribution book of gamlss, see Rigby et. al 2019.

Usage

```r
binom_1_31(family = BI, mu = c(0.1, 0.5, 0.7), bd = NULL, miny = 0, maxy = 20, cex.axis = 1.2, cex.all = 1.5)
binom_2_33(family = BB, mu = c(0.1, 0.5, 0.8), sigma = c(0.5, 1, 2), bd = NULL, miny = 0, maxy = 10, cex.axis = 1.5, cex.all = 1.5)
binom_3_33(family = ZIBB, mu = c(0.1, 0.5, 0.8), sigma = c(0.5, 1, 2), nu = c(0.01, 0.3), bd = NULL, miny = 0, maxy = 10, cex.axis = 1.5, cex.all = 1.5, cols = c("darkgray", "black"), spacing = 0.3, legend.cex=1, legend.x="topright",
```
legend.where=c("left","right", "center")
contR_2_12(family = "NO", mu = c(0, -1, 1), sigma = c(1, 0.5, 2),
cols=c(gray(.1),gray(.2),gray(.3)),
ltype = c(1, 2, 3), maxy = 7,
no.points = 201, y.axis.lim = 1.1,
cex.axis = 1.5, cex.all = 1.5,
legend.cex=1, legend.x="topleft" )
contR_3_11(family = "PE", mu = 0, sigma = 1, nu = c(1, 2, 3),
cols=c(gray(.1),gray(.2),gray(.3)), maxy = 7, no.points = 201,
ltype = c(1, 2, 3), y.axis.lim = 1.1, cex.axis = 1.5,
cex.all = 1.5, legend.cex=1, legend.x="topleft")
contR_4_13(family = "SEP3", mu = 0, sigma = 1, nu = c(0.5, 1, 2),
tau = c(1, 2, 5), cols=c(gray(.1),gray(.2),gray(.3)), maxy = 7,
no.points = 201, ltype = c(1, 2, 3),
y.axis.lim = 1.1, cex.axis = 1.5, cex.all = 1.5,
legend.cex=1, legend.x="topleft", legend.where=c("left","right"))
contRplus_2_11(family = GA, mu = 1, sigma = c(0.1, 0.6, 1),
cols=c(gray(.1),gray(.2),gray(.3)),
maxy = 4, no.points = 201,
y.axis.lim = 1.1, ltype = c(1, 2, 3),
cex.axis = 1.5, cex.all = 1.5,
legend.cex=1, legend.x="topright")
contRplus_3_13(family = "BCCG", mu = 1, sigma = c(0.15, 0.2, 0.5),
nu = c(-2, 0, 4),
cols=c(gray(.1),gray(.2),gray(.3)),
maxy = 4, ltype = c(1, 2, 3),
no.points = 201, y.axis.lim = 1.1,
cex.axis = 1.5, cex.all = 1.5,
legend.cex=1, legend.x="topright", legend.where=c("left","right"))
contRplus_4_33(family = BCT, mu = 1, sigma = c(0.15, 0.2, 0.5),
nu = c(-4, 0, 2), tau = c(100, 5, 1),
cols=c(gray(.1),gray(.2),gray(.3)),
maxy = 4, ltype = c(1, 2, 3),
no.points = 201, y.axis.lim = 1.1,
cex.axis = 1.5, cex.all = 1.5,
legend.cex=1, legend.x="topright", legend.where=c("left","right"))
contR01_2_13(family = "BE", mu = c(0.2, 0.5, 0.8), sigma = c(0.2, 0.5, 0.8),
cols=c(gray(.1),gray(.2),gray(.3)),
maxy = 4, ltype = c(1, 2, 3),
no.points = 201, y.axis.lim = 1.1,
cex.axis = 1.5, cex.all = 1.5,
legend.cex=1, legend.x="topright", legend.where=c("left","right"))
Arguments

- **family**
  - a gamlss family distribution

- **mu**
  - the mu parameter values

- **sigma**
  - The sigma parameter values
nu the nu parameter values
tau the tau parameter values
bd the binomial denominator
miny minimal value for the y axis
maxy maximal value for the y axis
cex.axis the size of the letters in the two axes
cex.all the overall size of all plotting characters
cols colours
spacing spacing between plots
ltype The type of lines used
no.points the number of points in the curve
y.axis.lim the maximum value for the y axis
maxYlim the maximum permissible value for Y
legend.cex the size of the legend
legend.x where in the figure to put the legend
legend.where where in the whole plot to put the legend

Details

The function plot different types of continuous and discrete distributions: i) contR: continuous distribution defined on minus infinity to plus infinity, ii) contRplus: continuous distribution defined from zero to plus infinity, iii) contR01: continuous distribution defined from zero to 1, iv) bimom binomial type discrete distributions, v) count count type discrete distributions.

The first number after the first underline in the name of the function indicates the number of parameters in the distribution. The two numbers after the second underline indicate how many rows and columns are in the plot.

Value

The result is a plot

Note

more notes

Author(s)

Mikis Stasinopoulos, Robert Rigby, Gillian Heller, Fernanda De Bastiani
References


See Also

gamlss.family

Examples

count_1_31()

---

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

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

Details

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

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

for \( y = 0, 1, 2, \ldots, \infty \), \( \mu > 0 \) 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

(see also https://www.gamlss.com/).
See Also

BI, BB

Examples

```r
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+.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)
```

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

```r
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

- `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 `nu.link`, with "log" link as the default for the `nu` parameter
- `x` vector of (non-negative integer) quantiles
- `p` vector of probabilities
- `q` vector of quantiles
- `mu` vector of positive `mu`
- `sigma` vector of positive dispersion parameter `sigma`
nuatherine vector of nu
log, log.p
logical; if TRUE, probabilities p are given as log(p)
lower.tail
logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$
n
number of 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 $y = 0, 1, 2, ..., \infty$, $\mu > 0$, $\sigma > 0$ and $\mu > 0$.

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 distribution 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)

References


(see also [https://www.gamlss.com/](https://www.gamlss.com/)).
DEL

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')

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

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 nu.link, with "logit" link as the default for the nu parameter
x 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
The probability function of the Delaporte distribution is given by

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

where

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

for \( y = 0, 1, 2, ..., \infty \) 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


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

(see also https://www.gamlss.com/).
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

```r
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

- `mu.link` the link function for `mu` with default log
- `sigma.link` the link function for `sigma` with default log
- `x, q` vector of (non-negative integer) quantiles
- `p` vector of probabilities
- `mu` the `mu` parameter
- `sigma` the `sigma` parameter

Examples

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

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

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

max.value a constant, set to the default value of 10000 for how far the algorithm should look for \( q \)

\( n \) how many random values to generate

Details

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

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

for \( y = 0, 1, 2, \ldots, \infty \), \( \mu > 0 \) 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 distributions calculates the constant of proportionality numerically therefore it can be slow for large data

Author(s)

Mikis Stasinopoulos, Bob Rigby and Marco Enea

References


(see also https://www.gamlss.com/).
See Also

PO

Examples

DPO()
# overdispense DPO
x <- 0:20
plot(x, dDPO(x, mu=5, sigma=3), type="h", col="red")
# underdispense 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

mu.link Defines the mu.link, with "identity" 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 nu.link, with "log" link as the default for the nu parameter.
tau.link Defines the tau.link, with "log" link as the default for the tau parameter.
Details

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

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

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

References


The ex-Gaussian distribution

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 \( \text{dexGAUS} \), \( \text{pexGAUS} \), \( \text{qexGAUS} \) and \( \text{rexGAUS} \) define the density, distribution function, quantile function and random generation for the ex-Gaussian distribution.

Usage

\[
\text{exGAUS}(\mu \text{.link} = \text{"identity"}, \sigma \text{.link} = \text{"log"}, \nu \text{.link} = \text{"log"})
\]
\[
\text{dexGAUS}(x, \mu = 5, \sigma = 1, \nu = 1, \log = \text{FALSE})
\]
\[
\text{pexGAUS}(q, \mu = 5, \sigma = 1, \nu = 1, \text{lower.tail} = \text{TRUE}, \log.p = \text{FALSE})
\]
\[
\text{qexGAUS}(p, \mu = 5, \sigma = 1, \nu = 1, \text{lower.tail} = \text{TRUE}, \log.p = \text{FALSE})
\]
\[
\text{rexGAUS}(n, \mu = 5, \sigma = 1, \nu = 1, \ldots)
\]

Arguments

- \( \mu \text{.link} \): Defines the \( \mu \text{.link} \), with "identity" link as the default for the \( \mu \) parameter.
- \( \sigma \text{.link} \): Defines the \( \sigma \text{.link} \), with "log" link as the default for the \( \sigma \) parameter.
- \( \nu \text{.link} \): Defines the \( \nu \text{.link} \), with "log" link as the default for the \( \nu \) parameter. Other links are "inverse", "identity", "logshifted" (shifted from one) and "own".
- \( x, q \): vector of quantiles
- \( \mu \): vector of \( \mu \) parameter values
exGAUS

sigma    vector of scale parameter values
nu       vector of nu 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 probability density function of the ex-Gaussian distribution, \( \text{exGAUS} \), is defined as

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

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

Value

\text{exGAUS}() returns a \text{gamlss.family} object which can be used to fit ex-Gaussian distribution in the \text{gamlss()} function. \text{dexGAUS}() gives the density, \text{pexGAUS}() gives the distribution function, \text{qexGAUS}() gives the quantile function, and \text{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


(see also \url{https://www.gamlss.com/}).
EXP

See Also

gamlss.family, BCCG, GA, IG LNO

Examples

```r
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")
```

Description

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

Usage

```r
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

- \texttt{mu.link} Defines the \texttt{mu} link, with "log" link as the default for the \texttt{mu} parameter, other links are "inverse" and "identity"
- \texttt{x,q} vector of quantiles
- \texttt{mu} vector of location parameter values
- \texttt{log, log.p} logical; if TRUE, probabilities \texttt{p} are given as \texttt{log(p)}.
- \texttt{lower.tail} logical; if TRUE (default), probabilities are \texttt{P[X \leq x]}, otherwise, \texttt{P[X > x]}
- \texttt{p} vector of probabilities
- \texttt{n} number of observations. If length(\texttt{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, Bob Rigby and Nicoleta Motpan

References


See Also

gamlss.family

Examples

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

Description

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

Usage

```r
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. `sum(object$pdf*diff(object$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


See Also
histSmo

Examples
# Normal
r1<-flexDist(quantiles=list(values=qNO(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)),
expectiles=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

\[
\text{GA}(\text{mu.link} = \text{"log"}, \text{sigma.link} = \text{"log"})
\]

\[
\text{dGA}(x, \text{mu} = 1, \text{sigma} = 1, \text{log} = \text{FALSE})
\]

\[
\text{pGA}(q, \text{mu} = 1, \text{sigma} = 1, \text{lower.tail} = \text{TRUE}, \text{log.p} = \text{FALSE})
\]

\[
\text{qGA}(p, \text{mu} = 1, \text{sigma} = 1, \text{lower.tail} = \text{TRUE}, \text{log.p} = \text{FALSE})
\]

\[
\text{rGA}(n, \text{mu} = 1, \text{sigma} = 1)
\]

Arguments

- **mu.link**
  Defines the mu.link, with "log" link as the default for the mu parameter, other links are "inverse", "identity" ans "own"

- **sigma.link**
  Defines the sigma.link, with "log" link as the default for the sigma parameter, other link is the "inverse", "identity" and "own"

- **x, q**
  vector of quantiles

- **mu**
  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 \( P[X \leq x] \), otherwise, \( P[X > x] \)

- **p**
  vector of probabilities.

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

Details

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, Bob Rigby and Calliope Akantziliotou
References


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)

GAF

The Gamma distribution family

Description

The function GAF() defines a gamma distribution family, which has three parameters. This is not the generalised gamma distribution which is called GG. The third parameter here is to model the mean and variance relationship. The distribution can be fitted using the function gamlss(). The mean of GAF 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. This is an instance of the Taylor’s power low, see Enki et al. (2017). The functions dGAF, pGAF, qGAF and rGAF define the density, distribution function, quantile function and random generation for the GAF parametrization of the gamma family.
GAF

Usage

GAF(mu.link = "log", sigma.link = "log", nu.link = "identity")
dGAF(x, mu = 1, sigma = 1, nu = 2, log = FALSE)
pGAF(q, mu = 1, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)
qGAF(p, mu = 1, sigma = 1, nu = 2, lower.tail = TRUE, log.p = FALSE)
rGAF(n, mu = 1, sigma = 1, nu = 2)

Arguments

mu.link Defines the mu.link, with "identity" 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 nu.link with "identity" link as the default for the nu parameter
x,q vector of quantiles
mu vector of location parameter values
sigma vector of scale parameter values
nu vector of power 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 parametrization of the gamma family given in the function GAF() is:

\[ f(y|\mu, \sigma_1) = \frac{y^{(1/\sigma_1^2)-1} \exp[-y/(\sigma_1^2\mu)]}{(\sigma_1^2\mu)^{(1/\sigma_1^2)}} \Gamma(1/\sigma_1^2) \]

for \( y > 0, \mu > 0 \) where \( \sigma_1 = \sigma \mu^{\nu/2}/(1 - 1) \).

Value

GAF() returns a gamlss.family object which can be used to fit the gamma family in the gamlss() function.

Note

For the function GAF(), \( \mu \) is the mean and \( \sigma \mu^{\nu/2} \) is the standard deviation of the gamma family. The GAF is design for fitting regression type models where the variance is proportional to a power of the mean.

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, Robert Rigby and Fernanda De Bastiani

References


See Also

gamlss.family, GA, GG

Examples

GAF()
## Not run:
m1<-gamlss(y~poly(x,2),data=abdom,family=GAF, method=mixed(1,100), c.crit=0.00001)
# using RS()
m2<-gamlss(y~poly(x,2),data=abdom,family=GAF, n.cyc=5000, c.crit=0.00001)
# the estimates of nu slightly different
fitted(m1, "nu")[1]
fitted(m2, "nu")[1]
# global deviance almost identical
AIC(m1, m2)
## End(Not run)

Description

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

```r
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.

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

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 <d.stasinopoulos@londonmet.ac.uk>, Bob Rigby and Calliope Akantziliotou

**References**


**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**

```r
normal <- NO(mu.link="log", sigma.link="log")
normal
```

---

**GB1**

The generalized Beta type 1 distribution for fitting a GAMLSS

**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**

```r
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**

- `mu.link` Defines the `mu.link`, with "identity" 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 `nu.link`, with "log" link as the default for the `nu` parameter.
- `tau.link` Defines the `tau.link`, with "log" link as the default for the `tau` parameter.
- `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 1, GB1, is defined as
\[ f(y | \mu, \sigma, \nu, \tau) = \frac{\tau^{\nu} \gamma^{\alpha - 1} (1 - y^{\nu})^{\beta - 1}}{B(\alpha, \beta)[\nu + (1 - \nu)y^{\tau}]^{\alpha + \beta}} \]
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

References
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

mu.link Defines the mu.link, with "identity" 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 nu.link, with "log" link as the default for the nu parameter.

tau.link Defines the tau.link, with "log" link as the default for the tau parameter.

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

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

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
References


See Also

gamlss.family, JSU, BCT

Examples

```r
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

```r
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()`, `qlogitNO()`, `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 and Bob Rigby

References


Examples

```r
# 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)
```
GEOM

Geometric distribution for fitting a GAMLSS model

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)
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 \( \text{dGEOM} \), \( \text{pGEOM} \), \( \text{qGEOM} \) and \( \text{rGEOM} \) 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)
pGEOMo(q, mu = 0.5, lower.tail = TRUE, log.p = FALSE)
qGEOMo(p, mu = 0.5, lower.tail = TRUE, log.p = FALSE)
rGEOMo(n, mu = 0.5)

Arguments

mu.link Defines the mu.link, with log link as the default for the mu parameter
x, q vector of quantiles
mu vector of location parameter values
log, log.p logical; if TRUE, probabilities p are given as log(p)
lower.tail logical; if TRUE (default), probabilities are \( P[X \leq x] \), otherwise \( P[X > x] \)
p vector of probabilities
n number of observations. If length(n) > 1, the length is taken to be the number required

Details

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


See Also

gamlss.family

Examples

```r
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

```r
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

- **mu.link**: Defines the mu.link, with "log" link as the default for the mu parameter, other links are "inverse" and "identity"
- **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 sigma parameter, other links are $1/\nu^2$ 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 $P[X \leq x]$, otherwise, $P[X > x]$
- **p**: vector of probabilities
- **n**: number of observations. If length(n) > 1, the length is taken to be the number required
Details

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

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

where \( z = (y/\mu)^{\nu} \), \( \theta = 1/(\sigma^2 |\nu|) \) 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


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**

```r
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**

- `mu.link`: Defines the mu.link, with "log" link as the default for the `mu` parameter, other links are "inverse" and "identity"
- `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 "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

\[ f(y|\mu, \sigma, \nu) = \left( \frac{c}{\mu} \right)^{\nu} \left( \frac{y^{\nu - 1}}{2K(\frac{1}{2}, \nu)} \right) \exp\left( \frac{-1}{2\sigma} \left( \frac{y}{\mu} + \frac{\mu}{y} \right) \right) \]

where

\[ c = \frac{K\left(\frac{1}{2}, \nu + 1\right)}{K\left(\frac{1}{2}, \nu\right)} \]

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, Bob Rigby and Nicoleta Motpan

References


(see also https://www.gamlss.com/).

See Also

gamlss.family, IG

Examples

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

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
x vector of (non-negative integer) quantiles
mu vector of positive mu
sigma vector of positive dispersion parameter sigma
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 Generalised Poisson distribution is given by

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

for \( y = 0, 1, 2, ..., \infty \) 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


See Also

`gamlss.family`, `PO`, `DPO`

Examples

```r
GPO() # gives information about the default links for the
gplot 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

\[ \text{GT}(\mu, \sigma, \nu, \tau) \]
\[ \text{dGT}(x, \mu = 0, \sigma = 1, \nu = 3, \tau = 1.5, \log = \text{FALSE}) \]
\[ \text{pGT}(q, \mu = 0, \sigma = 1, \nu = 3, \tau = 1.5, \text{lower.tail = TRUE, log.p = \text{FALSE}}) \]
\[ \text{qGT}(p, \mu = 0, \sigma = 1, \nu = 3, \tau = 1.5, \text{lower.tail = TRUE, log.p = \text{FALSE}}) \]
\[ \text{rGT}(n, \mu = 0, \sigma = 1, \nu = 3, \tau = 1.5) \]

Arguments

- `mu.link`: Defines the `mu.link` with "identity" 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 `nu.link` with "log" link as the default for the `nu` parameter.
- `tau.link`: Defines the `tau.link` with "log" link as the default for the `tau` parameter.
- `x`: vector of quantiles
- `q`: vector of location parameter values
- `mu`: vector of scale parameter values
- `sigma`: vector of skewness `nu` parameter values
- `nu`: vector of kurtosis `tau` parameter values
- `tau`: vector of kurtosis `tau` parameter values
- `log, log.p`: logical; if TRUE, probabilities 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 t distribution, \((GT)\), is defined as

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

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

References


(see also https://www.gamlss.com/).

See Also

`gamlss.family`, `JSU`, `BCT`

Examples

```r
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=2, nu=1, tau=4")
# library(gamlss)
# m1<-gamlss(y~1, family=GT)
```

---

**GU**

*The Gumbel distribution for fitting a GAMLSS*

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

```r
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

mu.link
Defines the mu.link, with "identity" link as the default for the mu parameter. Other available link is "inverse", "log" and "own"

sigma.link
Defines the sigma.link, with "log" link as the default for the sigma parameter, other links are the "inverse", "identity" and "own"

x,q
vector of quantiles

mu
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 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, Bob Rigby and Calliope Akantziliotou

References


See Also
gamlss.family, RG

Examples

```r
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**

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

**Arguments**

- `family` a `gamlss.family` object
- `...` for passing extra arguments

**Value**

A hazard function.

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Vlasios Voudouris
References


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)

---

**IG**

*Inverse Gaussian distribution for fitting a GAMLSS*

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

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
x,q vector of quantiles
mu 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

... ... can be used to pass the uppr.limit argument to qIG

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

(see also https://www.gamlss.com/).
IGAMMA

See Also
gamlss.family, GA, GIG

Examples
# gives information about the default links for the normal distribution
# library(gamlss)
# data(rent)
# gamlss(R~cs(Fl),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)"

IGAMMA

Inverse Gamma distribution for fitting a GAMLSS

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 den-
sity, 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
mu.link Defines the mu.link, with log 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
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 parameterization of the Inverse Gamma distribution in the function IGAMMA is

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

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


See Also

gamlss.family, GA

Examples

```r
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)
```
The Johnson’s Su distribution for fitting a GAMLSS

Description

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

Usage

\begin{verbatim}
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)
\end{verbatim}

Arguments

- \texttt{mu.link}: Defines the \texttt{mu.link}, with "identity" link as the default for the \texttt{mu} parameter. Other links are "inverse", "log" ans "own".
- \texttt{sigma.link}: Defines the \texttt{sigma.link}, with "log" link as the default for the \texttt{sigma} parameter. Other links are "inverse", "identity" ans "own".
- \texttt{nu.link}: Defines the \texttt{nu.link}, with "identity" link as the default for the \texttt{nu} parameter. Other links are "inverse", "log" and "own".
- \texttt{tau.link}: Defines the \texttt{tau.link}, with "log" link as the default for the \texttt{tau} parameter. Other links are "inverse", "identity" ans "own".
- \texttt{x,q}: vector of quantiles
- \texttt{mu}: vector of location parameter values
- \texttt{sigma}: vector of scale parameter values
- \texttt{nu}: vector of skewness \texttt{nu} parameter values
- \texttt{tau}: vector of kurtosis \texttt{tau} parameter values
- \texttt{log, log.p}: logical; if TRUE, probabilities \texttt{p} are given as \texttt{log(p)}.
- \texttt{lower.tail}: logical; if TRUE (default), probabilities are \texttt{P[X <= x]}, otherwise, \texttt{P[X > x]}
- \texttt{p}: vector of probabilities.
- \texttt{n}: number of observations. If \texttt{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

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

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

This is a reparameterization of the original Johnson Su distribution, Johnson (1954), so the parameters \( \nu \) 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 Normal density function. The distribution is appropriate for leptokurtic data that is data with kurtosis larger that the Normal distribution one.

Value

\( JSU() \) returns a \texttt{gamlss.family} object which can be used to fit a Johnson’s Su distribution in the \texttt{gamlss()} function. \texttt{dJSU()} gives the density, \texttt{pJSU()} gives the distribution function, \texttt{qJSU()} gives the quantile function, and \texttt{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


The original Johnson’s Su distribution for fitting a GAMLSS

Description

This function defines the Su distribution, 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 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

mu.link Defines the mu.link, with "identity" link as the default for the mu parameter. Other links are "inverse", "log" and "own"
sigma.link Defines the sigma.link, with "log" link as the default for the sigma parameter. Other links are "inverse", "identity" and "own"
uo.link Defines the nu.link, with "identity" link as the default for the nu parameter. Other links are "inverse", "log" and "own"
tau.link Defines the tau.link, with "log" link as the default for the tau parameter. Other links are "inverse", "identity" and "own"
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 original Jonhson’s SU distribution, (JSU), is defined as
\[
f(y|\mu, \sigma, \nu, \tau) = \frac{\tau}{\sigma} \frac{1}{(z^2 + 1)^{\frac{1}{2}}} \exp\left(-\frac{1}{2} \frac{r^2}{\sigma^2}\right)
\]

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 recomended to be used only with method=mixed(2,20)

Author(s)

Mikis Stasinopoulos and Bob Rigby

References


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")
Arguments

- **mu.link**: defines the mu.link, with logit link as the default for the mu parameter.
- **sigma.link**: defines the sigma.link, with logit link as the default for the sigma parameter which in this case is the probability at zero.
- **x**: vector of (non-negative integer)
- **mu**: vector of positive means
- **sigma**: vector of probabilities at zero
- **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**: 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, Bob Rigby

References


See Also

gamlss.family, PO, ZAP

Examples

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

LNO

Log Normal distribution for fitting in GAMLSS

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 \texttt{dLNO}, \texttt{pLNO}, \texttt{qLNO} and \texttt{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**

\begin{verbatim}
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)
\end{verbatim}

**Arguments**

- **mu.link**: Defines the \texttt{mu.link}, with "identity" or "log" link depending on the parameterization
- **sigma.link**: Defines the \texttt{sigma.link}, with "log" link as the default for the sigma parameter. Other links are "inverse", "identity" and "own"
- **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 \texttt{p} are given as \texttt{log}(\texttt{p}).
- **lower.tail**: logical; if TRUE (default), probabilities are \texttt{P}[X \leq \texttt{x}], otherwise, \texttt{P}[X > \texttt{x}]
- **p**: vector of probabilities.
- **n**: number of observations. If \texttt{length(n) > 1}, the length is taken to be the number required

**Details**

The probability density function in \texttt{LOGNO} is defined as

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


(see also https://www.gamlss.com/).

**See Also**

gamlss.family, BCCG
Examples

LOGNO() # gives information about the default links for the log normal distribution
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)

L0

Logistic distribution for fitting a GAMLSS

Description

The function L0(), 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

L0(mu.link = "identity", sigma.link = "log")
dL0(x, mu = 0, sigma = 1, log = FALSE)
pL0(q, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
qL0(p, mu = 0, sigma = 1, lower.tail = TRUE, log.p = FALSE)
rL0(n, mu = 0, sigma = 1)
Arguments

mu.link
Defines the mu.link, with "identity" link as the default for the mu parameter

sigma.link
Defines the sigma.link, with "log" link as the default for the sigma parameter

x,q
vector of quantiles

mu
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

Definition file for Logistic distribution.

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

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, Bob Rigby and Calliope Akantziliotou

References


See Also

`gamlss.family.NO, TF`

Examples

```r
LO() # gives information about the default links for the Logistic distribution
dplot(function(y) dLO(y, mu=10, sigma=2), 0, 20)
pplot(function(y) pLO(y, mu=10, sigma=2), 0, 20)
qplot(function(y) qLO(y, mu=10, sigma=2), 0, 1)
# library(gamlss)
data(abdom)
# h<-gamlss(y~cs(x,df=3), sigma.formula=~cs(x,1), family=LO, 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**

```r
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**

- `mu.link` the link function for `mu`
- `sigma.link` the link function for `sigma`
- `x, q` vector of quantiles
- `mu` 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 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 \(0 < y < 1\), \(\mu = (0, 1)\) 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


See Also

gamlss.family, LOGNO

Examples

```r
# 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(pLQNO(x, sigma=3), 0, 1)
curve(qLQNO(x, sigma=3), 0, 1)
Y<- rLQNO(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 μ. The variance is equal to μ*(1+σ*μ) so the standard deviation is sqrt(μ*(1+σ*μ)). 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

mu.link mu link function with "log" as default
sigma.link mu link function with "log" as default
x,q vector of quantiles
mu 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 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
LQNO

Details

LQNO stands for Linear Quadratic Normal Family, in which the variance is a linear quadratic function of the mean: \( \text{Var}(Y) = \mu*(1+\sigma*\mu) \). 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 \( \text{Var}(Y) = \mu*(1+\sigma*\mu) \) to be valid.

Author(s)

Christos Argyropoulos

References


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)
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/` 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 `logshift01`, and the own. For the use of the own link see the example below. `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

```r
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 gamlss distribution family

Details

The own link function is added to allow the user greater flexibility. In order to use 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

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

in version 1.0-0 of gamlss, to

```r
"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 use 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(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","Box Cox Power Exponential",substitute(tau.link),c("logshifted","log","identity"),par.link = c(1))

**Author(s)**

Mikis Stasinopoulos and Bob Rigby

**References**


**See Also**

gamlss.family
Examples

str(make.link.gamlss("logshiftto1"))
l2<-make.link.gamlss("logshiftto1")
l2$linkfun(2) # should close to zero (Note that 0.00001 is added)
l2$linkfun(-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 used 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)
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

mu.link the link function for mu
sigma.link the link function for sigma
nu.link the link function for nu
tau.link the link function for tau
x the x variable
q vector of quantiles
p vector of probabilities
lower.tail logical; if TRUE (default), probabilities are P[X <= 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


See Also
gamlss.family, BI
Examples

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

momentSK

Sample and theoretical Moment and Centile Skewness and Kurtosis Functions

Description

The functions momentSK(), centileSK(), centileSkew() and centileKurt(), calculate sample statistics related to skewness and kurtosis. The function theoCentileSK() calculates the theoretical centile statistics from a given gamlss.family distribution. The plotCentileSK() plots the theoretical centile skewness and kurtosis against p (see below).

The function checkMomentSK() can be use to check (a) whether the moment skewness and kurtosis of a fitted model are modelled adequately (the residuals of the model are used). (b) whether a given sample display skewness or kurtosis.

Usage

momentSK(x, weights=NULL)
centileSK(x, cent = c(1, 25), weights=NULL)
centileSkew(x, cent = 1, weights=NULL)
centileKurt(x, cent = 1, weights=NULL)
theoCentileSK(fam = "NO", p = 0.01, ...)
plotCentileSK(fam = "NO", plotting = c("skew", "kurt", "standKurt"),
add = FALSE, col = 1, lty = 1, lwd = 1, ylim = NULL, ...)
checkMomentSK(x, weights=NULL, add = FALSE, bootstrap = TRUE, no.bootstrap = 99,
col.bootstrap = "lightblue", pch.bootstrap = 21,
asCharacter = TRUE, col.point = "black", pch.point = 4,
lwd.point = 2, text.to.show = NULL, cex.text = 1.5,
col.text = "black", show.legend = TRUE)
checkCentileSK(x,weights=NULL, type = c("central", "tail"), add = FALSE,
bootstrap = TRUE, no.bootstrap = 99,
col.bootstrap = "lightblue", pch.bootstrap = 21,
asCharacter = TRUE, col.point = "black", pch.point = 4,
lwd.point = 2, text.to.show = NULL, cex.text = 1.5,
col.text = "black", show.legend = TRUE)
Arguments

- **x**  
  data vector or gamlss model
- **weights**  
  prior weights for the x
- **cent**  
  the centile required
- **type**  
  For centile skewness and kurtosis only whether "central" (default) or "tail"
- **fam**  
  A gamlss distribution family
- **plotting**  
  what to plot
- **add**  
  whether to add the line to the existing plot
- **col**  
  the colour of the line
- **lty**  
  the type of the line
- **lwd**  
  the width of the line
- **ylim**  
  the y limit of the graph
- **p**  
  the value determining the centile skewness or kurtosis
- **...**  
  additional arguments pass to theoCentileSK() function i.e. the values of the 
  distribution parameters
- **bootstrap**  
  whether a plot of the bootstrap skewness and kurtosis measures should be added 
  in the plot
- **no.bootstrap**  
  the number of bootstrap skewness and kurtosis measures
- **col.bootstrap**  
  the colour for bootstraps
- **pch.bootstrap**  
  the point type of bootstraps
- **asCharacter**  
  whether to plot the estimated skewness and kurtosis measure as character or as 
  point
- **col.point**  
  the colour of the skewness and kurtosis measure
- **pch.point**  
  the point type of the skewness and kurtosis measure
- **lwd.point**  
  the width of the plotted point
- **text.to.show**  
  to display text different from variable or model
- **cex.text**  
  the size of the text
- **col.text**  
  the colour of the text
- **show.legend**  
  whether to show the legend

Details

Those function calculate sample moment and centile skewness and kurtosis statistics and theoretical 
centile values for a specific distribution.

Value

Different functions produce different output: The function momentSK() produce:

- **mom.skew**  
  sample moment skewness
trans.mom.skew: sample transformed moment skewness
mom.kurt: sample moment kurtosis
excess.mom.kurt: sample excess moment kurtosis
trans.mom.kurt: sample transformed moment excess kurtosis
jarque.bera.test: the value of the Jarque-bera test for testing whether skewness and excess kurtosis are zero or not

The function centileSK() produces:

S0.25: sample centile central skewness
S0.01: sample centile tail skewness
K0.01: sample centile kurtosis
standK0.01: standardised centile kurtosis, (K0.01/3.449)
exc.K0.01: excess centile kurtosis, (K0.01-3.449)
trans.K0.01: transformed excess centile kurtosis, (exc.K0.01/(1+abs(exc.K0.01))

The function centileSkew() for a given argument p produces:

p: the value determining the centile skewness
Sp: sample centile skewness at p

The function centileKurt() for a given argument p produces:

p the value determining the centile kurtosis
Kp sample centile kurtosis at p
sKp sample standardised centile kurtosis at p
ex.Kp: sample excess centile kurtosis at p
tekp: sample transformed excess centile kurtosis at p

The function theoCentileSK for a given gamlss.family produces:

IR the interquartile range of the distribution
SIR the semi interquartile range of the distribution
S_0.25 the central skewness of the distribution
S_0.01: the tail skewness of the distribution
K_0.01: the centile kurtosis of the distribution
sK_0.01: the standardised centile kurtosis of the distribution

Author(s)
Mikis Stasinopoulos, Robert Rigby, Gillain Heller and Fernanda De Bastiani.
References


See Also
gamlss.family

Examples

```r
Y <- rSEP3(1000)
momentSK(Y)
centileSK(Y)
centileSkew(Y, cent=20)
centileKurt(Y, cent=30)

theoCentileSK("BCCG", mu=2, sigma=.2, nu=2)
plotCentileSK(fam="BCCG", mu=2, sigma=.2, nu=2)

checkMomentSK(Y)
checkCentileSK(Y)
checkCentileSK(Y, type="tail")
```

---

**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

\texttt{NBF(mu.link = "log", sigma.link = "log", nu.link = "log")}

\texttt{dNBF(x, mu = 1, sigma = 1, nu = 2, log = FALSE)}

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

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

\texttt{rNBF(n, mu = 1, sigma = 1, nu = 2)}

\texttt{ZINBF(mu.link = "log", sigma.link = "log", nu.link = "log",}
\texttt{ tau.link = "logit")}

\texttt{dZINBF(x, mu = 1, sigma = 1, nu = 2, tau = 0.1, log = FALSE)}

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

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

\texttt{rZINBF(n, mu = 1, sigma = 1, nu = 2, tau = 0.1)}

Arguments

- \texttt{mu.link} The link function for \texttt{mu}
- \texttt{sigma.link} The link function for \texttt{sigma}
- \texttt{nu.link} The link function for \texttt{nu}
- \texttt{tau.link} The link function for \texttt{tau}
- \texttt{x} vector of (non-negative integer)
- \texttt{mu} vector of positive means
- \texttt{sigma} vector of positive dispersion parameter
- \texttt{nu} vector of power parameter
- \texttt{tau} vector of inflation parameter
- \texttt{log, log.p} logical; if TRUE, probabilities \texttt{p} are given as \texttt{log(p)}
- \texttt{lower.tail} logical; if TRUE (default), probabilities are \texttt{P(X <= x)}, otherwise, \texttt{P(X > x)}
- \texttt{p} vector of probabilities
- \texttt{q} vector of quantiles
- \texttt{n} number of random values to return

Details

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


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]
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

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
x vector of (non-negative integer) quantiles
mu vector of positive means
sigma vector of positive dispersion 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]

Details

Definition file for Negative Binomial type I distribution.

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

for \( y = 0, 1, 2, \ldots, \infty, \mu > 0 \) 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, Bob Rigby and Calliope Akantziliotou

References


See Also
`gamlss.family`, `NBII`, `PIG`, `SI`

Examples
```R
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
rN <- rNBI(1000, mu=5, sigma=0.5))
r <- barplot(rN, col='lightblue')
# library(gamlss)
data(aids)
# h<-gamlss(y~cs(x,df=7)+qrt, family=NBI, data=aids) # fits the model
```
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

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
x           vector of (non-negative integer) quantiles
mu          vector of positive means
sigma       vector of positive dispersion 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]

Details

Definition file for Negative Binomial type II distribution.

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

for \( y = 0, 1, 2, \ldots, \infty, \mu > 0 \) 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, Bob Rigby and Calliope Akantziliotou

References


See Also

`gamlss.family`, `NBI`, `PIG`, `SI`

Examples

```r
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

mu.link Defines the mu.link, with "identity" link as the default for the mu parameter. Other links are "inverse", "log" and "own"
sigma.link Defines the sigma.link, with "log" link as the default for the sigma parameter. Other links are "inverse", "identity" and "own"
uu.link Defines the nu.link, and because nu is fixed we use "identity" link
tau.link Defines the tau.link, and because tau is fixed we use "identity" link
x,q vector of quantiles
p vector of probabilities
n number of observations.
mu vector of location 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]

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 abs(nu)<=abs(tau)) and Student-t with nu*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


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)
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

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


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

mu.link Defines the mu.link, with "identity" link as the default for the mu parameter
sigma.link Defines the sigma.link, with "log" link as the default for the sigma parameter
x,q vector of quantiles
mu 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 parametrization of the normal distribution given in the function NO2() is

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

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. [The function NO2() defines the normal distribution with \( \sigma \) as the variance.]
Author(s)
Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

References


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
```

Description
The function NOF() defines a normal distribution family, which has three parameters. The distribution can be used 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. This is an instance of the Taylor’s power low, see Enki et al. (2017). 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

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

Arguments

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

Details

The parametrization of the normal distribution given in the function \texttt{NOF()} is

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

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

Value

returns a \texttt{gamlss.family} object which can be used to fit a normal distribution family in the \texttt{gamlss()} function.

Note

For the function \texttt{NOF()}, \( \mu \) is the mean and \( \sigma^{\nu/2} \) 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 also 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 \texttt{mixed()} method should be used in the fitting.
Author(s)

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

References


See Also

`gamlss.family`, NO, NO2

Examples

```
NOF() # gives information about the default links for the normal distribution family
## Not run:
## the normal distribution, fitting a constant sigma
m1 <- gamlss(y~poly(x,2), sigma.fo=-1, family=NO, data=abdom)
## the normal family, fitting a variance proportional to the mean (mu)
m2 <- gamlss(y~poly(x,2), sigma.fo=-1, family=NOF, data=abdom, method=mixed(1,20))
## the normal distribution fitting the 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)
## End(Not run)
```
Description

The functions PARETO() defines the one parameter Pareto distribution for \( y > 1 \).
The functions PARETO1() defines the one parameter Pareto distribution for \( y > 0 \).
The functions PARETO01() defines the one parameter Pareto distribution for \( y > \mu \) therefore requires \( \mu \) to be fixed.
The functions PARETO2() and PARETO2o() define the Pareto Type 2 distribution, for \( y > 0 \), 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 corresponds 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

```
PARETO(mu.link = "log")
dPARETO(x, mu = 1, log = FALSE)
pPARETO(q, mu = 1, lower.tail = TRUE, log.p = FALSE)
qPARETO(p, mu = 1, lower.tail = TRUE, log.p = FALSE)
rPARETO(n, mu = 1)

PARETO1(mu.link = "log")
dPARETO1(x, mu = 1, log = FALSE)
pPARETO1(q, mu = 1, lower.tail = TRUE, log.p = FALSE)
qPARETO1(p, mu = 1, lower.tail = TRUE, log.p = FALSE)
rPARETO1(n, mu = 1)

PARETO1o(mu.link = "log", sigma.link = "log")
dPARETO1o(x, mu = 1, sigma = 0.5, log = FALSE)
pPARETO1o(q, mu = 1, sigma = 0.5, lower.tail = TRUE, log.p = FALSE)
qPARETO1o(p, mu = 1, sigma = 0.5, lower.tail = TRUE, log.p = FALSE)
rPARETO1o(n, mu = 1, sigma = 0.5)

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 sa 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
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 parameterization of the one parameter Pareto distribution in the function PARETO is:

\[ f(y | \mu) = \mu y^{\mu+1} \]

for \( y > 1 \) and \( \mu > 0 \).

The parameterization of the Pareto Type 2 distribution in the function PARETO2 is:

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

for \( y >= 0 \), \( \mu > 0 \) 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
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.

References


See Also

`gamlss.family`

Examples

```r
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"))
```
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

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

Details

Power Exponential distribution (PE) is defined as

\[
f(y | \mu, \sigma, \nu) = \nu \exp\left[-\left\{ \frac{1}{2} \left( \left| \frac{y - \mu}{\sigma c} \right| \right)^{\nu} \right\} \right] / \sigma 2^{(1+1/\nu)} \Gamma(\frac{3}{\nu})
\]

where \( c = \sqrt[2]{\left( \frac{1}{\nu} \Gamma(1/\nu) / \Gamma(3/\nu) \right)} \), 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

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

Value

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

μ is the mean and σ is the standard deviation of the Power Exponential distribution

**Author(s)**

Mikis Stasinopoulos, Bob Rigby

**References**


**See Also**

`gamlss.family`, `BCPE`

**Examples**

```r
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 PIG2() function is a reparametrization of PIG() where mu and sigma are orthogonal see Heller et al. (2018).

The functions dPIG, pPIG, qPIG and rPIG define the density, distribution function, quantile function and random generation for the Poisson-inverse Gaussian PIG(). distribution. Also codedPIG2, pPIG2, qPIG2 and rPIG2 are the equivalent functions for codePIG2()

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 Poisson Inverse Gaussian 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)

PIG2(mu.link = "log", sigma.link = "log")
dPIG2(x, mu=0.5, sigma=0.02, log = FALSE)
pPIG2(q, mu=0.5, sigma=0.02, lower.tail = TRUE, log.p = FALSE)
qPIG2(p, mu=0.5, sigma=0.02, lower.tail = TRUE, log.p = FALSE,
   max.value = 10000)
rPIG2(n, mu=0.5, sigma=0.02)

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 nu.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

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

where \( \alpha^2 = \frac{1}{\sigma^2} + \frac{2\mu}{\sigma} \) for \( y = 0, 1, 2, \ldots, \infty \) where \( \mu > 0 \) and \( \sigma > 0 \) and \( K_\lambda(t) = \frac{1}{2} \int_0^\infty x^{\lambda-1} \exp\left\{-\frac{1}{2}t(x+x^{-1})\right\} dx \) 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


See Also

gamlss.family, NBI, NBII, SI, SICHEL

Examples

```r
PIG()  # gives information about the default links for the Poisson-inverse Gaussian distribution
plot(function(y) dPIG(y, mu=10, sigma = 1 ), from=0, to=50, n=50+1, type="h")  # pdf
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()
```

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 dP0, pP0, qP0 and rP0 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

mu.link Defines the mu.link, with "log" link as the default for the mu parameter
x vector of (non-negative integer) quantiles
mu vector of positive means
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]

Details

Definition file for Poisson distribution.
\[ f(y|\mu) = \frac{e^{-\mu} \mu^y}{\Gamma(y + 1)} \]

for \( y = 0, 1, 2, ... \) 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, and Kalliope Akantziliotou

References


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

mu.link
Defines the mu.link, with "identity" link as the default for the mu parameter.
other available link is "inverse", "log" and "own"

sigma.link
Defines the sigma.link, with "log" link as the default for the sigma parameter,
other links are the "inverse", "identity" and "own"

x,q
vector of quantiles

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

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

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, Bob Rigby and Calliope Akantziliotou

References


(see also https://www.gamlss.com/).

See Also

gamlss.family

Examples

```r
plot(function(x) dRGE(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

```r
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

- **mu.link**: Defines the `mu.link`, with "identity" 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 `nu.link`, with "log" link as the default for the `nu` parameter
- **x,q**: vector of quantiles
- **mu**: 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 (ξ, θ, γ) → (µ, σ, ν)].

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

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

\[ f(y|µ, σ, ν) = \frac{1}{σ} \left[ 1 + \frac{ν(y - µ)}{σ} \right]^{\frac{1}{ν} - 1} S_1(y|µ, σ, ν) \]

for

\[ µ - \frac{σ}{ν} < y < ∞ \]

where

\[ S_1(y|µ, σ, ν) = \exp \left\{ - \left[ 1 + \frac{ν(y - µ)}{σ} \right]^{\frac{1}{ν}} \right\} \]

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

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

\[ f(y|α_1, α_2, α_3) = \frac{α_3}{α_2} \left[ \frac{y - α_1}{α_2} \right]^{α_3 - 1} \exp \left[ - \left( \frac{y - α_1}{α_2} \right)^{α_3} \right] \]

given by setting α₁ = µ - σ/ν, α₂ = σ/ν, α₃ = 1/ν.

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 and Kalliope Akantziliotou

References


See Also

gamlss.family

Examples

RGE() # default links for the reverse generalized extreme family distribution
code = 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^2" 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^2" and "log".
- **tau.link** Defines the tau.link, with "log" link as the default for the tau parameter. Other links are "1/tau^2", 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

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

for \(-\infty < y < \infty, \mu = (-\infty, +\infty), \sigma > 0, \nu = (-\infty, +\infty)\) and \(\tau > 0\). where \(z = \frac{y - \mu}{\sigma}, \omega = sign(z)z^{\nu/2} \sqrt{2/\tau}\) 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

References


See Also

gamlss.family, JSU, BCT

Examples

```r
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))
```
The Skew Power exponential type 1-4 distribution for fitting a GAMLSS

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

```r
SEP1(mu.link = "identity", sigma.link = "log", nu.link = "identity", 
    tau.link = "log")
```

```r
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)
```

```r
SEP2(mu.link = "identity", sigma.link = "log", nu.link = "identity", 
    tau.link = "log")
```

```r
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)
```

```r
SEP3(mu.link = "identity", sigma.link = "log", nu.link = "log", 
    tau.link = "log")
```

```r
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)
```

```r
SEP4(mu.link = "identity", sigma.link = "log", nu.link = "log", 
    tau.link = "log")
```

```r
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)
```
log.p = FALSE)
 rSEP4(n, mu = 0, sigma = 1, nu = 2, tau = 2)

Arguments

mu.link Defines the mu.link, with "identity" link as the default for the mu parameter. Other links are "inverse" 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 "log" link as the default for the nu parameter. Other links are "identity" and "inverse"
tau.link Defines the tau.link, with "log" link as the default for the tau parameter. Other links are "inverse", 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

Details

The probability density function of the Skew Power exponential distribution type 2, (SEP2), is defined as

\[ f_Y(y|\mu, \sigma, \nu, \tau) = \frac{\nu}{\sigma(1 + \nu^2)2^{1/\tau}/\Gamma(1 + 1/\tau)} \left\{ \exp \left( -\frac{1}{2} \left| \frac{\nu(y - \mu)}{\sigma} \right|^{\tau} \right) I(y < \mu) + \exp \left( -\frac{1}{2} \left| \frac{(y - \mu)}{\sigma\nu} \right|^{\tau} \right) I(y \geq \mu) \right\} \]

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
SHASH

The Sinh-Arcsinh (SHASH) distribution for fitting a GAMLSS

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.

References


See Also

gamlss.family, SEP

Examples

```r
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)
```
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

mu.link Defines the mu.link, with "identity" 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 nu.link, with "log" link as the default for the nu parameter.
tau.link Defines the tau.link, with "log" link as the default for the tau parameter.
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 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

\[ r = \frac{1}{2} \left\{ \exp \left[ \tau \sinh^{-1}(z) \right] - \exp \left[ -\nu \sinh^{-1}(z) \right] \right\} \]

and

\[ c = \frac{1}{2} \left\{ \tau \exp \left[ \tau \sinh^{-1}(z) \right] + \nu \exp \left[ -\nu \sinh^{-1}(z) \right] \right\} \]

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,

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

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

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

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 and Fiona McElduff

**References**


**See Also**

`gamlss.family`, `JSU`, `BCT`

**Examples**

```r
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))
```
The Sichel distribution for fitting a GAMLSS model

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

```r
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

- **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 nu.link, with "identity" link as the default for the nu parameter
- **x**: vector of (non-negative integer) quantiles
- **mu**: vector of positive mu
- **sigma**: vector of positive despersion 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
- **y**: the y variable. The function tofyS() should be not used on its own.
- **what**: take values 1 or 2, for function tofyS().
Details

The probability function of the Sichel distribution is given by

\[ f(y|\mu, \sigma, \nu) = \frac{\mu^y K_{y+\nu}(\alpha)}{(\alpha \sigma)^{y+\nu} y! K_\nu(\frac{1}{2})} \]

where \( \alpha^2 = \frac{1}{\sigma^2} + \frac{2\mu}{\sigma^2} \), for \( y = 0, 1, 2, ..., \infty \) where \( \mu > 0 \), \( \sigma > 0 \) and \( -\infty < \nu < \infty \) and \( K_\lambda(t) = \frac{1}{\pi} \int_0^\infty x^{\lambda-1} \exp\{-\frac{1}{2} t(x+x^{-1})\} dx \) 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 \texttt{gamlss.family} object which can be used to fit a Sichel distribution in the \texttt{gamlss()} function.

Author(s)

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

References


(see also https://www.gamlss.com/).

See Also

\texttt{gamlss.family}, \texttt{PIG}, \texttt{NBI}, \texttt{NBII}

Examples

\texttt{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,
SICHEL

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

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 nu.link, with "identity" link as the default for the nu parameter
tau.link  Defines the tau.link, with "logit" link as the default for the tau parameter
x  vector of (non-negative integer) quantiles
mu  vector of positive mu
sigma  vector of positive dispersion parameter sigma
nu  vector of nu
tau  vector of probabilities tau
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
obj  a fitted Sichel gamlss model
y  the y variable, the tofySICHEL() should not be used on its own.

Details

The probability function of the Sichel distribution is given by

\[ f(y|\mu, \sigma, \nu) = \frac{\mu^y K_{\nu+y}(\alpha)}{y!(\alpha \sigma)^{y+\nu} K_{\nu+1}(\frac{1}{\sigma})} \]

for \( y = 0, 1, 2, \ldots, \infty, \mu > 0, \sigma > 0 \) 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.
SICHEL

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 $\mu^2 \left[ \frac{2\pi(\nu+1)}{c} + \frac{1}{c^2} - 1 \right]$.

Author(s)


References


(see also [https://www.gamlss.com/](https://www.gamlss.com/)).

See Also

gamlss.family, PIG, SI

Examples

```r
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))
```
**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**

```r
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**

- `mu.link`: the `mu` link function with default `logit`
- `sigma.link`: the `sigma` link function with default `log`
- `x,q`: vector of quantiles
- `mu`: 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 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


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

```r
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

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


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

mu.link Defines the mu.link, with "identity" links the default for the mu parameter
sigma.link Defines the sigma.link, with "log" as the default for the sigma parameter
nu.link Defines the nu.link, with "log" as the default for the sigma parameter
x, q vector of quantiles
mu vector of location parameter values
sigma vector of scale parameter values
nu 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 \( P[X \leq x] \), otherwise \( P[X > x] \)

\( p \) vector of probabilities

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

Details
The parameterization of the Skew Normal Type 2 distribution in the function \( \text{SN2} \) is ...

Value
returns a \text{gamlss.family} object which can be used to fit a Skew Normal Type 2 distribution in the \text{gamlss()} function.

Note
This is a special case of the Skew Exponential Power type 3 distribution (\text{SEP3}) where \( \tau=2 \).

Author(s)
Mikis Stasinopoulos, Bob Rigby and Fiona McEllduff.

References


See Also
\text{gamlss.family}

Examples
```r
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)

# The skew t distributions, type 1 to 5

**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**

```r
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^2" 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^2" 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

Details

\[ f(y|\mu, \sigma, \nu, z) = \frac{z}{\sigma} f_{z_1}(z) F_{z_2}(w)\tau = \]

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

The functions $ST1()$, $ST2()$, $ST3()$, $ST4()$ and $ST5()$ return a `gamlss.family` object which can be used to fit the skew t type 1-5 distribution in the `gamlss()` function.

The functions $dST1()$, $dST2()$, $dST3()$, $dST4()$ and $dST5()$ give the density functions.

The funcions $pST1()$, $pST2()$, $pST3()$, $pST4()$ and $pST5()$ give the cumulative distribution functions.

The functions $qST1()$, $qST2()$, $qST3()$, $qST4()$ and $qST5()$ give the quantile function, and $rST1()$, $rST2()$, $rST3()$, $rST4()$ and $rST3()$ 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


See Also

gamlss.family, SEP1, SHASH

Examples

```r
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 \( \sigma \left( \frac{\nu}{\nu-2} \right)^{0.5} \) with the degrees of freedom \( \nu \). The function `TF2` is a different parametrization where `sigma` is the standard deviation.

**Usage**

```r
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

mu.link Defines the mu.link, with "identity" 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 nu.link, with "log" link as the default for the nu parameter
x,q vector of quantiles
mu vector of location parameter values
sigma vector of scale parameter values
nu vector of the degrees of freedom 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 t family distribution.

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

\[ y = (-\infty, +\infty), \mu = (-\infty, +\infty), \sigma > 0 \text{ and } \nu > 0. \text{ Note that } z = (y - \mu)/\sigma \text{ has a standard t distribution with degrees of freedom } \nu. \]

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

\[ \mu \text{ is the mean and } \sigma[\nu/(\nu - 2)]^{0.5} \text{ is the standard deviation of the t family distribution. } \nu > 0 \text{ is a positive real valued parameter.} \]

Author(s)

Mikis Stasinopoulos, Bob Rigby and Kalliope Akantziliotou
WARING

**References**


**See Also**

[gmlls.family](#)

**Examples**

```r
tf() # gives information about the default links for the t-family distribution
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)
```

**Description**

The function `WARING()` defines the Waring distribution, a two parameter distribution, for a `gmlls.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**

```r
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

- **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.
- **x**: Vector of (non-negative integer) quantiles.
- **q**: Vector of quantiles.
- **p**: 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,

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

for \( y = 0, 1, 2, \ldots, \mu > 0 \) 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.mceluff@ich.ucl.ac.uk>

References


See Also

`gamlss.family`
Examples

```r
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**

```r
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**

- **mu.link**
  - Defines the `mu.link`, with "log" link as the default for the mu parameter, other links are "inverse", "identity" and "own"

- **sigma.link**
  - Defines the `sigma.link`, with "log" link as the default for the sigma parameter, other link is the "inverse", "identity" and "own"

- **x, q**
  - vector of quantiles

- **mu**
  - vector of the mu parameter

- **sigma**
  - vector of sigma parameter

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

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

- **p**
  - vector of probabilities.

- **n**
  - number of observations. If length(n) > 1, 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 \( \mu^2 \left[ \Gamma(\frac{2}{\sigma} + 1) - (\Gamma(\frac{1}{\sigma} + 1))^2 \right] \)

Author(s)

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

References


See Also
gamlss.family, WEI2, WEI3
WEI2

A specific parameterization of the Weibull distribution for fitting a GAMLSS

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

mu.link Defines the mu.link, with "log" link as the default for the mu parameter, other links are "inverse" and "identity"
sigma.link Defines the sigma.link, with "log" link as the default for the sigma parameter, other link is the "inverse" and "identity"
x,q vector of quantiles
mu vector of the mu parameter values
sigma vector of sigma 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

Examples

WEI()
dat<-rWEI(100, mu=10, sigma=2)
# library(gamlss)
# gamlss(dat~1, family=WEI)
The parameterization of the function \(\text{WEI2}\) 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 \(d\text{WEI2}, p\text{WEI2}, q\text{WEI2}\), and \(r\text{WEI2}\) 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 \(\text{WEI}\) for a different parameterization of the Weibull.]

**Value**

\(\text{WEI2}()\) returns a \text{gamlss.family} object which can be used to fit a Weibull distribution in the \text{gamlss()} function. \(d\text{WEI2}()\) gives the density, \(p\text{WEI2}()\) gives the distribution function, \(q\text{WEI2}()\) gives the quantile function, and \(r\text{WEI2}()\) generates random deviates. The latest functions are based on the equivalent \(R\) functions for Weibull distribution.

**Warning**

In \(\text{WEI2}\) the estimated parameters \(mu\) and \(sigma\) can be highly correlated so it is advisable to use the \(\text{CG()}\) method for fitting [as the \(\text{RS()}\) method can be veru slow in this situation.]

**Note**

The mean in \(\text{WEI2}\) is given by \(\mu^{-1/\sigma} \Gamma\left(\frac{1}{\sigma} + 1\right)\) and the variance \(\mu^{-2/\sigma} \left(\Gamma\left(\frac{2}{\sigma} + 1\right) - \left[\Gamma\left(\frac{1}{\sigma} + 1\right)\right]^2\right)\)

**Author(s)**

Mikis Stasinopoulos, Bob Rigby and Calliope Akantziliotou

**References**


**See Also**

\texttt{gamlss.family, WEI, WEI3}.
Examples

```r
WEI2()
dat<-rWEI(100, mu=.1, sigma=2)
hist(dat)
# library(gamlss)
# gamlss(dat=1, family=WEI2, method=CG())
```

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

```r
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

- `mu.link` Defines the mu.link, with "log" link as the default for the mu parameter, other links are "inverse" and "identity"
- `sigma.link` Defines the sigma.link, with "log" link as the default for the sigma parameter, other link is the "inverse" and "identity"
- `x,q` vector of quantiles
- `mu` vector of the mu parameter values
- `sigma` vector of sigma 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 parameterization of the function \(\text{WEI3}\) is given by

\[
f(y|\mu, \sigma) = \frac{\sigma^\beta}{\beta} \left( \frac{y}{\beta} \right)^{\sigma-1} e^{-\left( \frac{y}{\beta} \right)^\sigma}
\]

where \(\beta = \frac{\mu}{\Gamma(1/\sigma+1)}\) for \(y > 0, \mu > 0\) and \(\sigma > 0\). The GAMLSS functions \(d\text{WEI3}, p\text{WEI3}, q\text{WEI3},\) and \(r\text{WEI3}\) 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 \(\text{WEI}\) for a different parameterization of the Weibull.]

Value

\(\text{WEI3}()\) returns a \texttt{gamlss.family} object which can be used to fit a Weibull distribution in the \texttt{gamlss()} function. \(d\text{WEI3}()\) gives the density, \(p\text{WEI3}()\) gives the distribution function, \(q\text{WEI3}()\) gives the quantile function, and \(r\text{WEI3}()\) generates random deviates. The latest functions are based on the equivalent R functions for Weibull distribution.

Warning

In \(\text{WEI3}\) the estimated parameters \(\mu\) and \(\sigma\) can be highly correlated so it is advisable to use the \(\text{CG()}\) method for fitting [as the \(\text{RS()}\) method can be very slow in this situation.]

Note

The mean in \(\text{WEI3}\) is given by \(\mu\) and the variance \(\mu^2 \left\{ \Gamma(2/\sigma + 1) / [\Gamma(1/\sigma + 1)]^2 - 1 \right\}\)

Author(s)

Bob Rigby and Mikis Stasinopoulos

References


See Also

\texttt{gamlss.family}, \texttt{WEI}, \texttt{WEI2}
Examples

```r
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**

```r
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**

- `mu.link`: Defines the `mu.link`, with "log" link as the default for the `mu` parameter.
- `x`: vector of (non-negative integer) quantiles.
- `q`: vector of quantiles.
- `p`: vector of probabilities.
- `n`: number of random values to return.
- `mu`: vector of positive `mu` 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 Yule distribution has density

$$P(Y = y | \mu) = (\mu^{-1} + 1)B(y + 1, \mu^{-1} + 2)$$

for $y = 0, 1, 2, \ldots$ 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


(see also https://www.gamlss.com/).

See Also

gamlss.family

Examples

```r
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

mu.link Defines the mu.link, with "logit" link as the default for the mu parameter. Other links are "probit" and "cloglog" (complementary log-log)
sigma.link Defines the sigma.link, with "log" link as the default for the sigma parameter.
u.link Defines the sigma.link, with "logit" link as the default for the mu parameter. Other links are "probit" and "cloglog" (complementary log-log)
x vector of (non-negative integer) quantiles
mu vector of positive probabilities
sigma vector of positive dispertion parameter
nu vector of positive probabilities
bd vector of binomial denominators
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]

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, Bob Rigby

References


See Also

gamlss.family, NBI, NBII

Examples

```r
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')
```
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

mu.link: Defines the mu.link, with "logit" link as the default for the mu parameter. Other links are "probit" and "cloglog" (complementary log-log)
sigma.link: Defines the sigma.link, with "logit" link as the default for the mu parameter. Other links are "probit" and "cloglog" (complementary log-log)
x: vector of (non-negative integer) quantiles
mu: vector of positive probabilities
sigma: vector of positive probabilities
bd: vector of binomial denominators
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]

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


See Also

`gamlss.family.BI`

Examples

```r
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')
```
The zero adjusted Gamma distribution for fitting a GAMLSS model

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

```r
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

- `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 nu.link, with "logit" link as the default for the sigma parameter
- `x,q` vector of quantiles
- `mu` vector of location parameter values
- `sigma` vector of scale parameter values
- `nu` vector of probability at zero 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
- `from` where to start plotting the distribution from
- `to` up to where to plot the distribution
obj a fitted `gamlss` object
main for title in the plot
... ... can be used to pass the `upr.limit` argument to `qIG`

Details

The Zero adjusted GA distribution is given as

\[ f(y|\mu, \sigma, \nu) = \begin{cases} 
\nu & \text{if } y=0 \\
(1-\nu) \left[ \frac{1}{(\sigma^2 \mu)^{1/\sigma^2}} \frac{y^{\frac{1}{\sigma^2}-1} e^{-y/(\sigma^2 \mu)}}{\Gamma(1/\sigma^2)} \right] & \text{otherwise}
\end{cases} \]

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


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)

The zero adjusted Inverse Gaussian distribution for fitting a GAMLSS model

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)
rZAIG(n, mu = 1, sigma = 1, nu = 0.1, ...)
plotZAIG(mu = 5, sigma = 1, nu = 0.1, from = 0, to = 10, n = 101,
main = NULL,...)
meanZAIG(obj)

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 nu.link, with "logit" link as the default for the sigma parameter
x,q vector of quantiles
mu vector of location parameter values
| sigma | vector of scale parameter values |
| nu | vector of probability at zero 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 |
| from | where to start plotting the distribution from |
| to | up to where to plot the distribution |
| obj | a fitted BEINF object |
| main | for title in the plot |

... can be used to pass the uppr.limit argument to qIG

**Details**

The Zero adjusted IG distribution is given as

\[
f(y|\mu, \sigma, \nu) = \nu
\]

if \(y=0\)

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

otherwise

for \(y = (0, \infty), \mu > 0, \sigma > 0\) and \(0 < \nu < 1\). \(E(y) = (1 - \nu)\mu\) and \(Var(y) = (1 - \nu)\mu^2(\nu + \mu\sigma^2)\).

**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**


**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]
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

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]

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, Bob Rigby
References


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

mu.link defines the mu.link, with "log" link as the default for the mu parameter
sigma.link defines the sigma.link, with "logit" link as the default for the sigma parameter which in this case is the probability at zero. Other links are "probit" and "cloglog" (complementary log-log)
x vector of (non-negative integer)
mu vector of positive means
sigma vector of probabilities at zero
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]

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, Bob Rigby

References


See Also

gamlss.family, PO, ZIP, ZIP2, ZALG
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

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

sigma.link defines the sigma.link, with "logit" link as the default for the sigma parameter which in this case is the probability at zero. Other links are "probit" and "cloglog"(complementary log-log)

x vector of (non-negative integer) quantiles

mu vector of positive means

sigma vector of probabilities at zero

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]
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 \( y = 0, 1, ..., \).

Value

returns a \texttt{gamlss.family} object which can be used to fit a zero inflated poisson distribution in the \texttt{gamlss()} function.

Author(s)

Mikis Stasinopoulos, Bob Rigby

References


See Also

\texttt{gamlss.family}, \texttt{PO}, \texttt{ZIP2}

Examples

\texttt{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

mu.link defines the mu.link, with "log" link as the default for the mu parameter
sigma.link defines the sigma.link, with "logit" link as the default for the sigma parameter which in this case is the probability at zero. Other links are "probit" and "cloglog" (complementary log-log)
x vector of (non-negative integer) quantiles
mu vector of positive means
sigma vector of probabilities at zero
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]
Details

Let $Y = 0$ with probability $\sigma$ and $Y \sim Po(\mu/\left(1 - \sigma\right))$ with probability $(1 - \sigma)$ then $Y$ has a Zero inflated Poisson type 2 distribution given by

$$f(y|\mu, \sigma) = \sigma + (1 - \sigma)e^{-\mu/(1-\sigma)} \quad \text{if } y = 0$$

$$f(y|\mu, \sigma) = (1 - \sigma)e^{-\mu/(1-\sigma)} \frac{\mu/(1-\sigma)^y}{y!} \quad \text{if } y = 1, 2, 3, \ldots$$

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


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 \( y = 1, 2, 3, \ldots, \infty \), the zero adjusted zipf permits values on \( y = 0, 1, 2, \ldots, \infty \). 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

```r
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 = 100000)

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 = 100000)
rZAZIPF(n, mu = 0.5, sigma = 0.1, max.value = 100000)
```

Arguments

- mu.link: the link function for the parameter mu with default log
- x, q: vectors of (non-negative integer) quantiles
- p: vector of probabilities
- mu: vector of positive parameter
log, log.p logical; if TRUE, probabilities p are given as \( \log(p) \)
lower.tail logical; if TRUE (default), probabilities are \( P[X \leq x] \), otherwise, \( P[X > x] \)
n number of random values to return
max.value a constant, set to the default value of 10000. It is used in the q function which numerically calculates how far the algorithm should look for q. Maybe for zipf data the values has to increase at a considerable computational cost.
sigma.link the link function for the parameter sigma with default logit
sigma 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 \( y = 1, 2, \ldots, \infty, \mu > 0 \) 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


See Also

PO, LG, GEOM, YULE

Examples

```r
# 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

* distribution
  BB, 8
  BCCG, 11
  BCPE, 13
  BCT, 16
  BE, 18
  BEINF, 20
  BEOI, 24
  BEZI, 27
  BI, 30
  BNB, 31
  count_1_31, 35
  DBI, 39
  DBURR12, 41
  DEL, 43
  DPO, 45
  EGB2, 47
  exGAUS, 49
  EXP, 51
  flexDist, 53
  GA, 54
  GAF, 56
  GB1, 62
  GB2, 64
  gen.Family, 66
  GEOM, 69
  GG, 71
  GIG, 73
  GPO, 75
  GT, 76
  GU, 78
  hazardFun, 80
  IG, 81
  IGAMMA, 83
  JSU, 85
  JSUo, 87
  LG, 89
  LNO, 91
  LO, 94
  LOGITNO, 96
  LQNO, 98
  MN3, 104
  momentSK, 106
  NBF, 109
  NBI, 112
  NBII, 114
  NET, 116
  NO, 118
  NO2, 119
  NOF, 121
  PARETO2, 124
  PE, 126
  PIG, 129
  PO, 131
  RG, 133
  RGE, 135
  SEP, 137
  SEP1, 140
  SHASH, 142
  SI, 146
  SICHEL, 148
  SIMPLEX, 151
  SN1, 152
  SN2, 154
  ST1, 156
  TF, 159
  WARING, 161
  WEI, 163
  WEI2, 165
  WEI3, 167
  YULE, 169
  ZABB, 170
  ZABI, 172
  ZAGA, 175
  ZAIG, 177
  ZANBI, 179
  ZAP, 181
  ZIP, 183
INDEX

ZIP2, 185
ZIPF, 187

* package
  gamlss.dist-package, 4

* regression
  BB, 8
  BCCG, 11
  BCPE, 13
  BCT, 16
  BE, 18
  BEINF, 20
  BEOI, 24
  BEZI, 27
  BI, 30
  BNB, 31
  checklink, 34
  count_1_31, 35
  DBI, 39
  DBURR12, 41
  DEL, 43
  DPO, 45
  EGB2, 47
  exGAUS, 49
  EXP, 51
  flexDist, 53
  GA, 54
  GAF, 56
  gamlss.family, 58
  GB1, 62
  GB2, 64
  gen.Family, 66
  GEOM, 69
  GG, 71
  GIG, 73
  GPO, 75
  GT, 76
  GU, 78
  hazardFun, 80
  Ig, 81
  IGAMMA, 83
  JSU, 85
  JSUo, 87
  LG, 89
  LNO, 91
  LQ, 94
  LOGITNO, 96
  LQNO, 98
  make.link.gamlss, 100

MN3, 104
NBF, 109
NBI, 112
NBII, 114
NET, 116
NO, 118
NO2, 119
NOF, 121
PARETO2, 124
PE, 126
PIG, 129
PO, 131
RG, 133
RGE, 135
SEP, 137
SEP1, 140
SHASH, 142
SI, 146
SICHEL, 148
SIMPLEX, 151
SN1, 152
SN2, 154
ST1, 156
TF, 159
WARING, 161
WEI, 163
WEI2, 165
WEI3, 167
YULE, 169
ZABB, 170
ZABI, 172
ZAGA, 175
ZAIG, 177
ZANBI, 179
ZAP, 181
ZIP, 183
ZIP2, 185
ZIPF, 187

as.family(gamlss.family), 58
as.gamlss.family(gamlss.family), 58

BB, 8, 41, 59, 62
BCCG, 11, 18, 51, 59, 62, 91, 93
BCCGo (BCCG), 11
BCCGuntr (BCCG), 11
BCPE, 13, 18, 59, 62, 117, 128
BCPEo (BCPE), 13
BCPEuntr (BCPE), 13
BCT, 13, 15, 16, 49, 59, 62, 64, 66, 78, 87, 89, 139, 145
BCTo (BCT), 16
BCTuntr (BCT), 16
BE, 18, 20, 23, 59, 62
BEINF, 20, 20, 59, 62
BEINF0 (BEINF), 20
BEINF1 (BEINF), 20
BEO, 23
BEo (BE), 18
BEoI, 21, 23, 24, 26, 59
BEZI, 21, 23, 27, 29, 59
BI, 10, 30, 41, 59, 62, 105, 174
binom_1.31 (count_1.31), 35
binom_2.33 (count_1.31), 35
binom_3.33 (count_1.31), 35
BNB, 31, 59
boundary (momentSK), 106
cEGB2_1 (momentSK), 106
cEGB2_1Data (momentSK), 106
cEGB2_2 (momentSK), 106
centileKurt (momentSK), 106
centileSkew (momentSK), 106
checkBCPE (BCPE), 13
checkCentileSK (momentSK), 106
checkklink, 34
checkMomentSK (momentSK), 106
cJSU (momentSK), 106
cntR01_2_13 (count_1.31), 35
cntR01_4.33 (count_1.31), 35
cntR_2.12 (count_1.31), 35
cntR_3.11 (count_1.31), 35
cntR_4.13 (count_1.31), 35
cntRplus_2.11 (count_1.31), 35
cntRplus_3.13 (count_1.31), 35
cntRplus_4.33 (count_1.31), 35
cnt_1.22 (count_1.31), 35
cnt_1.31, 35
cnt_2.32 (count_1.31), 35
cnt_2.32R (count_1.31), 35
cnt_2.33 (count_1.31), 35
cnt_3.32 (count_1.31), 35
cnt_3.33 (count_1.31), 35
cSB (momentSK), 106
cSEP3 (momentSK), 106
cSHASH (momentSK), 106
cST3_1 (momentSK), 106
cST3_2 (momentSK), 106
dBB (BB), 8
dBCGG (BCCG), 11
dBCCGo (BCCG), 11
dBCPE (BCPE), 13
dBCPEo (BCPE), 13
dBCT (BCT), 16
dBCTo (BCT), 16
dBE (BE), 18
dBEINF (BEINF), 20
dBEINF0 (BEINF), 20
dBEINF1 (BEINF), 20
dBEO (BE), 18
dBEOI (BEOI), 24
dBEZI (BEZI), 27
dBI, 39, 59
dBI (BI), 30
dBNB (BNB), 31
dBURR12, 41, 59
dDBI (DBI), 39
dDBURR12 (DBURR12), 41
dDEL (DEL), 43
dDPO (DPO), 45
dEGB2 (EGB2), 47
dEL, 43, 59
dexGAUS (exGAUS), 49
dEXP (EXP), 51
dGA (GA), 54
dGAF (GAF), 56
dGB1 (GB1), 62
dGB2 (GB2), 64
dGEOM (GEOM), 69
dGEOMo (GEOM), 69
dGG (GG), 71
dGIG (GIG), 73
dGP (GB2), 64
dGPO (GPO), 75
dGT (GT), 76
dGU (GU), 78
dIG (IG), 81
dIGAMMA (IGAMMA), 83
dJSU (JSU), 85
dJSUo (JSUo), 87
dLG (LG), 89
dLNO (LNO), 91
dLO (LO), 94
dLOGITNO (LOGITNO), 96
dLOGNO (LNO), 91
INDEX

dLOGNO2 (LNO), 91
dLQNO (LQNO), 98
dMN3 (MN3), 104
dMN4 (MN3), 104
dMN5 (MN3), 104
dNBF (NBF), 109
dNBII (NBII), 112
dNBII (NBII), 114
dNET (NET), 116
dNO (NO), 118
dNO2 (NO2), 119
dNOF (NOF), 121
dPARETO (PARETO2), 124
dPARETO (PARETO2), 124
dPARETO (PARETO2), 124
dPARETO (PARETO2), 124
dPARETO2 (PARETO2), 124
dPE (PE), 126
dPE2 (PE), 126
dPIG (PIG), 129
dPIG2 (PIG), 129
dP0, 43, 45, 59, 76
dP0 (P0), 131
dRG (RG), 133
dRGE (RGE), 135
dSEP (SEP), 137
dSEP1 (SEP1), 140
dSEP2 (SEP1), 140
dSEP3 (SEP1), 140
dSEP4 (SEP1), 140
dSHASH (SHASH), 142
dSHASHo (SHASH), 142
dSHASHo (SHASH), 142
dSI (SI), 146
dSICHEL (SICHEL), 148
dSIMPLEX (SIMPLEX), 151
dSN1 (SN1), 152
dSN2 (SN2), 154
dST (ST), 156
dST1 (ST1), 156
dST2 (ST1), 156
dST3 (ST1), 156
dST3C (ST1), 156
dST4 (ST1), 156
dST5 (ST1), 156
dTF (TF), 159
dTF2 (TF), 159
dWARING (WARING), 161
dWEI (WEI), 163
dWEI2 (WEI2), 165
dWEI3 (WEI3), 167
dYULE (YULE), 169
dZABB (ZABB), 170
dZABI (ZABI), 172
dZABN (BNB), 31
dZAGA (ZAGA), 175
dZAIG (ZAIG), 177
dZALG (LG), 89
dZANBI (ZANBI), 179
dZAP (ZAP), 181
dZAPIG (PIG), 129
dZASICHEL (SICHEL), 148
dZAZIPF (ZIPF), 187
dZABB (ZABB), 170
dZABI (ZABI), 172
dZIBN (BNB), 31
dZINBI (BNB), 109
dZINBI (ZANBI), 179
dZIP (ZIP), 183
dZIP2 (ZIP2), 185
dZIPF (ZIPF), 187
dZIPIG (PIG), 129
dZISICHEL (SICHEL), 148

EGB2, 47, 59
exGAUS, 49, 59
EXP, 51, 59

Family (gen.Family), 66
fEGB2_1 (momentSK), 106
fEGB2_2 (momentSK), 106
fittedMN (MN3), 104
fJSU (momentSK), 106
flexDist, 53
fSEP3 (momentSK), 106
fST3_1 (momentSK), 106
fST3_2 (momentSK), 106

GA, 51, 54, 58, 59, 62, 72, 83, 84, 176
GAF, 56
gamlss.dist (gamlss.dist-package), 4
gamlss.dist-package, 4
gamlss.family, 8, 10, 13, 15, 18, 20, 23, 26, 29, 31, 35, 39, 43, 45, 49, 51, 52, 56, 58, 58, 64, 66, 70, 72, 74, 76, 78, 80, 81, 83, 84, 87, 89, 91, 93, 96, 97, 101, 105, 109, 113, 115, 117, 119,
<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>121, 123, 126, 128, 131, 133, 135, 137, 139, 142, 145, 147, 150, 154, 155, 159, 161, 162, 164, 166–168, 170, 172, 174, 176, 179, 181, 182, 184, 186</td>
<td>MN5 (MN3), 104</td>
</tr>
<tr>
<td>GB1, 20, 59, 62</td>
<td>momentSK, 106</td>
</tr>
<tr>
<td>GB2, 59, 64</td>
<td>MULTIN (MN3), 104</td>
</tr>
<tr>
<td>gen.Family, 66</td>
<td>NBF, 60, 109</td>
</tr>
<tr>
<td>gen.hazard (hazardFun), 80</td>
<td>NBI, 34, 60, 62, 111, 112, 115, 131, 133, 147, 172, 181</td>
</tr>
<tr>
<td>GEOM, 59, 68, 189</td>
<td>NBII, 34, 60, 62, 111, 113, 114, 131, 133, 147, 172, 181</td>
</tr>
<tr>
<td>GEOMo, 59</td>
<td>NET, 60, 116</td>
</tr>
<tr>
<td>GEOMo (GEOM), 69</td>
<td>NO, 60, 62, 96, 99, 118, 121, 123</td>
</tr>
<tr>
<td>get_C (DPO), 45</td>
<td>NO2, 99, 119, 119, 123</td>
</tr>
<tr>
<td>GetB1_C (DBI), 39</td>
<td>NOF, 60, 99, 121</td>
</tr>
<tr>
<td>GG, 58, 59, 71</td>
<td>own.linkfun (make.link.gamlss), 100</td>
</tr>
<tr>
<td>GIG, 59, 73, 83</td>
<td>own.linkinv (make.link.gamlss), 100</td>
</tr>
<tr>
<td>GP (GB2), 64</td>
<td>own.mu.eta (make.link.gamlss), 100</td>
</tr>
<tr>
<td>GPO, 75</td>
<td>own.valideta (make.link.gamlss), 100</td>
</tr>
<tr>
<td>GT, 59, 76</td>
<td>PARETO (PARETO2), 124</td>
</tr>
<tr>
<td>GU, 59, 62, 78</td>
<td>PARETO1 (PARETO2), 124</td>
</tr>
<tr>
<td>hazardFun, 80</td>
<td>PARETO1o (PARETO2), 124</td>
</tr>
<tr>
<td>histSmo, 54</td>
<td>PARETO2, 60, 124</td>
</tr>
<tr>
<td>IG, 51, 60, 62, 74, 81, 179</td>
<td>PARETO2o, 60</td>
</tr>
<tr>
<td>IGAMMA, 59, 83</td>
<td>PARETO2o (PARETO2), 124</td>
</tr>
<tr>
<td>JSU, 49, 60, 62, 64, 66, 78, 85, 89, 139, 145</td>
<td>pBB (BB), 8</td>
</tr>
<tr>
<td>JSUo, 87, 87</td>
<td>pBCG (BCCG), 11</td>
</tr>
<tr>
<td>LG, 60, 89, 189</td>
<td>pBCCGo (BCCG), 11</td>
</tr>
<tr>
<td>LNO, 51, 60, 62, 91</td>
<td>pBCPE (BCPE), 13</td>
</tr>
<tr>
<td>LO, 60, 62, 94</td>
<td>pBCPEo (BCPE), 13</td>
</tr>
<tr>
<td>LOGITNO, 20, 60, 96</td>
<td>pbCt (BCT), 16</td>
</tr>
<tr>
<td>LOGNO, 60, 97</td>
<td>pBCTo (BCT), 16</td>
</tr>
<tr>
<td>LOGNO (LNO), 91</td>
<td>pBE (BE), 18</td>
</tr>
<tr>
<td>LOGNO2 (LNO), 91</td>
<td>pBEINF (BEINF), 20</td>
</tr>
<tr>
<td>LQNO, 60, 98</td>
<td>pBEINF0 (BEINF), 20</td>
</tr>
<tr>
<td>make.link.gamlss, 100</td>
<td>pBEINF1 (BEINF), 20</td>
</tr>
<tr>
<td>meanBEINF (BEINF), 20</td>
<td>pBEINF (BEINF), 20</td>
</tr>
<tr>
<td>meanBEINF0 (BEINF), 20</td>
<td>pBEI (BE), 18</td>
</tr>
<tr>
<td>meanBEINF1 (BEINF), 20</td>
<td>pBEINF1 (BEINF), 20</td>
</tr>
<tr>
<td>meanBEI (BEI), 24</td>
<td>pBEo (BE), 18</td>
</tr>
<tr>
<td>meanBEZI (BEZI), 27</td>
<td>pBEoi (BEoi), 24</td>
</tr>
<tr>
<td>meanZAGA (ZAGA), 175</td>
<td>pBEZI (BEZI), 27</td>
</tr>
<tr>
<td>meanZAIG (ZAIG), 177</td>
<td>pBI (BI), 30</td>
</tr>
<tr>
<td>MN3, 104</td>
<td>pBNI (BNB), 31</td>
</tr>
<tr>
<td>MN4 (MN3), 104</td>
<td>pDBI (DBI), 39</td>
</tr>
<tr>
<td>pDBURR12 (DBURR12), 41</td>
<td>pDEL (DEL), 43</td>
</tr>
<tr>
<td>pDOM (DOM), 45</td>
<td>PE, 60, 62, 126</td>
</tr>
<tr>
<td>PE2, 60</td>
<td>PE2 (PE), 126</td>
</tr>
<tr>
<td>pEG2 (EG2), 47</td>
<td></td>
</tr>
</tbody>
</table>
INDEX

pexGAUS (exGAUS), 49
pEXP (EXP), 51
pG (GA), 54
pGAF (GAF), 56
pGB1 (GB1), 62
pGB2 (GB2), 64
pGEOM (GEOM), 69
pGEOIm2 (GEOM), 69
pGG (GG), 71
pGIG (GIG), 73
pGP (GB2), 64
pGPM (GP0), 75
pGT (GT), 76
pGU (GU), 78
Pig, 60, 62, 113, 115, 129, 147, 150
pIG (IG), 81
Pig2 (Pig), 129
pIGamma (IGAMMA), 83
pJSU (JSU), 85
pJSU0 (JSU0), 87
pLG (LG), 89
pLNO (LNO), 91
pL2 (L2), 94
pLOGITNO (LOGITNO), 96
pLOGNO (LOGNO), 91
pLOGNO2 (LOGNO), 91
plotBEINF (BEINF), 20
plotBEINF0 (BEINF0), 20
plotBEINF1 (BEINF1), 20
plotBEINF0 (BEINF0), 24
plotBEI (BEI), 27
plotCentileSK (momentSK), 106
plotZAIG (ZAIG), 175
pLQNO (LQNO), 98
pMN3 (MN3), 104
pMN4 (MN4), 104
pMN5 (MN5), 104
pNBF (NBF), 109
pNB1 (NB1), 112
pNBII (NBII), 114
pNET (NET), 116
pNO (NO), 118
pNO2 (NO2), 119
pNOF (NOF), 121
PO, 47, 60, 62, 76, 91, 131, 182, 184, 189
pPARETO (PARETO), 124
pPARETOo (PARETO2), 124
pPARETOo (PARETO2), 124
pPARETO (PARETO), 124
pPARETOo (PARETO2), 124
pPE (PE), 126
pPE2 (PE), 126
pPIG (PIG), 129
pPIG (PIG), 129
pP0 (P0), 131
pRIG (RG), 133
pRGE (RGE), 135
print.gamlss.family (gamlss.family), 58
pSEP (SEP), 137
pSEP1 (SEP1), 140
pSEP2 (SEP2), 140
pSEP3 (SEP3), 140
pSEP4 (SEP4), 140
pSHASH (SHASH), 142
pSHASHo (SHASH), 142
pSHASHo2 (SHASH), 142
pSI (SI), 146
pSICHEL (SICHEL), 148
pSIMPLEX (SIMPLEX), 151
pSN1 (SN1), 152
pSN2 (SN2), 154
pSST (ST1), 156
pST1 (ST1), 156
pST2 (ST1), 156
pST3 (ST1), 156
pST3C (ST1), 156
pST4 (ST1), 156
pST5 (ST1), 156
pTF (TF), 159
pTF2 (TF), 159
pWARING (WARING), 161
pWEI (WEI), 163
pWEI2 (WEI2), 165
pWEI3 (WEI3), 167
pYULE (YULE), 169
pZABB (ZABB), 170
pZABI (ZABI), 172
pZABN (ZABN), 31
pZAGA (ZAGA), 175
pZAJIG (ZAIG), 177
pZALG (LG), 89
pZANBI (ZANBI), 179
pZAP (ZAP), 181
pZAP (ZAP), 129
pZASICHEL (SICHEL), 148
INDEX

pZABPF (ZIPF), 187
pZABB (ZABB), 170
pZABI (ZABI), 172
pZANBI (ZANBI), 179
pZIP (ZIP), 183
pZIP2 (ZIP2), 185
pZIPIG (ZIPF), 187
pZISICHEL (SICHEL), 148
qBB (BB), 8
qBCCG (BCCG), 11
qBCCGo (BCCG), 11
qBCPE (BCPE), 13
qBCPNeo (BCPE), 13
qBCTo (BCT), 16
qBCTo (BCT), 16
qBE (BE), 18
qBEINF (BEINF), 20
qBEINF0 (BEINF), 20
qBEINF1 (BEINF), 20
qBEO (BE), 18
qBEOI (BEOI), 24
qBEZI (BEZI), 27
qBI (BI), 30
qBNB (BNB), 31
qDBI (DBI), 39
qDBURR12 (DBURR12), 41
qDEL (DEL), 43
qDPO (DPO), 45
qEGB2 (EGB2), 47
qexGAUS (exGAUS), 49
qEXP (EXP), 51
qGA (GA), 54
qGAF (GAF), 56
qGB1 (GB1), 62
qGB2 (GB2), 64
qGEOM (GEOM), 69
qGEOMo (GEOM), 69
qGG (GG), 71
qGI (GI), 73
qGP (GB2), 64
qGPO (GPO), 75
qGT (GT), 76
qGU (GU), 78
qIG (IG), 81
qIGAMMA (IGamma), 83
qJSU (JSU), 85
qJSUo (JSUo), 87
qLG (LG), 89
qLNO (LNO), 91
qL0 (L0), 94
qLOGITNO (LOGITNO), 96
qLOGNO (LOGNO), 91
qLOGNO2 (LOGNO), 91
qLQNO (LQNO), 98
qMN3 (MN3), 104
qMN4 (MN3), 104
qMN5 (MN3), 104
qNBF (NBF), 109
qNBI (NBI), 112
qNBII (NBII), 114
qNET (NET), 116
qNO (NO), 118
qNO2 (NO2), 119
qNOF (NOF), 121
qPARETO (PARETO), 124
qPARETO1 (PARETO), 124
qPARETO1o (PARETO), 124
qPARETO2 (PARETO), 124
qPARETO2o (PARETO), 124
qPE (PE), 126
qPE2 (PE), 126
qPI (PI), 129
qPIG (PIG), 129
qPIG2 (PIG), 129
qP0 (P0), 131
qRG (RG), 133
qRGE (RGE), 135
qSEP (SEP), 137
qSEP1 (SEP1), 140
qSEP2 (SEP1), 140
qSEP3 (SEP1), 140
qSEP4 (SEP1), 140
qSHASH (SHASH), 142
qSHASHo (SHASH), 142
qSHASHo2 (SHASH), 142
qSI (SI), 146
qSICHEL (SICHEL), 148
qSIMPLEX (SIMPLEX), 151
qSN1 (SN1), 152
qSN2 (SN2), 154
qSST (ST1), 156
qST1 (ST1), 156
qST2 (ST1), 156
qST3 (ST1), 156
INDEX

qST3C (ST1), 156
qST4 (ST1), 156
qST5 (ST1), 156
qTF (TF), 159
qTF2 (TF), 159
qWARING (WARING), 161
qWEI (WEI), 163
qWEI2 (WEI2), 165
qWEI3 (WEI3), 167
qYULE (YULE), 169
qZABB (ZABB), 170
qZABI (ZABI), 172
qZABNB (BNB), 31
qZAGA (ZAGA), 175
qZAIG (ZAIG), 177
qZALG (LG), 89
qZNBI (ZANBI), 179
qZAP (ZAP), 181
qZAPIG (PIG), 129
qZASICHEL (SICHEL), 148
qZAZIPF (ZIPF), 187
qZIBB (ZABB), 170
qZIBI (ZABI), 172
qZIBNB (BNB), 31
qZINBF (NB), 109
qZINBI (ZANBI), 179
qZIP (ZIP), 183
qZIP2 (ZIP2), 185
qZIPF (ZIPF), 187
qZIPIG (PIG), 129
qZISICHEL (SICHEL), 148
rBB (BB), 8
rBCCG (BCCG), 11
rBCCGo (BCCG), 11
rBCPE (BCPE), 13
rBCPEx (BCPE), 13
rBCT (BCT), 16
rBCTo (BCT), 16
rBE (BE), 18
rBEINF (BEINF), 20
rBEINF0 (BEINF), 20
rBEINF1 (BEINF), 20
rBEn (BE), 18
rBEIOI (BEIOI), 24
rBEZI (BEZI), 27
rBI (BI), 30
rBNB (BNB), 31
rDBI (DBI), 39
rDBURR12 (DBURR12), 41
rDEL (DEL), 43
rDPO (DPO), 45
rEGB2 (EGB2), 47
rexGAUS (exGAUS), 49
rEXP (EXP), 51
RGE, 60, 62, 80, 133
rGA (GA), 54
rGAF (GAF), 56
rGB1 (GB1), 62
rGB2 (GB2), 64
RGE, 60, 135
rGEOM (GEOM), 69
rGEOMo (GEOM), 69
rGG (GG), 71
rGIG (GIG), 73
rGP (GB2), 64
rGPO (GPO), 75
rGT (GT), 76
rGU (GU), 78
rIG (IG), 81
rIGAMMA (IGAMMA), 83
rJSU (JSU), 85
rJSUo (JSUo), 87
rLG (LG), 89
rLNO (LNO), 91
rLO (LO), 94
rLOGITNO (LOGITNO), 96
rLOGNO (LNO), 91
rLOGNO2 (LNO), 91
rLNO2 (LNO), 98
rMN3 (MN3), 104
rMN4 (MN3), 104
rMN5 (MN3), 104
rNBF (NB), 109
rNBI (NBI), 112
rNBII (NBII), 114
rNET (NET), 116
rNO (NO), 118
rNO2 (NO2), 119
rNOF (NOF), 121
rPARETO (PARETO2), 124
rPARETO1 (PARETO2), 124
rPARETO1o (PARETO2), 124
rPARETO2 (PARETO2), 124
rPARETO2o (PARETO2), 124
rPE (PE), 126
rPE2 (PE), 126
INDEX

rPIG (PIG), 129
rPIG2 (PIG), 129
rPO (PO), 131
rRG (RG), 133
rRGE (RGE), 135
rSEP (SEP), 137
rSEP1 (SEP1), 140
rSEP2 (SEP1), 140
rSEP3 (SEP1), 140
rSEP4 (SEP1), 140
rSHASH (SHASH), 142
rSHASHo (SHASH), 142
rSHASHo2 (SHASH), 142
rSI (SI), 146
rSICHEL (SICHEL), 148
rSIMPLEX (SIMPLEX), 151
rSN1 (SN1), 152
rSN2 (SN2), 154
rST1 (ST1), 156
rST1 (ST1), 156
rST2 (ST1), 156
rST3 (ST1), 156
rST3C (ST1), 156
rST4 (ST1), 156
rST5 (ST1), 156
rTF (TF), 159
rTF2 (TF), 159
rWARING (WARING), 161
rWEI1 (WEI1), 163
rWEI2 (WEI2), 165
rWEI3 (WEI3), 167
rYULE (YULE), 169
rZABB (ZABB), 170
rZABI (ZABI), 172
rZABNB (ZABNB), 172
rZAGA (ZAGA), 175
rZAIG (ZAIG), 177
rZALG (LG), 89
rZANBI (ZANBI), 179
rZAP (ZAP), 181
rZAPIG (PIG), 129
rZASPICH (SICHEL), 148
rZAZIPF (ZIPF), 187
rZABB (ZABB), 170
rZABI (ZABI), 172
rZABNB (ZABNB), 172
rZINBF (ZINBF), 109
rZINBI (ZINBI), 179
rZIP (ZIP), 183
rZIP2 (ZIP2), 185
rZIPF (ZIPF), 187
rZIPIG (PIG), 129
rZISICHEL (SICHEL), 148
SEP, 137, 142
SEP1, 137, 142
SEP2, 60, 140, 159
SEP3, 60
SEP4 (SEP1), 140
SHASH, 60, 142, 159
SHASHo, 60
SHASHo2 (SHASH), 142
SHASHo (SHASH), 142
SIMPLEX, 60, 151
SKcentile_col (momentSK), 106
SKcentile_gray (momentSK), 106
SKcentileBoth (momentSK), 106
SKmoment_col (momentSK), 106
SKmoment_gray (momentSK), 106
SKmomentBoth (momentSK), 106
SN1, 152
SN2, 154
SST (ST1), 156
ST1, 60, 156
ST2, 60
ST3 (ST1), 156
ST3, 60
ST3C (ST1), 156
ST4 (ST1), 156
ST5, 60
ST5 (ST1), 156
tEGB2_1 (momentSK), 106
tEGB2_2 (momentSK), 106
TF, 60, 62, 96, 159
TF2 (TF), 159
theoCentileSK (momentSK), 106
tJSU (momentSK), 106
tofyS (SI), 146
INDEX

tofySICHEL (SICHEL), 148

tSB (momentSK), 106
tSEP3 (momentSK), 106
tSHASH (momentSK), 106
tST3_1 (momentSK), 106
tST3_2 (momentSK), 106

VSICHEL (SICHEL), 148

WARING, 60, 161
WEI, 60, 62, 163, 166–168
WEI2, 60, 62, 164, 165, 167, 168
WEI3, 60, 164, 166, 167

YULE, 60, 169, 189

ZABB, 170
ZABI, 31, 60, 172
ZABNB, 60
ZABNB (BNB), 31
ZAGA, 175
ZAIG, 60, 176, 177
ZALG, 60, 182
ZALG (LG), 89
ZANBI, 60, 179
ZAP, 61, 91, 181
ZAPIG (PIG), 129
ZASICHEL, 61
ZASICHEL (SICHEL), 148
ZAZIPF, 61
ZAZIPF (ZIPF), 187
zetaP (ZIPF), 187
ZIBB (ZABB), 170
ZIBI, 31, 61
ZIBI (ZABI), 172
ZIBNB, 61
ZIBNB (BNB), 31
ZINBF (NBF), 109
ZINBI, 61
ZINBI (ZANBI), 179
ZIP, 61, 62, 182, 183, 186
ZIP2, 61, 182, 184, 185
ZIPF, 61, 187
ZIPIG, 61
ZIPIG (PIG), 129
ZISICHEL, 61
ZISICHEL (SICHEL), 148