Package ‘gamlss.inf’

March 12, 2019

Type Package
Title Fitting Mixed (Inflated and Adjusted) Distributions
Version 1.0-1
Date 2019-03-12
Depends R (>= 2.2.1), gamlss, gamlss.dist
Suggests MASS, survival, gamlss.cens, gamlss.tr
Imports methods
Author Marco Enea [aut, cre, cph],
Mikis Stasinopoulos [aut],
Bob Rigby [aut],
Abu Hossain [aut]
Maintainer Marco Enea <marco.enea@unipa.it>
Description This is an add-on package to 'gamlss'. The purpose of this package is to allow users to fit GAMLSS (Generalised Additive Models for Location Scale and Shape) models when the response variable is defined either in the intervals [0,1), (0,1] and [0,1] (inflated at zero and/or one distributions), or in the positive real line including zero (zero-adjusted distributions). The mass points at zero and/or one are treated as extra parameters with the possibility to include a linear predictor for both. The package also allows transformed or truncated distributions from the GAMLSS family to be used for the continuous part of the distribution. Standard methods and GAMLSS diagnostics can be used with the resulting fitted object.
License GPL-2 | GPL-3
URL http://www.gamlss.com/
NeedsCompilation no
Repository CRAN
Date/Publication 2019-03-12 18:02:56 UTC

R topics documented:

  gamlss.inf-package .........................................................  2
centiles.Inf0to1 ..............................................................  4
gamlssInf0to1 .................................................................  6
**Description**

This package allows mixed distribution fitting in GAMLSS. A mixed distribution is one containing both continuous and discrete parts, see Chapter 5 of Stasinopoulos et al. (2017). There are some mixed distribution in the GAMLSS implementation in R like the BEINF, BEINF0, BEINF1 for data defined on [0,1] or ZAGA, ZAG for data defined on a positive real line but the choice is very limited. This package enhance the availability of mixed distribution within the GAMLSS framework.

For historical reasons the authors use the terminology "Inflated" for models on [0,1], "Adjusted" for models on [0,Inf]. We will follow the same terminology here. So this package allows the fit of an inflated GAMLSS model when the response variable is defined in the intervals [0,1), (0,1] and [0,1] and the fit of zero adjusted models when the response variable is defined in the positive real line, (but where there are zeros in the data).

For models with inflated proportion response variables the package provides up to two extra parameters, a mass point at zero and a mass point at one. Adding an extra inflation point at zero (or at one), is equivalent to fit two separate GAMLSS models, a GAMLSS model with a continuous distribution defined at the interval (0,1), and a logit model for zero (or ones). When both zero and one are present, i.e. [0,1], a multinomial model is needed to fit the non-(0,1) part.

For the zero adjusted models with a response defined on the positive real line (but where zeros exist in the data), the actual fitting can be achieved by fitting two separate GAMLSS models one with a distribution on the real positive line and one binomial model for the zeros and non-zeros.

This package uses the two models fitting procedures but the resulting fitted object behaves like a typical GAMLSS object so a lot of standard GAMLSS diagnostics can be used with it. It also allows transformed or truncated `gamlss.family` distributions to be used for the continuous part of the distribution therefore giving a great flexibility of the type of mixed distribution to be used.

The functions `gamlssInf0to1()` and `gamlssZadj()` can fit the Inflated and Adjusted models, respectively, and they are described in the two vignettes available with the package.
Details

Package: gamlss.inf
Type: Package
Version: 1.0-1
Date: 2019-03-12
License: GPL-2 | GPL-3

Author(s)

Marco Enea <marco.enea@unipa.it>, Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Abu Hossain

Maintainer: Marco Enea <marco.enea@unipa.it>

References


See Also
gamlss

Examples

# An artificial example using simulated data in [0,1)

# Firstly, we use function gen.Family() to create
# the logit skew student t (logitsst) distribution which is defined in the (0,1)
# interval. Then we use function gen.Inf0to1() to create the 0-inflated logitsst
# distribution defined in [0,1).
# now we can generate the data and run the model

```r
set.seed(10)
Y <- rlogitSSInf0(300, mu=0.5, sigma=0.7, nu=0.5, tau=5, xi0=0.5, log=FALSE)
m1 <- gamlssInf0to1(y=Y, mu.formula=1,
                   sigma.formula=1,
                   nu.formula=1,
                   tau.formula=1,
                   xi0.formula=1,
                   family=logitSS,
                   trace = TRUE)
summary(m1)
```

---

**centiles.Inf0to1**  
*Plotting centile curves for a gamlssInf0to1 and gamlssZadj object*

## Description

Function `centiles.Inf0to1()` plots centile curves for distributions belonging to the GAMLSS family of distributions defined in the intervals (0,1], [0,1) and [0,1]. The function also tabulates the sample percentages below each centile curve (for comparison with the model percentages given by the argument `cent`). A restriction of the function is that it applies to models with one explanatory variable only.

## Usage

```r
centiles.Inf0to1(obj, xvar = NULL, cent = c(0.4, 2, 10, 25, 50, 75,
90, 98, 99.6), legend = TRUE, ylab = "y", xlab = "x", main = NULL,
main.gsub = "@", xleg = min(xvar), yleg = max(obj$y), xlim = range(xvar),
ylim = range(obj$y), save = FALSE, plot = TRUE, points = TRUE,
pch = 15, cex = 0.5, col = gray(0.7), col.centiles = 1:length(cent) +
2, lty.centiles = 1, lwd.centiles = 1, ...)
centiles.Zadj(obj, xvar = NULL, cent = c(0.4, 2, 10, 25, 50, 75,
90, 98, 99.6), legend = TRUE, ylab = "y", xlab = "x", main = NULL,
main.gsub = "@", xleg = min(xvar), yleg = max(obj$y), xlim = range(xvar),
ylim = range(obj$y), save = FALSE, plot = TRUE, points = TRUE,
pch = 15, cex = 0.5, col = gray(0.7), col.centiles = 1:length(cent) +
2, lty.centiles = 1, lwd.centiles = 1, ...)
```

## Arguments

- **obj**: a fitted `gamlss` object from fitting a `gamlss` distribution  
- **xvar**: the unique explanatory variable
Centiles are calculated using the fitted values in obj and xvar must correspond exactly to the predictor in obj to plot correctly.

The function is appropriate only when one continuous explanatory variable is fitted in the model.

Author(s)

Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby <r.rigby@londonmet.ac.uk> with contribution from Steve Ellison
References


See Also

gamlssInf0to1, gamlss, centiles.split, centiles.com

Examples

```r
## Not run:
gen.Family("SST", "logit")
gen.Inf0to1("logitSST","One")
set.seed(30)
x <- seq(0,1,l=2000)
dat <- data.frame(x)
dat$Y <- rlogitSSTInf1(2000,mu=-3+10*x-0.7*x^2,sigma=0.9,nu=0.5, tau=5,x1=plogis(-0.5*ifelse(x>0.7,-1,13)),log=FALSE)

m1 <- gamlssInf0to1(y=Y,mu.formula=-pb(x), sigma.formula=-pb(x), nu.formula=-pb(x), tau.formula=-pb(x), x1.formula=-pb(x),
data=dat, family=logitSST)

centiles.Inf0to1(m1,xvar= dat$x, cent=c(2,10,25,50,75,90,98),
col.centiles=c(1,7:2),
ylab="proportion", xlab="x",legend=FALSE,main="(c) Inf. logitSST")

## End(Not run)
```

---

**gamlssInf0to1**  
GAMLSS model for a proportion response variable with point(s) mass at 0 and or 1.
Description

Function `gamlssInf0to1()` allows to fit inflated gamlss models when the response variable distribution is defined in the intervals [0,1), (0,1] and [0,1]. The `gamlssInf0to1` model for inflated proportion variables is a `gamlss` model provided of up to two extra parameters for the mass point(s). In the case of inflation point at zero (one), this is equivalent to fit two separate models, a `gamlss` model for the (0,1) part, and a logit model for zero (one) vs non-zero (non-one) part. When both zero and one are present, a multinomial model is involved to fit the non-(0,1) part.

Usage

```r
gamlssInf0to1(y = NULL, mu.formula = ~1, sigma.formula = ~1, 
mu.formula = ~1, tau.formula = ~1, 
nu.formula = ~1, xi0.formula = ~1, xi1.formula = ~1, data = NULL, 
family = BE, weights = rep(1, length(Y_)), 
trace = FALSE, ...)
```

Arguments

- `y` the proportion response variable with inflation at zero and/or one
- `mu.formula` a model formula for mu
- `sigma.formula` a model formula for sigma
- `nu.formula` a model formula for nu
- `tau.formula` a model formula for tau
- `xi0.formula` a model formula for the probability at zero
- `xi1.formula` a model formula for the probability at one
- `data` a data frame containing the variables occurring in the formula.
- `family` any `gamlss` distribution family defined in (0,1)
- `weights` a vector of weights as in `gamlss`
- `trace` logical, if TRUE information on model estimation will be printed during the fitting
- `...` for extra parameters

Details

The default family is a Beta distribution (BE), but other (0,1) distributions can be used, e.g. those generated from existing continuous `gamlss` family distributions by using `gen.Family` with link "logit".

Value

returns a `gamlssInf0to1` object which has its own methods

Author(s)

Mikis Stasinopoulos, Robert Rigby, Abu Hossain and Marco Enea
References


See Also

gamlss.family, BEINF, BE, BEo, BEZI, BEOI

Examples

# 1. An artificial example using simulated data

# Firstly, we use function gen.Family() to create the logit skew
# student t (logitsst) distribution defined in the (0,1) interval,
# and function gen.Inf0to1() to create the 0-inflated logitSST
# distribution defined in [0,1).

gen.Family("SST", "logit")
gen.Inf0to1("logitSST","Zero")

#now we can generate the data and run the model
set.seed(10)
Y <- rlogitsstInf0(500,mu=0.5,sigma=0.7,nu=0.5,tau=5,xi0=0.5,log=FALSE)
dat <- data.frame(Y)
dat$x <- rnorm(500)
m1 <- gamlssInf0to1(y=Y,mu.formula=x, sigma.formula=-x, 
nu.formula=-x, tau.formula=-x, 
xi0.formula=-x, data=dat, family=logitSST)
summary(m1)

# 2. Example of equivalent gamlss models for an inflated-at-1 Beta distribution

Y <- rBEINF(500,mu=0.5,sigma=0.7,nu=0.5)
m2 <- gamlss(Y~1, sigma.formula=-1, mu.formula=-1, family=BEINF)
m3.1 <- gamlss(Y[Y<1]~1, sigma.formula=-1, family=BE)
m3.2 <- gamlss(Y[Y==1]~1, family=BI)
m4 <- gamlssInf0to1(Y, mu.formula=-1, sigma.formula=-1, xi1=-1, family=BE)
stopifnot(all.equal(deviance(m2), (deviance(m3.1)+deviance(m3.2))), 
all.equal(deviance(m2), deviance(m4)))
Function \texttt{gamlssZadj()} allows to fit zero adjusted \texttt{gamlss} models when the response variable distribution is defined on the positive real line. The \texttt{gamlssZadj} model for adjusted positive variables is a \texttt{gamlss} model provides one extra parameters for the mass point at zero. This is equivalent to fit two separate models, a \texttt{gamlss} model for the (0,\text{Inf}) part, and a logit model for zero part versus the non-zero part. The function works similarly but provides one fitted object.

\textbf{Usage}

\begin{verbatim}
gamlssZadj(y = NULL, mu.formula = ~1, sigma.formula = ~1, nu.formula = ~1, tau.formula = ~1, xi0.formula = ~1, data = NULL, family = GA, weights = rep(1, length(Y_)), trace = FALSE, ...)
\end{verbatim}

\textbf{Arguments}

\begin{itemize}
  \item \texttt{y} the response variable
  \item \texttt{mu.formula} a model formula for \texttt{mu}
  \item \texttt{sigma.formula} a model formula for \texttt{sigma}
  \item \texttt{nu.formula} a model formula for \texttt{nu}
  \item \texttt{tau.formula} a model formula for \texttt{tau}
  \item \texttt{xi0.formula} a model formula for \texttt{xi0}
  \item \texttt{data} a data frame containing the variables occurring in the formula.
  \item \texttt{family} any \texttt{gamlss} distribution family defined on the real line
  \item \texttt{weights} a vector of weights as in \texttt{gamlss}
  \item \texttt{trace} logical, if \texttt{TRUE} information on model estimation will be printed during the fitting
  \item ... for extra arguments to pass to \texttt{gamlss}
\end{itemize}

\textbf{Details}

The default family is a gamma distribution (GA), but other distributions on the positive real line can be used, e.g. those generated from existing continuous \texttt{gamlss} family distributions using say \texttt{gen.Family()} with "log" or \texttt{gen.trun()} from package \texttt{gamlss.tr}

\textbf{Value}

\begin{verbatim}
. Returns a \texttt{gamlssZadj} object which has its own methods
\end{verbatim}
Author(s)

Mikis Stasinopoulos, Robert Rigby and Marco Enea

References


See Also

gamlss.family, ZAGA, ZAIG

Examples

```r
y0 <- rZAGA(1000, mu=.3, sigma=.4, nu=.15)# p0=0.13
g0 <- gamlss(y~1, family=ZAGA)
t0 <- gamlssZadj(y=y0, mu.formula=-1, family=GA, trace=TRUE)
AIC(g0,t0, k=0)
```

gen.Inf0to1

Functions to generate inflated 0-to-1 distributions from existing continuous gamlss.family distributions defined in (0,1).

Description

There are six functions here. Only the function gen.Inf0to1() should be used. The remaining five functions will be automatically created once gen.Inf0to1() has been run.

Usage

gen.Inf0to1(family = "BE", type.of.Inflation = c( "Zero&One", "Zero", "One"), ...)
Inf0to1.d(family = "BE", type.of.Inflation = c( "Zero&One", "Zero", "One"), ...)
Inf0to1.p(family = "BE", type.of.Inflation = c( "Zero&One", "Zero", "One"), ...)
Arguments

family a continuous (0,1) distribution (extremes not included) gamlss.family distribution
type.of.Inflation the type of inflation
... for passing extra arguments

Details

Functions Inf0to1.d, Inf0to1.p, Inf0to1.q and Inf0to1.r allow to create the density function, distribution function, quantile function and random generation, respectively. Function plotInf0to1 can be used to create the plot the distributions.

Alternatively, the function gen.Inf0to1 creates the all the standard d,p,q,r functions plus the plotting function.

For example, let us take the case of the logit SST distribution with inflation at 1. First generate the "logitSST" distribution by using gen.Family("SST", "logit"), and then, by use gen.Inf0to1("logitSST", "One"). The functions dlogitSSTInf1, plogitSSTInf1, qlogitSSTInf1, rlogitSSTInf1 and plotlogitSSTInf1 will be automatically generated. Note that gen.Inf0to1 never creates a fitting function of the type "logitSSTInf1", but the existing logitSST must be specified instead as an argument family of function gamlssInf0to1().

Value

The function gen.Inf0to1 returns the d, p, q and r functions plus the plotting function.

Author(s)

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

References


Examples

```r
# 1.
gen.Inf0to1("BE","Zero&One")
rBETAINf0to1 <- Inf0to1.r("BE","Zero&One")
all.equal(rBETAINf0to1, rBEInf0to1)
plotBEBInf0to1()
plotBEBInf0to1(mu=0.3, sigma=0.35, xi0=0.5, xi1=0.3)

# 2.
gen.Family("SST", "logit")
gen.Inf0to1("logitSST","One")
set.seed(30)
args(rlogitSSTInf1)
y <- rlogitSSTInf1(1000, mu=0.2, sigma=0.5, nu=1, tau=5, xi1=0.2)
quantile(y, c(0.1, 0.25, 0.5, 0.75, 0.9))

args(qlogitSSTInf1)
qlogitSSTInf1(p=c(0.1, 0.25, 0.5, 0.75, 0.9), mu=0.2, sigma=0.5, nu=1, tau=5, xi1=0.2)
plotqlogitSSTInf1(mu=0.2, sigma=0.5, nu=1, tau=5, xi1=0.2)
```

gen.Zadj

*Functions to generate zero adjusted distributions from existing continuous gamlss.family distributions defined on positive real line.*

Description

There are six functions here. Only the function gen.Zadj() should be used. The remaining four functions will be automatically created once gen.Zadj() has been run.

Usage

```r
gen.Zadj(family = "GA", ...)
Zadj.d(family = "GA", ...)
Zadj.p(family = "GA", ...)
Zadj.q(family = "GA", ...)
Zadj.r(family = "GA", ...)
plotZadj(family = "GA", ...)
```

Arguments

- **family**  
  a continuous positive real line distribution
- ...  
  for additional arguments
Details

Functions `Zadj.d`, `Zadj.p`, `Zadj.q` and `Zadj.r` allow to create the density function, distribution function, quantile function and random generation, respectively. Function `plotZadj` can be used to create a plot for the distribution.

Alternatively, the function `gen.Zadj` creates the all the standard `d`, `p`, `q`, `r` functions plus the plotting function.

Value

The function `gen.Zadj` returns the `d`, `p`, `q` and `r` functions plus the plotting function.

Author(s)

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

References


See Also

gamlssZadj

Examples

```r
# 1.
gen.Zadj("BCT")

plotBCTZadj()
plotBCTZadj(mu=3,sigma=0.35,xi0=0.5)

# 2.
gen.Family("SST", "log")
gen.Zadj("logSST")
plotlogSSTZadj()
```
**predict.gamlssinf0to1**

*Extract Predictor Values and Standard Errors For New Data In a gamlssinf0to1 Model*

**Description**

`predict.gamlssinf0to1` is the `gamlssinf0to1` specific method which produce predictors for a new data set for a specified parameter from a `gamlssinf0to1` objects. The `predict.gamlssinf0to1` can be used to extract the linear predictors, fitted values and specific terms in the model at new data values in the same way that the `predict.lm()` and `predict.glm()` functions can be used for `lm` or `glm` objects. Note that linear predictors, fitted values and specific terms in the model at the current data values can also be extracted using the function `lpredHI()` (which is called from predict if new data is NULL).

**Usage**

```r
## S3 method for class 'gamlssinf0to1'
predict(object, parameter = c("mu", "sigma", "nu", "tau", "xi0", "xi1"),
       newdata = NULL, type = c("link", "response", "terms"),
       terms = NULL, se.fit = FALSE, data = NULL, ...)```

**Arguments**

- `object`: a `gamlssinf0to1` fitted model
- `parameter`: which distribution (or inflation) parameter is required, default parameter="mu"
- `newdata`: a data frame containing new values for the explanatory variables used in the model
- `type`: the default, gets the linear predictor for the specified distribution (or inflation) parameter. `type="response"` gets the fitted values for the parameter while `type="terms"` gets the fitted terms contribution
- `terms`: if `type="terms"`, which terms to be selected (default is all terms)
- `se.fit`: if TRUE the approximate standard errors of the appropriate type are extracted if exist
- `data`: the data frame used in the original fit if is not defined in the call
- `...`: for extra arguments

**Details**

The predict function assumes that the object given in `newdata` is a data frame containing the right x-variables used in the model. This could possible cause problems if transformed variables are used in the fitting of the original model. For example, let us assume that a transformation of age is needed in the model i.e. `nage<~age^.5`. This could be fitted as `mod<-gamlss(y~cs(age^.5), data=mydata)` or as `nage<~age^.5; mod<-gamlss(y~cs(nage), data=mydata)`. The later could more efficient
if the data are in thousands rather in hundreds. In the first case, the code `predict(mod, newdata=data.frame(age=c(34, 56)))` would produce the right results. In the second case a new data frame has to be created containing the old data plus any new transform data. This data frame has to be declared in the data option. The option newdata should contain a data.frame with the new names and the transformed values in which prediction is required, (see the last example).

**Value**

A vector or a matrix depending on the options.

**Author(s)**

Abu Hossain, Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Marco Enea

**References**


**See Also**

`gamlssInf0to1`

**Examples**

```r
gen.Family("SST", "logit")
gen.Inf0to1("logitSST","Zero")
set.seed(10)
Y <- rlogitSSTInf0(500,mu=0.5,sigma=0.7,nu=0.5,tau=5,xi0=0.5,log=FALSE)
dat <- data.frame(Y)
dat$x <- rnorm(500)
m1 <- gamlssInf0to1(y=Y,mu.formula=x, sigma.formula=-x, nu.formula=-x, tau.formula=-x, xi0.formula=-x, data=dat, family=logitSST)
predict(m1)
```
predict.gamlssZadj

Extract Predictor Values and Standard Errors For New Data in a
gamlssZadj Model

Description

predict.gamlssZadj is the specific method which produce predictors for a new data set for a
specified parameter from a gamlssZadj objects. The predict.gamlssZadj can be used to extract
the linear predictors, fitted values and specific terms in the model at new data values in the same
way that the predict.lm() and predict.glm() functions can be used for lm or glm objects. Note
that linear predictors, fitted values and specific terms in the model at the current data values can also
be extracted using the function lpred() (which is called from predict if new data is NULL).

Usage

## S3 method for class 'gamlssZadj'
predict(object, parameter = c("mu", "sigma", "nu", "tau", "xi0"),
newdata = NULL, type = c("link", "response", "terms"),
terms = NULL, se.fit = FALSE, data = NULL, ...)

Arguments

object a gamlssZadj fitted model
parameter which distribution (or inflation) parameter is required, default parameter="mu"
newdata a data frame containing new values for the explanatory variables used in the
model
type the default, gets the linear predictor for the specified distribution (or inflation)
parameter. type="response" gets the fitted values for the parameter while
type="terms" gets the fitted terms contribution
terms if type="terms", which terms to be selected (default is all terms)
se.fit if TRUE the approximate standard errors of the appropriate type are extracted if
exist
data the data frame used in the original fit if is not defined in the call
... for extra arguments

Details

The predict function assumes that the object given in newdata is a data frame containing the right x-
variables used in the model. This could possible cause problems if transformed variables are used in
the fitting of the original model. For example, let us assume that a transformation of age is needed in
the model i.e. nage<-age^.5. This could be fitted as mod<-gamlss(y~cs(age^.5), data=mydata)
or as nage<-age^.5; mod<-gamlss(y~cs(nage), data=mydata). The later could more efficient
if the data are in thousands rather in hundreds. In the first case, the code predict(mod, newdata=data.frame(age=c(34,56))
would produce the right results. In the second case a new data frame has to be created containing
the old data plus any new transform data. This data frame has to be declared in the data option.
The option `newdata` should contain a data.frame with the new names and the transformed values in which prediction is required, (see the last example).

**Value**

A vector or a matrix depending on the options.

**Author(s)**

Abu Hossain, Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Marco Enea

**References**


**See Also**

`gamlssZadj`

**Examples**

```r
set.seed(3210)
x <- (runif(800)*4)-2
data(sda)
fmu <- splinefun(sda$x, sda$mu)
curve(fmu, -2,2)
fsigma <- splinefun(sda$x, sda$sigma)
curve(fsigma, -2,2)
fnu <- function(x)
  {f <- splinefun(sda$x, sda$nu)
   f(x)/6}
curve(fnu, -2,2)
set.seed(321)
y0 <- rZAGA(800, mu=fmu(x), sigma=fsigma(x), nu=fnu(x))
da <- data.frame(y0,x)
g0p <- gamlss(y0~pb(x), sigma.fo=~pb(x), nu.fo=~pb(x), data=da, family=ZAGA)
t0p <- gamlssZadj(y=y0, mu.fo=~pb(x), sigma.fo=~pb(x),data=da,
```
Data for using for simulation

Description

Those data values are used to create simulated data

Usage

data("sda")

Format

A data frame with 120 observations on the following 5 variables.

- x the explanatory variable
- mu the fitted mu
- sigma the fitted sigma
- nu the fitted nu
- tau the fitted tau

Source

The data are fitted values of model

References


**Examples**

```r
data(sda)
fmu <- splinefun(sda$x, sda$mu)
curve(fmu, -2, 2)
fsigma <- splinefun(sda$x, sda$sigma)
curve(fsigma, -2, 2)
fnu <- splinefun(sda$x, sda$nu)
curve(fnu, -2, 2)
ftau <- splinefun(sda$x, sda$tau)
curve(ftau, -2, 2)
```

**summary.gamlssinf0to1**  
*Summarizes an inflated GAMLSS fitted model*

**Description**

These are specific methods for the generic function `summary` which summarize objects returned by `gamlssinf0to1` or `gamlssZadj`.

**Usage**

```r
## S3 method for class 'gamlssinf0to1'
summary(object, type = c("vcov", "qr"),
         robust = FALSE, save = FALSE, hessian.fun = c("R", "PB"),
         digits = max(3,getOption("digits") - 3),...)

## S3 method for class 'gamlssZadj'
summary(object, type = c("vcov", "qr"),
         robust = FALSE, save = FALSE, hessian.fun = c("R", "PB"),
         digits = max(3,getOption("digits") - 3),...)
```

**Arguments**

- `object`  
a `gamlssinf0to1` or `gamlssZadj` fitted model
- `type`  
the default value `vcov` uses the `vcov()` method for `gamlss` to get the variance-covariance matrix of the estimated beta coefficients, see details below. The alternative `qr` is the original method used in `gamlss` to estimate the standard errors but it is not reliable since it do not take into the account the inter-correlation between the distributional parameters `mu`, `sigma`, `nu` and `tau`, while the inflation parameters `xi0` and `xi1` are uncorrelated anyway.
- `robust`  
whether robust (sandwich) standard errors are required
- `save`  
whether to save the environment of the function so to have access to its values
- `hessian.fun`  
whether when calculate the Hessian should use the "R" function `optimHess()` or a function based on Pinheiro and Bates `nlme` package, "PB".
- `digits`  
the number of digits in the output
- `...`  
for extra arguments
Details

Using the default value type="vcov", the vcov() method is used to get the variance covariance matrix (and consequently the standard errors) of the beta parameters. The variance covariance matrix is calculated using the inverse of the numerical second derivatives of the observed information matrix. This is a more reliable method since it takes into account the inter-correlation between the all the parameters. The type="qr" assumes that the parameters are fixed at the estimated values. Note that both methods are not appropriate and should be used with caution if smoothing terms are used in the fitting.

Value

Print summary of a gamlssinf0to1 or a gamlssZadj object

Author(s)

Abu Hossain, Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>, Bob Rigby and Marco Enea

References


See Also

gamlssInf0to1, gamlssZadj

Examples

```r
# The gamlssZadj example
set.seed(3210)
x <- (runif(1000)+4)-2
data(sda)
fmu <- splinefun(sda$x, sda$mu)
fsigma <- splinefun(sda$x, sda$sigma)
fnu <- function(x)
  (f <- splinefun(sda$x, sda$nu))
f(x)/6
```
term.plotInf0to1

y0 <- rZAGA(1000, mu=fmu(x), sigma=fsigma(x), nu=fnu(x))
da <- data.frame(y0, x)
t0p <- gamlssZadj(y=y0, mu.fo=pb(x), sigma.fo=pb(x), data=da, trace=TRUE, xi0.fo=pb(x), family="GA")
summary(t0p)

term.plotInf0to1  Plot regression terms for a specified parameter of a fitted gamlss-Inf0to1 object

Description
This is a wrapper to function term.plot. term.plotInf0to1 produces term plots for a specified parameter from a gamlssInf0to1 object.

Usage

term.plotInf0to1(object, parameter = c("mu", "sigma", "nu", "tau", "xi0", "xi1"), ...)

Arguments

- object: a gamlssInf0to1 fitted model
- parameter: which distribution (or inflation) parameter is required, default parameter="mu"
- ... extra arguments, the same of term.plot (except 'what')

Details
see function term.plot

Value
A plot of fitted terms.

Author(s)
Marco Enea, Mikis Stasinopoulos, Bob Rigby and Abu Hossain

References


See Also
gamlssInf0to1

Examples

gen.Family("SST", "logit")
gen.Inf0to1("logitSST", "Zero")
set.seed(10)
Y <- rlogitSSInf0(500, mu=0.5, sigma=0.7, nu=0.5, tau=5, xi0=0.5, log=FALSE)
dat <- data.frame(Y)
dat$x <- rnorm(500)
nl <- gamlssInf0to1(y=Y, mu.formula=-x, sigma.formula=-x,
nu.formula=-x, tau.formula=-x,
xi0.formula=-x, data=dat, family=logitSST)

# term.plot for the mu parameter
term.plot(nl$dist, parameter="mu")
term.plotInf0to1(nl, parameter="mu", col.shaded = 2)

# term.plot for the binomial parameter
term.plot(nl$multinom, parameter="mu")
term.plotInf0to1(nl, parameter="xi0", col.shaded = 2)

term.plotZadj

Plot regression terms for a specified parameter of a fitted gamlssZadj object

Description

\ This is a wrapper to function term.plot. codeterm.plotZadj produces term plots for a specified parameter from a gamlssZadj object.

Usage
term.plotZadj(object, parameter = c("mu", "sigma", "nu", "tau", "xi0"),...
term.plotZadj

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>a <code>gamlssZadj</code> fitted model</td>
</tr>
<tr>
<td>parameter</td>
<td>which distribution (or inflation) parameter is required, default parameter = &quot;mu&quot;</td>
</tr>
<tr>
<td>...</td>
<td>extra arguments, the same of <code>term.plot</code> (except 'what')</td>
</tr>
</tbody>
</table>

Details

see function `term.plot`

Value

A plot of fitted terms.

Author(s)

Marco Enea, Mikis Stasinopoulos, Bob Rigby and Abu Hossain

References


See Also

`gamlssZadj`

Examples

```r
set.seed(3210)
x <- (runif(1000)*4)-2
data(sda)
fmu <- splinefun(sda$x, sda$mu)
curve(fmu, -2,2)
fsigma <- splinefun(sda$x, sda$sigma)
curve(fsigma, -2,2)
fnu <- function(x)
  {f <- splinefun(sda$x, sda$nu)
   f(x)/6}
```
curve(fnu, -2, 2)
set.seed(321)
y0 <- rZAGA(1000, mu=fnu(x), sigma=fsigma(x), nu=fnu(x))
da <- data.frame(y0, x)
g0p <- gamlss(y0~pb(x), sigma.fo=-pb(x), nu.fo=-pb(x), data=da, family=ZAGA)
t0p <- gamlssZadj(y=y0, mu.fo=-pb(x), sigma.fo=-pb(x), data=da,
trace=TRUE, xi0.fo=-pb(x), family="GA")

# term.plot for the mu parameter
term.plot(g0p);title("gamlss")
term.plot(t0p$dist,"mu");title("gamlssZadj")
term.plotZadj(t0p,"mu",col.shaded = 3);title("gamlssZadj")

# term.plot for the sigma parameter
term.plot(g0p, "sigma");title("gamlss")
term.plot(t0p$dist,"sigma");title("gamlssZadj")
term.plotZadj(t0p,"sigma",col.shaded = 3);title("gamlssZadj")

# term.plot for the binomial parameter
term.plot(g0p, "nu");title("gamlss")
term.plot(t0p$binom,"mu");title("gamlssZadj")
term.plotZadj(t0p,"xi0",col.shaded = 3);title("gamlssZadj")
Index

*Topic **datasets**
  sda, 18

*Topic **distribution**
  gen.Inf0to1, 10
  gen.Zadj, 12

*Topic **package**
  gamlss.inf-package, 2

*Topic **regression**
  centiles.Inf0to1, 4
  gamlssInf0to1, 6
  gamlssZadj, 9
  gen.Inf0to1, 10
  gen.Zadj, 12
  predict.gamlssInf0to1, 14
  predict.gamlssZadj, 16
  summary.gamlssInf0to1, 19
  term.plotInf0to1, 21
  term.plotZadj, 22

BE, 8
BEINF, 8
BEo, 8
BEOI, 8
BEZI, 8

centiles.com, 6
centiles.Inf0to1, 4
centiles.split, 6
centiles.Zadj(cenentiles.Inf0to1), 4

gamlss, 3, 6, 7, 9
gamlss.family, 8, 10
gamlss.inf (gamlss.inf-package), 2
gamlss.inf-package, 2
gamlssInf0to1, 6, 6, 15, 20, 22
gamlssZadj, 9, 13, 17, 20, 23
gen.Family, 7, 9
gen.Inf0to1, 10
gen.Zadj, 12
Inf0to1.d(gen.Inf0to1), 10

Inf0to1.p(gen.Inf0to1), 10
Inf0to1.q(gen.Inf0to1), 10
Inf0to1.r(gen.Inf0to1), 10
plotInf0to1(gen.Inf0to1), 10
plotZadj(gen.Zadj), 12
predict.gamlssInf0to1, 14
predict.gamlssZadj, 16

sda, 18
summary.gamlssInf0to1, 19
summary.gamlssZadj
  (summary.gamlssInf0to1), 19

term.plot, 21–23

term.plotInf0to1, 21
term.plotZadj, 22

Zadj.d (gen.Zadj), 12
Zadj.p (gen.Zadj), 12
Zadj.q (gen.Zadj), 12
Zadj.r (gen.Zadj), 12
ZAGA, 10
ZAIG, 10