Package ‘gamlss.mx’
November 18, 2020

Type Package
Title Fitting Mixture Distributions with GAMLSS
Version 6.0-0
Date 2020-12-05
Author Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>, Bob Rigby <r.rigby@londonmet.ac.uk>
Maintainer Mikis Stasinopoulos <d.stasinopoulos@londonmet.ac.uk>
LazyLoad yes
Depends R (>= 2.2.1), gamlss.dist, gamlss, nnet, stats, graphics
Suggests MASS
Description The main purpose of this package is to allow fitting of mixture distributions with generalised additive models for location scale and shape models see Chapter 7 of Stasinopoulos et al. (2017) <doi:10.1201/b21973-4>.
License GPL-2 | GPL-3
URL https://www.gamlss.com/
NeedsCompilation no
Repository CRAN
Date/Publication 2020-11-18 09:30:02 UTC

R topics documented:

```
gamlss.mix-package ............................................ 2
dMX ............................................................. 3
enzyme .......................................................... 5
gamlssMX ........................................................ 6
gamlssNP ......................................................... 8
MX.control ...................................................... 12
NP.control ....................................................... 13
plotMP .......................................................... 14
```

Index 16
gamlss.mix-package

The GAMLSS add on package for mixture distributions

Description

The main purpose of this package is to allow the user of the GAMLSS models to fit mixture distributions.

Details

Package: gamlss.mx  
Type: Package  
Version: 0.0  
Date: 2005-08-3  
License: GPL (version 2 or later)

This package has two main function the `gamlssMX()` which is loosely based on the package `flexmix` of R and the function `gamlssNP()` which is based on the `npmlreg` package of Jochen Einbeck, Ross Darnell and John Hinde (2006) which in turns is based on several GLIM4 macros originally written by Murray Aitkin and Brian Francis. It also contains the function `gqz()` which is written by Nick Sofroniou and the function `gauss.quad()` written by Gordon Smyth.

Author(s)

Mikis Stasinopoulos <<d.stasinopoulos@londonmet.ac.uk>> and Bob Rigby <<r.rigby@londonmet.ac.uk>>  
Maintainer: Mikis Stasinopoulos <mikis.stasinopoulos@gamlss.org>

References

Jochen Einbeck, Ross Darnell and John Hinde (2006) `npmlreg`: Nonparametric maximum likelihood estimation for random effect models, R package version 0.34


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

Evaluate the d (pdf) and p (cdf) functions from GAMLSS mixtures

Description

The functions dMX() and pMX() can be used to evaluate the pdf (p function) and the cdf (p function) receptively from a gamlss.family mixture. The functions getpdfMX() and getpdfNP() can be used to evaluate the fitted d function at a specified observation and therefore for plotting the fitted distribution of a fitted model at this observation.

Usage

dMX(y, mu = list(mu1 = 1, mu2 = 5), sigma = list(sigma1 = 1, sigma2 = 1),
   nu = list(nu1 = 1, nu2 = 1), tau = list(tau1 = 1, tau2 = 1),
   pi = list(pi1 = 0.2, pi2 = 0.8), family = list(fam1 = "NO", fam2 = "NO"),
   log = FALSE, ...)  
pMX(q, mu = list(mu1 = 1, mu2 = 5), sigma = list(sigma1 = 1, sigma2 = 1),
   nu = list(nu1 = 1, nu2 = 1), tau = list(tau1 = 1, tau2 = 1),
   pi = list(pi1 = 0.2, pi2 = 0.8), family = list(fam1 = "NO", fam2 = "NO"),
   log = FALSE, ...)  
getpdfMX(object = NULL, observation = 1)  
getpdfNP(object = NULL, observation = 1)

Arguments

y, q  vector of quantiles  
mu  a vector of mu's  
sigma  a vector of sigma's

Examples

data(enzyme)
mmNO <- gamlssMX(act~1, family=NO, K=2, data=enzyme)
mmNO

# also to make sure that it reaches the maximum
mmNOs <- gamlssMXfits(n=10, act~1, family=NO, K=2, data=enzyme)
fyNO<-dMX(y=seq(0,3,.01), mu=list(1.253, 0.1876), sigma=list(exp(-0.6665 ), exp(-2.573 )), 
   pi=list(0.4079609, 0.5920391 ), family=list("NO","NO") )
hist(enzyme$act,freq=FALSE,ylim=c(0,3.5),xlim=c(0,3),br=21)
lines(seq(0,3,.01),fyNO, col="red")

# equivalent model using gamlssNP
mmNP <- gamlssNP(act~1, data=enzyme, random=~1, sigma.fo=~MASS,family=NO, K=2)
nu         a vector of nu's
tau        a vector of tau's
pi         a vector of pi's
family     a vector of GAMLSS family's
log        whether the log of the function or not
object     a fitted gamlssMX object
observation the observation number in which we want to plot the fitted mixture
...        for extra arguments

Value

Returns values or pdf or cdf.

Author(s)

Mikis Stasinopoulos

References


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

Examples

fyNO<-dMX(y=seq(0,3,.01), mu=list(1.253, 0.1876), sigma=list(exp(-0.6665 ), exp(-2.573 )), pi=list(0.4079609, 0.5920391 ), family=list("NO","NO") )
plot(fyNO~seq(0,3,.01), type="l")

FyNO<-pMX(q=seq(0,3,.01), mu=list(1.253, 0.1876), sigma=list(exp(-0.6665 ), exp(-2.573 )), pi=list(0.4079609, 0.5920391 ), family=list("NO","NO") )
plot(FyNO~seq(0,3,.01), type="l")
enzyme

Data used in gamlss.mx

Description
enzyme: The data comprise independent measurement of enzyme activity in the blood of 245 individuals. The data were analysed by Bechker at al. (1993).
brains: the brain size, brain, and body weight, body, for 28 different animals.

Usage
data(enzyme)
data(brains)

Format
enzyme: data frame with 245 observations on the following variable act.
brains: data frame with 28 observations on the following variables: body, brain

act: a numeric vector showing enzyme activity in the blood of 245 individuals.
body: a numeric vector showing the body weight of 28 different animals
brain: a numeric vector showing the brain size of 28 different animals

References

Examples
data(enzyme)
hist(enzyme$act)
data(brains)
brains$lbrain<-log(brains$brain)
brains$lbody<-log(brains$body)
with(brains, plot(lbrain~lbody))
Function to fit finite mixture of `gamlss` family distributions

### Description

The function `gamlssMX` is designed for fitting a K fold non-parametric mixture of `gamlss` family distributions.

### Usage

```r
library(gamlssMX)

# For a single model
gamlssMX(formula = formula(data), pi.formula = ~1, 
         family = "NO", weights = NULL, K = 2, prob = NULL, 
         data, control = MX.control(...), 
         g.control = gamlss.control(trace = FALSE, ...), 
         zero.component = FALSE, ...)

gamlssMXfits(n = 5, formula = formula(data), pi.formula = ~1, 
             family = "NO", weights = NULL, K = 2, prob = NULL, 
             data, control = MX.control(), 
             g.control = gamlss.control(trace = FALSE), 
             zero.component = FALSE, ...)
```

### Arguments

- **formula**
  - This argument should be a formula (or a list of formulae of length K) for modelling the `mu` parameter of the model. Note that modelling the rest of the distributional parameters can be done by using the usual arguments to `gamlss`

- **pi.formula**
  - This should be a formula for modelling the prior probabilities as a function of explanatory variables. Note that no smoothing of other additive terms are allowed here only the usual linear terms. The modelling here is done using the `multinom` function from package `nnet`

- **family**
  - This should be a `gamlss.family` distribution (or a list of distributions). Note that if different distributions are used here their parameters should be comparable for ease of interpretation.

- **weights**
  - Prior weights if needed

- **K**
  - The number of finite mixtures with default K=2

- **prob**
  - Prior probabilities if required for starting values

- **data**
  - The data frame needed for the fit. Note that this is compulsory if `pi.formula` is used.

- **control**
  - This argument sets the control parameters for the EM iterations algorithm. The default setting are given in the `MX.control` function

- **g.control**
  - This argument can be used to pass to `gamlss()` control parameters, as in `gamlss.control`

- **n**
  - The number of fits required in `gamlssMXfits`

- **zero.component**
  - Whether zero component models exist, default is `FALSE`

- **...**
  - For extra arguments
Author(s)

Mikis Stasinopoulos and Bob Rigby

References


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

See Also

gamlss, gamlss.family

Examples

library(MASS)
data(geyser)
# fitting 2 finite normal mixtures
m1<-gamlssMX(waiting~1,data=geyser,family=NO, K=2)

# fitting 2 finite gamma mixtures
m2<-gamlssMX(waiting~1,data=geyser,family=GA, K=2)
# fitting a model for pi
# first create a data frame
geyser1<-matrix(0,ncol=2, nrow=298)
geyser1[,1] <-geyser$waiting[-1]
geyser1[,2] <-geyser$duration[-299]
rownames(geyser1)<- c("waiting", "duration")
geyser1 <-data.frame(geyser1)
# get the best of 5 fits
m3<-gamlssMXfits(n=5, waiting~1, pi.formula=~duration, data=geyser1,family=NO, K=2)
m3
**gamlssNP**

*A function to fit finite mixtures using the gamlss family of distributions*

**Description**

This function will fit a finite (or normal) mixture distribution where the kernel distribution can belong to any gamlss family of distributions using the EM algorithm. The function is based on functions alldist() and allvc of the npmlreg package of Jochen Einbeck, John Hinde and Ross Darnell.

**Usage**

```
gamlssNP(formula, random = ~1, family = NO(), data = NULL, K = 4,  
mixture = c("np", "gq"),  
tol = 0.5, weights, pluginz, control = NP.control(...),  
g.control = gamlss.control(trace = FALSE, ...), ...)  
```

**Arguments**

- **formula**: a formula defining the response and the fixed effects for the mu parameters
- **random**: a formula defining the random part of the model
- **family**: a gamlss family object
- **data**: the data frame which for this function is mandatory even if it the data are attached
- **K**: the number of mass points/integration points (supported values are 1:10,20)
- **mixture**: the mixing distribution, "np" for non-parametric or "gq" for Gaussian Quadrature
- **tol**: the tolerance scalar ussually between zero and one
- **weights**: prior weights
- **pluginz**: optional
- **control**: this sets the control parameters for the EM iterations algorithm. The default setting is the NP.control function
- **g.control**: the gamlss control function, gamlss.control, passed to the gamlss fit
- **...**: for extra arguments

**Details**

The function `gamlssNP()` is a modification of the R functions alldist() and allvc created by Jochen Einbeck and John Hinde. Both functions were originally created by Ross Darnell (2002). Here the two functions are merged to one `gamlssNP` and allows finite mixture from gamlss family of distributions.

The following are comments from the original Einbeck and Hinde documentation.
"The nonparametric maximum likelihood (NPML) approach was introduced in Aitkin (1996) as a tool to fit overdispersed generalized linear models. Aitkin (1999) extended this method to generalized linear models with shared random effects arising through variance component or repeated measures structure. Applications are two-stage sample designs, when firstly the primary sampling units (the upper-level units, e.g. classes) and then the secondary sampling units (lower-level units, e.g. students) are selected, or longitudinal data. Models of this type have also been referred to as multi-level models (Goldstein, 2003). This R function is restricted to 2-level models. The idea of NPML is to approximate the unknown and unspecified distribution of the random effect by a discrete mixture of \( k \) exponential family densities, leading to a simple expression of the marginal likelihood, which can then be maximized using a standard EM algorithm. When option 'gq' is set, then Gauss-Hermite masses and mass points are used and considered as fixed, otherwise they serve as starting points for the EM algorithm. The position of the starting points can be concentrated or extended by setting tol smaller or larger than one, respectively. Variance component models with random coefficients (Aitkin, Hinde & Francis, 2005, p. 491) are also possible, in this case the option random.distribution is restricted to the setting 'np'. The weights have to be understood as frequency weights, i.e. setting all weights equal to 2 will duplicate each data point and hence double the disparity and deviance. Warning: There might be some options and circumstances which had not been tested and where the weights do not work." Note that in keeping with the gamlss notation disparity is called global deviance.

**Value**

The function `gamlssNP` produces an object of class "gamlssNP". This object contain several components.

- **family**: the name of the gamlss family
- **type**: the type of distribution which in this case is "Mixture"
- **parameters**: the parameters for the kernel gamlss family distribution
- **call**: the call of the gamlssNP function
- **y**: the response variable
- **bd**: the binomial denominator, only for BI and BB models
- **control**: the NP.control settings
- **weights**: the vector of weights of te expanded fit
- **G.deviance**: the global deviance
- **N**: the number of observations in the fit
- **rqres**: a function to calculate the normalized (randomized) quantile residuals of the object (here is the gamlss object rather than gamlssNP and it should change??)
- **iter**: the number of external iterations in the last gamlss fitting (?? do we need this?)
- **type**: the type of the distribution or the response variable here set to "Mixture"
- **method**: which algorithm is used for the gamlss fit, RS(), CG() or mixed()
- **contrasts**: the type of contrasts use in the fit
- **converged**: whether the gamlss fit has converged
- **residuals**: the normalized (randomized) quantile residuals of the model
The fitted values of the extended mu model, also sigma.fv, nu.fv, tau.fv for the other parameters if present

The linear predictor of the extended mu model, also sigma.lp, nu.lp, tau.lp for the other parameters if present

The working variable of the extended mu model, also sigma.wv, nu.wv, tau.wv for the other parameters if present

The working weights of the mu model, also sigma.wt, nu.wt, tau.wt for the other parameters if present

The link function for the mu model, also sigma.link, nu.link, tau.link for the other parameters if present

The terms for the mu model, also sigma.terms, nu.terms, tau.terms for the other parameters if present

The design matrix for the mu, also sigma.x, nu.x, tau.x for the other parameters if present

The QR decomposition of the mu model, also sigma.qr, nu.qr, tau.qr for the other parameters if present

The linear coefficients of the mu model, also sigma.coefficients, nu.coefficients, tau.coefficients for the other parameters if present

The formula for the mu model, also sigma.formula, nu.formula, tau.formula for the other parameters if present

The mu degrees of freedom also sigma.df, nu.df, tau.df for the other parameters if present

The non linear degrees of freedom, also sigma.nl.df, nu.nl.df, tau.nl.df for the other parameters if present

The total degrees of freedom use by the model

The residual degrees of freedom left after the model is fitted

The original data set

The number of EM iterations

Whether the EM has converged

The residuals for the long fit

The estimates mass point (if "np" mixture is used)

The number of mass points used

Contains a matrix of posteriori probabilities,

The estimated mixture probabilities

The Akaike information criterion

The Bayesian information criterion

The formula used in the expanded fit

The random effect formula

Prior weights
the Empirical Bayes Predictions (Aitkin, 1996b) on the scale of the linear predictor.

Note that in case of Gaussian quadrature, the coefficient given at ‘z’ in coefficients corresponds to the standard deviation of the mixing distribution.

As a by-product, gamlssNP produces a plot showing the global deviance against the iteration number. Further, a plot with the EM trajectories is given. The x-axis corresponds to the iteration number, and the y-axis to the value of the mass points at a particular iteration. This plot is not produced when mixture is set to “gq”

Author(s)
Mikis Stasinopoulos based on function created by Jochen Einbeck John Hinde and Ross Darnell

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

See Also

gamlss, gamlss.family
Examples

data(enzyme)
# equivalent model using gamlssNP
mmNP1 <- gamlssNP(act~1, data=enzyme, random=-1, family=NO, K=2)
mmNP2 <- gamlssNP(act~1, data=enzyme, random=-1, sigma.fo=-MASS, family=NO, K=2)
AIC(mmNP1, mmNP2)

MX.control
The control function for gamlssMX

Description
The function sets controls for the gamlssMX function.

Usage
MX.control(cc = 1e-04, n.cyc = 200, trace = FALSE, seed = NULL, plot = TRUE, sample = NULL, ...)

Arguments
cc convergent criterion for the EM
n.cyc number of cycles for EM
trace whether to print the EM iterations
seed a number for setting the seeds for starting values
plot whether to plot the sequence of global deviance up to convergence
sample how large the sample to be in the starting values
... for extra arguments

Value
Returns a list

Author(s)
Mikis Stasinopoulos and Bob Rigby

References

NP.control


See Also
gamlss, gamlssMX, gamlssMXfits

NP.control

Control function for gamlssNP

Description

This is a control function for gamlssNP function.

Usage

NP.control(EMcc = 0.001, EMn.cyc = 200, damp = TRUE, trace = TRUE, plot.opt = 3, ...)

Arguments

EMcc convergence criterion for the EM
EMn.cyc number of cycles for the EM
damp Not in used
trace whether to print the EM iterations
plot.opt plotting the
... for extra arguments

Value

Returns a list.

Author(s)

Mikis Stasinopoulos
References


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

See Also

gamlss, gamlssNP

Description

A utility function for plotting two dimension non-parametric distribution. The function uses the persp() function.

Usage

plotMP(x, y, prob, theta = 20, phi = 20, expand = 0.5, col = "lightblue", xlab = "intercept", ylab = "slope", ...)

Arguments

x a vector containg points in the x axis
y a vector containg points in the y axis
prob vector containing probabilities which should add up to one
theta, phi, expand, col arguments to pass to the persp() function
xlab the x label
ylab the y label
... additinal argument to be passed to persp()
**Details**

The function call

**Value**

A graph is produced.

**Author(s)**

Mikis Stasinopoulos

**References**


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

**See Also**

`gamlssNP`, `persp`

**Examples**

```r
gamma_0 <- c( -4.4, -3,-2.2, -.5, 0.1, 1, 1.5, 2.2, 3.5, 4.1 )
gamma_1 <- c( 2.2, 1.2, 0.1, -1, -2.3, -4.6 , 5.1, -3.2, 0.1, -1.2) prob <- c(0.1, .05, .12, 0.25, 0.08, 0.12, 0.10, 0.05, 0.10, 0.03 ) plotMP(gamma_0, gamma_1,prob)
```
Index

* datasets
  - enzyme, 5

* distribution
  - dMX, 3
    - gamlss.mix-package, 2

* package
  - gamlss.mix-package, 2

* regression
  - dMX, 3
    - gamlss.mix-package, 2
    - gamlssMX, 6
    - gamlssNP, 8
    - MX.control, 12
    - NP.control, 13
    - plotMP, 14

brains (enzyme), 5

dMX, 3

enzyme, 5

- gamlss, 3, 7, 11, 13, 14
- gamlss.family, 3, 7, 11
- gamlss.mix-package, 2
- gamlss.mx (gamlss.mix-package), 2
- gamlssMX, 6, 13
- gamlssMXfits, 13
- gamlssMXfits (gamlssMX), 6
- gamlssNP, 8, 14, 15
- getpdfMX (dMX), 3
- getpdfNP (dMX), 3

- MX.control, 12

- NP.control, 13

- persp, 15
- plotMP, 14
- pMX (dMX), 3