Package ‘lqmm’

December 13, 2019

Type Package
Title Linear Quantile Mixed Models
Version 1.5.5
Date 2019-12-12
Author Marco Geraci
Maintainer Marco Geraci <geraci@mailbox.sc.edu>
Depends R (>= 3.0.0)
Imports stats, utils, nlme (>= 3.1-124), SparseGrid
License GPL (>= 2)
LazyLoad yes
RoxygenNote 7.0.2
Encoding UTF-8
NeedsCompilation yes
Repository CRAN
Date/Publication 2019-12-13 05:40:13 UTC

R topics documented:

lqmm-package ........................................ 2
boot .................................................. 3
coef.lqm .............................................. 5
coef.lqmm ........................................... 6
covHandling ........................................ 6
dal .................................................... 7
### lqmm-package

**Linear Quantile Models and Linear Quantile Mixed Models**

**Description**

Fit quantile regression models for independent and hierarchical data
Details

Package: lqmm
Type: Package
Version: 1.5.5
Date: 2019-12-12
License: GPL (>=2)
LazyLoad: yes

Author(s)

Marco Geraci
Maintainer: Marco Geraci <geraci@mailbox.sc.edu>

References


boot

Bootstrap functions for LQM and LQMM

Description

This function is used to obtain a bootstrap sample of a fitted LQM or LQMM. It is a generic function.

Usage

boot(object, R = 50, seed = round(runif(1, 1, 10000)), startQR = FALSE)
## S3 method for class 'lqm'
boot(object, R = 50, seed = round(runif(1, 1, 10000)), startQR = FALSE)
## S3 method for class 'lqmm'
boot(object, R = 50, seed = round(runif(1, 1, 10000)), startQR = FALSE)
Arguments

object an object of class "lqm" or "lqmm".
R number of bootstrap replications.
seed optional random number generator seed.
startQR logical flag. If TRUE the estimated parameters in object are used as starting values in the algorithm applied to each bootstrap sample. This may cause the algorithm to converge too often to a similar optimum, which would ultimately result in underestimated standard errors. If FALSE (recommended), starting values are based on lm.

Value

An object of class boot.lqm is a data frame with R rows and npars columns containing the bootstrap estimates of theta. If object contains results for multiple quantiles, boot.lqm returns an array of dimension c(R, npars, nt), where nt is the length of tau.

An object of class boot.lqmm is a data frame with R rows and npars columns containing the bootstrap estimates of theta_x, theta_z, and scale. If object contains results for multiple quantiles, boot.lqmm returns an array of dimension c(R, npars, nt), where nt is the length of tau. The elements of theta_z are labelled with reStruct. See function covHandling and the example below on how to derive the variance-covariance matrix of the random effects starting from theta_z.

The following attributes are available:

tau index of the quantile(s).
estimated the estimated parameter as given by object.
R number of bootstrap replications.
seed the random number generator seed used to produce the bootstrap sample.
npars total number of parameters.
rdf the number of residual degrees of freedom.
indices the bootstrap sample of independent data units.

Author(s)

Marco Geraci

Examples

# boot.lqm
set.seed(123)
n <- 500
test <- data.frame(x = runif(n, 0, 1))
test$y <- 30 + test$x + rnorm(n)
fit.lqm <- lqm(y ~ x, data = test, tau = 0.5)
fit.boot <- boot(fit.lqm)
str(fit.boot)
# boot.lqmm

data(Orthodont)
fit <- lqmm(distance ~ age, random = ~ 1, group = Subject, 
           tau = 0.5, data = Orthodont)
fit.boot <- boot(fit)
str(fit.boot)

coefficients of LQM

---

coeff.lqm

Extract LQM Coefficients

Description

coef extracts model coefficients from lqm, lqm.counts objects.

Usage

## S3 method for class 'lqm'
coef(object, ...)

Arguments

object an lqm or lqm.counts object.

... not used.

Value

a vector for single quantiles or a matrix for multiple quantiles.

Author(s)

Marco Geraci

See Also

lqm summary.lqm lqm.counts
\texttt{coef.lqmm} \quad \textit{Extract LQMM Coefficients}

\section*{Description}
\texttt{coef} extracts model coefficients from \texttt{lqmm} objects.

\section*{Usage}
\begin{verbatim}
## S3 method for class 'lqmm'
coef(object, ...)
\end{verbatim}

\section*{Arguments}
- \texttt{object} \quad a fitted object of \texttt{class "lqmm"}.
- \texttt{...} \quad not used.

\section*{Value}
a vector for single quantiles or a matrix for multiple quantiles.

\section*{Author(s)}
Marco Geraci

\section*{See Also}
- \texttt{lqmm}
- \texttt{summary.lqmm}

\texttt{covHandling} \quad \textit{Variance-Covariance Matrix}

\section*{Description}
This is an auxiliary function.

\section*{Usage}
\begin{verbatim}
covHandling(theta, n, cov\_name, quad\_type)
\end{verbatim}

\section*{Arguments}
- \texttt{theta} \quad unique parameters of the variance-covariance matrix of the random effects as returned by \texttt{lqmm} in \texttt{theta\_z}.
- \texttt{n} \quad dimension of the vector of random effects.
- \texttt{cov\_name} \quad see argument covariance in \texttt{lqmm}.
- \texttt{quad\_type} \quad type of quadrature "c("normal","robust")".
**Description**

Density, distribution function, quantile function and random generation for the asymmetric Laplace distribution.

**Usage**

dal(x, mu = 0, sigma = 1, tau = 0.5, log = FALSE)
pal(x, mu = 0, sigma = 1, tau = 0.5)
qal(x, mu = 0, sigma = 1, tau = 0.5)
ral(n, mu = 0, sigma = 1, tau = 0.5)

**Arguments**

- **x**: vector of quantiles (dal, pal) or probabilities (qal).
- **n**: number of observations.
- **mu**: location parameter.
- **sigma**: positive scale parameter.
- **tau**: skewness parameter (0,1).
- **log**: logical; if TRUE, probabilities are log–transformed.

**Details**

The asymmetric Laplace distribution with parameters (mu, sigma, tau) has density

\[
f(x) = \tau (1 - \tau) / \sigma e^{-(1/2\tau)(\theta_{\max}(x,0) + (1-\theta)\max(-x,0))}
\]

**Author(s)**

Marco Geraci

**See Also**

VarCorr.lqmm, lqmm, lqm
extractBoot

Extract Fixed and Random Bootstrapped Parameters

Description

This generic function extracts the fixed and random components of bootstrapped estimates of an lqmm object.

Usage

extractBoot(object, which = "fixed")
## S3 method for class 'boot.lqmm'
extractBoot(object, which = "fixed")

Arguments

object an object of class boot.lqmm.
which character indicating whether "fixed" or "random" parameters.

Details

The "random" parameters refer to the "raw" parameters of the variance-covariance matrix of the random effects as returned by lqmm.fit.gs and lqmm.fit.df.

Value

a matrix of bootstrapped estimates.

Author(s)

Marco Geraci

See Also

boot.lqmm, lqmm.fit.gs, lqmm.fit.df

Examples

## Orthodont data
data(Orthodont)

# Random intercept model
fit <- lqmm(distance ~ age, random = ~ 1, group = Subject,
tau = 0.5, data = Orthodont)
fit.boot <- boot(fit)

# extract fixed effects
B <- extractBoot(fit.boot, which = "fixed")
# covariance matrix estimated fixed parameters
cov(B)

### gauss.quad

**Gaussian Quadrature**

**Description**

This function calculates nodes and weights for Gaussian quadrature. See help("gauss.quad") from package statmod.

**Author(s)**

Original version by Gordon Smyth

**Source**


### gauss.quad.prob

**Gaussian Quadrature**

**Description**

This function calculates nodes and weights for Gaussian quadrature in terms of probability distributions. See help("gauss.quad.prob") from package statmod.

**Author(s)**

Original version by Gordon Smyth

**Source**

**is.positive.definite  Test for Positive Definiteness**

**Description**

This function tests whether all eigenvalues of a symmetric matrix are positive. See help("is.positive.definite") from package corpcor.

**Author(s)**

Original version by Korbinian Strimmer

**Source**


---

**labor  Labor Pain Data**

**Description**

The labor data frame has 358 rows and 4 columns of the change in pain over time for several 83 women in labor.

**Format**

This data frame contains the following columns:

- **subject** an ordered factor indicating the subject on which the measurement was made. The levels are labelled 1 to 83.
- **pain** a numeric vector of self-reported pain scores on a 100mm line.
- **treatment** a dummy variable with values 1 for subjects who received a pain medication and 0 for subjects who received a placebo.
- **time** a numeric vector of times (minutes since randomization) at which pain was measured.

**Details**

The labor pain data were reported by Davis (1991) and successively analyzed by Jung (1996) and Geraci and Bottai (2007). The data set consists of repeated measurements of self-reported amount of pain on N = 83 women in labor, of which 43 were randomly assigned to a pain medication group and 40 to a placebo group. The response was measured every 30 min on a 100-mm line, where 0 means no pain and 100 means extreme pain. A nearly monotone pattern of missing data was found for the response variable and the maximum number of measurements for each woman was six.
**Source**


**References**


---

**logLik.lqm**

*Extract Log-Likelihood*

**Description**

`logLik.lqm` extracts the log-likelihood of a fitted LQM.

**Usage**

```r
## S3 method for class 'lqm'
logLik(object, ...)
```

**Arguments**

- `object`: an object of class "lqm".
- `...`: not used.

**Author(s)**

Marco Geraci

**See Also**

- `lqm`  
- `AIC`
logLik.lqmm  

Extract Log-Likelihood

Description

logLik.lqmm extracts the log-likelihood of a fitted LQMM.

Usage

## S3 method for class 'lqmm'
logLik(object, ...)

Arguments

object  
an object of class "lqmm".

...  
not used.

Author(s)

Marco Geraci

See Also

lqmm  
lqm AIC

lqm  

Fitting Linear Quantile Models

Description

lqm is used to fit linear quantile models based on the asymmetric Laplace distribution.

Usage

lqm(formula, data, subset, na.action, weights = NULL, tau = 0.5,  
contrasts = NULL, control = list(), fit = TRUE)

Arguments

formula  
an object of class formula for fixed effects: a symbolic description of the model to be fitted.

data  
an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which lqm is called.
subset an optional vector specifying a subset of observations to be used in the fitting process.

na.action a function which indicates what should happen when the data contain NAs. The default is set by the na.action setting of options.

weights An optional vector of weights to be used in the fitting process.

tau the quantile(s) to be estimated. This must be a number between 0 and 1, otherwise the execution is stopped. If more than one quantile is specified, rounding off to the 4th decimal must give non–duplicated values of tau, otherwise the execution is stopped.

contrasts an optional list. See the contrasts.arg of model.matrix.default.

control list of control parameters of the fitting process. See lqmControl.

fit logical flag. If FALSE the function returns a list of arguments to be passed to lqm.fit.gs.

Details

The function computes an estimate on the tau-th quantile function of the response, conditional on the covariates, as specified by the formula argument. The quantile predictor is assumed to be linear. The function maximizes the (log)likelihood of a Laplace regression which is equivalent to the minimization of the weighted sum of absolute residuals (Koenker and Bassett, 1978). The optimization algorithm is based on the gradient of the Laplace log–likelihood (Bottai, Orsini and Geraci, 2013).

Value

lqm returns an object of class lqm.

The function summary is used to obtain and print a summary of the results.

An object of class lqm is a list containing the following components:

theta a vector of coefficients. theta is a named matrix of coefficients when tau is a vector of values.

scale the scale parameter.

gradient the gradient.

logLik the log–likelihood.

opt details on optimization (see lqm.fit.gs).

call the matched call.

term.labels names for theta.

terms the terms object used.

nobs the number of observations.

edf, dim_theta the length of theta.

rdf the number of residual degrees of freedom.

tau the estimated quantile(s).

x the model matrix.
y               the model response.
weights        the weights used in the fitting process (a vector of 1’s if weights = NULL).
InitialPar     starting values for theta.
control        list of control parameters used for optimization (see lqmControl).

Note

Updates/FAQ/news are published here http://marcogeraci.wordpress.com/. New versions are usually published here https://r-forge.r-project.org/R/?group_id=1396 before going on CRAN.

Author(s)

Marco Geraci

References


See Also

summary.lqm, coef.lqm, predict.lqm, residuals.lqm

Examples

set.seed(123)
n <- 500
p <- 1:3/4
test <- data.frame(x = runif(n,0,1))
test$y <- 30 + test$x + rnorm(n)
fit.lqm <- lqm(y ~ x, data = test, tau = p, control = list(verbose = FALSE, loop_tol_ll = 1e-9), fit = TRUE)
fit.lqm

lqm.counts  Quantile Regression for Counts

Description

This function is used to fit a quantile regression model when the response is a count variable.
Usage

lqm.counts(formula, data, weights = NULL, offset = NULL, contrasts = NULL,
          tau = 0.5, M = 50, zeta = 1e-05, B = 0.999, cn = NULL, alpha = 0.05,
          control = list())

Arguments

formula  an object of class formula: a symbolic description of the model to be fitted.
data     an optional data frame, list or environment (or object coercible by as.data.frame to a data frame) containing the variables in the model. If not found in data, the variables are taken from environment(formula), typically the environment from which lqm is called.
weights  an optional vector of weights to be used in the fitting process.
offset   an optional offset to be included in the model frame.
contrasts an optional list. See the contrasts.arg of model.matrix.default.
tau      quantile to be estimated.
M        number of dithered samples.
zeta     small constant (see References).
B         right boundary for uniform random noise U[0,B] to be added to the response variable (see References).
alpha    small constant to be passed to F.lqm (see References).
control  list of control parameters of the fitting process. See lqmControl.

Details

A linear quantile regression model if fitted to the log–transformed response. Additional transformation functions will be implemented. The notation used here follows closely that of Machado and Santos Silva (2005).

Value

an object of class "lqm.counts" containing the following components

tau      the estimated quantile.
theta    regression quantile (on the log–scale).
fitted   predicted quantile (on the response scale).
tTable   coefficients, standard errors, etc.
x        the model matrix.
y        the model response.
offset   offset.
nobs     the number of observations.
M        specified number of dithered samples for standard error estimation.
Mn

actual number of dithered samples used for standard error estimation that gave an invertible D matrix (Machado and Santos Silva, 2005).

term.labels

names for theta.

terms

the terms object used.

rdf

the number of residual degrees of freedom.

InitialPar

starting values for theta.

control

list of control parameters used for optimization (see lqmControl).

Author(s)

Marco Geraci

References


Examples

```r
n <- 100
x <- runif(n)
test <- data.frame(x = x, y = rpois(n, 2*x))
lqm.counts(y ~ x, data = test, M = 50)
```

lqm.fit.gs

Quantile Regression Fitting by Gradient Search

Description

This function controls the arguments to be passed to routines written in C for LQM estimation. The optimization algorithm is based on the gradient of the Laplace log–likelihood (Bottai, Orsini and Geraci, 2013).

Usage

```r
lqm.fit.gs(theta, x, y, weights, tau, control)
```

Arguments

theta

starting values for the regression coefficients.

x

the model matrix.

y

the model response.

weights

the weights used in the fitting process.

tau

the quantile to be estimated.

control

list of control parameters used for optimization (see lqmControl).
Details

See argument `fit` in `lqm` for generating a list of arguments to be called by this function.

Value

An object of class `list` containing the following components:

- `theta`: a vector of coefficients.
- `scale`: the scale parameter.
- `gradient`: the gradient.
- `logLik`: the log–likelihood.
- `opt`: number of iterations when the estimation algorithm stopped.

Author(s)

Marco Geraci

References


See Also

`lqm`

Examples

```r
set.seed(123)
n <- 500
test <- data.frame(x = runif(n,0,1))
test$y <- 30 + test$x + rnorm(n)
lqm.ls <- lqm(y ~ x, data = test, fit = FALSE)
do.call("lqm.fit.gs", lqm.ls)
```
lqmControl

Control parameters for lqm estimation

Description
A list of parameters for controlling the fitting process.

Usage
lqmControl(method = "gs1", loop_tol_ll = 1e-5, loop_tol_theta = 1e-3, check_theta = FALSE, loop_step = NULL, beta = 0.5, gamma = 1.25, reset_step = FALSE, loop_max_iter = 1000, smooth = FALSE, omicron = 0.001, verbose = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>method</td>
<td>character vector that specifies which code to use for carrying out the gradient search algorithm: &quot;gs1&quot; (default) based on C code and &quot;gs2&quot; based on R code. Method &quot;gs3&quot; uses a smoothed loss function. See details.</td>
</tr>
<tr>
<td>loop_tol_ll</td>
<td>tolerance expressed as relative change of the log-likelihood.</td>
</tr>
<tr>
<td>loop_tol_theta</td>
<td>tolerance expressed as relative change of the estimates.</td>
</tr>
<tr>
<td>check_theta</td>
<td>logical flag. If TRUE the algorithm performs a check on the change in the estimates in addition to the likelihood.</td>
</tr>
<tr>
<td>loop_step</td>
<td>step size (default standard deviation of response).</td>
</tr>
<tr>
<td>beta</td>
<td>decreasing step factor for line search (0,1).</td>
</tr>
<tr>
<td>gamma</td>
<td>nondecreasing step factor for line search (&gt;= 1).</td>
</tr>
<tr>
<td>reset_step</td>
<td>logical flag. If TRUE the step size is re-setted to the initial value at each iteration.</td>
</tr>
<tr>
<td>loop_max_iter</td>
<td>maximum number of iterations.</td>
</tr>
<tr>
<td>smooth</td>
<td>logical flag. If TRUE the standard loss function is replaced with a smooth approximation.</td>
</tr>
<tr>
<td>omicron</td>
<td>small constant for smoothing the loss function when using smooth = TRUE. See details.</td>
</tr>
<tr>
<td>verbose</td>
<td>logical flag.</td>
</tr>
</tbody>
</table>

Details
The methods "gs1" and "gs2" implement the same algorithm (Bottai et al, 2015). The former is based on C code, the latter on R code. While the C code is faster, the R code seems to be more efficient in handling large datasets. For method "gs2", it is possible to replace the classical non-differentiable loss function with a smooth version (Chen, 2007).

Value
a list of control parameters.
Author(s)

Marco Geraci

References


See Also

lqm

lqmm

Fitting Linear Quantile Mixed Models

Description

lqmm is used to fit linear quantile mixed models based on the asymmetric Laplace distribution.

Usage

lqmm(fixed, random, group, covariance = "pdDiag", tau = 0.5, nK = 7, type = "normal", rule = 1, data = sys.frame(sys.parent()), subset, weights, na.action = na.fail, control = list(), contrasts = NULL, fit = TRUE)

Arguments

fixed an object of class formula for fixed effects: a symbolic description of the model to be fitted.
random a one-sided formula of the form ~x1 + x2 + ... + xn for random effects: a symbolic description of the model to be fitted.
group grouping factor.
covariance variance–covariance matrix of the random effects. Default is pdDiag (see details).
tau the quantile(s) to be estimated.
nK number of quadrature knots.
type type of quadrature "c("normal","robust")" (see details).
rule quadrature rule (see details).
data an optional data frame containing the variables named in fixed, random and group. By default the variables are taken from the environment from which lqmm is called.
subset  an optional vector specifying a subset of observations to be used in the fitting process.
weights an optional vector of weights to be used in the fitting process of the same length as the number of rows of data. Weights are given to clusters, therefore units within the same cluster receive the same weight (see details).
na.action a function that indicates what should happen when the data contain NAs. The default action (na.fail) causes lqmm to print an error message and terminate if there are any incomplete observations.
control list of control parameters of the fitting process. See lqmmControl.
contrasts not yet implemented.
fit logical flag. If FALSE the function returns a list of arguments to be passed to lqmm.fit.

Details
The function computes an estimate on the tau-th quantile function of the response, conditional on the covariates, as specified by the formula argument, and on random effects, as specified by the random argument. The quantile predictor is assumed to be linear. The function maximizes the (log)likelihood of the Laplace regression proposed by Geraci and Bottai (2014). The likelihood is numerically integrated via Gaussian quadrature techniques. The optimization algorithm is based on the gradient of the Laplace log–likelihood (control = list(method = "gs")). An alternative optimization algorithm is based on a Nelder-Mead algorithm (control = list(method = "df")) via optim. The scale parameter is optimized in a refinement step via optimize.

Quadrature approaches include Gauss-Hermite (type = "normal") and Gauss-Laguerre (type = "robust") quadrature. The argument rule takes one of the following: 1 (product rule quadrature), 2 (sparse grid quadrature), 3 (nested quadrature rule - only for type = "normal"), 4 (quadrature rule with the smallest number of nodes between rules 1 or 2). Rules 2 and 3 have not yet been tested extensively.

Different standard types of positive–definite matrices for the random effects can be specified: pdIdent multiple of an identity; pdCompSymm compound symmetry structure (constant diagonal and constant off–diagonal elements); pdDiag diagonal; pdSymm general positive–definite matrix, with no additional structure.

Weights are given to clusters, therefore it is expected that these are constant within cluster. When the weights are specified in the main call, then the first value by group in the vector weights will be replicated for the same length of each group. Alternatively, different weights within the same cluster can be introduced with a direct call to lqmm.fit.gs or lqmm.fit.df.

The lqmm vignette can be accessed by typing help(package = "lqmm") and then following the link 'User guides, package vignettes and other documentation'.

Value
lqmm returns an object of class lqmm.
The function summary is used to obtain and print a summary of the results.
An object of class lqmm is a list containing the following components:
theta a vector containing fixed regression coefficients and parameters of the variance–covariance matrix of the random effects. See VarCorr.lqmm to extract the variance–covariance of the random effects from an "lqmm" object.

theta_x, theta_z
partition of theta: fixed regression coefficients (theta_x) and unique variance–covariance parameters (theta_z).

scale the scale parameter.

gradient the gradient (control = list(method = "gs")).

logLik the log–likelihood.

opt details on optimization (see lqmm.fit.gs and lqmm.fit.df).

call the matched call.

nn column names of mmf.

mm column names of mmr.

nobs the number of observations.

dim_theta the number of columns in mmf and mmr.

dim_theta_z the length of theta_z.

edf length of theta.

rdf the number of residual degrees of freedom.

df edf + 1 (scale parameter).

tau the estimated quantile(s).

mmf the model matrix – fixed effects.

mmr the model matrix – random effects.

y the model response.

revOrder original order of observations (now ordered according to group).

weights the likelihood weights used in the fitting process (a vector of 1's if weights is missing or NULL).

group the grouping factor.

ngroups the number of groups.

QUAD quadrature nodes and weights.

type the type of quadrature.

rule quadrature rule.

InitialPar starting values for theta.

control list of control parameters used for optimization (see lqmmControl).

cov_name class of variance-covariance matrix for the random effects.

mfArgs arguments for model.frame to return the full data frame.

Note

Updates/FAQ/news are published here http://marcogeraci.wordpress.com/. New versions are usually published here https://r-forge.r-project.org/R/?group_id=1396 before going on CRAN.
Author(s)

Marco Geraci

References


See Also

lqm,summary.lqmm,coef.lqmm,VarCorr.lqmm,predict.lqmm,residuals.lqmm

Examples

# Test example
set.seed(123)

M <- 50
n <- 10
test <- data.frame(x = runif(n*M,0,1), group = rep(1:M,each=n))
test$y <- 10*test$x + rep(rnorm(M, 0, 2), each = n) + rchisq(n*M, 3)
fit.lqmm <- lqmm(fixed = y ~ x, random = ~ 1, group = group,
data = test, tau = 0.5, nK = 11, type = "normal")
fit.lqmm

#Call: lqmm(fixed = y ~ x, random = ~1, group = group, tau = 0.5, nK = 11, type = "normal")
#Quantile 0.5

#Fixed effects:
#(Intercept) x
# 3.443 9.258

#Covariance matrix of the random effects:
#(Intercept)
# 3.426

#Residual scale parameter: 0.8697 (standard deviation 2.46)
#Log-likelihood: -1178
```r
# Number of observations: 500
# Number of groups: 50

## Orthodont data
data(Orthodont)

# Random intercept model
fitOi.lqmm <- lqmm(distance ~ age, random = ~ 1, group = Subject,
tau = c(0.1, 0.5, 0.9), data = Orthodont)
coef(fitOi.lqmm)

# Random slope model
fitOs.lqmm <- lqmm(distance ~ age, random = ~ age, group = Subject,
tau = c(0.1, 0.5, 0.9), cov = "pdDiag", data = Orthodont)

# Extract estimates
VarCorr(fitOs.lqmm)
coef(fitOs.lqmm)
ranef(fitOs.lqmm)

# AIC
AIC(fitOi.lqmm)
AIC(fitOs.lqmm)
```

---

**Description**

This function controls the arguments to be passed to `optim` and `optimize` for LQMM estimation.

**Usage**

`lqmm.fit.df(theta_0, x, y, z, weights, cov_name, V, W, sigma_0, tau, group, control)`

**Arguments**

- `theta_0`: starting values for the linear predictor.
- `x`: the model matrix for fixed effects (see details).
- `y`: the model response (see details).
- `z`: the model matrix for random effects (see details).
- `weights`: the weights used in the fitting process (see details).
- `cov_name`: variance–covariance matrix of the random effects. Default is `pdIdent`. See details.
V nodes of the quadrature.
W weights of the quadrature.
sigma_0 starting value for the scale parameter.
tau the quantile(s) to be estimated.
group the grouping factor (see details).
control list of control parameters used for optimization (see lqmmControl).

Details

In lqmm, see argument fit for generating a list of arguments to be called by this function; see argument covariance for alternative variance–covariance matrices.

NOTE: the data should be ordered by group when passed to lqmm.fit.df (such ordering is performed by lqmm).

Value

An object of class "list" containing the following components:

theta a vector of coefficients, including the "raw" variance–covariance parameters (see VarCorr.lqmm).
scale the scale parameter.
logLik the log–likelihood.
opt number of iterations when the estimation algorithm stopped for lower (theta) and upper (scale) loop.

Author(s)

Marco Geraci

See Also

lqmm

Examples

set.seed(123)

M <- 50
n <- 10
test <- data.frame(x = runif(n*M,0,1), group = rep(1:M,each=n))
test$y <- 10*test$x + rep(rnorm(M, 0, 2), each = n) + rchisq(n*M, 3)
lqmm.ls <- lqmm(fixed = y ~ x, random = ~ 1, group = group, data = test, fit = FALSE)

do.call("lqmm.fit.df", lqmm.ls)
Description

This function controls the arguments to be passed to routines written in C for LQMM estimation. The optimization algorithm is based on the gradient of the Laplace log–likelihood (Bottai, Orsini and Geraci, 2014; Geraci and Bottai, 2014).

Usage

```r
lqmm.fit.gs(theta_0, x, y, z, weights, cov_name, V, W, sigma_0, tau, group, control)
```

Arguments

- `theta_0`: starting values for the linear predictor.
- `x`: the model matrix for fixed effects (see details).
- `y`: the model response (see details).
- `z`: the model matrix for random effects (see details).
- `weights`: the weights used in the fitting process (see details).
- `cov_name`: variance–covariance matrix of the random effects. Default is `pdIdent`. See details.
- `V`: nodes of the quadrature.
- `W`: weights of the quadrature.
- `sigma_0`: starting value for the scale parameter.
- `tau`: the quantile(s) to be estimated.
- `group`: the grouping factor (see details).
- `control`: list of control parameters used for optimization (see `lqmmControl`).

Details

In `lqmm`, see argument `fit` for generating a list of arguments to be called by this function; see argument `covariance` for alternative variance–covariance matrices.

NOTE: the data should be ordered by `group` when passed to `lqmm.fit.gs` (such ordering is performed by `lqmm`).

Value

An object of class "list" containing the following components:

- `theta`: a vector of coefficients, including the "raw" variance–covariance parameters (see `VarCorr.lqmm`).
scale the scale parameter.
gradient the gradient.
logLik the log–likelihood.

opt number of iterations when the estimation algorithm stopped for lower (theta) and upper (scale) loop.

Author(s)
Marco Geraci

References


See Also
lqmm

Examples
set.seed(123)

M <- 50
n <- 10
test <- data.frame(x = runif(n*M,0,1), group = rep(1:M,each=n))
test$y <- 10*test$x + rep(rnorm(M, 0, 2), each = n) + rchisq(n*M, 3)
lqmm.ls <- lqmm(fixed = y ~ x, random = ~ 1, group = group,
data = test, fit = FALSE)
do.call("lqmm.fit.gs", lqmm.ls)
Usage

lqmmControl(method = "gs", LP_tol_ll = 1e-5, LP_tol_theta = 1e-5,
check_theta = FALSE, LP_step = NULL, beta = 0.5, gamma = 1,
reset_step = FALSE, LP_max_iter = 500, UP_tol = 1e-4,
UP_max_iter = 20, startQR = FALSE, verbose = FALSE)

Arguments

method character vector that specifies the estimation method: "gs" for gradient search
(default) and "df" for Nelder-Mead.
LP_tol_ll tolerance expressed as absolute change of the log-likelihood.
LP_tol_theta tolerance expressed as absolute change of theta.
check_theta logical flag. If TRUE the algorithm performs an additional check on the change
in the estimates.
LP_step step size (default standard deviation of response).
beta decreasing step factor for line search (0,1).
gamma nondecreasing step factor for line search (>= 1).
reset_step logical flag. If TRUE the step size is reset to the initial value at each iteration.
LP_max_iter maximum number of iterations.
UP_tol tolerance expressed as absolute change of the scale parameter.
UP_max_iter maximum number of iterations.
startQR logical flag. If FALSE (default) the least squares estimate of the fixed effects is
used as starting value of theta_x and scale. If TRUE the lqm estimate is used.
verbose logical flag.

Details

LP (lower loop) refers to the estimation of regression coefficients and variance-covariance parame-
ters. UP (upper loop) refers to the estimation of the scale parameter.

Value

a list of control parameters.

Author(s)

Marco Geraci

See Also

lqmm
**make.positive.definite**

*Compute Nearest Positive Definite Matrix*

**Description**

This function computes the nearest positive definite of a real symmetric matrix. See `help("make.positive.definite")` from package `corpcor`.

**Author(s)**

Original version by Korbinian Strimmer

**Source**


---

**meanAL**

*Functions for Asymmetric Laplace Distribution Parameters*

**Description**

Accessory functions.

**Usage**

```r
meanAL(mu, sigma, tau)
varAL(sigma, tau)
invvarAL(x, tau)
```

**Arguments**

- `mu`: location parameter.
- `sigma`: scale parameter.
- `tau`: skewness parameter.
- `x`: numeric value.

**Details**

`meanAL` computes the mean of an asymmetric Laplace with parameters `mu`, `sigma` and `tau`.
`varAL` computes the variance of an asymmetric Laplace with parameters `sigma` and `tau`.
`invvarAL` computes the scale parameter of an asymmetric Laplace with parameter `tau` and variance `x`. 

Author(s)

Marco Geraci

References


See Also

dal, mleAL

mleAL

Maximum Likelihood Estimation of Asymmetric Laplace Distribution

Description

This function estimates the parameters of an asymmetric Laplace distribution for a sample.

Usage

mleAL(x)

Arguments

x a numeric vector.

Value

an object of class list containing the following components:

m location parameter
sigma scale parameter
tau skewness parameter
r number of iterations

Author(s)

Marco Geraci

References


See Also

dal, meanAL
Description

The Orthodont data frame has 108 rows and 4 columns of the change in an orthodontic measurement over time for several young subjects.

Format

This data frame contains the following columns:

- **distance** a numeric vector of distances from the pituitary to the pterygomaxillary fissure (mm). These distances are measured on x-ray images of the skull.
- **age** a numeric vector of ages of the subject (yr).
- **Subject** an ordered factor indicating the subject on which the measurement was made. The levels are labelled M01 to M16 for the males and F01 to F13 for the females. The ordering is by increasing average distance within sex.
- **Sex** a factor with levels Male and Female

Details

Investigators at the University of North Carolina Dental School followed the growth of 27 children (16 males, 11 females) from age 8 until age 14. Every two years they measured the distance between the pituitary and the pterygomaxillary fissure, two points that are easily identified on x-ray exposures of the side of the head.

Source


**predict.lqm**

**Predictions from LQM Objects**

**Description**

This function computes predictions based on fitted linear quantile model.

**Usage**

```r
# S3 method for class 'lqm'
predict(object, newdata, interval = FALSE,
        level = 0.95, na.action = na.pass, ...)  
# S3 method for class 'lqm.counts'
predict(object, newdata,
        na.action = na.pass, ...)  
```

**Arguments**

- `object` an `lqm` or `lqm.counts` object.
- `newdata` an optional data frame in which to look for variables with which to predict. If omitted, the fitted values are used.
- `interval` logical flag. If `TRUE`, bootstrap percentile intervals for predictions are provided. This argument is for `lqm` objects only.
- `level` confidence level. This argument is for `lqm` objects only.
- `na.action` function determining what should be done with missing values in `newdata`. The default is to predict NA.
- `...` further arguments passed to `boot.lqm`.

**Value**

a vector or a matrix or an array of predictions.

**Author(s)**

Marco Geraci

**See Also**

`residuals.lqm`, `residuals.lqm.counts`, `lqm`, `lqm.counts`, `coef.lqm`, `boot.lqm`
Predictions from an lqmm Object

Description

The predictions at level 0 correspond to predictions based only on the fixed effects estimates. The predictions at level 1 are obtained by adding the best linear predictions of the random effects to the predictions at level 0. See details for interpretation. The function `predint` will produce $1-\alpha$ confidence intervals based on bootstrap centiles.

Usage

```r
## S3 method for class 'lqmm'
predict(object, level = 0, ...)
## S3 method for class 'lqmm'
predint(object, level = 0, alpha = 0.05, R = 50,
       seed = round(runif(1, 1, 10000)))
```

Arguments

- `object`: an `lqmm` object.
- `level`: an optional integer vector giving the level of grouping to be used in obtaining the predictions.
- `alpha`: $1-\alpha$ is the confidence level.
- `R`: number of bootstrap replications.
- `seed`: optional random number generator seed.
- `...`: not used.

Details

As discussed by Geraci and Bottai (2014), integrating over the random effects will give "weighted averages" of the cluster-specific quantile effects. These may be interpreted strictly as population regression quantiles only for the median ($\tau=0.5$). Therefore, predictions at the population level (code=0) should be interpreted analogously.

Value

A vector or a matrix of predictions for `predict.lqmm`. A data frame or a list of data frames for `predint.lqmm` containing predictions, lower and upper bounds of prediction intervals, and standard errors.

Author(s)

Marco Geraci
print.lqm

References

See Also
lqmm, ranef.lqmm, coef.lqmm

Examples
## Orthodont data
data(Orthodont)

# Random intercept model
fitO1.lqmm <- lqmm(distance ~ age, random = ~ 1, group = Subject, tau = c(0.1, 0.5, 0.9), data = Orthodont)

# Predict (y - Xb)
predict(fitO1.lqmm, level = 0)

# Predict (y - Xb - Zu)
predict(fitO1.lqmm, level = 1)

# 95% confidence intervals
predint(fitO1.lqmm, level = 0, alpha = 0.05)

print.lqm

Print LQM Objects

Description
Print an object generated by lqm or lqm.counts.

Usage
## S3 method for class 'lqm'
print(x, digits = max(6, getOption("digits")), ...)

Arguments

x
an lqm or lqm.counts object.
digits
a non-null value for digits specifies the minimum number of significant digits to be printed in values.
... not used.

Author(s)
Marco Geraci
See Also

lqm, lqm.counts

---

**print.lqmm**  
*Print an lqmm Object*

**Description**  
Print an object generated by `lqmm`.

**Usage**  

```r
## S3 method for class 'lqmm'
print(x, digits = max(3, getOption("digits") - 3), ...)  
```

**Arguments**  

- `x`  
  - an `lqmm` object.
- `digits`  
  - a non-null value for `digits` specifies the minimum number of significant digits to be printed in values.
- `...`  
  - not used.

**Author(s)**  

Marco Geraci

See Also

lqmm

---

**print.summary.lqm**  
*Print an lqm Summary Object*

**Description**  
Print summary of an lqm object.

**Usage**  

```r
## S3 method for class 'summary.lqm'
print(x, ...)  
```

**Arguments**  

- `x`  
  - a `summary.lqm` object.
- `...`  
  - not used.
print.summary.lqmm

Author(s)
Marco Geraci

See Also
lqm, summary.lqm

print.summary.lqmm  Print an lqmm Summary Object

Description
Print summary of an lqmm object.

Usage
```
## S3 method for class 'summary.lqmm'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments
- `x` a summary.lqmm object.
- `digits` a non-null value for digits specifies the minimum number of significant digits to be printed in values.
- `...` not used.

Author(s)
Marco Geraci

See Also
lqm, summary.lqmm
**Description**

This function computes random effects for a linear quantile mixed model.

**Usage**

```r
## S3 method for class 'lqmm'
ranef(object, ...)
```

**Arguments**

- `object`: an object of class `lqmm`.
- `...`: not used.

**Details**

The prediction of the random effects is done via estimated best linear prediction (Geraci and Bottai, 2014). The generic function `ranef` is imported from the `nlme` package (Pinheiro et al, 2014).

**Value**

A data frame or a list of data frames of predicted random effects.

**Author(s)**

Marco Geraci

**References**


**See Also**

`lqmm`, `coef.lqmm`
residuals.lqm  

Residuals from an LQM Objects

Description
This function computes the residuals from a fitted linear quantile model.

Usage

## S3 method for class 'lqm'
residuals(object, ...)

Arguments

object  an lqm or lqm.counts object.

...  not used.

Value

a vector or matrix of residuals.

Author(s)

Marco Geraci

See Also

lqm, lqm.counts, predict.lqm, coef.lqm

residuals.lqmm  

Residuals from an lqmm Object

Description

The residuals at level 0 correspond to population residuals (based only on the fixed effects estimates). The residuals at level 1 are obtained by adding the best linear predictions of the random effects to the predictions at level 0 and the subtracting these from the model response.

Usage

## S3 method for class 'lqmm'
residuals(object, level = 0, ...)

Arguments

object: an lqmm object.
level: an optional integer vector giving the level of grouping to be used in obtaining the predictions. Level zero corresponds to the population residuals.

Value

a matrix of residuals.

Author(s)

Marco Geraci

References


See Also

lqmm, predict.lqmm, coef.lqmm, ranef.lqmm,

Description

Summary method for class boot.lqm.

Usage

## S3 method for class 'boot.lqm'
summary(object, alpha = 0.05, digits = max(3,getOption("digits") - 3), ...)

Arguments

object: an object of class lqm.
alpha: numeric value for the interval confidence level (1-alpha).
digits: a non-null value for digits specifies the minimum number of significant digits to be printed in values.
...: not used.

Author(s)

Marco Geraci
summary.boot.lqmm

See Also

boot.lqmm, lqmm,

summary.boot.lqmm  Summary for a boot.lqmm Object

Description

This function gives a summary of a bootstrapped lqmm object

Usage

## S3 method for class 'boot.lqmm'
summary(object, alpha = 0.05, digits = max(3, getOption("digits") - 3), ...)

Arguments

- object: an object of class lqmm.
- alpha: numeric value for the interval confidence level (1-alpha).
- digits: a non-null value for digits specifies the minimum number of significant digits to be printed in values.
- ...: not used.

Author(s)

Marco Geraci

References


See Also

boot.lqmm, lqmm,
Description

Summary method for class lqm.

Usage

```r
## S3 method for class 'lqm'
summary(object, method = "boot", alpha = 0.05, covariance = FALSE, ...)
```

Arguments

- `object`: an object of class `lqm`
- `method`: specifies the method used to compute standard errors: "boot" for bootstrap (default), "nid" for large sample approximations under `nid` assumptions.
- `alpha`: significance level.
- `covariance`: logical flag. If TRUE the covariance matrix of the bootstrap estimates is provided.
- `...`: see `boot.lqm` for additional arguments.

Details

`print.summary.lqm` formats the coefficients, standard errors, etc. and additionally gives 'significance stars'.

Value

an object of class `summary.lqm`. The function `summary.lqm` computes and returns a list of summary statistics of the fitted linear quantile mixed model given in `object`, using the components (list elements) from its argument, plus

- `Cov`: the covariance matrix obtained from the bootstrapped estimates (if `covariance` = TRUE).
- `tTable`: a matrix with estimates, standard errors, etc.

Author(s)

Marco Geraci

Source

The code for the "nid" method has been adapted from the function `summary.rq` in package quantreg. It depends on the function `bandwidth.rq`.

See Also

print.summary.lqm

Examples

```r
set.seed(12356)
n <- 200
p <- 1:3/4
test <- data.frame(x = runif(n,0,1))
test$y <- 30 + test$x + rnorm(n)
fit.lqm <- lqm(y ~ x, data = test, tau = p)
summary(fit.lqm, R = 50)
```

**summary.lqmm**  
*Summary for an lqmm Object*

**Description**

Summary method for class lqmm.

**Usage**

```r
## S3 method for class 'lqmm'
summary(object, method = "boot", alpha = 0.05, covariance = FALSE, ...)
```

**Arguments**

- `object` an object of class lqmm.
- `method` specifies the method used to compute standard errors. Currently, only the bootstrap method ("boot") is available.
- `alpha` significance level.
- `covariance` logical flag. If TRUE the covariance matrix of the bootstrap estimates is provided.
- `...` see boot.lqmm for additional arguments.

**Details**

print.summary.lqmm formats the coefficients, standard errors, etc. and additionally gives ‘significance stars’.
VarCorr.lqmm

Value

an object of class summary.lqmm. The function summary.lqmm computes and returns a list of summary statistics of the fitted linear quantile mixed model given in object, using the components (list elements) from its argument, plus

- Cov: the covariance matrix obtained from the bootstrapped estimates (if covariance = TRUE).
- tTable: a matrix with estimates, standard errors, etc.
- B: the matrix of all bootstrapped parameters.

Author(s)

Marco Geraci

See Also

print.summary.lqmm lqmm

Examples

data(Orthodont)
fitOi.lqmm <- lqmm(distance ~ age, random = ~ 1, group = Subject, tau = c(0.1,0.5,0.9), data = Orthodont)
summary(fitOi.lqmm)

---

**VarCorr.lqmm**

*Extract Variance-Covariance Matrix*

Description

This function extracts the variance-covariance matrix of the random effects from a fitted lqmm object.

Usage

```r
## S3 method for class 'lqmm'
VarCorr(x, sigma = NULL, ...)
```

Arguments

- `x`: an object of class "lqmm".
- `sigma`: not used.
- `...`: not used.
Details

This function returns the variance or the variance-covariance matrix of the random effects. It calls \texttt{covHandling} to manage the output of \texttt{lqmm.fit.gs} or \texttt{lqmm.fit.df}. A post-fitting approximation to the nearest positive (semi)definite matrix (Higham, 2002) is applied if necessary. The generic function \texttt{VarCorr} is imported from the \texttt{nlme} package (Pinheiro et al, 2014).

Author(s)

Marco Geraci

References


See Also

\texttt{lqmm} \texttt{coef.lqmm}
Index

*Topic asymmetric Laplace distribution
dal, 7
meanAL, 28
mleAL, 29

*Topic bootstrap
boot, 3
extractBoot, 8
summary.boot.lqm, 38
summary.boot.lqmm, 39
summary.lqm, 40
summary.lqmm, 41

*Topic coefficients
coef.lqm, 5
coeef.lqmm, 6
ranef.lqmm, 36
VarCorr.lqmm, 42

*Topic control
lqmControl, 18
lqmmControl, 26

*Topic covariance
covHandling, 6
is.positive.definite, 10
make.positive.definite, 28
VarCorr.lqmm, 42

*Topic datasets
labor, 10
Orthodont, 30

*Topic fitting
lqm.fit.gs, 16
lqmControl, 18
lqmm.fit.df, 23
lqmm.fit.gs, 25
lqmmControl, 26

*Topic gaussian quadrature
gauss.quad, 9
gauss.quad.prob, 9

*Topic maximum likelihood estimation
meanAL, 28
mleAL, 29

*Topic models
logLik.lqm, 11
logLik.lqmm, 12

*Topic positive definite
is.positive.definite, 10
make.positive.definite, 28

*Topic prediction
predict.lqm, 31
predict.lqmm, 32

*Topic print
print.lqm, 33
print.lqmm, 34
print.summary.lqm, 34
print.summary.lqmm, 35

*Topic quantile regression
lqm, 12
lqmm, 19
lqmm-package, 2

*Topic quantiles for counts
lqm.counts, 14

*Topic random effects
ranef.lqmm, 36

*Topic residuals
residuals.lqm, 37
residuals.lqmm, 37

*Topic standard errors
boot, 3
summary.lqm, 40
summary.lqmm, 41

*Topic summary
print.summary.lqm, 34
print.summary.lqmm, 35
summary.boot.lqm, 38
summary.boot.lqmm, 39

AIC, 11, 12
as.data.frame, 12
<table>
<thead>
<tr>
<th>Page</th>
<th>Index</th>
<th>Entries</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>boot</td>
<td></td>
</tr>
<tr>
<td>31, 39, 40</td>
<td>boot.lqm</td>
<td></td>
</tr>
<tr>
<td>8, 39, 41</td>
<td>boot.lqmm</td>
<td></td>
</tr>
<tr>
<td>4, 6, 8, 11–13, 20, 36, 38–42</td>
<td>class</td>
<td></td>
</tr>
<tr>
<td>5, 14, 31, 37</td>
<td>coef.lqm</td>
<td></td>
</tr>
<tr>
<td>6, 22, 33, 36, 38, 43</td>
<td>coef.lqmm</td>
<td></td>
</tr>
<tr>
<td>4, 6, 43</td>
<td>covHandling</td>
<td></td>
</tr>
<tr>
<td>7, 29</td>
<td>dal</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>extractBoot</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>F.lqm</td>
<td></td>
</tr>
<tr>
<td>12, 15, 19</td>
<td>formula</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>gauss.quad</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>gauss.quad.prob</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>invvarAL (meanAL)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>is.positive.definite</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>labor</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>lm</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>logLik.lqm</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>logLik.lqmm</td>
<td></td>
</tr>
<tr>
<td>13, 16</td>
<td>lqm</td>
<td></td>
</tr>
<tr>
<td>5, 7, 11, 12, 17, 19, 22, 27, 31, 33–35, 37, 39, 41</td>
<td>lqm.counts</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>lqmm</td>
<td></td>
</tr>
<tr>
<td>13–16, 18</td>
<td>lqmmControl</td>
<td></td>
</tr>
<tr>
<td>6, 7, 12, 19, 24–27, 33–36, 38, 39, 42, 43</td>
<td>lqmm-package</td>
<td></td>
</tr>
<tr>
<td>8, 20, 21, 23, 43</td>
<td>lqmm.fit.df</td>
<td></td>
</tr>
<tr>
<td>8, 20, 21, 25, 43</td>
<td>lqmm.fit.gs</td>
<td></td>
</tr>
<tr>
<td>20, 21, 24, 25</td>
<td>lqmmControl</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>meanAL</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>mleAL</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>model.frame</td>
<td></td>
</tr>
<tr>
<td>13, 15</td>
<td>model.matrix.default</td>
<td></td>
</tr>
<tr>
<td>20, 23</td>
<td>optim</td>
<td></td>
</tr>
<tr>
<td>20, 23</td>
<td>optimize</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>options</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>Orthodont</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>pal (dal)</td>
<td></td>
</tr>
<tr>
<td>14, 31, 37</td>
<td>predict.lqm</td>
<td></td>
</tr>
<tr>
<td>22, 32, 38</td>
<td>predict.lqmm</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>predint (predict.lqmm)</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>print.lqm</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>print.lqmm</td>
<td></td>
</tr>
<tr>
<td>34, 41</td>
<td>print.summary.lqm</td>
<td></td>
</tr>
<tr>
<td>35, 42</td>
<td>print.summary.lqmm</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>qal (dal)</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>ranef (ranef.lqmm)</td>
<td></td>
</tr>
<tr>
<td>33, 36, 38</td>
<td>ranef.lqmm</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>residuals.lqm</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>residuals.lqm.counts</td>
<td></td>
</tr>
<tr>
<td>22, 37</td>
<td>residuals.lqmm</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>summary.boot.lqm</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>summary.boot.lqmm</td>
<td></td>
</tr>
<tr>
<td>5, 14, 35, 40</td>
<td>summary.lqm</td>
<td></td>
</tr>
<tr>
<td>6, 22, 35, 41</td>
<td>summary.lqmm</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>varAL (meanAL)</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>VarCorr (VarCorr.lqmm)</td>
<td></td>
</tr>
<tr>
<td>7, 21, 22, 24, 25, 42</td>
<td>VarCorr.lqmm</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The entries in the index are related to the functions and concepts mentioned in the document. The page numbers indicate the sections where these functions and concepts are discussed.