Package 'mcemGLM'

May 8, 2026

Type Package

Title Maximum Likelihood Estimation for Generalized Linear Mixed Models

Version 1.1.3

Date 2023-3-28

Author Felipe Acosta Archila

Maintainer Felipe Acosta Archila <acosta.felipe@gmail.com>

Description

Maximum likelihood estimation for generalized linear mixed models via Monte Carlo EM. For a description of the algorithm see Brian S. Caffo, Wolfgang Jank and Galin L. Jones (2005) <DOI:10.1111/j.1467-9868.2005.00499.x>.

License GPL (>= 2)

Depends trust, stats

Imports Rcpp (>= 0.11.3)

LinkingTo Rcpp, RcppArmadillo

LazyData true

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-04-01 07:00:06 UTC

Contents

`mcemGLM-package` *Generalized Linear Mixed Model Estimation via Monte Carlo EM*

Description

`mcemGLM` performs maximum likelihood estimation for logistic, Poisson, and negative binomial regression when random effects are present. The package uses an MCEM algorithm to estimate the model's fixed parameters and variance components with their respective standard errors.

A Wald test based `anova` is available to test significance of multi-leveled variables and for multiple contrast testing.

Details

Package:	mcemGLM
Type:	Package
Version:	1.1.2
Date:	2023-01-12
License:	GPL (>= 2)

Author(s)

Felipe Acosta Archila

Maintainer: Felipe Acosta Archila <acosta@umn.edu>

Examples

```
set.seed(123)
x <- rnorm(30, 10, 1)
z <- factor(rep(1:6, each = 5))
obs <- sample(0:1, 30, TRUE)
fit <- mcemGLMM(obs ~ x, random = ~ 0 + z, family = "bernoulli",
vcDist = "normal", controlEM = list(EMit = 15, MCit = 10000),
initial = c(3.30, -0.35, 0.005))
summary(fit)
anova(fit)
```

anova.mcemGLMM *Anova method for mcemGLMM objects*

Description

ANOVA table based on Wald tests for a model fitted with `mcemGLMM`.

Usage

```
## S3 method for class 'mcemGLMM'
anova(object, ...)
```

Arguments

`object`	a model fitted with the mcemGLMM function.
`...`	additional arguments.

Value

A matrix with the rows corresponding to a test for the different terms of the model and the following columns:

Df degrees of freedom for the term.

Wald Wald's chi squared statistic.

Pr p value for the test statistic.

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

coef.mcemGLMM *Anova method for mcemGLMM objects*

Description

Extract the fixed effect coefficients from a model fitted with `mcemGLMM`.

Usage

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

Arguments

`object`	a model fitted with the mcemGLMM function.
`...`	additional arguments.

Value

A vector with the fixed effect coefficients.

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

`contrasts.mcemGLMM` *Contrast estimation for mcemGLMM objects*

Description

Contrast testing for a model fitted with `mcemGLMM`.

Usage

```
contrasts.mcemGLMM(object, ctr.mat)
```

Arguments

`object`	a model fitted with the mcemGLMM function.
`ctr.mat`	contrast matrix. Each row corresponds to a linear combination of the fixed effect coefficients.

Value

A matrix with each row corresponding to a contrast and with the following columns:

Estimate contrasts's point estimator.

Std. Err. standard error for each fitted contrast.

Wald Wald statistic for each fitted contrast.

Adj. p-value p-value adjusted for multiple comparison (Bonferroni.)

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

Examples

```
set.seed(72327)
data(exdata)
fit <- mcemGLMM(obs ~ z2 + x, random = ~ 0 + z1,
                data = exdata,
                family = "bernoulli", vcDist = "normal",
                controlEM = list(verb = FALSE, EMit = 15, MCit = 10000),
                initial = c(-0.13, -0.15, -0.21, 1.59, 0.002))
mat <- rbind("1 - 2" = c(0, -1, 0, 0), "1 - 3" = c(0, 0, -1, 0), "2 - 3" = c(0, 1, -1, 0))
contrasts.mcemGLMM(fit, mat)
```

covMat.mcemGLMM *Anova method for mcemGLMM objects*

Description

Extract the fixed effect's covariance matrix from a model fitted with `mcemGLMM`.

Usage

```
covMat.mcemGLMM(object, ...)
```

Arguments

`object`	a model fitted with the mcemGLMM function.
`...`	additional arguments.

Value

A matrix corresponding to the covariance matrix of the fixed effects.

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

epilepsy *Epilesy Data*

Description

Data from an experiment with $i=1,...,59$ epilepsy patients. Each of the patients was assigned to a control group or a treatment group. The experiment recorded the number of seizures experienced by each patient over four two-week periods. The experiment also recorded a baseline count of the number of seizures the patients had experienced during the previous eight weeks.

Usage

```
data("epilepsy")
```

Format

A data frame with 236 observations on the following 6 variables.

`count` Number of seizures experienced.

`id` Patient ID.

`group` Treatment or control groups.

`age` The logarithm of the patient's age.

`base` The logarithm of baseline/4.

`visit` The corresponding time period.

References

Thall, P. F. and Vail, S. C. (1990) Some covariance models for longitudinal count data with overdispersion In *Biometrics* 46, 657–671

`exdata` *Example data*

Description

Simulated binary response dataset to use in examples.

Usage

```
data("exdata")
```

Format

A data frame with 60 observations on the following 4 variables.

`obs` a numeric binary vector

`obs2` a numeric count vector

`x` a numeric vector

`z1` a factor with levels `1`, and `2`

`z2` a factor with levels `1`, `2`, and `3`

The observations were generated independently with the code shown in the examples section.

Examples

```
set.seed(123)
obs <- c(sample(0:1, 30, TRUE, prob = c(0.5, 0.5)), sample(0:1, 30, TRUE, prob = c(0.3, 0.7)))
obs2 <- c(rpois(30, 5), rpois(30, 10))
```

`mcemGLMM` *Generalized Linear Mixed Models Estimation*

Description

Maximum likelihood estimation for logistic, Poisson, and negative binomial models with random effects using a Monte Carlo EM algorithm.

Usage

```
mcemGLMM(fixed, random, data, family = c("bernoulli", "poisson",
    "negbinom", "gamma"), vcDist = c("normal", "t"), df,
    controlEM = list(), controlTrust = list(), initial)
```

Arguments

`fixed`	the fixed effects model. This is specified by a `formula` object.
`random`	the random effects models. This is specified by a `formula` object or a list of `formula` objects. See details below.
`data`	an optional data frame containing the variables in the model. If missing the variables are taken from the current environment.
`family`	a string indicating the type of model to be fitted. The options are "bernoulli" for logistic regression, "Poisson" for Poisson count regression, and "negbinom" for negative binomial count regression.
`vcDist`	a string indicating the distribution of the marginal variance components. The options are "normal" and "t" for normal and t distributed random effects respectively.
`df`	a vector of degrees of freedom of the random effects when these are t distributed. The length of the vector must be equal to the number of variance components in the model.
`controlEM`	a list of options for the algorithm. See Details below.
`controlTrust`	a list of options to be passed to the `trust` optimizer. See details below.
`initial`	optional initial values for the parameters. If missing the initial values for the fixed effects are taken from a generalized linear model fitted without random effects and the initial values for the variance components are set to 5.

Details

The function `mcemGLMM` allows the fitting of generalized linear mixed models when the random effects are normal or a t distributed. The supported models are the logistic, Poisson and negative binomial. The degrees of freedom for the t case must be supplied by the user with a vector in the `df` argument. The length of the vector must be equal to the number of variance components. For normal random effects the argument `df` does not need to be included.

To fit a model with one random effect a formula must be supplied in the `random` argument. Note that it is necessary that the variable is a factor and to specify that there is no intercept for the random part of the model. To use more than one random effects a list of formulas must be supplied. Each member must be formula with in which the variables involved must be factors and it also it is necessary to specify that there is no intercept. To fit crossed random effects each variable must no appear in its own formula. To fit nested random effects a formula with the highest level variable must be specified and each subsequent variable must be specified with an interaction of the variables above it. See examples below.

A note on the negative binomial overdispersion:

The variance for the negative binomial model is set equal to $(1 + \mu/\alpha)$ so we have that there is no overdispersion as α goes to infinity.

The variance for the gamma distribution is set to μ^2/α. The value of $\alpha = 1$ corresponds to an exponential regression.

Stopping rules and convergence criteria:

The algorithm runs for a maximum of `EMit` iterations or until the criteria

$$\max_i \left\{ \frac{|\theta_i^{(t)} - \theta_i^{(t-1)}|}{|\theta_i^{(t)}| + \delta} \right\} < \epsilon$$

is satisfied three times in a row for pre-defined values of ϵ and δ. Once this criterion has been achieved two times we increase the Monte Carlo sample size more rapidly to have a better estimation of the model's information matrix. For a detailed discussion on convergence diagnostics see Neath R.C. (2012). After fitting a model it is recommended to plot the EM estimates at each step to assess convergence.

Control options for EM:

EMit maximum number of EM iterations.

MCit initial number of Monte Carlo iterations for the MCMC step.

MCf factor in which the MC iterations increase in each EM iteration.

verb logical. If TRUE at each EM iteration the function will print convergence information and a trace plot for on of the random effects. This is useful to assess the performance of the algorithm but it can impact the actual running time.

MCsd initial standard deviation for the proposal density of the MCMC step. If zero (default) an auto-tuning step will be performed.

EMdelta constant for the EM error assessment.

EMepsilon constant for the EM error assessment.

Control options for trust, see `help(trust)` for more details:

rinit starting trust region radius. Default value set to 20.

rmax maximum allowed trust region radius. Default value set to 200.

iterlim maximum number of iterations. Default value set to 100.

Value

A list of class "mcemGLMM" with the following items:

mcemEST a matrix with the value of the maximum likelihood estimators at the end of each EM step.

iMatrix Fisher's information matrix.

QfunVal value (up to a constant) of the Q function. Used to perform likelihood ratio tests.

QfunMCMC Q function MCMC sample.

randeff a sample from the conditional distribution of the random effects given the data and the maximum likelihood estimators.

y vector of observations.

x design matrix for the fixed effects.

z design matrix for the random effects.

EMerror relative error at the last iteration. See details.

MCsd last value for MCMC step size.

call original call.

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

References

Neath, R. C. (2012) On Convergence Properties of the Monte Carlo EM Algorithm In Advances in Modern Statistical Theory and Applications: A Festschrift in Honor of Morris L. Eaton. *Institute of Mathematical Statistics* 43–62

Examples

```
# Data set for a logistic model with one binary fixed effects and two
# possible random effects.
# Initial values and MC iterations are given to speed up the examples
# but these are not necessary in general.
set.seed(0123210)
data(exdata)

# To fit a model with one random effect
fit.1 <- mcemGLMM(obs ~ x, random = ~ 0 + z1, data = exdata,
                family = "bernoulli", vcDist = "normal",
                controlEM = list(MCit = 10000),
                initial = c(0.27, -0.13, 0.003))
summary(fit.1)

# We can assess convergence by looking at a trace plot of the EM estimates
# and the loglikelihood values
ts.plot(fit.1$mcemEST)
ts.plot(fit.1$QfunVal)

# To fit a model with crossed random effects
fit.crossed <- mcemGLMM(obs ~ x, random = list(~ 0 + z1, ~ 0 + z2),
                data = exdata,
                family = "bernoulli", vcDist = "normal",
                controlEM = list(EMit = 10, MCit = 10000),
                initial = c(0.28, -0.15, 0.001, 0.4))
summary(fit.crossed)


# To fit a model with crossed random effects
fit.nested <- mcemGLMM(obs ~ x, random = list(~ 0 + z2, ~ 0 + z2:z1),
                data = exdata,
                family = "bernoulli", vcDist = "normal",
                controlEM = list(EMit = 10, MCit = 10000),
                initial = c(0.31, -0.15, 0.29, 0.27))
summary(fit.nested)

# Fit a Poisson model
fit.pois <- mcemGLMM(obs2 ~ x, random = ~ 0 + z1, data = exdata,
                family = "poisson", vcDist = "normal",
                controlEM = list(EMit = 10, MCit = 10000),
                initial = c(1.95, 0.03, 0.003))
```

```
summary(fit.pois)
```

`mcemGLMMext` *Extending the Iterations of a Model Fitted with mcemGLMM*

Description

Given a model fitted with the function `mcemGLMM` this function will add iterations and update the model estimates for more accurate results.

This is recommended if the initial fitting seems to have a large Monte Carlo error. This function will use the previous maximum likelihood estimate as its initial point and will also start with a Monte Carlo sample size equal to the sample size used in the last iteration of the previous fitting.

Usage

```
mcemGLMMext(object, it = 20, controlEM)
```

Arguments

`object`	an model fitted with `mcemGLMM`
`it`	the maximum number of iterations to be performed.
`controlEM`	a list. New set of options for the EM algorithm. Can be missing

Value

An updated object of class `mcemGLMM`.

Note

If `controlEM` is supplied it is important that the value for `MCit` is at least equal to number of Monte Carlo iterations used in the last EM step to fit `object` since providing a lower number will increase the Monte Carlo error.

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

See Also

`mcemGLMM`

Examples

```
set.seed(72327)
data(exdata)
fit1 <- mcemGLMM(obs ~ z2 + x, random = ~ 0 + z1,
                 data = exdata,
                 family = "bernoulli", vcDist = "normal",
                 controlEM = list(verb = FALSE, EMit = 5, MCit = 8000),
                 initial = c(-0.13, -0.15, -0.21, 1.59, 0.002))

# Now we extend the algorithm to do at least another 10 iterations
fit2 <- mcemGLMMext(fit1, it = 10)
```

`predict.mcemGLMM` *Predict method for mcemGLMM objects*

Description

This functions returns predicted link function of observations for a model fitted with `mcemGLMM`.

Usage

```
  ## S3 method for class 'mcemGLMM'
predict(object, newdata, type = c("link", "response"), se.fit = FALSE, ...)
```

Arguments

`object`	a model fitted with the mcemGLMM function.
`newdata`	optional data frame with new data. The variable names must match the original variables. If missing, the function will return predicted values at each observation.
`type`	character string. The type of predictions to be returned. Either "link" or "response" predictions are available.
`se.fit`	logical. If true, standard errors will be returned.
`...`	additional arguments.

Value

A vector with the predictions from the observed data or by using the supplied new data.

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

ranef.mcemGLMM *Anova method for mcemGLMM objects*

Description

Extract the random effect predictions from a model fitted with mcemGLMM.

Usage

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

Arguments

object	a model fitted with the mcemGLMM function.
...	additional arguments.

Value

A vector with the random effect predictions.

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

residuals.mcemGLMM *Residual extraction method for mcemGLMM objects*

Description

This functions returns the residuals of a model fitted with mcemGLMM.

Usage

```
## S3 method for class 'mcemGLMM'
residuals(object, type = c("deviance", "pearson"), ...)
```

Arguments

object	a model fitted with the mcemGLMM function.
type	character string. The type of residuals to be returned.
...	additional arguments.

Value

A vector with the residuals of the model.

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

`salamander` *Salamander Data*

Description

Salamander mating experiment data from McCullagh and Nelder.

Usage

```
data("salamander")
```

Format

A data frame with 60 observations on the following 4 variables.

`Cross` the type of salamanders involved in the crossing.

`Female` the female salamander's ID.

`Male` the male salamander's ID.

`Mate` whether the crossing was successful or not.

References

McCullagh, P and Nelder J. A. (1989) Generalized Linear Models, Second Edition. Chapman and Hall, 1989

`simData` *Data used for fitting examples*

Description

Example data for logistic, Poisson and negative binomial models.

Usage

```
data("simData")
```

Format

A data frame with 200 observations on the following 8 variables.

`obs` a binary vector. Used as a response for a logistic model.

`x1` a numeric vector.

`x2` a numeric vector.

`x3` a categorical vector with levels `blue`, `red`, and `yellow`.

`z1` a categorical vector with levels `D1`, `D2`, `D3`, `D4`, and `D5`.

`z2` a categorical vector with levels `1`, `2`, `3`, `4`, and `5`.

`z3` a categorical vector with levels `A`, `B`, and `D`.

`count` a numeric vector. Used as a response for a Poisson model.

Details

The levels of `z2` can be nested within $z1$. The observations were generated with the code shown in the examples section.

Examples

```
set.seed(47819)
x1 <- rnorm(200, 10, 1)
x2 <- rnorm(200, 5, 1)
x3 <- sample(c("red", "blue", "yellow"), size = 200, replace = TRUE)
z1 <- factor(rep(c("D1", "D2", "D3", "D4", "D5"), each = 40))
z2 <- factor(rep(rep(1:4, each = 5), 10))
z3 <- factor(c(rep("A", 100), rep("B", 60), rep("D", 40)))
kX <- model.matrix(~x1 + x2 + x3)
kZ <- cbind(model.matrix(~ 0+z1), model.matrix(~ 0+z1:z2), model.matrix(~ 0+z3))
kBeta <- c(5, -4, 5, 0, 8)
kU <- 3 * rt(28, 5)
linf0 <- kX
prob0 <- exp(linf0)/(1+exp(linf0))
obs <- as.numeric(runif(100) < prob0)
simData <- data.frame(obs, x1, x2, x3, z1, z2, z3)
```

`summary.mcemGLMM` *Summary method for mcemGLMM objects*

Description

Summary for an object obtained from `mcemGLMM`.

Usage

```
## S3 method for class 'mcemGLMM'
summary(object, ...)
```

Arguments

`object`	a model fitted with the mcemGLMM function.
`...`	additional arguments.

Details

The function prints a table for Wald tests for the fixed effect coefficients and the variance estimators. For the negative binomial and the gamma distributions the estimate of α is reported with its respective standard error.

Value

A list with the following items:

coefficients a list with the fixed effects coefficients and the predicted random effects.

var.est the estimated variances for each variance component.

std.err the standard errors for the fixed effects coefficients and the variance estimates.

z.val z test values for the fixed effects coefficients and the variance estimators.

Author(s)

Felipe Acosta Archila <acosta@umn.edu>

Index