Package ‘mumm’

August 15, 2018

Type Package

Title Multiplicative Mixed Models using the Template Model Builder

Version 0.2.1

Date 2018-08-14

Maintainer Sofie Poedenphant <sofp@dtu.dk>

Description Fit multiplicative mixed models using maximum likelihood estimation via the Template Model Builder (TMB), Kristensen K, Nielsen A, Berg CW, Skaug H, Bell BM (2016) <doi:10.18637/jss.v070.i05>.
One version of the multiplicative mixed model is applied in Piepho (1999) <doi:10.1111/j.0006-341X.1999.01120.x>.
The package provides functions for calculating confidence intervals for the model parameters and for performing likelihood ratio tests.

License GPL (>= 2)

Imports TMB, Rcpp, Matrix, stringr, methods

Depends lme4

LinkingTo TMB, RcppEigen, Rcpp

LazyData TRUE

RoxygenNote 6.1.0

URL http://github.com/sofpj/mumm

BugReports http://github.com/sofpj/mumm/issues

Encoding UTF-8

Suggests knitr

Repository CRAN

NeedsCompilation yes

Author Sofie Poedenphant [aut, cre],
Per Bruun Brockhoff [aut]

Date/Publication 2018-08-15 11:00:03 UTC

R topics documented:

---

confint.mumm *Confidence Intervals for Model Parameters*

Description

Computes confidence intervals for the fixed effect parameters and the variance components for an object of class mumm.

Usage

```
## S3 method for class 'mumm'
confint(object, parm = "all", level = 0.95, ...)
```

Arguments

object	an object of class mumm.
parm	a vector of parameter names or a matrix, where the rows specify linear combinations of the model parameters. If missing, confidence intervals will be computed for all of the fixed effect parameters and all of the variance components.
level	the confidence level.
...	Currently not used.

Details

The confidence intervals are computed by the profile likelihood method.

Value

A matrix with the first column showing the lower confidence limit and the second column showing the upper limit for each parameter or linear combination of parameters.

Examples

```
set.seed(100)
sigma_e <- 1.5
sigma_a <- 0.8
sigma_b <- 0.5
sigma_d <- 0.7
nu <- c(8.2, 6.2, 2.3, 10.4, 7.5, 1.9)
```

```
nA <- 15
nP <- 6
nR <- 5

a <- rnorm(nA, mean = 0, sd = sigma_a)
b <- rnorm(nA, mean = 0, sd = sigma_b)
d <- rnorm(nA*nP, mean = 0, sd = sigma_d)
e <- rnorm(nA*nP*nR, mean = 0, sd = sigma_e)

Assessor <- factor(rep(seq(1,nA),each = (nP*nR)))
Product <- factor(rep(rep(seq(1,nP),each = nR), nA))
AssessorProduct <- (Assessor:Product)

y <- nu[Product] + a[Assessor] + b[Assessor]*(nu[Product]-mean(nu)) + d[AssessorProduct] + e

sim_data <- data.frame(y, Assessor, Product)

fit <- mumm(y ~ 1 + Product + (1|Assessor) + (1|Assessor:Product) +
             mp(Assessor,Product) ,data = sim_data)

confint(fit, parm = c('Product3', 'mp Assessor:Product'), level = 0.90)
```

lrt *Likelihood Ratio Test*

Description

A function to perform a likelihood ratio test for testing two nested models against each other.

Usage

```
lrt(fit1, fit2)
```

Arguments

fit1	a fitted model object of class mumm.
fit2	a fitted model object of class mumm, lm or merMod.

Details

Performs the likelihood ratio test for testing two nested models against each other. The model in `fit2` should be nested within the model in `fit1`.

Value

A matrix with the likelihood ratio test statistic and the corresponding p-value.

Examples

```
set.seed(100)
sigma_e <- 1.5
sigma_a <- 0.8
sigma_b <- 0.5
sigma_d <- 0.7
nu <- c(8.2, 6.2, 2.3, 10.4, 7.5, 1.9)

nA <- 15
nP <- 6
nR <- 5

a <- rnorm(nA, mean = 0, sd = sigma_a)
b <- rnorm(nA, mean = 0, sd = sigma_b)
d <- rnorm(nA*nP, mean = 0, sd = sigma_d)
e <- rnorm(nA*nP*nR, mean = 0, sd = sigma_e)

Assessor <- factor(rep(seq(1,nA),each = (nP*nR)))
Product <- factor(rep(rep(seq(1,nP),each = nR), nA))
AssessorProduct <- (Assessor:Product)

y <- nu[Product] + a[Assessor] + b[Assessor]*(nu[Product]-mean(nu)) + d[AssessorProduct] + e

sim_data <- data.frame(y, Assessor, Product)

fit <- mumm(y ~ 1 + Product + (1|Assessor) + (1|Assessor:Product) +
             mp(Assessor,Product) ,data = sim_data)

fit2 <- mumm(y ~ 1 + Product + (1|Assessor) + mp(Assessor,Product) ,data = sim_data)
lrt(fit,fit2)
```

mumm *Fit Multiplicative Mixed Models with TMB*

Description

Fit a multiplicative mixed-effects model to data with use of the Template Model Builder.

Usage

```
mumm(formula, data, cor = TRUE, start = c(), control = list())
```

Arguments

`formula` a two-sided formula object describing the linear fixed-effects and random-effects part together with the multiplicative part. The response is on the left of a ~ operator and the terms which are separated by + operators are on the right. The random-effect terms are recognized by vertical bars "|", separating an expression for a model matrix and a grouping factor. The syntax for the multiplicative term is 'mp("random effect","fixed effect")'.

`data` a data frame containing the variables in the formula.

`cor` logical. If FALSE the random effect in the multiplicative term is assumed to be independent of the corresponding random main effect.

`start` a numeric vector of starting values for the parameters in the model.

`control` a list of control parameters passed on to the `nlminb` function used for the optimization.

Details

Fit a multiplicative mixed model via maximum likelihood with use of the Template Model Builder. A multiplicative mixed model is here considered as a model with a linear mixed model part and one multiplicative term. A multiplicative term is here defined as a product of a random effect and a fixed effect, i.e. a term that models a part of the interaction as a random coefficient model based on linear regression on a fixed main effect.

Value

An object of class `mumm`.

Examples

```
set.seed(100)
sigma_e <- 1.5
sigma_a <- 0.8
sigma_b <- 0.5
sigma_d <- 0.7
nu <- c(8.2, 6.2, 2.3, 10.4, 7.5, 1.9)
nA <- 15
nP <- 6
nR <- 5
a <- rnorm(nA, mean = 0, sd = sigma_a)
b <- rnorm(nA, mean = 0, sd = sigma_b)
d <- rnorm(nA*nP, mean = 0, sd = sigma_d)
e <- rnorm(nA*nP*nR, mean = 0, sd = sigma_e)
Assessor <- factor(rep(seq(1,nA),each = (nP*nR)))
Product <- factor(rep(rep(seq(1,nP),each = nR), nA))
AssessorProduct <- (Assessor:Product)
y <- nu[Product] + a[Assessor] + b[Assessor]*(nu[Product]-mean(nu)) + d[AssessorProduct] + e
sim_data <- data.frame(y, Assessor, Product)
fit <- mumm(y ~ 1 + Product + (1|Assessor) + (1|Assessor:Product) +
             mp(Assessor,Product) ,data = sim_data)
```

ranef.mumm *Extract Random Effects*

Description

A function to extract the estimated random effects from a model object of class mumm.

Usage

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

Arguments

object	an object of class "mumm"
...	Currently not used

Value

A named list with the estimated random effects, where each element in the list is a numeric vector consisting of the estimated random effect coefficients for a random factor in the model.

Examples

```
set.seed(100)
sigma_e <- 1.5
sigma_a <- 0.8
sigma_b <- 0.5
sigma_d <- 0.7
nu <- c(8.2, 6.2, 2.3, 10.4, 7.5, 1.9)

nA <- 15
nP <- 6
nR <- 5

a <- rnorm(nA, mean = 0, sd = sigma_a)
b <- rnorm(nA, mean = 0, sd = sigma_b)
d <- rnorm(nA*nP, mean = 0, sd = sigma_d)
e <- rnorm(nA*nP*nR, mean = 0, sd = sigma_e)

Assessor <- factor(rep(seq(1,nA),each = (nP*nR)))
Product <- factor(rep(rep(seq(1,nP),each = nR), nA))
AssessorProduct <- (Assessor:Product)

y <- nu[Product] + a[Assessor] + b[Assessor]*(nu[Product]-mean(nu)) + d[AssessorProduct] + e

sim_data <- data.frame(y, Assessor, Product)

fit <- mumm(y ~ 1 + Product + (1|Assessor) + (1|Assessor:Product) +
                mp(Assessor,Product) ,data = sim_data)

ranef(fit)
```

Index