Package 'Rdta'

October 12, 2022

Version 1.0.0

Date 2020-1-7

Title Data Transforming Augmentation for Linear Mixed Models

Author Hyungsuk Tak, Kisung You, Sujit K. Ghosh, and Bingyue Su

Maintainer Hyungsuk Tak <hyungsuk.tak@gmail.com>

Depends R (>= 2.2.0)

Imports MCMCpack(>= 1.4-4), mvtnorm(>= 1.0-11), Rdpack, stats

Description We provide a toolbox to fit univariate and multivariate linear mixed models via data transforming augmentation. Users can also fit these models via typical data augmentation for a comparison. It returns either maximum likelihood estimates of unknown model parameters (hyperparameters) via an EM algorithm or posterior samples of those parameters via a Markov chain Monte Carlo method. Also see Tak, You, Ghosh, Su, and Kelly (2019+) <doi:10.1080/10618600.201

License GPL-2

Encoding UTF-8

RdMacros Rdpack

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2020-01-24 16:50:03 UTC

R topics documented:

`lmm` *Fitting univariate and multiviarate linear mixed models via data transforming augmentation*

Description

The function `lmm` fits univariate and multivariate linear mixed models (also called two-level Gaussian hierarchical models) whose first-level hierarchy is about a distribution of observed data and second-level hierarchy is about a prior distribution of random effects.

Usage

```
lmm(y, v, x = 1, n.burn, n.sample, tol = 1e-10,
  method = "em", dta = TRUE, print.time = FALSE)
```

Arguments

- `y` — Response variable. In a univariate case, it is a vector of length k for the observed data. In a multivariate case, it is a (k by p) matrix, where k is the number of observations and p denotes the dimensionality.
- `v` — Known measurement error variance. In a univariate case, it is a vector of length k. In a multivariate case, it is a (p, p, k) array of known measurement error covariance matrices, i.e., each of the k array components is a (p by p) covariance matrix.
- `x` — (Optional) Covariate information. If there is one covariate for each object, e.g., weight, it is a vector of length k for the weight. If there are two covariates for each object, e.g., weight and height, it is a (k by 2) matrix, where each column contains a covariate variable. Default is no covariate (`x = 1`).
- `n.burn` — Number of warming-up iterations for a Markov chain Monte Carlo method. It must be specified for `method = "mcmc"`
- `n.sample` — Number of iterations (size of a posterior sample for each parameter) for a Markov chain Monte Carlo method. It must be specified for `method = "mcmc"`
- `tol` — Tolerance that determines the stopping rule of the EM algorithm. The EM algorithm iterates until the change of log-likelihood function is within the tolerance. Default is 1e-10.
- `method` — `"em"` will return maximum likelihood estimates of the unknown hyper-parameters and `"mcmc"` returns posterior samples of those parameters.
- `dta` — A logical; Data transforming augmentation is used if `dta = TRUE`, and typical data augmentation is used if `dta = FALSE`.
- `print.time` — A logical; `TRUE` to display two time stamps for initiation and termination, `FALSE` otherwise.

Details

For each group i, let y_i be an unbiased estimate of random effect θ_i, and V_i be a known measurement error variance. The linear mixed model of interest is specified as follows:

$$[y_i \mid \theta_i] \sim N(\theta_i, V_i)$$

$$[\theta_i \mid \mu_{0i}, A) \sim N(\mu_{0i}, A)$$

$$\mu_{0i} = x_i'\beta$$

independently for $i = 1, \ldots, k$, where k is the number of groups (units) and dimension of each element is appropriately adjusted in a multivariate case.

The function `lmm` produces maximum likelihood estimates of hyper-parameters, A and β, their update histories of EM iterations, and the number of EM iterations if `method` is `"em"`.

For a Bayesian implementation, we put a jointly uniform prior distribution on A and β, i.e.,

$$f(A, \beta) \propto 1,$$

which is known to have good frequency properties. This joint prior distribution is improper, but their resulting posterior distribution is proper if $k \geq m + p + 2$, where k is the number of groups, m is the number of regression coefficients, and p is the dimension of y_i. We note that an R package `Rgbp` also fits this model in a univariate case ($p = 1$) via ADM (approximation for density maximization). `lmm` produces the posterior samples through a Gibbs sampler if `method` is `"bayes"`.

Value

The outcome of `lmm` is composed of:

A If `method` is "mcmc". It contains the posterior sample of A.

beta If `method` is "mcmc". It contains the posterior sample of β.

A.mle If `method` is "em". It contains the maximum likelihood estimate of A.

beta.mle If `method` is "em". It contains the maximum likelihood estimate of $beta$.

A.trace If `method` is "em". It contains the update history of A at each iteration.

beta.trace If `method` is "em". It contains the update history of $beta$ at each iteration.

n.iter If `method` is "em". It contains the number of EM iterations.

Author(s)

Hyungsuk Tak (maintainer), Kisung You, Sujit K. Ghosh, and Bingyue Su

References

Tak H, You K, Ghosh SK, Su B, Kelly J (2019). "Data Transforming Augmentation for Heteroscedastic Models." *arXiv:1911.02748 [stat]*. doi: 10.1080/10618600.2019.1704295, arXiv: 1911.02748, accepted for a publication in Journal of Computational and Graphical Statistics, `http://arxiv.org/abs/1911.02748`.

Examples

```
### Univariate linear mixed model

# response variable for 10 objects
y <- c(5.42, -1.91, 2.82, -0.14, -1.83, 3.44, 6.18, -1.20, 2.68, 1.12)
# corresponding measurement error standard deviations
se <- c(1.05, 1.15, 1.22, 1.45, 1.30, 1.29, 1.31, 1.10, 1.23, 1.11)
# one covariate information for 10 objects
x <- c(2, 3, 0, 2, 3, 0, 1, 1, 0, 0)

## Fitting without covariate information
# (DTA) maximum likelihood estimates of A and beta via an EM algorithm
res <- lmm(y = y, v = se^2, method = "em", dta = TRUE)
# (DTA) posterior samples of A and beta via an MCMC method
res <- lmm(y = y, v = se^2, n.burn = 1e1, n.sample = 1e1,
           method = "mcmc", dta = TRUE)
# (DA) maximum likelihood estimates of A and beta via an EM algorithm
res <- lmm(y = y, v = se^2, method = "em", dta = FALSE)
# (DA) posterior samples of A and beta via an MCMC method
res <- lmm(y = y, v = se^2, n.burn = 1e1, n.sample = 1e1,
           method = "mcmc", dta = FALSE)

## Fitting with the covariate information
# (DTA) maximum likelihood estimates of A and beta via an EM algorithm
res <- lmm(y = y, v = se^2, x = x, method = "em", dta = TRUE)
# (DTA) posterior samples of A and beta via an MCMC method
res <- lmm(y = y, v = se^2, x = x, n.burn = 1e1, n.sample = 1e1,
           method = "mcmc", dta = TRUE)
# (DA) maximum likelihood estimates of A and beta via an EM algorithm
res <- lmm(y = y, v = se^2, x = x, method = "em", dta = FALSE)
# (DA) posterior samples of A and beta via an MCMC method
res <- lmm(y = y, v = se^2, x = x, n.burn = 1e1, n.sample = 1e1,
           method = "mcmc", dta = FALSE)


### Multivariate linear mixed model

# (arbitrary) 10 hospital profiling data (two response variables)
y1 <- c(10.19, 11.53, 16.28, 12.32, 12.84, 11.85, 14.81, 13.24, 14.43, 9.35)
y2 <- c(12.06, 14.97, 11.50, 17.88, 19.21, 14.69, 13.96, 11.07, 12.71, 9.63)
y <- cbind(y1, y2)

# making measurement error covariance matrices for 10 hospitals
n <- c(24, 34, 38, 42, 49, 50, 79, 84, 96, 102) # number of patients
v0 <- matrix(c(186.87, 120.43, 120.43, 250.60), nrow = 2) # common cov matrix
temp <- sapply(1 : length(n), function(j) { v0 / n[j] })
v <- array(temp, dim = c(2, 2, length(n)))

# covariate information (severity measure)
severity <- c(0.45, 0.67, 0.46, 0.56, 0.86, 0.24, 0.34, 0.58, 0.35, 0.17)

## Fitting without covariate information
```

```
# (DTA) maximum likelihood estimates of A and beta via an EM algorithm

res <- lmm(y = y, v = v, method = "em", dta = TRUE)

# (DTA) posterior samples of A and beta via an MCMC method

res <- lmm(y = y, v = v, n.burn = 1e1, n.sample = 1e1,
           method = "mcmc", dta = TRUE)

# (DA) maximum likelihood estimates of A and beta via an EM algorithm

res <- lmm(y = y, v = v, method = "em", dta = FALSE)

# (DA) posterior samples of A and beta via an MCMC method

res <- lmm(y = y, v = v, n.burn = 1e1, n.sample = 1e1,
           method = "mcmc", dta = FALSE)


## Fitting with the covariate information
# (DTA) maximum likelihood estimates of A and beta via an EM algorithm

res <- lmm(y = y, v = v, x = severity, method = "em", dta = TRUE)

# (DTA) posterior samples of A and beta via an MCMC method

res <- lmm(y = y, v = v, x = severity, n.burn = 1e1, n.sample = 1e1,
           method = "mcmc", dta = TRUE)

# (DA) maximum likelihood estimates of A and beta via an EM algorithm

res <- lmm(y = y, v = v, x = severity, method = "em", dta = FALSE)

# (DA) posterior samples of A and beta via an MCMC method

res <- lmm(y = y, v = v, x = severity, n.burn = 1e1, n.sample = 1e1,
           method = "mcmc", dta = FALSE)
```

Rdta *Data Transforming Augmentation for Linear Mixed Models*

Description

The R package **Rdta** provides a toolbox to fit univariate and multivariate linear mixed models via data transforming augmentation. Users can also fit these models via typical data augmentation for a comparison. It returns either maximum likelihood estimates of unknown model parameters (hyper-parameters) via an EM algorithm or posterior samples of those parameters via a Markov chain Monte Carlo method.

Details

Package:	Rdta
Type:	Package
Version:	1.0.0
Date:	2020-1-7
License:	GPL-2
Main functions:	`lmm`

Author(s)

Hyungsuk Tak (maintainer), Kisung You, Sujit K. Ghosh, and Bingyue Su

References

Tak H, You K, Ghosh SK, Su B, Kelly J (2019). "Data Transforming Augmentation for Heteroscedastic Models." *arXiv:1911.02748 [stat]*. doi: 10.1080/10618600.2019.1704295, arXiv: 1911.02748, accepted for a publication in Journal of Computational and Graphical Statistics, `http://arxiv.org/abs/1911.02748`.

Index