Package ‘blme’

August 29, 2016

Version 1.0-4

Date 2015-06-13

Title Bayesian Linear Mixed-Effects Models

Author Vincent Dorie <vjd@nyu.edu>

Maintainer Vincent Dorie <vjd@nyu.edu>

Description Maximum a posteriori estimation for linear and generalized
linear mixed-effects models in a Bayesian setting. Extends
‘lme4’ by Douglas Bates, Martin Maechler, Ben Bolker, and Steve Walker.

Depends R (>= 3.0-0), lme4 (>= 1.0-6)

Imports methods, stats, utils

Suggests testthat (>= 0.9-0)

License GPL (>= 2)

URL https://github.com/vdorie/blme

BugReports https://github.com/vdorie/blme/issues

NeedsCompilation no

Repository CRAN

Date/Publication 2015-06-14 01:21:14

R topics documented:

   blme ................................................................. 2
   bmerDist-class ..................................................... 4
   bmerMod-class ..................................................... 7

Index 8
Fit Bayesian Linear and Generalized Linear Mixed-Effects Models

Description

Maximum a posteriori estimation for linear and generalized linear mixed-effects models in a Bayesian setting. Built off of `lmer`.

Usage

```r
blmer(formula, data, REML = TRUE,
      control = lmerControl(), start = NULL, verbose = 0L,
      subset, weights, na.action, offset, contrasts = NULL,
      devFunOnly = FALSE, cov.prior = wishart,
      fixef.prior = NULL, resid.prior = NULL, ...)

bglmer(formula, data, family = gaussian,
       control = glmerControl(), start = NULL, verbose = 0L,
       maxit = 100L,
       nAGQ = 1L, subset, weights, na.action, offset,
       contrasts = NULL, mustart, etastart,
       devFunOnly = FALSE, cov.prior = wishart,
       fixef.prior = NULL, ...)
```

Arguments

- `cov.prior`: a BLME prior or list of priors with allowable distributions: wishart, invwishart, gamma, invgamma, or NULL. Imposes a prior over the covariance of the random effects/modeled coefficients. Default is wishart. The NULL argument imposes flat priors over all relevant parameters.

- `fixef.prior`: a BLME prior of family normal, t, or NULL. Imposes a prior over the fixed effects/modeled coefficients. Default is NULL.

- `resid.prior`: a BLME prior of family gamma, invgamma, point or NULL. Imposes a prior over the noise/residual variance, also known as common scale parameter or the conditional variance given the random effects. Default is NULL.

- `start`: like the start arguments for `lmer` and `glmer` a numeric vector or named list. Unlike the aforementioned, list members of fixef and sigma are applicable to linear mixed models provided that numeric optimization is required for these parameters.

- `formula, data, REML, family, control, verbose, maxit, nAGQ, mustart, etastart, devFunOnly, ...`: model specification arguments as in `lmer` and `glmer`; see there for details.

- `subset, weights, na.action, offset, contrasts`: further model specification arguments as in `lm`; see there for details.
Details
The bulk of the usage for `blmer` and `bglmer` closely follows the functions `lmer` and `glmer`. Those help pages provide a good overview of fitting linear and generalized linear mixed models. The primary distinction is that `blmer` and `bglmer` allow the user to do Bayesian inference or penalized maximum likelihood, with priors imposed on the different model components. For the specifics of any distribution listed below, see the `distributions` page.

Covariance Prior
The `cov.prior` argument applies a prior over the covariance matrix of the random effects/modelled coefficients. As there is one covariance matrix for every named grouping factor - that is every element that appears to the right of a vertical bar ("|") in the model formula - it is possible to apply as many different priors as there are said factors.

The general formats of an argument to `blmer` or `bglmer` for such a prior are of the form:

- `cov.prior = factor.name ~ covariance.distribution(option1 = value1, ...)`
- `cov.prior = list(fc.nm ~ dist1, fc.nm ~ dist2, ..., default.distribution)`

If the “factor.name ~” construct is ommitted, the prior is interpreted as a default and applied to all factors that lack specific priors of their own. Options are not required, but permit fine-tuning of the model.

Supported distributions are `gamma`, `invgamma`, `wishart`, `invwishart`, `null`, and `custom`.

The `common.scale` option, a logical, determines whether or not the prior applies to in the absolute-real world sense (value = FALSE), or if the prior is applied to the random effect covariance divided by the estimated residual variance (TRUE). As a practical matter, when false computation can be slower as the profiled common scale may no longer have a closed-form solution. As such, the default for all cases is TRUE.

Other options are specified along with the specific distributions and defaults are explained in the `blme distributions` page.

Fixed Effects Prior
Priors on the fixed effects, or unmodeled coefficients, are specified in a fashion similar to that of covariance priors. The general format is

- `fixef.prior = multivariate.distribution(options1 = value1, ...)`

At present, the implemented multivariate distributions are `normal`, `t`, and `null`. `t` priors cannot be used when REML is TRUE, as that integral does not have a closed form solution.

Residual Variance Prior
The general format for a residual variance prior is the same as for a fixed effect prior. The supported distributions are `point`, `gamma`, `invgamma`.

Value
An object of class "bmerMod", for which many methods are available. See there for details.

See Also
`lmer`, `glmer`, `merMod` class, and `lm`. 
Examples

```r
## covariance prior
data("sleepstudy", package = "lme4")
(fm1 <- blmer(Reaction ~ Days + (0 + Days|Subject), sleepstudy,  
    cov.prior = gamma))
(fm2 <- blmer(Reaction ~ Days + (0 + Days|Subject), sleepstudy,  
    cov.prior = gamma(shape = 2, rate = 0.5, posterior.scale = "sd")))
(fm3 <- blmer(Reaction ~ Days + (1 + Days|Subject), sleepstudy,  
    cov.prior = wishart))
(fm4 <- blmer(Reaction ~ Days + (1 + Days|Subject), sleepstudy,  
    cov.prior = invwishart(df = 5, scale = diag(0.5, 2))))

## custom prior
penaltyFn <- function(sigma)
    dcauchy(sigma, 0, 10, log = TRUE)
(fm5 <- blmer(Reaction ~ Days + (0 + Days|Subject), sleepstudy,  
    cov.prior = custom(penaltyFn, chol = TRUE, scale = "log")))

## fixed effect prior
(fm6 <- blmer(Reaction ~ Days + (1 + Days|Subject), sleepstudy,  
    cov.prior = NULL,  
    fixef.prior = normal))
(fm7 <- blmer(Reaction ~ Days + (1 + Days|Subject), sleepstudy,  
    cov.prior = NULL,  
    fixef.prior = normal(cov = diag(0.5, 2), common.scale = FALSE)))

## residual variance prior
## eight schools example
y <- c(28, 8, -3, 7, -1, 1, 18, 12);  
sigma <- c(15, 10, 16, 11, 9, 11, 10, 18);  
g <- 1:8;
(schools <- blmer(y ~ 1 + (1 | g), resid.prior = point,  
    cov.prior = NULL, REML = FALSE,  
    weights = 1 / sigma^2));
```

### bmerDist-class

**Bayesian Linear Mixed-Effects Model Prior Representations and bmer*Dist Methods**

#### Description

Objects created in the initialization step of a `blme` model that represent the type of prior being applied.

#### Objects from the Class

Objects can be created by calls of the form `new("bmerPrior", ...)` or, more commonly, as side effects of the `blmer` and `bglmer` functions.
When using the main `blme` functions, the prior-related arguments can be passed what essentially are function calls with the distinction that they are delayed in evaluation until information about the model is available. At that time, the functions are defined in a special environment and then evaluated in an environment that directly inherits form the one in which `blmer` or `bglmer` was called. This is reflected in some of the prototypes of various prior-creating functions which depend on parameters not available in the top-level environment.

Finally, if the trailing parentheses are omitted from a `blmer/bglmer` prior argument, they are simply added as a form of "syntactic sugar".

**Prior Distributions**

This section lists the prototypes for the functions that are called to parse a prior during a model fit.

**Fixed Effect Priors**

- `normal(sd = c(10, 2.5), cov, common.scale = TRUE)`
  Applies a Gaussian prior to the fixed effects. Normal priors are constrained to have a mean of 0 - non-zero priors are equivalent to shifting covariates.
  The covariance hyperparameter can be specified either as a vector of standard deviations, using the `sd` argument, a vector of variances using the `cov` argument, or the entire variance/covariance matrix itself. When specifying standard deviations, a vector of length less than the number of fixed effects will have its tail repeated, while the first element is assumed to apply only to the intercept term. So in the default of `c(10, 2.5)`, the intercept receives a standard deviation of 10 and the various slopes are all given a standard deviation of 2.5.
  The `common.scale` argument specifies whether or not the prior is to be interpreted as being on the same scale as the residuals. To specify a prior in an absolute sense, set to `FALSE`. Argument is only applicable to linear mixed models.

- `t(df = 3, scale = c(10^2, 2.5^2), common.scale = TRUE)`
  The degrees of freedom - `df` argument - must be positive. If `scale` is of length 1, it is repeated along the diagonal for every component. Length 2 repeats just the second element for all slopes. Length equal to the number of fixed effects sees the vector simply turned into a diagonal matrix. Finally, it can a full scale matrix, so long as it is positive definite.
  `t` priors for linear mixed models require that the fixed effects be added to set of parameters that are numerically optimized, and thus can substantially increase running time. In addition, when `common.scale` is `TRUE`, the residual variance must be numerically optimized as well. `normal` priors on the common scale can be fully profiled and do not suffer from this drawback.
  At present, `t` priors cannot be used with the `REML = TRUE` argument as that implies an integral without a closed form solution.

**Covariance Priors**

- `gamma(shape = 2.5, rate = 0, common.scale = TRUE, posterior.scale = "sd")`
  Applicable only for univariate grouping factors. A rate of 0 or a shape of 0 imposes an improper prior. The posterior scale can be "sd" or "var" and determines the scale on which the prior is meant to be applied.

- `invgamma(shape = 0.5, scale = 10^2, common.scale = TRUE, posterior.scale = "sd")`
  Applicable only for univariate grouping factors. A scale of 0 or a shape of 0 imposes an improper prior. Options are as above.
• wishart(df = level.dim + 2.5, scale = Inf, common.scale = TRUE, posterior.scale = "cov")
  A scale of Inf or a shape of 0 imposes an improper prior. The behavior for singular matrices
  with only some infinite eigenvalues is undefined. Posterior scale can be "cov" or "sqrt", the
  latter of which applies to the unique matrix root that is also a valid covariance matrix.

• invwishart(df = level.dim - 0.5, scale = diag(10^2 / (df + level.dim + 1), level.dim),
  A scale of 0 or a shape of 0 imposes an improper prior. The behavior for singular matrices
  with only some zero eigenvalues is undefined.

• custom(fn, chol = FALSE, common.scale = TRUE, scale = "none")
  Applies to the given function (fn). If chol is TRUE, fn is passed a right factor of covariance
  matrix; FALSE results in the matrix being passed directly. scale can be "none", "log", or
  "dev" corresponding to \( p(\Sigma) \), \( \log p(\Sigma) \), and \( -2 \log p(\Sigma) \) respectively.

  Since the prior is may have an arbitrary form, setting common.scale to FALSE for a linear
  mixed model means that full profiling may no longer be possible. As such, that parameter is
  numerically optimized.

Residual Variance Priors

• point(value = 1.0, posterior.scale = "sd")
  Fixes the parameter to a specific value given as either an "sd" or a "var".

• gamma(shape = 0, rate = 0, posterior.scale = "var")
  As above with different defaults.

• invgamma(shape = 0, scale = 0, posterior.scale = "var")
  As above with different defaults.

Evaluating Environment

The variables that the defining environment have populated are:

• p aliased to n_fixef - the number of fixed effects
• n aliased to n_obs - the number of observations
• q.k aliased to level.dim - for covariance priors, the dimension of the grouping factor/grouping
  level
• j.k aliased to n_grps - also for covariance priors, the number of groups that comprise a
  specific grouping factor

Methods

  toString  Pretty-prints the distribution and its parameters.

See Also

blmer() and bglmer(), which produce these objects, and bmerMod-class objects which contain
them.
**bmerMod-class**

Class "bmerMod" of Fitted Mixed-Effect Models

---

### Description

The bmerMod class represents linear or generalized linear or nonlinear mixed-effects models with possible priors over model components. It inherits from the merMod class.

### Objects from the Class

Objects are created by calls to `blmer` or `bglmer`.

### Slots

A bmerMod object contains one additional slot beyond the base merMod class:

- **priors**: A named list comprised of `covPriors`, `fixefPrior`, and `residPrior`.

In addition, the devcomp slot, element cmp includes the penalty item which is the computed deviance for the priors. Add this to the regular deviance to obtain the value of the objective function that is used in optimization.

### See Also

`blmer` and `bglmer`, which produce these objects.

merMod, from which this class inherits.

### Examples

```r
showClass("bmerMod")
methods(class = "bmerMod")
```
Index

*Topic classes
  bmerDist-class, 4
  bmerMod-class, 7

*Topic methods
  blme, 2

*Topic models
  blme, 2

bglmer, 4, 6, 7
bglmer(blme), 2
bglmerMod-class (bmerMod-class), 7
blme, 2
blmer, 4, 6, 7
blmer(blme), 2
blmerMod-class (bmerMod-class), 7
bmerDist (bmerDist-class), 4
bmerDist-class, 4
bmerMod, 3
bmerMod (bmerMod-class), 7
bmerMod-class, 7
distributions, 3

glm, 2, 3
lm, 2, 3
lmer, 2, 3

merMod, 3, 7

print, bmerDist-method (bmerDist-class), 4
print. bmerMod (bmerMod-class), 7
print. summary. bmerMod (bmerMod-class), 7
prior, 2

show, bmerDist-method (bmerDist-class), 4
show, bmerMod-method (bmerMod-class), 7
show, bmerMod (bmerMod-class), 7
summary, bmerMod (bmerMod-class), 7

summary.summary.bmerMod
  (bmerMod-class), 7

vcov.summary.bmerMod(bmerMod-class), 7