Package ‘blme’

January 10, 2026

Version 1.0-7

Date 2026-01-05

Title Bayesian Linear Mixed-Effects Models

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

Imports methods, stats, utils

Suggests expint (>= 0.1-3), testthat

Description Maximum a posteriori estimation for linear and generalized linear mixed-effects models in a Bayesian setting, implementing the methods of Chung, et al. (2013) <doi:10.1007/s11336-013-9328-2>. Extends package 'lme4' (Bates, Maechler, Bolker, and Walker (2015) <doi:10.18637/jss.v067.i01>).

License GPL (>= 2)

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

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

NeedsCompilation no

Author Vincent Dorie [aut, cre] (ORCID: <`https://orcid.org/0000-0002-9576-3064`>),
Douglas Bates [ctb] (lme4 non-modular functions, ORCID: <`https://orcid.org/0000-0001-8316-9503`>),
Martin Maechler [ctb] (lme4 non-modular functions, ORCID: <`https://orcid.org/0000-0002-8685-9910`>),
Ben Bolker [ctb] (lme4 non-modular functions, ORCID: <`https://orcid.org/0000-0002-2127-0443`>),
Steven Walker [ctb] (lme4 non-modular functions, ORCID: <`https://orcid.org/0000-0002-4394-9078`>)

Maintainer Vincent Dorie <`vdorie@gmail.com`>

Repository CRAN

Date/Publication 2026-01-10 07:51:49 UTC

Contents

`blme` *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

```
blmer(formula, data = NULL, 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 = NULL, family = gaussian,
       control = glmerControl(), start = NULL, verbose = 0L,
       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`, `horseshoe`, or `NULL`. Imposes a prior over the fixed effects/modeled coefficients. Default is `NULL`.
`resid.prior`	a BLME prior of family `gamma`, `invamma`, `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`, `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/modeled 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 omitted, the prior is interpretted 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`, `horseshoe`, and `NULL`. `t` and `horseshoe` 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

```
data("sleepstudy", package = "lme4")

### Examples using a covariance prior ##

# Here we are ignoring convergence warnings just to illustate how the package
# is used: this is not a good idea in practice..
control <- lmerControl(check.conv.grad = "ignore")
(fm1 <- blmer(Reaction ~ Days + (0 + Days|Subject), sleepstudy,
              control = control,
              cov.prior = gamma))
(fm2 <- blmer(Reaction ~ Days + (0 + Days|Subject), sleepstudy,
              control = control,
              cov.prior = gamma(shape = 2, rate = 0.5, posterior.scale = 'sd')))
(fm3 <- blmer(Reaction ~ Days + (1 + Days|Subject), sleepstudy,
              control = control,
              cov.prior = wishart))
(fm4 <- blmer(Reaction ~ Days + (1 + Days|Subject), sleepstudy,
              control = control,
              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")))


### Examples using a 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)))

### Example using a residual variance prior ###
# This is the "eight schools" data set; the mode should be at the boundary
# of the space.

control <- lmerControl(check.conv.singular = "ignore",
                       check.nobs.vs.nRE   = "ignore",
                       check.nobs.vs.nlev  = "ignore")
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), control = control, REML = FALSE,
                  resid.prior = point, cov.prior = NULL,
                  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 from 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 interpretted 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, mean = 0, scale = c(10^2, 2.5^2), common.scale = TRUE)`
  The degrees of freedom - `df` argument - must be positive. If `mean` is of length 1, it is repeated for every fixed effect. Length 2 repeats just the second element for all slopes. Otherwise, the length must be equal to that of the number of fixed effects.

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 be 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.

- `horseshoe(mean = 0, global.shrinkage = 2.5, common.scale = TRUE)`

  The horseshoe shrinkage prior is implemented similarly to the `t` prior, in that it requires adding the fixed effects to the parameter set for numeric optimization.

  `global.shrinkage`, also referred to as τ, must be positive and is on the scale of a standard deviation. Local shrinkage parameters are treated as independent across all fixed effects and are integrated out. See *Carvalho et al. (2009)* in the references.

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), common.scale = TRUE, posterior.scale = "cov")`

  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.

References

Carvalho, Carlos M., Nicholas G. Polson, and James G. Scott. "Handling Sparsity via the Horseshoe." AISTATS. Vol. 5. 2009.

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

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

Index