Package ‘glmmFEL’

January 9, 2026

Type Package

Title Generalized Linear Mixed Models via Fully Exponential Laplace in EM

Version 1.0.5

Date 2026-01-07

Description Fit generalized linear mixed models (GLMMs) with normal random effects using first-order Laplace, fully exponential Laplace (FEL) with mean-only corrections, and FEL with mean and covariance corrections in the E-step of an expectation-maximization (EM) algorithm. The current development version provides a matrix-based interface (y, X, Z) and supports binary logit and probit, and Poisson log-link models. An EM framework is used to update fixed effects, random effects, and a single variance component tau^2 for G = tau^2 I, with staged approximations (Laplace -> FEL mean-only -> FEL full) for efficiency and stability. A pseudo-likelihood engine glmmFEL_pl() implements the working-response / working-weights linearization approach of Wolfinger and O'Connell (1993) <doi:10.1080/00949659308811554>, and is adapted from the implementation used in the 'RealVAMS' package (Broatch, Green, and Karl (2018)) <doi:10.32614/RJ-2018-033>. The FEL implementation follows Karl, Yang, and Lohr (2014) <doi:10.1016/j.csda.2013.11.019> and related work (e.g., Tierney, Kass, and Kadane (1989) <doi:10.1080/01621459.1989.10478824>; Rizopoulos, Verbeke, and Lesaffre (2009) <doi:10.1111/j.1467-9868.2008.00704.x>; Steele (1996) <doi:10.2307/2532845>). Package code was drafted with assistance from generative AI tools.

License GPL-3

Encoding UTF-8

Imports Matrix, numDeriv, stats, methods

Suggests testthat (>= 3.0.0), MASS, knitr, rmarkdown, nlme, mvglmmRank, lme4

Config/testthat/edition 3

RoxygenNote 7.3.3

NeedsCompilation no

Author Andrew T. Karl [cre, aut] (ORCID: <https://orcid.org/0000-0002-5933-8706>)

Maintainer Andrew T. Karl <akarl@asu.edu>

Repository CRAN

Date/Publication 2026-01-09 14:50:02 UTC

Contents

`glmmFEL-package` *glmmFEL: Generalized Linear Mixed Models via Fully Exponential Laplace in EM*

Description

glmmFEL fits generalized linear mixed models (GLMMs) with normal random effects using a matrix-based interface where users supply (y, X, Z) directly. Model fitting is performed with an EM algorithm whose E-step can be approximated using first-order Laplace or fully exponential Laplace (mean-only or mean + covariance corrections), and includes pseudo-likelihood alternatives based on working-response / working-weights linearization.

This development branch is intentionally **matrix-first** and supports a flexible (including multiple-membership) random-effects design matrix Z; see the package NOTE for details on current covariance support and planned extensions.

Supported families in this branch:

- `family = stats::binomial(link = "probit")` (binary probit),
- `family = stats::binomial(link = "logit")` (binary logit),
- `family = stats::poisson(link = "log")` (Poisson log-link).

NOTE

This matrix-only development branch is a streamlined rewrite based on the binary/Poisson engines in **mvglmmRank** (which were based on Karl, Yang, and Lohr, 2014). The fully exponential Laplace EM strategy is described in Karl, Yang, and Lohr (2014) and related fully exponential Laplace work such as Tierney, Kass, and Kadane (1989) and Rizopoulos, Verbeke, and Lesaffre (2009); the FE-within-EM lineage is attributed to Steele (1996). Karl, Yang, and Lohr (2014) referenced the source code of package JM when writing their own code.

At present, the package supports a single random-effect vector (i.e., one variance component). However, the theory in the references above—and the joint Poisson-binary model already implemented in **mvglmmRank**—extends naturally to a block-diagonal random-effects covariance matrix G. This would allow multiple independent random effects and/or random intercept–slope specifications, with intercepts and slopes either correlated or uncorrelated (depending on the block structure). Enabling this functionality primarily involves allowing the user to specify a G structure in the main fitting function. The maintainer has working prototype code for selected G structures, but it has not yet been tested sufficiently for release; future support may be added. If you would like a particular G structure supported, please email the package maintainer with a request.

Importantly, the current implementation already supports a multiple-membership (or otherwise arbitrary) random-effects design matrix Z, which is more general than what is available in some other implementations. In particular, the multi-membership setting allows more than one random effect from the same variance component to be active on the same observation, possibly with different weight entries per random effect; see Karl, Yang, and Lohr (2014) for details.

The `RSPL`/`MSPL` pseudo-likelihood code paths are adapted from the RealVAMS implementation described in Broatch, Green, and Karl (2018), which follows Wolfinger and O'Connell (1993) closely (working response + working weights).

Acknowledgments

OpenAI's GPT models (such as GPT-5 Pro) were used to assist with coding and roxygen documentation; all content was reviewed and finalized by the author.

Approximations

`glmmFEL()` supports:

- `"Laplace"`: first-order Laplace approximation,
- `"FE_mean"`: fully exponential Laplace corrections to $\widehat{\eta}$ only,
- `"FE_full"` (or `"FE"`): fully exponential Laplace corrections to both $\widehat{\eta}$ and $\widehat{\mathrm{Var}}(\eta \mid y)$,
- `"RSPL"` / `"MSPL"`: restricted/marginal pseudo-likelihood (working response / working weights).

Output

`glmmFEL()` returns an object of class `"glmmFELMod"` containing:

- `beta`: fixed-effect estimates,
- `eta`: empirical Bayes predictions of random effects,
- `tau2`: the scalar variance component,

- `G`: $q \times q$ covariance matrix (diagonal in this branch),
- `var_eta`: prediction-error covariance for `eta` (approx.),
- `vcov_beta`: approximate covariance of `beta` when available,
- `convergence`: iteration counts and flags.

Author(s)

Maintainer: Andrew T. Karl <akarl@asu.edu> (ORCID)

References

Broatch, J., Green, J. G., & Karl, A. T. (2018). RealVAMS: An R Package for Fitting a Multivariate Value-added Model (VAM). *The R Journal*, *10*(1), 22–30. doi:10.32614/RJ2018033

Karl, A. T., Yang, Y., & Lohr, S. L. (2014). Computation of maximum likelihood estimates for multiresponse generalized linear mixed models with non-nested, correlated random effects. *Computational Statistics & Data Analysis*, *73*, 146–162. doi:10.1016/j.csda.2013.11.019

Rizopoulos, D., Verbeke, G., & Lesaffre, E. (2009). Fully exponential Laplace approximations in joint models for longitudinal and survival data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *71*(3), 637–654. doi:10.1111/j.14679868.2008.00704.x

Rizopoulos, D. (2010). JM: An R package for the joint modeling of longitudinal and time-to-event data. *Journal of Statistical Software*, *35*(9), 1–33. doi:10.18637/jss.v035.i09

Steele, B. M. (1996). A modified EM algorithm for estimation in generalized mixed models. *Biometrics*, *52*(4), 1295–1310. doi:10.2307/2532845

Tierney, L., Kass, R. E., & Kadane, J. B. (1989). Fully exponential Laplace approximations to expectations and variances of nonpositive functions. *Journal of the American Statistical Association*, *84*(407), 710–716. doi:10.1080/01621459.1989.10478824

Wolfinger, R., & O'Connell, M. (1993). Generalized linear mixed models: a pseudo-likelihood approach. *Journal of Statistical Computation and Simulation*, *48*(3–4), 233–243. doi:10.1080/00949659308811554

See Also

`glmmFEL_pl()` for the pseudo-likelihood engines.

`coef.glmmFELMod` *Extract model coefficients (fixed effects)*

Description

Extract model coefficients (fixed effects)

Usage

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

Arguments

`object`	A `glmmFELMod` object.
`...`	Unused.

Value

A named numeric vector of estimated fixed-effect regression coefficients (on the linear predictor scale). Names correspond to columns of the fixed-effects design matrix `X` when available.

`fitted.glmmFELMod` *Extract fitted values*

Description

Extract fitted values

Usage

```
## S3 method for class 'glmmFELMod'
fitted(object, type = c("response", "link"), ...)
```

Arguments

`object`	A `glmmFELMod` object.
`type`	Either `"link"` (linear predictor) or `"response"`.
`...`	Unused.

Value

A numeric vector of fitted values. If `type = "link"`, the fitted linear predictor values are returned. If `type = "response"`, the fitted mean response values on the response scale are returned. The length equals the number of observations in the fitted object.

glmmFEL *Fit GLMMs via Laplace and fully exponential Laplace (matrix interface)*

Description

`glmmFEL()` fits a generalized linear mixed model (GLMM) with multivariate normal random effects using EM-type algorithms and likelihood approximations:

- first-order Laplace (`approx = "Laplace"`),
- fully exponential corrections to the random-effects mean (`approx = "FE_mean"`),
- fully exponential corrections to both mean and covariance (`approx = "FE_full"` / `"FE"`),
- pseudo-likelihood / PL linearization (`approx = "RSPL"` or `"MSPL"`) via `glmmFEL_pl()`.

This **development branch is matrix-only**: you provide the response `y`, fixed-effects design matrix `X`, and random-effects design matrix `Z`. A formula interface (via optional 'lme4' helpers) and structured G parameterizations are in development.

Random effects are assumed $\eta \sim N(0, G)$ with a **single** variance component

$$G = \tau^2 I_q,$$

allowing arbitrary (including multi-membership) `Z` while keeping the variance update simple and stable.

Usage

```
glmmFEL(
  y,
  X,
  Z,
  family = stats::binomial(link = "probit"),
  approx = c("FE", "Laplace", "FE_mean", "FE_full", "RSPL", "MSPL"),
  max_iter = 200,
  tol = 1e-06,
  control = list()
)
```

Arguments

`y`	Numeric response vector of length n. For `family = "binomial_probit"` / `binomial(link = "probit")` or `family = "binomial_logit"` / `binomial(link = "logit")`, values must be 0 or 1.
`X`	Fixed-effects design matrix of dimension $n \times p$. May be a base R matrix or a matrix-like object; it is internally coerced to a base numeric matrix. Must have full column rank.

`Z` Random-effects design matrix of dimension $n \times q$. May be a base R matrix or a **Matrix** object. Internally it is coerced to a sparse "dgCMatrix" where possible (to preserve sparsity). Must have at least one column (no purely fixed-effects models).

`family` Either a character string or a stats::family object indicating the model family. The argument is resolved via `glmmfe_resolve_family()`.

`approx` Approximation type, resolved via `glmmfe_resolve_approx()`. Accepted values (case-insensitive) include:

- "Laplace" – first-order Laplace approximation,
- "FE_mean" – staged algorithm: Laplace phase then FE mean corrections,
- "FE" / "FE_full" – staged algorithm: Laplace phase, then FE mean, then FE covariance corrections (default),
- "RSPL" – restricted pseudo-likelihood (REML-style) linearization,
- "MSPL" – marginal pseudo-likelihood (ML-style) linearization.

`max_iter` Maximum number of EM iterations (outer iterations over β and τ^2). Can be overridden by `control$em_max_iter`.

`tol` Baseline convergence tolerance for the EM algorithm. The staged thresholds default to:

- Laplace stage: `tol_laplace = 10 * tol`,
- FE-mean stage: `tol_fe_mean = 3 * tol`,
- FE-full stage: `tol_fe_full = tol`.

You can override these via `control$tol_laplace`, `control$tol_fe_mean`, and `control$tol_fe_full`.

`control` List of optional control parameters. Recognized entries include:

- `em_max_iter`, `em_tol`,
- `tol_laplace`, `tol_fe_mean`, `tol_fe_full`,
- `eta_max_iter`, `eta_tol_grad`,
- `beta_max_iter`, `beta_tol`,
- `tau2_init` (initial value for τ^2),
- `vc_eps` (lower bound for τ^2),
- `max_nq_mem` (memory guard for FE trace intermediates),
- `verbose` (logical),
- `beta_step_max` (max Newton step size for beta; default 2),
- `beta_ls_max_iter` (max line-search halvings; default 12),
- `beta_hess_ridge_init` (initial ridge for Hessian; default 1e-8),
- `beta_hess_ridge_max` (max ridge; default 1e2)

Value

A fitted model object of class `glmmFELMod`.

NOTE

This matrix-only development branch is a streamlined rewrite based on the binary/Poisson engines in **mvglmmRank** (which were based on Karl, Yang, and Lohr, 2014). The fully exponential Laplace EM strategy is described in Karl, Yang, and Lohr (2014) and related fully exponential Laplace work such as Tierney, Kass, and Kadane (1989) and Rizopoulos, Verbeke, and Lesaffre (2009); the FE-within-EM lineage is attributed to Steele (1996). Karl, Yang, and Lohr (2014) referenced the source code of package JM when writing their own code.

At present, the package supports a single random-effect vector (i.e., one variance component). However, the theory in the references above—and the joint Poisson-binary model already implemented in **mvglmmRank**—extends naturally to a block-diagonal random-effects covariance matrix G. This would allow multiple independent random effects and/or random intercept–slope specifications, with intercepts and slopes either correlated or uncorrelated (depending on the block structure). Enabling this functionality primarily involves allowing the user to specify a G structure in the main fitting function. The maintainer has working prototype code for selected G structures, but it has not yet been tested sufficiently for release; future support may be added. If you would like a particular G structure supported, please email the package maintainer with a request.

Importantly, the current implementation already supports a multiple-membership (or otherwise arbitrary) random-effects design matrix Z, which is more general than what is available in some other implementations. In particular, the multi-membership setting allows more than one random effect from the same variance component to be active on the same observation, possibly with different weight entries per random effect; see Karl, Yang, and Lohr (2014) for details.

The `RSPL`/`MSPL` pseudo-likelihood code paths are adapted from the RealVAMS implementation described in Broatch, Green, and Karl (2018), which follows Wolfinger and O'Connell (1993) closely (working response + working weights).

Acknowledgments

OpenAI's GPT models (such as GPT-5 Pro) were used to assist with coding and roxygen documentation; all content was reviewed and finalized by the author.

References

Broatch, J., Green, J. G., & Karl, A. T. (2018). RealVAMS: An R Package for Fitting a Multivariate Value-added Model (VAM). *The R Journal*, *10*(1), 22–30. doi:10.32614/RJ2018033

Karl, A. T., Yang, Y., & Lohr, S. L. (2014). Computation of maximum likelihood estimates for multiresponse generalized linear mixed models with non-nested, correlated random effects. *Computational Statistics & Data Analysis*, *73*, 146–162. doi:10.1016/j.csda.2013.11.019

Rizopoulos, D., Verbeke, G., & Lesaffre, E. (2009). Fully exponential Laplace approximations in joint models for longitudinal and survival data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *71*(3), 637–654. doi:10.1111/j.14679868.2008.00704.x

Rizopoulos, D. (2010). JM: An R package for the joint modeling of longitudinal and time-to-event data. *Journal of Statistical Software*, *35*(9), 1–33. doi:10.18637/jss.v035.i09

Steele, B. M. (1996). A modified EM algorithm for estimation in generalized mixed models. *Biometrics*, *52*(4), 1295–1310. doi:10.2307/2532845

Tierney, L., Kass, R. E., & Kadane, J. B. (1989). Fully exponential Laplace approximations to expectations and variances of nonpositive functions. *Journal of the American Statistical Association*, *84*(407), 710–716. doi:10.1080/01621459.1989.10478824

Wolfinger, R., & O'Connell, M. (1993). Generalized linear mixed models: a pseudo-likelihood approach. *Journal of Statistical Computation and Simulation*, *48*(3–4), 233–243. doi:10.1080/00949659308811554

Examples

```
## Example 1: Simulated probit random-intercept GLMM (matrix interface)
set.seed(1)
n_id <- 30
m_per_id <- 6
n <- n_id * m_per_id
id <- factor(rep(seq_len(n_id), each = m_per_id))
x  <- rnorm(n)
X  <- model.matrix(~ x)
Z  <- Matrix::sparseMatrix(i = seq_len(n),
                           j = as.integer(id),
                           x = 1,
                           dims = c(n, n_id))
beta_true <- c(0.2, 0.7)
tau2_true <- 0.5
eta_true  <- rnorm(n_id, sd = sqrt(tau2_true))
lp <- as.vector(X %*% beta_true + Z %*% eta_true)
y  <- rbinom(n, 1, pnorm(lp))

fit <- glmmFEL(y, X, Z, family = "binomial_probit", approx = "Laplace")
fit$beta
fit$tau2

## Example 2: Get X, y, Z from an lme4 formula (without a glmmFEL formula wrapper)

if (requireNamespace("lme4", quietly = TRUE)) {
  dat <- data.frame(y = y, x = x, id = id)
  lf  <- lme4::lFormula(y ~ x + (1 | id), data = dat)
  X_lme4 <- lf$X
  Z_lme4 <- Matrix::t(lf$reTrms$Zt)
  y_lme4 <- lf$fr$y

  fit2 <- glmmFEL(y_lme4, X_lme4, Z_lme4, family = "binomial_probit", approx = "Laplace")
}
```

`logLik.glmmFELMod` *Extract log-likelihood (approximate)*

Description

Extract log-likelihood (approximate)

Usage

```
## S3 method for class 'glmmFELMod'
logLik(object, ...)
```

Arguments

`object`	A `glmmFELMod` object.
`...`	Unused.

Value

An object of class `"logLik"` giving the approximate log-likelihood stored in `object$logLik`, with attributes `"df"` (effective number of parameters, taken as `length(beta) + 1` for the variance component) and `"nobs"` (number of observations).

`predict.glmmFELMod` *Predict from a fitted glmmFEL model*

Description

Predict from a fitted glmmFEL model

Usage

```
## S3 method for class 'glmmFELMod'
predict(object, newdata = NULL, type = c("response", "link"), ...)
```

Arguments

`object`	A `glmmFELMod` object.
`newdata`	Not supported in this branch (matrix interface only).
`type`	Either `"link"` or `"response"`.
`...`	Unused.

Value

A numeric vector of predictions. If `type = "link"`, predictions are returned on the linear predictor scale. If `type = "response"`, predictions are returned on the response scale. The length equals the number of observations used to fit the model. Supplying `newdata` triggers an error in this matrix-only branch.

`print.glmmFELMod` *Print a glmmFEL model object*

Description

Print a glmmFEL model object

Usage

```
## S3 method for class 'glmmFELMod'
print(x, ...)
```

Arguments

`x`	A `glmmFELMod` object.
`...`	Unused.

Value

Returns `x` invisibly (a `glmmFELMod` object), called for its side effect of printing model information to the console.

`print.summary.glmmFELMod` *Print a summary.glmmFELMod object*

Description

Print a summary.glmmFELMod object

Usage

```
## S3 method for class 'summary.glmmFELMod'
print(x, ...)
```

Arguments

`x`	A `summary.glmmFELMod` object.
`...`	Unused.

Value

Returns `x` invisibly (a `summary.glmmFELMod` object), called for its side effect of printing the summary to the console.

`summary.glmmFELMod` *Summary for a glmmFEL model object*

Description

Summary for a glmmFEL model object

Usage

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

Arguments

`object`	A `glmmFELMod` object.
`...`	Unused.

Value

An object of class `summary.glmmFELMod` (a list) containing summary information for the fitted model, including the call, family, approximation label, approximate log-likelihood, estimated variance component `tau2`, a coefficient table for fixed effects (and standard errors when available), number of observations, and convergence information.

`vcov.glmmFELMod` *Extract the covariance matrix of the fixed effects*

Description

Extract the covariance matrix of the fixed effects

Usage

```
## S3 method for class 'glmmFELMod'
vcov(object, ...)
```

Arguments

`object`	A `glmmFELMod` object.
`...`	Unused.

Value

A variance-covariance matrix for the estimated fixed-effect regression coefficients. Row and column names correspond to coefficient names when available.

Index