Package: EMLMLasso (via r-universe)

November 18, 2026

Title EM Algorithm for Lasso-Penalized Linear Mixed Models

Version 0.1.0

Description Fits linear mixed models with a lasso penalty on the fixed-effects coefficients via an Expectation-Maximization (EM) algorithm. Random effects are treated as missing data at each E-step and the fixed-effects update is solved with a coordinate-descent lasso ('glmnet'), so the same penalized-likelihood machinery used for ordinary linear models extends naturally to clustered/longitudinal data. By default, EMLMLasso() fits the model over a data-driven grid of penalty values and selects the tuning parameter by Bayesian Information Criterion. This is a modification of the method of Oliveira, Schumacher, and Lachos (2026) <doi:10.1177/09622802251399913>, in which the tuning parameter is re-selected by cross-validation at every EM iteration.

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 3.5)

Imports glmnet, graphics, mvtnorm, stats, utils

Suggests testthat (>= 3.0.0), knitr, rmarkdown, lme4

Config/testthat/edition 3

VignetteBuilder knitr

URL https://github.com/fernandalschumacher/EMLMLasso

BugReports https://github.com/fernandalschumacher/EMLMLasso/issues

Config/roxygen2/version 8.1.0

NeedsCompilation no

Author Fernanda Lang Schumacher [aut, cre], Daniela C.R. Oliveira [aut], Victor H. Lachos [aut], Marcos Santos de Oliveira [aut]

Maintainer Fernanda Lang Schumacher <fernandalschumacher@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-11-17 18:30:09 UTC

RemoteUrl https://github.com/cran/EMLMLasso

RemoteRef HEAD

RemoteSha 834f447cebb5d21de46f51a7e023e8bb09ffc4ef

Contents

coef.EMLMLasso 2
EMLMLasso 3
plot.EMLMLasso 7
riboflavinV100 7
simulate_lasso_lmm 9

Index **11**

`coef.EMLMLasso` *Extract coefficients, fitted values or residuals from an* `EMLMLasso` *fit*

Description

Extract coefficients, fitted values or residuals from an `EMLMLasso` fit

Usage

```
## S3 method for class 'EMLMLasso'
coef(object, refit = FALSE, ...)

## S3 method for class 'EMLMLasso'
fitted(object, refit = FALSE, ...)

## S3 method for class 'EMLMLasso'
residuals(object, refit = FALSE, ...)
```

Arguments

`object`	an "`EMLMLasso`" object.
`refit`	if `TRUE`, return values from the post-selection `lme4::lmer()` refit instead of the lasso fit's (see the `refit` argument to `EMLMLasso()`; the object must have been created with `refit = TRUE`) – for `coef()`, its fixed-effect point estimates (`NA` for covariates the lasso did not select); for `fitted()`, its fitted values (fixed and random effects combined); for `residuals()`, `y` minus those fitted values. Default `FALSE`.
`...`	unused.

Value

For `coef()`, a named numeric vector of fixed-effect estimates, one per column of the fixed-effects design matrix (including the intercept when `intercept = TRUE` in `EMLMLasso()`); covariates shrunk out by the lasso are exactly `0`, or `NA` when `refit = TRUE`. For `fitted()` and `residuals()`, a numeric vector of length equal to the total number of observations, in the same order as the stored response `y` (observations grouped by `groupVar`), with fitted values combining the fixed effects (including `implied_intercept` when the data were centred) and the predicted random effects.

`EMLMLasso` *Fit a lasso-penalized linear mixed model via EM*

Description

Fits a linear mixed model with a lasso penalty on the fixed-effects coefficients, using an Expectation-Maximization (EM) algorithm (at each E-step, the random effects are treated as missing data and their conditional means and covariances given the data are computed in closed form; at each M-step, the fixed effects are updated by a coordinate-descent lasso fit (`glmnet::glmnet()`) on the random-effects-adjusted response, and the variance components are updated in closed form). This is the package's main function and the argument `lambda` decides whether it fits one model or searches a grid and selects by BIC.

Usage

```
EMLMLasso(
  data,
  formFixed,
  groupVar,
  formRandom = ~1,
  lambda = NULL,
  intercept = FALSE,
  center = TRUE,
  n_lambdas = 30,
  lambda_min_ratio = 0.001,
  precision = 1e-04,
  MaxIter = 200,
  penalty_factor = NULL,
  standardize_x = TRUE,
  refit = FALSE,
  verbose = interactive()
)
```

Arguments

`data`	a data.frame containing all variables referenced by `formFixed`, `formRandom` and `groupVar`.

`formFixed` two-sided formula for the fixed effects, e.g. `y ~ x1 + x2`. `y ~ .` is also accepted, meaning every other column of `data` is a fixed-effect candidate except the response and `groupVar` itself.

`groupVar` character string naming the subject/cluster variable in `data`.

`formRandom` one-sided formula for the random effects, e.g. `~1` (random intercept, the default) or `~time` (random intercept and slope).

`lambda` controls what gets fit:

`NULL` **(the default)** search a data-driven grid of `n_lambdas` values (as in `glmnet::glmnet()`'s own grid construction, spanning down to `lambda_min_ratio` of the largest value) and select by BIC.

a single non-negative number fit that one lambda directly, no search, no BIC comparison.

a numeric vector of two or more non-negative values search exactly this grid (sorted, deduplicated, descending) and select by BIC, instead of building one automatically – `n_lambdas` and `lambda_min_ratio` are ignored in this case.

`intercept` if `TRUE`, force an intercept into the fixed-effects design matrix; if `FALSE` (the default), force it out. This always wins over whatever `formFixed` itself implies – `y ~ x1 + x2` and `y ~ x1 + x2 - 1` are treated identically. See also `center`.

`center` if `TRUE` (the default) and `intercept = FALSE`, the response and every fixed-effect covariate are centred at their overall means before fitting, matching the centred setting of Equation 1 of Oliveira et al. (2026). Coefficients are still returned on the original scale, and the implied intercept ($\bar{y} - \bar{x}'\hat{\beta}$) is stored as `implied_intercept` and included by `fitted()` and `residuals()`. Set to `FALSE` to fit the uncentred data without an intercept (only sensible if the data are already centred). Ignored when `intercept = TRUE`, since the intercept then absorbs the means.

`n_lambdas` number of `lambda` values in the data-driven grid (as in `glmnet::glmnet()`'s `nlambda`). Only used when `lambda = NULL`. Default `30`.

`lambda_min_ratio`

smallest lambda in the data-driven grid, as a fraction of the largest (as in `glmnet::glmnet()`'s `lambda.min.ratio`). Only used when `lambda = NULL`. Default `0.001`, ten times smaller than glmnet's own default for `p > n` data (`0.01`).

`precision` convergence tolerance on the Euclidean distance between successive parameter vectors (beta, sigma2, D). Default `1e-4`.

`MaxIter` maximum number of EM iterations. Default `200`.

`penalty_factor` optional vector of per-coefficient penalty multipliers, in the same order as the columns of the fixed-effects design matrix (see `stats::model.matrix()` applied to `formFixed`). If `NULL` (the default), all coefficients are penalized equally except an intercept, if present, which is left unpenalized. Set an entry to `0` to force a coefficient to stay in the model.

`standardize_x` if `TRUE` (the default), each fixed-effect covariate (other than an intercept, if `formFixed` includes one) is scaled to unit variance once before fitting, and coefficients are converted back to the original scale of `x` before being returned. Covariates are scaled here but not centred (centring is controlled by `center`).

Because the penalty is applied on this scale, the selected model can depend on this option, but `coef()`, `fitted()` and `residuals()` always report results on the original scale of `x`. Set to `FALSE` to fit on the original scale of `x` instead (relying only on glmnet's own default internal standardization, recomputed every M-step).

`refit` if `TRUE`, once the (selected or only) model is available, also refits an ordinary (unpenalized) `lme4::lmer()` model on just its non-zero covariates – same random-effects structure – to correct the shrinkage bias lasso point estimates always carry. When `lambda` triggers a search, only the selected (best) model is refit, not every grid point. Requires the `lme4` package. **The refit's own standard errors, p-values and confidence intervals must not be used or reported** – the covariate set was chosen using this same data, so they do not account for that selection step and understate the true uncertainty (classic post-selection inference). The original lasso fit (`coefficients`, `AIC`, `BIC`, etc.) is always kept unchanged alongside it. Default `FALSE`.

`verbose` if `TRUE`, prints progress (one line every 10 EM iterations, plus, when searching, one line per grid value). Defaults to `interactive()`.

Details

Following Oliveira et al. (2026), the model is $Y_i = X_i\beta + Z_i b_i + \epsilon_i$, with $b_i \sim N_q(0, D)$ independent of $\epsilon_i \sim N_{n_i}(0, \sigma^2 I_{n_i})$, so the covariance for subject/cluster i is $\Sigma_i = Z_i D Z_i' + \sigma^2 I_{n_i}$. `lambda` plays the role of $\lambda^{(k)}$ in the M-step of Algorithm 1 of Oliveira et al. (2026) (their Equation 8) – the penalty passed directly to `glmnet::glmnet()`. Following Equation 1 of the same paper, which assumes a centred response and centred covariates, no intercept is fit by default; instead, the response and covariates are centred before fitting (see `center`).

This package implements the EMLMLasso algorithm of Oliveira, Schumacher and Lachos (2026) with modifications. Most notably, the penalty λ is selected by BIC over a grid of candidate values, with λ held fixed within each EM fit, instead of being re-selected by cross-validation at every EM iteration as in Algorithm 1 of the paper. Convergence is also assessed on the change in parameter estimates rather than on the relative change in the penalized log-likelihood. Results may therefore differ from those reported in the paper.

Per-iteration cross-validation was dropped because it was slow (one k-fold cross-validation per EM iteration) and, in simulations, produced a lambda path that was not monotone in the number of active covariates. See `NEWS.md` for further details.

Value

An object of class `"EMLMLasso"`, a list with elements:

coefficients named vector of estimated fixed-effects coefficients for the selected (or only) model.

implied_intercept the intercept implied by centring, $\bar{y} - \bar{x}'\hat{\beta}$, or `0` if the data were not centred (see `center`).

centered whether the response and covariates were centred.

sigma2 estimated residual variance.

D estimated random-effects covariance matrix.

ranef matrix of estimated (empirical Bayes) random effects, one row per group.

lambda the penalty value of the selected (or only) model.

loglik the marginal (unpenalized) log-likelihood at each iteration, for the selected (or only) model.

AIC, BIC information criteria for the selected (or only) model.

npar number of estimated parameters used in AIC and BIC: the non-zero fixed effects, plus σ^2 and the $q(q+1)/2$ unique elements of D.

n, N number of groups and total number of observations.

penalty_factor the penalty multipliers used.

iter number of EM iterations performed for the selected (or only) model.

converged whether the selected (or only) model's EM algorithm reached `precision` within `MaxIter` iterations and its M-steps' `glmnet::glmnet()` fits themselves converged (see `glmnet_converged`). `FALSE` if `MaxIter` was reached first.

glmnet_converged whether every M-step of the selected (or only) model's `glmnet::glmnet()` calls converged.

x, z, y, nj, groups the design matrices/response/group sizes used to fit the model.

call, formFixed, formRandom, groupVar the matched call and model specification.

search_performed `TRUE` if `lambda` triggered a grid search (`NULL` or a vector), `FALSE` if a single lambda was fit directly.

lambda_grid the full grid of lambda values considered, or `NULL` if `search_performed` is `FALSE`.

bics, active_vars BIC and number of active (non-zero) covariates at each grid value (`NA` at any lambda excluded because its M-step's `glmnet::glmnet()` fit did not converge), or `NULL` if `search_performed` is `FALSE`.

grid_converged logical vector, one per grid value: whether that lambda's fit converged and was used in the BIC comparison, or `NULL` if `search_performed` is `FALSE`.

all_models list of all fits considered, one per grid value (including any excluded from BIC comparison, for inspection), or `NULL` if `search_performed` is `FALSE`.

If `refit = TRUE`, this additionally carries `refit_fit`, `refit_coefficients`, `refit_formula` and `refit_singular` for the selected (or only) model – see the `refit` argument above.

References

Oliveira DCR, Schumacher FL, Lachos VH (2026). The use of the EM algorithm for regularization problems in high-dimensional linear mixed-effects models. *Statistical Methods in Medical Research*, 35(2), 240-254. doi:10.1177/09622802251399913

Examples

```
set.seed(1)
p <- 5
dat <- simulate_lasso_lmm(n = 25, nj = 4, p = p, pstar = 2, seed = 1)
fixedForm <- stats::as.formula(paste("y ~", paste0("x", seq_len(p), collapse = " + ")))

# fit a single lambda directly
fit <- EMLMLasso(dat, formFixed = fixedForm, groupVar = "id", formRandom = ~time,
                 lambda = 0.1, MaxIter = 50, verbose = FALSE)
```

```
fit
coef(fit)

# search a data-driven grid and select by BIC (the common case)
search <- EMLMLasso(dat, formFixed = fixedForm, groupVar = "id", formRandom = ~time,
                    n_lambdas = 8, MaxIter = 50, verbose = FALSE)
search
coef(search)
```

`plot.EMLMLasso` *Plot BIC against the lambda grid*

Description

Only applicable to an `"EMLMLasso"` object created with a BIC search (`lambda = NULL` or a vector of candidate values, the default) – errors informatively if `x` was fit at a single `lambda` instead, since there is then no grid to plot.

Usage

```
## S3 method for class 'EMLMLasso'
plot(x, ...)
```

Arguments

`x`	an `"EMLMLasso"` object.
`...`	further arguments passed to `graphics::plot()`.

Value

Invisibly returns `x`, the input `"EMLMLasso"` object, unchanged. Called for its side effect of plotting BIC against the lambda grid, with the selected lambda highlighted in red.

`riboflavinV100` *Longitudinal riboflavin production data (100-gene reduction)*

Description

Gene expression and riboflavin production data on *Bacillus subtilis*, with a longitudinal (repeated-measures) structure recovered from the original microarray metadata, reproducing the real-data application in Oliveira, Schumacher and Lachos (2026), Section 5.

Usage

```
riboflavinV100
```

Format

A data frame with 71 rows (observations) and 103 columns. `strain` (factor, 28 levels) is the *B. subtilis* strain/isolate a given observation belongs to (the grouping/cluster variable), each measured 2 to 4 times; `time` (numeric) is the time of measurement in hours after induction (24, 30, 48 or 52); `y` (numeric) is log-transformed riboflavin (vitamin B2) production rate, the response variable. The remaining 100 columns are numeric log-expression levels, one per gene (column names are the genes' probe identifiers, e.g. `YCIC_at`) – the 100 genes of largest empirical variance across these 71 observations (see Details).

Details

The expression matrix and response are the standard `riboflavin` dataset (Bühlmann, Kalisch and Meier, 2014), most commonly distributed via the `hdi` package, which records 71 microarray samples and 4088 gene expression levels but exposes no grouping or time variable. The original .CEL microarray file names (preserved as that object's row names) encode the *B. subtilis* strain (e.g. `"Fbat1070"`) and the measurement time in hours (e.g. `"PT24"`) each sample belongs to. Parsing these names recovers exactly the longitudinal structure described in Oliveira, Schumacher and Lachos (2026, Section 5): 28 strains, each measured 2-4 times, 71 observations in total.

Following prior work, genes are filtered to the 100 of largest empirical variance across these 71 observations – the same `riboflavinV100` reduction used by the paper itself (following Schelldorfer, Bühlmann and van de Geer, 2011), so that the analysis setting matches the paper's (results differ from its Table 5 because of the change in λ selection; see `EMLMLasso()`) and fitting is fast. See `vignette("EMLMLasso-riboflavin")` for a worked example fitting `EMLMLasso()` to it.

Source

Derived from the `riboflavin` dataset distributed with the `hdi` package (https://github.com/cran/hdi), combined with strain/time metadata parsed from that dataset's original microarray file names.

References

Oliveira DCR, Schumacher FL, Lachos VH (2026). "The use of the EM algorithm for regularization problems in high-dimensional linear mixed-effects models." *Statistical Methods in Medical Research*, 35(2), 240-254. doi:10.1177/09622802251399913

Bühlmann P, Kalisch M, Meier L (2014). "High-Dimensional Statistics with a View Toward Applications in Biology." *Annual Review of Statistics and Its Application*, 1, 255-278.

Schelldorfer J, Bühlmann P, van de Geer S (2011). "Estimation for high-dimensional linear mixed-effects models using ℓ_1-penalization." *Scandinavian Journal of Statistics*, 38(2), 197-214.

Examples

```
data(riboflavinV100)
dim(riboflavinV100)
table(table(riboflavinV100$strain))  # group sizes: 2 to 4 per strain
# see vignette("EMLMLasso-riboflavin") for a full worked example
```

`simulate_lasso_lmm` *Simulate longitudinal data from a sparse linear mixed model*

Description

Generates balanced longitudinal/clustered data from a linear mixed model with `p` fixed-effect covariates (only the first `pstar` have a non-zero coefficient) and a random intercept + random slope on `time`. This is the data-generating mechanism used in the package examples, tests and vignette (adapted from Scenario 1 of Oliveira et al. 2026, except that covariates are drawn from N(0, 1) rather than N(6, 1) and then centred).

Usage

```
simulate_lasso_lmm(
  n = 50,
  nj = 5,
  p = 10,
  pstar = 4,
  beta_value = 1,
  sigmae = 1,
  D = NULL,
  seed = NULL
)
```

Arguments

`n`	number of subjects/clusters.
`nj`	number of observations per subject, n_i in Oliveira et al. (2026) (balanced design).
`p`	total number of fixed-effect covariates.
`pstar`	number of covariates with a non-zero true coefficient (`beta_value`); the remaining `p - pstar` have a true coefficient of 0. Must be `<= p` (p^* in Oliveira et al. 2026).
`beta_value`	the common value used for the non-zero true coefficients.
`sigmae`	true residual variance.
`D`	true 2x2 covariance matrix of the random intercept and random slope on `time`. Defaults to `matrix(c(1, 0.25, 0.25, 1), 2, 2)`, the matrix in Equation 11 of Oliveira et al. (2026).
`seed`	optional seed passed to `set.seed()` for reproducibility.

Value

A data.frame with columns `id`, `time`, `y`, and `x1, ..., xp`.

References

Oliveira DCR, Schumacher FL, Lachos VH (2026). The use of the EM algorithm for regularization problems in high-dimensional linear mixed-effects models. *Statistical Methods in Medical Research*, 35(2), 240-254. doi:10.1177/09622802251399913

Examples

```
dat <- simulate_lasso_lmm(n = 20, nj = 4, p = 6, pstar = 2, seed = 42)
head(dat)
```

Index

∗ **datasets**
riboflavinV100, 7

coef.EMLMLasso, 2

EMLMLasso, 3
EMLMLasso(), *2*, *3*, 8

fitted.EMLMLasso *(*coef.EMLMLasso*)*, 2

glmnet::glmnet(), *3–5*
graphics::plot(), *7*

plot.EMLMLasso, 7

residuals.EMLMLasso *(*coef.EMLMLasso*)*, 2
riboflavinV100, 7

set.seed(), *9*
simulate_lasso_lmm, 9
stats::model.matrix(), *4*