Package: regMR (via r-universe)

October 5, 2026

Title Regularized Finite Mixture Regression Models Using MM Algorithm

Version 1.0.0

Description Provides a comprehensive framework for fitting regularized finite mixture regression models via an MM algorithm. The sparse group lasso (sgl) penalty is applied to parameter updates within the MM algorithm for variable selection with respect to groups and covariates. The package provides multiple functions for estimation and allows users to fit models over different lambda-alpha sgl penalties and group counts.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 4.1.0)

Imports furrr, future, ggplot2, graphics, mclust, purrr, Rcpp, reshape2, plotly, stats

Suggests mvtnorm, rlang, testthat (>= 3.0.0)

Config/testthat/edition 3

URL https://github.com/vjoshy/regMR

BugReports https://github.com/vjoshy/regMR/issues

NeedsCompilation yes

Author Cameron Bechthold [aut, cre, cph], Vinay Joshy [aut, ctb, cph], Zeny Feng [aut, ths, cph], Grace Stelter [ctb]

Maintainer Cameron Bechthold <cbechtho@uoguelph.ca>

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

Date/Publication 2026-07-31 14:10:02 UTC

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

RemoteRef HEAD

RemoteSha 7863c7c509270ca3425608b5c0ee1fed425dd090

Contents

FMRM 2
MM 5
MM_Grid 8
plot.FMRM 12
plot2 13
print.FMRM 15
summary.FMRM 16

Index **19**

FMRM *Regularized Finite Mixture Regression Model Using MM Algorithm*

Description

Applies the Majorization-Minimization Algorithm to the inputted data over all group counts from 2 to G and all lambda-alpha pairs given the specified parameters and distribution (family) to estimate a finite mixture regression model. The function chooses the model with the lowest information criteria (as specified). It can be run sequentially or in parallel. This function is for model estimation.

Usage

```
FMRM(
  x,
  y,
  G,
  family = c("gaussian", "poisson", "binomial", "gamma"),
  tol = 0.001,
  irwls_tol = 1e-08,
  max_iter = 500,
  reps = 1,
  lambda = NULL,
  lambda_max = NULL,
  n_lambda = 100,
  alpha = seq(0, 1, by = 0.1),
  verbose = TRUE,
  penalty = TRUE,
  random = FALSE,
  n_random_la = 100,
  information_criteria = c("bic", "gebic", "aic", "icl"),
  automatic_stopping = FALSE,
  parallel = FALSE,
  common_sigma = FALSE,
  sigma_penalty = TRUE,
  pi_penalty = TRUE
)
```

Arguments

x	Predictor/design matrix. A numeric matrix of size n x p where the number of rows is equal to the number of observations n, and the number of columns is equal to the number of covariates p.
y	Response vector. Either a numeric vector, or something coercible to one (i.e. matrix with one column). If family is Binomial, y becomes a numeric matrix of size n x 2, where the first column corresponds to the successes and the second the failures.
G	An integer greater than or equal to two specifying the maximum number of mixture components (groups) in the estimated model that the function will attempt to fit the data to.
family	A string of characters specifying the distribution of the finite mixture regression model being fit to the data. Parameter updates are altered depending on the inputted family. Current accepted types include Gaussian ("gaussian" or gaussian(), default value), Poisson ("poisson" or poisson()), Binomial ("binomial" or binomial()), and Gamma ("gamma" or Gamma()). Input is converted to all lowercase within the function for simplification.
tol	A non-negative numeric value specifying the stopping criterion for the MM algorithm (default value is 1e-03). If the difference in value of the objective function being minimized is within tol in two consecutive iterations, the algorithm stops.
irwls_tol	A non-negative numeric value specifying the stopping criterion for the IRWLS procedure (default value is 1e-08). If the difference in value of the beta values is within irwls_tol in two consecutive iterations, the procedure stops.
max_iter	An integer greater than or equal to one specifying the maximum number of iterations run within the MM algorithm. Default value is 500.
reps	An integer greater than or equal to one specifying the number of times the MM algorithm is repeated on the same initial parameters. Default value is 1.
lambda	A list of length G of numeric vectors containing non-negative tuning parameters specifying various strengths of the sparse group lasso (sgl) penalty to be applied. Finite mixture regression models will be estimated using each lambda value. Default value is NULL as the function will initialize a lambda vector for each group count using an algorithm.
lambda_max	A non-negative numeric value specifying the maximum lambda value (tuning parameter) used in the creation of each lambda vector. Default value is NULL as the function will initialize lambda_max for each group.
n_lambda	An integer greater than one (default value 100) specifying the length of the lambda vector for each group.
alpha	A numeric vector containing values between zero and one inclusive specifying different weights between the lasso penalty and group lasso penalty being applied. Alpha = 1 gives the lasso fit and alpha = 0 gives the group lasso fit. Default value is a numeric vector of length 11: c(0, 0.1, 0.2, ..., 1).
verbose	A logical value which, if true (default value), allows the function to print progress updates.

`penalty` A logical value which, if true (default value), allows the function to apply the sgl penalty to the regression parameter updates and objective function within iterations of the MM algorithm.

`random` A logical value which, if true (false is the default value), allows the function to take a random sample of size n_random_la from the lambda-alpha pairs and run the MM algorithm over the reduced penalty grid.

`n_random_la` A positive integer (default value 100) specifying the number of lambda-alpha pairs to be sampled when random is TRUE.

`information_criteria` A string of characters specifying the information criteria for model selection purposes. The model that minimizes the information criteria over all group counts and lambda-alpha sgl penalty combinations will be selected. Current accepted types include the default Bayesian Information Criterion (BIC) ("bic"), group-structured Extended BIC (gEBIC) ("gebic"), Akaike Information Criterion (AIC) ("aic"), and Integrated Classification Likelihood (ICL) Criterion ("icl").

`automatic_stopping` A logical value which, if true (false is the default value), allows the function to implement IC-based automatic stopping on the mixture components. When the condition for stopping is met, the function stops iterating over the group count.

`parallel` A logical value which, if true (false is the default value), allows the function to run parallel workers to increase computational speed.

`common_sigma` A logical value which, if true (false is the default value) and family = "gaussian" or gaussian(), estimates the standard deviations as equivalent across mixture components.

`sigma_penalty` A logical value which, if true (default value) and family = "gaussian" or gaussian(), allows a variance-induced penalty to be applied to the objective function being minimized within the MM algorithm.

`pi_penalty` A logical value which, if true (default value), scales the Sparse Group LASSO penalty by mixing proportion sizes.

Value

An object, depending on inputted family, of class FGMRM, FPMRM, FBMRM, or FGamMRM and FMRM containing the parameters of the estimated finite mixture regression model (parameters depend on inputted family), number of mixture components, parameters of models with the same alpha, a numeric matrix containing the alpha, lambda, and ic values of all estimated models for plotting purposes, and the function call for summary purposes.

Examples

```
set.seed(2025)

# ----Simulate data----
n <- 250   # total samples
p <- 6     # number of covariates
```

```
G <- 3     # number of mixture components
rho = 0.2  # correlation

# ----True parameters for 3 clusters----
betas <- matrix(c(
  1,  2, -1,  0.5, 0, 0, 0,  # component 1
  5, -2,  1,  0, 0, 0, 0,  # component 2
  -3, 0,  2, 0, 0, 0, 0     # component 3
), nrow = G, byrow = TRUE)
pis <- c(0.4, 0.4, 0.2)
sigmas <- c(3, 1.5, 1)/2

# ----Generate correlation matrix----
cor_mat <- outer(1:p, 1:p, function(i, j) rho^abs(i - j))
Sigma <- cor_mat

# ----Simulate design matrix X (n × p)----
X <- mvtnorm::rmvnorm(n, mean = rep(0, p), sigma = Sigma)

# ----Generate responsibilities----
z <- rmultinom(n, size = 1, prob = pis)
groups <- apply(z, 2, which.max)

# ----b0 + b1x1 + b2x2 + ... + bkxp----
mu_vec <- rowSums(cbind(1, X) * betas[groups, ])

# ----Simulate response y----
y <- rnorm(n, mean = mu_vec, sd = sigmas[groups])

# ----Fit model----
mod <- FMRM(x = X,
            y = y,
            G = 3,
            family = gaussian(),
            parallel = TRUE,
            random = TRUE,
            n_random_la = 25,
            verbose = FALSE)
```

`MM` *Majorization-Minimization Algorithm for Finite Mixture Regression Models*

Description

Applies the Majorization-Minimization Algorithm to the inputted data given the specified parameters and distribution (family) to estimate a finite mixture regression model. Initial estimates for model parameters are provided within the function, but can be specified in the function call. This function is used for model estimation.

Usage

```
MM(
  x,
  y,
  G,
  family = c("gaussian", "poisson", "binomial", "gamma"),
  tol = 0.001,
  irwls_tol = 1e-08,
  max_iter = 500,
  reps = 1,
  lambda = 0,
  alpha = 0,
  init_parameters = NULL,
  verbose = TRUE,
  penalty = TRUE,
  information_criteria = c("bic", "gebic", "aic", "icl"),
  common_sigma = FALSE,
  sigma_penalty = TRUE,
  pi_penalty = TRUE
)
```

Arguments

x	Predictor/design matrix. A numeric matrix of size n x p where the number of rows is equal to the number of observations n, and the number of columns is equal to the number of covariates p.
y	Response vector. Either a numeric vector, or something coercible to one (i.e. matrix with one column). If family is Binomial, y becomes a numeric matrix of size n x 2, where the first column corresponds to the successes and the second the failures.
G	An integer greater than or equal to one representing the number of mixture components (groups) in a finite mixture regression model.
family	A string of characters specifying the distribution of the finite mixture regression model being fit to the data. Parameter updates are altered depending on the inputted family. Current accepted types include Gaussian ("gaussian" or gaussian(), default value), Poisson ("poisson" or poisson()), Binomial ("binomial" or binomial()), and Gamma ("gamma" or Gamma()). Input is converted to all lowercase within the function for simplification.
tol	A non-negative numeric value specifying the stopping criterion for the MM algorithm (default value is 1e-03). If the difference in value of the objective function being minimized is within tol in two consecutive iterations, the algorithm stops.
irwls_tol	A non-negative numeric value specifying the stopping criterion for the IRWLS procedure (default value is 1e-08). If the difference in value of the beta values is within irwls_tol in two consecutive iterations, the procedure stops.
max_iter	An integer greater than or equal to one specifying the maximum number of iterations run within the MM algorithm. Default value is 500.

`reps` An integer greater than or equal to one specifying the number of times the MM algorithm is repeated on the same initial parameters. Default value is 1.

`lambda` A non-negative numeric value (tuning parameter) specifying the strength of the sparse group lasso (sgl) penalty. Default value is zero (no penalty applied).

`alpha` A numeric value between zero (default value) and one inclusive specifying the weight between the lasso penalty and group lasso penalty being applied. Alpha = 1 gives the lasso fit and alpha = 0 gives the group lasso fit.

`init_parameters`
A list containing the initial parameter estimates for the finite mixture regression model being fit. Default value is NULL as the function contains procedures for initialization. If Gaussian, init_parameters = list(pi, beta, sigma, z), if Poisson or Binomial, init_parameters = list(pi, beta, z), and if Gamma, init_parameters = list(pi, beta, nu, z).

`verbose` A logical value which, if true (default value), allows the function to print progress updates.

`penalty` A logical value which, if true (default value), allows the function to apply the sgl penalty to the regression parameter updates and objective function within iterations of the MM algorithm.

`information_criteria`
A string of characters specifying the information criteria for model selection purposes. The model that minimizes the information criteria will be selected. Current accepted types include the default Bayesian Information Criterion (BIC) ("bic"), group-structured Extended BIC (gEBIC) ("gebic"), Akaike Information Criterion (AIC) ("aic"), and Integrated Classification Likelihood (ICL) Criterion ("icl").

`common_sigma` A logical value which, if true (false is the default value) and family = "gaussian" or gaussian(), estimates the standard deviations as equivalent across mixture components.

`sigma_penalty` A logical value which, if true (default value) and family = "gaussian" or gaussian(), allows a variance-induced penalty to be applied to the objective function being minimized within the MM algorithm.

`pi_penalty` A logical value which, if true (default value), scales the Sparse Group LASSO penalty by mixing proportion sizes.

Value

A list containing the parameters of the estimated finite mixture regression model.

Examples

```
set.seed(2025)

# ----Simulate data----
n <- 250   # total samples
p <- 6     # number of covariates
G <- 3     # number of mixture components
```

```
rho = 0.2  # correlation

# ----True parameters for 3 clusters----
betas <- matrix(c(
  1,  2, -1,  0.5, 0, 0, 0,  # component 1
  5, -2,  1,  0, 0, 0, 0,  # component 2
  -3, 0,  2, 0, 0, 0, 0     # component 3
), nrow = G, byrow = TRUE)
pis <- c(0.4, 0.4, 0.2)
sigmas <- c(3, 1.5, 1)/2

# ----Generate correlation matrix----
cor_mat <- outer(1:p, 1:p, function(i, j) rho^abs(i - j))
Sigma <- cor_mat

# ----Simulate design matrix X (n × p)----
X <- mvtnorm::rmvnorm(n, mean = rep(0, p), sigma = Sigma)

# ----Generate responsibilities----
z <- rmultinom(n, size = 1, prob = pis)
groups <- apply(z, 2, which.max)

# ----b0 + b1x1 + b2x2 + ... + bkxp----
mu_vec <- rowSums(cbind(1, X) * betas[groups, ])

# ----Simulate response y----
y <- rnorm(n, mean = mu_vec, sd = sigmas[groups])

mod <- MM(X, y, G = 3, family = gaussian(), lambda = 5, alpha = 1)
```

`MM_Grid` *Majorization-Minimization Algorithm over Lambda-Alpha Grid for Finite Mixture Regression Models*

Description

Applies the Majorization-Minimization Algorithm to the inputted data over all lambda-alpha pairs given the specified parameters and distribution (family) to estimate a finite mixture regression model. The function chooses the model with the lowest information criteria (as specified). It can be run sequentially or in parallel. This function is used for model estimation.

Usage

```
MM_Grid(
  g,
  x,
  y,
  family = c("gaussian", "poisson", "binomial", "gamma"),
```

```
  tol = 0.001,
  irwls_tol = 1e-08,
  max_iter = 500,
  reps = 1,
  lambda = NULL,
  lambda_max = NULL,
  n_lambda = 100,
  alpha = seq(0, 1, by = 0.1),
  verbose = TRUE,
  penalty = TRUE,
  random = FALSE,
  n_random_la = 100,
  information_criteria = c("bic", "gebic", "aic", "icl"),
  parallel = FALSE,
  common_sigma = FALSE,
  sigma_penalty = TRUE,
  pi_penalty = TRUE
)
```

Arguments

g	An integer greater than or equal to one representing the number of mixture components (groups) in a finite mixture regression model.
x	Predictor/design matrix. A numeric matrix of size n x p where the number of rows is equal to the number of observations n, and the number of columns is equal to the number of covariates p.
y	Response vector. Either a numeric vector, or something coercible to one (i.e. matrix with one column). If family is Binomial, y becomes a numeric matrix of size n x 2, where the first column corresponds to the successes and the second the failures.
family	A string of characters specifying the distribution of the finite mixture regression model being fit to the data. Parameter updates are altered depending on the inputted family. Current accepted types include Gaussian ("gaussian" or gaussian(), default value), Poisson ("poisson" or poisson()), Binomial ("binomial" or binomial()), and Gamma ("gamma" or Gamma()). Input is converted to all lowercase within the function for simplification.
tol	A non-negative numeric value specifying the stopping criterion for the MM algorithm (default value is 1e-03). If the difference in value of the objective function being minimized is within tol in two consecutive iterations, the algorithm stops.
irwls_tol	A non-negative numeric value specifying the stopping criterion for the IRWLS procedure (default value is 1e-08). If the difference in value of the beta values is within irwls_tol in two consecutive iterations, the procedure stops.
max_iter	An integer greater than or equal to one specifying the maximum number of iterations run within the MM algorithm. Default value is 500.
reps	An integer greater than or equal to one specifying the number of times the MM algorithm is repeated on the same initial parameters. Default value is 1.

`lambda` A list of length G of numeric vectors containing non-negative tuning parameters specifying various strengths of the sparse group lasso (sgl) penalty to be applied. Finite mixture regression models will be estimated using each lambda value. Default value is NULL as the function will initialize a lambda vector for each group count using an algorithm.

`lambda_max` A non-negative numeric value specifying the maximum lambda value (tuning parameter) used in the creation of each lambda vector. Default value is NULL as the function will initialize lambda_max for each group.

`n_lambda` An integer greater than one (default value 100) specifying the length of the lambda vector for each group.

`alpha` A numeric vector containing values between zero and one inclusive specifying different weights between the lasso penalty and group lasso penalty being applied. Alpha = 1 gives the lasso fit and alpha = 0 gives the group lasso fit. Default value is a numeric vector of length 11: c(0, 0.1, 0.2, ..., 1).

`verbose` A logical value which, if true (default value), allows the function to print progress updates.

`penalty` A logical value which, if true (default value), allows the function to apply the sgl penalty to the regression parameter updates and objective function within iterations of the MM algorithm.

`random` A logical value which, if true (false is the default value), allows the function to take a random sample of size n_random_la from the lambda-alpha pairs and run the MM algorithm over the reduced penalty grid.

`n_random_la` A positive integer (default value 100) specifying the number of lambda-alpha pairs to be sampled when random is TRUE.

`information_criteria` A string of characters specifying the information criteria for model selection purposes. The model that minimizes the information criteria over all lambda-alpha sgl penalty combinations will be selected. Current accepted types include the default Bayesian Information Criterion (BIC) ("bic"), group-structured Extended BIC (gEBIC) ("gebic"), Akaike Information Criterion (AIC) ("aic"), and Integrated Classification Likelihood (ICL) Criterion ("icl").

`parallel` A logical value which, if true (false is the default value), allows the function to run parallel workers to increase computational speed.

`common_sigma` A logical value which, if true (false is the default value) and family = "gaussian" or gaussian(), estimates the standard deviations as equivalent across mixture components.

`sigma_penalty` A logical value which, if true (default value) and family = "gaussian" or gaussian(), allows a variance-induced penalty to be applied to the objective function being minimized within the MM algorithm.

`pi_penalty` A logical value which, if true (default value), scales the Sparse Group LASSO penalty by mixing proportion sizes.

Value

An object, depending on inputted family, of class FGMRM, FPMRM, FBMRM, or FGamMRM and FMRM containing the parameters of the estimated finite mixture regression model (parameters

depend on inputted family), number of mixture components, parameters of models with the same alpha, a numeric matrix containing the alpha, lambda, and ic values of all estimated models for plotting purposes, and the function call for summary purposes.

Examples

```
set.seed(2025)

# ----Simulate data----
n <- 250   # total samples
p <- 6     # number of covariates
G <- 3     # number of mixture components
rho = 0.2  # correlation

# ----True parameters for 3 clusters----
betas <- matrix(c(
  1,  2, -1,  0.5, 0, 0, 0,  # component 1
  5, -2,  1,  0, 0, 0, 0,  # component 2
  -3, 0,  2, 0, 0, 0, 0     # component 3
), nrow = G, byrow = TRUE)
pis <- c(0.4, 0.4, 0.2)
sigmas <- c(3, 1.5, 1)/2

# ----Generate correlation matrix----
cor_mat <- outer(1:p, 1:p, function(i, j) rho^abs(i - j))
Sigma <- cor_mat

# ----Simulate design matrix X (n × p)----
X <- mvtnorm::rmvnorm(n, mean = rep(0, p), sigma = Sigma)

# ----Generate responsibilities----
z <- rmultinom(n, size = 1, prob = pis)
groups <- apply(z, 2, which.max)

# ----b0 + b1x1 + b2x2 + ... + bkxp----
mu_vec <- rowSums(cbind(1, X) * betas[groups, ])

# ----Simulate response y----
y <- rnorm(n, mean = mu_vec, sd = sigmas[groups])

mod <- MM_Grid(g = 3,
               X,
               y,
               family = gaussian(),
               parallel = TRUE,
               random = TRUE,
               verbose = FALSE)
```

plot.FMRM *Plot Method for a Finite Mixture Regression Model of Class FMRM*

Description

This function creates plots for finite mixture regression models of class FMRM. It generates three plots: lambdas vs. ics for all alpha values, lambdas vs. regression coefficients, and lambdas vs. group norms for all models with the same alpha as the optimal alpha.

Usage

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

Arguments

x	An object of class FMRM, the result of calling FMRM() or MM_Grid().
...	Additional arguments for plotting (currently unused).

Value

A list of three ggplot objects: lambdas vs. ics for all alpha values, lambdas vs. regression coefficients, and lambdas vs. group norms for all models with the same alpha as the optimal alpha.

Examples

```
set.seed(2025)

# ----Simulate data----
n <- 250   # total samples
p <- 6     # number of covariates
G <- 3     # number of mixture components
rho = 0.2  # correlation

# ----True parameters for 3 clusters----
betas <- matrix(c(
  1,  2, -1,  0.5, 0, 0, 0,  # component 1
  5, -2,  1,  0, 0, 0, 0,  # component 2
  -3, 0,  2, 0, 0, 0, 0     # component 3
), nrow = G, byrow = TRUE)
pis <- c(0.4, 0.4, 0.2)
sigmas <- c(3, 1.5, 1)/2

# ----Generate correlation matrix----
cor_mat <- outer(1:p, 1:p, function(i, j) rho^abs(i - j))
Sigma <- cor_mat

# ----Simulate design matrix X (n × p)----
```

```
X <- mvtnorm::rmvnorm(n, mean = rep(0, p), sigma = Sigma)

# ----Generate responsibilities----
z <- rmultinom(n, size = 1, prob = pis)
groups <- apply(z, 2, which.max)

# ----b0 + b1x1 + b2x2 + ... + bkxp----
mu_vec <- rowSums(cbind(1, X) * betas[groups, ])

# ----Simulate response y----
y <- rnorm(n, mean = mu_vec, sd = sigmas[groups])

# ----Fit model----
mod <- FMRM(x = X,
            y = y,
            G = 3,
            family = gaussian(),
            parallel = TRUE,
            random = TRUE,
            n_random_la = 25,
            verbose = FALSE)

# ----Call plot----
plots <- plot(mod)

# ----Display plots----
plots[[1]] # ----lambdas vs. ics----
plots[[2]] # ----lambdas vs. regression coefficients----
plots[[3]] # ----lambdas vs. group norms----
```

`plot2` *Plot Covariates of X Against Response With Group Assignments*

Description

This function creates a 3-D plot for finite mixture regression models of class FMRM. It plots the specified covariates of x against a response (y or y_hat), with the group assignments highlighted in colour.

Usage

```
plot2(mod, x, y, covariate_one, covariate_two, ...)
```

Arguments

`mod`	An object of class FMRM, the result of calling FMRM() or MM_Grid().
`x`	Predictor/design matrix. A numeric matrix of size n x p, where the number of rows is equal to the number of observations n, and the number of columns is equal to the number of covariates p.

y	Response vector. Either a numeric vector, or something coercible to one. Can either be y or y_hat, the expected predicted responses. The latter is accessed through mod$parameters$y_hat.
covariate_one	A numeric value specifying the first covariate of x to be plotted.
covariate_two	A numeric value specifying the second covariate of x to be plotted.
...	Additional arguments for plotting (currently unused).

Value

A 3-D plotly plot of the specified covariates of x against a response, with the group assignments highlighted in colour.

Examples

```
set.seed(2025)

# ----Simulate data----
n <- 250   # total samples
p <- 6     # number of covariates
G <- 3     # number of mixture components
rho = 0.2  # correlation

# ----True parameters for 3 clusters----
betas <- matrix(c(
 1,  2, -1,  0.5, 0, 0, 0,  # component 1
 5, -2,  1,  0, 0, 0, 0,  # component 2
 -3, 0,  2, 0, 0, 0, 0     # component 3
), nrow = G, byrow = TRUE) / 2
pis <- c(0.4, 0.4, 0.2)
sigmas <- c(0.5, 0.4, 0.3)

# ----Generate correlation matrix----
cor_mat <- outer(1:p, 1:p, function(i, j) rho^abs(i - j))
Sigma <- cor_mat

# ----Simulate design matrix X (n × p)----
X <- mvtnorm::rmvnorm(n, mean = rep(0, p), sigma = Sigma)

# ----Generate responsibilities----
z <- rmultinom(n, size = 1, prob = pis)
groups <- apply(z, 2, which.max)

# ----b0 + b1x1 + b2x2 + ... + bkxp----
mu_vec <- rowSums(cbind(1, X) * betas[groups, ])

# ----Simulate response y----
y <- rnorm(n, mean = mu_vec, sd = sigmas[groups])

# ----Fit model----
mod <- FMRM(x = X,
```

```
            y = y,
            G = 3,
            family = gaussian(),
            parallel = TRUE,
            random = TRUE,
            n_random_la = 25,
            verbose = FALSE)

# ----Call plot2----
plot <- plot2(mod, X, y, 1, 2)

# ----Display plot----
plot
```

`print.FMRM` *Print Method for a Finite Mixture Regression Model of Class FMRM*

Description

This function prints the elements of finite mixture regression models of class FMRM. It displays the number of mixture components, optimal lambda-alpha, log-likelihood, information criteria, mean-squared-error, and parameters of the model.

Usage

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

Arguments

`x`	An object of class FMRM, the result of calling FMRM() or MM_Grid().
`...`	Additional arguments for printing (currently unused).

Value

No return value, called for side effects.

Examples

```
set.seed(2025)

# ----Simulate data----
n <- 250   # total samples
p <- 6     # number of covariates
G <- 3     # number of mixture components
rho = 0.2  # correlation
```

```
# ----True parameters for 3 clusters----
betas <- matrix(c(
  1,  2, -1,  0.5, 0, 0, 0,  # component 1
  5, -2,  1,  0, 0, 0, 0,  # component 2
  -3, 0,  2, 0, 0, 0, 0     # component 3
), nrow = G, byrow = TRUE)
pis <- c(0.4, 0.4, 0.2)
sigmas <- c(3, 1.5, 1)/2

# ----Generate correlation matrix----
cor_mat <- outer(1:p, 1:p, function(i, j) rho^abs(i - j))
Sigma <- cor_mat

# ----Simulate design matrix X (n × p)----
X <- mvtnorm::rmvnorm(n, mean = rep(0, p), sigma = Sigma)

# ----Generate responsibilities----
z <- rmultinom(n, size = 1, prob = pis)
groups <- apply(z, 2, which.max)

# ----b0 + b1x1 + b2x2 + ... + bkxp----
mu_vec <- rowSums(cbind(1, X) * betas[groups, ])

# ----Simulate response y----
y <- rnorm(n, mean = mu_vec, sd = sigmas[groups])

# ----Fit model----
mod <- FMRM(x = X,
            y = y,
            G = 3,
            family = gaussian(),
            parallel = TRUE,
            random = TRUE,
            n_random_la = 25,
            verbose = FALSE)

# ----Print model----
print(mod)
```

`summary.FMRM` *Summary Method for a Finite Mixture Regression Model of Class FMRM*

Description

This function displays a summary for finite mixture regression models of class FMRM. It displays the number of mixture components, optimal lambda-alpha, log-likelihood, information criteria, mean-squared-error, and parameters of the model.

Usage

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

Arguments

object	An object of class FMRM, the result of calling FMRM() or MM_Grid().
...	Additional arguments for summary (currently unused).

Value

No return value, called for side effects.

Examples

```
set.seed(2025)

# ----Simulate data----
n <- 250   # total samples
p <- 6     # number of covariates
G <- 3     # number of mixture components
rho = 0.2  # correlation

# ----True parameters for 3 clusters----
betas <- matrix(c(
  1,  2, -1,  0.5, 0, 0, 0,  # component 1
  5, -2,  1,  0, 0, 0, 0,  # component 2
  -3, 0,  2, 0, 0, 0, 0     # component 3
), nrow = G, byrow = TRUE)
pis <- c(0.4, 0.4, 0.2)
sigmas <- c(3, 1.5, 1)/2

# ----Generate correlation matrix----
cor_mat <- outer(1:p, 1:p, function(i, j) rho^abs(i - j))
Sigma <- cor_mat

# ----Simulate design matrix X (n × p)----
X <- mvtnorm::rmvnorm(n, mean = rep(0, p), sigma = Sigma)

# ----Generate responsibilities----
z <- rmultinom(n, size = 1, prob = pis)
groups <- apply(z, 2, which.max)

# ----b0 + b1x1 + b2x2 + ... + bkxp----
mu_vec <- rowSums(cbind(1, X) * betas[groups, ])

# ----Simulate response y----
y <- rnorm(n, mean = mu_vec, sd = sigmas[groups])

# ----Fit model----
```

```
mod <- FMRM(x = X,
            y = y,
            G = 3,
            family = gaussian(),
            parallel = TRUE,
            random = TRUE,
            n_random_la = 25,
            verbose = FALSE)

# ----Display summary----
summary(mod)
```

Index

FMRM, 2

MM, 5
MM_Grid, 8

plot.FMRM, 12
plot2, 13
print.FMRM, 15

summary.FMRM, 16