# Package 'SSLFMM'

November 20, 2025

**Type** Package

**Title** Semisupervised Learning under a Mixed Missingness Mechanism in Finite Mixture Models

**Version** 0.1.0

**Maintainer** Jinran Wu <jinran.wu@uq.edu.au>

**Description** Implements a semi-supervised learning framework for finite mixture models under a mixed label-missingness mechanism. The approach models both missing completely at random (MCAR) and entropy-based missing at random (MAR) processes using a logistic–entropy formulation. Estimation is carried out via an Expectation–Conditional Maximisation (ECM) algorithm with robust initialisation routines for stable convergence. The methodology relates to the statistical perspective and informative missingness behaviour discussed in Ahfock and McLachlan (2020) <doi:10.1007/s11222-020-09971-5> and Ahfock and McLachlan (2023) <doi:10.1016/j.ecosta.2022.03.007>. The package provides functions for data simulation, model estimation, prediction, and theoretical Bayes error evaluation for analysing partially labelled data under realistic mixed missingness mechanisms.

**License** GPL-3

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**Depends** R (>= 4.2.0)

**Imports** stats, mvtnorm, matrixStats

## Contents

bayesclassifier . . . . . 2
compute_d2 . . . . . 3
EM_FMM_SemiSupervised . . . . . 3
EM_FMM_SemiSupervised_Complete_Initial . . . . . 5
EM_FMM_SemiSupervised_Initial . . . . . 6
error_beta_classification . . . . . 7
get_clusterprobs . . . . . 8
get_entropy . . . . . 9
initialestimate . . . . . 10
logsumexp . . . . . 11
neg_loglik . . . . . 12

normalise_logprob . . . . . 13
pack_theta . . . . . 14
rmix . . . . . 14
simulate_mixed_missingness . . . . . 15
unpack_theta . . . . . 16

**Index** **17**

---

`bayesclassifier` *Bayes' Rule Classifier*

---

## Description

Classifier specified by Bayes' rule. Assigns $\arg\max_k\{\log(\pi_k) + \log \mathcal{N}_p(x_i \mid \mu_k, \Sigma_k)\}$.

## Usage

```
bayesclassifier(dat, p, g, pi = NULL, mu = NULL, sigma = NULL, paralist = NULL)
```

## Arguments

| | |
|---|---|
| `dat` | An $n \times p$ numeric matrix (or a length-$p$ numeric vector, treated as $1 \times p$). |
| `p` | Integer; dimension of the observation vector. |
| `g` | Integer; number of Gaussian components/classes. |
| `pi` | Numeric length-$g$ vector of mixing proportions (must sum to 1). |
| `mu` | Either:<br>• A $p \times g$ numeric matrix (column $k$ is $\mu_k$), or<br>• A length-$g$ list of length-$p$ numeric vectors. |
| `sigma` | Either:<br>• A $p \times p$ covariance matrix (shared), or<br>• A length-$g$ list of $p \times p$ covariance matrices, or<br>• A $p \times p \times g$ numeric array. |
| `paralist` | Optional list with elements `pi`, `mu`, `sigma` (overrides explicit args if provided and non-`NULL`). |

## Value

An integer vector of length $n$ with predicted class labels in $1 : g$.

## Examples

```
# Minimal example with list-style mu and sigma:
set.seed(1)
p <- 2; g <- 2
pi <- c(0.5, 0.5)
mu <- list(`1`=c(0.96, 0.02), `2`=c(-1.02, -0.03))
sigma <- list(
  matrix(c(0.9417379,0.5447264, 0.5447264,0.9811853), 2, 2, byrow=TRUE),
  matrix(c(0.9984812,0.3314474, 0.3314474,1.1316865), 2, 2, byrow=TRUE)
)
X <- mvtnorm::rmvnorm(5, mean = mu[[1]], sigma = sigma[[1]])
bayesclassifier(X, p=p, g=g, pi=pi, mu=mu, sigma=sigma)
```

## compute_d2 *Squared Discriminant Score for Two-Group LDA (Equal Covariance)*

### Description

Computes $d^2$ where $d = \beta_0 + y^\top \beta_1$ for two classes under the common-covariance (LDA) model:

$$\beta_1 = \Sigma^{-1}(\mu_1 - \mu_2), \quad \beta_0 = \log(\pi_1/\pi_2) - \tfrac{1}{2}(\mu_1 + \mu_2)^\top \Sigma^{-1}(\mu_1 - \mu_2).$$

### Usage

```
compute_d2(y, mu1, mu2, Sigma_inv, pi1, pi2)
```

### Arguments

| | |
|---|---|
| y | Numeric vector (length $p$) or numeric matrix with $p$ columns; rows are observations. |
| mu1, mu2 | Numeric vectors of length $p$: class means. |
| Sigma_inv | $p \times p$ numeric precision matrix (inverse covariance). |
| pi1, pi2 | Positive scalars: class prior probabilities (need not sum to 1). |

### Value

If y is a vector, a single numeric $d^2$. If y is a matrix, a numeric vector of $d^2$ values for each row.

### Examples

```
set.seed(1)
mu1 <- c(0, 0); mu2 <- c(1, 1)
S <- matrix(c(1, .2, .2, 1), 2, 2)
Sigma_inv <- solve(S)
x <- c(0.5, -0.2)
compute_d2(x, mu1, mu2, Sigma_inv, pi1 = 0.6, pi2 = 0.4)

X <- matrix(rnorm(10 * 2), ncol = 2)
compute_d2(X, mu1, mu2, Sigma_inv, 0.5, 0.5)
```

## EM_FMM_SemiSupervised *EM for Semi-Supervised FMM with Mixed Missingness (MCAR + entropy-based MAR)*

### Description

Runs an EM-like procedure that models a mixed missingness mechanism: unlabeled indicator $m_j$ follows a mixture of MCAR (prob $\alpha$) and entropy-driven MAR via a logistic link $q_j = \text{logit}^{-1}(\xi_0 + \xi_1 \log e_j)$. Supports shared (ncov = 1) or class-specific (ncov = 2) covariance.

## Usage

```
EM_FMM_SemiSupervised(
  data,
  g = 2,
  init_res,
  max_iter = 5,
  tol = 1e-06,
  ncov = 1
)
```

## Arguments

| | |
|---|---|
| data | A data.frame or matrix with $p + 2$ columns: first $p$ are features, then missing (0=labelled, 1=unlabelled), and obers (class label for labelled rows; ignored otherwise). |
| g | Integer, number of mixture components (classes). |
| init_res | A list with initial parameters:<br>• pi: numeric length-g (mixture weights, sum to 1)<br>• mu: list of length g, each length-p mean vector<br>• Sigma: if ncov=1, a p x p matrix; if ncov=2, a list of g p x p matrices<br>• alpha: scalar in (0,1)<br>• xi: numeric length-2, logistic coefficients (xi0, xi1) |
| max_iter | Integer, max EM iterations. |
| tol | Convergence tolerance on log-likelihood increase. |
| ncov | Integer covariance structure: 1 = shared/equal, 2 = class-specific/unequal. |

## Details

This function expects the following helpers in scope:

- pack_theta(pi_k, mu_k, Sigma_k, g, p, ncov)
- unpack_theta(theta, g, p, ncov)
- neg_loglik(theta, Y_all, m_j, Z_all, d2_yj, xi, alpha_k, unpacker)
- get_entropy(dat, n, p, g, paralist) returning per-observation entropy-like values

## Value

A list with elements: pi, mu, Sigma, xi, alpha, loglik, d2_yj, m1j_k, m2j_k, and ncov.

## Examples

```
# Requires definitions of pack_theta, unpack_theta, neg_loglik, get_entropy.
# EM_FMM_SemiSupervised(data, g = 2, init_res = init, ncov = 1)
```

`EM_FMM_SemiSupervised_Complete_Initial` *Complete-data warm-up initialization for semi-supervised FMM with mixtures of label missingness mechanisms*

## Description

Uses both labeled and unlabeled subsets of the data to obtain quick initial estimates for mixture parameters and missingness mechanism parameters (`alpha`, `xi`) via a warm-up EM procedure.

## Usage

```
EM_FMM_SemiSupervised_Complete_Initial(
  data,
  g = 2,
  ncov = 1,
  alpha_init = 0.01,
  warm_up_iter = 200,
  tol = 1e-06
)
```

## Arguments

| | |
|---|---|
| `data` | A data frame containing:<br>• The first p columns: numeric variables used in the FMM.<br>• A column `missing`: indicator (0 = labeled, 1 = unlabeled/missing).<br>• A column `obers`: class labels for labeled rows (`1:g`); `NA` for unlabeled. |
| `g` | Integer, number of mixture components (default `2`). |
| `ncov` | Integer, covariance structure: `1` = shared (equal), `2` = class-specific (unequal). |
| `alpha_init` | Numeric in (0,1), initial MCAR proportion (default `0.01`). |
| `warm_up_iter` | Integer, number of warm-up EM iterations (default `200`). |
| `tol` | Convergence tolerance on `alpha` (default `1e-6`). |

## Details

- This function first calls `initialestimate` to get initial $\pi$, $\mu$, $\Sigma$.
- Then it calls `EM_FMM_SemiSupervised_Initial` with these values for a short warm-up run.
- Covariance structure (`equal` vs. `unequal`) is determined by `ncov`.

## Value

A list with initial values from `EM_FMM_SemiSupervised_Initial`:

- `pi` - mixture weights.
- `mu` - list of component mean vectors.
- `Sigma` - covariance matrix/matrices.
- `alpha` - MCAR proportion.
- `xi` - logistic regression coefficients for MAR mechanism.

## Examples

```
## Not run:
# Suppose df contains p features, a 'missing' column, and 'obers' labels
res_init <- EM_FMM_SemiSupervised_Complete_Initial(df, g = 3, ncov = 2)

## End(Not run)
```

---

`EM_FMM_SemiSupervised_Initial` *Quick initializer for alpha, xi, and mixture parameters*

---

## Description

Provides rough initial estimates of the missingness parameters `alpha` and `xi`, together with mixture parameters `pi`, `mu`, and covariance matrices, using a lightweight EM-style routine. The covariance structure is chosen automatically based on `Sigma_init`:

- If `Sigma_init` is a $p \times p$ matrix, a **shared (equal)** covariance is used.
- If `Sigma_init` is a list of length `g` of $p \times p$ matrices or a $p \times p \times g$ array, **class-specific (unequal)** covariances are used.
- If `Sigma_init` is `NULL`, a shared covariance is estimated from the labeled data.

This function is intended as a fast, heuristic initializer rather than a final estimator for the mixed missingness model.

## Usage

```
EM_FMM_SemiSupervised_Initial(
  Y_labelled,
  Z_labelled,
  Y_unlabelled,
  g = 2,
  pi_init = NULL,
  mu_init = NULL,
  Sigma_init = NULL,
  alpha_init = 0.01,
  warm_up_iter = 50,
  tol = 1e-06
)
```

## Arguments

| | |
|---|---|
| `Y_labelled` | Numeric matrix of labeled observations ($n_L \times p$). |
| `Z_labelled` | Integer vector of class labels in `1:g` for `Y_labelled`. |
| `Y_unlabelled` | Numeric matrix of unlabeled observations ($n_U \times p$). |
| `g` | Integer, number of mixture components (default 2). |
| `pi_init` | Optional numeric length-g vector of initial mixing proportions. |
| `mu_init` | Optional list of length g of initial mean vectors (each of length p). |

| | |
|---|---|
| Sigma_init | Optional initial covariance: a $p \times p$ matrix (shared), or a list of g $p \times p$ matrices, or a $p \times p \times g$ array (class-specific). |
| alpha_init | Numeric in $(0, 1)$, initial MCAR proportion (default 0.01). |
| warm_up_iter | Integer, number of warm-up EM iterations used to refine the quick initial estimates (default 50). |
| tol | Convergence tolerance on alpha (default 1e-6). |

## Value

A list with elements:

- pi - length-g vector of mixing proportions.
- mu - list of g mean vectors.
- Sigma - shared $p \times p$ matrix (equal-Sigma) or list of g matrices (unequal-Sigma).
- xi - length-2 numeric vector c(xi0, xi1) from the logistic MAR model.
- alpha - estimated MCAR proportion.
- gamma - $n \times g$ responsibility matrix.
- d2_yj - numeric vector of entropy-based scores used in the missingness model.

## Note

This is a heuristic warm-up routine. It requires helper functions normalise_logprob() and get_entropy() to be available in the package namespace.

---

# error_beta_classification

*Compute Theoretical Bayes Error for a Binary Gaussian Mixture*

---

## Description

Computes the Bayes classification error rate for a two-component Gaussian mixture given mu_hat, Sigma_hat, and pi_hat. If mu_hat is supplied as a list of length 2, it is converted to a $p \times 2$ matrix internally.

## Usage

```
error_beta_classification(mu_hat, Sigma_hat, pi_hat)
```

## Arguments

| | |
|---|---|
| mu_hat | Either a numeric matrix of size $p \times 2$ whose columns are component means, or a list of two numeric vectors. |
| Sigma_hat | Numeric $p \times p$ covariance matrix shared across components. |
| pi_hat | Numeric vector of length 2 with mixing proportions $(\pi_1, \pi_2)$ that are non-negative and sum to 1. |

### Details

The linear discriminant is

$$\beta_1 = \Sigma^{-1}(\mu_1 - \mu_2), \qquad \beta_0 = -\frac{1}{2}(\mu_1 + \mu_2)^\top \Sigma^{-1}(\mu_1 - \mu_2) + \log(\pi_1/\pi_2)$$

and the Bayes error is

$$\mathrm{Err} = \sum_{k=1}^{2} \pi_k \, \Phi\left( \frac{(-1)^k \{\beta_0 + \beta_1^\top \mu_k\}}{\|\beta_1\|} \right),$$

where $\Phi$ is the standard normal cdf.

### Value

A numeric scalar giving the theoretical Bayes classification error rate.

### Examples

```
mu_hat <- matrix(c(1, 0, -1, 0), nrow = 2)  # columns are mu1, mu2
Sigma_hat <- diag(2)
pi_hat <- c(0.5, 0.5)
error_beta_classification(mu_hat, Sigma_hat, pi_hat)
```

---

## get_clusterprobs *Posterior cluster probabilities for a Gaussian mixture*

---

### Description

Compute posterior membership probabilities $\tau_{ik}$ for each observation under a Gaussian mixture with either a shared covariance matrix or component-specific covariances.

### Usage

```
get_clusterprobs(
  dat,
  n,
  p,
  g,
  pi = NULL,
  mu = NULL,
  sigma = NULL,
  paralist = NULL
)
```

### Arguments

| | |
|---|---|
| dat | Numeric matrix $n \times p$. Data matrix (rows = observations). |
| n | Integer. Number of observations (checked against `nrow(dat)`). |
| p | Integer. Number of variables (checked against `ncol(dat)`). |
| g | Integer. Number of components. |

| | |
|---|---|
| pi | Optional numeric vector length $g$. Mixing proportions (sum to 1). |
| mu | Optional numeric matrix $p \times g$. Column $j$ is mean of component $j$. |
| sigma | Optional numeric matrix $p \times p$ (shared covariance) or array $p \times p \times g$ (component-specific covariances). |
| paralist | Optional list with elements pi, mu, sigma. If provided, these take precedence over the corresponding explicit args. |

## Details

Uses a stable log-sum-exp normalization:

$$\tau_{ik} = \exp(\ell_{ik} - \mathrm{LSE}_i)$$

where $\ell_{ik} = \log p(x_i \mid k) + \log \pi_k$ and $\mathrm{LSE}_i = \log \sum_{k=1}^{g} \exp(\ell_{ik})$.

## Value

Numeric matrix $n \times g$ of posterior probabilities $\tau_{ik}$.

## Examples

```
## Not run:
  n <- 100; p <- 2; g <- 2
  X <- matrix(rnorm(n*p), n, p)
  pi <- c(0.6, 0.4)
  mu <- cbind(c(0,0), c(1,1))
  Sig <- array(0, dim = c(p,p,g)); Sig[,,1] <- diag(p); Sig[,,2] <- diag(p)
  tau <- get_clusterprobs(X, n, p, g, pi = pi, mu = mu, sigma = Sig)
  head(tau)

## End(Not run)
```

---

# get_entropy *Per-row entropy of posterior cluster probabilities*

---

## Description

Compute the Shannon entropy (in nats) of posterior membership probabilities for each observation, given a Gaussian mixture model. Posterior probabilities $\tau_{ik}$ are obtained via `get_clusterprobs()`.

## Usage

```
get_entropy(dat, n, p, g, pi = NULL, mu = NULL, sigma = NULL, paralist = NULL)
```

## Arguments

| | |
|---|---|
| dat | Numeric matrix $n \times p$. The data matrix. |
| n | Integer. Number of rows in dat. |
| p | Integer. Number of columns (features) in dat. |
| g | Integer. Number of mixture components. |

| | |
|---|---|
| pi | Optional numeric vector length $g$. Mixing proportions (sum to 1). |
| mu | Optional numeric matrix $p \times g$. Column $j$ is the mean of component $j$. |
| sigma | Optional numeric matrix $p \times p$ (shared covariance) or array $p \times p \times g$ (component-specific covariances). |
| paralist | Optional list with elements pi, mu, sigma. If supplied, these take precedence over the corresponding explicit arguments. |

## Details

The entropy for observation $i$ is

$$H_i = -\sum_{k=1}^{g} \tau_{ik} \log(\tau_{ik}),$$

where $\tau_{ik}$ are the posterior probabilities returned by `get_clusterprobs()`. Zeros are handled safely in the log via a small lower bound to maintain numerical stability (equivalent to treating $0 \log 0 = 0$).

## Value

Numeric vector of length $n$, the entropy per observation (in nats).

## See Also

get_clusterprobs

## Examples

```
## Not run:
  # Suppose you have get_clusterprobs(), and a fitted/pi, mu, sigma:
  n <- 100; p <- 2; g <- 2
  X <- matrix(rnorm(n * p), n, p)
  pi <- c(0.6, 0.4)
  mu <- cbind(c(0,0), c(1,1))
  Sigma <- array(0, dim = c(p, p, g))
  Sigma[,,1] <- diag(p); Sigma[,,2] <- diag(p)
  ent <- get_entropy(dat = X, n = n, p = p, g = g, pi = pi, mu = mu, sigma = Sigma)
  summary(ent)

## End(Not run)
```

---

# initialestimate *Initialize Parameters for a FMM from Labeled Subset*

---

## Description

Builds initial estimates $(\pi, \mu, \Sigma)$ for a g-component Gaussian mixture using only rows with observed labels in zm. Supports either a shared covariance (ncov = 1) or class-specific covariances (ncov = 2).

## Usage

```
initialestimate(dat, zm, g, ncov = 2, ridge = 1e-06)
```

### Arguments

| | |
|---|---|
| dat | A numeric matrix or data frame of features (n x p). |
| zm | Integer vector of length n with class labels in 1:g; use NA for unlabeled rows. Only labeled rows contribute to the initialization. |
| g | Integer, number of mixture components. |
| ncov | Integer, 1 for a shared covariance matrix, 2 for class-specific covariance matrices. Default 2. |
| ridge | Numeric, small diagonal ridge added to covariance(s) for numerical stability. Default 1e-6. |

### Details

If a class has zero or one labeled sample, its covariance is set to the global empirical covariance (from labeled data) with a small ridge. Class means for empty classes default to the global mean with a small jitter.

### Value

A list with

- pi: length-g vector of mixing proportions (summing to 1).
- mu: p x g matrix of class means (column i is $\mu_i$).
- sigma: if ncov = 1, a p x p shared covariance matrix; if ncov = 2, a p x p x g array of class-specific covariances.

### Examples

```
set.seed(1)
n <- 50; p <- 3; g <- 2
X <- matrix(rnorm(n*p), n, p)
z <- sample(c(1:g, NA), n, replace = TRUE, prob = c(0.4, 0.4, 0.2))
init <- initialestimate(X, z, g, ncov = 2)
str(init)
```

---

## logsumexp *Numerically Stable Log-Sum-Exp*

---

### Description

Computes $\log(\sum_i \exp(x_i))$ in a numerically stable way by subtracting the maximum value before exponentiation.

### Usage

```
logsumexp(x)
```

### Arguments

| | |
|---|---|
| x | A numeric vector. |

## Value

A single numeric value: the log-sum-exp of `x`.

## Examples

```
logsumexp(c(1000, 1001, 1002))
```

---

`neg_loglik` *Negative Log-Likelihood for Semi-supervised FMM with Mixtures of Label-Missingness Mechanisms*

---

## Description

Computes the negative log-likelihood for a semi-supervised Gaussian mixture model under a hybrid missing-data mechanism (MCAR + entropy-based MAR). Assumes a ** covariance matrix** $\Sigma$ across all mixture components.

## Usage

```
neg_loglik(theta, Y, m_j, Z, d2_yj, xi, alpha_k, unpack_fn)
```

## Arguments

| | |
|---|---|
| `theta` | Numeric vector of packed model parameters to be unpacked by `unpack_fn`. |
| `Y` | Numeric matrix of observations (n x p). |
| `m_j` | Integer or logical vector of length n indicating missingness: 0 for observed (labeled block), 1 for unlabeled/missingness block. |
| `Z` | Integer vector of length n with class labels for labeled samples (1..g); use `NA` for unlabeled rows. |
| `d2_yj` | Numeric vector of length n with the entropy-like score used in the MAR mechanism (e.g., posterior entropy or any scalar proxy). |
| `xi` | Numeric length-2 vector `c(xi0, xi1)` for the logistic MAR model $q_j = logistic(xi0 + xi1 * d2_yj)$. |
| `alpha_k` | Numeric scalar in (0,1), the MCAR mixing proportion in the missingness mechanism. |
| `unpack_fn` | Function that takes `theta` and returns a list with elements:<br>`pi` Numeric vector of length g with mixture weights.<br>`mu` List of length g; each element is a numeric mean vector (length p).<br>`sigma` Shared covariance matrix (p x p). |

## Details

The total log-likelihood is composed of three parts:

1. Labeled samples ($m_j = 0$) with observed class labels $Z_j$.
2. Unlabeled samples attributed to MCAR with probability mass $m_{1j}$.
3. Unlabeled samples attributed to MAR with probability mass $m_{2j}$.

The MAR probability for each sample is $q_j = \text{logistic}(xi0 + xi1\, d2\_yj)$. Internally, the function uses a numerically stable `logSumExp`.

## Value

A single numeric value: the negative log-likelihood.

## Note

This implementation is for the **equal covariance** case (shared $\Sigma$).

## Examples

```
## Not run:
# Minimal example (illustrative only):
library(mvtnorm)
set.seed(1)
n <- 20; p <- 2; g <- 2
Y <- matrix(rnorm(n*p), n, p)
Z <- sample(c(1:g, rep(NA, n - g)), n, replace = TRUE)
m_j <- ifelse(is.na(Z), 1L, 0L)
d2_yj <- runif(n)
xi <- c(-1, 2)
alpha_k <- 0.4
unpack_fn <- function(theta) {
  list(pi = c(0.6, 0.4),
       mu = list(c(0,0), c(1,1)),
       sigma = diag(p))
}
theta <- numeric(1) # not used in this toy unpack_fn
neg_loglik(theta, Y, m_j, Z, d2_yj, xi, alpha_k, unpack_fn)

## End(Not run)
```

---

# normalise_logprob *Normalise Log-Probabilities*

---

## Description

Converts log-probabilities into a probability distribution by exponentiating in a numerically stable way.

## Usage

```
normalise_logprob(log_probs)
```

## Arguments

| | |
|---|---|
| log_probs | A numeric vector of log-probabilities. |

## Value

A numeric vector summing to 1 (the normalised probabilities).

## Examples

```
lp <- c(-1000, -999, -998)
normalise_logprob(lp)
```

---

`pack_theta` *Pack FMM Parameters into a Vector*

---

## Description

Packs mixture weights, means, and covariance(s) into a single numeric vector. Uses the last component as the baseline for mixture weights (g-1 logits stored).

## Usage

```
pack_theta(pi_k, mu_k, Sigma, g, p, ncov = 1)
```

## Arguments

| | |
|---|---|
| `pi_k` | Numeric vector of length g with mixture weights (positive, sum to 1). |
| `mu_k` | List of length g; each element a numeric vector of length p (component means). |
| `Sigma` | Covariance: if `ncov = 1`, a single p x p matrix; if `ncov = 2`, a list of g p x p matrices. |
| `g` | Integer: number of components. |
| `p` | Integer: dimension. |
| `ncov` | Integer: covariance structure; 1 for shared covariance, 2 for class-specific. |

## Value

Numeric vector with parameters packed.

---

`rmix` *Draw from a Gaussian mixture model*

---

## Description

Generate i.i.d. samples from a finite Gaussian mixture with either a shared covariance matrix or component-specific covariance matrices.

## Usage

```
rmix(n, pi, mu, sigma, seed_number)
```

## Arguments

| | |
|---|---|
| `n` | Integer. Number of observations to generate. |
| `pi` | Numeric vector of length $g$. Mixing proportions (must sum to 1). |
| `mu` | Numeric matrix $p \times g$. Column $j$ is the mean for component $j$. |
| `sigma` | Either a numeric matrix $p \times p$ (shared covariance), or a numeric array $p \times p \times g$ (component-specific covariances). |
| `seed_number` | Integer. Seed for reproducibility. |

## Value

A list with:

| | |
|---|---|
| `Y` | Numeric matrix $n \times p$ of generated features. |
| `Z` | Numeric matrix $n \times g$ of one-hot component indicators. |
| `clust` | Integer vector $n$, the component labels in `1:g`. |

## Examples

```
## Not run:
  set.seed(1)
  g  <- 2; p <- 2
  pi <- c(0.5, 0.5)
  mu <- cbind(c(1,0), c(-1,0))
  Sigma <- diag(p)
  out <- rmix(500, pi, mu, Sigma, seed_number = 123)
  str(out)

## End(Not run)
```

---

# simulate_mixed_missingness

*Simulate a Gaussian mixture dataset with mixed missingness (MAR + MCAR)*

---

## Description

Simulate a Gaussian mixture dataset with mixed missingness (MAR + MCAR)

## Usage

```
simulate_mixed_missingness(
  n = 500,
  pi,
  mu,
  sigma,
  xi0 = 2,
  xi1 = 3,
  alpha = 0.1,
  seed_id = 123
)
```

## Arguments

| | |
|---|---|
| `n` | Integer; sample size. |
| `pi` | Numeric vector; mixing proportions (sum to 1). |
| `mu` | Matrix (p x K); component means, columns = components. |
| `sigma` | Array (p x p x K); component covariance matrices. |
| `xi0` | Numeric; MAR logit intercept. |

| | |
|---|---|
| xi1 | Numeric; MAR logit slope on entropy. |
| alpha | Numeric in [0,1]; MCAR rate applied within both MAR and observed groups. |
| seed_id | Integer; seed passed to rmix() (your generator). |

## Details

Requires user-provided functions:

- rmix(n, pi, mu, sigma, seed_number)
- get_entropy(dat, n, p, g, paralist)

Missingness mechanism codes:

- 0 = fully observed
- 1 = MCAR
- 2 = MAR (entropy-based)

## Value

A list with:

- data: data.frame with columns x1..xp, en, missing, label, truth
- true_setup: list(pi, mu, sigma)
- groups: list(mar_group, obs_group, mcar_in_mar, mcar_in_obs)
- probs: vector prob_mar
- raw: original rmix output dat augmented with en and labels

---

# unpack_theta *Unpack FMM Parameter Vector*

---

## Description

Unpacks mixture weights, means, and covariance(s) from a parameter vector.

## Usage

```
unpack_theta(theta, g, p, ncov = 1)
```

## Arguments

| | |
|---|---|
| theta | Numeric vector as returned by pack_theta(). |
| g | Integer: number of components. |
| p | Integer: dimension. |
| ncov | Integer: covariance structure; 1 for shared covariance, 2 for class-specific. |

## Value

A list with:

**pi** Mixture weights (length g).

**mu** List of g mean vectors.

**sigma** Shared covariance matrix (ncov=1) or list of g covariance matrices (ncov=2).

# Index

bayesclassifier, 2

compute_d2, 3

EM_FMM_SemiSupervised, 3
EM_FMM_SemiSupervised_Complete_Initial, 5
EM_FMM_SemiSupervised_Initial, *5*, 6
error_beta_classification, 7

get_clusterprobs, 8, *10*
get_entropy, 9

initialestimate, *5*, 10

logsumexp, 11

neg_loglik, 12
normalise_logprob, 13

pack_theta, 14

rmix, 14

simulate_mixed_missingness, 15

unpack_theta, 16