Package 'GMMinit'

January 24, 2026

Type Package

Title Optimal Initial Value for Gaussian Mixture Model

Date 2026-01-20

Version 1.0.0

Maintainer Jing Li <jli178@crimson.ua.edu>

Author Jing Li [aut, cre],
Yana Melnykov [aut]

Description Generating, evaluating, and selecting initialization strategies for Gaussian Mixture Models (GMMs), along with functions to run the Expectation-Maximization (EM) algorithm. Initialization methods are compared using log-likelihood, and the best-fitting model can be selected using BIC. Methods build on initialization strategies for finite mixture models described in Michael and Melnykov (2016) <doi:10.1007/s11634-016-0264-8> and Biernacki et al. (2003) <doi:10.1016/S0167-9473(02)00163-9>, and on the EM algorithm of Dempster et al. (1977) <doi:10.1111/j.2517-6161.1977.tb01600.x>. Background on model-based clustering includes Fraley and Raftery (2002) <doi:10.1198/016214502760047131> and McLachlan and Peel (2000, ISBN:9780471006268).

License GPL (>= 2)

Encoding UTF-8

Repository CRAN

ByteCompile true

Imports mvtnorm, mclust, mvnfast, stats

Config/testthat/edition 3

RoxygenNote 7.3.1

NeedsCompilation no

Date/Publication 2026-01-24 10:40:07 UTC

Contents

`GMMinit-package` *Optimal Initial Value for Gaussian Mixture Model*

Description

Provides an approach to compute an optimal initial value for the Expectation-Maximization (EM) algorithm when fitting a Gaussian Mixture Model (GMM). This ensures better convergence and improved model fitting.

Details

Package:	GMMinit
Type:	Package
Version:	1.0.0
Date:	2026-01-20
License:	GPL (>= 2)
LazyLoad:	no

This package includes functions for:

- Computing an optimal initialization for GMM.
- Running the Expectation-Maximization (EM) algorithm.

Author(s)

Jing Li <jli178@crimson.ua.edu> [aut, cre]
Yana Melnykov <ymelnykov@ua.edu> [aut]

References

Michael, S., & Melnykov, V. (2016). An effective strategy for initializing the EM algorithm in finite mixture models. *Advances in Data Analysis and Classification*, 10(4), 563–583.

Fraley, C., & Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97(458), 611-631. [mclust]

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1), 100-108. [k-means]

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1), 1-22. [EM Algorithm]

Celeux, G., & Govaert, G. (1992). A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics & Data Analysis*, 14(3), 315-332. [CEM & SEM]

McLachlan, G., & Peel, D. (2000). Finite Mixture Models. John Wiley & Sons. [General EM & GMM]

Biernacki, C., Celeux, G., & Govaert, G. (2003). Choosing starting values for the EM algorithm in Gaussian mixture models. *Computational Statistics & Data Analysis*, 41(3-4), 561-575. [Alternative EM Initializations]

See Also

getInit, runGMM, BestGMM, getBestInit, runEM

Examples

```
# Generate sample data
set.seed(123)
data <- matrix(rnorm(100 * 2), ncol = 2)

# Compute optimal initialization
library(GMMinit)
init <- getInit(data, k = 2, method = "Random")
print(init)
```

BestGMM *Select the Best Gaussian Mixture Model (GMM) Based on BIC*

Description

Identifies and returns the best Gaussian Mixture Model (GMM) result from multiple initialization strategies by selecting the model with the lowest Bayesian Information Criterion (BIC).

Usage

```
BestGMM(all_result)
```

Arguments

`all_result`	A named list of GMM model results produced by different initialization methods. Each element is typically an output from `runGMM()` or a similar GMM-fitting function. Failed initializations may appear as `NULL`.

Details

The `BestGMM` takes the output from `runGMM()` and selects the best model by identifying the initialization method that results in the lowest BIC.

Value

A list describing the selected best GMM model. The returned list contains:

- `best` The GMM model object corresponding to the lowest BIC. This object typically contains:
  - `BIC` The model's BIC value.
  - `param` A list of estimated GMM parameters:
    - `pi_k` Mixing proportions.
    - `mu` Cluster mean matrix.
    - `sigma` Covariance matrices.
  - `cluster` Cluster assignments for each observation.
  - `Z` Posterior probability matrix of cluster memberships.
- `initial_method` A character string identifying which initialization method produced the best model.
- `BIC` A numeric vector of BIC values for all initialization methods, with `NA` for failed fits.

If all candidate models fail, `best` and `initial_method` are returned as `NULL`.

See Also

`runGMM`, `runEM`

Examples

```
# Generate sample data
set.seed(123)
data <- matrix(rnorm(100 * 2), ncol = 2)

# Run GMM clustering with multiple initialization methods
results <- runGMM(data, k = 2)

# Select the best GMM based on BIC
best_model <- BestGMM(results)

# Print the best model
print(best_model)
```

---

getBestInit *Select the Best Initialization Method for a Gaussian Mixture Model (GMM)*

Description

Runs multiple GMM initialization strategies, computes the log-likelihood for each initial parameter set, and selects the best initialization method based on the highest log-likelihood value.

Usage

```
getBestInit(
  x,
  k,
  init_methods = c("Random", "emEM", "emAEM",
                   "hierarchical average", "hierarchical ward",
                   "kmeans", "mclust", "cem", "sem"),
  run_number = 10,
  max_iter = 3,
  tol = 1e-6,
  burn_in = 3,verbose = FALSE
)
```

Arguments

x	A numeric matrix or data frame containing the data to be clustered. Each row is an observation, and each column is a feature.
k	The number of mixture components (clusters).
init_methods	A character vector of initialization strategies. Each method is passed to `getInit()` to generate initial GMM parameters. The default includes: • "Random" • "emEM" • "emAEM" • "hierarchical average" • "hierarchical ward" • "kmeans" • "mclust" • "cem" • "sem"
run_number	Number of short EM runs for EM-based initializers. Passed to `getInit()`.
max_iter	Maximum number of iterations for short EM algorithms.
tol	Convergence tolerance for short EM runs.
burn_in	Burn-in iterations for SEM-based initialization.
verbose	Logical; if TRUE, prints progress messages.

Value

A list summarizing the best initialization method and associated results:

- `best_method` The name of the initialization method that achieved the highest log-likelihood.
- `best_result` The initial parameter set (as returned by `getInit()`) corresponding to the best method.
- `loglik_table` A data frame containing the log-likelihood achieved by each initialization method:
  - `method` — Method name
  - `loglik` — Computed log-likelihood

See Also

getInit, runGMM, runEM

Examples

```
# Simulated data
set.seed(123)
x <- matrix(rnorm(200), ncol = 2)

# Run selection of best initialization
 result <- getBestInit(x, k = 2)

 result$best_method
 result$loglik_table
```

---

getInit *Initialize Parameters for the EM Algorithm in Gaussian Mixture Models*

Description

Selects an initialization method to generate starting parameters for the Expectation-Maximization (EM) algorithm in Gaussian Mixture Models (GMMs).

Usage

```
getInit(x, k, method, run_number = 10, max_iter = 3, tol = 1e-6, burn_in = 3)
```

Arguments

x	A numeric matrix or data frame where rows represent observations and columns represent variables.
k	An integer specifying the number of clusters.
method	A character string specifying the initialization method to use. Options include: • "Random": Randomly generates cluster centers. • "emEM": Multi-start EM initialization and choose the one with maximum loglikelihood. • "emAEM": Alternative Expectation-Maximization initialization. • "hierarchical average": Uses hierarchical clustering (average linkage). • "hierarchical ward": Uses hierarchical clustering (Ward's method). • "kmeans": Uses k-means clustering to initialize parameters. • "mclust": Uses model-based clustering from the **mclust** package. • "cem": Classification Expectation-Maximization (CEM). • "sem": Stochastic Expectation-Maximization (SEM).
run_number	An integer specifying the number of times the initialization process should be repeated for emEM and emAEM (default is 10).

max_iter	An integer specifying the maximum number of iterations for emEM and emAEM initialization (default is 3).
tol	A numeric value specifying the convergence tolerance for the EM algorithm (default is 1e-6).
burn_in	An integer specifying the number of burn-in iterations for stochastic methods (default is 3).

Details

This function selects an appropriate initialization method and returns the starting values for the EM algorithm. It helps to prevent local optima issues and improve convergence in Gaussian mixture models.

Value

A list containing the initialized parameters for the EM algorithm, including:

- mu: Cluster means.
- sigma: Covariance matrices.
- pi: Mixing proportions.

See Also

mclust, kmeans

Examples

```
# Generate sample data
set.seed(123)
data <- matrix(rnorm(100 * 2), ncol = 2)

# Compute optimal initialization using k-means
library(GMMinit)
init_params <- getInit(data, k = 2, method = "kmeans")
print(init_params)
```

---

runEM *Expectation-Maximization (EM) Algorithm for Gaussian Mixture Models*

Description

Implements the Expectation-Maximization (EM) algorithm for fitting Gaussian Mixture Models (GMMs). It iteratively updates the model parameters (means, covariances, and mixing proportions) until convergence.

Usage

```
runEM(x, param, max_iter = 100, tol = 1e-5)
```

Arguments

`x`	A numeric matrix or data frame where rows represent observations and columns represent variables.
`param`	A list containing the initial parameters for the EM algorithm: • `param[[1]]`: Mixing proportions (π_k). • `param[[2]]`: Cluster means (μ). • `param[[3]]`: Covariance matrices (Σ).
`max_iter`	An integer specifying the maximum number of iterations allowed for the EM algorithm. (default is 100)
`tol`	A numeric value specifying the convergence tolerance threshold (default is 1e-5). The algorithm stops when the relative change in log-likelihood is below this value.

Details

The EM algorithm iteratively refines the estimates of the Gaussian Mixture Model (GMM) parameters by alternating between two steps:

- **E-step**: Computes the posterior probabilities (responsibilities) of cluster membership for each observation.
- **M-step**: Updates the parameters (means, covariances, and mixing proportions) based on the computed responsibilities.

Convergence is assessed using the log-likelihood function.

Value

A list containing the following components:

- `BIC`: Bayesian Information Criterion (BIC) value for model selection.
- `param`: A list containing the estimated model parameters:
  - `param[[1]]`: Updated mixing proportions.
  - `param[[2]]`: Updated cluster means.
  - `param[[3]]`: Updated covariance matrices.
- `cluster_labels`: Cluster assignments (most probable cluster for each observation).
- `Z`: Posterior probability matrix (γ), rounded to 4 decimal places.

See Also

`getInit`

Examples

```
# Generate synthetic data
set.seed(123)
data <- matrix(rnorm(100 * 2), ncol = 2)

# Initialize parameters using k-means
init_params <- getInit(data, k = 2, method = "kmeans")

# Run the EM algorithm
em_results <- runEM(data, param = init_params, max_iter = 100, tol = 1e-5)

# Print results
print(em_results$BIC)
print(em_results$cluster)
```

---

runGMM *Run Gaussian Mixture Model (GMM) Clustering with Multiple Initialization Strategies*

Description

Applies the Gaussian Mixture Model (GMM) to a dataset using multiple initialization strategies. It runs the Expectation-Maximization (EM) algorithm for each initialization method and returns results for all methods.

Usage

```
runGMM(x, k, max_iter = 100,
    run_number = 10, smax_iter = 3,
       s_iter = 10, c_iter = 10, tol = 1e-6, burn_in = 3,verbose = FALSE)
```

Arguments

x	A numeric matrix or data frame where rows represent observations and columns represent variables.
k	An integer specifying the number of clusters.
max_iter	maximum iteration for running long EM
run_number	number of short em for emEM and emAEM initialization methods(default is 10).
smax_iter	An integer specifying the maximum number of iterations for short EM (default is 3).
s_iter	An integer specifying the number of iterations for the Stochastic Expectation-Maximization (SEM) algorithm (default is 10).
c_iter	An integer specifying the number of iterations for the Classification Expectation-Maximization (CEM) algorithm (default is 10).
tol	A numeric value specifying the convergence tolerance threshold (default is 1e-6).

burn_in	An integer specifying the number of burn-in iterations for stochastic methods (default is 3).
verbose	Logical; if TRUE, prints progress messages.

Details

The runGMM applies multiple initialization strategies for fitting a Gaussian Mixture Model (GMM) using the Expectation-Maximization (EM) algorithm. Each initialization method is evaluated separately, and the results are returned for all tested methods.

Value

A named list where each element corresponds to an initialization method and contains the results of the EM algorithm:

- "Random": Results using random initialization.
- "hierarchical.average": Results using hierarchical clustering (average linkage).
- "hierarchical.ward": Results using hierarchical clustering (Ward's method).
- "kmeans": Results using K-means clustering initialization.
- "emEM": Results using multi-start Expectation-Maximization (EM).
- "emAEM": Results using the alternative EM initialization method.
- "sem": Results using the Stochastic Expectation-Maximization (SEM).
- "cem": Results using the Classification Expectation-Maximization (CEM).
- "mclust": Results using model-based clustering from the **mclust** package.

Each element in the returned list contains:

- BIC: The Bayesian Information Criterion (BIC) value for model selection.
- param: A list with the estimated GMM parameters:
  - pi_k: Updated mixing proportions.
  - mu: Updated cluster means.
  - sigma: Updated covariance matrices.
- cluster_assignments: Cluster labels assigned to each observation.
- Z: Posterior probability matrix of cluster memberships.

If an initialization method fails, the corresponding list element will contain an error message.

See Also

runEM, getInit

Examples

```
# Generate sample data
set.seed(123)
data <- matrix(rnorm(100 * 2), ncol = 2)

# Run GMM clustering with different initialization strategies
results <- runGMM(data, k = 2)
results
```

Index