Package ‘MGMM’

September 30, 2023

Title  Missingness Aware Gaussian Mixture Models
Date  2023-09-30
Version  1.0.1.1

Description  Parameter estimation and classification for Gaussian Mixture Models (GMMs) in the presence of missing data. This package complements existing implementations by allowing for both missing elements in the input vectors and full (as opposed to strictly diagonal) covariance matrices. Estimation is performed using an expectation conditional maximization algorithm that accounts for missingness of both the cluster assignments and the vector components. The output includes the marginal cluster membership probabilities; the mean and covariance of each cluster; the posterior probabilities of cluster membership; and a completed version of the input data, with missing values imputed to their posterior expectations. For additional details, please see McCaw ZR, Julienne H, Aschard H. "Fitting Gaussian mixture models on incomplete data." <doi:10.1186/s12859-022-04740-9>.

Depends  R (>= 3.5.0)
License  GPL-3
Encoding  UTF-8
LinkingTo  Rcpp, RcppArmadillo
Imports  cluster, methods, mvnfast, plyr, Rcpp (>= 1.0.3), stats
Suggests  testthat (>= 3.0.0), knitr, rmarkdown, withr
VignetteBuilder  knitr
RoxygenNote  7.2.3
Config/testthat/edition  3
NeedsCompilation  yes
Author  Zachary McCaw [aut, cre] (<https://orcid.org/0000-0002-2006-9828>)
Maintainer  Zachary McCaw <zmccaw@alumni.harvard.edu>
Repository  CRAN
Date/Publication  2023-09-30 15:42:38 UTC
Description

Calculates the Calinski-Harabaz index.

Usage

CalHar(data, assign, means)

Arguments

data Observations.
assign Assignments.
means List of cluster means.
ChooseK

Value
Scalar metric.

Cluster Number Selection

Description
Function to choose the number of clusters k. Examines cluster numbers between k0 and k1. For each cluster number, generates boot bootstrap data sets, fits the Gaussian Mixture Model (FitGMM), and calculates quality metrics (ClustQual). For each metric, determines the optimal cluster number k_opt, and the k_1SE, the smallest cluster number whose quality is within 1 SE of the optimum.

Usage
ChooseK(
  data,
  k0 = 2,
  k1 = NULL,
  boot = 100,
  init_means = NULL,
  fix_means = FALSE,
  init_covs = NULL,
  init_props = NULL,
  maxit = 10,
  eps = 1e-04,
  report = TRUE
)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Numeric data matrix.</td>
</tr>
<tr>
<td>k0</td>
<td>Minimum number of clusters.</td>
</tr>
<tr>
<td>k1</td>
<td>Maximum number of clusters.</td>
</tr>
<tr>
<td>boot</td>
<td>Bootstrap replicates.</td>
</tr>
<tr>
<td>init_means</td>
<td>Optional list of initial mean vectors.</td>
</tr>
<tr>
<td>fix_means</td>
<td>Fix the means to their starting value? Must provide initial values.</td>
</tr>
<tr>
<td>init_covs</td>
<td>Optional list of initial covariance matrices.</td>
</tr>
<tr>
<td>init_props</td>
<td>Optional vector of initial cluster proportions.</td>
</tr>
<tr>
<td>maxit</td>
<td>Maximum number of EM iterations.</td>
</tr>
<tr>
<td>eps</td>
<td>Minimum acceptable increment in the EM objective.</td>
</tr>
<tr>
<td>report</td>
<td>Report bootstrap progress?</td>
</tr>
</tbody>
</table>
Value

List containing Choices, the recommended number of clusters according to each quality metric, and Results, the mean and standard error of the quality metrics at each cluster number evaluated.

See Also

See ClustQual for evaluating cluster quality, and FitGMM for estimating the GMM with a specified cluster number.

Examples

set.seed(100)
mean_list <- list(c(2, 2), c(2, -2), c(-2, 2), c(-2, -2))
data <- rGMM(n = 500, d = 2, k = 4, means = mean_list)
choose_k <- ChooseK(data, k0 = 2, k1 = 6, boot = 10)
choose_k$Choices

<table>
<thead>
<tr>
<th>ClustQual</th>
<th>Cluster Quality</th>
</tr>
</thead>
</table>

Description

Evaluates cluster quality. Returns the following metrics:

- **BIC**: Bayesian Information Criterion, lower value indicates better clustering quality.
- **CHI**: Calinski-Harabaz Index, higher value indicates better clustering quality.
- **DBI**: Davies-Bouldin, lower value indicates better clustering quality.
- **SIL**: Silhouette Width, higher value indicates better clustering quality.

Usage

ClustQual(fit)

Arguments

fit Object of class mix.

Value

List containing the cluster quality metrics.

See Also

See ChooseK for using quality metrics to choose the cluster number.
Examples

```r
set.seed(100)

# Data generation
mean_list = list(
  c(2, 2, 2),
  c(-2, 2, 2),
  c(2, -2, 2),
  c(2, 2, -2)
)
data <- rGMM(n = 500, d = 3, k = 4, means = mean_list)
fit <- FitGMM(data, k = 4)

# Clustering quality
cluster_qual <- ClustQual(fit)
```

---

**CombineMIs**

**Combine Multiple Imputations**

**Description**

Combines point estimates and standard errors across multiple imputations.

**Usage**

```r
CombineMIs(points, covs)
```

**Arguments**

- `points`: List of point estimates, potentially vector valued.
- `covs`: List of sampling covariances, potentially matrix valued.

**Value**

List containing the final point estimate (`point`) and sampling covariance (`cov`).

**Examples**

```r
set.seed(100)

# Generate data and introduce missingness.
data <- rGMM(n = 25, d = 2, k = 1)
data[1, 1] <- NA
data[2, 2] <- NA
data[3, 3] <- NA

# Fit GMM.
fit <- FitGMM(data)
```
# Lists to store summary statistics.
points <- list()
covs <- list()

# Perform 50 multiple imputations.
# For each, calculate the marginal mean and its sampling variance.
for (i in seq_len(50)) {
  imputed <- GenImputation(fit)
  points[[i]] <- apply(imputed, 2, mean)
  covs[[i]] <- cov(imputed) / nrow(imputed)
}

# Combine summary statistics across imputations.
results <- CombineMIs(points, covs)

---

DavBou

**Davies-Bouldin Index**

**Description**

Calculates the Davies-Bouldin index.

**Usage**

DavBou(data, assign, means)

**Arguments**

- **data**: Observations
- **assign**: Assignments
- **means**: List of cluster means

**Value**

Scalar index.

---

FitGMM

**Estimate Multivariate Normal Mixture**

**Description**

Given an $n \times d$ matrix of random vectors, estimates the parameters of a Gaussian Mixture Model (GMM). Accommodates arbitrary patterns of missingness at random (MAR) in the input vectors.
**Usage**

```r
FitGMM(
  data,
  k = 1,
  init_means = NULL,
  fix_means = FALSE,
  init_covs = NULL,
  init_props = NULL,
  maxit = 100,
  eps = 1e-06,
  report = TRUE
)
```

**Arguments**

- `data` Numeric data matrix.
- `k` Number of mixture components. Defaults to 1.
- `init_means` Optional list of initial mean vectors.
- `fix_means` Fix the means to their starting value? Must provide initial values.
- `init_covs` Optional list of initial covariance matrices.
- `init_props` Optional vector of initial cluster proportions.
- `maxit` Maximum number of EM iterations.
- `eps` Minimum acceptable increment in the EM objective.
- `report` Report fitting progress?

**Details**

Initial values for the cluster means, covariances, and proportions are specified using \( \mathbf{M0} \), \( \mathbf{S0} \), and \( \pi0 \), respectively. If the data contains complete observations, i.e. observations with no missing elements, then `fit.GMM` will attempt to initialize these parameters internally using K-means. If the data contains no complete observations, then initial values are required for \( \mathbf{M0} \), \( \mathbf{S0} \), and \( \pi0 \).

**Value**

- For a single component, an object of class `mvn`, containing the estimated mean and covariance, the final objective function, and the imputed data.
- For a multicomponent model \( k > 1 \), an object of class `mix`, containing the estimated means, covariances, cluster proportions, cluster responsibilities, and observation assignments.

**See Also**

See `rGMM` for data generation, and `ChooseK` for selecting the number of clusters.
Examples

# Single component without missingness
# Bivariate normal observations
sigma <- matrix(c(1, 0.5, 0.5, 1), nrow = 2)
data <- rGMM(n = 1e3, d = 2, k = 1, means = c(2, 2), covs = sigma)
fit <- FitGMM(data, k = 1)

# Single component with missingness
# Trivariate normal observations
mean_list <- list(c(-2, -2, -2), c(2, 2, 2))
sigma <- matrix(c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)
data <- rGMM(n = 1e3, d = 3, k = 2, means = mean_list, covs = sigma)
fit <- FitGMM(data, k = 2)

# Two components without missingness
# Trivariate normal observations
mean_list <- list(c(-2, -2, -2), c(2, 2, 2))
sigma <- matrix(c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)
data <- rGMM(n = 1e3, d = 3, k = 2, means = mean_list, covs = sigma)
fit <- FitGMM(data, k = 2)

# Four components with missingness
# Bivariate normal observations
# Note: Fitting is slow.
mean_list <- list(c(2, 2), c(2, -2), c(-2, 2), c(-2, -2))
sigma <- 0.5 * diag(2)
data <- rGMM(n = 1000, d = 2, k = 4, pi = c(0.35, 0.15, 0.15, 0.35),
m = 0.1, means = mean_list, covs = sigma)
fit <- FitGMM(data, k = 4)

FitMix

Fit Multivariate Mixture Distribution

Description

Given a matrix of random vectors, estimates the parameters for a mixture of multivariate normal distributions. Accommodates arbitrary patterns of missingness, provided the elements are missing at random (MAR).
Usage

FitMix(
  data,
  k = 2,
  init_means = NULL,
  fix_means = FALSE,
  init_covs = NULL,
  init_props = NULL,
  maxit = 100,
  eps = 1e-06,
  report = FALSE
)

Arguments

data          Numeric data matrix.
k             Number of mixture components. Defaults to 2.
init_means    Optional list of initial mean vectors.
fix_means     Fix means to their starting values? Must initialize.
init_covs     Optional list of initial covariance matrices.
init_props    Optional vector of initial cluster proportions.
maxit         Maximum number of EM iterations.
eps           Minimum acceptable increment in the EM objective.
report        Report fitting progress?

Value

Object of class mix.

Description

Given a matrix of n x d-dimensional random vectors, possibly containing missing elements, estimates the mean and covariance of the best fitting multivariate normal distribution.

Usage

FitMVN(
  data,
  init_mean = NULL,
  fix_mean = FALSE,
  init_cov = NULL,
maxit = 100,
eps = 1e-06,
report = TRUE
)

Arguments

data Numeric data matrix.
init_mean Optional initial mean vector.
fix_mean Fix the mean to its starting value? Must initialize.
init_cov Optional initial covariance matrix.
maxit Maximum number of EM iterations.
eps Minimum acceptable increment in the EM objective.
report Report fitting progress?

Value

An object of class mvn.

Description

Generates a stochastic imputation of a data set from a fitted data set.

Usage

GenImputation(fit)

Arguments

fit Fitted model.

Value

Numeric matrix with missing values imputed.
Examples

```r
set.seed(100)

# Generate data and introduce missingness.
data <- rGMM(n = 25, d = 2, k = 1)
data[1, 1] <- NA
data[2, 2] <- NA
data[3, ] <- NA

# Fit GMM.
fit <- FitGMM(data)

# Generate imputation.
imputed <- GenImputation(fit)
```

---

**logLik.mix**  
*Log likelihood for Fitted GMM*

**Description**

Log likelihood for Fitted GMM

**Usage**

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

**Arguments**

| object | A mix object. |
| ...    | Unused.      |

---

**logLik.mvn**  
*Log likelihood for Fitted MVN Model*

**Description**

Log likelihood for Fitted MVN Model

**Usage**

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

**Arguments**

| object | A mvn object. |
| ...    | Unused.      |
mean.mix

Mean for Fitted GMM

Description

Mean for Fitted GMM

Usage

## S3 method for class 'mix'
mean(x, ...)

Arguments

x
A mix object.

... Unused.

mean.mvn

Mean for Fitted MVN Model

Description

Mean for Fitted MVN Model

Usage

## S3 method for class 'mvn'
mean(x, ...)

Arguments

x
A mvn object.

... Unused.
mix-class

Mixture Model Class

Description

Defines a class to hold Gaussian Mixture Models.

Slots

Assignments  Maximum a posteriori assignments.
Completed  Completed data, with missing values imputed to their posterior expectations.
Components  Number of components.
Covariances  List of fitted cluster covariance matrices.
Data  Original data, with missing values present.
Density  Density of each component at each example.
Means  List of fitted cluster means.
Objective  Final value of the EM objective.
Proportions  Fitted cluster proportions.
Responsibilities  Posterior membership probabilities for each example.

MixUpdateMeans

Mean Update for Mixture of MVNs with Missingness.

Description

Mean Update for Mixture of MVNs with Missingness.

Usage

MixUpdateMeans(split_data, means, covs, gamma)

Arguments

split_data  Data partitioned by missingness.
means  List of component means.
covs  List of component covariances.
gamma  List of component responsibilities.

Value

List containing the updated component means.
### mvn-class

**Multivariate Normal Model Class**

#### Description

Defines a class to hold multivariate normal models.

#### Slots

- **Completed** Completed data, with missing values imputed to their posterior expectations.
- **Covariance** Fitted covariance matrix.
- **Data** Original data, with missing values present.
- **Mean** Fitted mean vector.
- **Objective** Final value of the EM objective.

### PartitionData

**Partition Data by Missingness Pattern**

#### Description

Returns a list with the input data split in separate matrices for complete cases, incomplete cases, and empty cases.

#### Usage

```r
PartitionData(data)
```

#### Arguments

- **data** Data.frame.

#### Value

List containing:

- The original row and column names: ‘orig_row_names’, ‘orig_col_names’.
- The original row and column numbers: ‘n_row’ and ‘n_col’.
- The complete cases ‘data_comp’.
- The incomplete cases ‘data_incomp’.
- The empty cases ‘data_empty’.
- Counts of complete ‘n0’, incomplete ‘n1’, and empty ‘n2’ cases.
- Initial order of the observations ‘init_order’.
**Description**

Print method for objects of class `mix`.

**Usage**

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

**Arguments**

- `x` A `mix` object.
- `...` Unused.

---

**print.mvn** 
Print for Fitted MVN Model

**Description**

Print for Fitted MVN Model

**Usage**

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

**Arguments**

- `x` A `mvn` object.
- `...` Unused.
ReconstituteData  

**Description**  
Reassembles a data matrix split by missingness pattern.

**Usage**  
ReconstituteData(split_data)

**Arguments**  
- **split_data**  
  Split data are returned by *PartitionData*.

**Value**  
Numeric matrix.

rGMM  

**Generate Data from Gaussian Mixture Models**  

**Description**  
Generates an $n \times d$ matrix of multivariate normal random vectors with observations (examples) as rows. If $k = 1$, all observations belong to the same cluster. If $k > 1$ the observations are generated via two-step procedure. First, the cluster membership is drawn from a multinomial distribution, with mixture proportions specified by $p_i$. Conditional on cluster membership, the observation is drawn from a multivariate normal distribution, with cluster-specific mean and covariance. The cluster means are provided using $\text{means}$, and the cluster covariance matrices are provided using $\text{covs}$. If $\text{miss} > 0$, missingness is introduced, completely at random, by setting that proportion of elements in the data matrix to NA.

**Usage**  
rGMM(n, d = 2, k = 1, pi = NULL, miss = 0, means = NULL, covs = NULL)

**Arguments**  
- **n**  
  Observations (rows).
- **d**  
  Observation dimension (columns).
- **k**  
  Number of mixture components. Defaults to 1.
- **pi**  
  Mixture proportions. If omitted, components are assumed equiprobable.
- **miss**  
  Proportion of elements missing, $\text{miss} \in [0, 1)$.
means Either a prototype mean vector, or a list of mean vectors. Defaults to the zero vector.

covs Either a prototype covariance matrix, or a list of covariance matrices. Defaults to the identity matrix.

Value

Numeric matrix with observations as rows. Row numbers specify the true cluster assignments.

See Also

For estimation, see `FitGMM`.

Examples

```r
set.seed(100)
# Single component without missingness.
# Bivariate normal observations.
cov <- matrix(c(1, 0.5, 0.5, 1), nrow = 2)
data <- rGMM(n = 1e3, d = 2, k = 1, means = c(2, 2), covs = cov)

# Single component with missingness.
# Trivariate normal observations.
mean_list <- list(c(-2, -2, -2), c(2, 2, 2))
cov <- matrix(c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)
data <- rGMM(n = 1e3, d = 3, k = 2, means = mean_list, covs = cov)

# Two components without missingness.
# Trivariate normal observations.
mean_list <- list(c(-2, -2, -2), c(2, 2, 2))
cov <- matrix(c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), nrow = 3)
data <- rGMM(n = 1e3, d = 3, k = 2, means = mean_list, covs = cov)

# Four components with missingness.
# Bivariate normal observations.
mean_list <- list(c(2, 2), c(2, -2), c(-2, 2), c(-2, -2))
cov <- 0.5 * diag(2)
data <- rGMM(n = 1000,
d = 2,
k = 4,
pi = c(0.35, 0.15, 0.15, 0.35),
miss = 0.1,
means = mean_list,
covs = cov)
```
### show, mix-method

**Show for Fitted Mixture Models**

**Description**

Show for Fitted Mixture Models

**Usage**

```r
## S4 method for signature 'mix'
show(object)
```

**Arguments**

- `object` A `mix` object.

### show, mvn-method

**Show for Multivariate Normal Models**

**Description**

Show for Multivariate Normal Models

**Usage**

```r
## S4 method for signature 'mvn'
show(object)
```

**Arguments**

- `object` A `mvn` object.
**vcov.mix**

Covariance for Fitted GMM

Description

Covariance for Fitted GMM

Usage

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

Arguments

- `object` A `mix` object.
- `...` Unused.

---

**vcov.mvn**

Covariance for Fitted MVN Model

Description

Covariance for Fitted MVN Model

Usage

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

Arguments

- `object` A `mvn` object.
- `...` Unused.
Index

CalHar, 2
ChooseK, 3, 4, 7
ClustQual, 3, 4, 4
CombineMIs, 5

DavBou, 6

FitGMM, 3, 4, 6, 17
FitMix, 8
FitMVN, 9

GenImputation, 10

logLik.mix, 11
logLik.mvn, 11

mean.mix, 12
mean.mvn, 12
mix-class, 13
MixUpdateMeans, 13
mvn-class, 14

PartitionData, 14, 16
print.mix, 15
print.mvn, 15

ReconstituteData, 16
rGMM, 7, 16

show,mix-method, 18
show,mvn-method, 18

vcov.mix, 19
vcov.mvn, 19