

# Package ‘fabMix’

September 20, 2018

**Type** Package

**Title** Overfitting Bayesian Mixtures of Factor Analyzers with an  
Unknown Number of Components

**Version** 3.0

**Date** 2018-09-05

**Author** Panagiotis Papastamoulis

**Maintainer** Panagiotis Papastamoulis <papapast@yahoo.gr>

## Description

Model-based clustering of multivariate continuous data with or without missing values. The underlying model is a Bayesian mixture of factor analyzers with a large number of components (overfitting mixture). Suitable prior assumptions ensure that asymptotically the extra components will have zero posterior weight (empty), therefore, the inference on the number of clusters is based on the “alive” components. The number of factors is considered fixed, and the optimal one can be estimated using information criteria. Identifiability issues related to label switching are dealt by post-processing the simulated output with the ECR algorithm.

**Imports** MASS, doParallel, foreach, label.switching, mvtnorm, doRNG,  
RColorBrewer, corrplot, mclust

**License** GPL-2

**NeedsCompilation** no

## R topics documented:

fabMix-package . . . . .	3
complete.log.likelihood . . . . .	4
complete.log.likelihood_q0 . . . . .	5
complete.log.likelihood_q0_sameSigma . . . . .	6
complete.log.likelihood_Sj . . . . .	6
compute_A_B_G_D_and_simulate_mu_Lambda . . . . .	7
compute_A_B_G_D_and_simulate_mu_Lambda_CCU . . . . .	8
compute_A_B_G_D_and_simulate_mu_Lambda_CUU . . . . .	9
compute_A_B_G_D_and_simulate_mu_Lambda_q0 . . . . .	9
compute_A_B_G_D_and_simulate_mu_Lambda_q0_sameSigma . . . . .	10
compute_A_B_G_D_and_simulate_mu_Lambda_Sj . . . . .	11

compute_sufficient_statistics . . . . .	12
compute_sufficient_statistics_given_mu . . . . .	12
compute_sufficient_statistics_q0 . . . . .	13
dealWithLabelSwitching . . . . .	14
fabMix . . . . .	15
fabMix_CxC . . . . .	17
fabMix_CxU . . . . .	20
fabMix_missing_values . . . . .	22
fabMix_UxC . . . . .	24
fabMix_UxU . . . . .	27
getStuffForDIC . . . . .	29
log_dirichlet_pdf . . . . .	31
myDirichlet . . . . .	31
observed.log.likelihood0 . . . . .	32
observed.log.likelihood0_q0_sameSigma . . . . .	32
observed.log.likelihood0_Sj . . . . .	33
observed.log.likelihood0_Sj_q0 . . . . .	34
overfittingMFA . . . . .	34
overfittingMFA_CCC . . . . .	35
overfittingMFA_CCU . . . . .	36
overfittingMFA_CUC . . . . .	37
overfittingMFA_CUU . . . . .	38
overfittingMFA_missing_values . . . . .	39
overfittingMFA_Sj . . . . .	40
overfittingMFA_Sj_missing_values . . . . .	41
overfittingMFA_UCC . . . . .	42
overfittingMFA_UUC . . . . .	43
overfitting_q0 . . . . .	44
overfitting_q0_sameSigma . . . . .	45
plot.fabMix.object . . . . .	46
print.fabMix.object . . . . .	47
simData . . . . .	48
update_all_y . . . . .	49
update_all_y_Sj . . . . .	50
update_OmegaINV . . . . .	50
update_OmegaINV_Cxx . . . . .	51
update_SigmaINV_faster . . . . .	52
update_SigmaINV_faster_q0 . . . . .	52
update_SigmaINV_faster_q0_sameSigma . . . . .	53
update_SigmaINV_faster_Sj . . . . .	54
update_SigmaINV_xCC . . . . .	54
update_SigmaINV_xUC . . . . .	55
update_z2 . . . . .	56
update_z2_Sj . . . . .	56
update_z4 . . . . .	57
update_z4_Sj . . . . .	58
update_z_b . . . . .	58
update_z_b_Sj . . . . .	59

<i>fabMix-package</i>	3
update_z_q0 . . . . .	60
update_z_q0_sameSigma . . . . .	60
waveDataset1500 . . . . .	61
<b>Index</b>	<b>62</b>

---

fabMix-package	<i>Overfitting Bayesian Mixtures of Factor Analyzers with an Unknown Number of Components</i>
----------------	---

---

**Description**

Model-based clustering of multivariate continuous data with or without missing values. The underlying model is a Bayesian mixture of factor analyzers with a large number of components (overfitting mixture). Suitable prior assumptions ensure that asymptotically the extra components will have zero posterior weight (empty), therefore, the inference on the number of clusters is based on the “alive” components. The number of factors is considered fixed, and the optimal one can be estimated using information criteria. Identifiability issues related to label switching are dealt by post-processing the simulated output with the ECR algorithm.

**Author(s)**

Panagiotis Papastamoulis  
Maintainer: Panagiotis Papastamoulis <papapast@yahoo.gr>

**References**

Fokoue, E. and Titterington, D.M. (2003). Mixtures of Factor Analysers: Bayesian Estimation and Inference by Stochastic Simulation. *Machine Learning*, 50(1): 73-94.

Papastamoulis P. and Iliopoulos G. (2010). An artificial allocations based solution to the label switching problem in Bayesian analysis of mixtures of distributions. *Journal of Computational and Graphical Statistics*, 19: 313-331.

van Havre, Z., White, N., Rousseau, J. and Mengersen, K. (2015). Overfitting Bayesian Mixture Models with an Unknown Number of Components. *PLOS ONE*, 10(7): 1-27.

Papastamoulis, P. (2016). *label.switching*: An R Package for Dealing with the Label Switching Problem in MCMC Outputs. *Journal of Statistical Software*, 69(1), 1-24.

Papastamoulis, P. (2017). Overfitting Bayesian mixtures of factor analyzers with an unknown number of components. *arXiv:1701.04605 [stat.ME]*

**See Also**

[fabMix](#), [dealWithLabelSwitching](#), [getStuffForDIC](#)

**Examples**

```

# simulate a synthetic dataset along the lines of the paper:
n = 1000          # sample size
p = 40            # number of variables
q = 4             # number of factors
K = 10            # number of clusters
sINV_diag = 1/((1:p)) # diagonal of inverse variance of errors
set.seed(10)
syntheticDataset <- simData(K.true = K, n = n, q = q, p = p, sINV_values = sINV_diag )

## Not run:

# define parameters
Kmax <- 20        # number of overfitted mixture components
nChains <- 8      # number of parallel chains
dN <- 1
# Dirichlet prior of mixture weights per chain.
# The target chain corresponds to the first entry.
dirPriorAlphas <- c(1, 1 + dN * (2:nChains - 1))/Kmax
outputFolder <- "fabMixExample"
# Run algorithm
fabMix( dirPriorAlphas = dirPriorAlphas,
        rawData = syntheticDataset$data,
        outDir = outputFolder, Kmax = Kmax, mCycles = 1200,
        burnCycles = 200, q = q)

# Compute information criteria:
getStuffForDIC(x_data = syntheticDataset$data, outputFolder = outputFolder, q = q)

# Deal with label switching:
dealWithLabelSwitching(x_data = syntheticDataset$data,
                       outputFolder = outputFolder, q = q,
                       compute_regularized_expression = TRUE, Km = Kmax)

## End(Not run)

```

---

complete.log.likelihood

*Complete log-likelihood function*


---

**Description**

Complete log-likelihood function

**Usage**

```
complete.log.likelihood(x_data, w, mu, Lambda, SigmaINV, z)
```

**Arguments**

x_data	Data
w	Mixture weights
mu	Marginal means
Lambda	Factor loadings
SigmaINV	Precision matrix (inverse covariance)
z	Allocation vector of the data to the mixture components

**Value**

complete log-likelihood value

**Author(s)**

Panagiotis Papastamoulis

---

complete.log.likelihood\_q0

*Complete log-likelihood function for  $q = 0$*

---

**Description**

Complete log-likelihood function

**Usage**

```
complete.log.likelihood_q0(x_data, w, mu, SigmaINV, z)
```

**Arguments**

x_data	Data
w	Mixture weights
mu	Marginal means
SigmaINV	Precision matrix (inverse covariance)
z	Allocation vector of the data to the mixture components

**Value**

complete log-likelihood value

**Author(s)**

Panagiotis Papastamoulis

---

`complete.log.likelihood_q0_sameSigma`*Complete log-likelihood function for  $q = 0$* 

---

**Description**

Complete log-likelihood function

**Usage**

```
complete.log.likelihood_q0_sameSigma(x_data, w, mu, SigmaINV, z)
```

**Arguments**

<code>x_data</code>	Data
<code>w</code>	Mixture weights
<code>mu</code>	Marginal means
<code>SigmaINV</code>	Precision matrix (inverse covariance)
<code>z</code>	Allocation vector of the data to the mixture components

**Value**

complete log-likelihood value

**Author(s)**

Panagiotis Papastamoulis

---

`complete.log.likelihood_Sj`*Complete log-likelihood function*

---

**Description**

Complete log-likelihood function

**Usage**

```
complete.log.likelihood_Sj(x_data, w, mu, Lambda, SigmaINV, z)
```

**Arguments**

x_data	Data
w	Mixture weights
mu	Marginal means
Lambda	Factor loadings
SigmaINV	Precision matrix (inverse covariance) per component
z	Allocation vector of the data to the mixture components

**Value**

complete log-likelihood value

**Author(s)**

Panagiotis Papastamoulis

---

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda  
*Computation and simulations*

---

**Description**

This function simulates  $\mu$  and  $\Lambda$ .

**Usage**

```
compute_A_B_G_D_and_simulate_mu_Lambda(SigmaINV,
suff_statistics, OmegaINV, K, priorConst1, T_INV, v_r)
```

**Arguments**

SigmaINV	Precision matrix $\Sigma^{-1}$
suff_statistics	Sufficient statistics
OmegaINV	Prior parameter: $\Omega^{-1}$
K	Number of overfitting mixture components
priorConst1	Prior constant: $T^{-1}\xi$
T_INV	Prior parameter: $T^{-1}\xi$
v_r	This vector is used to set to zero the upper right $(q-1) \times (q-1)$ diagonal block of factor loadings for identifiability purposes.

**Value**

A list containing  $A$ ,  $B$ ,  $\Gamma$ ,  $\Delta$  and a draw from the conditional distributions of  $\mu$  and  $\Lambda$ .

**Author(s)**

Panagiotis Papastamoulis

---

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_CCU  
*Computation and simulations for CCU*

---

**Description**

This function simulates  $\mu$  and  $\Lambda$  for the CCU model.

**Usage**

```
compute_A_B_G_D_and_simulate_mu_Lambda_CCU(SigmaINV,
suff_statistics, OmegaINV, K, priorConst1, T_INV, v_r)
```

**Arguments**

SigmaINV	Precision matrix $\Sigma^{-1}$
suff_statistics	Sufficient statistics
OmegaINV	Prior parameter: $\Omega^{-1}$
K	Number of overfitting mixture components
priorConst1	Prior constant: $T^{-1}\xi$
T_INV	Prior parameter: $T^{-1}\xi$
v_r	This vector is used to set to zero the upper right $(q-1) \times (q-1)$ diagonal block of factor loadings for identifiability purposes.

**Value**

A list containing  $A$ ,  $B$ ,  $\Gamma$ ,  $\Delta$  and a draw from the conditional distributions of  $\mu$  and  $\Lambda$ .

**Author(s)**

Panagiotis Papastamoulis



---

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_CUU  
*Computation and simulations for CUU*

---

### Description

This function simulates  $\mu$  and  $\Lambda$  for the CUU model.

### Usage

```
compute_A_B_G_D_and_simulate_mu_Lambda_CUU(SigmaINV,
suff_statistics, OmegaINV, K, priorConst1, T_INV, v_r)
```

### Arguments

SigmaINV	Precision matrix $\Sigma^{-1}$
suff_statistics	Sufficient statistics
OmegaINV	Prior parameter: $\Omega^{-1}$
K	Number of overfitting mixture components
priorConst1	Prior constant: $T^{-1}\xi$
T_INV	Prior parameter: $T^{-1}\xi$
v_r	This vector is used to set to zero the upper right $(q-1) \times (q-1)$ diagonal block of factor loadings for identifiability purposes.

### Value

A list containing  $A$ ,  $B$ ,  $\Gamma$ ,  $\Delta$  and a draw from the conditional distributions of  $\mu$  and  $\Lambda$ .

### Author(s)

Panagiotis Papastamoulis

---

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_q0  
*Computation and simulations for  $q = 0$ .*

---

### Description

This function simulates  $\mu$ .

### Usage

```
compute_A_B_G_D_and_simulate_mu_Lambda_q0(SigmaINV,
suff_statistics, K, priorConst1, T_INV, v_r)
```

**Arguments**

SigmaINV	Precision matrix $\Sigma^{-1}$
suff_statistics	Sufficient statistics
K	Number of overfitting mixture components
priorConst1	Prior constant: $T^{-1}\xi$
T_INV	Prior parameter: $T^{-1}\xi$
v_r	This vector is used to set to zero the upper right $(q-1) \times (q-1)$ diagonal block of factor loadings for identifiability purposes.

**Value**

A list containing  $A$ ,  $B$ ,  $\Gamma$ ,  $\Delta$  and a draw from the conditional distributions of  $\mu$  and  $\Lambda$ .

**Author(s)**

Panagiotis Papastamoulis

---

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_q0\_sameSigma  
*Computation and simulations for  $q = 0$ .*

---

**Description**

This function simulates  $\mu$ .

**Usage**

```
compute_A_B_G_D_and_simulate_mu_Lambda_q0_sameSigma(SigmaINV,
suff_statistics, K, priorConst1, T_INV, v_r)
```

**Arguments**

SigmaINV	Precision matrix $\Sigma^{-1}$
suff_statistics	Sufficient statistics
K	Number of overfitting mixture components
priorConst1	Prior constant: $T^{-1}\xi$
T_INV	Prior parameter: $T^{-1}\xi$
v_r	This vector is used to set to zero the upper right $(q-1) \times (q-1)$ diagonal block of factor loadings for identifiability purposes.

**Value**

A list containing  $A$ ,  $B$ ,  $\Gamma$ ,  $\Delta$  and a draw from the conditional distributions of  $\mu$  and  $\Lambda$ .

**Author(s)**

Panagiotis Papastamoulis

---

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_Sj  
*Computation and simulations*

---

**Description**

This function simulates  $\mu$  and  $\Lambda$ .

**Usage**

```
compute_A_B_G_D_and_simulate_mu_Lambda_Sj(SigmaINV,
suff_statistics, OmegaINV, K, priorConst1, T_INV, v_r)
```

**Arguments**

SigmaINV	Precision matrix $\Sigma^{-1}$ per component
suff_statistics	Sufficient statistics
OmegaINV	Prior parameter: $\Omega^{-1}$
K	Number of overfitting mixture components
priorConst1	Prior constant: $T^{-1}\xi$
T_INV	Prior parameter: $T^{-1}\xi$
v_r	This vector is used to set to zero the upper right $(q-1) \times (q-1)$ diagonal block of factor loadings for identifiability purposes.

**Value**

A list containing  $A$ ,  $B$ ,  $\Gamma$ ,  $\Delta$  and a draw from the conditional distributions of  $\mu$  and  $\Lambda$ .

**Author(s)**

Panagiotis Papastamoulis

---

compute\_sufficient\_statistics  
*Compute sufficient statistics*

---

**Description**

Compute sufficient statistics given  $y$  and  $z$ .

**Usage**

```
compute_sufficient_statistics(y, z, K, x_data)
```

**Arguments**

$y$	Matrix of factors
$z$	Allocation vector
$K$	Number of components
$x\_data$	Data

**Value**

A list with six entries of sufficient statistics.

**Author(s)**

Panagiotis Papastamoulis

---

compute\_sufficient\_statistics\_given\_mu  
*Compute sufficient statistics given mu*

---

**Description**

Compute sufficient statistics given  $y$  and  $z$ .

**Usage**

```
compute_sufficient_statistics_given_mu(y, z, K, x_data, mu)
```

**Arguments**

$y$	Matrix of factors
$z$	Allocation vector
$K$	Number of components
$x\_data$	Data
$\mu$	Means per component

**Value**

A list with six entries of sufficient statistics.

**Author(s)**

Panagiotis Papastamoulis

---

`compute_sufficient_statistics_q0`

*Compute sufficient statistics for  $q = 0$*

---

**Description**

Compute sufficient statistics given  $z$ .

**Usage**

```
compute_sufficient_statistics_q0(z, K, x_data)
```

**Arguments**

<code>z</code>	Allocation vector
<code>K</code>	Number of components
<code>x_data</code>	Data

**Value**

A list with six entries of sufficient statistics.

**Author(s)**

Panagiotis Papastamoulis

---

dealWithLabelSwitching

*Apply label switching algorithms*


---

## Description

This functions is a wrapper for the `label.switching` package and applies the ECR and ECR.ITERATIVE.1 algorithms. The model may have the same variance of error terms per cluster or not.

## Usage

```
dealWithLabelSwitching(sameSigma, x_data, outputFolder, q, burn,
  z.true, compute_regularized_expression, Km)
```

## Arguments

sameSigma	Logical value indicating whether the parameterization with the same error variance per cluster is used.
x_data	Data
outputFolder	Name of the folder where the <code>fabMix</code> function has saved its output
q	Number of factors
burn	Discard observations as burn-in period (optional).
z.true	An (optional) vector of cluster assignments which is considered as the ground-truth clustering of the data. Useful for direct comparisons against the real parameter values in simulated data.
compute_regularized_expression	Logical. Should regularized expression be computed?
Km	Number of components in the overfitted mixture model.

## Value

The following files are produced in the output folder:

## Author(s)

Panagiotis Papastamoulis

## References

Papastamoulis, P. (2016). `label.switching`: An R Package for Dealing with the Label Switching Problem in MCMC Outputs. *Journal of Statistical Software*, 69(1), 1-24.

---

fabMix	<i>Main function</i>
--------	----------------------

---

**Description**

This function runs parallel chains under a prior tempering scheme of the Dirichlet prior distribution of mixture weights.

**Usage**

```
fabMix(model, dirPriorAlphas, rawData, outDir, Kmax, mCycles,
burnCycles, g, h, alpha_sigma, beta_sigma, q, normalize,
thinning, zStart, nIterPerCycle, gibbs_z,
warm_up_overfitting, warm_up, overfittingInitialization,
progressGraphs, gwar)
```

**Arguments**

model	Any subset of "UUU" "CUU" "UCU" "CCU" "UCC" "UUC" "CUC", "CCC" indicating the fitted models.
dirPriorAlphas	The prior Dirichlet parameters for each chain.
rawData	The observed data as an $n \times p$ matrix. Clustering is performed on the rows of the matrix.
outDir	Name of the output folder.
Kmax	Number of components in the overfitted mixture. Default: 20.
mCycles	Number of MCMC cycles. Each cycle consists of nIterPerCycle MCMC iterations. At the end of each cycle a swap of the state of two randomly chosen adjacent chains is attempted.
burnCycles	Number of cycles that will be discarded as burn-in period.
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
q	A vector containing the number of factors to be fitted.
normalize	Should the observed data be normalized? Default value: TRUE.
thinning	Optional integer denoting the thinning of the kept MCMC cycles.
zStart	Optional starting value for the allocation vector.
nIterPerCycle	Number of iteration per MCMC cycle. Default value: 10.
gibbs_z	Select the gibbs sampling scheme for updating latent allocations of mixture model. Default value: 1.
warm_up_overfitting	Number of iterations for the overfitting initialization scheme. Default value: 100.

warm_up	Number of iterations that will be used to initialize the models before starting proposing switchings. Default value: 500.
overfittingInitialization	Logical value indicating whether the chains are initialized via the overfitting initialization scheme. Default: TRUE.
progressGraphs	Logical value indicating whether to plot successive states of the chains while the sampler runs. Default: FALSE.
gwar	Initialization parameter. Default: 0.05.

**Value**

An object of class `fabMix.object`, that is, a list consisting of the following entries:

bic	Bayesian Information Criterion per model and number of factors.
class	The estimated single best clustering of the observations according to the selected model.
n_Clusters_per_model	The most probable number of clusters (number of non-empty components of the overfitted mixture) per model and number of factors.
posterior_probability	The posterior probability of the estimated allocations according to the selected model.
covariance_matrix	The estimated posterior mean of the covariance matrix per cluster according to the selected model.
mu	The estimated posterior mean of the mean per cluster according to the selected model.
weights	The estimated posterior mean of the mixing proportions according to the selected model.
selected_model	Data frame containing the parameterization, number of clusters and factors of the selected model.
mcmc	A list containing the MCMC draws for the parameters of the selected models. All component-specific parameters have been reordered according to the ECR algorithm in order to undo the label switching problem.
data	The observed data.

**Note**

It is recommended to always use: `normalize = TRUE` (default). Tuning of `dirPriorAlphas` may be necessary to achieve reasonable acceptance rates of chain swaps. Note that the output is reordered in order to deal with the label switching problem, according to the ECR algorithm applied by `dealWithLabelSwitching` function.

**Author(s)**

Panagiotis Papastamoulis



## References

Papastamoulis, P. (2017). Overfitting Bayesian mixtures of factor analyzers with an unknown number of components. arXiv:1701.04605 [stat.ME]

## See Also

[dealWithLabelSwitching](#)

## Examples

```
## Not run:
# simulate a synthetic dataset along the lines of the paper:
n = 1000          # sample size
p = 40            # number of variables
q = 4             # number of factors
K = 10            # number of clusters
sINV_diag = 1/((1:p)) # diagonal of inverse variance of errors
set.seed(10)
syntheticDataset <- simData(K.true = K, n = n, q = q, p = p, sINV_values = sINV_diag )

# define parameters
Kmax <- 20        # number of overfitted mixture components
nChains <- 8      # number of parallel chains
dN <- 1
# Dirichlet prior of mixture weights per chain.
# The target chain corresponds to the first entry.
dirPriorAlphas <- c(1, 1 + dN * (2:nChains - 1))/Kmax
outputFolder <- "fabMixExample"
# Run algorithm
fabMix( dirPriorAlphas = dirPriorAlphas,
        rawData = syntheticDataset$data,
        outDir = outputFolder, Kmax = Kmax, mCycles = 1200,
        burnCycles = 200, q = q)

## End(Not run)
```

## Description

This function runs parallel chains under a prior tempering scheme of the Dirichlet prior distribution of mixture weights.

**Usage**

```
fabMix_CxC(sameSigma, dirPriorAlphas, rawData, outDir, Kmax, mCycles,
burnCycles, g, h, alpha_sigma, beta_sigma, q, normalize,
thinning, zStart, nIterPerCycle, gibbs_z,
warm_up_overfitting, warm_up, overfittingInitialization,
progressGraphs, gwar, cccStart)
```

**Arguments**

sameSigma	Logical value denoting the parameterization of the error variance per component. If TRUE, the parameterization CCU is fitted. Otherwise, the parameterization CUU is fitted.
dirPriorAlphas	The prior Dirichlet parameters for each chain.
rawData	The observed data as an $n \times p$ matrix. Clustering is performed on the rows of the matrix.
outDir	Name of the output folder.
Kmax	Number of components in the overfitted mixture. Default: 20.
mCycles	Number of MCMC cycles. Each cycle consists of nIterPerCycle MCMC iterations. At the end of each cycle a swap of the state of two randomly chosen adjacent chains is attempted.
burnCycles	Number of cycles that will be discarded as burn-in period.
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
q	Number of factors $q$ , where $1 \leq q \leq L$ . An error is thrown if the Ledermann bound ( $L$ ) is exceeded.
normalize	Should the observed data be normalized? Default value: TRUE.
thinning	Optional integer denoting the thinning of the kept MCMC cycles.
zStart	Optional starting value for the allocation vector.
nIterPerCycle	Number of iteration per MCMC cycle. Default value: 10.
gibbs_z	Select the gibbs sampling scheme for updating latent allocations of mixture model. Default value: 1.
warm_up_overfitting	Number of iterations for the overfitting initialization scheme. Default value: 100.
warm_up	Number of iterations that will be used to initialize the models before starting proposing switchings. Default value: 500.
overfittingInitialization	Logical value indicating whether the chains are initialized via the overfitting initialization scheme. Default: TRUE.
progressGraphs	Logical value indicating whether to plot successive states of the chains while the sampler runs. Default: FALSE.
gwar	Initialization parameter. Default: 0.05.
cccStart	Initialization from the CCC model.

**Value**

List of files written to outDir

**Note**

It is recommended to always use: `normalize = TRUE` (default). Tuning of `dirPriorAlphas` may be necessary to achieve reasonable acceptance rates of chain swaps. Also note that the output is not identifiable due to label switching and the user has to subsequently call the `dealWithLabelSwitching` function.

**Author(s)**

Panagiotis Papastamoulis

**References**

Papastamoulis, P. (2017). Overfitting Bayesian mixtures of factor analyzers with an unknown number of components. arXiv:1701.04605 [stat.ME]

**See Also**

[dealWithLabelSwitching](#)

**Examples**

```
## Not run:
# simulate a synthetic dataset along the lines of the paper:
n = 1000          # sample size
p = 40            # number of variables
q = 4             # number of factors
K = 10            # number of clusters
sINV_diag = 1/((1:p)) # diagonal of inverse variance of errors
set.seed(10)
syntheticDataset <- simData(K.true = K, n = n, q = q, p = p, sINV_values = sINV_diag )

# define parameters
Kmax <- 20        # number of overfitted mixture components
nChains <- 8      # number of parallel chains
dN <- 1
# Dirichlet prior of mixture weights per chain.
# The target chain corresponds to the first entry.
dirPriorAlphas <- c(1, 1 + dN * (2:nChains - 1))/Kmax
outputFolder <- "fabMixExample"
# Run algorithm
fabMix_CxC( dirPriorAlphas = dirPriorAlphas,
            rawData = syntheticDataset$data,
            outDir = outputFolder, Kmax = Kmax, mCycles = 1200,
            burnCycles = 200, q = q)

# Compute information criteria:
#getStuffForDIC(x_data = syntheticDataset$data, outputFolder = outputFolder, q = q)
```

```
# Deal with label switching:
#dealWithLabelSwitching(x_data = syntheticDataset$data,
#      outputFolder = outputFolder, q = q,
#      compute_regularized_expression = TRUE, Km = Kmax)

## End(Not run)
```

---

fabMix\_CxU

*Main function of the package for CCU, CUU models*


---

## Description

This function runs parallel chains under a prior tempering scheme of the Dirichlet prior distribution of mixture weights.

## Usage

```
fabMix_CxU(sameSigma, dirPriorAlphas, rawData, outDir, Kmax, mCycles,
burnCycles, g, h, alpha_sigma, beta_sigma, q, normalize,
thinning, zStart, nIterPerCycle, gibbs_z,
warm_up_overfitting, warm_up, overfittingInitialization,
progressGraphs, gwar)
```

## Arguments

sameSigma	Logical value denoting the parameterization of the error variance per component. If TRUE, the parameterization CCU is fitted. Otherwise, the parameterization CUU is fitted.
dirPriorAlphas	The prior Dirichlet parameters for each chain.
rawData	The observed data as an $n \times p$ matrix. Clustering is performed on the rows of the matrix.
outDir	Name of the output folder.
Kmax	Number of components in the overfitted mixture. Default: 20.
mCycles	Number of MCMC cycles. Each cycle consists of nIterPerCycle MCMC iterations. At the end of each cycle a swap of the state of two randomly chosen adjacent chains is attempted.
burnCycles	Number of cycles that will be discarded as burn-in period.
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .

<code>q</code>	Number of factors $q$ , where $1 \leq q \leq L$ . An error is thrown if the Ledermann bound ( $L$ ) is exceeded.
<code>normalize</code>	Should the observed data be normalized? Default value: TRUE.
<code>thinning</code>	Optional integer denoting the thinning of the kept MCMC cycles.
<code>zStart</code>	Optional starting value for the allocation vector.
<code>nIterPerCycle</code>	Number of iteration per MCMC cycle. Default value: 10.
<code>gibbs_z</code>	Select the gibbs sampling scheme for updating latent allocations of mixture model. Default value: 1.
<code>warm_up_overfitting</code>	Number of iterations for the overfitting initialization scheme. Default value: 100.
<code>warm_up</code>	Number of iterations that will be used to initialize the models before starting proposing switchings. Default value: 500.
<code>overfittingInitialization</code>	Logical value indicating whether the chains are initialized via the overfitting initialization scheme. Default: TRUE.
<code>progressGraphs</code>	Logical value indicating whether to plot successive states of the chains while the sampler runs. Default: FALSE.
<code>gwar</code>	Initialization parameter. Default: 0.05.

**Value**

List of files written to `outDir`

**Note**

It is recommended to always use: `normalize = TRUE` (default). Tuning of `dirPriorAlphas` may be necessary to achieve reasonable acceptance rates of chain swaps. Also note that the output is not identifiable due to label switching and the user has to subsequently call the `dealWithLabelSwitching` function.

**Author(s)**

Panagiotis Papastamoulis

**References**

Papastamoulis, P. (2017). Overfitting Bayesian mixtures of factor analyzers with an unknown number of components. arXiv:1701.04605 [stat.ME]

**See Also**

[dealWithLabelSwitching](#)

## Examples

```
## Not run:
# simulate a synthetic dataset along the lines of the paper:
n = 1000          # sample size
p = 40           # number of variables
q = 4            # number of factors
K = 10           # number of clusters
sINV_diag = 1/((1:p)) # diagonal of inverse variance of errors
set.seed(10)
syntheticDataset <- simData(K.true = K, n = n, q = q, p = p, sINV_values = sINV_diag )

# define parameters
Kmax <- 20      # number of overfitted mixture components
nChains <- 8    # number of parallel chains
dN <- 1
# Dirichlet prior of mixture weights per chain.
# The target chain corresponds to the first entry.
dirPriorAlphas <- c(1, 1 + dN * (2:nChains - 1))/Kmax
outputFolder <- "fabMixExample"
# Run algorithm
fabMix_CxU( dirPriorAlphas = dirPriorAlphas,
            rawData = syntheticDataset$data,
            outDir = outputFolder, Kmax = Kmax, mCycles = 1200,
            burnCycles = 200, q = q)

# Compute information criteria:
#getStuffForDIC(x_data = syntheticDataset$data, outputFolder = outputFolder, q = q)

# Deal with label switching:
#dealWithLabelSwitching(x_data = syntheticDataset$data,
#                        outputFolder = outputFolder, q = q,
#                        compute_regularized_expression = TRUE, Km = Kmax)

## End(Not run)
```

---

fabMix\_missing\_values *Main function for the case of missing values*

---

## Description

This function runs parallel chains under a prior tempering scheme of the Dirichlet prior distribution of mixture weights. Missing values are simulated from their full conditional posterior distribution.

## Usage

```
fabMix_missing_values(sameSigma, dirPriorAlphas, rawData, outDir, Kmax, mCycles,
burnCycles, g, h, alpha_sigma, beta_sigma, q, normalize,
thinning, zStart, nIterPerCycle, gibbs_z, warm_up,
progressGraphs, gwar)
```

**Arguments**

sameSigma	Logical value denoting the parameterization of the error variance per component. If TRUE, the parameterization $\Sigma_1 = \dots = \Sigma_K$ is fitted.
dirPriorAlphas	The prior Dirichlet parameters for each chain.
rawData	The observed data as an $n \times p$ matrix. Clustering is performed on the rows of the matrix.
outDir	Name of the output folder.
Kmax	Number of components in the overfitted mixture. Default: 20.
mCycles	Number of MCMC cycles. Each cycle consists of nIterPerCycle MCMC iterations. At the end of each cycle a swap of the state of two randomly chosen adjacent chains is attempted.
burnCycles	Number of cycles that will be discarded as burn-in period.
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
q	Number of factors $q$ , where $1 \leq q \leq L$ . An error is thrown if the Ledermann bound ( $L$ ) is exceeded.
normalize	Should the observed data be normalized? Default value: TRUE.
thinning	Optional integer denoting the thinning of the kept MCMC cycles.
zStart	Optional starting value for the allocation vector.
nIterPerCycle	Number of iteration per MCMC cycle. Default value: 10.
gibbs_z	Select the gibbs sampling scheme for updating latent allocations of mixture model. Default value: 1.
warm_up	Number of iterations that will be used to initialize the models before starting proposing switchings. Default value: 500.
progressGraphs	Logical value indicating whether to plot successive states of the chains while the sampler runs. Default: FALSE.
gwar	Initialization parameter. Default: 0.05.

**Value**

List of files written to outDir

**Note**

It is recommended to always use: `normalize = TRUE` (default). Tuning of `dirPriorAlphas` may be necessary to achieve reasonable acceptance rates of chain swaps. Also note that the output is not identifiable due to label switching and the user has to subsequently call the `dealWithLabelSwitching` function.

**Author(s)**

Panagiotis Papastamoulis

## References

Papastamoulis, P. (2017). Overfitting Bayesian mixtures of factor analyzers with an unknown number of components. arXiv:1701.04605 [stat.ME]

## See Also

[dealWithLabelSwitching](#)

## Examples

```
## Not run:
# simulate a synthetic dataset along the lines of the paper:
n = 1000          # sample size
p = 40           # number of variables
q = 4            # number of factors
K = 10           # number of clusters
sINV_diag = 1/((1:p)) # diagonal of inverse variance of errors
set.seed(10)
syntheticDataset <- simData(K.true = K, n = n, q = q, p = p, sINV_values = sINV_diag )

# define parameters
Kmax <- 20      # number of overfitted mixture components
nChains <- 8    # number of parallel chains
dN <- 1
# Dirichlet prior of mixture weights per chain.
# The target chain corresponds to the first entry.
dirPriorAlphas <- c(1, 1 + dN * (2:nChains - 1))/Kmax
outputFolder <- "fabMixExample"
# Run algorithm
fabMix( dirPriorAlphas = dirPriorAlphas,
        rawData = syntheticDataset$data,
        outDir = outputFolder, Kmax = Kmax, mCycles = 1200,
        burnCycles = 200, q = q)

# Compute information criteria:
getStuffForDIC(x_data = syntheticDataset$data, outputFolder = outputFolder, q = q)

# Deal with label switching:
dealWithLabelSwitching(x_data = syntheticDataset$data,
                       outputFolder = outputFolder, q = q,
                       compute_regularized_expression = TRUE, Km = Kmax)

## End(Not run)
```



## Description

This function runs parallel chains under a prior tempering scheme of the Dirichlet prior distribution of mixture weights.

## Usage

```
fabMix_UxC(sameSigma, dirPriorAlphas, rawData, outDir, Kmax, mCycles,
burnCycles, g, h, alpha_sigma, beta_sigma, q, normalize,
thinning, zStart, nIterPerCycle, gibbs_z,
warm_up_overfitting, warm_up, overfittingInitialization,
progressGraphs, gwar)
```

## Arguments

sameSigma	Logical value denoting the parameterization of the error variance per component. If TRUE, the parameterization CCU is fitted. Otherwise, the parameterization CUU is fitted.
dirPriorAlphas	The prior Dirichlet parameters for each chain.
rawData	The observed data as an $n \times p$ matrix. Clustering is performed on the rows of the matrix.
outDir	Name of the output folder.
Kmax	Number of components in the overfitted mixture. Default: 20.
mCycles	Number of MCMC cycles. Each cycle consists of nIterPerCycle MCMC iterations. At the end of each cycle a swap of the state of two randomly chosen adjacent chains is attempted.
burnCycles	Number of cycles that will be discarded as burn-in period.
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
q	Number of factors $q$ , where $1 \leq q \leq L$ . An error is thrown if the Ledermann bound ( $L$ ) is exceeded.
normalize	Should the observed data be normalized? Default value: TRUE.
thinning	Optional integer denoting the thinning of the kept MCMC cycles.
zStart	Optional starting value for the allocation vector.
nIterPerCycle	Number of iteration per MCMC cycle. Default value: 10.
gibbs_z	Select the gibbs sampling scheme for updating latent allocations of mixture model. Default value: 1.
warm_up_overfitting	Number of iterations for the overfitting initialization scheme. Default value: 100.
warm_up	Number of iterations that will be used to initialize the models before starting proposing switchings. Default value: 500.

overfittingInitialization	Logical value indicating whether the chains are initialized via the overfitting initialization scheme. Default: TRUE.
progressGraphs	Logical value indicating whether to plot successive states of the chains while the sampler runs. Default: FALSE.
gwar	Initialization parameter. Default: 0.05.

**Value**

List of files written to outDir

**Note**

It is recommended to always use: `normalize = TRUE` (default). Tuning of `dirPriorAlphas` may be necessary to achieve reasonable acceptance rates of chain swaps. Also note that the output is not identifiable due to label switching and the user has to subsequently call the `dealWithLabelSwitching` function.

**Author(s)**

Panagiotis Papastamoulis

**References**

Papastamoulis, P. (2017). Overfitting Bayesian mixtures of factor analyzers with an unknown number of components. arXiv:1701.04605 [stat.ME]

**See Also**

[dealWithLabelSwitching](#)

**Examples**

```
## Not run:
# simulate a synthetic dataset along the lines of the paper:
n = 1000          # sample size
p = 40            # number of variables
q = 4             # number of factors
K = 10            # number of clusters
sINV_diag = 1/((1:p)) # diagonal of inverse variance of errors
set.seed(10)
syntheticDataset <- simData(K.true = K, n = n, q = q, p = p, sINV_values = sINV_diag )

# define parameters
Kmax <- 20        # number of overfitted mixture components
nChains <- 8      # number of parallel chains
dN <- 1
# Dirichlet prior of mixture weights per chain.
# The target chain corresponds to the first entry.
dirPriorAlphas <- c(1, 1 + dN * (2:nChains - 1))/Kmax
outputFolder <- "fabMixExample"
```

```

# Run algorithm
fabMix_UxC( dirPriorAlphas = dirPriorAlphas,
            rawData = syntheticDataset$data,
            outDir = outputFolder, Kmax = Kmax, mCycles = 1200,
            burnCycles = 200, q = q)

# Compute information criteria:
#getStuffForDIC(x_data = syntheticDataset$data, outputFolder = outputFolder, q = q)

# Deal with label switching:
#dealWithLabelSwitching(x_data = syntheticDataset$data,
#                       outputFolder = outputFolder, q = q,
#                       compute_regularized_expression = TRUE, Km = Kmax)

## End(Not run)

```

fabMix\_UxU

*Fit UxU mixtures*

## Description

This function runs parallel chains under a prior tempering scheme of the Dirichlet prior distribution of mixture weights.

## Usage

```

fabMix_UxU(sameSigma, dirPriorAlphas, rawData, outDir, Kmax, mCycles,
            burnCycles, g, h, alpha_sigma, beta_sigma, q, normalize,
            thinning, zStart, nIterPerCycle, gibbs_z,
            warm_up_overfitting, warm_up, overfittingInitialization,
            progressGraphs, gwar)

```

## Arguments

sameSigma	Logical value denoting the parameterization of the error variance per component. If TRUE, the parameterization $\Sigma_1 = \dots = \Sigma_K$ is fitted.
dirPriorAlphas	The prior Dirichlet parameters for each chain.
rawData	The observed data as an $n \times p$ matrix. Clustering is performed on the rows of the matrix.
outDir	Name of the output folder.
Kmax	Number of components in the overfitted mixture. Default: 20.
mCycles	Number of MCMC cycles. Each cycle consists of nIterPerCycle MCMC iterations. At the end of each cycle a swap of the state of two randomly chosen adjacent chains is attempted.
burnCycles	Number of cycles that will be discarded as burn-in period.

<code>g</code>	Prior parameter $g$ . Default value: $g = 2$ .
<code>h</code>	Prior parameter $h$ . Default value: $h = 1$ .
<code>alpha_sigma</code>	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
<code>beta_sigma</code>	Prior parameter $\beta$ . Default value: $\beta = 1$ .
<code>q</code>	Number of factors $q$ , where $1 \leq q \leq L$ . An error is thrown if the Ledermann bound ( $L$ ) is exceeded.
<code>normalize</code>	Should the observed data be normalized? Default value: TRUE.
<code>thinning</code>	Optional integer denoting the thinning of the kept MCMC cycles.
<code>zStart</code>	Optional starting value for the allocation vector.
<code>nIterPerCycle</code>	Number of iteration per MCMC cycle. Default value: 10.
<code>gibbs_z</code>	Select the gibbs sampling scheme for updating latent allocations of mixture model. Default value: 1.
<code>warm_up_overfitting</code>	Number of iterations for the overfitting initialization scheme. Default value: 100.
<code>warm_up</code>	Number of iterations that will be used to initialize the models before starting proposing switchings. Default value: 500.
<code>overfittingInitialization</code>	Logical value indicating whether the chains are initialized via the overfitting initialization scheme. Default: TRUE.
<code>progressGraphs</code>	Logical value indicating whether to plot successive states of the chains while the sampler runs. Default: FALSE.
<code>gwar</code>	Initialization parameter. Default: 0.05.

**Value**

List of files written to `outDir`

**Note**

It is recommended to always use: `normalize = TRUE` (default). Tuning of `dirPriorAlphas` may be necessary to achieve reasonable acceptance rates of chain swaps. Also note that the output is not identifiable due to label switching and the user has to subsequently call the `dealWithLabelSwitching` function.

**Author(s)**

Panagiotis Papastamoulis

**References**

Papastamoulis, P. (2017). Overfitting Bayesian mixtures of factor analyzers with an unknown number of components. arXiv:1701.04605 [stat.ME]

**See Also**

[dealWithLabelSwitching](#)

**Examples**

```
## Not run:
# simulate a synthetic dataset along the lines of the paper:
n = 1000          # sample size
p = 40           # number of variables
q = 4            # number of factors
K = 10           # number of clusters
sINV_diag = 1/((1:p)) # diagonal of inverse variance of errors
set.seed(10)
syntheticDataset <- simData(K.true = K, n = n, q = q, p = p, sINV_values = sINV_diag )

# define parameters
Kmax <- 20      # number of overfitted mixture components
nChains <- 8    # number of parallel chains
dN <- 1
# Dirichlet prior of mixture weights per chain.
# The target chain corresponds to the first entry.
dirPriorAlphas <- c(1, 1 + dN * (2:nChains - 1))/Kmax
outputFolder <- "fabMixExample"
# Run algorithm
fabMix_UxU( dirPriorAlphas = dirPriorAlphas,
            rawData = syntheticDataset$data,
            outDir = outputFolder, Kmax = Kmax, mCycles = 1200,
            burnCycles = 200, q = q)

# Compute information criteria:
getStuffForDIC(x_data = syntheticDataset$data, outputFolder = outputFolder, q = q)

# Deal with label switching:
dealWithLabelSwitching(x_data = syntheticDataset$data,
                      outputFolder = outputFolder, q = q,
                      compute_regularized_expression = TRUE, Km = Kmax)

## End(Not run)
```

---

getStuffForDIC

---

*Compute information criteria*


---

**Description**

This function computes four information criteria for a given run of the fabMix algorithm, namely: AIC, BIC, DIC and DIC<sub>2</sub>. Given various runs with different number of factors, the selected model corresponds to the one with the smallest value of the selected criterion.

**Usage**

```
getStuffForDIC(sameSigma, sameLambda, isotropic, x_data, outputFolder, q, burn,  
Km, normalize, discardLower)
```

**Arguments**

sameSigma	Logical value indicating whether the parameterization with the same variance of errors per component is used. Default: TRUE.
sameLambda	Logical value indicating whether the parameterization with same loadings per component is used. Default: FALSE.
isotropic	Logical value indicating whether the parameterization with isotropic error variance per component is used. Default: FALSE.
x_data	Observed data.
outputFolder	Name of the folder where the fabMix function has saved its output.
q	Number of factors. Note that this should coincide with the number of factors in the fabMix run.
burn	Discard observations as burn-in period (optional).
Km	Number of components in the overfitted mixture model. Note that this should coincide with the same entry in the fabMix run.
normalize	Should the observed data be normalized? Note that this should coincide with the same entry in the fabMix run. Default value: TRUE.
discardLower	Discard draws with log-likelihood values lower than the specific quantile. This applied only for the DIC computation.

**Details**

If necessary, more details than the description above

**Value**

The information criteria are saved to the `informationCriteria_map_model.txt` file in the `code-outputFolder`.

**Note**

It is well known that DIC tends to overfit, so it is advised to compare models with different number of factors using AIC or BIC.

**Author(s)**

Panagiotis Papastamoulis

---

log_dirichlet_pdf	<i>Log-density function of the Dirichlet distribution</i>
-------------------	---

---

**Description**

Log-density function of the Dirichlet distribution

**Usage**

```
log_dirichlet_pdf(alpha, weights)
```

**Arguments**

alpha	Parameter vector
weights	Vector of weights

**Value**

Log-density of the  $D(\alpha_1, \dots, \alpha_k)$  evaluated at  $w_1, \dots, w_k$ .

**Author(s)**

Panagiotis Papastamoulis

---

myDirichlet	<i>Simulate from the Dirichlet distribution</i>
-------------	---

---

**Description**

Generate a random draw from the Dirichlet distribution  $D(\alpha_1, \dots, \alpha_k)$ .

**Usage**

```
myDirichlet(alpha)
```

**Arguments**

alpha	Parameter vector
-------	------------------

**Value**

Simulated vector

**Author(s)**

Panagiotis Papastamoulis

observed.log.likelihood0

*Log-likelihood of the mixture model*

---

### Description

Log-likelihood of the mixture model evaluated only at the alive components.

### Usage

```
observed.log.likelihood0(x_data, w, mu, Lambda, Sigma, z)
```

### Arguments

x_data	The observed data
w	Vector of mixture weights
mu	Vector of marginal means
Lambda	Factor loadings
Sigma	Common covariance matrix of the errors per cluster
z	Allocation vector

### Value

Log-likelihood value

### Author(s)

Panagiotis Papastamoulis

---

observed.log.likelihood0\_q0\_sameSigma

*Log-likelihood of the mixture model for  $q = 0$  and same variance of errors*

---

### Description

Log-likelihood of the mixture model evaluated only at the alive components.

### Usage

```
observed.log.likelihood0_q0_sameSigma(x_data, w, mu, Sigma, z)
```



**Arguments**

x_data	The observed data
w	Vector of mixture weights
mu	Vector of marginal means
Sigma	Covariance matrix of the errors per cluster
z	Allocation vector

**Value**

Log-likelihood value

**Author(s)**

Panagiotis Papastamoulis

---

observed.log.likelihood0\_Sj

*Log-likelihood of the mixture model*

---

**Description**

Log-likelihood of the mixture model evaluated only at the alive components.

**Usage**

```
observed.log.likelihood0_Sj(x_data, w, mu, Lambda, Sigma, z)
```

**Arguments**

x_data	The observed data
w	Vector of mixture weights
mu	Vector of marginal means
Lambda	Factor loadings
Sigma	Covariance matrix of the errors per cluster
z	Allocation vector

**Value**

Log-likelihood value

**Author(s)**

Panagiotis Papastamoulis

---

```
observed.log.likelihood0_Sj_q0
```

*Log-likelihood of the mixture model for  $q = 0$*

---

### Description

Log-likelihood of the mixture model evaluated only at the alive components.

### Usage

```
observed.log.likelihood0_Sj_q0(x_data, w, mu, Sigma, z)
```

### Arguments

x_data	The observed data
w	Vector of mixture weights
mu	Vector of marginal means
Sigma	Covariance matrix of the errors per cluster
z	Allocation vector

### Value

Log-likelihood value

### Author(s)

Panagiotis Papastamoulis

---

```
overfittingMFA
```

*Basic MCMC sampler*

---

### Description

Gibbs sampling for fitting a mixture model of factor analyzers.

### Usage

```
overfittingMFA(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma,
start_values, q, zStart, gibbs_z)
```

**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfittingMFA\_CCC      *Basic MCMC sampler for CCC*

---

**Description**

Gibbs sampling for fitting a CCC mixture model of factor analyzers.

**Usage**

```
overfittingMFA_CCC(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma,
start_values, q, zStart, gibbs_z)
```

**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfittingMFA_CCU	<i>Basic MCMC sampler for CCU</i>
--------------------	-----------------------------------

---

**Description**

Gibbs sampling for fitting a CCU mixture model of factor analyzers.

**Usage**

```
overfittingMFA_CCU(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma,
start_values, q, zStart, gibbs_z)
```

**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfittingMFA\_CUC      *Basic MCMC sampler for CUC*

---

**Description**

Gibbs sampling for fitting a CUC mixture model of factor analyzers.

**Usage**

```
overfittingMFA_CUC(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma,
start_values, q, zStart, gibbs_z)
```

**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfittingMFA_CUU	<i>Basic MCMC sampler for CUU</i>
--------------------	-----------------------------------

---

**Description**

Gibbs sampling for fitting a CUU mixture model of factor analyzers.

**Usage**

```
overfittingMFA_CUU(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma,
start_values, q, zStart, gibbs_z)
```

**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfittingMFA\_missing\_values

*Basic MCMC sampler for the case of missing data*

---

**Description**

Gibbs sampling for fitting a mixture model of factor analyzers.

**Usage**

```
overfittingMFA_missing_values(missing_entries, x_data, originalX, outputDirectory, Kmax,
m, thinning, burn, g, h, alpha_prior, alpha_sigma,
beta_sigma, start_values, q, zStart, gibbs_z)
```

**Arguments**

missing_entries	list which contains the row number (1st entry) and column indexes (subsequent entries) for every row containing missing values.
x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfittingMFA_Sj	<i>Basic MCMC sampler using different error variance per component</i>
-------------------	--

---

**Description**

Gibbs sampling for fitting a mixture model of factor analyzers.

**Usage**

```
overfittingMFA_Sj(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma,
start_values, q, zStart, gibbs_z)
```



**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfittingMFA\_Sj\_missing\_values

*Basic MCMC sampler for the case of missing data and different error variance*

---

**Description**

Gibbs sampling for fitting a mixture model of factor analyzers.

**Usage**

```
overfittingMFA_Sj_missing_values(missing_entries, x_data, originalX,
outputDirectory, Kmax,
m, thinning, burn, g, h, alpha_prior, alpha_sigma,
beta_sigma, start_values, q, zStart, gibbs_z)
```

**Arguments**

missing_entries	list which contains the row number (1st entry) and column indexes (subsequent entries) for every row containing missing values.
x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfittingMFA\_UCC      *Basic MCMC sampler for CCC*

---

**Description**

Gibbs sampling for fitting a UCC mixture model of factor analyzers.

**Usage**

```
overfittingMFA_UCC(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma,
start_values, q, zStart, gibbs_z)
```

**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfittingMFA_UUC	<i>Basic MCMC sampler for UUC</i>
--------------------	-----------------------------------

---

**Description**

Gibbs sampling for fitting a UUC mixture model of factor analyzers.

**Usage**

```
overfittingMFA_UUC(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma,
start_values, q, zStart, gibbs_z)
```

**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfitting_q0	<i>MCMC sampler for <math>q = 0</math></i>
----------------	--

---

**Description**

Gibbs sampling for fitting a mixture model with diagonal covariance structure.

**Usage**

```
overfitting_q0(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma, start_values, q, zStart, gibbs_z)
```

**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

overfitting\_q0\_sameSigma

*MCMC sampler for  $q = 0$  and same error variance parameterization*

---

**Description**

Gibbs sampling for fitting a mixture model with diagonal covariance structure.

**Usage**

```
overfitting_q0_sameSigma(x_data, originalX, outputDirectory, Kmax, m, thinning, burn,
g, h, alpha_prior, alpha_sigma, beta_sigma, start_values, q, zStart, gibbs_z)
```

**Arguments**

x_data	normalized data
originalX	observed raw data (only for plotting purpose)
outputDirectory	Name of the output folder
Kmax	Number of mixture components
m	Number of iterations
thinning	Thinning of chain
burn	Burn-in period
g	Prior parameter $g$ . Default value: $g = 2$ .
h	Prior parameter $h$ . Default value: $h = 1$ .
alpha_prior	Parameters of the Dirichlet prior distribution of mixture weights.
alpha_sigma	Prior parameter $\alpha$ . Default value: $\alpha = 2$ .
beta_sigma	Prior parameter $\beta$ . Default value: $\beta = 1$ .
start_values	Optional (not used)
q	Number of factors.
zStart	Optional (not used)
gibbs_z	Optional

**Value**

List of files

**Author(s)**

Panagiotis Papastamoulis

---

plot.fabMix.object	<i>Plot function</i>
--------------------	----------------------

---

**Description**

This function plots fabMix function.

**Usage**

```
## S3 method for class 'fabMix.object'
plot(x, what, variableSubset, ...)
```

**Arguments**

x	An object of class fabMix.object, which is returned by the fabMix function.
what	One of the "BIC", "classification_matplot", "classification_pairs", "correlation", "regularized_expression". The plot will display the BIC values per model and number of factors (along with the most probable number of clusters as text), a matplot per cluster for the selected model, scatterplots pairs, the estimated correlation matrix per cluster, and the estimated regularized expression of each variable to the factor space for the selected model, respectively.
variableSubset	An optional subset of the variables. By default, all variables are selected.
...	ignored.

**Details**

The function plots the BIC values obtained for each of the fitted models, visualizes the clusters.

**Author(s)**

Panagiotis Papastamoulis

---

print.fabMix.object     *Print function*

---

**Description**

This function prints a summary of objects returned by the fabMix function.

**Usage**

```
## S3 method for class 'fabMix.object'
print(x, printSubset, ...)
```

**Arguments**

x	An object of class fabMix.object, which is returned by the fabMix function.
printSubset	Logical indicating whether to print the header or the whole matrix of estimates. Default value: TRUE.
...	ignored.

**Details**

The function prints the estimated distribution of the number of clusters, the estimated number of observations assigned to each cluster after post-processing the output with three label switching algorithms, as well as the header of the posterior mean estimates of  $\theta_{kj}$  (probability of success for cluster  $k$  and feature  $j$ ) (conditionally on the most probable number of clusters).

**Author(s)**

Panagiotis Papastamoulis

---

simData	<i>Synthetic data generator</i>
---------	---------------------------------

---

### Description

Simulate data from a multivariate normal mixture using a mixture of factor analyzers mechanism.

### Usage

```
simData(sameSigma, sameLambda, p, q, K.true, n, loading_means, loading_sd, sINV_values)
```

### Arguments

sameSigma	Logical.
sameLambda	Logical.
p	The dimension of the multivariate normal distribution ( $p > 1$ ).
q	Number of factors. It should be strictly smaller than p.
K.true	The number of mixture components (clusters).
n	Sample size.
loading_means	A vector which contains the means of blocks of factor loadings. Default: <code>loading_means = c(-30,-20,-10,10, 20, 30)</code> .
loading_sd	A vector which contains the standard deviations of blocks of factor loadings. Default: <code>loading_sd &lt;- rep(2, length(loading_means))</code> .
sINV_values	A vector which contains the values of the diagonal of the (common) inverse covariance matrix, if <code>sigmaTrue = TRUE</code> . An $K \times p$ matrix which contains the values of the diagonal of the inverse covariance matrix per component, if <code>sigmaTrue = FALSE</code> . Default: <code>sINV_values = rgamma(p, shape = 1, rate = 1)</code> .

### Value

A list with the following entries:

data	$n \times p$ array containing the simulated data.
class	$n$ -dimensional vector containing the class of each observation.
factorLoadings	$K.true \times p \times q$ -array containing the factor loadings $\Lambda_{krj}$ per cluster $k$ , feature $r$ and factor $j$ , where $k = 1, \dots, K$ ; $r = 1, \dots, p$ ; $j = 1, \dots, q$ .
means	$K.true \times p$ matrix containing the marginal means $\mu_{kr}$ , $k = 1, \dots, K$ ; $r = 1, \dots, p$ .
variance	$p \times p$ diagonal matrix containing the variance of errors $\sigma_{rr}$ , $r = 1, \dots, p$ . Note that the same variance of errors is assumed for each cluster.
factors	$n \times q$ matrix containing the simulated factor values.
weights	$K.true$ -dimensional vector containing the weight of each cluster.



**Note**

The marginal variance for cluster  $k$  is equal to  $\Lambda_k \Lambda_k^T + \Sigma$ .

**Author(s)**

Panagiotis Papastamoulis

---

update_all_y	<i>Gibbs sampling for y</i>
--------------	-----------------------------

---

**Description**

Gibbs sampling for  $y$

**Usage**

```
update_all_y(x_data, mu, SigmaINV, Lambda, z)
```

**Arguments**

x_data	Data
mu	Marginal means
SigmaINV	Precision matrix
Lambda	Factor loadings
z	Allocation vector

**Value**

A matrix with generated factors

**Author(s)**

Panagiotis Papastamoulis

---

update_all_y_Sj	<i>Gibbs sampling for <math>y</math></i>
-----------------	--

---

**Description**

Gibbs sampling for  $y$

**Usage**

update\_all\_y\_Sj(x\_data, mu, SigmaINV, Lambda, z)

**Arguments**

x_data	Data
mu	Marginal means
SigmaINV	Precision matrix per component
Lambda	Factor loadings
z	Allocation vector

**Value**

A matrix with generated factors

**Author(s)**

Panagiotis Papastamoulis

---

update_OmegaINV	<i>Gibbs sampling for <math>\Omega^{-1}</math></i>
-----------------	--

---

**Description**

Gibbs sampling for  $\Omega^{-1}$

**Usage**

update\_OmegaINV(Lambda, K, g, h)

**Arguments**

Lambda	Factor loadings
K	Number of components
g	Prior parameter
h	Prior parameter

**Value**

$$\Omega^{-1}$$

**Author(s)**

Panagiotis Papastamoulis

---

update_OmegaINV_Cxx	<i>Gibbs sampling for <math>\hat{\Omega}^{-1}</math> for Cxx model</i>
---------------------	--

---

**Description**

Gibbs sampling for  $\Omega^{-1}$  for Cxx model

**Usage**

update\_OmegaINV\_Cxx(Lambda, K, g, h)

**Arguments**

Lambda	Factor loadings
K	Number of components
g	Prior parameter
h	Prior parameter

**Value**

$$\Omega^{-1}$$

**Author(s)**

Panagiotis Papastamoulis

---

update\_SigmaINV\_faster

*Gibbs sampling for  $\Sigma^{-1}$* 


---

**Description**

Gibbs sampling for  $\Sigma^{-1}$

**Usage**

```
update_SigmaINV_faster(x_data, z, y, Lambda, mu, K, alpha_sigma, beta_sigma)
```

**Arguments**

x_data	Data
z	Allocation vector
y	Factors
Lambda	Factor loadings
mu	Marginal means
K	Number of components
alpha_sigma	Prior parameter
beta_sigma	Prior parameter

**Value**

$\Sigma^{-1}$

**Author(s)**

Panagiotis Papastamoulis

---

update\_SigmaINV\_faster\_q0

*Gibbs sampling for  $\Sigma^{-1}$  per component for  $q = 0$* 


---

**Description**

Gibbs sampling for  $\Sigma^{-1}$  per component

**Usage**

```
update_SigmaINV_faster_q0(z, mu, K, alpha_sigma, beta_sigma, x_data)
```

**Arguments**

z	Allocation vector
mu	Marginal means
K	Number of components
alpha_sigma	Prior parameter
beta_sigma	Prior parameter
x_data	Data

**Value** $\Sigma^{-1}$ **Author(s)**

Panagiotis Papastamoulis

---

`update_SigmaINV_faster_q0_sameSigma`*Gibbs sampling for  $\Sigma^{-1}$  per component for  $q = 0$* 

---

**Description**Gibbs sampling for  $\Sigma^{-1}$  per component**Usage**`update_SigmaINV_faster_q0_sameSigma( z, mu, K, alpha_sigma, beta_sigma, x_data)`**Arguments**

z	Allocation vector
mu	Marginal means
K	Number of components
alpha_sigma	Prior parameter
beta_sigma	Prior parameter
x_data	Data

**Value** $\Sigma^{-1}$ **Author(s)**

Panagiotis Papastamoulis

---

update\_SigmaINV\_faster\_Sj

*Gibbs sampling for  $\Sigma^{-1}$  per component*


---

**Description**

Gibbs sampling for  $\Sigma^{-1}$  per component

**Usage**

```
update_SigmaINV_faster_Sj(x_data, z, y, Lambda, mu, K, alpha_sigma, beta_sigma)
```

**Arguments**

x_data	Data
z	Allocation vector
y	Factors
Lambda	Factor loadings
mu	Marginal means
K	Number of components
alpha_sigma	Prior parameter
beta_sigma	Prior parameter

**Value**

$\Sigma^{-1}$

**Author(s)**

Panagiotis Papastamoulis

---

update\_SigmaINV\_xCC

*Gibbs sampling for  $\Sigma^{-1}$  for xCC models*


---

**Description**

Gibbs sampling for  $\Sigma^{-1}$  for xCC models

**Usage**

```
update_SigmaINV_xCC(x_data, z, y, Lambda, mu, K, alpha_sigma, beta_sigma)
```

**Arguments**

x_data	Data
z	Allocation vector
y	Factors
Lambda	Factor loadings
mu	Marginal means
K	Number of components
alpha_sigma	Prior parameter
beta_sigma	Prior parameter

**Value**

$$\Sigma^{-1}$$

**Author(s)**

Panagiotis Papastamoulis

---

update\_SigmaINV\_xUC     *Gibbs sampling for  $\Sigma^{-1}$  per component for xUC models*

---

**Description**

Gibbs sampling for  $\Sigma^{-1}$  per component for xUC models

**Usage**

```
update_SigmaINV_xUC(x_data, z, y, Lambda, mu, K, alpha_sigma, beta_sigma)
```

**Arguments**

x_data	Data
z	Allocation vector
y	Factors
Lambda	Factor loadings
mu	Marginal means
K	Number of components
alpha_sigma	Prior parameter
beta_sigma	Prior parameter

**Value**

$$\Sigma^{-1}$$

**Author(s)**

Panagiotis Papastamoulis

---

update_z2	<i>Collapsed Gibbs for <math>z</math> using matrix inversion lemma</i>
-----------	--

---

**Description**

Collapsed Gibbs for  $z$  using matrix inversion lemma

**Usage**

update\_z2(w, mu, Lambda, SigmaINV, K, x\_data)

**Arguments**

w	Mixture weights
mu	Marginal means
Lambda	Factor loadings
SigmaINV	Precision matrix
K	Number of components
x_data	Data

**Value**

Allocation vector

**Author(s)**

Panagiotis Papastamoulis

---

update_z2_Sj	<i>Collapsed Gibbs for <math>z</math> using matrix inversion lemma</i>
--------------	--

---

**Description**

Collapsed Gibbs for  $z$  using matrix inversion lemma

**Usage**

update\_z2\_Sj(w, mu, Lambda, SigmaINV, K, x\_data)



**Arguments**

w	Mixture weights
mu	Marginal means
Lambda	Factor loadings
SigmaINV	Precision matrix per component
K	Number of components
x_data	Data

**Value**

Allocation vector

**Author(s)**

Panagiotis Papastamoulis

---

update_z4	<i>Collapsed Gibbs for <math>z</math></i>
-----------	---

---

**Description**

Collapsed Gibbs for  $z$

**Usage**

update\_z4(w, mu, Lambda, SigmaINV, K, x\_data)

**Arguments**

w	Mixture weights
mu	Marginal means
Lambda	Factor loadings
SigmaINV	Precision matrix
K	Number of components
x_data	Data

**Value**

Allocation vector

**Author(s)**

Panagiotis Papastamoulis

---

update_z4_Sj	<i>Collapsed Gibbs for <math>z</math></i>
--------------	---

---

**Description**

Collapsed Gibbs for  $z$

**Usage**

```
update_z4_Sj(w, mu, Lambda, SigmaINV, K, x_data)
```

**Arguments**

w	Mixture weights
mu	Marginal means
Lambda	Factor loadings
SigmaINV	Precision matrix per component
K	Number of components
x_data	Data

**Value**

Allocation vector

**Author(s)**

Panagiotis Papastamoulis

---

update_z_b	<i>Gibbs sampling for <math>z</math></i>
------------	--

---

**Description**

Gibbs sampling for  $z$

**Usage**

```
update_z_b(w, mu, Lambda, y, SigmaINV, K, x_data)
```

**Arguments**

w	Mixture weights
mu	Marginal means
Lambda	Factor loadings
y	Matrix of factors
SigmaINV	Precision matrix
K	Number of components
x_data	Data

**Value**

Allocation vector

**Author(s)**

Panagiotis Papastamoulis

---

update_z_b_Sj	<i>Gibbs sampling for <math>z</math></i>
---------------	--

---

**Description**

Gibbs sampling for  $z$

**Usage**

```
update_z_b_Sj(w, mu, Lambda, y, SigmaINV, K, x_data)
```

**Arguments**

w	Mixture weights
mu	Marginal means
Lambda	Factor loadings
y	Matrix of factors
SigmaINV	Precision matrix per component
K	Number of components
x_data	Data

**Value**

Allocation vector

**Author(s)**

Panagiotis Papastamoulis

---

update_z_q0	<i>Gibbs sampling for <math>z</math> for <math>q = 0</math></i>
-------------	---

---

**Description**

Gibbs sampling for  $z$

**Usage**

update\_z\_q0(w, mu, SigmaINV, K, x\_data)

**Arguments**

w	Mixture weights
mu	Marginal means
SigmaINV	Precision matrix per component
K	Number of components
x_data	Data

**Value**

Allocation vector

**Author(s)**

Panagiotis Papastamoulis

---

update_z_q0_sameSigma	<i>Gibbs sampling for <math>z</math> for <math>q = 0</math></i>
-----------------------	---

---

**Description**

Gibbs sampling for  $z$

**Usage**

update\_z\_q0\_sameSigma(w, mu, SigmaINV, K, x\_data)

**Arguments**

w	Mixture weights
mu	Marginal means
SigmaINV	Precision matrix per component
K	Number of components
x_data	Data

**Value**

Allocation vector

**Author(s)**

Panagiotis Papastamoulis

---

waveDataset1500

*Wave dataset*

---

**Description**

A subset of 1500 randomly sampled observations from the wave dataset (version 1), available from the UCI machine learning repository. It contains 3 classes of waves (variable `class` with values “1”, “2” and “3”) and 21 attributes. Each class is generated from a combination of 2 of 3 base waves with noise.

**Usage**

`waveDataset1500`

**Format**

A data frame with 1500 rows and 22 columns. The first column denotes the class of each observation.

**Source**

[https://archive.ics.uci.edu/ml/datasets/Waveform+Database+Generator+\(Version+1\)](https://archive.ics.uci.edu/ml/datasets/Waveform+Database+Generator+(Version+1))

**References**

Lichman, M. (2013). UCI Machine Learning Repository <http://archive.ics.uci.edu/ml>. Irvine, CA: University of California, School of Information and Computer Science.

Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.J. (1984). Classification and Regression Trees. Wadsworth International Group: Belmont, California.

# Index

## \*Topic **datasets**

waveDataset1500, [61](#)

## \*Topic **package**

fabMix-package, [3](#)

complete.log.likelihood, [4](#)

complete.log.likelihood\_q0, [5](#)

complete.log.likelihood\_q0\_sameSigma,  
[6](#)

complete.log.likelihood\_Sj, [6](#)

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda,  
[7](#)

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_CCU,  
[8](#)

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_CUU,  
[9](#)

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_q0,  
[9](#)

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_q0\_sameSigma,  
[10](#)

compute\_A\_B\_G\_D\_and\_simulate\_mu\_Lambda\_Sj,  
[11](#)

compute\_sufficient\_statistics, [12](#)

compute\_sufficient\_statistics\_given\_mu,  
[12](#)

compute\_sufficient\_statistics\_q0, [13](#)

dealWithLabelSwitching, [3](#), [14](#), [17](#), [19](#), [21](#),  
[24](#), [26](#), [29](#)

fabMix, [3](#), [15](#)

fabMix-package, [3](#)

fabMix\_CxC, [17](#)

fabMix\_CxU, [20](#)

fabMix\_missing\_values, [22](#)

fabMix\_UxC, [24](#)

fabMix\_UxU, [27](#)

getStuffForDIC, [3](#), [29](#)

log\_dirichlet\_pdf, [31](#)

myDirichlet, [31](#)

observed.log.likelihood0, [32](#)

observed.log.likelihood0\_q0\_sameSigma,  
[32](#)

observed.log.likelihood0\_Sj, [33](#)

observed.log.likelihood0\_Sj\_q0, [34](#)

overfitting\_q0, [44](#)

overfitting\_q0\_sameSigma, [45](#)

overfittingMFA, [34](#)

overfittingMFA\_CCC, [35](#)

overfittingMFA\_CCU, [36](#)

overfittingMFA\_CUC, [37](#)

overfittingMFA\_CUU, [38](#)

overfittingMFA\_missing\_values, [39](#)

overfittingMFA\_Sj, [40](#)

overfittingMFA\_Sj\_missing\_values, [41](#)

overfittingMFA\_UCC, [42](#)

overfittingMFA\_UUC, [43](#)

plot.fabMix.object, [46](#)

print.fabMix.object, [47](#)

simData, [48](#)

update\_all\_y, [49](#)

update\_all\_y\_Sj, [50](#)

update\_OmegaINV, [50](#)

update\_OmegaINV\_Cxx, [51](#)

update\_SigmaINV\_faster, [52](#)

update\_SigmaINV\_faster\_q0, [52](#)

update\_SigmaINV\_faster\_q0\_sameSigma,  
[53](#)

update\_SigmaINV\_faster\_Sj, [54](#)

update\_SigmaINV\_xCC, [54](#)

update\_SigmaINV\_xUC, [55](#)

update\_z2, [56](#)

update\_z2\_Sj, [56](#)

update\_z4, [57](#)

update\_z4\_Sj, [58](#)

update\_z\_b, [58](#)  
update\_z\_b\_Sj, [59](#)  
update\_z\_q0, [60](#)  
update\_z\_q0\_sameSigma, [60](#)  
  
waveDataset1500, [61](#)