

Package: bpgmm (via r-universe)

October 5, 2024

Type Package

Title Bayesian Model Selection Approach for Parsimonious Gaussian Mixture Models

Version 1.0.9

Date 2022-05-31

Depends R(>= 3.1.0)

Imports methods (>= 3.5.1), mcmcse (>= 1.3-2), pgmm (>= 1.2.3), mvtnorm (>= 1.0-10), MASS (>= 7.3-51.1), Rcpp (>= 1.0.1), gtools (>= 3.8.1), label.switching (>= 1.8), fabMix (>= 5.0), mclust (>= 5.4.3)

Author Xiang Lu <Xiang_Lu at urmc.rochester.edu>, Yaoxiang Li <yl814 at georgetown.edu>, Tanzy Love <tanzy_love at urmc.rochester.edu>

Maintainer Yaoxiang Li <yl814@georgetown.edu>

Description Model-based clustering using Bayesian parsimonious Gaussian mixture models. MCMC (Markov chain Monte Carlo) are used for parameter estimation. The RJMCMC (Reversible-jump Markov chain Monte Carlo) is used for model selection. GREEN et al. (1995) <[doi:10.1093/biomet/82.4.711](https://doi.org/10.1093/biomet/82.4.711)>.

SystemRequirements C++11

License GPL-3

Encoding UTF-8

RoxygenNote 6.1.1

Suggests testthat

LinkingTo Rcpp, RcppArmadillo

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-06-01 10:20:12 UTC

Contents

| | |
|-------------------------|----|
| CalculateProposalLambda | 2 |
| CalculateProposalPsy | 6 |
| calculateRatio | 9 |
| calculateVarList | 10 |
| changeConstraintFormat | 11 |
| clearCurrentThetaYlist | 12 |
| combineClusterPara | 14 |
| evaluatePrior | 16 |
| evaluatePriorLambda | 17 |
| evaluatePriorPsi | 18 |
| EvaluateProposalLambda | 19 |
| EvaluateProposalPsy | 22 |
| generatePriorLambda | 25 |
| generatePriorPsi | 27 |
| generatePriorThetaY | 27 |
| getIndThetaY | 28 |
| getmode | 30 |
| getRemovedIndThetaY | 31 |
| getThetaYWithEmpty | 32 |
| getZmat | 34 |
| Hparam-class | 35 |
| likelihood | 35 |
| listToStrVec | 37 |
| MstepRJMCMCupdate | 38 |
| pgmmRJMCMC | 40 |
| stayMCMCupdate | 42 |
| sumerizeZ | 44 |
| summerizePgmmRJMCMC | 44 |
| ThetaYList | 45 |
| toEthetaYlist | 46 |
| toNEthetaYlist | 48 |
| updatePostThetaY | 49 |
| updatePostZ | 52 |
| VstepRJMCMCupdate | 53 |

Index

57

CalculateProposalLambda

CalculateProposalLambda

Description

CalculateProposalLambda

Usage

```
CalculateProposalLambda(hparam, thetaYList, CxyList, constraint, m, p,
  qVec)
```

Arguments

| | |
|------------|--|
| hparam | hparam |
| thetaYList | thetaYList |
| CxyList | CxyList |
| constraint | constraint |
| m | the number of clusters |
| p | the number of features |
| qVec | the vector of the number of factors in each clusters |

Examples

```
set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
m <- 1
muBar <- c(0, 0)

hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
```

```

tao = 0.366618687752634,
psy = list(structure(
  c(
    4.18375613018654,
    0, 0, 5.46215996830771
  ),
  .Dim = c(2L, 2L)
)),
M = list(structure(
  c(
    3.27412045866392,
    -2.40544145363349
  ),
  .Dim = 1:2
)),
lambda = list(structure(
  c(
    2.51015961514781,
    -0.0741189919182549
  ),
  .Dim = 2:1
)),
Y = list(structure(
  c(
    -0.244239011725104,
    -0.26876172736886,
    0.193431511203083,
    0.41624466812811,
    -0.54581548068437,
    -0.0479517628308146,
    -0.633383997203325,
    0.856855296613208,
    0.792850576988512,
    0.268208848994559
  ),
  .Dim = c(1L, 10L)
))
)
CxyList <-
list(
  A = list(structure(
    c(0.567755037123148, 0, 0, 1.1870201935945),
    .Dim = c(2L, 2L)
  )),
  nVec = structure(10, .Dim = c(1L, 1L)),
  Cxxk = list(structure(
    c(
      739.129405647622,
      671.040583460732,
      671.040583460732,
      618.754338945564
    ),
    .Dim = c(2L, 2L)
  )
)

```

```

)),
Cxyk = list(structure(
  c(-18.5170828875512, -16.5748393456787),
  .Dim = 2:1
)),
Cyyk = list(structure(2.4786991560888, .Dim = c(
  1L,
  1L
))),
Cytyk = list(structure(
  c(
    10, 0.787438922114998, 0.787438922114998,
    2.4786991560888
  ),
  .Dim = c(2L, 2L)
)),
Cxtyk = list(structure(
  c(
    -57.5402230447872,
    -54.6677145995824,
    -18.5170828875512,
    -16.5748393456787
  ),
  .Dim = c(
    2L,
    2L
  )
)),
CxL1k = list(structure(
  c(-59.5168204264758, -54.6093504204781),
  .Dim = 2:1
)),
Cxmyk = list(structure(
  c(
    -21.0952527723962,
    -14.6807011202188
  ),
  .Dim = 2:1
)),
sumCxmyk = structure(c(
  -21.0952527723962,
  -14.6807011202188
), .Dim = 2:1),
sumCyyk = structure(3.6657193496833, .Dim = c(
  1L,
  1L
))
)

```

CalculateProposalLambda(hparam, thetaYList, CxyList, constraint, m, p, qVec)

CalculateProposalPsy *CalculateProposalPsy*

Description

CalculateProposalPsy

Usage

```
CalculateProposalPsy(hparam, thetaYList, CxyList, constraint, m, p, qVec)
```

Arguments

| | |
|------------|--|
| hparam | hparam |
| thetaYList | thetaYList |
| CxyList | CxyList |
| constraint | constraint |
| m | the number of clusters |
| p | the number of features |
| qVec | the vector of the number of factors in each clusters |

Examples

```
set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
```

```

delta = 2,
ggamma = 2,
bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
new(
  "ThetaYList",
  tao = 0.366618687752634,
  psy = list(structure(
    c(
      4.18375613018654,
      0, 0, 5.46215996830771
    ),
    .Dim = c(2L, 2L)
  )),
  M = list(structure(
    c(
      3.27412045866392,
      -2.40544145363349
    ),
    .Dim = 1:2
  )),
  lambda = list(structure(
    c(
      2.51015961514781,
      -0.0741189919182549
    ),
    .Dim = 2:1
  )),
  Y = list(structure(
    c(
      -0.244239011725104,
      -0.26876172736886,
      0.193431511203083,
      0.41624466812811,
      -0.54581548068437,
      -0.0479517628308146,
      -0.633383997203325,
      0.856855296613208,
      0.792850576988512,
      0.268208848994559
    ),
    .Dim = c(1L, 10L)
  ))
)
constraint <- c(0, 0, 0)
CxyList <-
list(
  A = list(structure(
    c(0.567755037123148, 0, 0, 1.1870201935945),
    .Dim = c(2L, 2L)
  )),

```

```

nVec = structure(10, .Dim = c(1L, 1L)),
Cxxk = list(structure(
  c(
    739.129405647622,
    671.040583460732,
    671.040583460732,
    618.754338945564
  ),
  .Dim = c(2L, 2L)
)),
Cxyk = list(structure(
  c(-18.5170828875512, -16.5748393456787),
  .Dim = 2:1
)),
Cyyk = list(structure(2.4786991560888, .Dim = c(
  1L,
  1L
))),
Cytyk = list(structure(
  c(
    10, 0.787438922114998, 0.787438922114998,
    2.4786991560888
  ),
  .Dim = c(2L, 2L)
)),
Cxtyk = list(structure(
  c(
    -57.5402230447872,
    -54.6677145995824,
    -18.5170828875512,
    -16.5748393456787
  ),
  .Dim = c(
    2L,
    2L
  )
)),
CxL1k = list(structure(
  c(-59.5168204264758, -54.6093504204781),
  .Dim = 2:1
)),
Cxmyk = list(structure(
  c(
    -21.0952527723962,
    -14.6807011202188
  ),
  .Dim = 2:1
)),
sumCxmyk = structure(c(
  -21.0952527723962,
  -14.6807011202188
), .Dim = 2:1),
sumCyyk = structure(3.6657193496833, .Dim = c(

```



```
        1L,  
        1L  
      ))  
    )  
  #'
```

```
CalculateProposalPsy(hparam, thetaYList, CxyList, constraint, m, p, qVec)
```

| | |
|----------------|------------------------------------|
| calculateRatio | <i>Log scale ratio calculation</i> |
|----------------|------------------------------------|

Description

Log scale ratio calculation

Usage

```
calculateRatio(deno, nume)
```

Arguments

| | |
|------|--------------|
| deno | denominator. |
| nume | numerator. |

Value

result of ratio

Examples

```
deno <- log(1)  
nume <- log(2)  
#'  
  
calculateRatio(deno, nume)
```

| | |
|------------------|-------------------------|
| calculateVarList | <i>calculateVarList</i> |
|------------------|-------------------------|

Description

calculateVarList

Usage

```
calculateVarList(psyList, lambdaList)
```

Arguments

| | |
|------------|------------|
| psyList | psyList |
| lambdaList | lambdaList |

Examples

```
set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
```

```

tao = 0.366618687752634,
psy = list(structure(
  c(
    4.18375613018654,
    0, 0, 5.46215996830771
  ),
  .Dim = c(2L, 2L)
)),
M = list(structure(
  c(
    3.27412045866392,
    -2.40544145363349
  ),
  .Dim = 1:2
)),
lambda = list(structure(
  c(
    2.51015961514781,
    -0.0741189919182549
  ),
  .Dim = 2:1
)),
Y = list(structure(
  c(
    -0.244239011725104,
    -0.26876172736886,
    0.193431511203083,
    0.41624466812811,
    -0.54581548068437,
    -0.0479517628308146,
    -0.633383997203325,
    0.856855296613208,
    0.792850576988512,
    0.268208848994559
  ),
  .Dim = c(1L, 10L)
))
)
#'

calculateVarList(thetaYList@psy, thetaYList@lambda)

```

changeConstraintFormat

changeConstraintFormat

Description

changeConstraintFormat

Usage

```
changeConstraintFormat(strNum)
```

Arguments

```
strNum      strNum
```

Examples

```
#'
changeConstraintFormat(c(0, 0, 0))
```

```
clearCurrentThetaYlist
      clearCurrentThetaYlist
```

Description

```
clearCurrentThetaYlist
```

Usage

```
clearCurrentThetaYlist(thetaYList, clusInd, mMax)
```

Arguments

```
thetaYList  thetaYList
clusInd     clusInd
mMax       mMax
```

Examples

```
set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
```

```

    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    )),
    lambda = list(structure(
      c(
        2.51015961514781,
        -0.0741189919182549
      ),
      .Dim = 2:1
    )),
    Y = list(structure(
      c(
        -0.244239011725104,
        -0.26876172736886,
        0.193431511203083,
        0.41624466812811,
        -0.54581548068437,
        -0.0479517628308146,
        -0.633383997203325,
        0.856855296613208,
        0.792850576988512,
        0.268208848994559
      ),

```

```

        .Dim = c(1L, 10L)
      ))
    )
    clusInd <- rep(1, m)
    mMax <- 1
    #'

    clearCurrentThetaYlist(thetaYList, clusInd, mMax)

```

combineClusterPara *combineClusterPara*

Description

combineClusterPara

Usage

```
combineClusterPara(oldList, newList, ind)
```

Arguments

| | |
|---------|---------|
| oldList | oldList |
| newList | newList |
| ind | ind |

Examples

```

set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)

```

```

hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
newList <- oldList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    )),
    lambda = list(structure(
      c(
        2.51015961514781,
        -0.0741189919182549
      ),
      .Dim = 2:1
    )),
    Y = list(structure(
      c(
        -0.244239011725104,
        -0.26876172736886,
        0.193431511203083,
        0.41624466812811,
        -0.54581548068437,
        -0.0479517628308146,
        -0.633383997203325,
        0.856855296613208,
        0.792850576988512,
        0.268208848994559
      ),
      .Dim = c(1L, 10L)
    ))
  )
#'
combineClusterPara(oldList, newList, 1)

```

| | |
|---------------|-----------------------|
| evaluatePrior | <i>evaluate Prior</i> |
|---------------|-----------------------|

Description

evaluate prior value for parameter Theta and Y.

Usage

```
evaluatePrior(m, p, muBar, hparam, thetaYList, ZOneDim, qVec, constraint,
  clusInd)
```

Arguments

| | |
|------------|--------------------------|
| m | m |
| p | p |
| muBar | mu_bar |
| hparam | hyper parameter class |
| thetaYList | theta Y list |
| ZOneDim | one dim of z |
| qVec | q vector |
| constraint | type of constraint |
| clusInd | cluster indicator vector |

Examples

```
m <- 20
n <- 500
p <- 10
muBar <- c(
  -33.1342706763595, -35.2699639183419, 48.276928009445, 16.2370659601218,
  19.0023163870536, -23.4900965314972, 37.1081269873873, 4.74944562930846,
  4.6918997353449, -4.55088073255655
)
hparam <- new("Hparam",
  alpha1 = 0.567755037123148, alpha2 = 1.1870201935945,
  delta = 2, ggamma = 2, bbeta = 3.39466184520673
)
qVec <- c(4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
constraint <- c(0, 0, 0)
thetaYList <- generatePriorThetaY(m, n, p, muBar, hparam, qVec, ZOneDim, constraint)
clusInd <- rep(1, m)
#'
```



```

evaluatePrior(
  m,
  p,
  muBar,
  hparam,
  thetaYList,
  ZOneDim,
  qVec,
  constraint,
  clusInd
)

```

evaluatePriorLambda *evaluatePriorLambda*

Description

evaluate prior value for parameter Lambda

Usage

```
evaluatePriorLambda(p, m, alpha2, qVec, psy, lambda, constraint, clusInd)
```

Arguments

| | |
|------------|---|
| p | the number of features |
| m | the number of clusters |
| alpha2 | hyper parameter |
| qVec | the vector of the number of factors in each clusters |
| psy | parameter |
| lambda | parameter |
| constraint | the pgmm constraint, a vector of length three with binary entry. For example, c(1,1,1) means the fully constraint model |
| clusInd | cluster indicator vector |

Examples

```

p <- 10
m <- 20
alpha2 <- 1.18
qVec <- rep(4, m)
delta <- 2
bbeta <- 2
constraint <- c(0, 0, 0)
psy <- generatePriorPsi(
  p,

```

```

    m,
    delta,
    bbeta,
    constraint
  )
  lambda <- generatePriorLambda(
    p,
    m,
    alpha2,
    qVec,
    psy,
    constraint
  )
  clusInd <- rep(1, m)
  #'

  evaluatePriorLambda(
    p,
    m,
    alpha2,
    qVec,
    psy,
    lambda,
    constraint,
    clusInd
  )

```

evaluatePriorPsi *evaluatePriorPsi*

Description

evaluate prior value for parameter Psi

Usage

```
evaluatePriorPsi(psy, p, m, delta, bbeta, constraint, clusInd)
```

Arguments

| | |
|------------|--------------------------|
| psy | parameter |
| p | the number of features |
| m | the number of clusters |
| delta | parameter |
| bbeta | parameter |
| constraint | parameter |
| clusInd | cluster indicator vector |

Examples

```
p <- 10
m <- 20
delta <- 2
bbeta <- 2
constraint <- c(0, 0, 0)
psy <- generatePriorPsi(
  p,
  m,
  delta,
  bbeta,
  constraint
)
clusInd <- rep(1, m)
#'

evaluatePriorPsi(
  psy,
  p,
  m,
  delta,
  bbeta,
  constraint,
  clusInd
)
```

EvaluateProposalLambda

EvaluateProposalLambda

Description

EvaluateProposalLambda

Usage

```
EvaluateProposalLambda(hparam, thetaYList, CxyList, constraint, newlambda,
  m, qVec, p)
```

Arguments

| | |
|------------|------------|
| hparam | hparam |
| thetaYList | thetaYList |
| CxyList | CxyList |
| constraint | constraint |
| newlambda | newlambda |

m the number of clusters
 qVec the vector of the number of factors in each clusters
 p the number of features

Examples

```

set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    ))
  )

```

```

)),
lambda = list(structure(
  c(
    2.51015961514781,
    -0.0741189919182549
  ),
  .Dim = 2:1
)),
Y = list(structure(
  c(
    -0.244239011725104,
    -0.26876172736886,
    0.193431511203083,
    0.41624466812811,
    -0.54581548068437,
    -0.0479517628308146,
    -0.633383997203325,
    0.856855296613208,
    0.792850576988512,
    0.268208848994559
  ),
  .Dim = c(1L, 10L)
))
)
CxyList <-
list(
  A = list(structure(
    c(0.567755037123148, 0, 0, 1.1870201935945),
    .Dim = c(2L, 2L)
  )),
  nVec = structure(10, .Dim = c(1L, 1L)),
  Cxxk = list(structure(
    c(
      739.129405647622,
      671.040583460732,
      671.040583460732,
      618.754338945564
    ),
    .Dim = c(2L, 2L)
  )),
  Cxyk = list(structure(
    c(-18.5170828875512, -16.5748393456787),
    .Dim = 2:1
  )),
  Cyyk = list(structure(2.4786991560888, .Dim = c(
    1L,
    1L
  ))),
  Cytyk = list(structure(
    c(
      10, 0.787438922114998, 0.787438922114998,
      2.4786991560888
    ),

```

```

    .Dim = c(2L, 2L)
 )),
  Cxtytk = list(structure(
    c(
      -57.5402230447872,
      -54.6677145995824,
      -18.5170828875512,
      -16.5748393456787
    ),
    .Dim = c(
      2L,
      2L
    )
  )),
  CxL1k = list(structure(
    c(-59.5168204264758, -54.6093504204781),
    .Dim = 2:1
  )),
  Cxmyk = list(structure(
    c(
      -21.0952527723962,
      -14.6807011202188
    ),
    .Dim = 2:1
  )),
  sumCxmyk = structure(c(
    -21.0952527723962,
    -14.6807011202188
  ), .Dim = 2:1),
  sumCyyk = structure(3.6657193496833, .Dim = c(
    1L,
    1L
  ))
)
#'

```

```
EvaluateProposalLambda(hparam, thetaYList, CxyList, constraint, thetaYList@lambda, m, qVec, p)
```

EvaluateProposalPsy *EvaluateProposalPsy*

Description

EvaluateProposalPsy

Usage

```
EvaluateProposalPsy(hparam, thetaYList, CxyList, constraint, newpsy, m, p,
  qVec, delta)
```

Arguments

| | |
|------------|--|
| hparam | hparam |
| thetaYList | thetaYList |
| CxyList | CxyList |
| constraint | constraint |
| newpsy | newpsy |
| m | the number of clusters |
| p | the number of features |
| qVec | the vector of the number of factors in each clusters |
| delta | hyperparameters |

Examples

```

set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,

```

```

      0, 0, 5.46215996830771
    ),
    .Dim = c(2L, 2L)
  )),
  M = list(structure(
    c(
      3.27412045866392,
      -2.40544145363349
    ),
    .Dim = 1:2
  )),
  lambda = list(structure(
    c(
      2.51015961514781,
      -0.0741189919182549
    ),
    .Dim = 2:1
  )),
  Y = list(structure(
    c(
      -0.244239011725104,
      -0.26876172736886,
      0.193431511203083,
      0.41624466812811,
      -0.54581548068437,
      -0.0479517628308146,
      -0.633383997203325,
      0.856855296613208,
      0.792850576988512,
      0.268208848994559
    ),
    .Dim = c(1L, 10L)
  ))
)
constraint <- c(0, 0, 0)
CxyList <-
list(
  A = list(structure(
    c(0.567755037123148, 0, 0, 1.1870201935945),
    .Dim = c(2L, 2L)
  )),
  nVec = structure(10, .Dim = c(1L, 1L)),
  Cxxk = list(structure(
    c(
      739.129405647622,
      671.040583460732,
      671.040583460732,
      618.754338945564
    ),
    .Dim = c(2L, 2L)
  )),
  Cxyk = list(structure(
    c(-18.5170828875512, -16.5748393456787),

```



```

        .Dim = 2:1
    )),
  Cyyk = list(structure(2.4786991560888, .Dim = c(
    1L,
    1L
  )),
  Cytyk = list(structure(
    c(
      10, 0.787438922114998, 0.787438922114998,
      2.4786991560888
    ),
    .Dim = c(2L, 2L)
  )),
  Cxtyk = list(structure(
    c(
      -57.5402230447872,
      -54.6677145995824,
      -18.5170828875512,
      -16.5748393456787
    ),
    .Dim = c(
      2L,
      2L
    )
  )),
  CxL1k = list(structure(
    c(-59.5168204264758, -54.6093504204781),
    .Dim = 2:1
  )),
  Cxmyk = list(structure(
    c(
      -21.0952527723962,
      -14.6807011202188
    ),
    .Dim = 2:1
  )),
  sumCxmyk = structure(c(
    -21.0952527723962,
    -14.6807011202188
  ), .Dim = 2:1),
  sumCyyk = structure(3.6657193496833, .Dim = c(
    1L,
    1L
  ))
)

```

EvaluateProposalPsy(hparam, thetaYList, CxyList, constraint, thetaYList@psy, m, p, qVec, delta)

Description

evaluate prior value for parameter Lambda

Usage

```
generatePriorLambda(p, m, alpha2, qVec, psy, constraint)
```

Arguments

| | |
|------------|------------------------|
| p | the number of features |
| m | the number of clusters |
| alpha2 | hyper parameter |
| qVec | parameter |
| psy | parameter |
| constraint | parameter |

Examples

```
p <- 10
m <- 20
alpha2 <- 1.18
qVec <- rep(4, m)
delta <- 2
bbeta <- 2
constraint <- c(0, 0, 0)
psy <- generatePriorPsi(
  p,
  m,
  delta,
  bbeta,
  constraint
)
#'

generatePriorLambda(
  p,
  m,
  alpha2,
  qVec,
  psy,
  constraint
)
```

generatePriorPsi *generatePriorPsi*

Description

generate prior value for parameter Psi

Usage

```
generatePriorPsi(p, m, delta, bbeta, constraint)
```

Arguments

| | |
|------------|---|
| p | the number of features |
| m | the number of clusters |
| delta | hyperparameters |
| bbeta | hyperparameters |
| constraint | the pgmm constraint, a vector of length three with binary entry. For example, c(1,1,1) means the fully constraint model |

Examples

```
p <- 10
m <- 20
delta <- 2
bbeta <- 2
constraint <- c(0, 0, 0)
```

```
generatePriorPsi(
  p,
  m,
  delta,
  bbeta,
  constraint
)
```

generatePriorThetaY *PriorThetaY list*

Description

generate prior value for parameter Theta and Y.

Usage

```
generatePriorThetaY(m, n, p, muBar, hparam, qVec, ZOneDim, constraint)
```

Arguments

| | |
|------------|--|
| m | the number of cluster |
| n | sample size |
| p | number of covariates |
| muBar | parameter |
| hparam | hyperparameters |
| qVec | the vector of the number of factors in each clusters |
| ZOneDim | ZOneDim |
| constraint | constraint |

Examples

```
m <- 20
n <- 500
p <- 10
muBar <- c(
  -33.1342706763595, -35.2699639183419, 48.276928009445, 16.2370659601218,
  19.0023163870536, -23.4900965314972, 37.1081269873873, 4.74944562930846,
  4.6918997353449, -4.55088073255655
)
hparam <- new("Hparam",
  alpha1 = 0.567755037123148, alpha2 = 1.1870201935945,
  delta = 2, ggamma = 2, bbeta = 3.39466184520673
)
qVec <- c(4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
constraint <- c(0, 0, 0)
#'

generatePriorThetaY(m, n, p, muBar, hparam, qVec, ZOneDim, constraint)
```

```
getIndThetaY
```

```
getIndThetaY
```

Description

```
getIndThetaY
```

Usage

```
getIndThetaY(thetaYList, Ind)
```

Arguments

| | |
|------------|------------|
| thetaYList | thetaYList |
| Ind | Ind |

Examples

```

set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 2
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    )
  )

```

```

)),
lambda = list(structure(
  c(
    2.51015961514781,
    -0.0741189919182549
  ),
  .Dim = 2:1
)),
Y = list(structure(
  c(
    -0.244239011725104,
    -0.26876172736886,
    0.193431511203083,
    0.41624466812811,
    -0.54581548068437,
    -0.0479517628308146,
    -0.633383997203325,
    0.856855296613208,
    0.792850576988512,
    0.268208848994559
  ),
  .Dim = c(1L, 10L)
))
)
#'

getIndThetaY(thetaYList, 1)

```

getmode

getmode

Description

getmode

Usage

getmode(v)

Arguments

v v

Examples

```

# '

getmode(c(1, 1, 2, 3))

```

```
getRemovedIndThetaY  getRemovedIndThetaY
```

Description

```
getRemovedIndThetaY
```

Usage

```
getRemovedIndThetaY(thetaYList, Ind)
```

Arguments

| | |
|------------|------------|
| thetaYList | thetaYList |
| Ind | Ind |

Examples

```
set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 2
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new("ThetaYList", tao = c(0.90162050961987, 0.0983794903801295),
```

```

psy = list(structure(c(3.68472841602225, 0, 0, 8.34691978354054),
  .Dim = c(2L, 2L)), structure(c(0.785011896130842, 0, 0, 1.19022383323437),
  .Dim = c(2L, 2L))), M = list(structure(c(
  2.96424305287004,
  1.08454861414306
), .Dim = 1:2), structure(c(
  -0.232625450433964,
  0.984505960868685
), .Dim = 1:2)), lambda = list(structure(c(
  -0.964026624054337,
  0.89378616732449
), .Dim = 2:1), structure(c(
  0.533334148228635,
  -1.80033696090263
), .Dim = 2:1)), Y = list(structure(c(
  -0.15346475266988,
  1.6584112693271, 0.409294936277862, -1.46628591247549, -0.532753243163142,
  -0.332143130316749, 0.307558110800446, -0.525374243612587, 0.527667526535661,
  0.748193650431916
), .Dim = c(1L, 10L)), structure(c(
  0.571325118638535,
  0.542462985882966, 0.559971315637159, -1.73905343105432, -0.583549598471542,
  1.71264245945391, -0.327119395945831, 1.02464651767821, -1.11462280255215,
  0.81095592501554
), .Dim = c(1L, 10L))))
Ind <- 1
#'

getRemovedIndThetaY(thetaYList, Ind)

```

getThetaYWithEmpty *getThetaYWithEmpty*

Description

getThetaYWithEmpty

Usage

```
getThetaYWithEmpty(NEthetaYList, clusInd)
```

Arguments

| | |
|--------------|--------------|
| NEthetaYList | NEthetaYList |
| clusInd | clusInd |

Examples

```

set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    )),
    lambda = list(structure(
      c(
        2.51015961514781,
        -0.0741189919182549
      )
    )
  )

```

```

    ),
    .Dim = 2:1
  )),
  Y = list(structure(
    c(
      -0.244239011725104,
      -0.26876172736886,
      0.193431511203083,
      0.41624466812811,
      -0.54581548068437,
      -0.0479517628308146,
      -0.633383997203325,
      0.856855296613208,
      0.792850576988512,
      0.268208848994559
    ),
    .Dim = c(1L, 10L)
  ))
)
clusInd <- rep(1, m)

getThetaYWithEmpty(thetaYList, clusInd)

```

getZmat

Tool for vector to matrix

Description

Tool for vector to matrix

Usage

```
getZmat(ZOneDim, m, n)
```

Arguments

| | |
|---------|------------------------|
| ZOneDim | a vector. |
| m | the number of cluster. |
| n | sample size. |

Value

adjacency matrix

Examples

```

m <- 20
n <- 500
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
#'

getZmat(ZOneDim, m, n)

```

Hparam-class

An S4 class to represent a Hyper parameter.

Description

An S4 class to represent a Hyper parameter.

Slots

alpha1 A numeric value
alpha2 A numeric value
delta A numeric value
ggamma A numeric value
bbeta A numeric value

Examples

```
new("Hparam", alpha1 = 1, alpha2 = 2, bbeta = 3, delta = 4, ggamma = 5)
```

likelihood

likelihood

Description

likelihood

Usage

```
likelihood(thetaYList, ZOneDim, qqVec, muBar, X)
```

Arguments

| | |
|------------|------------|
| thetaYList | thetaYList |
| ZOneDim | ZOneDim |
| qqVec | qqVec |
| muBar | muBar |
| X | X |

Examples

```

set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    )),
    lambda = list(structure(
      c(
        2.51015961514781,
        -0.0741189919182549
      )
    )
  )

```

```

    ),
    .Dim = 2:1
  )),
  Y = list(structure(
    c(
      -0.244239011725104,
      -0.26876172736886,
      0.193431511203083,
      0.41624466812811,
      -0.54581548068437,
      -0.0479517628308146,
      -0.633383997203325,
      0.85685296613208,
      0.792850576988512,
      0.268208848994559
    ),
    .Dim = c(1L, 10L)
  ))
)
#'
likelihood(thetaYList, ZOneDim, qVec, muBar, X)

```

listToStrVec

Convert list of string to vector of string

Description

Convert list of string to vector of string

Usage

```
listToStrVec(stringList)
```

Arguments

stringList list of string

Value

vector of string

Examples

```

stringList <- list("abc")
#'

listToStrVec(stringList)

```

MstepRJMCMCupdate *MstepRJMCMCupdate*

Description

MstepRJMCMCupdate

Usage

```
MstepRJMCMCupdate(X, muBar, p, thetaYList, ZOneDim, hparam, hparamInit,
  qVec, qnew, dVec, sVec, constraint, clusInd, mVec, Mind)
```

Arguments

| | |
|------------|------------------------|
| X | X in MstepRJMCMCupdate |
| muBar | muBar |
| p | p |
| thetaYList | thetaYList |
| ZOneDim | ZOneDim |
| hparam | hparam |
| hparamInit | hparamInit |
| qVec | qVec |
| qnew | qnew |
| dVec | dVec |
| sVec | sVec |
| constraint | constraint |
| clusInd | clusInd |
| mVec | mVec |
| Mind | Mind |

Examples

```
set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 2
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
```

```

    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
new(
  "ThetaYList",
  tao = 0.366618687752634,
  psy = list(structure(
    c(
      4.18375613018654,
      0, 0, 5.46215996830771
    ),
    .Dim = c(2L, 2L)
  )),
  M = list(structure(
    c(
      3.27412045866392,
      -2.40544145363349
    ),
    .Dim = 1:2
  )),
  lambda = list(structure(
    c(
      2.51015961514781,
      -0.0741189919182549
    ),
    .Dim = 2:1
  )),
  Y = list(structure(
    c(
      -0.244239011725104,
      -0.26876172736886,
      0.193431511203083,
      0.41624466812811,
      -0.54581548068437,
      -0.0479517628308146,
      -0.633383997203325,
      0.856855296613208,
      0.792850576988512,

```

```

        0.268208848994559
      ),
      .Dim = c(1L, 10L)
    ))
  )
  qnew <- 1
  dVec <- c(1, 1, 1)
  sVec <- c(1, 1, 1)
  constraint <- c(0, 0, 0)
  clusInd <- rep(1, m)
  Mind <- "BD"
  mVec <- c(1, m)

  MstepRJMCMCupdate(
    X,
    muBar,
    p,
    thetaYList,
    ZOneDim,
    hparam,
    hparamInit,
    qVec,
    qnew,
    dVec,
    sVec,
    constraint,
    clusInd,
    mVec,
    Mind
  )

```

pgmmRJMCMC

bpgmm Model-Based Clustering Using Bayesian PGMM Carries out model-based clustering using parsimonious Gaussian mixture models. MCMC are used for parameter estimation. The RJMCMC is used for model selection.

Description

bpgmm Model-Based Clustering Using Bayesian PGMM Carries out model-based clustering using parsimonious Gaussian mixture models. MCMC are used for parameter estimation. The RJMCMC is used for model selection.

Usage

```

pgmmRJMCMC(X, mInit, mVec, qnew, delta = 2, ggamma = 2, burn = 20,
  niter = 1000, constraint = C(0, 0, 0), dVec = c(1, 1, 1),
  sVec = c(1, 1, 1), Mstep = 0, Vstep = 0, SCind = 0)

```


Arguments

| | |
|------------|---|
| X | the observation matrix with size $p * m$ |
| mInit | the number of initial clusters |
| mVec | the range of the number of clusters |
| qnew | the number of factor for a new cluster |
| delta | scaler hyperparameters |
| ggamma | scaler hyperparameters |
| burn | the number of burn in iterations |
| niter | the number of iterations |
| constraint | the pgmm initial constraint, a vector of length three with binary entry. For example, c(1,1,1) means the fully constraint model |
| dVec | a vector of hyperparameters with length three, shape parameters for alpha1, alpha2 and bbeta respectively |
| sVec | sVec a vector of hyperparameters with length three, rate parameters for alpha1, alpha2 and bbeta respectively |
| Mstep | the indicator of whether do model selection on the number of clusters |
| Vstep | the indicator of whether do model selection on variance structures |
| SCind | the indicator of whether use split/combine step in Mstep |

Examples

```

library("fabMix")
library("mclust")
library("pgmm")
library("mvtnorm")
library("mcmcse")
library("MASS")
library("gtools")
n <- 500
p <- 10
q <- 4
K <- 10
nsim <- 10
burn <- 20
qnew <- 4
Mstep <- 1
Vstep <- 1
constraint <- c(0, 0, 0)
mInit <- 20
mVec <- c(1, 20)
X <- t(simData(
  sameLambda = TRUE,
  sameSigma = TRUE,
  K.true = K, n = n, q = q, p = p, sINV_values = 1 / ((1:p))
)$data)

```

```
pgmmRJCMC(X,
  mInit, mVec, qnew,
  niter = nsim, burn = burn,
  constraint = constraint, Mstep = Mstep, Vstep = Vstep
)
```

stayMCMCupdate

stayMCMCupdate

Description

stayMCMCupdate

Usage

```
stayMCMCupdate(X, thetaYList, ZOneDim, hparam, qVec, qnew, dVec, sVec,
  constraint, clusInd)
```

Arguments

| | |
|------------|------------|
| X | X |
| thetaYList | thetaYList |
| ZOneDim | ZOneDim |
| hparam | hparam |
| qVec | qVec |
| qnew | qnew |
| dVec | dVec |
| sVec | sVec |
| constraint | constraint |
| clusInd | clusInd |

Examples

```
##
set.seed(110)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 2
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
```

```

X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
new("ThetaYList", tao = c(0.90162050961987, 0.0983794903801295),
psy = list(structure(c(3.68472841602225, 0, 0, 8.34691978354054),
.Dim = c(2L, 2L)), structure(c(0.785011896130842, 0, 0, 1.19022383323437),
.Dim = c(2L, 2L))), M = list(structure(c(
  2.96424305287004,
  1.08454861414306
), .Dim = 1:2), structure(c(
  -0.232625450433964,
  0.984505960868685
), .Dim = 1:2)), lambda = list(structure(c(
  -0.964026624054337,
  0.89378616732449
), .Dim = 2:1), structure(c(
  0.533334148228635,
  -1.80033696090263
), .Dim = 2:1)), Y = list(structure(c(
  -0.15346475266988,
  1.6584112693271, 0.409294936277862, -1.46628591247549, -0.532753243163142,
  -0.332143130316749, 0.307558110800446, -0.525374243612587, 0.527667526535661,
  0.748193650431916
), .Dim = c(1L, 10L)), structure(c(
  0.571325118638535,
  0.542462985882966, 0.559971315637159, -1.73905343105432, -0.583549598471542,
  1.71264245945391, -0.327119395945831, 1.02464651767821, -1.11462280255215,
  0.81095592501554
), .Dim = c(1L, 10L))))
qnew <- 1
dVec <- c(1, 1, 1)
sVec <- c(1, 1, 1)
constraint <- c(0, 0, 0)
clusInd <- rep(1, m)

```

```

stayMCMCupdate(
  X,
  thetaYList,
  ZOneDim,
  hparam,
  qVec,
  qnew,
  dVec,
  sVec,
  constraint,
  clusInd
)

```

summerizeZ

summerizeZ

Description

summerizeZ

Usage

```
summerizeZ(Zlist, index = 1:length(Zlist))
```

Arguments

| | |
|-------|-------|
| Zlist | Zlist |
| index | index |

Examples

```

Zlist <- list(c(1, 2, 3), c(3, 2, 1), c(2, 2, 2))
#'
summerizeZ(Zlist)

```

summerizePgmmRJCMC

summerizePgmmRJCMC

Description

summerizePgmmRJCMC

Usage

```
summerizePgmmRJCMC(pgmmResList, trueCluster = NULL)
```

Arguments

pgmmResList result list from pgmmRJMCMC
 trueCluster true cluster allocation

Examples

```
library("fabMix")
library("mclust")
library("pgmm")
library("mvtnorm")
library("mcmcse")
library("MASS")
library("gtools")
n <- 50
p <- 10
q <- 4
K <- 10
syntheticDataset <- simData(
  sameLambda = TRUE, sameSigma = TRUE, K.true = K, n = n, q = q, p = p,
  sINV_values = 1 / ((1:p))
)
nsim <- 5
burn <- 0
X <- t(syntheticDataset$data)
qnew <- 4
Mstep <- 1
Vstep <- 1
constraint <- c(0, 0, 0)
mInit <- 20
mVec <- c(1, 20)

res <- pgmmRJMCMC(X, mInit, mVec, qnew,
  niter = nsim, burn = burn, constraint = constraint,
  Mstep = Mstep, Vstep = Vstep
)

summerizePgmmRJMCMC(res, syntheticDataset$class)
```

 ThetaYList

ThetaYList-class

Description

Definiton of ThetaYList parameter sets

Slots

tao A numeric vector
 psy A list value
 M A list value
 lambda A list value
 Y A list value

| | |
|---------------|--------------|
| toEthetaYlist | <i>Title</i> |
|---------------|--------------|

Description

Title

Usage

```
toEthetaYlist(NEthetaYList, NEZOneDim, qnew, clusInd)
```

Arguments

| | |
|--------------|--------------|
| NEthetaYList | NEthetaYList |
| NEZOneDim | NEZOneDim |
| qnew | qnew |
| clusInd | clusInd |

Examples

```

set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data

```

```

)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    )),
    lambda = list(structure(
      c(
        2.51015961514781,
        -0.0741189919182549
      ),
      .Dim = 2:1
    )),
    Y = list(structure(
      c(
        -0.244239011725104,
        -0.26876172736886,
        0.193431511203083,
        0.41624466812811,
        -0.54581548068437,
        -0.0479517628308146,
        -0.633383997203325,
        0.856855296613208,
        0.792850576988512,
        0.268208848994559
      ),
      .Dim = c(1L, 10L)
    ))
  )
clusInd <- rep(1, m)
qnew <- 1

```

```
toEthetaYlist(thetaYList, ZOneDim, qnew, clusInd)
```

| | |
|----------------|-----------------------|
| toNEthetaYlist | <i>toNEthetaYlist</i> |
|----------------|-----------------------|

Description

toNEthetaYlist

Usage

```
toNEthetaYlist(thetaYList, ZOneDim, qVec, clusInd)
```

Arguments

| | |
|------------|------------|
| thetaYList | thetaYList |
| ZOneDim | ZOneDim |
| qVec | qVec |
| clusInd | clusInd |

Examples

```
set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
```



```

)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    )),
    lambda = list(structure(
      c(
        2.51015961514781,
        -0.0741189919182549
      ),
      .Dim = 2:1
    )),
    Y = list(structure(
      c(
        -0.244239011725104,
        -0.26876172736886,
        0.193431511203083,
        0.41624466812811,
        -0.54581548068437,
        -0.0479517628308146,
        -0.633383997203325,
        0.856855296613208,
        0.792850576988512,
        0.268208848994559
      ),
      .Dim = c(1L, 10L)
    ))
  )
clusInd <- rep(1, m)

toNEthetaYlist(thetaYList, ZOneDim, qVec, clusInd)

```

Description

Update posterior theta Y list

Usage

```
updatePostThetaY(m, n, p, hparam, thetaYList, ZOneDim, qVec, constraint, X,
  ggamma)
```

Arguments

| | |
|------------|-----------------------------|
| m | the number of clusters. |
| n | the number of observations. |
| p | the number of variables |
| hparam | hyper parameters |
| thetaYList | theta Y list |
| ZOneDim | ZOneDim |
| qVec | qVec |
| constraint | constraint |
| X | X |
| ggamma | ggamma |

Examples

```
set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
```

```

    ggamma = 2,
    bbeta = 3.39466184520673
  )
  ZOneDim <- sample(seq_len(m), n, replace = TRUE)
  thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    )),
    lambda = list(structure(
      c(
        2.51015961514781,
        -0.0741189919182549
      ),
      .Dim = 2:1
    )),
    Y = list(structure(
      c(
        -0.244239011725104,
        -0.26876172736886,
        0.193431511203083,
        0.41624466812811,
        -0.54581548068437,
        -0.0479517628308146,
        -0.633383997203325,
        0.856855296613208,
        0.792850576988512,
        0.268208848994559
      ),
      .Dim = c(1L, 10L)
    ))
  )
  constraint <- c(0, 0, 0)
  #'

```

```

updatePostThetaY(m, n, p, hparam, thetaYList, ZOneDim, qVec, constraint, X, ggamma)

```

| | |
|-------------|--------------------|
| updatePostZ | <i>updatePostZ</i> |
|-------------|--------------------|

Description

updatePostZ

Usage

updatePostZ(X, m, n, thetaYList)

Arguments

| | |
|------------|------------|
| X | X |
| m | m |
| n | n |
| thetaYList | thetaYList |

Examples

```

set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
  alpha2 = 1.1870201935945,
  delta = 2,
  ggamma = 2,
  bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)

```

```

thetaYList <-
  new(
    "ThetaYList",
    tao = 0.366618687752634,
    psy = list(structure(
      c(
        4.18375613018654,
        0, 0, 5.46215996830771
      ),
      .Dim = c(2L, 2L)
    )),
    M = list(structure(
      c(
        3.27412045866392,
        -2.40544145363349
      ),
      .Dim = 1:2
    )),
    lambda = list(structure(
      c(
        2.51015961514781,
        -0.0741189919182549
      ),
      .Dim = 2:1
    )),
    Y = list(structure(
      c(
        -0.244239011725104,
        -0.26876172736886,
        0.193431511203083,
        0.41624466812811,
        -0.54581548068437,
        -0.0479517628308146,
        -0.633383997203325,
        0.856855296613208,
        0.792850576988512,
        0.268208848994559
      ),
      .Dim = c(1L, 10L)
    ))
  )

updatePostZ(X, m, n, thetaYList)

```

VstepRJMCMCupdate

VstepRJMCMCupdate

Description

VstepRJMCMCupdate

Usage

```
VstepRJMCMCupdate(X, muBar, p, thetaYList, ZOneDim, hparam, hparamInit,
  qVec, qnew, ggamma, dVec, sVec, constraint, clusInd)
```

Arguments

| | |
|------------|------------|
| X | X |
| muBar | muBar |
| p | p |
| thetaYList | thetaYList |
| ZOneDim | ZOneDim |
| hparam | hparam |
| hparamInit | hparamInit |
| qVec | qVec |
| qnew | qnew |
| ggamma | ggamma |
| dVec | dVec |
| sVec | sVec |
| constraint | constraint |
| clusInd | clusInd |

Examples

```
set.seed(100)
n <- 10
p <- 2
q <- 1
K <- 2
m <- 1
muBar <- c(0, 0)
qVec <- c(1, 1)
constraint <- c(0, 0, 0)
X <- t(
  fabMix::simData(
    sameLambda = TRUE,
    sameSigma = TRUE,
    K.true = K,
    n = n,
    q = q,
    p = p,
    sINV_values = 1 / ((1:p))
  )$data
)
hparam <- new(
  "Hparam",
  alpha1 = 0.567755037123148,
```

```

alpha2 = 1.1870201935945,
delta = 2,
ggamma = 2,
bbeta = 3.39466184520673
)
ZOneDim <- sample(seq_len(m), n, replace = TRUE)
thetaYList <-
new(
  "ThetaYList",
  tao = 0.366618687752634,
  psy = list(structure(
    c(
      4.18375613018654,
      0, 0, 5.46215996830771
    ),
    .Dim = c(2L, 2L)
  )),
  M = list(structure(
    c(
      3.27412045866392,
      -2.40544145363349
    ),
    .Dim = 1:2
  )),
  lambda = list(structure(
    c(
      2.51015961514781,
      -0.0741189919182549
    ),
    .Dim = 2:1
  )),
  Y = list(structure(
    c(
      -0.244239011725104,
      -0.26876172736886,
      0.193431511203083,
      0.41624466812811,
      -0.54581548068437,
      -0.0479517628308146,
      -0.633383997203325,
      0.856855296613208,
      0.792850576988512,
      0.268208848994559
    ),
    .Dim = c(1L, 10L)
  ))
)
qnew <- 1
dVec <- c(1, 1, 1)
sVec <- c(1, 1, 1)
constraint <- c(0, 0, 0)
clusInd <- rep(1, m)

```

```
VstepRJMCMCupdate(  
  X,  
  muBar,  
  p,  
  thetaYList,  
  ZOneDim,  
  hparam,  
  hparamInit,  
  qVec,  
  qnew,  
  ggamma,  
  dVec,  
  sVec,  
  constraint,  
  clusInd  
)
```


Index

CalculateProposalLambda, [2](#)
CalculateProposalPsy, [6](#)
calculateRatio, [9](#)
calculateVarList, [10](#)
changeConstraintFormat, [11](#)
clearCurrentThetaYlist, [12](#)
combineClusterPara, [14](#)

evaluatePrior, [16](#)
evaluatePriorLambda, [17](#)
evaluatePriorPsi, [18](#)
EvaluateProposalLambda, [19](#)
EvaluateProposalPsy, [22](#)

generatePriorLambda, [25](#)
generatePriorPsi, [27](#)
generatePriorThetaY, [27](#)
getIndThetaY, [28](#)
getmode, [30](#)
getRemovedIndThetaY, [31](#)
getThetaYWithEmpty, [32](#)
getZmat, [34](#)

Hparam-class, [35](#)

likelihood, [35](#)
listToStrVec, [37](#)

MstepRJMCMCupdate, [38](#)

pgmmRJMCMC, [40](#)

stayMCMCupdate, [42](#)
sumerizeZ, [44](#)
summerizePgmmRJMCMC, [44](#)

ThetaYList, [45](#)
toEthetaYlist, [46](#)
toNthetaYlist, [48](#)

updatePostThetaY, [49](#)
updatePostZ, [52](#)

VstepRJMCMCupdate, [53](#)