Rmd_Clustering_part1

Alessandro Colombi

20/5/2022

Inizio a fare dei test sul clustering.

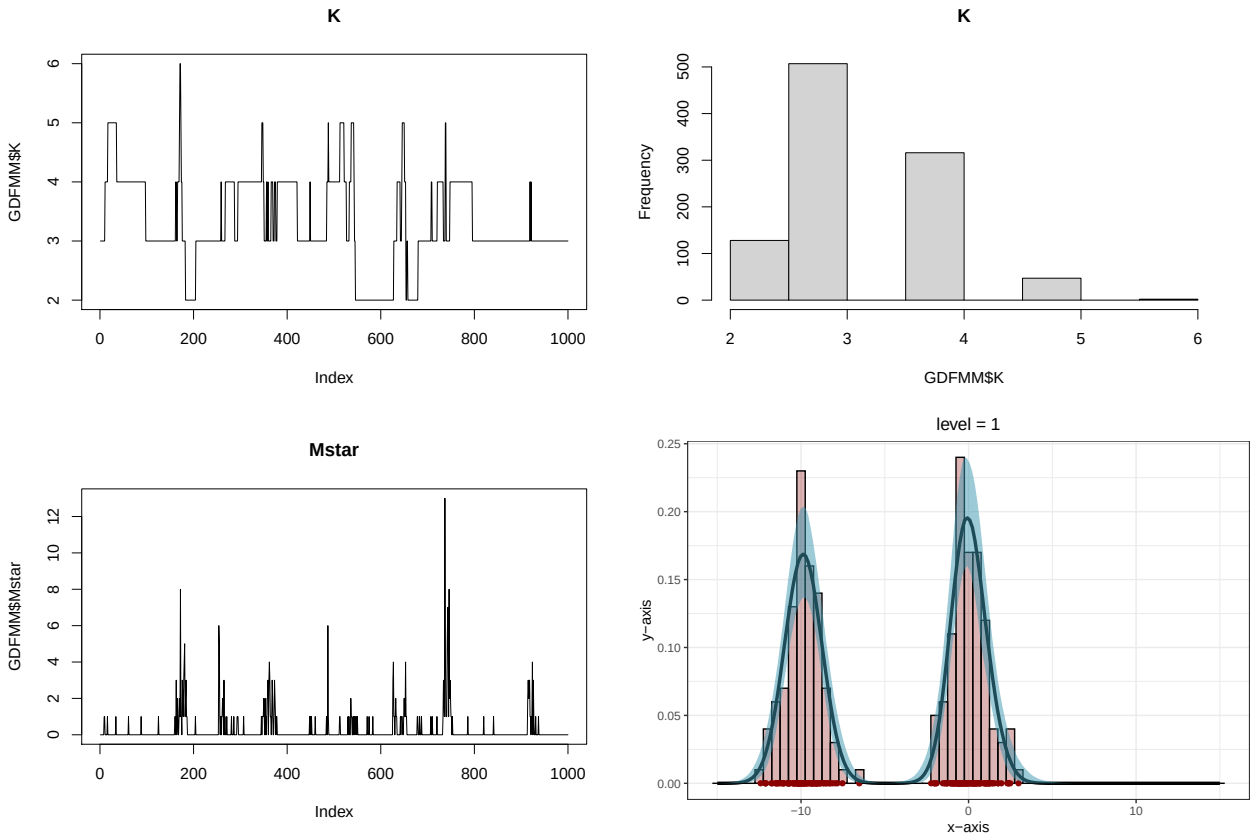
d=1

Primo caso, K = 2

Caso non simmetrico.

```
suppressWarnings(suppressPackageStartupMessages(library(GDFMM)))
suppressWarnings(suppressPackageStartupMessages(library(ACutils)))
suppressWarnings(suppressPackageStartupMessages(library(tidyverse)))
suppressWarnings(suppressPackageStartupMessages(library(RColorBrewer)))
suppressWarnings(suppressPackageStartupMessages(library(salso)))

# color palette ------------------------------------------------------------
mycol = hcl.colors(n=3,palette = "Zissou1")
# data generation ----------------------------------------------------------

d = 1               # number of groups
K = 2               # number of global clusters
mu = c(-10,0)    # vectors of means
sd = c(1,1)       # vector of sd
n_j = rep(200, d)  # set cardinality of the groups
seed = 1243
mycol_cluster = brewer.pal(n=K, name = "Dark2")
#> Warning in brewer.pal(n = K, name = "Dark2"): minimal value for n is 3, returning requested palette

genD = generate_data(d=d, K=K, mu = mu, sd = sd, n_j = n_j, seed = seed)
data = genD$data
real_partition = genD$real_partition

# Run  ---------------------------------------------------------------------


niter  <- 1000
burnin <- 3000
thin   <- 1

option<-list("nu" = 1, "Mstar0" = 2, "Lambda0" = 3, "mu0" = 0,"sigma0"= 1, "gamma0" = 1,
             "Adapt_MH_hyp1"= 0.7,"Adapt_MH_hyp2"= 0.234, "Adapt_MH_power_lim"=10, "Adapt_MH_var0"=1,
             "k0"= 1/10, "nu0"=10, "alpha_gamma"=1,
             "beta_gamma"=1, "alpha_lambda"=1, "beta_lambda"=1,
             "UpdateU" = T, "UpdateM" = T, "UpdateGamma" = T, "UpdateS" = T,
```

```
              "UpdateTau" = T, "UpdateLambda" = T
)

#GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, option = option)
GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, FixPartition = F, option = option)
```



Controllo la qualità del clustering

- `part_matrix`: This is a `(n_iter x n)` matrix, the j-th column contains all the labels assign to the j-th data point for all saved iterations.
- `sim_matrix`: This is a `(n x n)` matrix. Element (i,j) contains the probability that data points i and j are clustered together.
- `estimate_partition`: this is a vector of length `n` containing the estimated partition.

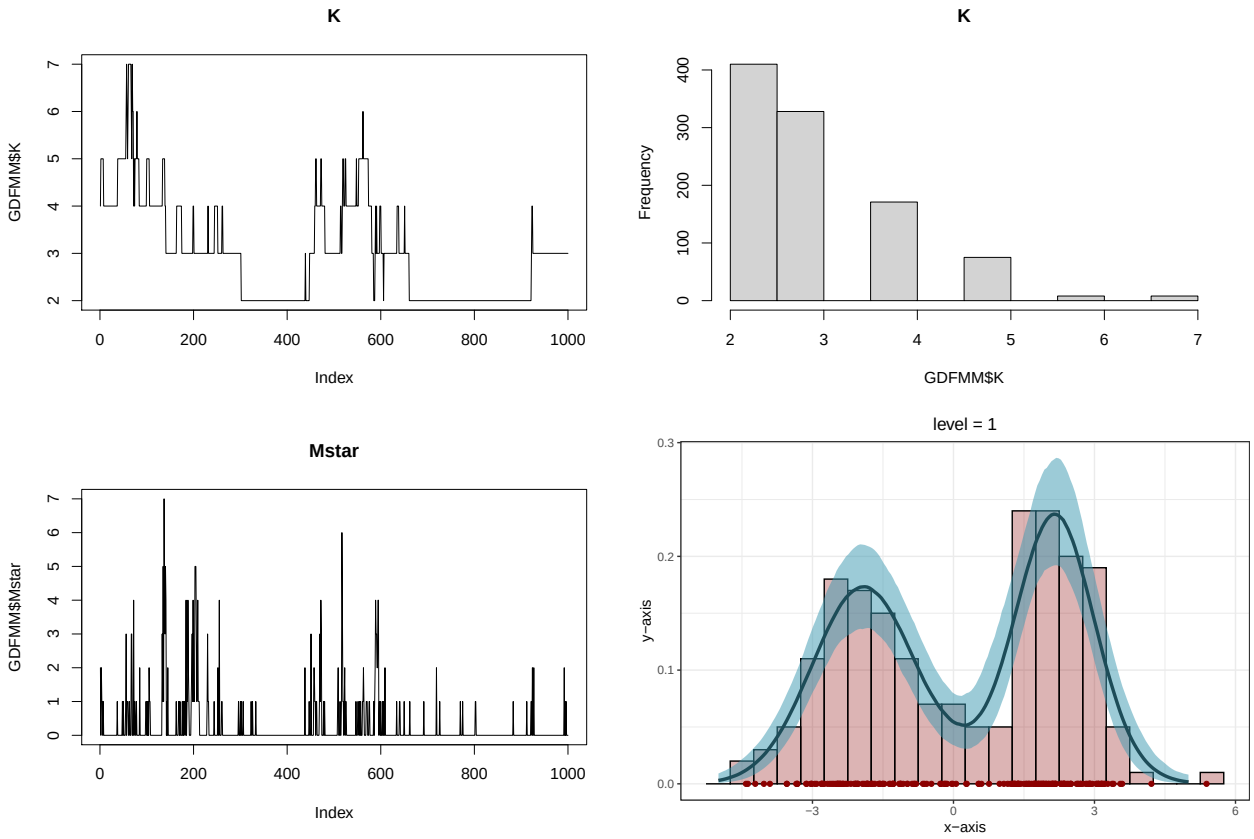
```
# COMPUTE BINDER LOSS FUNCTION TO SELECT BEST PARTITION --------------------

# Get labels for each iterations for each data point
part_matrix <- GDFMM$Partition

# Compute similarity matrix
sim_matrix <- psm(part_matrix)

# Get estimated partition according to binder (or VI) loss functions.
# VI_dahl <- dlso(matr, loss = 'VI', estimate=NULL)
binder_dahl <- dlso(part_matrix, loss = 'binder', estimate = sim_matrix)
estimate_partition = as.vector(binder_dahl)
```

```r
# Get quality indicies
Kest = length(unique(estimate_partition))
cat('\n Estimated number of clusters \n')
#>
#>  Estimated number of clusters
Kest
#> [1] 3
Binder_loss = salso::binder(truth = real_partition, estimate = estimate_partition)
cat('\n Binder loss function \n')
#>
#>  Binder loss function
Binder_loss
#> [1] 0.0384
Tab = table(real_partition,estimate_partition)
cat('\n Miss classification table \n')
#>
#>  Miss classification table
Tab
#>               estimate_partition
#> real_partition  0  1  2
#>              1 96  0  0
#>              2  0 96  8
```

Caso simmetrico

```r
suppressWarnings(suppressPackageStartupMessages(library(GDFMM)))
suppressWarnings(suppressPackageStartupMessages(library(ACutils)))
suppressWarnings(suppressPackageStartupMessages(library(tidyverse)))
suppressWarnings(suppressPackageStartupMessages(library(RColorBrewer)))

# color palette ------------------------------------------------------------
mycol = hcl.colors(n=3,palette = "Zissou1")
# data generation ----------------------------------------------------------

d = 1               # number of groups
K = 2               # number of global clusters
mu = c(-2,2)   # vectors of means
sd = c(1,1)      # vector of sd
n_j = rep(200, d)  # set cardinality of the groups
seed = 32156
mycol_cluster = brewer.pal(n=K, name = "Dark2")
#> Warning in brewer.pal(n = K, name = "Dark2"): minimal value for n is 3, returning requested palette

genD = generate_data(d=d, K=K, mu = mu, sd = sd, n_j = n_j, seed = seed)
data = genD$data
real_partition = genD$real_partition
# Run  ---------------------------------------------------------------------


niter  <- 1000
burnin <- 3000
thin   <- 1

option<-list("nu" = 1, "Mstar0" = 2, "Lambda0" = 3, "mu0" = 0,"sigma0"= 1, "gamma0" = 1,
```

```
              "Adapt_MH_hyp1"= 0.7,"Adapt_MH_hyp2"= 0.234, "Adapt_MH_power_lim"=10, "Adapt_MH_var0"=1,
              "k0"= 1/10, "nu0"=10, "alpha_gamma"=1,
              "beta_gamma"=1, "alpha_lambda"=1, "beta_lambda"=1,
              "UpdateU" = T, "UpdateM" = T, "UpdateGamma" = T, "UpdateS" = T,
              "UpdateTau" = T, "UpdateLambda" = T
)

#GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, option = option)
GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, FixPartition = F, option = option)
```




```
# COMPUTE BINDER LOSS FUNCTION TO SELECT BEST PARTITION --------------------

# Get labels for each iterations for each data point
part_matrix <- GDFMM$Partition

# Compute similarity matrix
sim_matrix <- psm(part_matrix)

# Get estimated partition according to binder (or VI) loss functions.
# VI_dahl <- dlso(matr, loss = 'VI', estimate=NULL)
binder_dahl <- dlso(part_matrix, loss = 'binder', estimate = sim_matrix)
estimate_partition = as.vector(binder_dahl)

# Get quality indicies
Kest = length(unique(estimate_partition))
cat('\n Estimated number of clusters \n')
```

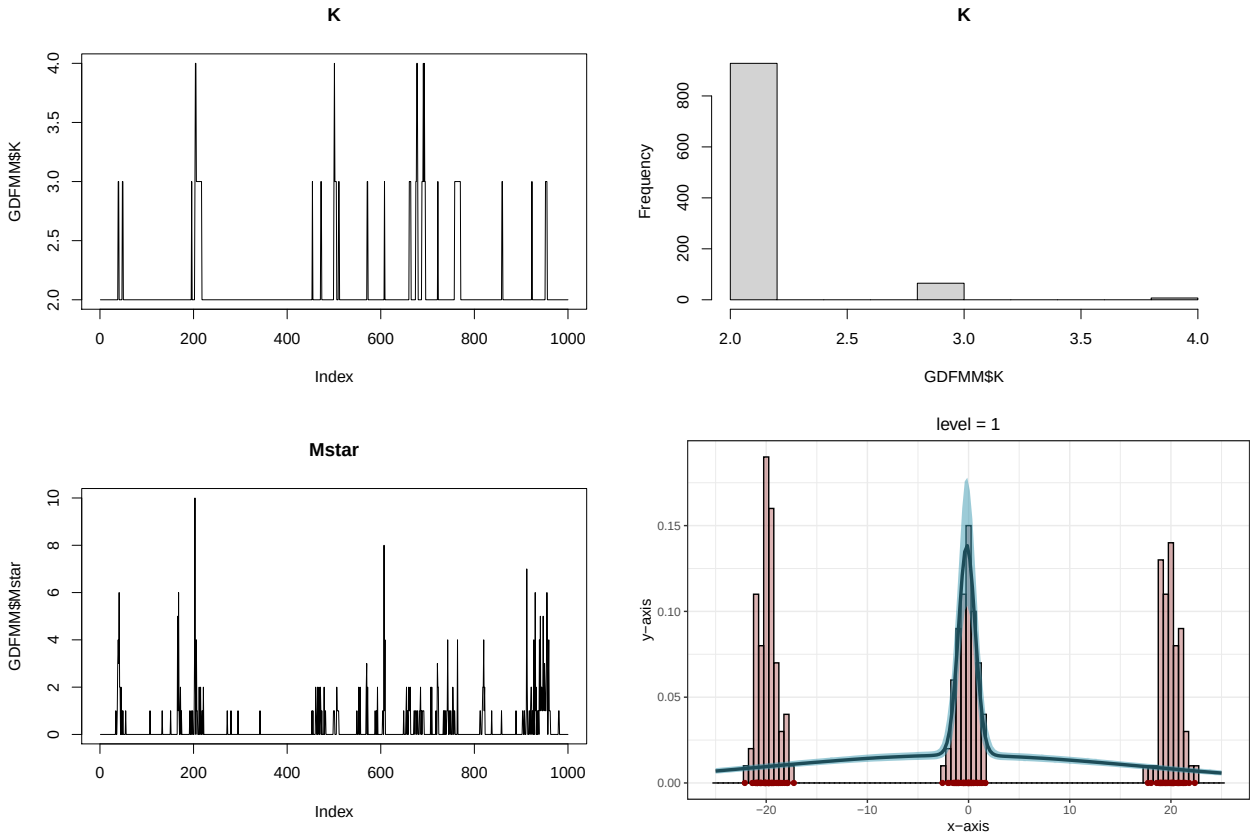
```
#>
#>  Estimated number of clusters
Kest
#> [1] 4
Binder_loss = salso::binder(truth = real_partition, estimate = estimate_partition)
cat('\n Binder loss function \n')
#>
#>  Binder loss function
Binder_loss
#> [1] 0.2062
Tab = table(real_partition,estimate_partition)
cat('\n Miss classification table \n')
#>
#>  Miss classification table
Tab
#>               estimate_partition
#> real_partition  0  1  2  3
#>              1 85  2  0  8
#>              2  1 63 32  9
```

Secondo caso, K = 3

Caso simmetrico.

```
suppressWarnings(suppressPackageStartupMessages(library(GDFMM)))
suppressWarnings(suppressPackageStartupMessages(library(ACutils)))
suppressWarnings(suppressPackageStartupMessages(library(tidyverse)))
suppressWarnings(suppressPackageStartupMessages(library(RColorBrewer)))
# data generation ------------------------------------------------------------

d = 1                # number of groups
K = 3                # number of global clusters
mu = c(-20,0, 20)    # vectors of means
sd = c(1,1,1)       # vector of sd
n_j = rep(200, d)   # set cardinality of the groups
seed = 124123
mycol_cluster = brewer.pal(n=K, name = "Dark2")

genD = generate_data(d=d, K=K, mu = mu, sd = sd, n_j = n_j, seed = seed)
data = genD$data
real_partition = genD$real_partition

# Run  -----------------------------------------------------------------------


niter  <- 1000
burnin <- 3000
thin   <- 1

option<-list("nu" = 1, "Mstar0" = 2, "Lambda0" = 3, "mu0" = 0,"sigma0"= 1, "gamma0" = 1,
             "Adapt_MH_hyp1"= 0.7,"Adapt_MH_hyp2"= 0.234, "Adapt_MH_power_lim"=10, "Adapt_MH_var0"=1,
             "k0"= 1/10, "nu0"=10, "alpha_gamma"=1,
             "beta_gamma"=1, "alpha_lambda"=1, "beta_lambda"=1,
             "UpdateU" = T, "UpdateM" = T, "UpdateGamma" = T, "UpdateS" = T,
             "UpdateTau" = T, "UpdateLambda" = T
```

```
)

#GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, option = option)
GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, FixPartition = F, option = option)
```

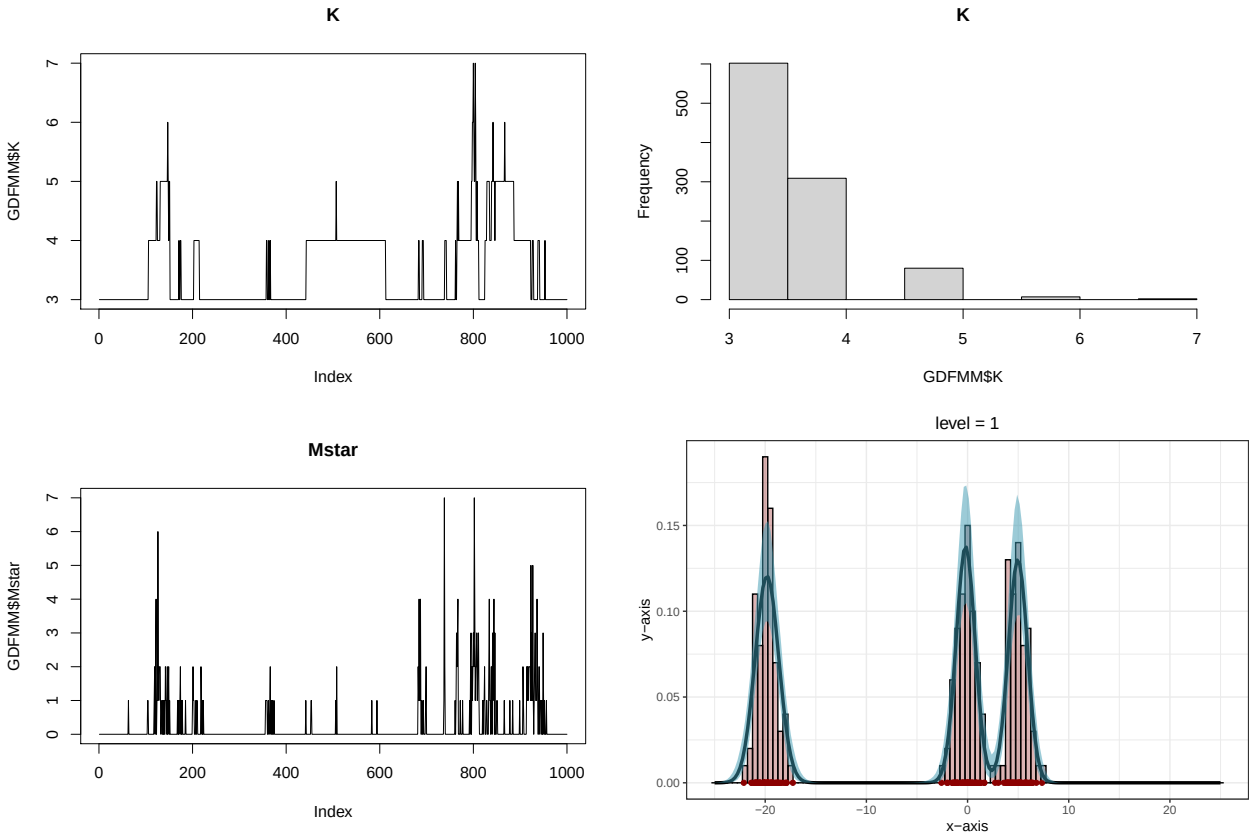



```
# COMPUTE BINDER LOSS FUNCTION TO SELECT BEST PARTITION --------------------

# Get labels for each iterations for each data point
part_matrix <- GDFMM$Partition

# Compute similarity matrix
sim_matrix <- psm(part_matrix)

# Get estimated partition according to binder (or VI) loss functions.
# VI_dahl <- dlso(matr, loss = 'VI', estimate=NULL)
binder_dahl <- dlso(part_matrix, loss = 'binder', estimate = sim_matrix)
estimate_partition = as.vector(binder_dahl)

# Get quality indicies
Kest = length(unique(estimate_partition))
cat('\n Estimated number of clusters \n')
#>
#>  Estimated number of clusters
Kest
#> [1] 2
Binder_loss = salso::binder(truth = real_partition, estimate = estimate_partition)
```

```
cat('\n Binder loss function \n')
#>
#>  Binder loss function
Binder_loss
#> [1] 0.3218
Tab = table(real_partition,estimate_partition)
cat('\n Miss classification table \n')
#>
#>  Miss classification table
Tab
#>               estimate_partition
#> real_partition  0  1
#>              1 72  0
#>              2 10 55
#>              3 63  0
```

Caso NON simmetrico.

```
suppressWarnings(suppressPackageStartupMessages(library(GDFMM)))
suppressWarnings(suppressPackageStartupMessages(library(ACutils)))
suppressWarnings(suppressPackageStartupMessages(library(tidyverse)))
suppressWarnings(suppressPackageStartupMessages(library(RColorBrewer)))
# data generation ------------------------------------------------------------

d = 1                # number of groups
K = 3                # number of global clusters
mu = c(-20,0, 5)    # vectors of means
sd = c(1,1,1)       # vector of sd
n_j = rep(200, d)  # set cardinality of the groups
seed = 124123
mycol_cluster = brewer.pal(n=K, name = "Dark2")

genD = generate_data(d=d, K=K, mu = mu, sd = sd, n_j = n_j, seed = seed)
data = genD$data
real_partition = genD$real_partition

# Run  -----------------------------------------------------------------------


niter  <- 1000
burnin <- 3000
thin   <- 1

option<-list("nu" = 1, "Mstar0" = 2, "Lambda0" = 3, "mu0" = 0,"sigma0"= 1, "gamma0" = 1,
             "Adapt_MH_hyp1"= 0.7,"Adapt_MH_hyp2"= 0.234, "Adapt_MH_power_lim"=10, "Adapt_MH_var0"=1,
             "k0"= 1/10, "nu0"=10, "alpha_gamma"=1,
             "beta_gamma"=1, "alpha_lambda"=1, "beta_lambda"=1,
             "UpdateU" = T, "UpdateM" = T, "UpdateGamma" = T, "UpdateS" = T,
             "UpdateTau" = T, "UpdateLambda" = T
)

#GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, option = option)
GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, FixPartition = F, option = option)
```



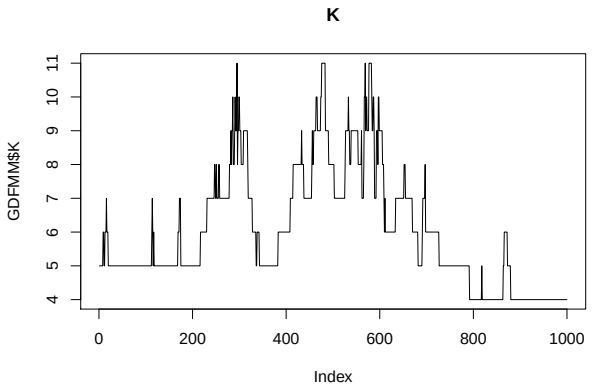

```
# COMPUTE BINDER LOSS FUNCTION TO SELECT BEST PARTITION -------------------

# Get labels for each iterations for each data point
part_matrix <- GDFMM$Partition

# Compute similarity matrix
sim_matrix <- psm(part_matrix)

# Get estimated partition according to binder (or VI) loss functions.
# VI_dahl <- dlso(matr, loss = 'VI', estimate=NULL)
binder_dahl <- dlso(part_matrix, loss = 'binder', estimate = sim_matrix)
estimate_partition = as.vector(binder_dahl)

# Get quality indicies
Kest = length(unique(estimate_partition))
cat('\n Estimated number of clusters \n')
#>
#>  Estimated number of clusters
Kest
#> [1] 3
Binder_loss = salso::binder(truth = real_partition, estimate = estimate_partition)
cat('\n Binder loss function \n')
#>
#>  Binder loss function
Binder_loss
#> [1] 0
```

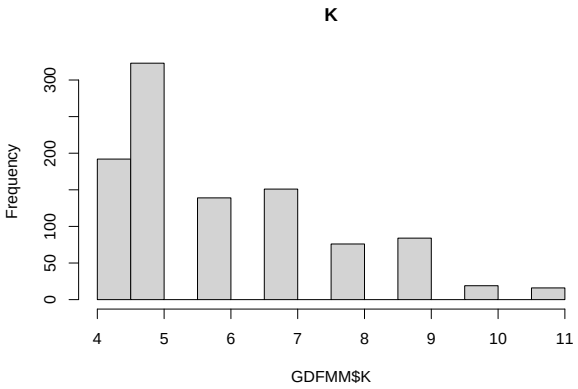
```r
Tab = table(real_partition,estimate_partition)
cat('\n Miss classification table \n')
#>
#>  Miss classification table
Tab
#>                estimate_partition
#> real_partition  0  1  2
#>              1 72  0  0
#>              2  0  0 65
#>              3  0 63  0
```

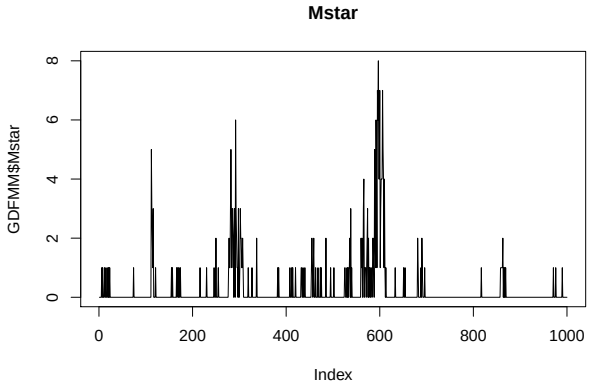
Caso d = 2

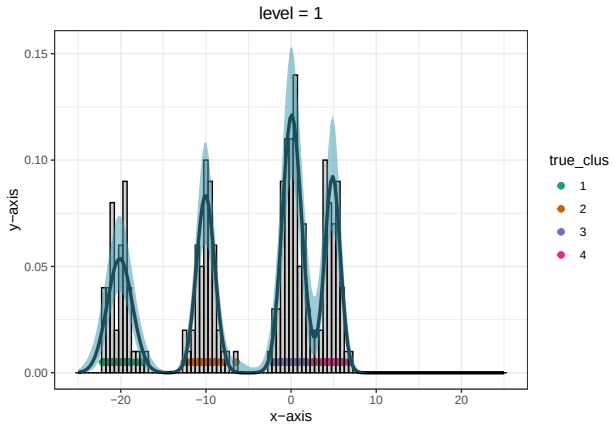
Fisso K = 4 Caso NON simmetrico.

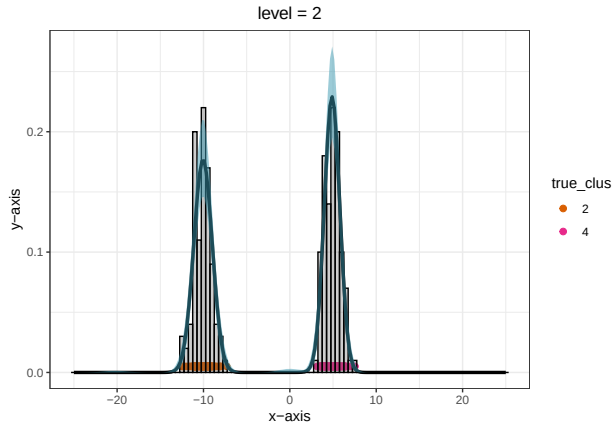
```r
suppressWarnings(suppressPackageStartupMessages(library(GDFMM)))
suppressWarnings(suppressPackageStartupMessages(library(ACutils)))
suppressWarnings(suppressPackageStartupMessages(library(tidyverse)))
suppressWarnings(suppressPackageStartupMessages(library(RColorBrewer)))

# color palette ------------------------------------------------------------
mycol = hcl.colors(n=3,palette = "Zissou1")
# data generation ----------------------------------------------------------

d = 2               # number of groups
K = 4               # number of global clusters
mu = c(-20,-10,0, 5)   # vectors of means
sd = c(1,1,1,1)      # vector of sd
n_j = rep(200, d)  # set cardinality of the groups
seed = 20051131
mycol_cluster = brewer.pal(n=K, name = "Dark2")

genD = generate_data(d=d, K=K, mu = mu, sd = sd, n_j = n_j, seed = seed)
data = genD$data
real_partition = genD$real_partition

# Run  ---------------------------------------------------------------------


niter  <- 1000
burnin <- 3000
thin   <- 1

option<-list("nu" = 1, "Mstar0" = 2, "Lambda0" = 3, "mu0" = 0,"sigma0"= 1, "gamma0" = 1,
             "Adapt_MH_hyp1"= 0.7,"Adapt_MH_hyp2"= 0.234, "Adapt_MH_power_lim"=10, "Adapt_MH_var0"=1,
             "k0"= 1/10, "nu0"=10, "alpha_gamma"=1,
             "beta_gamma"=1, "alpha_lambda"=1, "beta_lambda"=1,
             "UpdateU" = T, "UpdateM" = T, "UpdateGamma" = T, "UpdateS" = T,
             "UpdateTau" = T, "UpdateLambda" = T
)

#GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, option = option)
GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, FixPartition = F, option = option)
```












```
# COMPUTE BINDER LOSS FUNCTION TO SELECT BEST PARTITION --------------------

# Get labels for each iterations for each data point
part_matrix <- GDFMM$Partition

# Compute similarity matrix
sim_matrix <- psm(part_matrix)

# Get estimated partition according to binder (or VI) loss functions.
# VI_dahl <- dlso(matr, loss = 'VI', estimate=NULL)
binder_dahl <- dlso(part_matrix, loss = 'binder', estimate = sim_matrix)
estimate_partition = as.vector(binder_dahl)
```

```r
# Get quality indicies
Kest = length(unique(estimate_partition))
cat('\n Estimated number of clusters \n')
#>
#>  Estimated number of clusters
Kest
#> [1] 5
Binder_loss = salso::binder(truth = real_partition, estimate = estimate_partition)
cat('\n Binder loss function \n')
#>
#>  Binder loss function
Binder_loss
#> [1] 0.017975
Tab = table(real_partition,estimate_partition)
cat('\n Miss classification table \n')
#>
#>  Miss classification table
Tab
#>               estimate_partition
#> real_partition   0   1   2   3   4
#>              1  41   0   0   0   0
#>              2   0   0 142   0   0
#>              3   0   0   0  44  24
#>              4   0 147   0   2   0
```

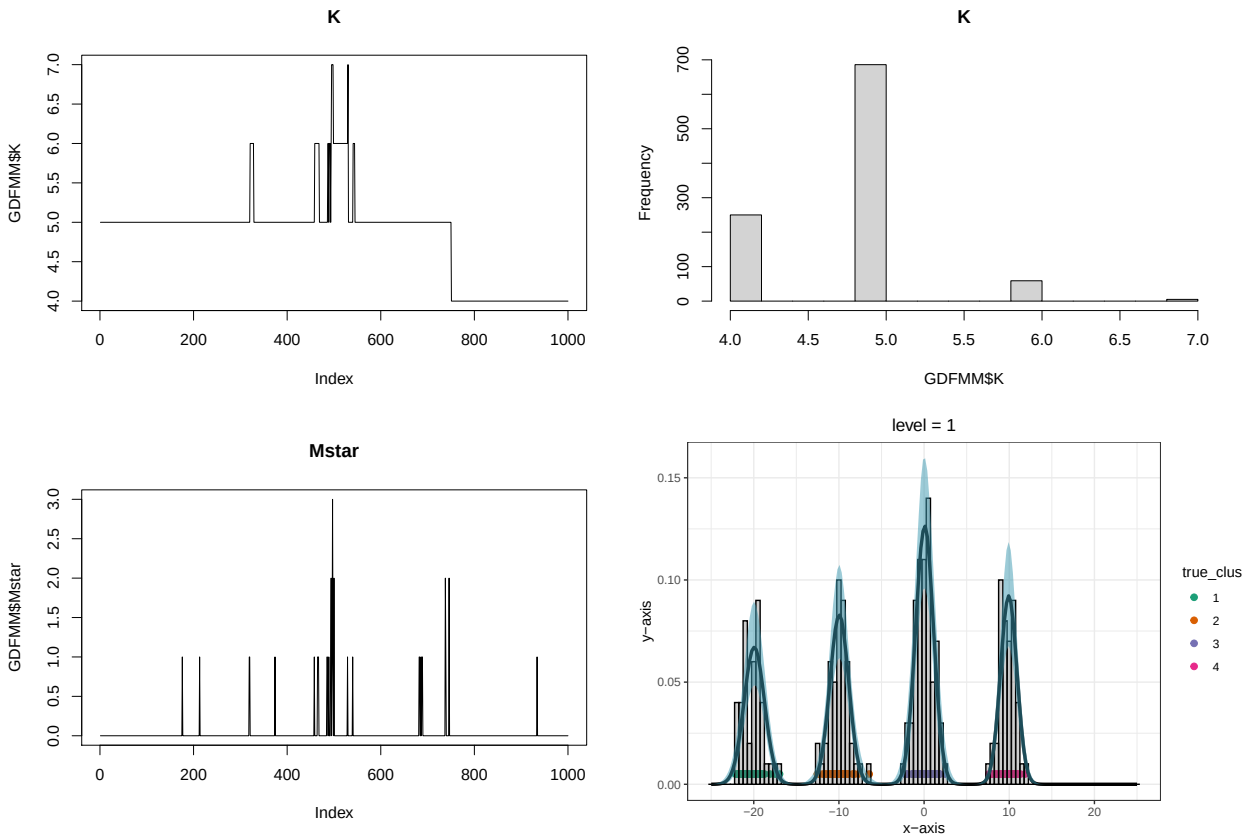
d=3

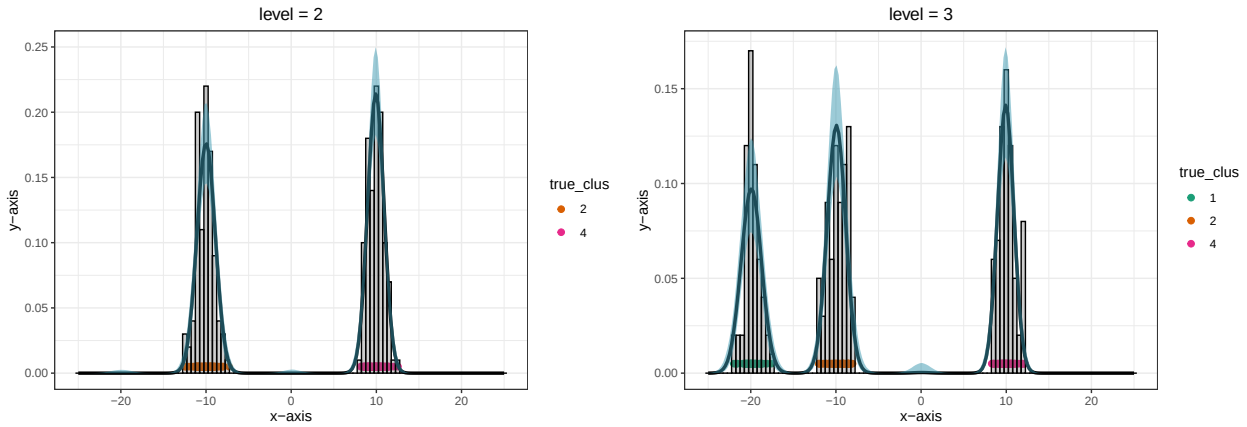
K = 4

Primo caso, default come gli altri.

```r
suppressWarnings(suppressPackageStartupMessages(library(GDFMM)))
suppressWarnings(suppressPackageStartupMessages(library(ACutils)))
suppressWarnings(suppressPackageStartupMessages(library(tidyverse)))
suppressWarnings(suppressPackageStartupMessages(library(RColorBrewer)))


# color palette -------------------------------------------------------------
mycol = hcl.colors(n=3,palette = "Zissou1")

# data generation -----------------------------------------------------------

d = 3               # number of groups
K = 4               # number of global clusters
mu = c(-20,-10,0, 10)   # vectors of means
sd = c(1,1,1,1)      # vector of sd
n_j = rep(200, d)  # set cardinality of the groups
seed = 20051131

genD = generate_data(d=d, K=K, mu = mu, sd = sd, n_j = n_j, seed = seed)
data = genD$data
real_partition = genD$real_partition
```

```r
mycol_cluster = brewer.pal(n=K, name = "Dark2")
# Run  ----------------------------------------------------------------------


niter  <- 1000
burnin <- 3000
thin   <- 1

option<-list("nu" = 1, "Mstar0" = 2, "Lambda0" = 3, "mu0" = 0,"sigma0"= 1, "gamma0" = 1,
             "Adapt_MH_hyp1"= 0.7,"Adapt_MH_hyp2"= 0.234, "Adapt_MH_power_lim"=10, "Adapt_MH_var0"=1,
             "k0"= 1/10, "nu0"=10, "alpha_gamma"=1,
             "beta_gamma"=1, "alpha_lambda"=1, "beta_lambda"=1,
             "UpdateU" = T, "UpdateM" = T, "UpdateGamma" = T, "UpdateS" = T,
             "UpdateTau" = T, "UpdateLambda" = T
)

#GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, option = option)
GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, FixPartition = F, option = option)
```

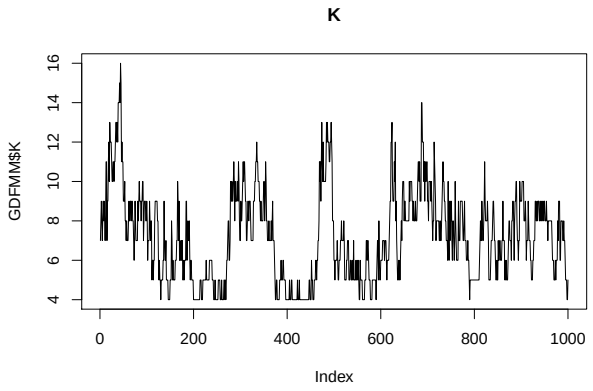


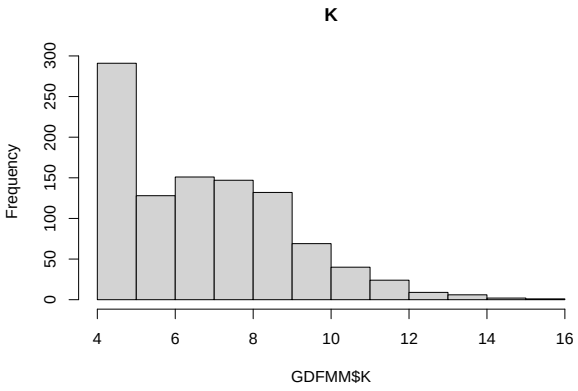


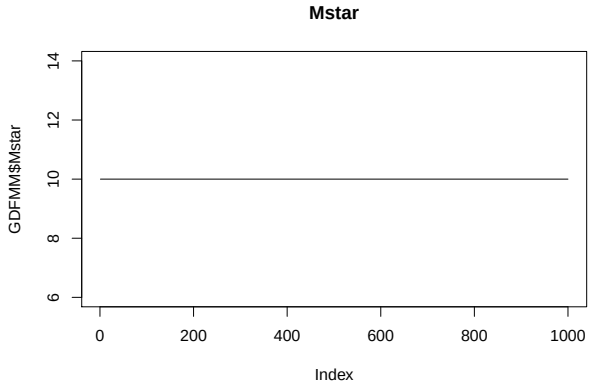

```
# COMPUTE BINDER LOSS FUNCTION TO SELECT BEST PARTITION -------------------

# Get labels for each iterations for each data point
part_matrix <- GDFMM$Partition

# Compute similarity matrix
sim_matrix <- psm(part_matrix)

# Get estimated partition according to binder (or VI) loss functions.
# VI_dahl <- dlso(matr, loss = 'VI', estimate=NULL)
binder_dahl <- dlso(part_matrix, loss = 'binder', estimate = sim_matrix)
estimate_partition = as.vector(binder_dahl)

# Get quality indicies
Kest = length(unique(estimate_partition))
cat('\n Estimated number of clusters \n')
#>
#>  Estimated number of clusters
Kest
#> [1] 5
Binder_loss = salso::binder(truth = real_partition, estimate = estimate_partition)
cat('\n Binder loss function \n')
#>
#>  Binder loss function
Binder_loss
#> [1] 0.048
Tab = table(real_partition,estimate_partition)
cat('\n Miss classification table \n')
#>
#>  Miss classification table
Tab
#>               estimate_partition
#> real_partition   0   1   2   3   4
#>              1   0 100   0   0   0
#>              2   0   0  54 160   0
#>              3   0   0   0   0  68
#>              4 218   0   0   0   0
```

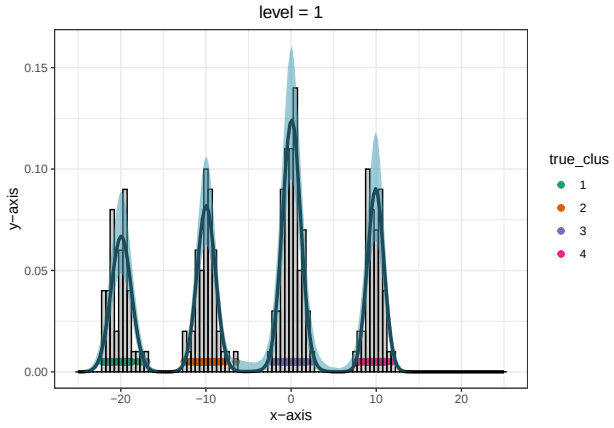
Secondo caso, fisso `Mstar = 10`. Mixxa molto meglio.

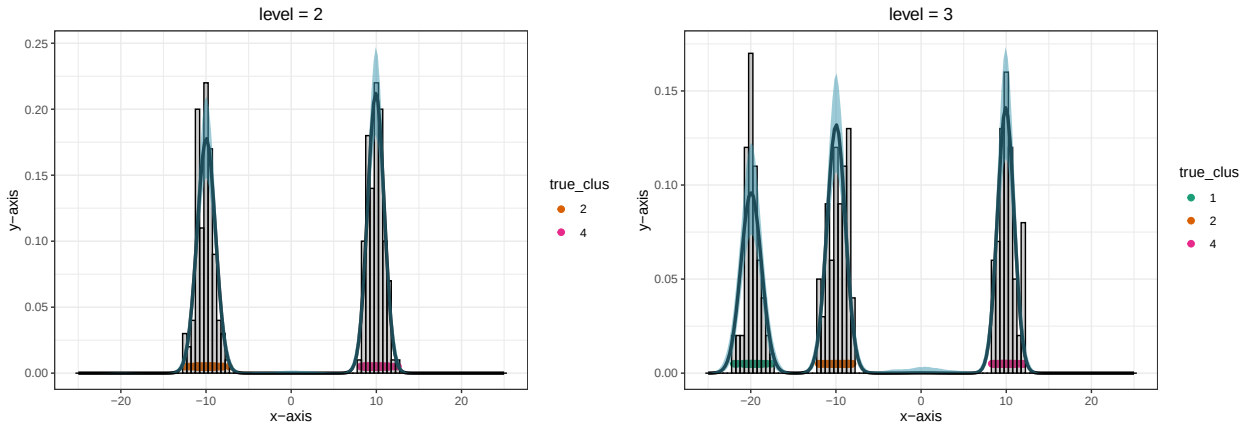
```r
# Run  ----------------------------------------------------------------------


niter  <- 1000
burnin <- 3000
thin   <- 1

option<-list("nu" = 1, "Mstar0" = 10, "Lambda0" = 3, "mu0" = 0,"sigma0"= 1, "gamma0" = 1,
             "Adapt_MH_hyp1"= 0.7,"Adapt_MH_hyp2"= 0.234, "Adapt_MH_power_lim"=10, "Adapt_MH_var0"=1,
             "k0"= 1/10, "nu0"=10, "alpha_gamma"=1,
             "beta_gamma"=1, "alpha_lambda"=1, "beta_lambda"=1,
             "UpdateU" = T, "UpdateM" = F, "UpdateGamma" = T, "UpdateS" = T,
             "UpdateTau" = T, "UpdateLambda" = T
)

#GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, option = option)
GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, FixPartition = F, option = option)
```









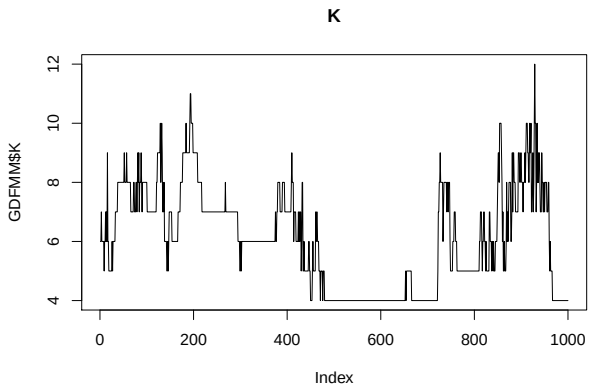


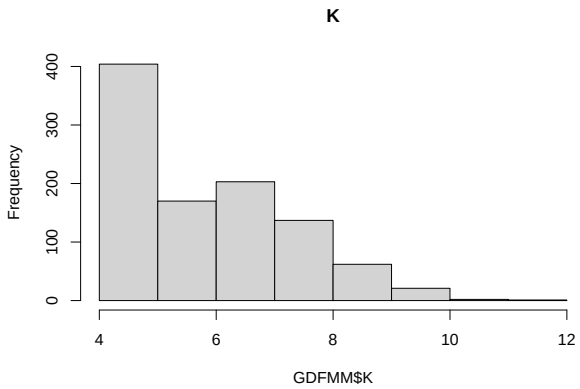

```
# COMPUTE BINDER LOSS FUNCTION TO SELECT BEST PARTITION -------------------

# Get labels for each iterations for each data point
part_matrix <- GDFMM$Partition

# Compute similarity matrix
sim_matrix <- psm(part_matrix)

# Get estimated partition according to binder (or VI) loss functions.
# VI_dahl <- dlso(matr, loss = 'VI', estimate=NULL)
binder_dahl <- dlso(part_matrix, loss = 'binder', estimate = sim_matrix)
estimate_partition = as.vector(binder_dahl)

# Get quality indicies
Kest = length(unique(estimate_partition))
cat('\n Estimated number of clusters \n')
#>
#>  Estimated number of clusters
Kest
#> [1] 8
Binder_loss = salso::binder(truth = real_partition, estimate = estimate_partition)
cat('\n Binder loss function \n')
#>
#>  Binder loss function
Binder_loss
#> [1] 0.006511111
Tab = table(real_partition,estimate_partition)
cat('\n Miss classification table \n')
#>
#>  Miss classification table
Tab
#>               estimate_partition
#> real_partition   0   1   2   3   4   5   6   7
#>              1 100   0   0   0   0   0   0   0
#>              2   0   0 213   0   0   1   0   0
#>              3   0   0   0  50  14   0   1   3
#>              4   0 218   0   0   0   0   0   0
```

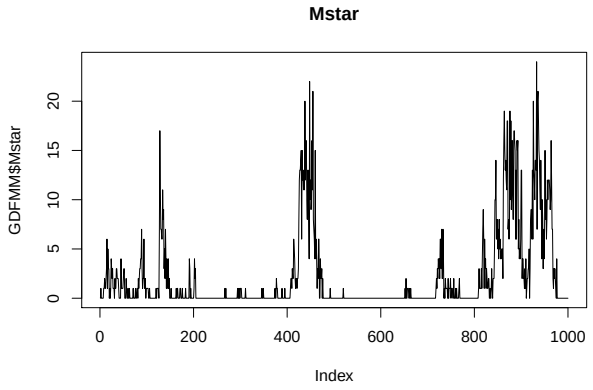
Terzo caso, `Mstar` random ma aumento prior su numbero di componenti.

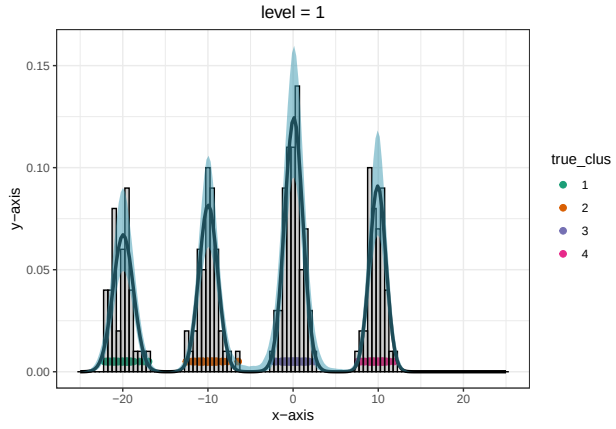
Visto che $\Lambda \sim Gamma(a_\Lambda, b_\Lambda)$, metto $a_\Lambda = 15$.

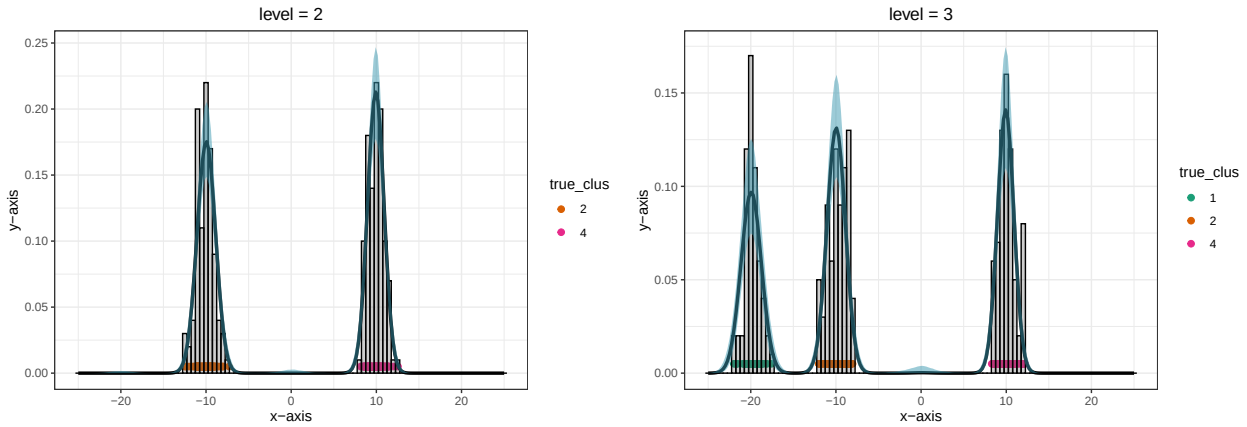
```
# Run  ----------------------------------------------------------------------


niter  <- 1000
burnin <- 3000
thin   <- 1

option<-list("nu" = 1, "Mstar0" = 10, "Lambda0" = 3, "mu0" = 0,"sigma0"= 1, "gamma0" = 1,
             "Adapt_MH_hyp1"= 0.7,"Adapt_MH_hyp2"= 0.234, "Adapt_MH_power_lim"=10, "Adapt_MH_var0"=1,
             "k0"= 1/10, "nu0"=10, "alpha_gamma"=1,
             "beta_gamma"=1, "alpha_lambda"=15, "beta_lambda"=1,
             "UpdateU" = T, "UpdateM" = T, "UpdateGamma" = T, "UpdateS" = T,
             "UpdateTau" = T, "UpdateLambda" = T
)

#GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, option = option)
GDFMM = GDFMM_sampler(data, niter, burnin, thin, seed = 123, FixPartition = F, option = option)
```












```
# COMPUTE BINDER LOSS FUNCTION TO SELECT BEST PARTITION --------------------

# Get labels for each iterations for each data point
part_matrix <- GDFMM$Partition

# Compute similarity matrix
sim_matrix <- psm(part_matrix)

# Get estimated partition according to binder (or VI) loss functions.
# VI_dahl <- dlso(matr, loss = 'VI', estimate=NULL)
binder_dahl <- dlso(part_matrix, loss = 'binder', estimate = sim_matrix)
estimate_partition = as.vector(binder_dahl)

# Get quality indicies
Kest = length(unique(estimate_partition))
cat('\n Estimated number of clusters \n')
#>
#>  Estimated number of clusters
Kest
#> [1] 6
Binder_loss = salso::binder(truth = real_partition, estimate = estimate_partition)
cat('\n Binder loss function \n')
#>
#>  Binder loss function
Binder_loss
#> [1] 0.003272222
Tab = table(real_partition,estimate_partition)
cat('\n Miss classification table \n')
#>
#>  Miss classification table
Tab
#>               estimate_partition
#> real_partition   0   1   2   3   4   5
#>              1 100   0   0   0   0   0
#>              2   0   0 214   0   0   0
#>              3   0   0   0   9  58   1
#>              4   0 218   0   0   0   0
```