A quick start guide to IMIX

Ziqiao Wang

2020/2/6

Contents

1 Introduction

IMIX is an R package for integration of multiple genomics data types to investigate the associations between genes and a specific outcome, including binary, continuous, survival, and categorical outcomes based on finite multivariate Gaussian mixture modelling using summary statistics. The input is summary statistics for each data type including either p-value or z-score. We support summary statistics of data outputs such as DNA methylation, copy number variation(CNV), and gene expression (RNAseq/microarray) at gene level. Nonethelss, IMIX is as flexible as it can be extended to other molecular level as long as the summary statistics of the multiple data types are coherent with each other. It provides features to select the true number of components behind the data, parameter estimation for the summary statistics via EM algorithm. The most important feature is that it evaluates the data through different covariance and mean structures of mixture modelling and selects the overall best fitting model, which in turn provides the oracle output while controlling for the global false discovery rate (FDR) at a user specified level.

This document gives a quick tour of IMIX functionalities. The tasks addressed in this package include assessment of the true number of components with respect to the data, identification of interesting genes for each data type combination, FDR control, and plotting functionality. See help(package="IMIX") for further details and references provided by citation("IMIX").

2 Data Preparation

2.1 Example data 1: p values for RNAseq and CNV data

Each row is a gene, each column is a data type. The dimension of the input data is 1000×2.

```
data("data_p")
dim(data_p)
```

```
## [1] 1000    2
```

```
head(data_p)
```

```
##             p.rnaseq     p.cnv
## ALOX12B 6.087266e-08 0.2283308
## DNAJC7  1.547188e-05 0.1438505
## S100A8  6.965336e-20 0.4140232
## SNORD8  9.907678e-01 0.3540862
## GSTM5   5.138138e-07 0.9842864
## VSTM2L  1.327199e-02 0.7146840
```

2.2 Example data 2: Simulate z scores for DNA methylation, RNAseq, and CNV data

The dimension is 1000×3.

```
library(MASS)
N = 1000
truelabel <- sample(1:8,
                    prob = rep(0.125, 8),
                    size = N,
                    replace = TRUE)
mu1 = c(0, 5)
mu2 = c(0, 5)
mu3 = c(0, 5)
mu1_mv = c(mu1[1], mu2[1], mu3[1])
mu2_mv = c(mu1[2], mu2[1], mu3[1])
mu3_mv = c(mu1[1], mu2[2], mu3[1])
mu4_mv = c(mu1[1], mu2[1], mu3[2])
mu5_mv = c(mu1[2], mu2[2], mu3[1])
mu6_mv = c(mu1[2], mu2[1], mu3[2])
mu7_mv = c(mu1[1], mu2[2], mu3[2])
mu8_mv = c(mu1[2], mu2[2], mu3[2])
```

```
cov_sim = list()
for (i in 1:8) {
  cov_sim[[i]] = diag(3)
}
data_z = array(0, c(N, 3))
data_z[which(truelabel == 1),] = mvrnorm(
  n = length(which(truelabel == 1)),
  mu = mu1_mv,
  Sigma = cov_sim[[1]],
  tol = 1e-6,
  empirical = FALSE
)
data_z[which(truelabel == 2),] = mvrnorm(
  n = length(which(truelabel == 2)),
  mu = mu2_mv,
  Sigma = cov_sim[[2]],
  tol = 1e-6,
  empirical = FALSE
)
data_z[which(truelabel == 3),] = mvrnorm(
  n = length(which(truelabel == 3)),
  mu = mu3_mv,
  Sigma = cov_sim[[3]],
  tol = 1e-6,
  empirical = FALSE
)
data_z[which(truelabel == 4),] = mvrnorm(
  n = length(which(truelabel == 4)),
  mu = mu4_mv,
  Sigma = cov_sim[[4]],
  tol = 1e-6,
  empirical = FALSE
)
data_z[which(truelabel == 5),] = mvrnorm(
  n = length(which(truelabel == 5)),
  mu = mu5_mv,
  Sigma = cov_sim[[5]],
  tol = 1e-6,
  empirical = FALSE
)
data_z[which(truelabel == 6),] = mvrnorm(
  n = length(which(truelabel == 6)),
  mu = mu6_mv,
  Sigma = cov_sim[[6]],
  tol = 1e-6,
  empirical = FALSE
)
data_z[which(truelabel == 7),] = mvrnorm(
  n = length(which(truelabel == 7)),
  mu = mu7_mv,
  Sigma = cov_sim[[7]],
  tol = 1e-6,
  empirical = FALSE
```

```
)
data_z[which(truelabel == 8),] = mvrnorm(
  n = length(which(truelabel == 8)),
  mu = mu8_mv,
  Sigma = cov_sim[[8]],
  tol = 1e-6,
  empirical = FALSE
)

rownames(data_z) = paste0("gene", 1:N)
colnames(data_z) = c("z.methy", "z.ge", "z.cnv")
dim(data_z)
```

```
## [1] 1000    3
```

3 Model selection: select the number of components for the mixture distribution

3.1 Example 1: Two data types, p values

```
select_comp1 = model_selection_component(data_p, data_type = "p", seed = 20)
```

```
## Start Number of Component Selections!
## Test for 1 Component!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 2 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 3 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 4 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Done!
```

```
names(select_comp1)
```

```
## [1] "Component_Selected_AIC" "Component_Selected_BIC" "AIC/BIC"
## [4] "IMIX_ind_unrestrict"    "IMIX_cor_twostep"       "IMIX_cor"
```

```
select_comp1$Component_Selected_AIC
```

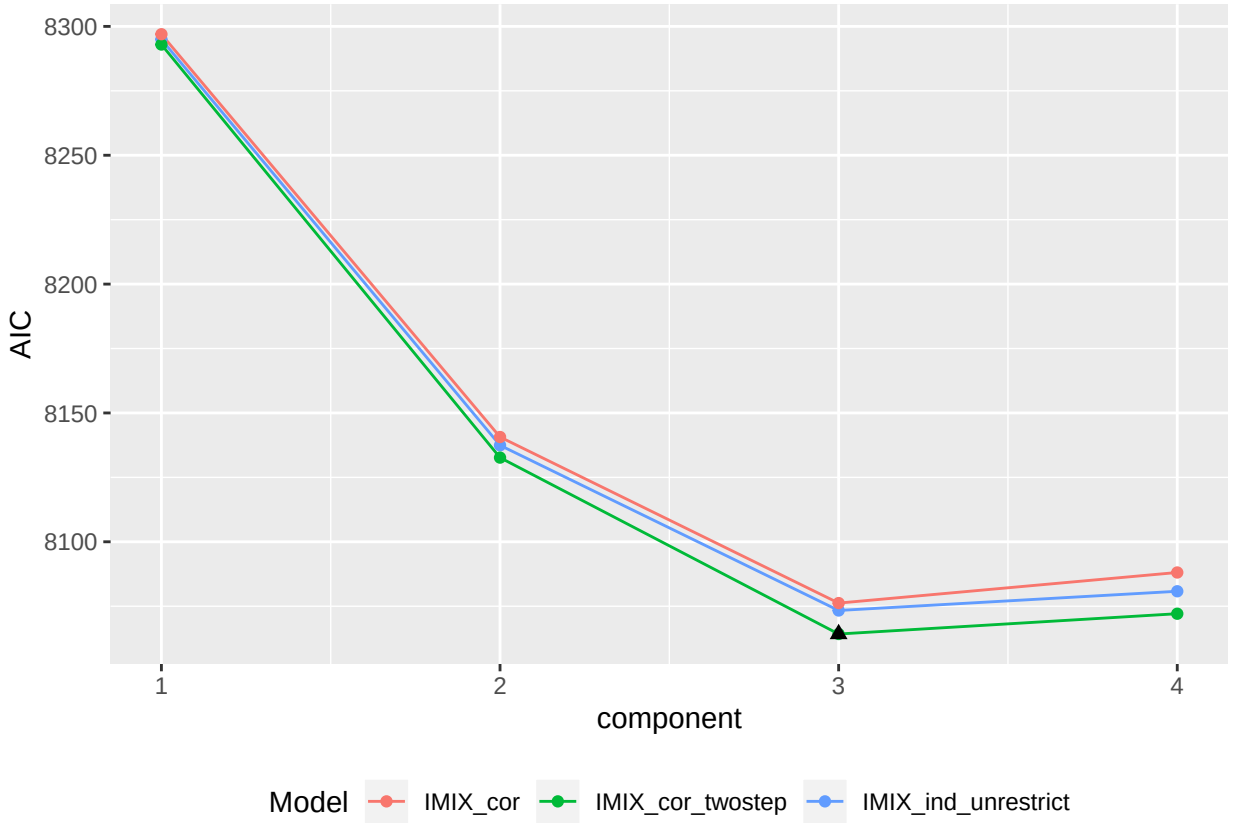
```
## [1] component3
## Levels: component1 component2 component3 component4
```

```
select_comp1$Component_Selected_BIC
```

```
## [1] component3
## Levels: component1 component2 component3 component4
```

The model selected 3 components out of 4. Then we visualize it.

```
plot_component(select_comp1, type = "AIC")
```



3.2 Example 2: Three data types, z scores

```
select_comp2 = model_selection_component(data_z, data_type = "z")
```

```
## Start Number of Component Selections!
```

```
## Test for 1 Component!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 2 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 3 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 4 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 5 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 6 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 7 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Test for 8 Components!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Done!
```

```
names(select_comp2)
```

```
## [1] "Component_Selected_AIC" "Component_Selected_BIC" "AIC/BIC"
## [4] "IMIX_ind_unrestrict"    "IMIX_cor_twostep"       "IMIX_cor"
```

```
select_comp2$Component_Selected_AIC
```

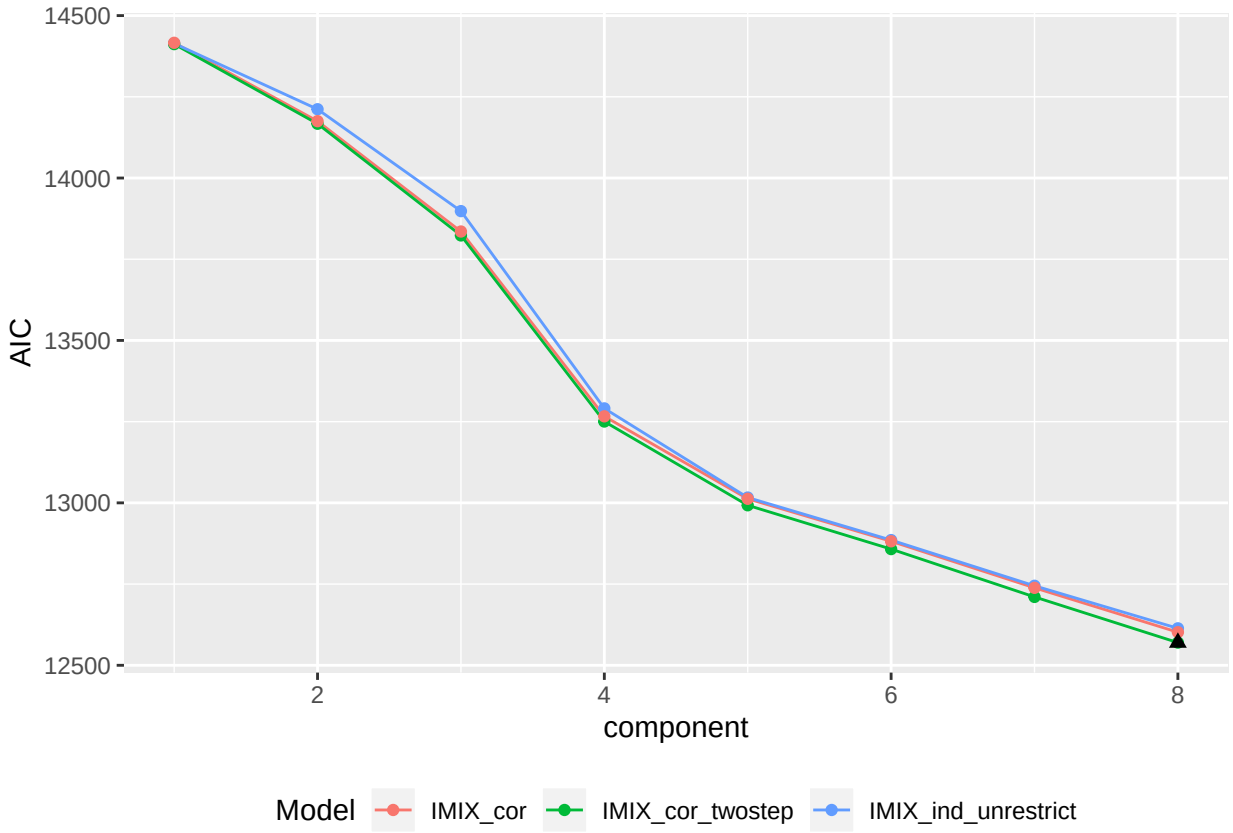
```
## [1] component8
## 8 Levels: component1 component2 component3 component4 ... component8
```

```
select_comp2$Component_Selected_BIC
```

```
## [1] component8
## 8 Levels: component1 component2 component3 component4 ... component8
```

The model selected all 8 components. Then we visualize it.

```
plot_component(select_comp2, type = "AIC")
```



4 Integrative genomics test for two and three omics data types

4.1 Example 1: Two data types, p values

Initial values for the p transformed z score mixture model, this step can be skipped

```
mu_input = list() # generate an initial list for mean
mu_input[[1]] = c(0, 0) # mean vector for component 1, data 1 null and data 2 null
mu_input[[2]] = c(3, 0) # mean vector for component 2, data 1 nonnull and data 2 null
mu_input[[3]] = c(0, 3) # mean vector for component 3, data 1 null and data 2 nonnull
mu_input[[4]] = c(3, 3) # mean vector for component 4, data 1 nonnull and data 2 nonnull

cov_input = list() # generate an initial list for the covariance matrices
for (i in 1:4) {
cov_input[[i]] = diag(3)
```

```
}

p_input = rep(0.25, 4) # initial value for the proportion of components
```

Start the test

```
test1 = IMIX(
  data_input = data_p,
  data_type = "p",
  mu_ini = mu_input,
  cov_ini = cov_input,
  p_ini = p_input,
  alpha = 0.1
  )
```

```
## Start IMIX-ind procedure!
## Successfully Done!
## Start IMIX-cor-twostep procedure!
## Successfully Done!
## Start IMIX-cor model procedure!
## Successfully Done!
## Start IMIX-cor-restrict procedure!
## Successfully Done!
## Start Model Selection!
## Start Adaptive FDR Control!
## All Done!
```

Result outputs of example 1

```
test1$estimatedFDR # Print the estimated global FDR for each component
```

```
## $estimated_mFDR_comp2
## [1] 0.09948009
##
## $estimated_mFDR_comp3
## [1] 0
##
## $estimated_mFDR_comp4
## [1] 0.09991635
```

```
test1$`AIC/BIC` # The AIC and BIC values for each model
```

```
##                        AIC      BIC
## IMIX_ind          8068.002 8121.987
## IMIX_cor_twostep  8067.853 8141.469
## IMIX_cor          8083.875 8196.753
## IMIX_cor_restrict 8075.222 8168.470
```

```
test1$`Selected Model` # The best fitted model selected by AIC
```

```
## [1] "IMIX_cor_twostep"
```

```
str(test1$IMIX_cor_twostep)
```

```
## List of 8
```

```
##  $ posterior prob      : num [1:1000, 1:4] 1.03e-03 1.93e-02 6.20e-10 7.28e-01 1.15e-03 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:1000] "ALOX12B" "DNAJC7" "S100A8" "SNORD8" ...
##   .. ..$ : chr [1:4] "component1" "component2" "component3" "component4"
##  $ Full LogLik all     : num [1:29] -52129 -4116 -4028 -4022 -4021 ...
##  $ Full MaxLogLik final: num -4019
##  $ iterations          : num 29
##  $ pi                  : num [1:4] 0.3679 0.58 0.0111 0.041
##  $ mu                  :List of 4
##   ..$ : num [1:2] 0.617 0.475
##   ..$ : num [1:2] 3.758 0.475
##   ..$ : num [1:2] 0.617 4.802
##   ..$ : num [1:2] 3.76 4.8
##  $ cov                 :List of 4
##   ..$ : num [1:2, 1:2] 1.495 0.133 0.133 1.33
##   ..$ : num [1:2, 1:2] 4.975 -0.152 -0.152 1.367
##   ..$ : num [1:2, 1:2] 3.545 -0.25 -0.25 0.168
##   ..$ : num [1:2, 1:2] 5.967 -0.476 -0.476 1.049
##  $ g                   : num 4
```

```
dim(test1$significant_genes_with_FDRcontrol)
```

```
## [1] 1000    3
```

```
head(test1$significant_genes_with_FDRcontrol)
```

```
##              localFDR class_withoutFDRcontrol class_FDRcontrol
## GLB1     4.181068e-05                       4                4
## SLC22A13 5.310557e-05                       4                4
## XIRP1    6.938574e-05                       4                4
## SCN10A   8.972468e-05                       4                4
## ZNF620   1.707402e-04                       4                4
## CHCHD4   9.825373e-04                       4                4
```

The results for each gene, this includes localFDR, classes with global FDR control at $\alpha = 0.1$ and classes without global FDR control. Here the class labels corresponds to 1=(ge-,cnv-),2=(ge+,cnv-),3=(ge-,cnv+),4=(ge+,cnv+). We could see that component 3 is missing here after controlling for FDR, and there are only 9 genes in component 3 before we control for FDR. This result is coherent with the model selection result.

4.2 Example 2: Three data types, tranformed z values

IMIX test without specifying the initial values of the parameters

```
test2 = IMIX(
  data_input = data_z,
  data_type = "z",
  p_ini = rep(0.125, 8),
  alpha = 0.05,
```

```
  verbose = TRUE
)
```

```
## Assign initial values!
## number of iterations= 15
## number of iterations= 16
## number of iterations= 15
## Start IMIX-ind procedure!
## iter=1: loglik=-197209.375508519
## iter=2: loglik=-6284.4305584308
## iter=3: loglik=-6281.71982828052
## iter=4: loglik=-6281.71358276159
## iter=5: loglik=-6281.71305003214
## iter=6: loglik=-6281.71292396187
## iter=7: loglik=-6281.71289302385
## iter=8: loglik=-6281.71288528516
## iter=9: loglik=-6281.71288331485
## iter=10: loglik=-6281.71288280508
## Successfully Done!
## Start IMIX-cor-twostep procedure!
## iter=1: loglik=-197439.981700034
## iter=2: loglik=-6282.29517733576
## iter=3: loglik=-6262.10291279091
## iter=4: loglik=-6260.32858091576
## iter=5: loglik=-6260.11382017127
## iter=6: loglik=-6260.07838424502
## iter=7: loglik=-6260.07069520054
## iter=8: loglik=-6260.06873555771
## iter=9: loglik=-6260.06817868917
## iter=10: loglik=-6260.0680085637
## iter=11: loglik=-6260.0679541293
## iter=12: loglik=-6260.06793620325
## iter=13: loglik=-6260.06793019414
## iter=14: loglik=-6260.06792815755
## iter=15: loglik=-6260.06792746257
## Successfully Done!
## Start IMIX-cor model procedure!
## iter=1: loglik=-197439.981700034
## iter=2: loglik=-6282.29517733576
## iter=3: loglik=-6262.15165178343
## iter=4: loglik=-6260.37364580364
## iter=5: loglik=-6260.16164098145
## iter=6: loglik=-6260.12614667984
## iter=7: loglik=-6260.11848025963
## iter=8: loglik=-6260.11656099818
## iter=9: loglik=-6260.11599717453
## iter=10: loglik=-6260.115795872
## iter=11: loglik=-6260.115708172
## iter=12: loglik=-6260.11566366546
## iter=13: loglik=-6260.11563896448
## iter=14: loglik=-6260.11562465458
## iter=15: loglik=-6260.11561621895
## iter=16: loglik=-6260.11561121864
## iter=17: loglik=-6260.11560825296
```

```
## iter=18: loglik=-6260.11560649613
## iter=19: loglik=-6260.11560545711
## iter=20: loglik=-6260.11560484357
## Successfully Done!
## Start IMIX-cor-restrict procedure!
## iter=1: loglik=-197439.981700034
## iter=2: loglik=-6282.29517733576
## iter=3: loglik=-6262.06584905916
## iter=4: loglik=-6260.25535164478
## iter=5: loglik=-6260.03441585679
## iter=6: loglik=-6259.99609270034
## iter=7: loglik=-6259.98722940864
## iter=8: loglik=-6259.98481207683
## iter=9: loglik=-6259.98407148466
## iter=10: loglik=-6259.98382505865
## iter=11: loglik=-6259.9837383004
## iter=12: loglik=-6259.98370658901
## iter=13: loglik=-6259.9836947086
## iter=14: loglik=-6259.98369018425
## iter=15: loglik=-6259.98368844202
## iter=16: loglik=-6259.98368776589
## Successfully Done!
## Start Model Selection!
## Start Adaptive FDR Control!
## All Done!
```

Results of example 2

```
test2$estimatedFDR
```

```
## $estimated_mFDR_comp2
## [1] 0.04712139
##
## $estimated_mFDR_comp3
## [1] 0.04971692
##
## $estimated_mFDR_comp4
## [1] 0.04424941
##
## $estimated_mFDR_comp5
## [1] 0.04514812
##
## $estimated_mFDR_comp6
## [1] 0.04857182
##
## $estimated_mFDR_comp7
## [1] 0.04937603
##
## $estimated_mFDR_comp8
## [1] 0.04952953
```

```
test2$`AIC/BIC`
```

```
##                         AIC      BIC
## IMIX_ind          12601.43 12694.67
```

```
## IMIX_cor_twostep  12630.14 12900.06
## IMIX_cor          12678.23 13065.94
## IMIX_cor_restrict 12641.97 12941.34
```

```
test2$`Selected Model`
```

```
## [1] "IMIX_ind"
```

```
str(test2$IMIX_ind)
```

```
## List of 7
##  $ posterior prob      :'data.frame':    1000 obs. of  8 variables:
##   ..$ component1: num [1:1000] 4.18e-04 7.88e-08 5.66e-08 2.75e-06 8.91e-08 ...
##   ..$ component2: num [1:1000] 6.19e-03 4.08e-15 1.29e-05 1.00 1.32e-03 ...
##   ..$ component3: num [1:1000] 5.70e-02 1.64e-11 3.69e-15 1.07e-13 6.07e-05 ...
##   ..$ component4: num [1:1000] 6.08e-14 1.00 4.18e-03 1.51e-13 1.05e-14 ...
##   ..$ component5: num [1:1000] 9.36e-01 9.40e-19 9.34e-13 4.33e-08 9.99e-01 ...
##   ..$ component6: num [1:1000] 9.41e-13 5.41e-08 9.96e-01 5.74e-08 1.63e-10 ...
##   ..$ component7: num [1:1000] 6.46e-12 1.62e-04 2.12e-10 4.58e-21 5.59e-12 ...
##   ..$ component8: num [1:1000] 1.18e-10 1.03e-11 5.97e-08 2.06e-15 1.02e-07 ...
##  $ Full LogLik all     : num [1:10] -197209 -6284 -6282 -6282 -6282 ...
##  $ Full MaxLogLik final: num -6282
##  $ iterations          : num 10
##  $ pi                  : num [1:8] 0.117 0.113 0.127 0.13 0.137 ...
##  $ mu                  : num [1:6] -0.0422 4.9898 0.0985 5.0021 0.0564 ...
##  $ sigma               : num [1:6] 0.993 1.044 1.041 0.963 1.014 ...
```

```
dim(test2$significant_genes_with_FDRcontrol)
```

```
## [1] 1000    3
```

```
head(test2$significant_genes_with_FDRcontrol)
```

```
##             localFDR class_withoutFDRcontrol class_FDRcontrol
## gene127 1.711382e-09                       8                8
## gene386 4.694697e-07                       8                8
## gene834 9.672581e-07                       8                8
## gene357 1.509129e-06                       8                8
## gene14  1.878799e-06                       8                8
## gene710 2.629312e-06                       8                8
```

For the output includes each model parameter estimations, the model selection AIC and BIC values and the best selected model. The estimated FDR corresponding to the prespecified $\alpha = 0.05$ threshold, the local FDR and class labels for each gene, both without FDR control and based on the FDR control at $\alpha = 0.05$. Here the class labels corresponds to 1=(meth-,ge-,cnv-),2=(meth+,ge-,cnv-),3=(meth-,ge+,cnv-),4=(meth-,ge-,cnv+),5=(meth+,ge+,cnv-),6=(meth+,ge-,cnv+),7=(meth-,ge+,cnv+),8=(meth+,ge+,cnv+).

5 Additional FDR control

We provide two additional FDR control functions for the IMIX test result to achieve

1. Another prespecified α level than the one prespecified using the IMIX function, users do not need to rerun IMIX test for another nominal level for the test result.
2. A combination of multiple components for the IMIX result output, for example, if we want to control the FDR for genes that are associated with the outcome through gene expression, regardless of cnv, then the combination of components would be (ge+,cnv-) and (ge+,cnv+).

5.1 A different α threshold from IMIX() function

Users can always have results controlling for FDR at a different α level than the output in IMIX() function, with no need to rerun the IMIX() test. Below is an R code example for data example 2 in controlling the FDR at $\alpha = 0.2$.

```
fdr_control1 = FDR_control_adaptive_imix(imix_output = test2, alpha = 0.2)
# The input is the result output from IMIX() function
fdr_control1$estimatedFDR
```

```
## $estimated_mFDR_comp2
## [1] 0.1961043
##
## $estimated_mFDR_comp3
## [1] 0.198963
##
## $estimated_mFDR_comp4
## [1] 0.1971467
##
## $estimated_mFDR_comp5
## [1] 0.197843
##
## $estimated_mFDR_comp6
## [1] 0.1966937
##
## $estimated_mFDR_comp7
## [1] 0.1999963
##
## $estimated_mFDR_comp8
## [1] 0.1971296
```

```
head(fdr_control1$significant_genes_with_FDRcontrol)
```

```
##             localFDR class_withoutFDRcontrol class_FDRcontrol
## gene127 1.711382e-09                       8                8
## gene386 4.694697e-07                       8                8
## gene834 9.672581e-07                       8                8
## gene357 1.509129e-06                       8                8
## gene14  1.878799e-06                       8                8
## gene710 2.629312e-06                       8                8
```

```
table(fdr_control1$significant_genes_with_FDRcontrol$class_FDRcontrol)
```

```
##
##   1   2   3   4   5   6   7   8
##  89 100 121 121 144 141 119 165
```

5.2 A combination of multiple components

Users can always have results controlling for FDR for a combination of components, we use data example 1 to illustrate this. Suppose we are interested in the genes that are associated with the outcome through gene expression, regardless of whether it associated with CNV, then we want to control FDR for (ge+,cnv-) and (ge+,cnv+). The corresponding label is class 2 & 4. Below is an R code example for data example 1 in controlling the FDR at $\alpha = 0.2$ for component 2 & component 4.

```
test1$`Selected Model`
```

```
## [1] "IMIX_cor_twostep"
```

```
lfdr_ge_combined = 1 - (test1$IMIX_cor_twostep$`posterior prob`[,2] +
                          test1$IMIX_cor_twostep$`posterior prob`[,4])
# Class 2: (ge+,cnv-); class 4: (ge+,cnv+)
names(lfdr_ge_combined) = rownames(test1$IMIX_cor_twostep$`posterior prob`)
fdr_control2 = FDR_control_adaptive(lfdr = lfdr_ge_combined, alpha = 0.2)
fdr_control2$estimatedFDR
```

```
## [1] 0.1995017
```

```
table(fdr_control2$significant_genes_with_FDRcontrol)
```

```
##
##   0   1
## 307 693
```

The result output fdr_control2 shows that 693 genes (indicator 1) are associated with the outcome through gene expression at FDR $\alpha = 0.2$ for data example 1. The estimated mFDR is 0.1995.

```
sessionInfo()
```

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Sierra 10.12.6
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] zh_CN.UTF-8/zh_CN.UTF-8/zh_CN.UTF-8/C/zh_CN.UTF-8/zh_CN.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] MASS_7.3-51.5 IMIX_0.1.0
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.3        pillar_1.4.3      compiler_3.6.1    mixtools_1.1.0
##  [5] tools_3.6.1       digest_0.6.23     mclust_5.4.5      lattice_0.20-38
```

```
##  [9] evaluate_0.14     tibble_2.1.3      gtable_0.3.0      pkgconfig_2.0.3
## [13] rlang_0.4.4       Matrix_1.2-17     rstudioapi_0.10   yaml_2.2.0
## [17] mvtnorm_1.0-12    xfun_0.11         stringr_1.4.0     dplyr_0.8.3
## [21] knitr_1.26        segmented_1.1-0   grid_3.6.1        tidyselect_1.0.0
## [25] glue_1.3.1        R6_2.4.1          survival_2.44-1.1 rmarkdown_2.0
## [29] ggplot2_3.2.1     purrr_0.3.3       magrittr_1.5      splines_3.6.1
## [33] scales_1.0.0      htmltools_0.3.6   assertthat_0.2.1  colorspace_1.4-1
## [37] labeling_0.3      stringi_1.4.5     lazyeval_0.2.2    munsell_0.5.0
## [41] crayon_1.3.4
```