BLMA: A package for bi-level meta-analysis

Tin Nguyen and Sorin Draghici
Department of Computer Science, Wayne State University, Detroit MI 48202

January 16, 2017

Abstract

This package provides a bi-level meta-analysis (BLMA) framework that can be applied in a wide range of applications: functional analysis, pathway analysis, differential expression analysis, and general hypothesis testing. The framework is able to integrate multiple studies to gain more statistical power, and can be used in conjunction with any statistical hypothesis testing method. It exploits not only the vast number of studies performed in independent laboratories, but also makes better use of the available number of samples within individual studies. In this document, we illustrate the advantage of BLMA over classical meta-analysis. We also provide example code that applies BLMA in all of the areas mentioned above.

Contents

1 Introduction

This document provides an introductory description on how to use the package. For the extended description of the methods, please consult Nguyen et al. [1]. The bi-level meta-analysis (BLMA) framework integrates independent experiments at two levels: an intra-experiment analysis, and an inter-experiment analysis. First, for each experiment, the intra-experiment analysis splits the dataset into smaller datasets, performs a statistical test on each of the newly created small datasets, then combines the p-values. Next, the inter-experiment analysis combines those processed p-values, from each of the individual experiments.

In this package, we implement useful functions that allow users to integrate data in many applications. First, we implement classical methods for combining independent p-values, including Fisher's method [2], Stouffer's method [3]. We also implement our new method named addCLT [1, 4, 5, 6], which is based on the Irwin-Hall distribution [7, 8] and the Central Limit Theorem [9]. These methods of combining p-values (addCLT, Fisher's, Stouffer's, minP, and maxP) are the basic building blocks of the BLMA framework.

Second, we implement functions for BLMA that can be applied in conjunction with classical tests, such as t-test, Wilcoxon test, etc. We provide code and examples that demonstrate the advantage of intra-experiment analysis and bi-level meta-analysis over classical approaches when analyzing stand-alone studies as well as when integrating multiple studies. The functions are flexible and can be applied for one-sample, two-samples, one-tailed, and two-tailed tests. By default, addCLT [1, 4, 5, **?**] is used to combine the p-values, but users can change it to Fisher's method [2], Stouffer's method [3], minP [10], or maxP [11], according to their preference.

Third, we implement functions for functional analysis and pathway analysis. Users can choose to apply the BLMA framework in conjunction with any of the 4 methods: Over-Representation Analysis (ORA) [12, 13], Gene Set Analysis (GSA) [14], Pathway Analysis with Down-weighting of Overlapping Genes (PADOG) [15], and Impact Analyis (IA) [16]. When there is only one dataset, the analysis is reduced to an intra-experiment analysis. The functions are flexible and easy to run.

Fourth, we implement functions for differential expression analysis. The package uses the moderated t-test (limma package [17]) as the test for differential expression. In the intra-experiment analysis, the framework splits a dataset into smaller datasets, performs the moderated t-test on these split datasets, and then combines the results. In the inter-experiment analysis, the framework combines the results obtained from the intra-experiment analysis of individual datasets. The output is a list of genes ranked according to how likely they are to be differentially expressed.

2 BLMA for classical hypothesis testings

Our bi-level meta-analysis framework is comprised of an intra-experiment and an inter-experiment analysis. The reasoning for the intra-experiment is that performing a statistical test on a large experiment is not as powerful as splitting it into smaller studies and then combining them. See Nguyen et al. [1] for a detailed explanation.

We design the function *IntraAnalysisClassic* in a way that it can be used in conjunction with classical tests without any restriction. For example, intead of calling one-sample left-tailed t-test or Wilcoxon test, users can call the function *IntraAnalysisClassic* with the same parameters:

```
> # one-sample tests
> library(BLMA)
> set.seed(1)
> x=rnorm(10, mean = 0)
```

```
> # one-sample left-tailed t-test
> t.test(x, mu=1, alternative = "less")$p.value

[1] 0.003280397

> # one-sample left-tailed intra-experiment analysis with t-test
> IntraAnalysisClassic(x, func=t.test, mu=1, alternative = "less")

[1] 0.003090177

> # one-sample right-tailed t-test
> t.test(x, mu=1, alternative = "greater")$p.value

[1] 0.9967196

> # one-sample right-tailed intra-experiment analysis with t-test
> IntraAnalysisClassic(x, func=t.test, mu=1, alternative = "greater")

[1] 0.9969098

> # one-sample two-tailed t-test
> t.test(x, mu=1)$p.value

[1] 0.006560794

> # one-sample two-tailed intra-experiment analysis with t-test
> IntraAnalysisClassic(x, func=t.test, mu=1)

[1] 0.01236071

> # one-sample left-tailed Wilcoxon test
> wilcox.test(x, mu=1, alternative = "less")$p.value

[1] 0.006835938

> # one-sample left-tailed intra-experiment analysis with Wilcoxon test
> IntraAnalysisClassic(x, func=wilcox.test, mu=1, alternative = "less")

[1] 0.004394531

> # one-sample right-tailed Wilcoxon test
> wilcox.test(x, mu=1, alternative = "greater")$p.value

[1] 0.9951172

> # one-sample right-tailed intra-experiment analysis with Wilcoxon test
> IntraAnalysisClassic(x, func=wilcox.test, mu=1, alternative = "greater")

[1] 0.9995117

> # one-sample two-tailed Wilcoxon test
> wilcox.test(x, mu=1)$p.value
```

```
[1] 0.01367188

> # one-sample two-tailed intra-experiment analysis with Wilcoxon test
> IntraAnalysisClassic(x, func=wilcox.test, mu=1)

[1] 0.01757812
```

Similarly, the intra-experiment analysis can be used with two-sample tests:

```
> # two-sample tests
> set.seed(1)
> x=rnorm(20, mean=0); y=rnorm(20, mean=1)
> # two-sample left-tailed t-test
> t.test(x,y,alternative="less")$p.value

[1] 0.003561452

> # two-sample left-tailed intra-experiment analysis with t-test
> IntraAnalysisClassic(x, y, func=t.test, alternative = "less")

[1] 0.0001387321

> # two-sample right-tailed t-test
> t.test(x,y,alternative="greater")$p.value

[1] 0.9964385

> # two-sample right-tailed intra-experiment analysis with t-test
> IntraAnalysisClassic(x, y, func=t.test, alternative = "greater")

[1] 0.9998613

> # two-sample two-tailed t-test
> t.test(x,y)$p.value

[1] 0.007122904

> # two-sample two-tailed intra-experiment analysis with t-test
> IntraAnalysisClassic(x, y, func=t.test)

[1] 0.002219713

> # two-sample left-tailed Wilcoxon test
> wilcox.test(x,y,alternative="less")$p.value

[1] 0.001942503

> # two-sample left-tailed intra-experiment analysis with Wilcoxon test
> IntraAnalysisClassic(x, y, func=wilcox.test, alternative = "less")

[1] 5.89458e-05
```

```
> # two-sample right-tailed Wilcoxon test
> wilcox.test(x,y,alternative="greater")$p.value

[1] 0.9982331

> # two-sample right-tailed intra-experiment analysis with Wilcoxon test
> IntraAnalysisClassic(x, y, func=wilcox.test, alternative = "greater")

[1] 0.9999669

> # two-sample two-tailed Wilcoxon test
> wilcox.test(x,y)$p.value

[1] 0.003885006

> # two-sample two-tailed intra-experiment analysis with Wilcoxon test
> IntraAnalysisClassic(x, y, func=wilcox.test)

[1] 0.0009431329
```

Some example code for bi-level meta-analysis:

```
> set.seed(1)
> l1 = lapply(as.list(seq(3)),FUN=function (x) rnorm(n=10, mean=1))
> l1

[[1]]
 [1] 0.3735462 1.1836433 0.1643714 2.5952808 1.3295078 0.1795316 1.4874291
 [8] 1.7383247 1.5757814 0.6946116

[[2]]
 [1]  2.5117812  1.3898432  0.3787594 -1.2146999  2.1249309  0.9550664
 [7]  0.9838097  1.9438362  1.8212212  1.5939013

[[3]]
 [1]  1.9189774  1.7821363  1.0745650 -0.9893517  1.6198257  0.9438713
 [7]  0.8442045 -0.4707524  0.5218499  1.4179416

> # one-sample t-test
> lapply(l1, FUN=function(x) t.test(x, alternative="greater")$p.value)

[[1]]
[1] 0.0006575675

[[2]]
[1] 0.002488991

[[3]]
[1] 0.009286192
```

```
> # combining the p-values of one-sample t-test:
> addCLT(unlist(lapply(l1, FUN=function(x)
+     t.test(x, alternative="greater")$p.value)))

[1] 3.202952e-07

> #Bi-level meta-analysis with t-test
> BilevelAnalysisClassic(x=l1, func=t.test, alternative="greater")

[1] 2.765896e-07

> # one-sample Wilcoxon test
> lapply(l1, FUN=function(x) wilcox.test(x, alternative="greater")$p.value)

[[1]]
[1] 0.0009765625

[[2]]
[1] 0.006835938

[[3]]
[1] 0.01367188

> # combining the p-values of one-sample Wilcoxon test:
> addCLT(unlist(lapply(l1, FUN=function(x)
+     wilcox.test(x, alternative="greater")$p.value)))

[1] 1.652787e-06

> #Bi-level meta-analysis with Wilcoxon test
> BilevelAnalysisClassic(x=l1, func=wilcox.test, alternative="greater")

[1] 9.052455e-07
```

3 BLMA for geneset/pathway analysis

For pathway/geneset analysis, the input of the framework is as follows. First, we have multiple studies (datasets) of the same disease. Each dataset consists of a group of control samples and a group of disease samples. Second, we have a list of genesets or pathways from an existing pathway database.

With the current implementation, the meta-analysis can be used in conjunction with the following approaches: Over-Representation Analysis (ORA) [12], Gene Set Analysis (GSA) [14], Pathway Analysis with Down-weighting of Overlapping Genes (PADOG) [15], and Impact Analyis (IA) [16]. By default, we use ORA as the enrichment method, which is very fast and is able to integrate hundreds of samples in a matter of seconds. Other enrichment methods are slower than ORA, and we encourage users to take advantage of our parallel computing by providing the number of processes via the *mc.cores* parameter.

3.1 Over-Representation Analysis (ORA)

We demonstrate this functionality using 6 acute myeloid leukemia (AML) datasets: GSE17054 (9 samples), GSE57194 (12 samples), GSE33223 (30 samples), GSE42140 (31 samples), GSE8023 (12 samples), and GSE14924_CD4 (19 samples). The platform for all datasets is Affymetrix Human Genome U133 Plus 2.0 array. Affymetrix *CEL* files containing raw expression data were downloaded from GEO for each dataset and processed using *R* and *Bioconductor 2.13*. Quality control was performed using the *qc* method from the package *simpleaffy 2.38.0* [18]. Pre-processing was performed on individual datasets using the *threestep* function from the package *affyPLM version 1.38.0* [19, 20, 21]. We calculate the expression value of a gene by taking the median of the probesets that are mapped to the gene. Below is the code for performing BLMA in conjunction with ORA for the 6 datasets:

```
> library(BLMA)
> # load KEGG pathways and create genesets
> x=loadKEGGPathways()
> gslist=lapply(x$kpg,FUN=function(y){return (nodes(y));})
> gs.names=x$kpn[names(gslist)]
> # load the 6 AML datasets
> dataSets=c("GSE14924_CD4", "GSE17054", "GSE57194", "GSE33223",
+            "GSE42140", "GSE8023")
> data(list=dataSets, package="BLMA")
> # prepare dataList and groupList
> dataList <- list()
> groupList <- list()
> for (i in 1:length(dataSets)) {
+    dataset=dataSets[i]
+    group <- get(paste("group_",dataset,sep=""))
+    data=get(paste("data_",dataset,sep=""))
+    dataList[[i]] = data
+    groupList[[i]] = group
+ }
> names(dataList)=names(groupList)=dataSets
> # perform bi-level meta-analysis in conjunction with ORA
> t1=Sys.time()
> ORAComb=BilevelAnalysisGeneset(gslist = gslist, gs.names = gs.names,
+             dataList = dataList, groupList = groupList, enrichment = "ORA")

Working on dataset GSE14924_CD4, 19 samples
Working on dataset GSE17054, 9 samples
Working on dataset GSE57194, 12 samples
Working on dataset GSE33223, 30 samples
Working on dataset GSE42140, 31 samples
Working on dataset GSE8023, 12 samples

> t2=Sys.time();
> # running time
> t2-t1

Time difference of 5.318668 secs
```

```
> #print the results
> options(digits=3)
> ORAComb[1:10, c("Name", "pBLMA", "pBLMA.fdr", "rBLMA")]

                                                 Name    pBLMA pBLMA.fdr rBLMA
path:hsa05221                    Acute myeloid leukemia 5.93e-06  0.000884     1
path:hsa05200                        Pathways in cancer 1.88e-05  0.001401     2
path:hsa05134                               Legionellosis 1.23e-04  0.006109     3
path:hsa05169             Epstein-Barr virus infection 5.89e-04  0.021934     4
path:hsa05202 Transcriptional misregulation in cancer 8.54e-04  0.025459     5
path:hsa04115                      p53 signaling pathway 1.19e-03  0.029569     6
path:hsa05168                 Herpes simplex infection 1.51e-03  0.032226     7
path:hsa04350               TGF-beta signaling pathway 1.99e-03  0.037044     8
path:hsa05322             Systemic lupus erythematosus 3.03e-03  0.050120     9
path:hsa05203                      Viral carcinogenesis 4.24e-03  0.063133    10
```

The running time for ORA is only 5 seconds. With a cutoff of 0.05, there are 8 significant pathways, among which the target pathway *Acute myeloid leukemia* is ranked on top with a FDR-corrected p-value 0.00088.

3.2 Gene Set Analysis (GSA)

We can also perform BLMA in conjunction with GSA. Since the function GSA (from the GSA package) is not as fast as ORA, we recommend users to take advantage of our parallel computing, by setting the number of cores using the *mc.cores* parameter:

```
> # perform bi-level meta-analysis in conjunction with GSA
> t1=Sys.time()
> GSAComb=BilevelAnalysisGeneset(gslist = gslist, gs.names = gs.names,
+             dataList = dataList, groupList = groupList, enrichment = "GSA",
+             mc.cores=2, nperms=200, random.seed = 1)

Working on dataset GSE14924_CD4, 19 samples
Working on dataset GSE17054, 9 samples
Working on dataset GSE57194, 12 samples
Working on dataset GSE33223, 30 samples
Working on dataset GSE42140, 31 samples
Working on dataset GSE8023, 12 samples

> t2=Sys.time();
> # running time
> t2-t1

Time difference of 52.7 secs

> #print the results
> options(digits=3)
> GSAComb[1:10, c("Name", "pBLMA", "pBLMA.fdr", "rBLMA")]
```

```
                                                 Name    pBLMA pBLMA.fdr rBLMA
path:hsa05202 Transcriptional misregulation in cancer 0.000106    0.0157     1
path:hsa05222                   Small cell lung cancer 0.000270    0.0181     2
path:hsa05221                   Acute myeloid leukemia 0.000366    0.0181     3
path:hsa04210                                Apoptosis 0.000682    0.0252     4
path:hsa04012                   ErbB signaling pathway 0.003103    0.0919     5
path:hsa05200                       Pathways in cancer 0.009397    0.2318     6
path:hsa05034                                Alcoholism 0.012128    0.2564     7
path:hsa04064            NF-kappa B signaling pathway 0.017061    0.3156     8
path:hsa05168                Herpes simplex infection 0.021529    0.3220     9
path:hsa05166                      HTLV-I infection 0.021757    0.3220    10
```

The running time of the meta-analysis in conjunction with GSA is approximately 1 minutes with 2 cores. With a cutoff of FDR=0.05, there are 4 significant pathways: *Transcriptional misregulation in cancer*, *Small cell lung cancer*, *Acute myeloid leukemia* and *Apoptosis*. All of them are known to be related to cancer and AML. The target pathway *Acute myeloid leukemia* is ranked 3^{rd} with a FDR-corrected p-value 0.018.

3.3 Pathway Analysis with Down-weighting of Overlapping Genes (PADOG)

Below is an example code for running BLMA in conjunction with PADOG:

```
> set.seed(1)
> t1=Sys.time()
> PADOGComb=BilevelAnalysisGeneset(gslist = gslist, gs.names = gs.names,
+             dataList = dataList, groupList = groupList, enrichment = "PADOG",
+             mc.cores=2, NI=200)

Working on dataset GSE14924_CD4, 19 samples
Working on dataset GSE17054, 9 samples
Working on dataset GSE57194, 12 samples
Working on dataset GSE33223, 30 samples
Working on dataset GSE42140, 31 samples
Working on dataset GSE8023, 12 samples

> t2=Sys.time();
> # running time
> t2-t1

Time difference of 57.8 secs

> #print the results
> options(digits=3)
> PADOGComb[1:10, c("Name", "pBLMA", "pBLMA.fdr", "rBLMA")]

                                                 Name    pBLMA pBLMA.fdr rBLMA
path:hsa04012                   ErbB signaling pathway 0.000278    0.0339     1
path:hsa05200                       Pathways in cancer 0.000542    0.0339     2
path:hsa05202 Transcriptional misregulation in cancer 0.000697    0.0339     3
```

```
path:hsa05221                  Acute myeloid leukemia 0.000910    0.0339     4
path:hsa04310                   Wnt signaling pathway 0.001530    0.0456     5
path:hsa05222                  Small cell lung cancer 0.002587    0.0581     6
path:hsa04390                 Hippo signaling pathway 0.002728    0.0581     7
path:hsa04210                              Apoptosis 0.005101    0.0950     8
path:hsa04660       T cell receptor signaling pathway 0.006871    0.1137     9
path:hsa05210                    Colorectal cancer 0.008735    0.1194    10
```

3.4 Impact Analysis (IA)

Impact Analysis (IA) is a topology-based pathway analysis approach that is able to take into consideration the interaction between genes [16]. Pathway information can be provided in the format of pathway graphs (e.g., graphNEL). Below is an example code for running BLMA in conjunction with IA:

```
> x=loadKEGGPathways()
> t1=Sys.time()
> IAComb=BilevelAnalysisPathway(kpg = x$kpg, kpn = x$kpn, dataList = dataList,
+                               groupList = groupList, mc.cores = 2)

Working on dataset GSE14924_CD4, 19 samples
Working on dataset GSE17054, 9 samples
Working on dataset GSE57194, 12 samples
Working on dataset GSE33223, 30 samples
Working on dataset GSE42140, 31 samples
Working on dataset GSE8023, 12 samples

> t2=Sys.time();
> # running time
> t2-t1

Time difference of 1.1 mins

> #print the results
> options(digits=3)
> IAComb[1:10, c("Name", "pBLMA", "pBLMA.fdr", "rBLMA")]

                                                Name    pBLMA pBLMA.fdr rBLMA
path:hsa05202 Transcriptional misregulation in cancer 3.20e-11  4.70e-09     1
path:hsa05221                  Acute myeloid leukemia 1.66e-06  1.22e-04     2
path:hsa05200                      Pathways in cancer 2.19e-04  1.07e-02     3
path:hsa05203                    Viral carcinogenesis 7.70e-04  2.83e-02     4
path:hsa05416                       Viral myocarditis 1.73e-03  5.08e-02     5
path:hsa04145                               Phagosome 2.11e-03  5.14e-02     6
path:hsa05323                    Rheumatoid arthritis 2.75e-03  5.14e-02     7
path:hsa04350             TGF-beta signaling pathway 2.80e-03  5.14e-02     8
path:hsa05168               Herpes simplex infection 3.35e-03  5.47e-02     9
path:hsa05134                          Legionellosis 4.75e-03  6.99e-02    10
```

4 BLMA for differential expression analysis

The package also provides functions for differential expression analysis across multiple datasets. The input is a set of datasets from the same condition while the output is a list of genes ranked according to their p-values. Here we use the moderated t-test (limma package [17]) as the test for differential expression. As described above, BLMA performs the hypothesis testing at two levels: an intra-experiment analysis and an inter-experiment analysis. At the intra-experiment analysis, BLMA splits a dataset into smaller datasets, performs the moderated t-test for individual genes, and then combines the results obtained from these split datasets. At the inter-experiment analysis, the processed p-values from individual experiments are combined again. By default, the method addCLT is used to combine the p-values, but users can set it to Fisher's, Stouffer's method, minP, or maxP, according to their preference.

4.1 Intra-experiment analysis

The input for intra-experiment analysis is a dataset provided in a data frame. The output consists of the following information: i) logFC: log foldchanges, ii) pLimma: p-values calculated by limma with out intra-experiment analysis, iii) FDR-correct p-values of pLimma, iv) pIntra: p-values obtained from the intra-experiment analysis, and v) FDR-corrected p-values of pIntra. The code for analyzing the dataset GSE14924_CD4 is as follows:

```
> #perform intra-experiment analysis of the dataset GSE14924_CD4 using addCLT
> library(BLMA)
> data(GSE14924_CD4)
> t1=Sys.time()
> X = IntraAnalysisGene(data_GSE14924_CD4, group_GSE14924_CD4)
> Sys.time()-t1

Time difference of 0.54 secs

> X = X[order(X$pIntra), ]
> # top 10 genes
> X[1:10,]

            logFC   pLimma pLimma.fdr   pIntra pIntra.fdr
hsa:867     1.579 6.98e-11   2.87e-07 1.13e-14   3.61e-11
hsa:3725    3.509 9.59e-10   9.87e-07 2.27e-14   3.61e-11
hsa:8341    0.926 7.59e-10   9.87e-07 2.63e-14   3.61e-11
hsa:9218   -1.147 2.31e-09   1.59e-06 1.06e-13   1.09e-10
hsa:2354    3.745 8.40e-09   2.31e-06 2.27e-13   1.85e-10
hsa:51691  -0.987 4.98e-09   1.71e-06 2.70e-13   1.85e-10
hsa:3399   -1.641 6.22e-10   9.87e-07 3.23e-13   1.90e-10
hsa:7095   -0.749 3.90e-09   1.71e-06 3.75e-13   1.93e-10
hsa:348995 -1.290 1.79e-09   1.47e-06 9.23e-13   4.22e-10
hsa:55870  -0.831 4.91e-09   1.71e-06 1.11e-12   4.45e-10

> # bottom 10 genes
> X[(nrow(X)-10):nrow(X),]
```

```
               logFC pLimma pLimma.fdr pIntra pIntra.fdr
hsa:245972 -0.004724  0.928      0.953  0.994      0.996
hsa:9341    0.016701  0.928      0.953  0.994      0.996
hsa:23607   0.001890  0.989      0.993  0.994      0.996
hsa:2161   -0.000950  0.993      0.996  0.995      0.996
hsa:5111   -0.006620  0.958      0.973  0.995      0.997
hsa:9688   -0.005436  0.945      0.966  0.996      0.997
hsa:10274   0.004111  0.962      0.975  0.997      0.998
hsa:747     0.003915  0.963      0.975  0.997      0.998
hsa:6195   -0.008342  0.949      0.967  0.997      0.998
hsa:5733    0.000852  0.989      0.993  0.999      0.999
hsa:9863   -0.001606  0.979      0.985  0.999      0.999
```

```
> #perform intra-experiment analysis of GSE14924_CD4 using Fisher's method
> t1=Sys.time()
> Y = IntraAnalysisGene(data_GSE14924_CD4, group_GSE14924_CD4,
+                       metaMethod=fishersMethod)
> Sys.time()-t1
```

```
Time difference of 0.433 secs
```

```
> Y = Y[order(Y$pIntra), ]
> # top 10 genes
> Y[1:10,]
```

```
            logFC   pLimma pLimma.fdr   pIntra pIntra.fdr
hsa:867     1.579 6.98e-11   2.87e-07 9.56e-15   3.93e-11
hsa:348995 -1.290 1.79e-09   1.47e-06 5.44e-14   1.12e-10
hsa:2354    3.745 8.40e-09   2.31e-06 2.63e-13   2.56e-10
hsa:8341    0.926 7.59e-10   9.87e-07 3.29e-13   2.56e-10
hsa:9218   -1.147 2.31e-09   1.59e-06 3.67e-13   2.56e-10
hsa:3725    3.509 9.59e-10   9.87e-07 3.74e-13   2.56e-10
hsa:7095   -0.749 3.90e-09   1.71e-06 1.37e-12   8.06e-10
hsa:51691  -0.987 4.98e-09   1.71e-06 1.94e-12   9.17e-10
hsa:26224  -0.876 8.42e-09   2.31e-06 2.01e-12   9.17e-10
hsa:3399   -1.641 6.22e-10   9.87e-07 3.08e-12   1.27e-09
```

```
> # bottom 10 genes
> Y[(nrow(Y)-10):nrow(Y),]
```

```
               logFC pLimma pLimma.fdr pIntra pIntra.fdr
hsa:245972 -0.004724  0.928      0.953  0.993      0.996
hsa:9341    0.016701  0.928      0.953  0.994      0.996
hsa:23607   0.001890  0.989      0.993  0.994      0.996
hsa:2161   -0.000950  0.993      0.996  0.995      0.996
hsa:5111   -0.006620  0.958      0.973  0.995      0.997
hsa:9688   -0.005436  0.945      0.966  0.996      0.997
hsa:10274   0.004111  0.962      0.975  0.997      0.998
hsa:747     0.003915  0.963      0.975  0.997      0.998
```

```
hsa:6195   -0.008342  0.949      0.967  0.997      0.998
hsa:5733    0.000852  0.989      0.993  0.999      0.999
hsa:9863   -0.001606  0.979      0.985  0.999      0.999
```

4.2 Bi-level analysis

For bi-level analysis, the input is a list of multiple datasets. The ouput consists of the following information: i) pLimma: combined p-values of limma p-values obtained from individual expriments, ii) pLimma.fdr: FDR-correct p-values of pLimma, iii) pBilevel: combined p-values of pIntra obtained from individual experiments, and iv) pBilevel.fdr: FDR-corrected p-values of pBilevel. We demonstrate the bi-level analysis using the 8 example datasets as follows:

```
> dataSets=c("GSE14924_CD4", "GSE17054", "GSE57194", "GSE33223", "GSE42140",
+            "GSE8023")
> data(list=dataSets, package="BLMA")
> dataList <- list()
> groupList <- list()
> for (i in 1:length(dataSets)) {
+    dataset=dataSets[i]
+    group <- get(paste("group_",dataset,sep=""))
+    data=get(paste("data_",dataset,sep=""))
+    dataList[[i]] = data
+    groupList[[i]] = group
+ }
> names(dataList)=names(groupList)=dataSets
> # running time
> t1=Sys.time()
> Z=BilevelAnalysisGene(dataList = dataList, groupList = groupList)

Working on dataset GSE14924_CD4, 19 samples
Working on dataset GSE17054, 9 samples
Working on dataset GSE57194, 12 samples
Working on dataset GSE33223, 30 samples
Working on dataset GSE42140, 31 samples
Working on dataset GSE8023, 12 samples

> Sys.time()-t1

Time difference of 3.89 secs

> # top 10 genes
> Z[1:10,]

            pLimma pLimma.fdr pBilevel pBilevel.fdr
hsa:55914 3.30e-08   0.000134 3.53e-09     1.43e-05
hsa:55120 1.79e-06   0.001212 2.86e-08     5.81e-05
hsa:23136 3.19e-05   0.004742 8.95e-08     9.10e-05
hsa:7322  1.72e-07   0.000233 8.96e-08     9.10e-05
hsa:54407 1.47e-07   0.000233 1.44e-07     1.17e-04
```

```
hsa:27    3.52e-06   0.001573 2.47e-07     1.53e-04
hsa:92335 4.22e-04   0.012802 2.99e-07     1.53e-04
hsa:3611  8.71e-07   0.000884 3.02e-07     1.53e-04
hsa:10505 1.23e-05   0.002636 7.83e-07     3.52e-04
hsa:3359  4.87e-06   0.001573 9.09e-07     3.52e-04
```

```
> # bottom 10 genes
> Z[(nrow(Z)-10):nrow(Z),]
```

```
           pLimma pLimma.fdr pBilevel pBilevel.fdr
hsa:4685    0.958      0.959    0.969        0.971
hsa:59277   0.975      0.976    0.969        0.971
hsa:3781    0.774      0.794    0.970        0.972
hsa:2977    0.900      0.905    0.972        0.974
hsa:84464   0.953      0.955    0.973        0.974
hsa:440279  0.846      0.858    0.978        0.979
hsa:27258   0.928      0.932    0.979        0.980
hsa:64106   0.826      0.841    0.979        0.980
hsa:7042    0.896      0.903    0.980        0.981
hsa:9723    0.931      0.934    0.996        0.996
hsa:5781    0.994      0.994    0.998        0.998
```

References

[1] T. Nguyen, R. Tagett, M. Donato, C. Mitrea, and S. Drăghici. A novel bi-level meta-analysis approach-applied to biological pathway analysis. *Bioinformatics*, 32(3):409–416, 2016.

[2] R. A. Fisher. *Statistical methods for research workers*. Oliver & Boyd, Edinburgh, 1925.

[3] S. Stouffer, E. Suchman, L. DeVinney, S. Star, and J. Williams, RM. *The American Soldier: Adjustment during army life*, volume 1. Princeton University Press, Princeton, 1949.

[4] T. Nguyen, C. Mitrea, R. Tagett, and S. Drăghici. DANUBE: Data-driven meta-ANalysis using UnBiased Empirical distributions - applied to biological pathway analysis. *Proceedings of the IEEE*, PP(99):1–20, March 2016.

[5] T. Nguyen, D. Diaz, R. Tagett, and S. Draghici. Overcoming the matched-sample bottleneck: an orthogonal approach to integrate omic data. *Nature Scientific Reports*, 6:29251, 2016.

[6] T. Nguyen, D. Diaz, and S. Draghici. TOMAS: A novel TOpology-aware Meta-Analysis approach applied to System biology. In *Proceedings of the 7th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*, pages 13–22. ACM, 2016.

[7] P. Hall. The distribution of means for samples of size n drawn from a population in which the variate takes values between 0 and 1, all such values being equally probable. *Biometrika*, 19(3-4):240–244, 1927.

[8] J. O. Irwin. On the frequency distribution of the means of samples from a population having any law of frequency with finite moments, with special reference to Pearson's Type II. *Biometrika*, 19(3-4):225–239, 1927.

[9] O. Kallenberg. *Foundations of modern probability*. Springer-Verlag, New York, 2002.

[10] L. H. C. Tippett. *The methods of statistics*. Williams & Norgate, London, 1931.

[11] B. Wilkinson. A statistical consideration in psychological research. *Psychological Bulletin*, 48(2):156, 1951.

[12] S. Drăghici, P. Khatri, R. P. Martins, G. C. Ostermeier, and S. A. Krawetz. Global functional profiling of gene expression. *Genomics*, 81(2):98–104, 2003.

[13] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22:281–285, 1999.

[14] B. Efron and R. Tibshirani. On testing the significance of sets of genes. *The Annals of Applied Statistics*, 1(1):107–129, 2007.

[15] A. L. Tarca, S. Drăghici, G. Bhatti, and R. Romero. Down-weighting overlapping genes improves gene set analysis. *BMC Bioinformatics*, 13(1):136, 2012.

[16] S. Drăghici, P. Khatri, A. L. Tarca, K. Amin, A. Done, C. Voichiţa, C. Georgescu, and R. Romero. A systems biology approach for pathway level analysis. *Genome Research*, 17(10):1537–1545, 2007.

[17] G. K. Smyth. Limma: linear models for microarray data. In R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, and W. Huber, editors, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, pages 397–420. Springer, New York, 2005.

[18] C. J. Miller. *simpleaffy: Very simple high level analysis of Affymetrix data*, 2013. R package version 2.38.0.

[19] B. M. Bolstad. *Low-level analysis of high-density oligonucleotide array data: background, normalization and summarization*. PhD thesis, University of California, 2004.

[20] B. M. Bolstad, F. Collin, J. Brettschneider, K. Simpson, L. Cope, R. Irizarry, and T. P. Speed. Quality assessment of Affymetrix GeneChip data. In *Bioinformatics and computational biology solutions using R and Bioconductor*, pages 33–47. Springer, New York, 2005.

[21] J. Brettschneider, F. Collin, B. M. Bolstad, and T. P. Speed. Quality assessment for short oligonucleotide microarray data. *Technometrics*, 50(3), 2008.