Introduction to RBM package

Dongmei Li

October 17, 2016

Clinical and Translational Science Institute, University of Rochester School of Medicine and
Dentistry, Rochester, NY 14642-0708

Contents

1 Overview

This document provides an introduction to the `RBM` package. The `RBM` package executes the resampling-based empirical Bayes approach using either permutation or bootstrap tests based on moderated t-statistics through the following steps.

- Firstly, the RBM package computes the moderated t-statistics based on the observed data set for each feature using the lmFit and eBayes function.
- Secondly, the original data are permuted or bootstrapped in a way that matches the null hypothesis to generate permuted or bootstrapped resamples, and the reference distribution is constructed using the resampled moderated t-statistics calculated from permutation or bootstrap resamples.
- Finally, the p-values from permutation or bootstrap tests are calculated based on the proportion of the permuted or bootstrapped moderated t-statistics that are as extreme as, or more extreme than, the observed moderated t-statistics.

Additional detailed information regarding resampling-based empirical Bayes approach can be found elsewhere (Li et al., 2013).

2 Getting started

The `RBM` package can be installed and loaded through the following R code.
Install the `RBM` package with:

```
> source("http://bioconductor.org/biocLite.R")
> biocLite("RBM")
```

Load the `RBM` package with:

```
> library(RBM)
```

3 `RBM_T` and `RBM_F` functions

There are two functions in the `RBM` package: `RBM_T` and `RBM_F`. Both functions require input data in the matrix format with rows denoting features and columns denoting samples. `RBM_T` is used for two-group comparisons such as study designs with a treatment group and a control group. `RBM_F` can be used for more complex study designs such as more than two groups or time-course studies. Both functions need a vector for group notation, i.e., ”1” denotes the treatment group and ”0” denotes the control group. For the `RBM_F` function, a contrast vector need to be provided by users to perform pairwise comparisons between groups. For example, if the design has three groups (0, 1, 2), the aContrast parameter will be a vector such as (”X1-X0”, ”X2-X1”, ”X2-X0”) to denote all pairwise comparisons. Users just need to add an extra ”X” before the group labels to do the contrasts.

- Examples using the `RBM_T` function: normdata simulates a standardized gene expression data and unifdata simulates a methylation microarray data. The *p*-values from the `RBM_T` function could be further adjusted using the `p.adjust` function in the `stats` package through the Bejamini-Hochberg method.

  ```
  > library(RBM)
  > normdata <- matrix(rnorm(1000*6, 0, 1),1000,6)
  > mydesign <- c(0,0,0,1,1,1)
  > myresult <- RBM_T(normdata,mydesign,100,0.05)
  > summary(myresult)
  ```

  ```
                 Length Class  Mode
  ordfit_t       1000   -none- numeric
  ordfit_pvalue  1000   -none- numeric
  ordfit_beta0   1000   -none- numeric
  ordfit_beta1   1000   -none- numeric
  permutation_p  1000   -none- numeric
  bootstrap_p    1000   -none- numeric
  ```

  ```
  > sum(myresult$permutation_p<=0.05)
  ```

  ```
  [1] 44
  ```

```
> which(myresult$permutation_p<=0.05)

 [1]  26  31  68  92 100 104 116 140 149 166 182 192 194 203 224 226 255 263 278
[20] 300 304 346 359 384 393 439 460 512 537 545 594 627 650 659 698 726 786 825
[39] 844 882 889 905 909 999

> sum(myresult$bootstrap_p<=0.05)

[1] 25

> which(myresult$bootstrap_p<=0.05)

 [1]  26  31  58 113 195 198 235 240 278 300 318 384 400 441 585 613 650 667 695
[20] 731 753 803 889 955 963

> permutation_adjp <- p.adjust(myresult$permutation_p, "BH")
> sum(permutation_adjp<=0.05)

[1] 9

> bootstrap_adjp <- p.adjust(myresult$bootstrap_p, "BH")
> sum(bootstrap_adjp<=0.05)

[1] 0

> unifdata <- matrix(runif(1000*7,0.10, 0.95), 1000, 7)
> mydesign2 <- c(0,0,0, 1,1,1,1)
> myresult2 <- RBM_T(unifdata,mydesign2,100,0.05)
> sum(myresult2$permutatioin_p<=0.05)

[1] 0

> sum(myresult2$bootstrap_p<=0.05)

[1] 23

> which(myresult2$bootstrap_p<=0.05)

 [1] 164 187 207 305 309 311 396 411 419 429 449 480 523 594 668 692 733 828 854
[20] 901 920 936 971

> bootstrap2_adjp <- p.adjust(myresult2$bootstrap_p, "BH")
> sum(bootstrap2_adjp<=0.05)

[1] 0
```

- Examples using the `RBM_F` function: normdata_F simulates a standardized gene expression data and unifdata_F simulates a methylation microarray data. In both examples, we were interested in pairwise comparisons.

```
> normdata_F <- matrix(rnorm(1000*9,0,2), 1000, 9)
> mydesign_F <- c(0, 0, 0, 1, 1, 1, 2, 2, 2)
> aContrast <- c("X1-X0", "X2-X1", "X2-X0")
> myresult_F <- RBM_F(normdata_F, mydesign_F, aContrast, 100, 0.05)
> summary(myresult_F)
```

```
              Length Class  Mode   
ordfit_t      3000   -none- numeric
ordfit_pvalue 3000   -none- numeric
ordfit_beta1  3000   -none- numeric
permutation_p 3000   -none- numeric
bootstrap_p   3000   -none- numeric
```

```
> sum(myresult_F$permutation_p[, 1]<=0.05)
```

```
[1] 59
```

```
> sum(myresult_F$permutation_p[, 2]<=0.05)
```

```
[1] 55
```

```
> sum(myresult_F$permutation_p[, 3]<=0.05)
```

```
[1] 59
```

```
> which(myresult_F$permutation_p[, 1]<=0.05)
```

```
 [1]  10  26  29  35  36  55  64  94 135 188 202 239 245 263 264 275 291 303 315
[20] 332 345 352 361 362 377 379 436 438 455 483 511 516 535 562 568 573 576 613
[39] 633 677 691 705 713 715 720 727 731 752 759 769 787 818 891 903 917 926 940
[58] 976 977
```

```
> which(myresult_F$permutation_p[, 2]<=0.05)
```

```
 [1]  10  26  29  35  55  64  94 135 188 202 245 263 264 275 284 286 298 315 330
[20] 332 345 352 361 362 377 436 438 455 483 511 516 562 568 573 576 613 633 677
[39] 687 711 713 715 720 727 731 764 765 769 787 792 903 917 940 976 977
```

```
> which(myresult_F$permutation_p[, 3]<=0.05)
```

```
 [1]  10  26  29  35  51  55  64  94 135 177 188 202 225 245 257 263 264 275 284
[20] 286 298 303 315 332 352 361 362 377 379 390 436 438 455 483 507 511 516 562
[39] 568 573 576 613 633 713 715 720 727 731 739 759 764 787 792 818 903 917 940
[58] 976 977
```

```
> con1_adjp <- p.adjust(myresult_F$permutation_p[, 1], "BH")
> sum(con1_adjp<=0.05/3)
```

```
[1] 11

> con2_adjp <- p.adjust(myresult_F$permutation_p[, 2], "BH")
> sum(con2_adjp<=0.05/3)

[1] 9

> con3_adjp <- p.adjust(myresult_F$permutation_p[, 3], "BH")
> sum(con3_adjp<=0.05/3)

[1] 15

> which(con2_adjp<=0.05/3)

[1]  10 202 275 362 562 713 903 917 976

> which(con3_adjp<=0.05/3)

 [1]  10  55 245 264 362 438 483 562 573 633 713 715 903 940 976

> unifdata_F <- matrix(runif(1000*18, 0.15, 0.98), 1000, 18)
> mydesign2_F <- c(rep(0, 6), rep(1, 6), rep(2, 6))
> aContrast <- c("X1-X0", "X2-X1", "X2-X0")
> myresult2_F <- RBM_F(unifdata_F, mydesign2_F, aContrast, 100, 0.05)
> summary(myresult2_F)

              Length Class  Mode
ordfit_t      3000   -none- numeric
ordfit_pvalue 3000   -none- numeric
ordfit_beta1  3000   -none- numeric
permutation_p 3000   -none- numeric
bootstrap_p   3000   -none- numeric

> sum(myresult2_F$bootstrap_p[, 1]<=0.05)

[1] 60

> sum(myresult2_F$bootstrap_p[, 2]<=0.05)

[1] 48

> sum(myresult2_F$bootstrap_p[, 3]<=0.05)

[1] 39

> which(myresult2_F$bootstrap_p[, 1]<=0.05)
```

```
 [1]   9  13  16  17  21  40  48 102 184 193 208 218 235 250 278 290 291 296 311
[20] 315 347 348 355 365 368 371 397 400 403 434 448 458 461 518 525 542 545 554
[39] 572 574 612 632 668 711 723 727 741 816 829 838 844 853 874 917 925 932 936
[58] 937 945 963
```

```
> which(myresult2_F$bootstrap_p[, 2]<=0.05)
```

```
 [1]   9  13  16  17  40  70  90 102 200 208 218 250 278 290 296 315 347 348 355
[20] 365 368 371 397 400 403 448 458 461 518 525 542 545 554 572 574 612 632 645
[39] 648 668 711 723 741 838 844 925 939 963
```

```
> which(myresult2_F$bootstrap_p[, 3]<=0.05)
```

```
 [1]  16  17  40  48 102 184 200 208 218 250 252 290 291 296 315 347 355 365 403
[20] 448 458 461 518 542 545 554 561 572 612 632 651 668 711 723 741 829 844 917
[39] 963
```

```
> con21_adjp <- p.adjust(myresult2_F$bootstrap_p[, 1], "BH")
> sum(con21_adjp<=0.05/3)
```

```
[1] 8
```

```
> con22_adjp <- p.adjust(myresult2_F$bootstrap_p[, 2], "BH")
> sum(con22_adjp<=0.05/3)
```

```
[1] 11
```

```
> con23_adjp <- p.adjust(myresult2_F$bootstrap_p[, 3], "BH")
> sum(con23_adjp<=0.05/3)
```

```
[1] 3
```

4 Ovarian cancer methylation example using the `RBM_T` function

Two-group comparisons are the most common contrast in biological and biomedical field. The ovarian cancer methylation example is used to illustrate the application of `RBM_T` in identifying differentially methylated loci. The ovarian cancer methylation example is taken from the gemone-wide DNA methylation profiling of United Kingdom Ovarian Cancer Population Study (UKOPS). This study used Illumina Infinium 27k Human DNA methylation Beadchip v1.2 to obtain DNA methylation profiles on over 27,000 CpGs in whole blood cells from 266 ovarian cancer women and 274 age-matched healthy controls. The data are downloaded from the NCBI GEO website with access number GSE19711. For illutration purpose, we chose the first 1000 loci in 8 randomly selected women with 4 ovariance cancer cases (pre-treatment) and 4 healthy controls. The following codes show the process of generating significant differential DNA methylation loci using the `RBM_T` function and presenting the results for further validation and investigations.

```
> system.file("data", package = "RBM")
```

```
[1] "C:/Users/biocbuild/bbs-3.4-bioc/tmpdir/RtmpwFDNN2/Rinst1cc42fc6b0c/RBM/data"
```

```
> data(ovarian_cancer_methylation)
> summary(ovarian_cancer_methylation)
```

```
        IlmnID          Beta          exmdata2[, 2]     exmdata3[, 2]
 cg00000292:  1   Min.   :0.01058   Min.   :0.01187   Min.   :0.009103
 cg00002426:  1   1st Qu.:0.04111   1st Qu.:0.04407   1st Qu.:0.041543
 cg00003994:  1   Median :0.08284   Median :0.09531   Median :0.087042
 cg00005847:  1   Mean   :0.27397   Mean   :0.28872   Mean   :0.283729
 cg00006414:  1   3rd Qu.:0.52135   3rd Qu.:0.59032   3rd Qu.:0.558575
 cg00007981:  1   Max.   :0.97069   Max.   :0.96937   Max.   :0.970155
 (Other)   :994                     NA's   :4
 exmdata4[, 2]     exmdata5[, 2]     exmdata6[, 2]     exmdata7[, 2]
 Min.   :0.01019   Min.   :0.01108   Min.   :0.01937   Min.   :0.01278
 1st Qu.:0.04092   1st Qu.:0.04059   1st Qu.:0.05060   1st Qu.:0.04260
 Median :0.09042   Median :0.08527   Median :0.09502   Median :0.09362
 Mean   :0.28508   Mean   :0.28482   Mean   :0.27348   Mean   :0.27563
 3rd Qu.:0.57502   3rd Qu.:0.57300   3rd Qu.:0.52099   3rd Qu.:0.52240
 Max.   :0.96658   Max.   :0.97516   Max.   :0.96681   Max.   :0.95974
                   NA's   :1
 exmdata8[, 2]
 Min.   :0.01357
 1st Qu.:0.04387
 Median :0.09282
 Mean   :0.28679
 3rd Qu.:0.57217
 Max.   :0.96268
```

```
> ovarian_cancer_data <- ovarian_cancer_methylation[, -1]
> label <- c(1, 1, 0, 0, 1, 1, 0, 0)
> diff_results <- RBM_T(aData=ovarian_cancer_data, vec_trt=label, repetition=100, alpha=0.05)
> summary(diff_results)
```

```
               Length Class  Mode
ordfit_t       1000   -none- numeric
ordfit_pvalue  1000   -none- numeric
ordfit_beta0   1000   -none- numeric
ordfit_beta1   1000   -none- numeric
permutation_p  1000   -none- numeric
bootstrap_p    1000   -none- numeric
```

```
> sum(diff_results$ordfit_pvalue<=0.05)
```

```
[1] 45
```

```
> sum(diff_results$permutation_p<=0.05)
```

```
[1] 62

> sum(diff_results$bootstrap_p<=0.05)

[1] 54

> ordfit_adjp <- p.adjust(diff_results$ordfit_pvalue, "BH")
> sum(ordfit_adjp<=0.05)

[1] 0

> perm_adjp <- p.adjust(diff_results$permutation_p, "BH")
> sum(perm_adjp<=0.05)

[1] 0

> boot_adjp <- p.adjust(diff_results$bootstrap_p, "BH")
> sum(boot_adjp<=0.05)

[1] 1

> diff_list_perm <- which(perm_adjp<=0.05)
> diff_list_boot <- which(boot_adjp<=0.05)
> sig_results_perm <- cbind(ovarian_cancer_methylation[diff_list_perm, ], diff_results$ordfit_
> print(sig_results_perm)

 [1] IlmnID
 [2] Beta
 [3] exmdata2[, 2]
 [4] exmdata3[, 2]
 [5] exmdata4[, 2]
 [6] exmdata5[, 2]
 [7] exmdata6[, 2]
 [8] exmdata7[, 2]
 [9] exmdata8[, 2]
[10] diff_results$ordfit_t[diff_list_perm]
[11] diff_results$permutation_p[diff_list_perm]
<0 rows> (or 0-length row.names)

> sig_results_boot <- cbind(ovarian_cancer_methylation[diff_list_boot, ], diff_results$ordfit_
> print(sig_results_boot)

        IlmnID       Beta exmdata2[, 2] exmdata3[, 2] exmdata4[, 2]
200 cg00183916 0.03525946    0.03984548    0.02765822    0.02789838
    exmdata5[, 2] exmdata6[, 2] exmdata7[, 2] exmdata8[, 2]
200    0.03034811    0.04302129    0.02753873    0.03067437
    diff_results$ordfit_t[diff_list_boot]
200                              2.272449
    diff_results$bootstrap_p[diff_list_boot]
200                                        0
```