Introduction to RBM package

Dongmei Li

May 19, 2021

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

Contents

1 Overview **1**

2 Getting started **2**

3 `RBM_T` and `RBM_F` functions **2**

4 Ovarian cancer methylation example using the `RBM_T` function **6**

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:

```
> if (!requireNamespace("BiocManager", quietly=TRUE))
+     install.packages("BiocManager")
> BiocManager::install("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] 22

> which(myresult$permutation_p<=0.05)

 [1]  73 110 127 150 250 317 327 354 372 402 407 558 564 663 665 688 695 704 847
[20] 889 947 967

> sum(myresult$bootstrap_p<=0.05)

[1] 2

> which(myresult$bootstrap_p<=0.05)

[1] 104 665

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

[1] 0

> 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] 25

> which(myresult2$bootstrap_p<=0.05)

 [1]  60  61  76 113 219 240 307 332 347 371 415 465 504 554 657 678 710 718 725
[20] 730 765 963 965 976 990

> 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] 69
```

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

```
[1] 63
```

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

```
 [1]  31  34  36  54  62  71 101 139 145 156 169 194 201 212 213 215 251 289 298
[20] 307 325 335 343 355 361 420 450 457 463 473 479 496 504 511 533 544 557 569
[39] 688 711 739 740 749 795 799 820 824 828 838 888 896 914 916 936 939 940 942
[58] 953 970
```

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

```
 [1]  31  36  54  62  71  74  78 102 104 139 145 156 169 194 201 210 212 213 215
[20] 251 279 289 298 314 325 331 335 343 355 361 420 426 457 463 472 473 479 496
[39] 504 511 533 544 557 570 580 638 685 688 711 739 740 743 749 785 795 799 820
[58] 824 828 838 888 896 914 916 936 939 940 942 970
```

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

```
 [1]  31  36  41  54  71  78 101 107 139 145 176 190 194 201 212 213 215 251 289
[20] 298 325 335 343 347 355 361 419 420 426 450 457 463 472 473 479 496 504 511
[39] 533 544 557 580 610 688 711 726 739 740 795 799 820 824 828 838 888 896 914
[58] 916 936 939 940 942 970
```

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

```
[1] 8

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

[1] 12

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

[1] 7

> which(con2_adjp<=0.05/3)

 [1] 139 201 289 420 463 479 496 711 739 896 914 916

> which(con3_adjp<=0.05/3)

[1] 139 145 289 457 463 711 824

> 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] 52

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

[1] 36

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

[1] 55

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

```
 [1]  21  29  46 119 120 128 133 162 175 181 200 219 241 275 292 312 338 352 360
[20] 364 370 399 422 424 429 432 533 538 566 578 660 704 705 717 728 730 754 762
[39] 800 806 830 839 843 866 877 943 951 954 972 980 987 996
```

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

```
 [1]  21  46 118 120 133 175 181 241 275 312 347 352 360 364 399 424 432 533 538
[20] 578 579 660 704 705 717 728 754 806 839 843 877 943 951 976 987 996
```

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

```
 [1]  21  29  46  97 119 120 121 128 162 175 219 236 262 275 292 312 338 352 360
[20] 364 386 399 422 424 432 533 538 542 544 579 593 660 704 705 707 717 723 728
[39] 730 754 762 806 830 839 843 866 877 890 943 945 951 954 976 980 987
```

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

```
[1] 10
```

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

```
[1] 7
```

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

```
[1] 9
```

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] "D:/biocbuild/bbs-3.13-bioc/tmpdir/RtmpCCSnEs/Rinst35cc1ac073c6/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] 55
```

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

[1] 41

> 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] 2

> 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_t
> 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_t
> print(sig_results_boot)

        IlmnID      Beta exmdata2[, 2] exmdata3[, 2] exmdata4[, 2]
146 cg00134539 0.6110132     0.5332178     0.4599934     0.4678742
979 cg00945507 0.1343225     0.2385460     0.3474976     0.2890334
    exmdata5[, 2] exmdata6[, 2] exmdata7[, 2] exmdata8[, 2]
146     0.6719151     0.6313738     0.4792961     0.4542830
979     0.1184851     0.1665385     0.3071842     0.2662474
    diff_results$ordfit_t[diff_list_boot]
```

```
146                                5.394750
979                               -4.750997
    diff_results$bootstrap_p[diff_list_boot]
146                                       0
979                                       0
```