Introduction to RBM package

Dongmei Li

April 26, 2022

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:

```
> 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] 23

> which(myresult$permutation_p<=0.05)

 [1]  69  93  98 107 188 238 264 307 365 404 440 443 617 629 639 746 858 891 918
[20] 960 985 989 991

> sum(myresult$bootstrap_p<=0.05)

[1] 5

> which(myresult$bootstrap_p<=0.05)

[1] 180 461 728 810 877

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

[1] 1

> 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]  24  32  41 143 170 175 207 269 318 325 337 367 434 442 530 601 665 703 784
[20] 824 832 897 898

> 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] 79

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

[1] 56

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

[1] 76

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

 [1]   2  18  73  84 109 112 121 131 140 161 168 175 184 190 196 208 224 225 242
[20] 248 254 270 290 297 308 322 326 331 357 367 377 387 425 430 433 443 472 475
[39] 510 538 541 581 585 590 591 592 596 602 625 658 659 715 724 727 733 751 760
[58] 761 762 765 770 785 786 790 803 834 864 867 879 880 882 883 893 907 909 914
[77] 928 945 990

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

 [1]   2  18  84 112 121 161 165 168 175 184 190 196 208 224 225 242 248 254 270
[20] 290 308 322 326 331 387 425 430 433 475 510 538 581 585 591 596 602 625 658
[39] 724 727 733 751 761 765 770 785 786 790 803 867 880 883 893 914 928 945

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

 [1]   2  18  73  84 112 161 168 175 184 190 205 208 224 225 242 248 254 264 270
[20] 275 290 308 310 321 322 326 331 367 387 425 430 433 443 452 475 490 492 510
[39] 538 541 581 590 591 592 596 599 602 625 653 658 715 724 727 733 760 761 765
[58] 770 785 786 788 790 797 803 834 864 867 882 883 893 907 914 928 932 945 990

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

```
[1] 25

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

[1] 8

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

[1] 22

> which(con2_adjp<=0.05/3)

[1]   2  84 112 208 326 591 785 883

> which(con3_adjp<=0.05/3)

 [1]   2  84 112 168 208 224 308 326 430 510 581 591 602 625 653 724 733 761 765
[20] 785 867 883

> 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] 63

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

[1] 77

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

[1] 71

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

```
 [1]   8  17  22  24  29  30  53  61  83 128 145 151 159 166 167 169 174 177 185
[20] 191 242 252 255 280 295 307 310 349 352 353 364 373 384 404 410 426 429 431
[39] 465 514 550 564 570 600 642 675 708 717 758 778 797 812 829 852 856 895 910
[58] 921 947 954 982 991 999
```

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

```
 [1]   8  17  22  24  29  30  53  61  71  77  83  85  92 102 128 134 141 145 151
[20] 156 159 166 167 169 172 174 177 185 246 252 255 280 295 307 310 314 338 349
[39] 352 364 373 384 394 401 403 404 410 426 429 431 465 514 550 553 570 638 642
[58] 675 708 717 778 785 797 829 851 852 856 877 895 910 921 947 950 954 982 991
[77] 999
```

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

```
 [1]   8  17  22  24  29  30  53  61  83  85  86 128 145 156 159 166 167 169 172
[20] 174 177 242 252 255 280 295 307 314 315 349 352 353 364 373 374 384 401 404
[39] 410 427 429 431 465 514 531 550 553 564 570 642 648 675 708 717 758 778 785
[58] 797 829 852 856 877 895 910 921 950 954 970 982 991 999
```

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

```
[1] 9
```

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

```
[1] 10
```

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

```
[1] 11
```

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] "/private/tmp/RtmpRiBKBo/Rinstf1bc1a2ae96a/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] 64

> sum(diff_results$bootstrap_p<=0.05)

[1] 67

> 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] 3

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

[1] 5

> 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)

        IlmnID      Beta exmdata2[, 2] exmdata3[, 2] exmdata4[, 2]
103 cg00094319 0.7378428     0.7353296     0.7557490     0.7383022
131 cg00121904 0.1544958     0.1794975     0.2360811     0.2435415
851 cg00830029 0.5836250     0.5939787     0.6473961     0.6726964
    exmdata5[, 2] exmdata6[, 2] exmdata7[, 2] exmdata8[, 2]
103     0.6734926     0.7351020     0.7571592     0.7898122
131     0.1735298     0.1256428     0.1819317     0.2084767
851     0.5082024     0.3465747     0.6627657     0.6463451
    diff_results$ordfit_t[diff_list_perm]
103                             -2.268711
131                             -3.451679
851                             -2.841244
    diff_results$permutation_p[diff_list_perm]
103                                          0
131                                          0
851                                          0

> 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    0.53321780     0.4599934    0.46787420
259 cg00234961 0.0419217    0.04321576     0.0570714    0.05327565
280 cg00260778 0.6431989    0.60488960     0.5673506    0.53150910
887 cg00862290 0.4364052    0.54047160     0.6078680    0.56325950
979 cg00945507 0.1343225    0.23854600     0.3474976    0.28903340
    exmdata5[, 2] exmdata6[, 2] exmdata7[, 2] exmdata8[, 2]
146    0.67191510    0.63137380    0.47929610    0.45428300
259    0.04030003    0.03996053    0.05086962    0.05445672
280    0.61920530    0.61925200    0.46753250    0.55632410
887    0.50259740    0.40111730    0.56646700    0.54552980
979    0.11848510    0.16653850    0.30718420    0.26624740
    diff_results$ordfit_t[diff_list_boot]
146                              5.394750
259                             -4.052697
280                              4.170347
887                             -3.217939
979                             -4.750997
    diff_results$bootstrap_p[diff_list_boot]
146                                       0
259                                       0
280                                       0
887                                       0
979                                       0
```