Introduction to RBM package

Dongmei Li

May 1, 2024

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]  38  51  62  66 199 234 259 290 301 326 418 436 463 470 485 493 532 601 633
[20] 641 674 770 832

> sum(myresult$bootstrap_p<=0.05)

[1] 20

> which(myresult$bootstrap_p<=0.05)

 [1]  44  51  62  67 108 127 156 177 202 228 334 379 409 413 695 706 832 863 927
[20] 944

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

[1] 5

> 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] 13

> which(myresult2$bootstrap_p<=0.05)

 [1]  41  88 143 162 180 321 375 510 535 541 578 662 950

> 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] 61
```

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

```
[1] 63
```

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

```
[1] 60
```

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

```
 [1]  40  44  67  93 134 135 139 149 155 169 171 178 188 206 210 211 251 285 298
[20] 303 315 335 359 399 426 429 434 459 472 514 535 540 548 557 594 602 606 608
[39] 630 648 657 674 685 700 713 717 742 744 786 790 810 813 828 846 864 900 924
[58] 927 946 953 970
```

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

```
 [1]  43  44  48  63  87  93 114 129 134 135 139 149 155 169 188 203 206 210 211
[20] 220 303 315 335 360 368 399 407 426 429 459 471 514 535 540 548 569 594 602
[39] 606 626 648 674 685 695 700 713 742 744 786 789 790 802 808 810 828 843 869
[58] 900 927 933 946 953 970
```

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

```
 [1]  44  67  87  90  93 129 134 135 139 149 155 171 178 188 206 210 211 263 295
[20] 303 315 335 399 407 426 429 459 493 540 548 557 569 594 602 626 648 674 685
[39] 700 713 717 730 742 744 786 790 802 810 813 828 846 864 869 876 900 920 927
[58] 946 953 970
```

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

```
[1] 9

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

[1] 10

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

[1] 13

> which(con2_adjp<=0.05/3)

 [1] 135 139 149 206 303 399 429 685 900 953

> which(con3_adjp<=0.05/3)

 [1]  44 135 139 155 206 399 429 459 548 713 786 790 900

> 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] 67

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

[1] 57

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

[1] 63

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

```
 [1]  34  45  57  92 124 133 136 140 141 151 153 161 171 179 220 253 259 299 312
[20] 315 330 341 358 378 382 436 452 480 481 490 492 500 514 547 555 562 587 588
[39] 590 593 600 620 645 661 669 698 727 734 737 742 749 759 782 819 839 840 846
[58] 892 904 926 937 941 959 980 987 990 994
```

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

```
 [1]   1  57  92  97 124 133 136 140 141 153 171 179 220 253 299 306 312 330 358
[20] 378 420 436 452 480 500 514 547 562 569 587 588 590 593 600 613 620 645 661
[39] 669 705 711 727 737 742 749 782 786 819 839 846 852 892 903 937 980 987 990
```

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

```
 [1]  34  45  57  92 124 136 140 141 151 153 171 179 215 220 253 259 299 306 312
[20] 330 341 358 378 382 420 436 452 480 492 500 514 547 555 562 569 587 590 593
[39] 600 613 620 645 661 669 698 711 727 734 742 749 782 786 819 839 846 852 892
[58] 903 937 959 980 990 994
```

```
> 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] 5
```

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

```
[1] 6
```

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/RtmppSFJM2/Rinst16d4b6676caaa/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] 63

> sum(diff_results$bootstrap_p<=0.05)

[1] 35

> 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] 6

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

[1] 0

> 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]
97  cg00083937 0.53046980    0.60529020    0.62733150    0.65623920
103 cg00094319 0.73784280    0.73532960    0.75574900    0.73830220
131 cg00121904 0.15449580    0.17949750    0.23608110    0.24354150
772 cg00743372 0.03922780    0.02919634    0.02187972    0.02568053
848 cg00826384 0.05721674    0.05612171    0.06644259    0.06358381
851 cg00830029 0.58362500    0.59397870    0.64739610    0.67269640
    exmdata5[, 2] exmdata6[, 2] exmdata7[, 2] exmdata8[, 2]
97     0.55974270    0.43157020    0.64046990    0.57876990
103    0.67349260    0.73510200    0.75715920    0.78981220
131    0.17352980    0.12564280    0.18193170    0.20847670
772    0.02796053    0.03512214    0.02575992    0.02093909
848    0.05230160    0.06119713    0.06542751    0.06240686
851    0.50820240    0.34657470    0.66276570    0.64634510
    diff_results$ordfit_t[diff_list_perm]
97                              -2.541586
103                             -2.268711
131                             -3.451679
772                              2.416991
```

```
848                           -2.314412
851                           -2.841244
    diff_results$permutation_p[diff_list_perm]
97                                        0
103                                       0
131                                       0
772                                       0
848                                       0
851                                       0

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

 [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$ordfit_t[diff_list_boot]
[11] diff_results$bootstrap_p[diff_list_boot]
<0 rows> (or 0-length row.names)
```