

Introduction to RBM package

Dongmei Li

April 22, 2025

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 Benjamini-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] 8

> which(myresult$permutation_p<=0.05)

[1] 217 418 521 526 669 722 794 883

> sum(myresult$bootstrap_p<=0.05)

[1] 15

> which(myresult$bootstrap_p<=0.05)

[1] 174 244 335 339 401 459 521 629 702 703 716 721 790 901 927

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

[1] 1

> bootstrap_adj_p <- p.adjust(myresult$bootstrap_p, "BH")
> sum(bootstrap_adj_p<=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] 30

> which(myresult2$bootstrap_p<=0.05)

[1] 15 27 88 116 130 263 265 290 291 356 368 380 402 404 414 436 503 528 547
[20] 553 574 617 684 732 735 777 811 855 947 994

> bootstrap2_adj_p <- p.adjust(myresult2$bootstrap_p, "BH")
> sum(bootstrap2_adj_p<=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] 62

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

[1] 69

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

[1] 48

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

[1] 15 16 22 36 47 49 69 88 93 118 123 138 157 164 188 207 242 258 265
[20] 301 309 314 331 341 354 363 373 384 412 432 445 455 457 458 463 472 475 482
[39] 488 528 541 543 605 612 613 627 636 696 733 739 774 777 784 794 799 824 853
[58] 863 892 900 902 994

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

[1] 15 16 22 36 47 49 69 89 93 100 118 123 138 157 164 188 207 242 258
[20] 265 300 301 309 329 333 341 346 363 373 384 412 432 445 455 457 458 463 464
[39] 475 482 488 528 535 540 541 561 605 612 613 627 636 655 691 733 739 768 774
[58] 784 794 799 824 836 842 853 862 881 894 902 994

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

[1] 15 16 22 36 47 69 118 123 188 207 242 258 265 300 301 309 341 354 363
[20] 384 412 432 445 455 457 458 463 528 541 605 612 613 627 636 655 696 733 739
[39] 768 784 799 842 853 863 892 894 900 994

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

```

```

[1] 5

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

[1] 19

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

[1] 8

> which(con2_adjp<=0.05/3)

[1] 15 47 69 118 207 265 341 363 412 432 455 458 463 528 613 627 739 799 994

> which(con3_adjp<=0.05/3)

[1] 69 207 341 412 432 613 739 994

> 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] 47

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

[1] 75

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

[1] 51

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

```

```

[1] 11 64 69 92 94 112 113 117 127 134 175 218 220 231 242 247 260 312 375
[20] 405 417 530 588 597 617 664 667 676 727 733 747 760 779 804 814 835 840 853
[39] 885 893 915 929 933 950 968 988 992

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

[1] 11 18 25 30 42 62 69 97 104 112 113 116 117 127 134 142 175 218 220
[20] 231 236 239 242 247 260 262 299 312 344 356 375 377 379 402 405 417 506 530
[39] 533 544 587 588 594 597 617 647 664 667 676 699 733 747 748 760 779 793 804
[58] 814 818 825 835 837 840 871 885 893 901 915 929 933 950 968 988 992 994

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

[1] 11 42 62 94 97 112 113 116 117 127 134 175 218 220 231 237 242 247 260
[20] 262 299 305 310 312 377 530 533 544 587 588 597 617 667 676 733 747 793 799
[39] 804 814 818 835 853 885 893 901 929 966 968 988 992

> con21_adj_p <- p.adjust(myresult2_F$bootstrap_p[, 1], "BH")
> sum(con21_adj_p<=0.05/3)

[1] 0

> con22_adj_p <- p.adjust(myresult2_F$bootstrap_p[, 2], "BH")
> sum(con22_adj_p<=0.05/3)

[1] 7

> con23_adj_p <- p.adjust(myresult2_F$bootstrap_p[, 3], "BH")
> sum(con23_adj_p<=0.05/3)

[1] 2

```

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 genome-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 illustration purpose, we chose the first 1000 loci in 8 randomly selected women with 4 ovarian 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/var/folders/db/4tvngx8jx4z3fm1gzlnlzw9rc0000gq/T/Rtmpk8F0am/Rinst2ed01b8ad8fc/RBM/d
```

```
> 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.:0.59031	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] 47
```

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

```

[1] 46

> sum(diff_results$bootstrap_p<=0.05)

[1] 55

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

[1] 0

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

[1] 0

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

[1] 0

> diff_list_perm <- which(perm_adj_p<=0.05)
> diff_list_boot <- which(boot_adj_p<=0.05)
> sig_results_perm <- cbind(ovarian_cancer_methylation[diff_list_perm, ], diff_results$ordfit_t[diff_list_perm, ])
> 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[diff_list_boot, ])
> 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)
```