Differential analyses with DSS

Hao Wu

Department of Biostatistics and Bioinformatics
Emory University
Atlanta, GA 303022

hao.wu@emory.edu

October 14, 2015

Contents

Abstract

This vignette introduces the use of the Bioconductor package DSS (Dispersion Shrinkage for Sequencing data), which is designed for differential analysis based on high-throughput sequencing data. It performs differential expression analyses for RNA-seq, and differential methylation analyses for bisulfite sequencing (BS-seq) data. The core of DSS is a new procedure to estimate and shrink gene- or CpG site-specific dispersions, then conduct Wald tests for differential expression/methylation. Compared with existing methods, DSS provides excellent statistical and computational performance.

1 Introduction

Recent advances in high-throughput sequencing technology have revolutionized genomic research. For example, RNA-seq is a new technology for measuring the abundance of RNA products. Compared to gene expression microarrays, it provides a better dynamic range and lower signal-to-noise ratio. Bisulfite sequencing (BS-seq) is a new technology for measuring DNA methylation. Compared to capture-based methods such as MeDIP-seq, it provides single-base resolution and eliminates biases associated with CpG density.

Fundamental questions for RNA-seq or BS-seq data analyses are whether gene expression regulation or DNA methylation dynamics vary under different biological contexts. Identifying sites or regions exhibiting differential expression (DE) or differential methylation (DM) are thus key tasks in functional genomics research.

RNA- or BS-seq experiments typically have a limited number of biological replicates due to cost constraints. This can lead to unstable estimation of within group variance, and subsequently undesirable results from hypothesis testing. Variance shrinkage methods have been widely used in DE analyses based on microarray data. The methods are typically based on a Bayesian hierarchical model, with a prior imposed on the gene-specific variances to provide a basis for information sharing across all genes/CpG sites. In these models, shrinkage is achieved for variance estimation. Using shrunk variance in hypothesis tests has been shown to provide better results.

A distinct feature of RNA-seq or BS-seq data is that the measurements are in the form of counts. These data are often assumed to be from the Poisson (for RNA-seq) or Binomial (for BS-seq) distributions. Unlike continuous distributions such as the Gaussian distribution, the variances depend on means in these discrete distributions. This implies that the sample variances do not account for biological variation between replicates, and shrinkage cannot be applied on variances directly.

In contrast, we assume that our count data come from the Gamma-Poisson (RNA-seq) or Beta-Binomial (BS-seq) distribution. These distributions can be parameterized by a mean and an over dispersion parameter. The over dispersion parameters, which represent the biological variation for replicates within a treatment group, play a central role in the differential analyses.

Here we present a new DE/DM detection algorithm, where shrinkage is performed on the dispersion parameters. We first impose a log-normal prior on the dispersions, and then combine data from all genes/CpG sites to shrink dispersions through a penalized likelihood approach. Finally, we construct Wald tests to test each gene/site for differential expression/methylation. Our results show that the new method provides excellent performance compared to existing methods, especially when the overall dispersion level is high or the number of replicates is small.

For details of the hierarchical model, the shrinkage method and test procedure, please read [3] for differential expression from RNA-seq, and [1] for differential methylation from BS-seq.

2 Using DSS for differential expression analysis

2.1 Single factor experiment

Required inputs for DSS are (1) gene expression values as a matrix of integers, rows are for genes and columns are for samples; and (2) a vector representing experimental designs. The length of the design vector must match the number of

columns of input counts. Optionally, normalization factors or additional annotation for genes can be supplied.

The basic data container in the package is `SeqCountSet` class, which is directly inherited from `ExpressionSet` class defined in `Biobase`. An object of the class contains all necessary information for a DE analysis: gene expression values, experimental designs, and additional annotations.

A typical DE analysis contains the following simple steps.

1. Create a `SeqCountSet` object using `newSeqCountSet`.
2. Estimate normalization factor using `estNormFactors`.
3. Estimate and shrink gene-wise dispersion using `estDispersion`
4. Two-group comparison using `waldTest`.

The usage of DSS is demonstrated in the simple simulation below.

1. First load in the library, and make a `SeqCountSet` object from some counts for 2000 genes and 6 samples.

```
> library(DSS)
> counts1=matrix(rnbinom(300, mu=10, size=10), ncol=3)
> counts2=matrix(rnbinom(300, mu=50, size=10), ncol=3)
> X1=cbind(counts1, counts2) ## these are 100 DE genes
> X2=matrix(rnbinom(11400, mu=10, size=10), ncol=6)
> X=rbind(X1,X2)
> designs=c(0,0,0,1,1,1)
> seqData=newSeqCountSet(X, designs)
> seqData
SeqCountSet (storageMode: lockedEnvironment)
assayData: 2000 features, 6 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: 1 2 ... 6 (6 total)
  varLabels: designs
  varMetadata: labelDescription
featureData: none
experimentData: use 'experimentData(object)'
Annotation:
```

2. Estimate normalization factor.

```
> seqData=estNormFactors(seqData)
```

3. Estimate and shrink gene-wise dispersions

```
> seqData=estDispersion(seqData)
```

4. With the normalization factors and dispersions ready, the two-group comparison can be conducted via a Wald test:

```
> result=waldTest(seqData, 0, 1)
> head(result,5)
   geneIndex       muA      muB       lfc   difExpr     stats         pval    local.fdr
76        76  7.666667 71.91111 -2.182299 -64.24444 -5.708904 1.137061e-08 0.0001326277
```

```
55        55  7.666667 62.42222 -2.041839 -54.75556 -5.499369 3.811532e-08 0.0001957644
21        21  5.666667 52.40000 -2.149245 -46.73333 -5.463938 4.656859e-08 0.0002114527
64        64  8.333333 63.42222 -1.979135 -55.08889 -5.448348 5.083991e-08 0.0002186075
86        86 11.000000 74.31111 -1.872619 -63.31111 -5.425914 5.765869e-08 0.0002303498
             fdr
76 0.0001326277
55 0.0001604846
21 0.0001779112
64 0.0001779112
86 0.0001779112
```

A higher level wrapper function `DSS.DE` is provided for simple RNA-seq DE analysis in a two-group comparison. User only needs to provide a count matrix and a vector of 0's and 1's representing the design, and get DE test results in one line. A simple example is listed below:

```
> counts = matrix(rpois(600, 10), ncol=6)
> designs = c(0,0,0,1,1,1)
> result = DSS.DE(counts, designs)
> head(result)
```

```
   geneIndex       muA       muB        lfc   difExpr     stats       pval local.fdr
78        78  8.966760 15.666667 -0.5351648 -6.699906 -1.873593 0.06098659 0.8148400
80        80  7.435969 12.666667 -0.5062829 -5.230697 -1.718522 0.08570154 0.7522293
16        16  7.304981 12.000000 -0.4709665 -4.695019 -1.539722 0.12362806 0.6968193
1          1 12.263927  7.666667  0.4465622  4.597260  1.503098 0.13281379 0.5871887
31        31  5.910067  9.666667 -0.4612447 -3.756600 -1.443946 0.14875410 0.6988676
14        14  6.411530 10.666667 -0.4797421 -4.255137 -1.403769 0.16038772 0.7094367
         fdr
78 0.8148400
80 0.7316221
16 0.7048760
1  0.7483291
31 0.7094317
14 0.7130576
```

2.2 Multifactor experiment

`DSS` provides functionalities for dispersion shrinkage for multifactor experimental designs. Downstream model fitting (through genearlized linear model) and hypothesis testing can be performed using other packages such as `edgeR`, with the dispersions estimated from DSS.

Below is an example, based a simple simulation, to illustrate the DE analysis of a crossed design.

1. First simulate data for a 2x2 crossed experiments. Note the counts are randomly generated.

```
> library(DSS)
> library(edgeR)
> counts=matrix(rpois(800, 10), ncol=8)
> design=data.frame(gender=c(rep("M",4), rep("F",4)), strain=rep(c("WT", "Mutant"),4))
> X=model.matrix(~gender+strain, data=design)
```

2. make SeqCountSet, then estimate size factors and dispersion

```
> seqData=newSeqCountSet(counts, as.data.frame(X))
> seqData=estNormFactors(seqData)
> seqData=estDispersion(seqData)
```

3. Using edgeR's function to do glm model fitting, but plugging in the estimated size factors and dispersion from DSS.

```
> fit.edgeR <- glmFit(counts, X, lib.size=normalizationFactor(seqData),
+               dispersion=dispersion(seqData))
```

4. Using edgeR's function to do hypothesis testing on the second parameter of the model (gender).

```
> lrt.edgeR <- glmLRT(glmfit=fit.edgeR, coef=2)
> head(lrt.edgeR$table)
         logFC   logCPM           LR    PValue
1  0.191996051 21.21143 0.2716968479 0.6021958
2  0.263443643 21.20925 0.5195711969 0.4710247
3 -0.007767414 21.11911 0.0004315119 0.9834268
4  0.271202296 21.14903 0.5348258777 0.4645850
5 -0.075900839 21.08641 0.0389720339 0.8435041
6 -0.243065760 21.25398 0.4453533097 0.5045499
```

3 Using DSS for differential methylation analysis

3.1 Overview

To detect differential methylation, statistical tests are conducted at each CpG site, and then the differential methylation loci (DML) or differential methylation regions (DMR) are called based on user specified threshold. A rigorous statistical tests should account for biological variations among replicates and the coverage depth. Most existing methods for DM analysis are based on *ad hoc* methods. For example, using Fisher's exact ignores the biological variations, using t-test on estimated methylation levels ignores the coverage depth. Sometimes arbitrary filtering are recommended: loci with coverages lower than an arbitrary threshold are filtered out.

The DM detection procedure implemented in DSS is based on a rigorous Wald test for beta-binomial distributions. The test statistics depend on the biological variations (captured by dispersion parameter) as well as the coverage depth. An important part of the algorithm is the estimation of dispersion parameter, which is achieved through a shrinkage estimator based on a Bayesian hierarchical model [1].

A great advantage of DSS is that the test can be performed even when there is no biological replicates. That's because by smoothing procedure, the neighboring CpG sites can be viewed as "pseudo-replicates", and the dispersion can still be

estimated with reasonable precision.

This package depends on bsseq Bioconductor package, which has neat definition of data structures and many useful utility functions. In order to use the DM detection functionalities, bsseq needs to be pre-installed.

3.2 Input data

For one BS-seq experiments, after sequence alignment and some processing, the BS-seq data are usually summarized into following information for each CG position: chromosome number, genomic coordinate, total number of reads, and number of reads showing methylation. For a sample, this information are saved in a simple text file, with each row representing a CpG site. Below shows an example of a small part of such a file:

```
chr     pos     N       X
chr18   3014904 26      2
chr18   3031032 33      12
chr18   3031044 33      13
chr18   3031065 48      24
chr18   3031069 17      4
chr18   3031082 93      37
chr18   3031089 76      25
chr18   3031092 76      28
```

DML/DMR detection using DSS starts from several such text files. A typical DML detection contains two simple steps. First one conduct DM test at each CpG site, then DML/DMR are called based on the test result and user specified threshold. We will use example data distributed with DSS to illustrate the procedure.

3.3 DML/DMR detection from two-group comparison

Below are the steps to call DML or DMR for BS-seq data in two-group comparison setting.

1. Load in library. Read in text files and create an object of BSseq class, which is defined in bsseq Bioconductor package. This step requires bsseq Bioconductor package. BSseq class is defined in that package.

```
> library(DSS)
> require(bsseq)
> path <- file.path(system.file(package="DSS"), "extdata")
> dat1.1 <- read.table(file.path(path, "cond1_1.txt"), header=TRUE)
> dat1.2 <- read.table(file.path(path, "cond1_2.txt"), header=TRUE)
> dat2.1 <- read.table(file.path(path, "cond2_1.txt"), header=TRUE)
> dat2.2 <- read.table(file.path(path, "cond2_2.txt"), header=TRUE)
> BSobj <- makeBSseqData( list(dat1.1, dat1.2, dat2.1, dat2.2),
+      c("C1","C2", "N1", "N2") )[1:10000,]
> BSobj
An object of type 'BSseq' with
  10000 methylation loci
```

```
  4 samples
has not been smoothed
```

2. Perform statistical test for DML by calling `DMLtest` function. This function basically performs following steps: (1) estimate mean methylation levels for all CpG site; (2) estimate dispersions at each CpG sites; (3) conduct Wald test. For the first step, there's an option for smoothing or not. Because the methylation levels show strong spatial correlations, smoothing can help obtain better estimates of mean methylation when the CpG sites are dense in the data (such as from the whole-genome BS-seq). However for data with sparse CpG, such as from RRBS or hydroxyl-methylation, smoothing is not recommended.

   To perform DML test without smoothing, do:

```
> dmlTest <- DMLtest(BSobj, group1=c("C1", "C2"), group2=c("N1", "N2"))
Estimating dispersion for each CpG site, this will take a while ...
> head(dmlTest)
    chr     pos       mu1       mu2        diff    diff.se       stat        phi1
1 chr18 3014904 0.3850276 0.4623677 -0.07734011 0.24899738 -0.3106061 0.286610813
2 chr18 3031032 0.3384423 0.1416667  0.19677555 0.11402507  1.7257217 0.009525212
3 chr18 3031044 0.3436302 0.3299846  0.01364560 0.12508174  0.1090935 0.010959806
4 chr18 3031065 0.4372540 0.3646882  0.07256585 0.10435917  0.6953471 0.010846613
5 chr18 3031069 0.2939132 0.5397749 -0.24586172 0.13554689 -1.8138499 0.012965376
6 chr18 3031082 0.3529066 0.3903499 -0.03744333 0.08197055 -0.4567900 0.008246131
        phi2       pval       fdr
1 0.01964971 0.75610007 0.9999077
2 0.05282909 0.08439748 0.7175525
3 0.02285714 0.91312835 0.9999077
4 0.01849672 0.48683778 0.9999077
5 0.02658296 0.06970083 0.6313442
6 0.01338446 0.64782202 0.9999077
```

   To perform statistical test for DML with smoothing, do:

```
> dmlTest.sm <- DMLtest(BSobj, group1=c("C1", "C2"), group2=c("N1", "N2"), smoothing=TRUE)
Smoothing ...
Estimating dispersion for each CpG site, this will take a while ...
```

   There are two options for smoothing: a simple moving average, or the BSmooth method implemented in `bsseq` package. The BSmooth method produces much smoother curve, which is good for visualization purpose. However, it is very computationally intensive, and the results are not very different from moving average in terms of DMR calling. So we recommend using moving average. Smoothing span is an important parameter in smoothing procedure and have non-trivial impact on DMR calling. We use 500 bp as default, and think that it performs well in real data tests.

3. With the test results, one can call DML by using `callDML` function. The results DMLs are sorted by the significance.

```
>   dmls <- callDML(dmlTest, p.threshold=0.001)
>   head(dmls)
      chr     pos        mu1        mu2       diff    diff.se       stat       phi1
450 chr18 3976129 0.01048590 0.93773387 -0.9272480 0.06785915 -13.664303 0.04319795
451 chr18 3976138 0.01048590 0.93773387 -0.9272480 0.06785915 -13.664303 0.04319795
```

```
582 chr18 4340682 0.96365233 0.03162952  0.9320228 0.09947722   9.369209 0.06181184
583 chr18 4340709 0.96365233 0.03162952  0.9320228 0.09947722   9.369209 0.06181184
638 chr18 4431501 0.01325735 0.94195478 -0.9286974 0.09390549  -9.889704 0.04380187
639 chr18 4431511 0.01321013 0.94195478 -0.9287446 0.09387324  -9.893604 0.04377619
          phi2 pval fdr postprob.overThreshold
450 0.02756948    0   0                     1
451 0.02756948    0   0                     1
582 0.05147878    0   0                     1
583 0.05147878    0   0                     1
638 0.08184949    0   0                     1
639 0.08184949    0   0                     1
```

By default, the test is based on the null hypothesis that the difference in methylation levels is 0. Alternatively, users can specify a threshold for difference. For example, to detect loci with difference greater than 0.1, do:

```
> dmls2 <- callDML(dmlTest, delta=0.1, p.threshold=0.001)
> head(dmls2)
      chr     pos        mu1        mu2       diff    diff.se       stat       phi1
450 chr18 3976129 0.01048590 0.93773387 -0.9272480 0.06785915 -13.664303 0.04319795
451 chr18 3976138 0.01048590 0.93773387 -0.9272480 0.06785915 -13.664303 0.04319795
582 chr18 4340682 0.96365233 0.03162952  0.9320228 0.09947722   9.369209 0.06181184
583 chr18 4340709 0.96365233 0.03162952  0.9320228 0.09947722   9.369209 0.06181184
638 chr18 4431501 0.01325735 0.94195478 -0.9286974 0.09390549  -9.889704 0.04380187
639 chr18 4431511 0.01321013 0.94195478 -0.9287446 0.09387324  -9.893604 0.04377619
          phi2 pval fdr postprob.overThreshold
450 0.02756948    0   0                     1
451 0.02756948    0   0                     1
582 0.05147878    0   0                     1
583 0.05147878    0   0                     1
638 0.08184949    0   0                     1
639 0.08184949    0   0                     1
```

When delta is specified, the function will compute the posterior probability that the difference of the means is greater than delta. So technically speaking, the threshold for p-value here actually refers to the threshold for 1-posterior probability, or the local FDR. Here we use the same parameter name for the sake of the consistence of function syntax.

4. DMR detection is also Based on the DML test results, by calling `callDMR` function. Regions with many statistically significant CpG sites are identified as DMRs. Some restrictions are provided by users, including the minimum length, minimum number of CpG sites, percentage of CpG site being significant in the region, etc. There are some *post hoc* procedures to merge nearby DMRs into longer ones.

```
> dmrs <- callDMR(dmlTest, p.threshold=0.01)
> head(dmrs)
      chr    start      end length nCG meanMethy1 meanMethy2 diff.Methy  areaStat
186 chr18 13131705 13131880    176   6  0.9689797 0.06107942  0.9079003 68.741081
33  chr18  4657576  4657639     64   4  0.5084704 0.31870495  0.1897655 14.421382
```

```
38  chr18  5027578  5027743    166   4  0.7045195 0.38058436  0.3239352  9.109377
```

Here the DMRs are sorted by "areaStat", which is defined in `bsseq` as the sum of the test statistics of all CpG sites within the DMR.

Similarly, users can specify a threshold for difference. For example, to detect regions with difference greater than 0.1, do:

```
> dmrs2 <- callDMR(dmlTest, delta=0.1, p.threshold=0.05)
> head(dmrs2)
      chr    start      end length nCG meanMethy1 meanMethy2 diff.Methy areaStat
229 chr18 13131705 13131880    176   6  0.9689797 0.06107942 0.90790031 68.74108
36  chr18  4657576  4657639     64   4  0.5084704 0.31870495 0.18976548 14.42138
373 chr18 23100427 23100601    175   4  0.5329967 0.43403678 0.09895989 12.76449
22  chr18  4222533  4222608     76   4  0.7882151 0.36126847 0.42694665 12.66181
41  chr18  5027549  5027743    195   5  0.6696348 0.32605410 0.34358066 11.42014
```

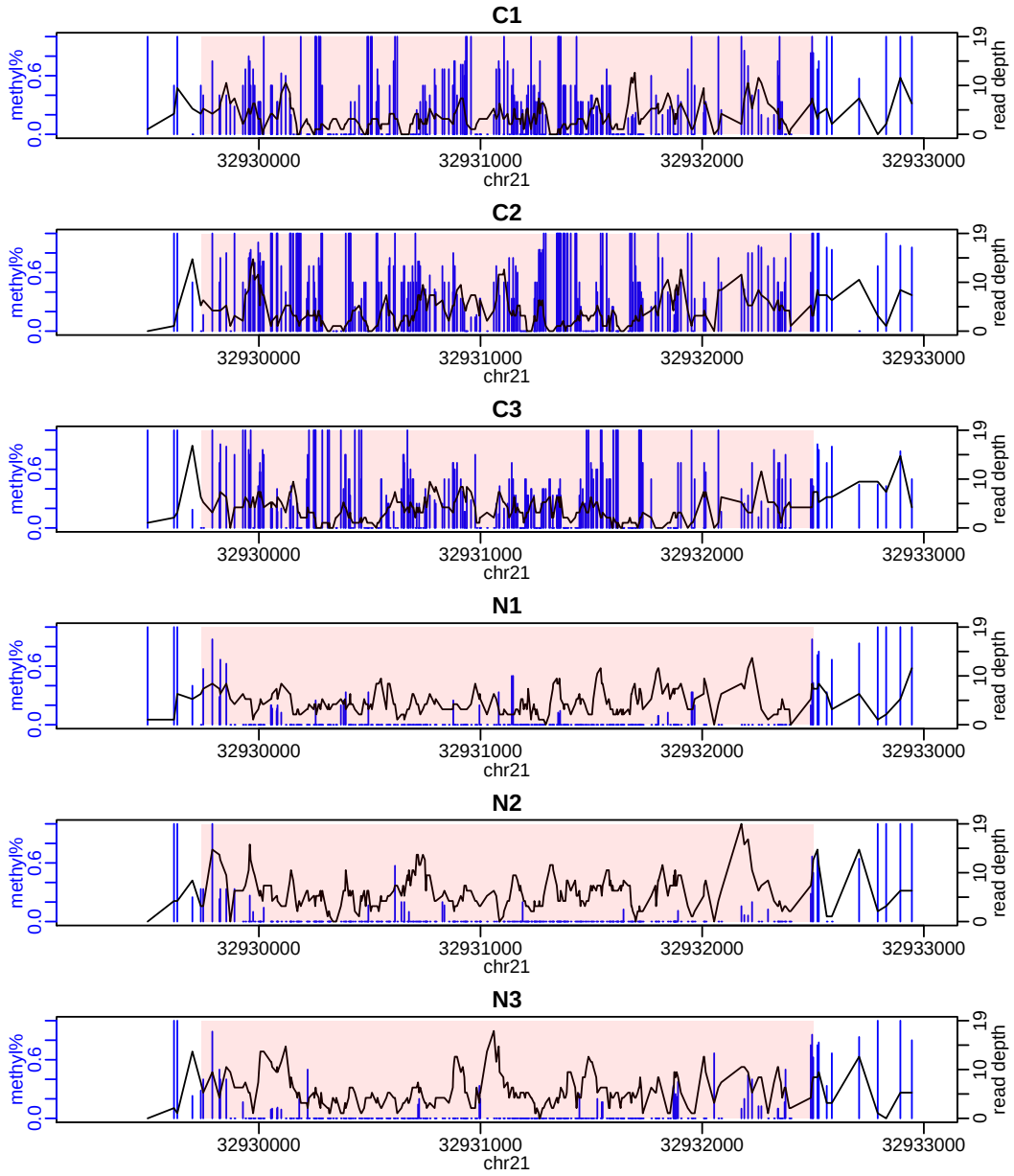
Note that the distribution of test statistics (and p-values) depends on the differences in methylation levels and biological variations, as well as technical factors such as coverage depth. It is very difficulty to select a natural and rigorous threshold for defining DMRs. We recommend users try different thresholds in order to obtain satisfactory results.

5. The DMRs can be visualized using `showOneDMR` function, This function provides more information than the `plotRegion` function in `bsseq`. It plots the methylation percentages as well as the coverage depths at each CpG sites, instead of just the smoothed curve. So the coverage depth information will be available in the figure.

   To use the function, do

   ```
   > showOneDMR(dmrs[1,], BSobj)
   ```

   The result figure looks like the following. Note that the figure below is not generated from the above example. The example data are from RRBS experiment so the DMRs are much shorter.



3.4 DML/DMR detection from general experimental design

In DSS, BS-seq data from a general experimental design (such as crossed experiment, or experiment with covariates) is modeled through a generalized linear model framework. We use arcsine link function instead of the typical logit link for it better deals with data at boundaries (methylation levels close to 0 or 1). Linear model fitting is done through ordinary least square on transformed methylation levels. Standard errors for the estimates are derived with consideration of count data distribution and transformation. A Wald test is applied to perform hypothesis testing.

1. Load in data distributed with DSS. This is a small portion of a set of RRBS experiments. There are 5000 CpG sites and 16 samples. The experiment is a 2 design (2 cases and 2 cell types). There are 4 replicates in each case:cell combination.

```
> data(RRBS)
> RRBS
An object of type 'BSseq' with
  5000 methylation loci
  16 samples
has not been smoothed
> design
   case cell
1    HC   rN
2    HC   rN
3    HC   rN
4   SLE   aN
5   SLE   rN
6   SLE   aN
7   SLE   rN
8   SLE   aN
9   SLE   rN
10  SLE   aN
11  SLE   rN
12   HC   aN
13   HC   aN
14   HC   aN
15   HC   aN
16   HC   rN
```

2. Fit a linear model using `DMLfit.multiFactor` function, include case, cell, and case:cell interaction.

```
> DMLfit = DMLfit.multiFactor(RRBS, design=design, formula=~case+cell+case:cell)
Fitting DML model for CpG site:
```

3. Use `DMLtest.multiFactor` function to test the cell effect. It is important to note that the `coef` parameter is the index of the coefficient to be tested for being 0. Because the model (as specified by `formula` in `DMLfit.multiFactor`) include intercept, the cell effect is the 3rd column in the design matrix, so we use `coef=3` here.

```
> DMLtest.cell = DMLtest.multiFactor(DMLfit, coef=3)
```

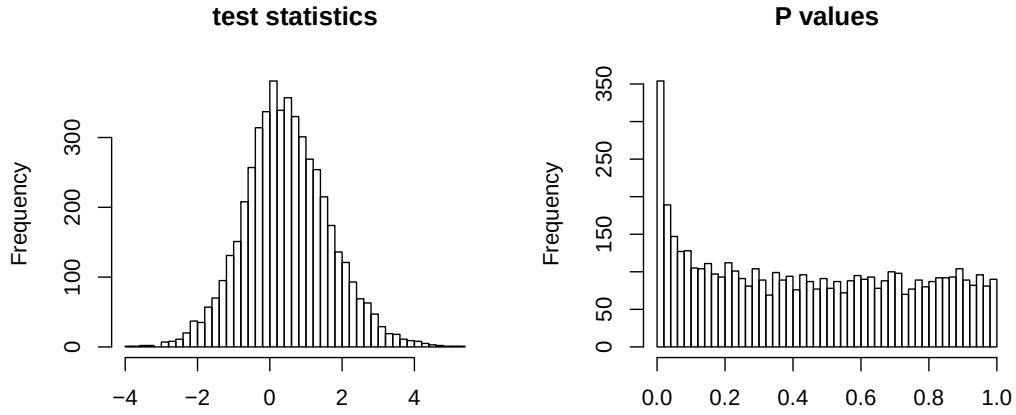
Result from this step is a data frame with chromosome number, CpG site position, test statistics, p-values (from normal distribution), and FDR. Rows are sorted by chromosome/position of the CpG sites. To obtain top ranked CpG sites, one can sort the data frame using following codes:

```
> ix=sort(DMLtest.cell[,"pvals"], index.return=TRUE)$ix
> head(DMLtest.cell[ix,])
      chr     pos     stat        pvals         fdrs
1273 chr1 2930315 5.280301 1.289720e-07 0.0006448599
4706 chr1 3321251 5.037839 4.708164e-07 0.0011770409
3276 chr1 3143987 4.910412 9.088510e-07 0.0015147517
2547 chr1 3069876 4.754812 1.986316e-06 0.0024828953
```

```
3061 chr1 3121473 4.675736 2.929010e-06 0.0029290097
527  chr1 2817715 4.441198 8.945925e-06 0.0065858325
```

Below is a figure showing the distributions of test statistics and p-values from this example dataset

```
> par(mfrow=c(1,2))
> hist(DMLtest.cell$stat, 50, main="test statistics", xlab="")
> hist(DMLtest.cell$pvals, 50, main="P values", xlab="")
```



These procedures are computationally very efficient. For a typical RRBS dataset with 4 million CpG sites, it takes around 20 minutes. In comparison, other methods such as RADMeth or BiSeq takes at least 10 times longer.

4 Session Info

```
> sessionInfo()

R version 3.2.2 Patched (2015-10-08 r69496)
Platform: x86_64-apple-darwin10.8.0 (64-bit)
Running under: OS X 10.6.8 (Snow Leopard)

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
 [1] splines   stats4    parallel  stats     graphics  grDevices utils     datasets
 [9] methods   base

other attached packages:
 [1] edgeR_3.12.0               limma_3.26.0               DSS_2.10.0
 [4] bsseq_1.6.0                SummarizedExperiment_1.0.0 GenomicRanges_1.22.0
 [7] GenomeInfoDb_1.6.0         IRanges_2.4.0              S4Vectors_0.8.0
[10] Biobase_2.30.0             BiocGenerics_0.16.0

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.1         XVector_0.10.0      zlibbioc_1.16.0     munsell_0.4.2
 [5] colorspace_1.2-6    lattice_0.20-33     plyr_1.8.3          tools_3.2.2
 [9] grid_3.2.2          data.table_1.9.6    R.oo_1.19.0         gtools_3.5.0
[13] matrixStats_0.14.2 R.utils_2.1.0       scales_0.3.0        R.methodsS3_1.7.0
[17] locfit_1.5-9.1     BiocStyle_1.8.0     chron_2.3-47
```

References

[1] Hao Feng, Karen Conneely and Hao Wu. (2014). A bayesian hierarchical model to detect differentially methylated loci from single nucleotide resolution sequencing data. *Nucleic acids research* **42**(8), e69–e69.

[2] Hao Wu, Tianlei Xu, Hao Feng, Li Chen, Ben Li, Bing Yao, Zhaohui Qin, Peng Jin and Karen N. Conneely. (2015). Detection of differentially methylated regions from whole-genome bisulfite sequencing data without replicates. *Nucleic acids research* doi: 10.1093/nar/gkv715.

[3] Hao Wu, Chi Wang and Zhijing Wu. (2013). A new shrinkage estimator for dispersion improves differential expression detection in rna-seq data. *Biostatistics* **14**(2), 232–243.