HW8

Guannan Shen

November 19, 2018

Contents

```r
## set up workspace

library(bPeaks)
library(knitr)
library(tidyverse)

library(magrittr)
library(FlowSorted.Blood.450k)
library(quadprog)
library(minfi)
library(IlluminaHumanMethylation450kmanifest)
library(stats)
options(stringsAsFactors = F)
options(dplyr.width = Inf)
getwd()
```

```
## [1] "/home/guanshim/Documents/Stats/CIDA_OMICs/7659Stats_Genetics/HW8"
```

```
## not in function
"%nin%" <- Negate("%in%")

# ######## clean memory ###################### rm(list =
# ls()) gc()
```

1 1. Cell Type Composition

Background on data set: The occurrence of gestational diabetes mellitus (GDM) during pregnancy is believed to alter obesity risk of offspring later in life. It was hypothesized that exposure to maternal GDM in utero will be associated with changes in DNA methylation patterns of key genes and pathways in the offspring, which will mediate the association between in utero exposure and childhood adiposity-related outcomes. Data was collected from the EPOCH (Exploring Perinatal Outcomes in CHildren) study, a historical prospective cohort that enrolled children aged 10.5 on average (T1) who were exposed or not exposed to maternal GDM during the intrauterine life. DNA was extracted from peripheral blood samples collected from children at the T1 EPOCH visit on 85 exposed to GDM and 85 unexposed to GDM, and methylation data was generated using the Illumina Infinium HumanMethylation450 BeadChip. We have provided data on a subset of 10 subjects (all Non-Hispanic Whites) from these two groups to identify methylation sites and nearby genes that show differential methylation between the two groups. The data for this problem is available through a link on Canvas to the "blood" folder on Dropbox.

1.1 (a) Use SWAN normalization from Homework 7, then find differentially methylated positions based on exposure status using dmpFinder(). Are there any DMPs with q-value <= 0.10 (or p-value cutoff of 10-5)? Summarize the results in a table and include the direction (hyper or hypo methylated based on exposure status).

```
## import data under directory
## ~/Documents/Stats/CIDA_OMICs/7659Stats_Genetics/HW8/
baseDir1 <- "blood/plate1"
targets1 <- read.metharray.sheet(baseDir1)
```

```
## [1] "blood/plate1/selected_plate1.csv"
```

```
baseDir2 <- "blood/plate2"
targets2 <- read.metharray.sheet(baseDir2)
```

```
## [1] "blood/plate2/selected_plate2.csv"
```

```
targets <- rbind(targets1, targets2)

rgSet <- read.metharray.exp(targets = targets, extended = T)
```

```
# for part c), use 'extended =F'
rgSet[[1]]
```

```
##  [1] 20209 20216 20279 20339 20342 20108 20531 20275 20281 20097 20505
## [12] 20513 20525 20563 20572 20029 20206 20027 20075 20071
```

```
# clinical and groups
sampleNames(rgSet) = rgSet[[1]]
getManifest(rgSet)
```

```
## IlluminaMethylationManifest object
## Annotation
##   array: IlluminaHumanMethylation450k
## Number of type I probes: 135476
## Number of type II probes: 350036
## Number of control probes: 850
## Number of SNP type I probes: 25
## Number of SNP type II probes: 40
```

```
clindat <- read.table("blood/demographic.txt", sep = "\t", header = T)
## grouping
class(clindat$Exposure)
```

```
## [1] "integer"
```

```
clindat$child_sex
```

```
##  [1] "M" "F" "M" "M" "F" "M" "F" "M" "M" "F" "M" "F" "M" "M" "M" "M" "F"
## [18] "M" "M" "M"
```

```
pData(rgSet)$Sample_Group <- clindat$Exposure
pData(rgSet)$child_sex <- clindat$child_sex

# swan normalization and exposure groups
mset <- preprocessRaw(rgSet)
msetSWAN <- preprocessSWAN(rgSet)
# m values
mvalueset <- getM(msetSWAN)
## dmp by exposure
dmp_set <- dmpFinder(mvalueset, pheno = clindat$Exposure, type = c("categorical"))

# sig table 0 rows
dmp_set[dmp_set$qval <= 0.1, ]
```

```
##    intercept  f pval qval
## NA        NA NA   NA   NA
```

```
# 4 rows
dmp_settop <- dmp_set[dmp_set$pval <= 1e-05, ] %>% mutate(methy_status = if_else(.$intercept >
    0, "hyper-methylated", "hypo-methylated"), CpGs = rownames(.)) %>%
    dplyr::select(CpGs, everything())

kable(dmp_settop, caption = "Top CpGs by Exposure Status", align = "c")
```

Table 1: Top CpGs by Exposure Status

CpGs	intercept	f	pval	qval	methy_status
cg25761397	4.1477634	50.65372	1.2e-06	0.5544406	hyper-methylated
cg00244352	3.9200485	41.62060	4.5e-06	0.6182157	hyper-methylated
cg15614119	0.9081008	39.78694	6.0e-06	0.6182157	hyper-methylated
cg21088165	4.1624960	37.19976	9.2e-06	0.6182157	hyper-methylated
NA	NA	NA	NA	NA	NA

The direction of methylation was defined by the value of intercept, positive intercept means hyper-methylation in exposure status 1.

1.2 (b) Because blood is a heterogeneous collection of different cell types, it has been recognized that it may be important to adjust for cell-type composition in your analysis. Why is cell-type composition relevant for DNA methylation studies? What are the methods available to adjust for cell-type composition when celltypes are not directly measured (as in this example)?

Blood is a heterogeneous collection of different cell types, each with a very different DNA methylation profile. Basically, DNA methylation has a very high variability on cell type level. In the Jaffe & Irizarry, Genome Biology (2014) paper, they showed in five previously published studies, cellular composition explained much of the observed variability in DNA methylation. That is, cellular composition is a confounder in blood sample methylation study.

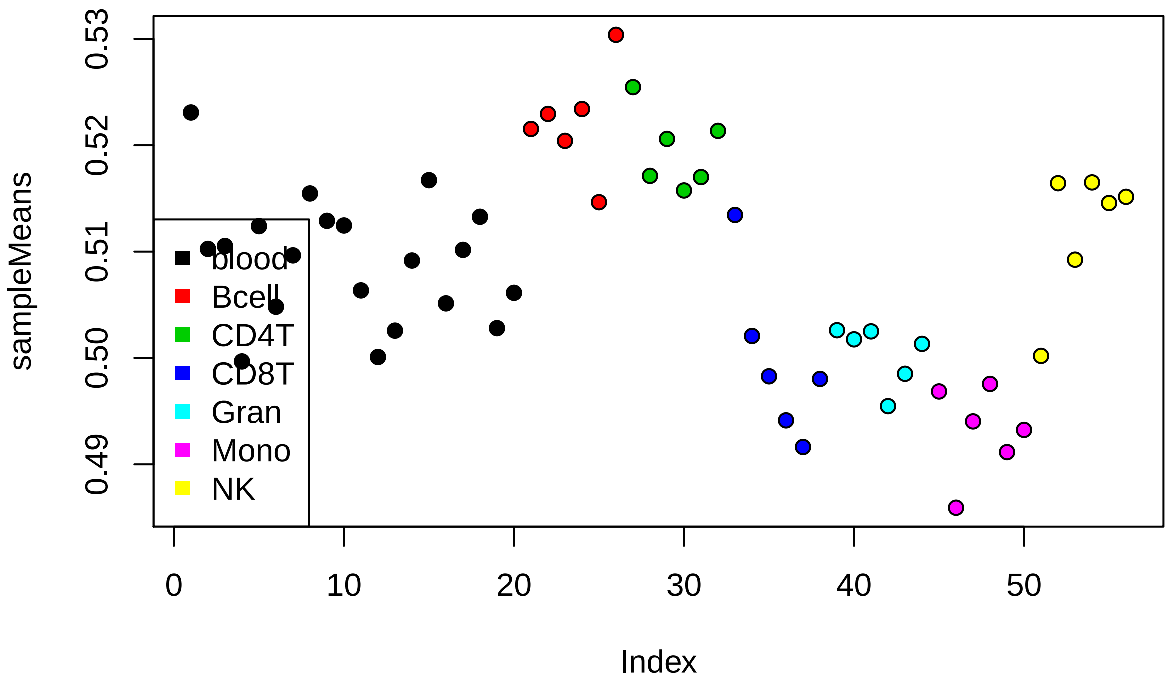
Methods to estimate cell counts have been developed by different approaches, such as reference-free deconvolution, reference-based deconvolution and Bayesian semi-supervised method (BayesCCE). In this example, the Houseman reference-based method, estimateCellCounts(), implemented in the R package minfi is the optimal method for reference-based deconvolution.

1.3 (c) Estimate cell counts using estimateCellCounts(). Explain the graph that is displayed by this function. Then repeat part a), but include cell-type composition as covariates in your model. Summarize your results again in a table. How do the results compare between the unadjusted and adjusted cell-type composition analysis? Provide possible explanations if you do not see differences.Based on the available clinical data file, discuss what other covariates that you may consider including in the model (no need to implement).

```r
rgSet_F <- read.metharray.exp(targets = targets, extended = F)
sampleNames(rgSet_F) = rgSet_F[[1]]
pData(rgSet_F)$Sample_Group <- clindat$Exposure
pData(rgSet_F)$child_sex <- clindat$child_sex

# swan normalization and exposure groups
mSet_F <- preprocessRaw(rgSet_F)
mSet_FSWAN <- preprocessSWAN(rgSet_F)
# m values
mvalueSet_F <- getM(mSet_FSWAN)
# estimate cell
```

```
cellcount <- estimateCellCounts(rgSet_F, compositeCellType = "Blood",
    meanPlot = T, cellTypes = c("CD8T", "CD4T", "NK", "Bcell",
        "Mono", "Gran"))
```



```
# samples and groups
sum(rownames(cellcount) != rgSet_F[[1]])
```

```
## [1] 0
```

```
cell <- as.data.frame(cellcount)
cell$exposure <- clindat$Exposure
# have a look
cell[1, ]
```

```
##            CD8T      CD4T         NK     Bcell      Mono      Gran
## 20209 0.2447301 0.2122113 0.01179639 0.1221149 0.1301415 0.3087963
##       exposure
## 20209        1
```

```
# reference with dmpfinder dmp by exposure
dmp_Set_F <- dmpFinder(mvalueSet_F, pheno = clindat$Exposure,
    type = c("categorical"))
dmp_Set_Ftop <- dmp_Set_F[dmp_Set_F$pval <= 1e-05, ] %>% mutate(methy_status = if_else(.$intercept >
    0, "hyper-methylated", "hypo-methylated"), CpGs = rownames(.)) %>%
    dplyr::select(CpGs, everything())

kable(dmp_Set_Ftop, caption = "Top CpGs by Exposure Status",
    align = "c")
```

Table 2: Top CpGs by Exposure Status

CpGs	intercept	f	pval	qval	methy_status
cg25761397	4.1623166	50.18147	1.3e-06	0.5926098	hyper-methylated
cg00244352	3.9295352	41.60359	4.5e-06	0.6276356	hyper-methylated

CpGs	intercept	f	pval	qval	methy_status
cg15614119	0.9096436	39.37280	6.5e-06	0.6276356	hyper-methylated
cg21088165	4.1790237	38.08853	7.9e-06	0.6276356	hyper-methylated

```r
# lm
n_cpgs <- nrow(mvalueSet_F)
outcome_lm <- lapply(1:n_cpgs, function(i) {
    lm = lm(mvalueSet_F[i, ] ~ exposure + Bcell + CD4T + CD8T +
        Gran + Mono + NK, data = cell)
    coef = c(summary(lm)$coefficients[2, 1], summary(lm)$coefficients[2,
        4])
    if (i %in% seq(0, 5e+05, by = 20000)) {
        print(i)
    }
    return(coef)

})
```

```
## [1] 20000
## [1] 40000
## [1] 60000
## [1] 80000
## [1] 100000
## [1] 120000
## [1] 140000
## [1] 160000
## [1] 180000
## [1] 200000
## [1] 220000
## [1] 240000
## [1] 260000
## [1] 280000
## [1] 300000
## [1] 320000
## [1] 340000
## [1] 360000
## [1] 380000
## [1] 400000
## [1] 420000
## [1] 440000
## [1] 460000
## [1] 480000
```

```r
outcome_lm <- data.frame(matrix(unlist(outcome_lm), ncol = 2,
    byrow = TRUE, dimnames = list(c(rownames(mvalueSet_F)), c("Estimate",
        "p.value"))))

outcome_lm <- outcome_lm %>% mutate(CpGs = rownames(mvalueSet_F)) %>%
    dplyr::select(CpGs, everything())
outcome_lm <- outcome_lm[order(outcome_lm$p.value, decreasing = F),
    ]

outcome_lm1 <- outcome_lm[outcome_lm$p.value <= 1e-05, ] %>%
    mutate(methy_status = if_else(.$Estimate > 0, "hyper-methylated",
        "hypo-methylated")) %>% dplyr::select(CpGs, everything())

kable(outcome_lm1, caption = "Top CpGs by Exposure Status, adjusting for cell types",
    align = "c")
```

Table 3: Top CpGs by Exposure Status, adjusting for cell types

CpGs	Estimate	p.value	methy_status
cg22685502	-0.3911621	5.0e-07	hypo-methylated
cg08829149	-0.4640818	8.0e-07	hypo-methylated
cg00147172	-0.5691915	1.3e-06	hypo-methylated
cg06759518	-0.4347827	3.2e-06	hypo-methylated
cg20911304	-0.3437518	3.7e-06	hypo-methylated
cg13912480	-0.5335481	4.2e-06	hypo-methylated
cg00850971	0.2083531	4.4e-06	hyper-methylated
cg03605666	-0.3267800	4.4e-06	hypo-methylated
cg10104683	-0.5824723	7.1e-06	hypo-methylated
cg02493905	-0.3557411	7.3e-06	hypo-methylated
cg15431137	-0.3245363	7.6e-06	hypo-methylated
cg15614119	-0.2355460	7.6e-06	hypo-methylated
cg15801967	0.2205414	8.7e-06	hyper-methylated
cg11088968	-0.3654090	9.3e-06	hypo-methylated
cg09065876	-0.3446293	9.7e-06	hypo-methylated
cg00549412	-0.4219419	9.9e-06	hypo-methylated

```
## compare
cpg_com <- dmp_Set_Ftop$CpGs[dmp_Set_Ftop$CpGs %in% outcome_lm1$CpGs]
```

The meanPlot is the average DNA methylation across the cell-type discrimating probes within the mixed and sorted samples. The table above showed the CpGs with p values less than 0.00001. The results between the unadjusted and adjusted cell-type composition analysis differ a lot. Only one CpG cg15614119 was shared in both approaches. The whole linear regression has the M-values of CpGs as the outcome (which is bimodal, might apply log10 transformation), and exposure, cell types as covariates. The direction here was defined by the value of the beta estimates of the exposure status. Both the directions and scopes of exposure effects on CpGs M values change a lot.

The gender of child may also be included in the model.

2 2. ChIP-Seq

• Download the data provided on Canvas (tup1 IP.txt, mock IP.txt, input IP.txt). These are three ChIP-Seq experiments in yeast from Park et al. 2013 PLoS One 8:12 e83506 (http://www.ncbi.nlm.nih.gov/pubmed/24349523). Tup1 is a transcriptional repressor and the mock and input are two different controls for comparison.

2.1 (a) By examining the GEO links and reference, what methods were used for sequencing, basecalling, mapping reads and dealing with non-uniquely mapped reads? What is the difference between mock and input controls?

Sequencing was carried out by either Illumina HiSeq 2000 or SOLiD V4. Basecalls performed using CASAVA version 1.8. ChIP-seq reads were aligned to the sacCer3 genome assembly using BWA (Version: 0.5.9-r16) with default options. Non-uniquely mapped reads were filtered out in order to remove the reads with low mapping quality.

Mock ChIP DNA was prepared by immunoprecipitation with IgG Sepharose in the wild type strain with no TAP-tagged protein expression. Input DNA was prepared in parallel with the SWI6-TAP ChIP sample but leaving out the immunoprecipitation step. So the mock had the no-antibody ip step, but the input did not have the ip step.

2.2 (b) Using baseLineCalc(), what is the average number of sequenced mapped at each position? Does the Tup1 ChIP or mock sample have more average reads? (Hint: This function and the function in part c) only need the last column of read counts from allData*IPdataandallData*controlData)

```
data(yeastCDS)  #for gene location annotation
## also input_IP.txt
allData <- dataReading("tup1_IP.txt", "mock_IP.txt", yeastSpecies = yeastCDS$Saccharomyces.cerevisiae)
```

```
## [1] "**********************************************"
## [1] "Reading IP and control datasets... "
## [1] "... done"
## [1] "**********************************************"
## [1] ""
```

```
# read in data

## average test group
mean_tup1 <- baseLineCalc(allData$IPdata[, 3])

## average control group
mean_mock <- baseLineCalc(allData$controlData[, 3])

##
mean_tup1 > mean_mock
```

```
## [1] TRUE
```

The average number of sequenced mapped at each position for Tup1 ChIP is 65.5266955, for mock control is 14.4172567. The Tup1 ChIP has more average reads.

2.3 (c) Examining only chromosome V (“chrV”), by subsetting the first column of allData*IPdataandallData*controlData, run peakDetection(), with the baseLineIP and baseLineControl values calculated in part b).

```
## chromosomes names
chromNames <- unique(allData$IPdata[, 1])
chromNames[5]
```

```
## [1] "chrV"
```

```
## mask
filter <- stats::filter
##
IPsignal <- dataSmoothing(allData$IPdata[allData$IPdata[, 1] ==
    "chrV", 3], 20)
controlSignal <- dataSmoothing(allData$controlData[allData$controlData[,
    1] == "chrV", 3], 20)
# peak detection
detectedPeaks <- peakDetection(IPdata = IPsignal, controlData = controlSignal,
    chrName = as.character(chromNames[5]), windowSize = 150,
    windowOverlap = 50, outputName = paste("bPeaks_example_",
        chromNames[5], sep = ""), baseLineIP = mean_tup1, baseLineControl = mean_mock,
    IPthreshold = 6, controlThreshold = 4, ratioThreshold = 2,
    averageThreshold = 0.7, peakDrawing = TRUE)
```

```
## [1] "1% of windows were analyzed"
## [1] "5% of windows were analyzed"
## [1] "10% of windows were analyzed"
## [1] "20% of windows were analyzed"
## [1] "50% of windows were analyzed"
## [1] "60% of windows were analyzed"
## [1] "70% of windows were analyzed"
## [1] "80% of windows were analyzed"
## [1] "90% of windows were analyzed"
## [1] "100% of windows were analyzed"
## [1] "65 significant window(s) were detected..."
## [1] "... starting merging procedure"
## [1] ""
## [1] "# of detected basic peaks (bPeaks) :  17"
## [1] ""
## [1] "** Saving chromosome information in PDF file:"
## [1] "bPeaks_example_chrV_dataSummary.pdf"
## [1] ""
```

```
## [1] "** Bed file saving in:"
## [1] "bPeaks_example_chrV.bed"
## [1] ""
## [1] "** Peak drawing in PDF file:"
## [1] "bPeaks_example_chrV_bPeaksDrawing.pdf"
```

```
# print detected genomic positions
print(detectedPeaks)
```

```
##        Start    End       IP   Control   log2FC averageLog2
##   [1,]  18801  19101 544.2422 20.599701 4.682779    6.739166
##   [2,]  23551  23851 602.2799 13.411807 5.391484    6.528834
##   [3,]  42151  42351 416.3720 34.667097 3.546992    6.929902
##   [4,]  69051  69401 649.0438 24.936424 4.687456    6.977404
##   [5,]  77551  77851 548.1505  9.372678 5.726180    6.228723
##   [6,]  85151  85451 657.5792 33.835124 4.229986    7.235636
##   [7,] 138601 138751 400.4414 57.064448 2.789463    7.254314
##   [8,] 141101 141451 577.4675 20.435729 4.741518    6.785068
##   [9,] 152551 152901 492.3532 21.662865 4.447711    6.710430
##  [10,] 153051 153251 415.6193 14.771442 4.723167    6.340463
## [11,] 174651 175051 727.1665 30.597157 4.586201    7.190346
## [12,] 225501 225701 439.7970 46.143192 3.225586    7.170489
## [13,] 241951 242351 603.2726 20.230873 4.984258    6.719984
## [14,] 311901 312201 530.5972 38.898361 3.740722    7.175914
## [15,] 461851 462151 599.5545 16.757349 5.074471    6.685168
## [16,] 491001 491401 692.5461 19.716659 5.045928    6.880126
## [17,] 492351 492501 394.4600 57.206913 2.764269    7.245253
```

```
#
nrow(detectedPeaks)
```

```
## [1] 17
```

```
## peak locations
peakLocation(bedFile = "bPeaks_example_chrV.bed", cdsPositions = yeastCDS$Saccharomyces.cerevisiae,
    withoutOverlap = FALSE, outputName = "bPeaks_example_chrV",
    promSize = 800)
```

```
## [1] "**********************************************"
## [1] "Opening BED file with peak information:"
## [1] "bPeaks_example_chrV.bed"
## [1] "**********************************************"
## [1] ""
## [1] "Starting peak location regarding ORF/CDS positions..."
## [1] ""
## [1] "# of analyzed peaks:  17"
## [1] "# of peaks UPSTREAM annotated CDS :  15"
## [1] "# of peaks IN annotated CDS :  2"
## [1] ""

## [1] "Saving the results in:"
## [1] "bPeaks_example_chrV_peakLocation_inPromoters.txt"
## [1] "bPeaks_example_chrV_peakLocation_inCDS.txt"
## [1] "**********************************************"

## $numPeaks
## [1] 17
##
## $upFeatures
## [1] 15
##
## $inFeatures
## [1] 2
```

```
## import the result
promo <- read.delim("bPeaks_example_chrV_peakLocation_inPromoters.txt",
    header = F, sep = "\t")
promos <- NULL
for (i in 1:nrow(promo)) {
    promos[i] <- unlist(strsplit(unlist(strsplit(promo[i, 3],
        "|", fixed = TRUE))[3], "=", fixed = T))[2]
}
promos
```

```
##  [1] "UTR2"                           "HXT13"
##  [3] "LSM5"                           "FTR1"
##  [5] "GO:0003674,GO:0005575,GO:0008150" "GO:0003674,GO:0005575,GO:0008150"
##  [7] "ACA1"                             "MIT1"
##  [9] "GO:0003674,GO:0005575,GO:0008150" "MNN1"
## [11] "IES6"                             "GLY1"
## [13] "DSF1"                             "TIR1"
## [15] "MNN1"                             "HYP2"
## [17] "ANP1"                             "HHY1"
## [19] "PRB1"                             "PHM8"
## [21] "ARB1"                             "BUR6"
```

```
cds <- read.delim("bPeaks_example_chrV_peakLocation_inCDS.txt",
    header = F, sep = "\t")
cdss <- NULL
for (i in 1:nrow(cds)) {
    cdss[i] <- unlist(strsplit(unlist(strsplit(cds[i, 3], "|",
        fixed = TRUE))[3], "=", fixed = T))[2]
}
cdss
```

```
## [1] "GO:0003674,GO:0008150,GO:0016021" "UTR5"
```

This method detected 17 peaks. This method using a sliding window to scan the genomic sequence. Four criterion are used to define the peak region, peak shape, and adjacent regions may be merged. Basically, to find the significant window first and then right peak shapes within the window.

These 4 parameters are used to define the interesting region. T1 = IPthreshold is a high number of reads in the IP sample. T2 = controlThreshold is a low number of reads in the control sample. "ratioThreshold" is a high value of log(IP/control) and "averageThreshold" is a good sequencing coverage for both IP and control samples.

"peak_datasummary" shows graphical representations of the detected basic peaks, together with the values of the parameters used to detect the region, and the number of windows and peaks.

"bPeaks-Drawing" 1st page showed the IP and control peak shape of peak #1 versus the location of this peak. Basically, by just plotting the 4 parameters of this peak against the location on the chromosome of this peak.

Find the attached pdf at the end of the document, "example" using the mock and "input" using the input as control.

2.4 (d) The bPeaks package uses simple fold change cutoffs. What alternative methods discussed in class would you apply for a more rigorous approach to detect peaks using a statistical testing framework? Describe the methods and statistical approach.

In the class, we discussed using empirical FDR or Irreproducible Discovery Rate (IDR). The IDR using the pairs of replicates, ranking the peaks by reproducibility, and giving each peak an index. At the end, overall expected rate of irreproducible discoveries (IDR) is calculated.

2.5 (e) Using peakLocation() with the file provided in Canvas "bPeaks results.bed" and yeastCDS$Saccharomyces.cerevisiae for the cdsPositions. How many of the peaks are in genes or promoters? What are those genes?

There are 15 peaks in promoters and 2 peaks in CDS (coding region of the gene). Promoters are UTR2, HXT13, LSM5, FTR1, GO:0003674,GO:0005575,GO:0008150, GO:0003674,GO:0005575,GO:0008150, ACA1, MIT1, GO:0003674,GO:0005575,GO:0008150, MNN1, IES6, GLY1, DSF1, TIR1, MNN1, HYP2, ANP1, HHY1, PRB1, PHM8, ARB1, BUR6. And cds are GO:0003674,GO:0008150,GO:0016021, UTR5.

2.6 f) Now repeat b), c) and e) using the input IP sample instead ("input IP.txt"). What differences do you find between the results using mock or input IP? How does that relate to the conclusions in the paper (see Discussion)?

```
## also input_IP.txt
allData <- dataReading("tup1_IP.txt", "input_IP.txt", yeastSpecies = yeastCDS$Saccharomyces.cerevisiae)
```

```
## [1] "**********************************************"
## [1] "Reading IP and control datasets... "
## [1] "... done"
## [1] "**********************************************"
## [1] ""
```

```
## average test group
mean_tup1 <- baseLineCalc(allData$IPdata[, 3])

## average control group
mean_input <- baseLineCalc(allData$controlData[, 3])
##
mean_tup1 > mean_input
```

```
## [1] TRUE
```

```
## chromosomes names
chromNames <- unique(allData$IPdata[, 1])
chromNames[5]
```

```
## [1] "chrV"
```

```
## mask
filter <- stats::filter
##
IPsignal <- dataSmoothing(allData$IPdata[allData$IPdata[, 1] ==
    "chrV", 3], 20)
controlSignal <- dataSmoothing(allData$controlData[allData$controlData[,
    1] == "chrV", 3], 20)

# peak detection
detectedPeaks <- peakDetection(IPdata = IPsignal, controlData = controlSignal,
    chrName = as.character(chromNames[5]), windowSize = 150,
    windowOverlap = 50, outputName = paste("bPeaks_input_", chromNames[5],
        sep = ""), baseLineIP = mean_tup1, baseLineControl = mean_input,
    IPthreshold = 6, controlThreshold = 4, ratioThreshold = 2,
    averageThreshold = 0.7, peakDrawing = TRUE)
```

```
## [1] "1% of windows were analyzed"
## [1] "5% of windows were analyzed"
## [1] "10% of windows were analyzed"
## [1] "20% of windows were analyzed"
## [1] "50% of windows were analyzed"
## [1] "60% of windows were analyzed"
## [1] "70% of windows were analyzed"
## [1] "80% of windows were analyzed"
## [1] "90% of windows were analyzed"
## [1] "100% of windows were analyzed"
## [1] "62 significant window(s) were detected..."
## [1] "... starting merging procedure"
## [1] ""
## [1] "# of detected basic peaks (bPeaks) :  19"
## [1] ""
## [1] "** Saving chromosome information in PDF file:"
## [1] "bPeaks_input_chrV_dataSummary.pdf"
## [1] ""

## [1] "** Bed file saving in:"
## [1] "bPeaks_input_chrV.bed"
## [1] ""
## [1] "** Peak drawing in PDF file:"
## [1] "bPeaks_input_chrV_bPeaksDrawing.pdf"
```

```
# print detected genomic positions
print(detectedPeaks)
```

```
##        Start    End       IP   Control   log2FC averageLog2
##  [1,]  18801  19101 544.2422  81.42731 2.716863    7.722124
##  [2,]  23551  23851 602.2799  57.63604 3.364353    7.542399
##  [3,]  42201  42351 437.1599 105.61993 2.038980    7.755824
##  [4,]  69051  69401 649.0438  90.92819 2.798917    7.921674
##  [5,]  77551  77851 548.1505  90.05766 2.584362    7.799632
##  [6,]  85201  85451 678.9836 137.14236 2.285811    8.252864
##  [7,]  86551  86701 495.5193 117.50218 2.066936    7.922238
##  [8,] 141151 141401 648.0759 126.22032 2.354067    8.162930
##  [9,] 152601 152851 540.4851 130.38497 2.041961    8.058553
## [10,] 153051 153251 415.6193  96.80221 2.090803    7.656645
## [11,] 174651 175051 727.1665  94.03729 2.948449    8.009223
## [12,] 225501 225651 453.4164 105.63027 2.091398    7.782172
## [13,] 241951 242301 627.3092 126.23163 2.305980    8.117719
## [14,] 311951 312151 585.9117 135.87845 2.100613    8.146693
## [15,] 354951 355101 419.9041  99.81473 2.061785    7.686455
## [16,] 442051 442501 646.1491 120.55380 2.413311    8.123727
## [17,] 461851 462151 599.5545  80.25828 2.880696    7.782056
## [18,] 491051 491401 737.4295  88.67479 3.033522    7.992179
## [19,] 492351 492501 394.4600  66.49524 2.550674    7.352051
```

```
#
nrow(detectedPeaks)
```

```
## [1] 19
```

```
## peak locations
peakLocation(bedFile = "bPeaks_input_chrV.bed", cdsPositions = yeastCDS$Saccharomyces.cerevisiae,
    withoutOverlap = FALSE, outputName = "bPeaks_input_chrV",
    promSize = 800)
```

```
## [1] "**********************************************"
## [1] "Opening BED file with peak information:"
## [1] "bPeaks_input_chrV.bed"
## [1] "**********************************************"
## [1] ""
## [1] "Starting peak location regarding ORF/CDS positions..."
## [1] ""
## [1] "# of analyzed peaks:  19"
## [1] "# of peaks UPSTREAM annotated CDS :  18"
## [1] "# of peaks IN annotated CDS :  2"
## [1] ""

## [1] "Saving the results in:"
## [1] "bPeaks_input_chrV_peakLocation_inPromoters.txt"
## [1] "bPeaks_input_chrV_peakLocation_inCDS.txt"
## [1] "**********************************************"

## $numPeaks
## [1] 19
##
## $upFeatures
## [1] 18
##
## $inFeatures
## [1] 2
```

```
## import the result
promo_in <- read.delim("bPeaks_input_chrV_peakLocation_inPromoters.txt",
    header = F, sep = "\t")
promos_in <- NULL
for (i in 1:nrow(promo_in)) {
    promos_in[i] <- unlist(strsplit(unlist(strsplit(promo_in[i,
        3], "|", fixed = TRUE))[3], "=", fixed = T))[2]
}
promos_in
```

```
##  [1] "HXT13"                            "GO:0003674,GO:0005575,GO:0008150"
##  [3] "GO:0003674,GO:0005575,GO:0008150" "TIR1"
##  [5] "LSM5"                             "FTR1"
##  [7] "IES6"                             "GLY1"
##  [9] "DSF1"                             "UTR2"
## [11] "BUR6"                             "GO:0003674,GO:0005575,GO:0008150"
## [13] "MIT1"                             "GO:0003674,GO:0005575,GO:0008150"
## [15] "ACA1"                             "HYP2"
## [17] "ANP1"                             "PHM8"
## [19] "ARB1"                             "MNN1"
## [21] "MCM3"                             "GO:0003674,GO:0005575,GO:0008150"
## [23] "GO:0003674,GO:0005575,GO:0008150" "UBP9"
## [25] "MNN1"                             "HHY1"
## [27] "PRB1"
```

```
cds_in <- read.delim("bPeaks_input_chrV_peakLocation_inCDS.txt",
    header = F, sep = "\t")
cdss_in <- NULL
for (i in 1:nrow(cds_in)) {
    cdss_in[i] <- unlist(strsplit(unlist(strsplit(cds_in[i, 3],
        "|", fixed = TRUE))[3], "=", fixed = T))[2]
}
cdss_in
```

```
## [1] "GO:0003674,GO:0008150,GO:0016021" "UTR5"
```

```
## difference
sum(promos_in %in% promos == 0)
```

```
## [1] 2
```

With input_IP as control, 19 peaks were detected. 18 peaks are in the promoter region and 2 peaks are in the CDS region. Promoters are HXT13, GO:0003674,GO:0005575,GO:0008150, GO:0003674,GO:0005575,GO:0008150, TIR1, LSM5, FTR1, IES6, GLY1, DSF1, UTR2, BUR6, GO:0003674,GO:0005575,GO:0008150, MIT1, GO:0003674,GO:0005575,GO:0008150, ACA1, HYP2, ANP1, PHM8, ARB1, MNN1, MCM3, GO:0003674,GO:0005575,GO:0008150, GO:0003674,GO:0005575,GO:0008150, UBP9, MNN1, HHY1, PRB1. And cds are GO:0003674,GO:0008150,GO:0016021, UTR5. I found the genes of CDS were identical by these two approaches. And by using input DNA as the control, 2 more genes (promoter region) were detected, others were shared by these 2 different control approaches.

In the paper discussion, they found there was a background signal deriving from expression bias by input DNA control. Thus, the background signal of transcription factor binding cannot be canceled. By using the corresponding mock ChIP data as the control, one can minimize false positives.

According to the discussion, the 2 more genes (promoter region) found by input DNA control might be false positive, might due to the expression bias.