HW7

Guannan Shen

November 10, 2018

Contents

```
## set up workspace
library(knitr)
library(tidyverse)
library(magrittr)
library(shinyMethyl)
library(minfi)
library(bumphunter)
library(qwraps2)
library(IlluminaHumanMethylation450kanno.ilmn12.hg19)
options(stringsAsFactors = F)
options(dplyr.width = Inf)
getwd()
```

```
## [1] "/home/guanshim/Documents/Stats/CIDA_OMICs/7659Stats_Genetics/HW7"
```

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

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

1 1. DNA Methylation QC and Normalization (Illumina 450K)

1.1 Introduction

• For problems #1 and #2 install the following packages from BioConductor: shinyMethyl, minfi, bumphunter and IlluminaHumanMethylation450kanno.ilmn12.hg19

• The data for this problem is available through a link on Canvas to the "idats" folder on Dropbox. This is an Illumina 450K dataset from The Cancer Genome Atlas (TCGA) http://cancergenome.nih.gov/

There are two files for the red and green channels from 6 subjects in the colon adenicarcinoma data set (COAD) from: https://portal.gdc.cancer.gov. The clinical and demographic data has been abbreviated in the targets.csv file.

• Here is the link to the minfi User's Guide: http://www.bioconductor.org/packages/release/bioc/vignettes/minfi/inst/doc/minfi.html

Here is another reference: https://www.bioconductor.org/help/course-materials/2014/BioC2014/minfi_BioC2014.pdf. For information on the control probes, see page 6-7 http://www.filgen.jp/Product/Bioscience/Methyl/Methylation_report.pdf

• Run the following code to load the raw .idat files. But change your path name, and also the path in the Basename column in the **SampleSheet** file in the idats directory.

baseDir = c("/Users/Katerina/Desktop/7659/homeworks/hw7/idats")

targets = read.metharray.sheet(baseDir) rgSet <- read.metharray.exp(targets = targets) annotation(rgSet)

1.2 (a) In clinical manuscripts, the first table often includes summaries of clinical and demographic data (e.g., disease status, race, age, etc.).

There are 3 unique subjects/patients, each patient provided two different biopsies.

```r
# load dataset
baseDir = c("~/Documents/Stats/CIDA_OMICs/7659Stats_Genetics/HW7/idats")
targets = read.metharray.sheet(baseDir)
```

```
## [1] "/home/guanshim/Documents/Stats/CIDA_OMICs/7659Stats_Genetics/HW7/idats/SampleSheet.csv"
```

```r
rgSet <- read.metharray.exp(targets = targets)
annotation(rgSet)
```

```
##                           array                      annotation
## "IlluminaHumanMethylation450k"                   "ilmn12.hg19"
```

```r
# clinical, demagraphics data
table1data <- pData(rgSet)
colnames(table1data)
```

```
##  [1] "id"
##  [2] "sample_type"
##  [3] "patient.age_at_initial_pathologic_diagnosis"
##  [4] "patient.height"
##  [5] "patient.race"
##  [6] "patient.weight"
##  [7] "Status"
##  [8] "Tissue"
##  [9] "Sex"
## [10] "Basename"
## [11] "Array"
## [12] "Slide"
## [13] "filenames"
```

```r
table1.1 <- as.data.frame(table1data) %>% dplyr::rename(Age = patient.age_at_initial_pathologic_diagnosi
    Height = patient.height, Weight = patient.weight, Race = patient.race)
table1.2 <- table1.1[c(1, 3, 5), ]
# make table 1
my_summary_1 <- list(Demographics = list(`Age, mean(SD)` = ~qwraps2::mean_sd(Age,
    denote_sd = "paren"), `Female, No. (%)` = ~n_perc0(Sex ==
    "FEMALE"), `Height, mean(SD)` = ~mean_sd(Height, denote_sd = "paren"),
    `Weight, mean(SD)` = ~mean_sd(Weight, denote_sd = "paren"),
    `White, No. (%)` = ~n_perc0(Race == "WHITE"), `Black, No. (%)` = ~n_perc0(Race ==
        "BLACK OR AFRICAN AMERICAN")))
# clinical part table 1
my_summary_2 <- list(`Clinical Status` = list(`Colon Tissue, No. (%)` = ~n_perc0(Tissue ==
    "colon"), `Cancer Tissue, No. (%)` = ~n_perc0(Status == "cancer"),
    `Normal Tissue, No. (%)` = ~n_perc0(Status == "normal")))
```

```r
tab1 <- rbind(summary_table(table1.2, my_summary_1), summary_table(table1.1,
    my_summary_2))
rownames(tab1)
```

```
## [1] "Age, mean(SD)"          "Female, No. (%)"
## [3] "Height, mean(SD)"       "Weight, mean(SD)"
## [5] "White, No. (%)"         "Black, No. (%)"
## [7] "Colon Tissue, No. (%)"  "Cancer Tissue, No. (%)"
## [9] "Normal Tissue, No. (%)"
```

```r
kable(tab1, col.names = "Subjects = 3, Samples = 6")
```

	Subjects = 3, Samples = 6
Age, mean(SD)	76.67 (7.23)
Female, No. (%)	2 (67)
Height, mean(SD)	166.30 (15.93)
Weight, mean(SD)	61.90 (8.08)
White, No. (%)	2 (67)
Black, No. (%)	1 (33)
Colon Tissue, No. (%)	6 (100)
Cancer Tissue, No. (%)	3 (50)
Normal Tissue, No. (%)	3 (50)

1.3 (b) From the array annotation information given by getManifest(rgSet), how many Type I and II probes are there?

Number of type I probes: 135476

Number of type II probes: 350036

```
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
```

1.4 (c) Display QC plots. In the information from the "targets" file, use "id" for sampNames and repeat the QC plots on "sample type" and "Sex" for sampGroups to see if there are differences in cancer versus normal subjects or by sex. Do you see any differences in the beta values between sample type or sex using the QC reports? Are there any samples that appear to be problematic?

Beta values are between 0 and 1 with 0 being unmethylated and 1 fully methylated.

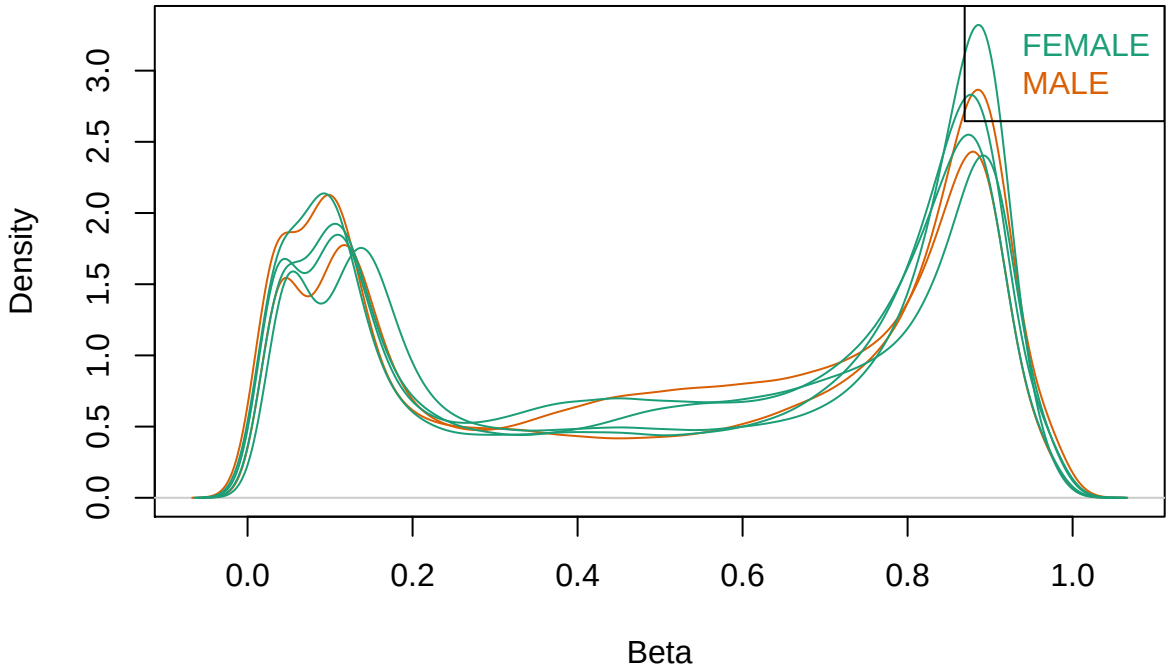
There is no obvious difference in the Beta values by gender based on the density plots. However, primary tumor sample appear to have higher unmethylated peaks.

The default in the function displayed control probes "BISULFITE CONVERSION I" and "BISULFITE CONVERSION II". These controls assess the efficiency of bisulfite conversion of the genomic DNA. During the experimental procedure, the unmethylated C was converted to T. "BISULFITE CONVERSION I" uses the type I probe design, and another one uses infinium II probe design. Red or green channel is to monitor the converted probes or unconverted, separately. In terms of this study, there is no sample appears to be problematic.

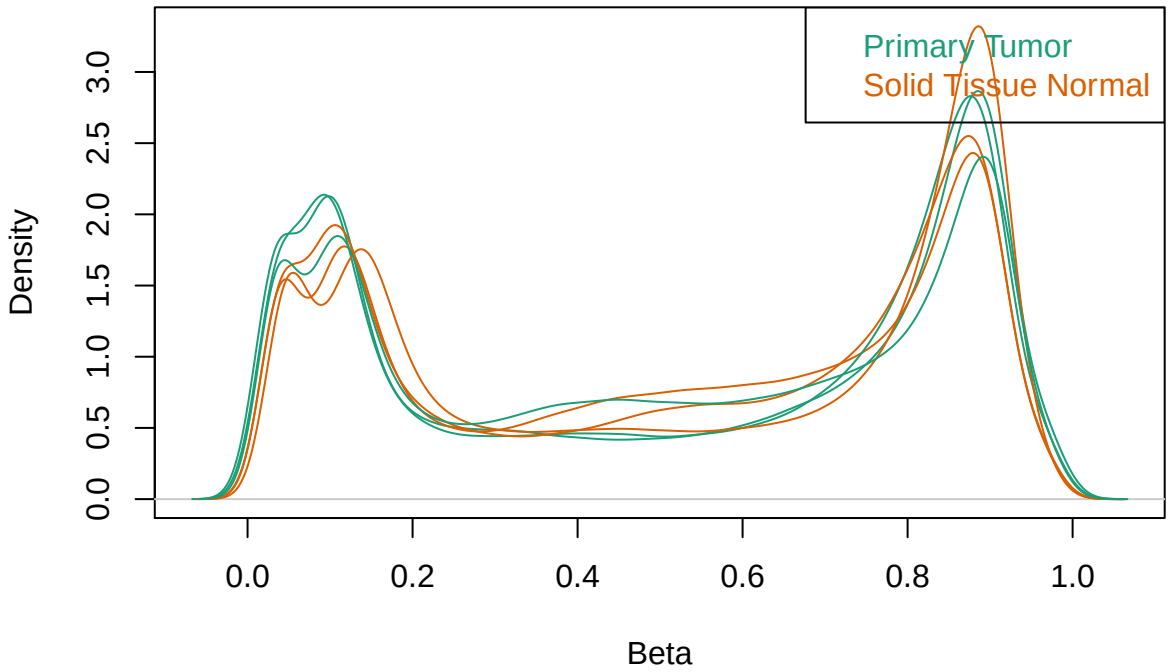
```
# labels
groups_sex <- pData(rgSet)$Sex
groups_sample <- pData(rgSet)$sample_type
pData(rgSet)$sample_type
```

```
## [1] "Primary Tumor"       "Solid Tissue Normal" "Primary Tumor"
## [4] "Solid Tissue Normal" "Solid Tissue Normal" "Primary Tumor"
```

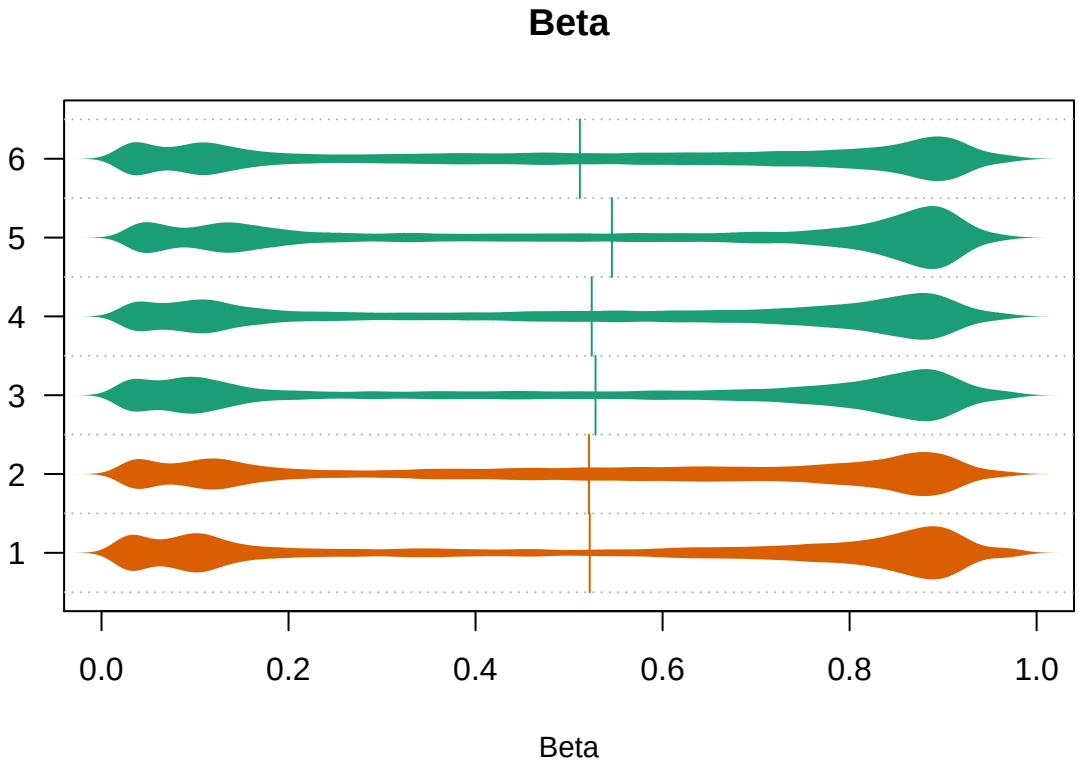
```
sampnames <- pData(rgSet)$id
# pdf
densityPlot(rgSet, sampGroups = groups_sex, sampNames = sampnames)
```



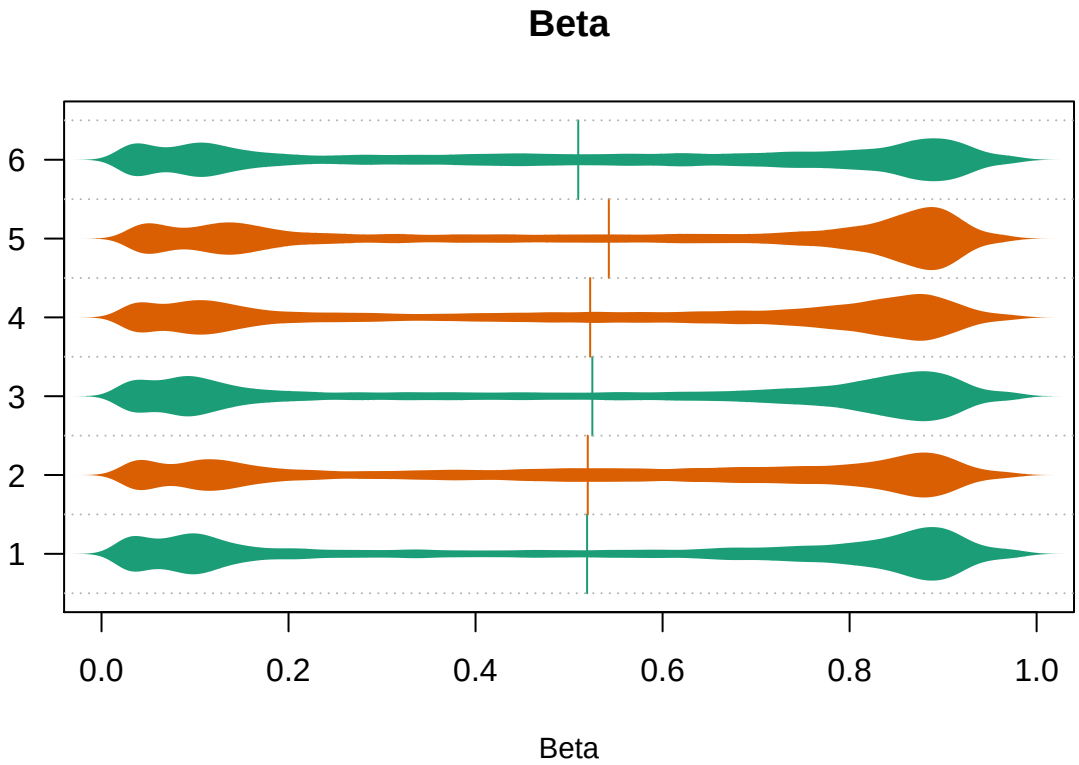

```
densityPlot(rgSet, sampGroups = groups_sample, sampNames = sampnames)
```



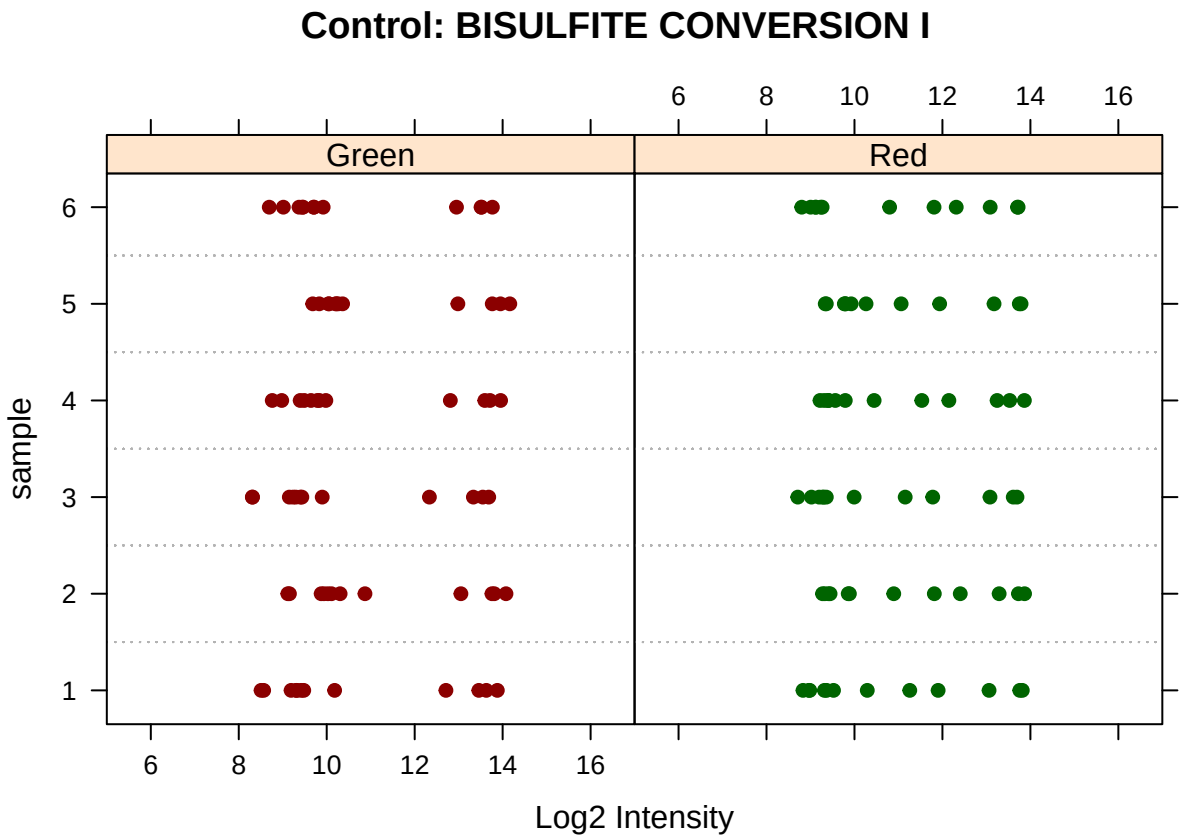

```
# density bean
densityBeanPlot(rgSet, sampGroups = groups_sex, sampNames = sampnames)
```

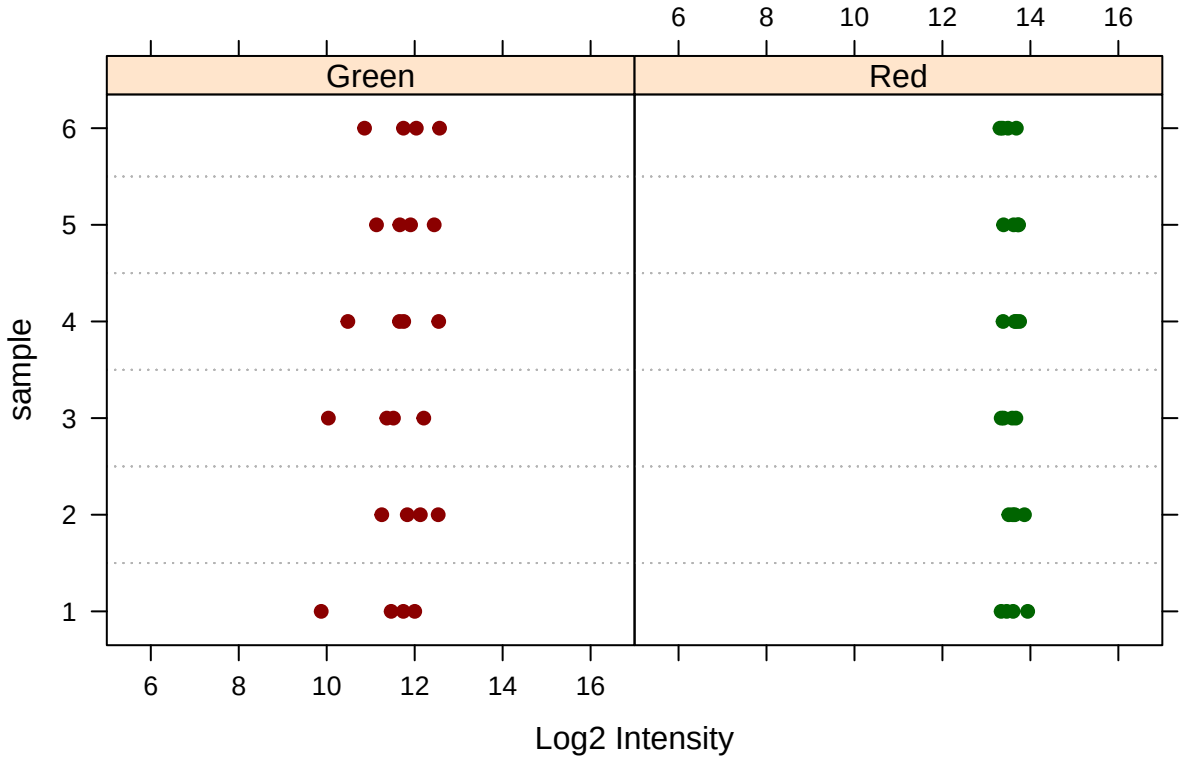



```
densityBeanPlot(rgSet, sampGroups = groups_sample, sampNames = sampnames)
```




```
# control
controlStripPlot(rgSet, sampNames = sampnames)
```


Control: BISULFITE CONVERSION I
Green
Red
sample
Log2 Intensity
6
8
10
12
14
16
1
2
3
4
5
6


Control: BISULFITE CONVERSION II
Green
Red
sample
Log2 Intensity
6
8
10
12
14
16
1
2
3
4
5
6

1.5 (d) Describe their purpose of the different control probes on the array (see link above and help(qcReport)).

Different control probes work as quality controls.

Staining controls are used to examine the efficiency of the staining step in both the red and green channels.

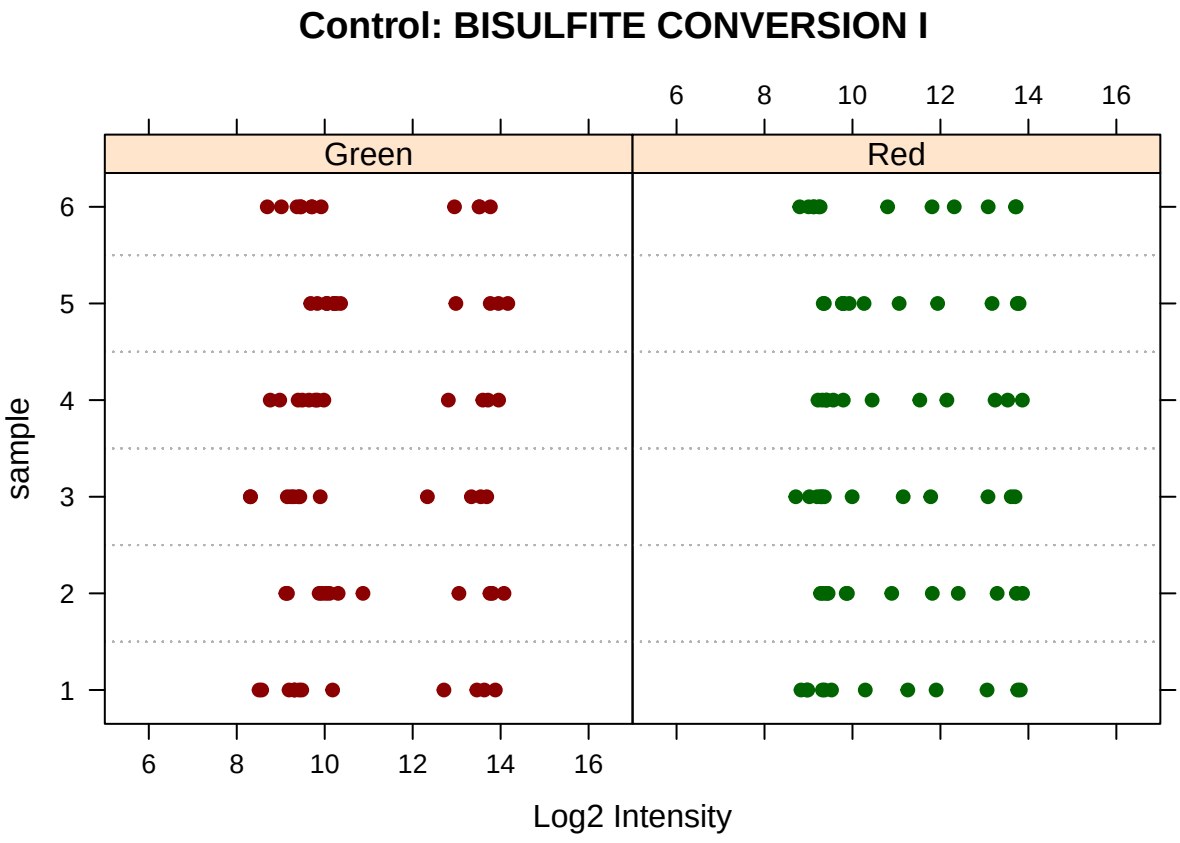
Extension controls test the extension efficiency.

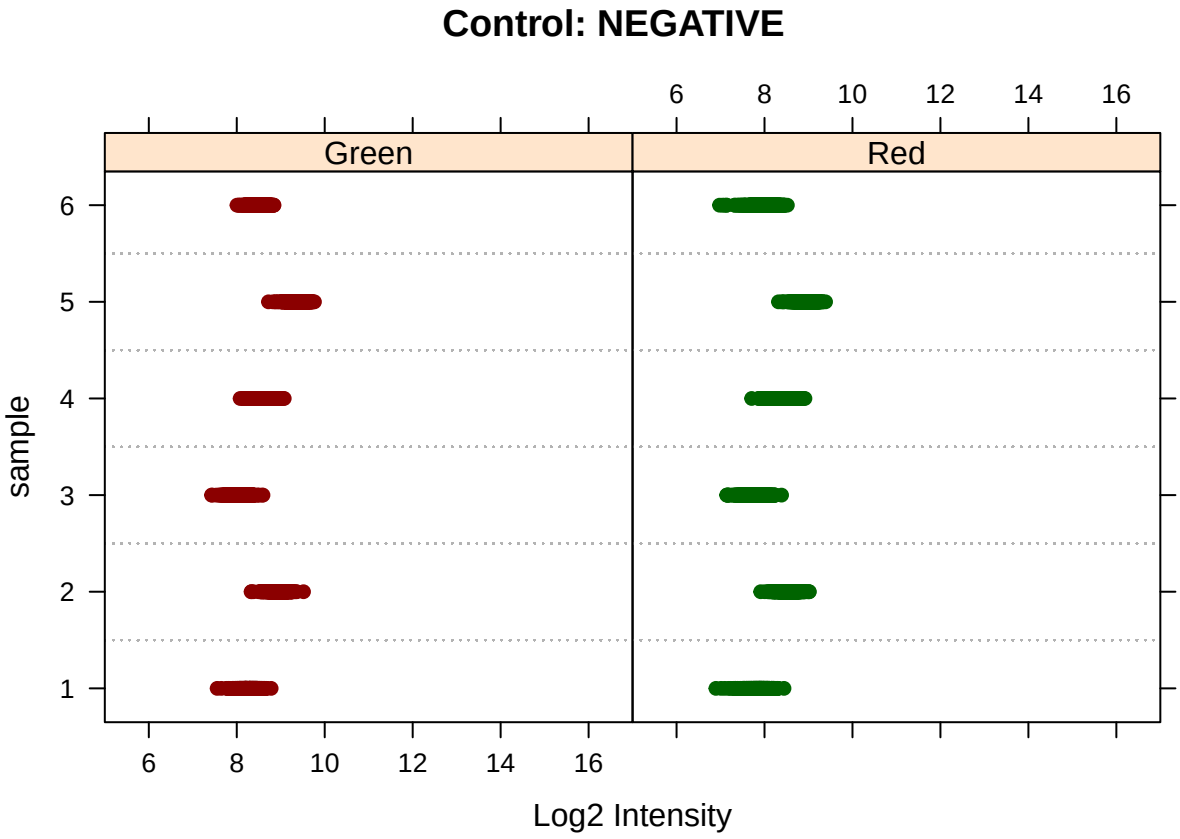
The default in the function displayed control probes "BISULFITE CONVERSION I" and "BISULFITE CONVERSION II". These controls assess the efficiency of bisulfite conversion of the genomic DNA. During the experimental procedure, the unmethylated C was converted to T. "BISULFITE CONVERSION I" uses the type I probe design, and another one uses infinium II probe design. Red or green channel is to monitor the converted probes or unconverted, separately. Type II only has one peak at each of red or green channel, while type I has dual peaks at each color channel.

Negative controls target bisulfite-converted sequences that do not contain CpG dinucleotides. This is to assess the system background by the mean signals from the negative controls, which should be done in both green and red channel.

The ranges of "BISULFITE CONVERSION I" are consistent across samples, telling us both channel has methylated and unmethylated signals. The ranges of negative controls show that the background intensities are around 8 (log2), and are relatively consistent across samples. The data is reliable.

```
help(qcReport)
controlStripPlot(rgSet, sampNames = sampnames, controls = c("BISULFITE CONVERSION I",
    "NEGATIVE"))
```





1.6 (e) Illumina also reports detection p-values, how are these calculated? Using the function detectionP(), which sample had the largest percentage of detection pvalues 0.05? How many probes have average detection p-value 0.05 across the 6 samples?

```
de_p <- as.data.frame(detectionP(rgSet))
colnames(de_p) <- sampnames
## large p proportion
de_p_prop <- sort(apply(de_p, 2, function(x) {
    sum(x >= 0.05)/length(x)
}))
de_p_prop
```

```
##            3            1            4            6            2
## 0.0002100875 0.0002739376 0.0010154229 0.0011410635 0.0011678393
##            5
## 0.0014026430
```

```
de_p_prop[6]
```

```
##           5
## 0.001402643
```

```
names(de_p_prop)[6]
```

```
## [1] "5"
```

```
## average p
n_de_p <- sum(rowMeans(de_p) >= 0.05)
n_de_p
```

```
## [1] 853
```

```
nrow(de_p)
```

```
## [1] 485512
```

The detection p-values are calculated as $p = 1 - \phi[(x - \mu_{neg})/\sigma_{neg}]$, where ϕ is the normal density function, x is the sum of two beads (Type I) or sum of two color intensities (Tpye II), the ϕ takes the Z score as the input. The sample with id 5 has the largest percentage of detection pvalues >= 0.05, 0.0014026. 853 probes have average detection p-value >= 0.05 across the 6 samples.

1.7 2nd Introduction

Save the methylation signals using the preProcess series of functions for the Raw data (without normalization) and the SWAN normalization method.

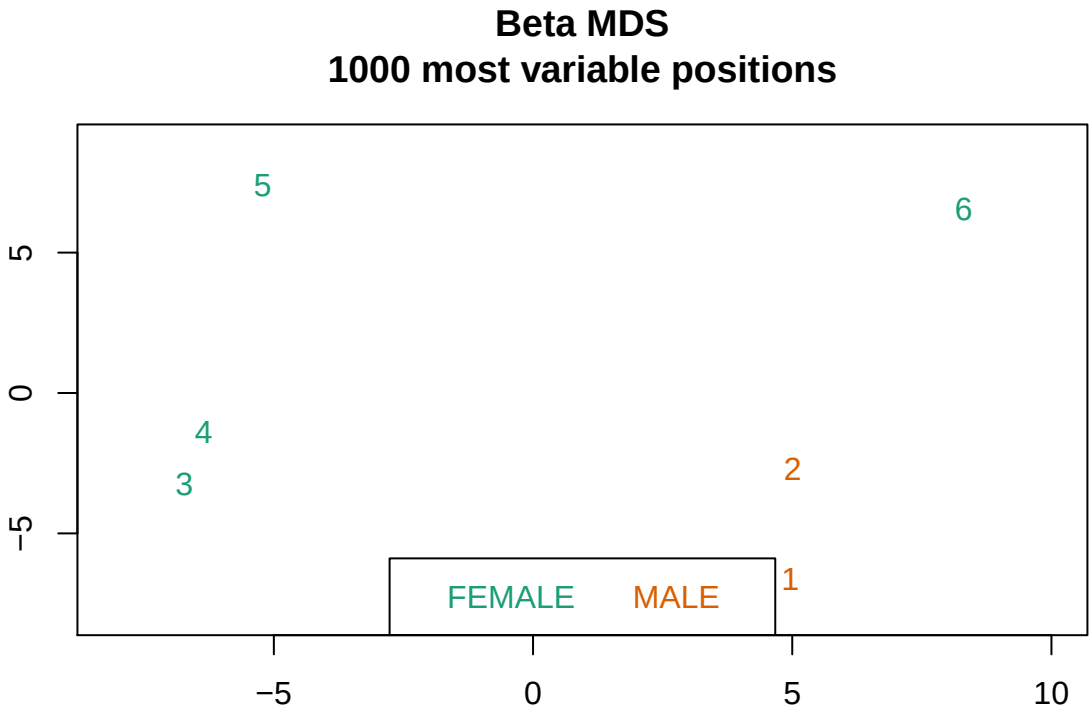
mset <- preprocessRaw(rgSet)

msetSWAN <- preprocessSWAN(rgSet)

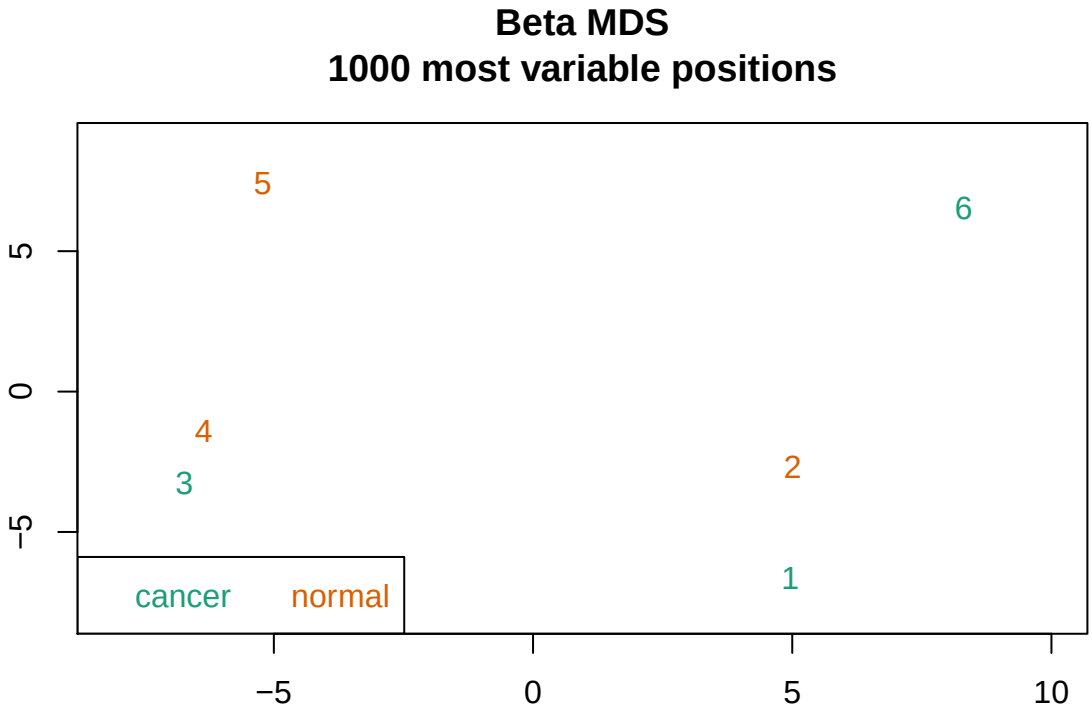
```
# swan normalization
mset <- preprocessRaw(rgSet)
msetSWAN <- preprocessSWAN(rgSet)
## cancer status
cancer.status <- pData(rgSet)$Status
## multidimensional scaling (MDS) plots
paste("By sex or by cancer status")
```

```
## [1] "By sex or by cancer status"
```

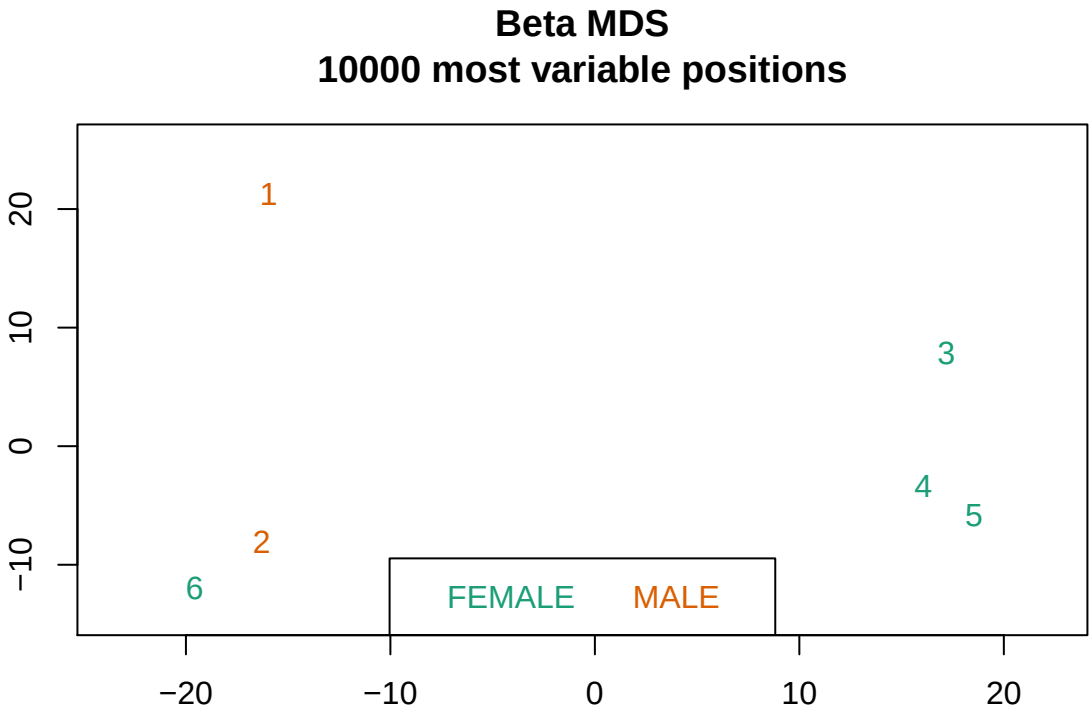
```
mdsPlot(msetSWAN, numPositions = 1000, sampGroups = groups_sex,
    sampNames = sampnames, legendPos = "bottom")
```



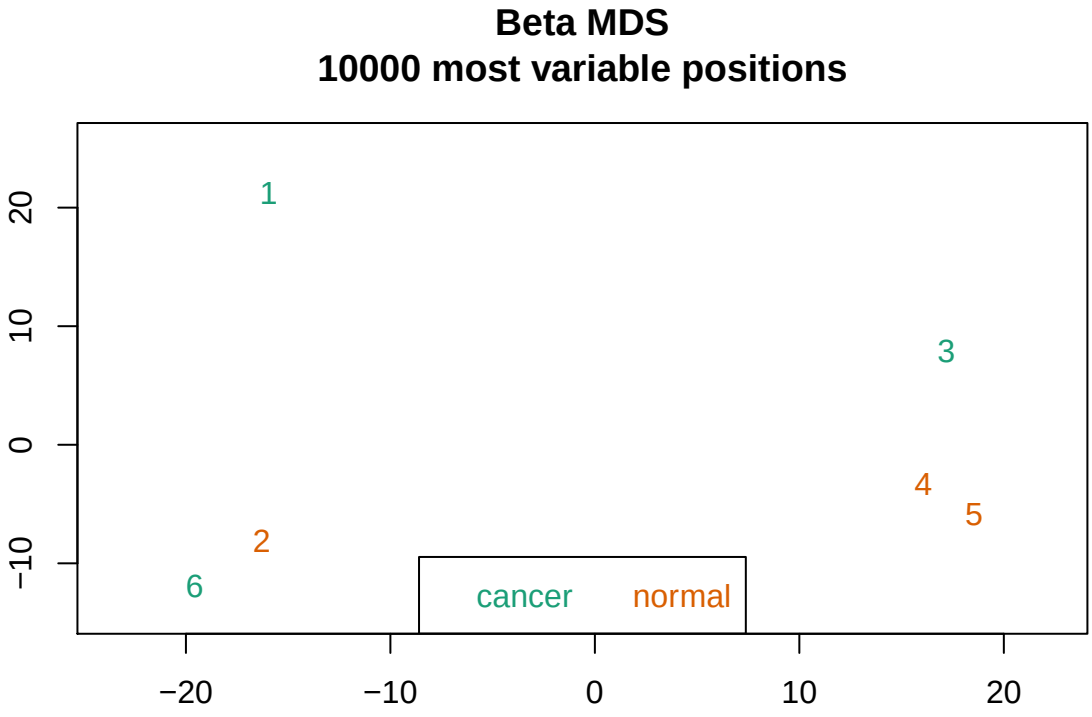

```
mdsPlot(msetSWAN, numPositions = 1000, sampGroups = cancer.status,
    sampNames = sampnames, legendPos = "bottomleft")
```




```
mdsPlot(msetSWAN, numPositions = 10000, sampGroups = groups_sex,
    sampNames = sampnames, legendPos = "bottom")
```



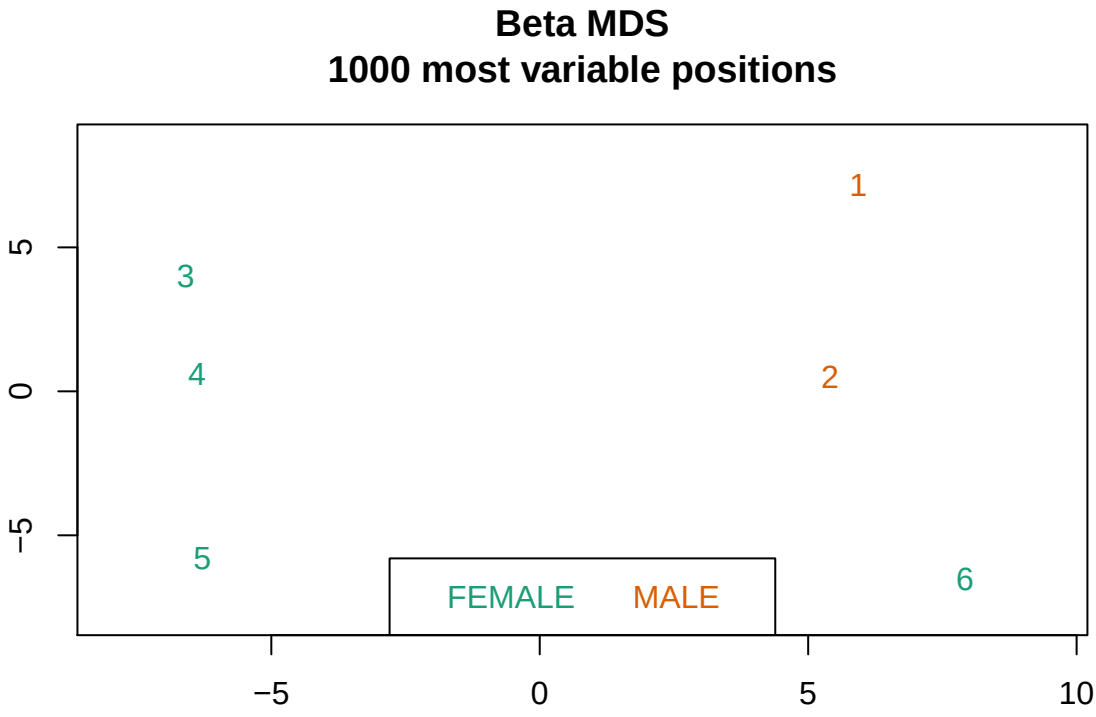

```
mdsPlot(msetSWAN, numPositions = 10000, sampGroups = cancer.status,
    sampNames = sampnames, legendPos = "bottom")
```



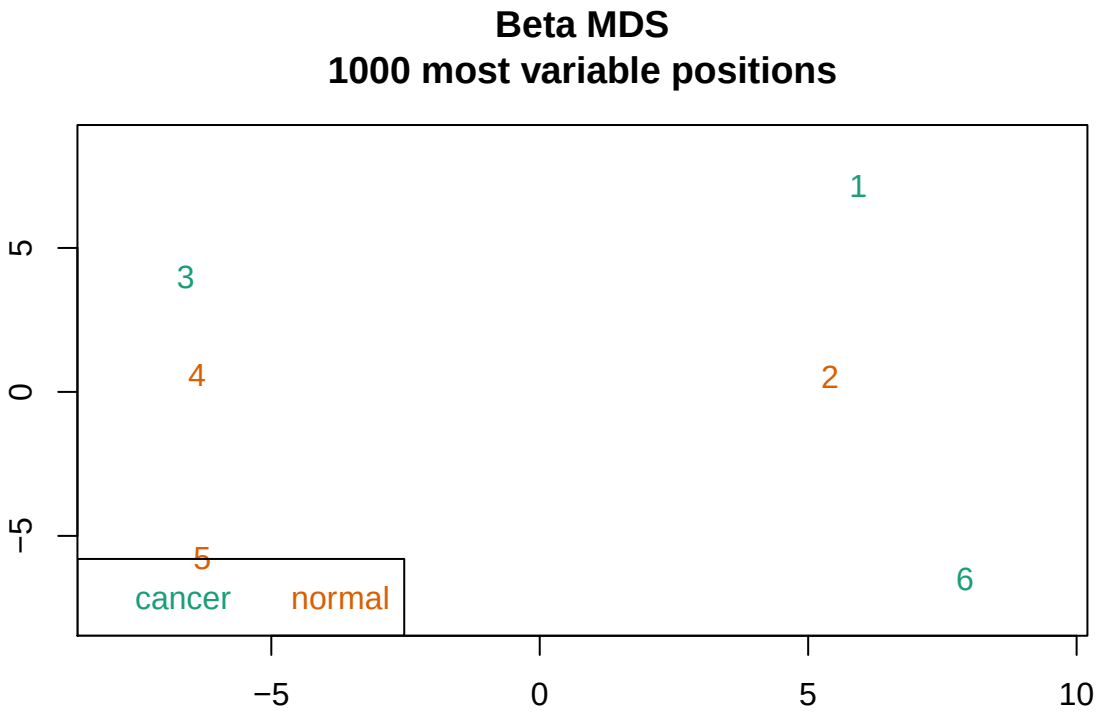

```
paste("Raw Data")
```

```
## [1] "Raw Data"
```

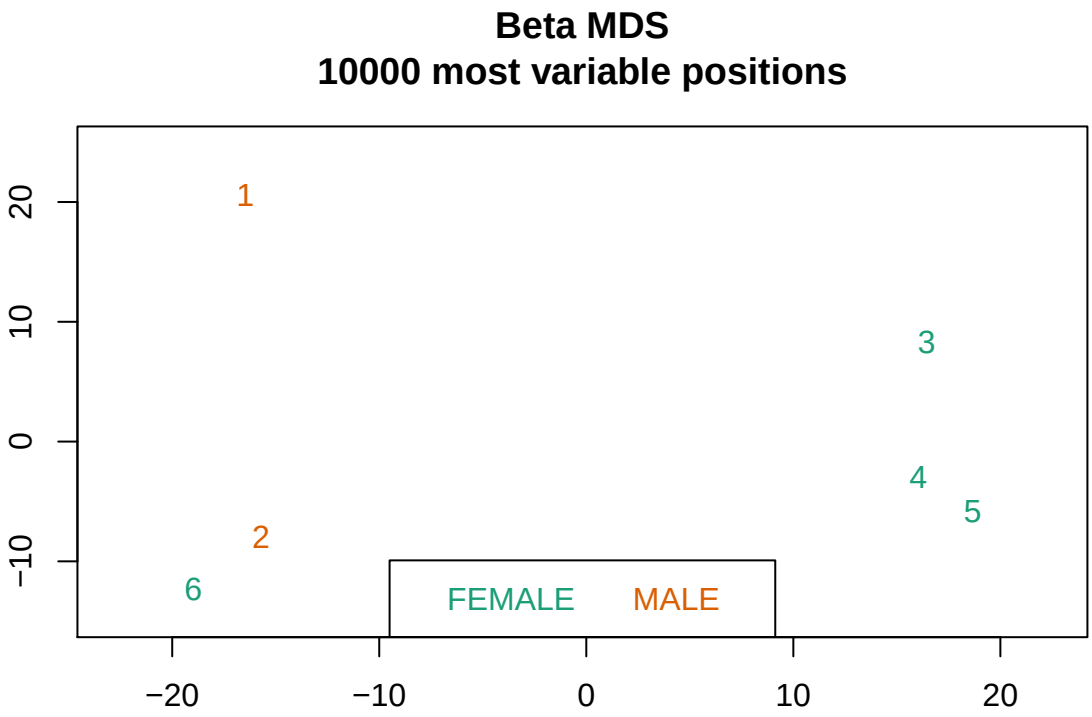
```
mdsPlot(mset, numPositions = 1000, sampGroups = groups_sex, sampNames = sampnames,
    legendPos = "bottom")
```



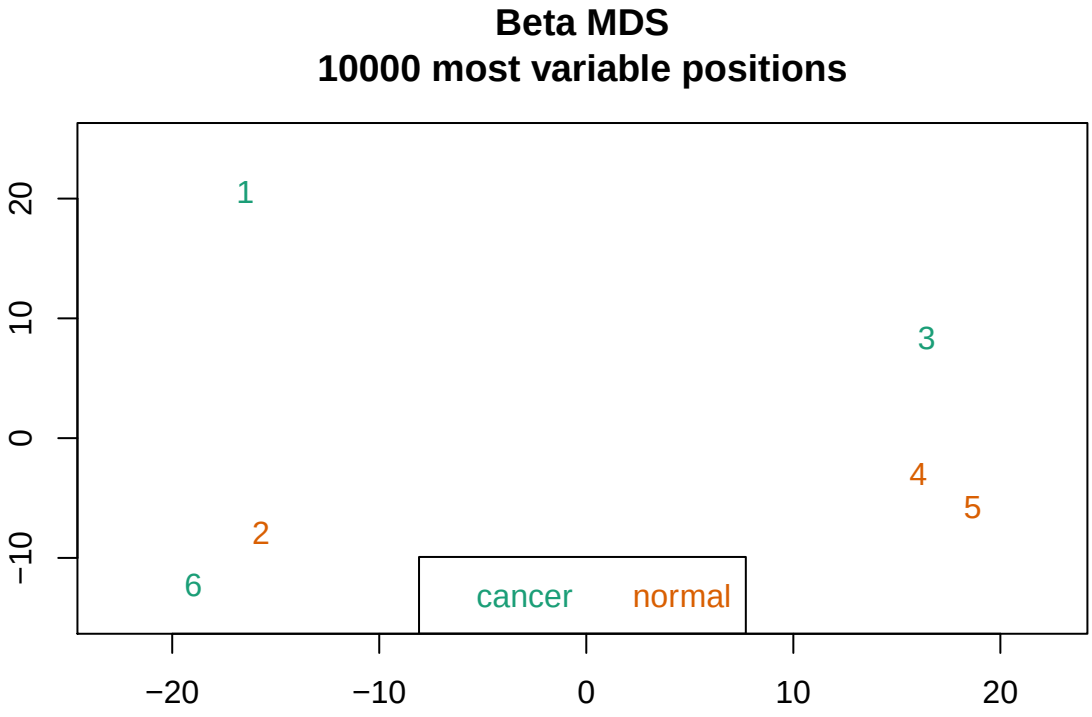

```
mdsPlot(mset, numPositions = 1000, sampGroups = cancer.status,
    sampNames = sampnames, legendPos = "bottomleft")
```



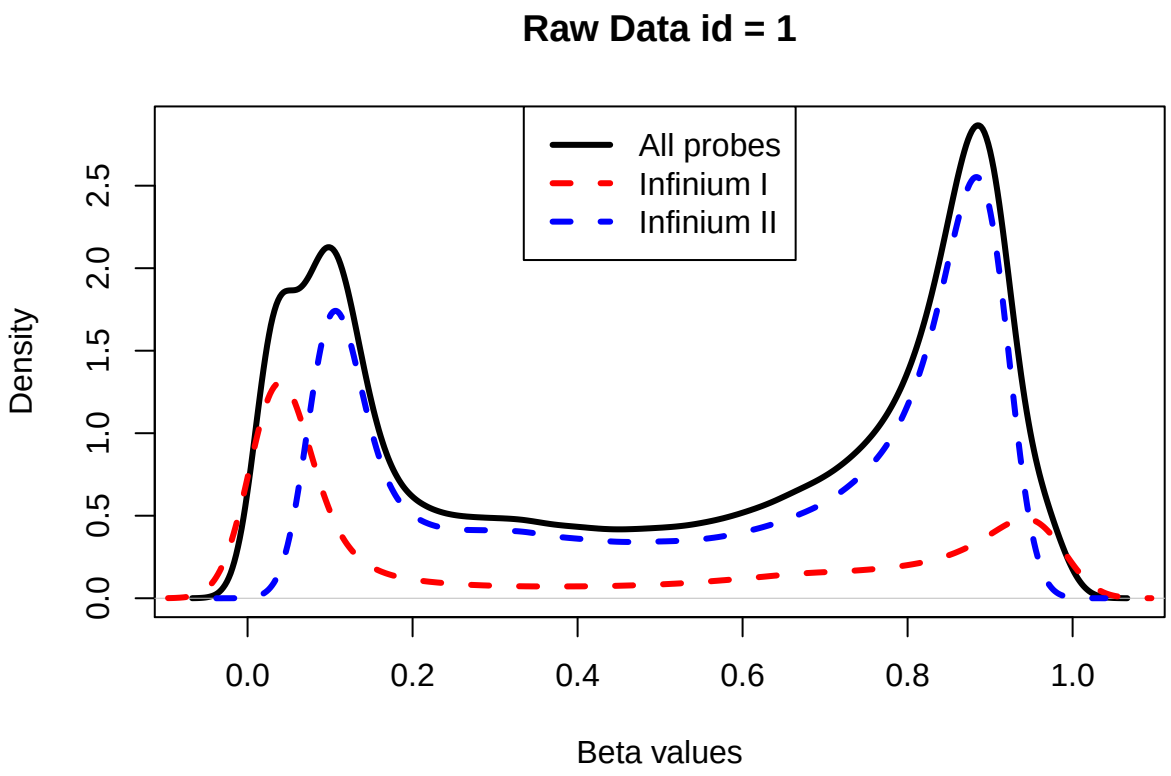

```
mdsPlot(mset, numPositions = 10000, sampGroups = groups_sex,
    sampNames = sampnames, legendPos = "bottom")
```



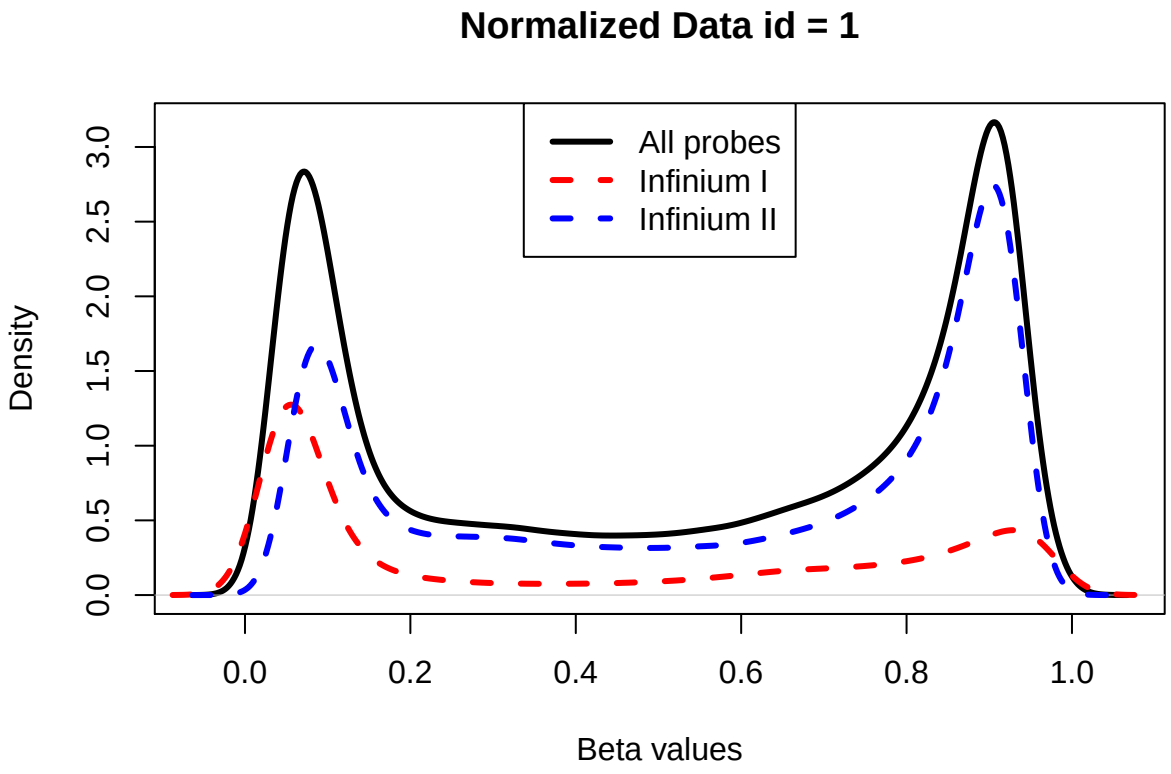

```
mdsPlot(mset, numPositions = 10000, sampGroups = cancer.status,
    sampNames = sampnames, legendPos = "bottom")
```



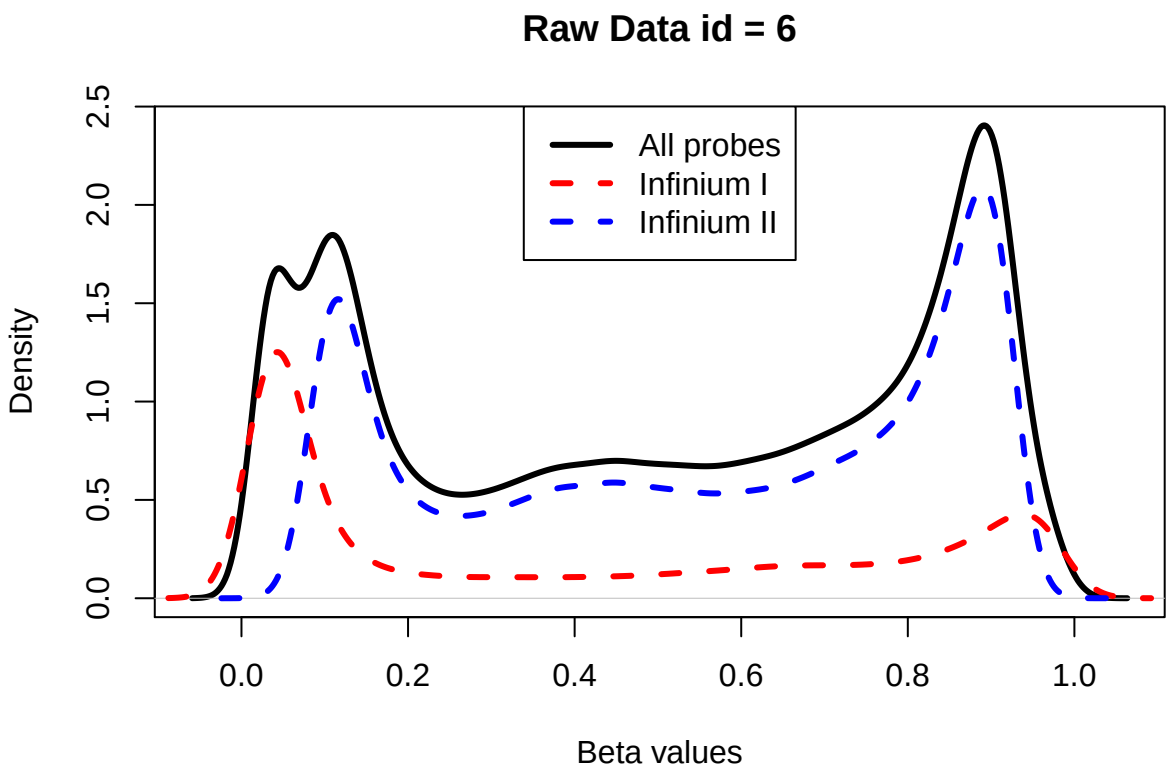

```
##
plotBetasByType(mset[, 1], main = "Raw Data id = 1")
```



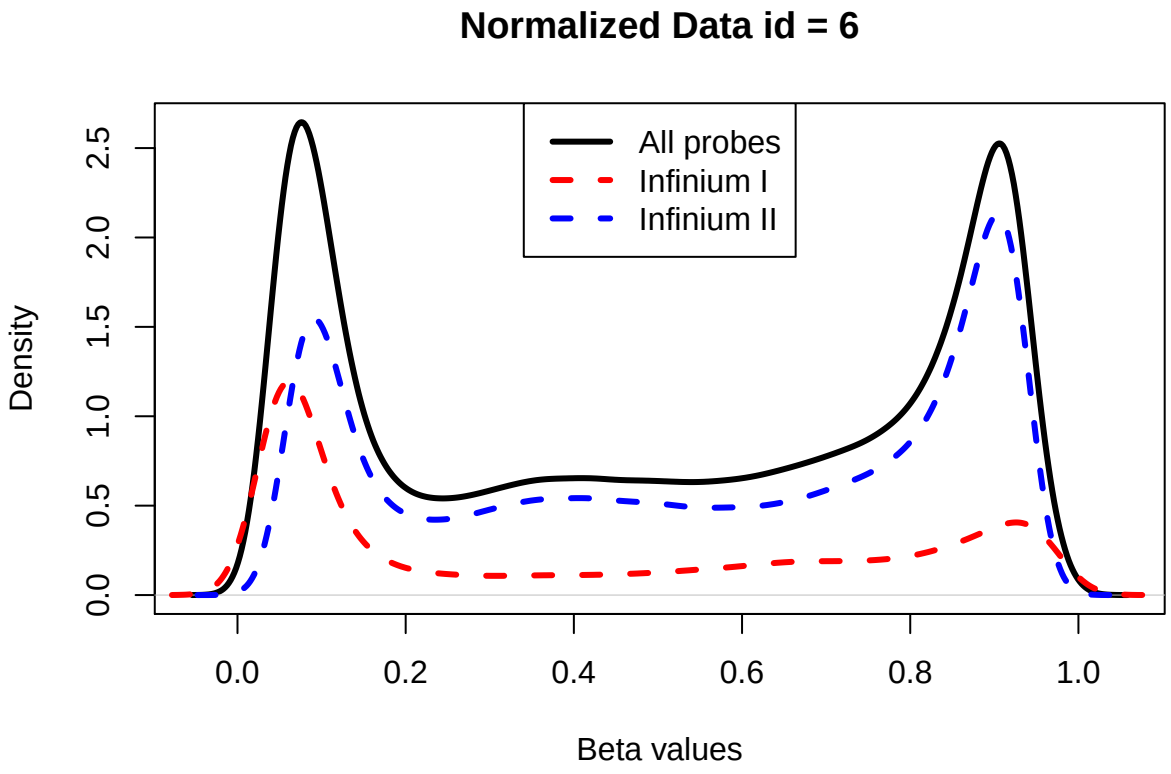

```
plotBetasByType(msetSWAN[, 1], main = "Normalized Data id = 1")
```




```
plotBetasByType(mset[, 6], main = "Raw Data id = 6")
```




```
plotBetasByType(msetSWAN[, 6], main = "Normalized Data id = 6")
```



1.8 (f) Use multidimensional scaling (MDS) plots to show how samples group by sex or cancer status with mdsPlot(). What do you conclude? Are conclusions different if you take more positions with the most methylation variability (1000 vs 10000 positions)? or by using the raw data mset compared to the SWAN normalized data msetSWAN?

With SWAN normalization data, the samples can be grouped by gender (linear separation) based on 1000 positions. However, the methylation data cannot separate the samples correctly as different tissue types. Compared to the 10000 positions MDS plot, the 1000 positions data gives a better separation. I cannot get the same conclusion from the 10000 positions MDS plot. By using the raw data mset, the conclusion remains the same as the normalized data.

1.9 (g) Plot the distribution of beta values before and after SWAN normalization using plotBetasByType(). What do you see in the density plots?

After SWAN normalization, for example, sample 1 and sample 6 here, the peaks at unmethylation region and methylation region of all probes become equally weighted. The unmethylation peaks come from type I and type II probes are merged together afther SWAN normalization.

2 2. DNA Methylation Annotation and Differentially Methylated Positions (Illumina 450K)

• Continuing with the the data from problem #1, get genome annotation information using the following code:

gset <-mapToGenome(msetSWAN)

annotation <-getAnnotation(gset)

```
# map cpg to genome
gset <- mapToGenome(msetSWAN)
annotation <- getAnnotation(gset)
levels(as.factor(annotation$Relation_to_Island))
```

```
## [1] "Island"  "N_Shelf" "N_Shore" "OpenSea" "S_Shelf" "S_Shore"
```

```
## in each region
cpg_island <- sum(annotation$Relation_to_Island == "Island")
cpg_Shelf <- sum(annotation$Relation_to_Island == "N_Shelf" |
    annotation$Relation_to_Island == "S_Shelf")
cpg_Shore <- sum(annotation$Relation_to_Island == "N_Shore" |
    annotation$Relation_to_Island == "S_Shore")
cpg_OpenSea <- sum(annotation$Relation_to_Island == "OpenSea")

cpg_island
```

```
## [1] 150254
```

```
cpg_Shelf
```

```
## [1] 47144
```

```
cpg_Shore
```

```
## [1] 112067
```

```
cpg_OpenSea
```

```
## [1] 176047
```

2.1 (a) What are CpG islands, shores, shelves and open seas? From annotation() how many CpG site probes are in each of these types?

CpG island is defined by the gene region G+C content > 0.50, CpG dinucleotide ratio > 0.60 and has > 200bp window, 40-50% genes have CpG islands in promoters. CpG shores are gene sequences within 2000bp region to the left or right of the CpG island (Up to 2kb from CpG island). CpG shelfs are gene sequences within 2000bp region to the left or right of the CpG shores (2-4 kb from CpG island). Open Seas are Isolated CpGs in the genome. There are 150254 CpG site probes in CpG islands. There are 47144 CpG site probes in CpG Shelf. There are 112067 CpG site probes in CpG Shore. There are 176047 CpG site probes in CpG OpenSea.

2.2 (b) Using the SWAN normalized data from problem #,1 msetSWAN, find differentially methylated positions (DMP) for cancer status with getM(), followed by dmpFinder() (which currently does not handle paired samples, so you will need to run it assuming independence). Are there any DMPs with q-value 0.10? Using a p-value cutoff of 10−5, how many DMPs show hyper or hypomethylation due to cancer status? Use plotCpg() to plot the beta values and then M-values for the top four DMPs. What do trends and effect sizes do you see in the plots?

```
# m values
mvalue <- getM(msetSWAN)

## cancer status
cancer.status <- pData(rgSet)$Status

## dmp by cancer
dmp_m_c <- dmpFinder(mvalue, pheno = cancer.status, type = c("categorical"))
head(dmp_m_c)
```

```
##            intercept         f         pval     qval
## cg12298140 -3.811887 1287.3071 3.601979e-06 0.375561
## cg13163765 -3.334239 1262.9968 3.741604e-06 0.375561
## cg03150279 -3.698391 1000.9703 5.948698e-06 0.375561
## cg08880423 -3.629930  987.6068 6.110224e-06 0.375561
## cg10033612 -3.282957  978.6019 6.222807e-06 0.375561
## cg14488592 -3.820052  969.2820 6.342636e-06 0.375561
```

```
head(dmp_m_c[order(dmp_m_c$pval, decreasing = F), ])
```

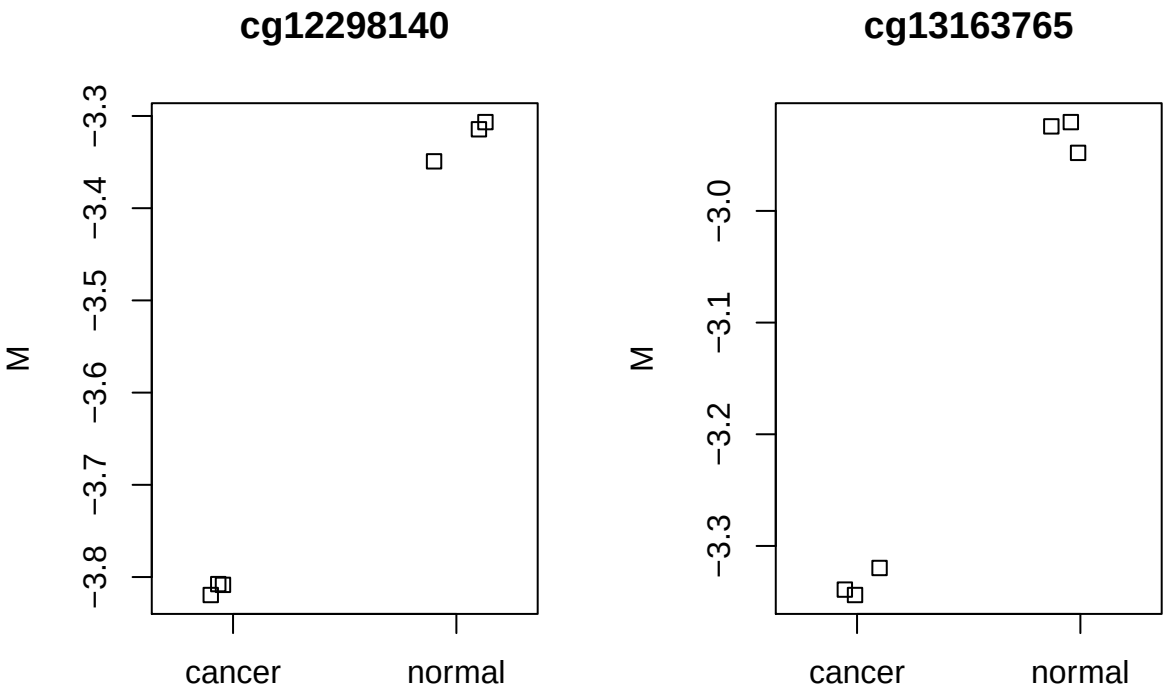
```
##            intercept         f         pval     qval
## cg12298140 -3.811887 1287.3071 3.601979e-06 0.375561
## cg13163765 -3.334239 1262.9968 3.741604e-06 0.375561
## cg03150279 -3.698391 1000.9703 5.948698e-06 0.375561
## cg08880423 -3.629930  987.6068 6.110224e-06 0.375561
## cg10033612 -3.282957  978.6019 6.222807e-06 0.375561
```

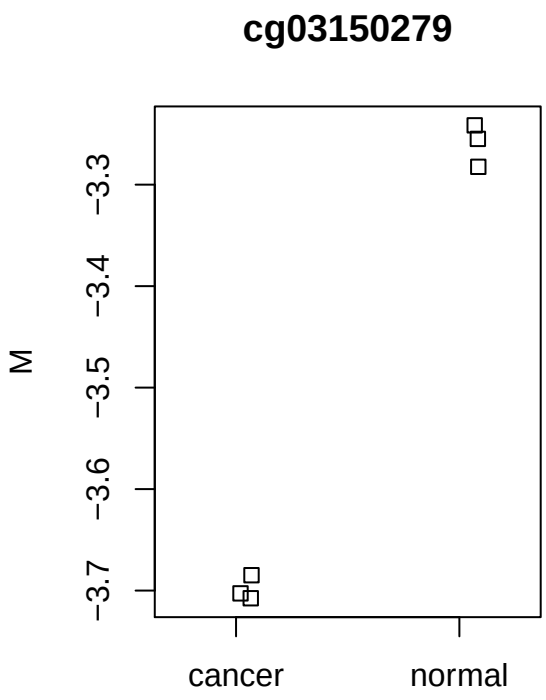
```
## cg14488592 -3.820052  969.2820 6.342636e-06 0.375561
```

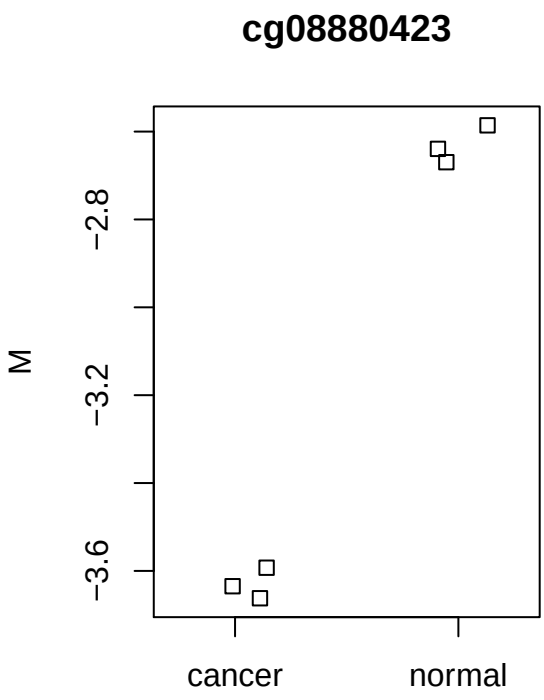
```
dmp_cancer <- sum(dmp_m_c$pval <= 1e-05)
dmp_cancer
```

```
## [1] 6
```

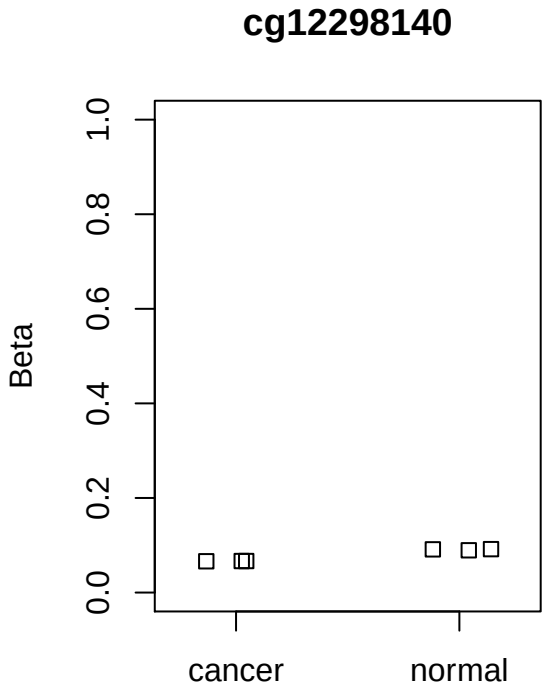
```
## plotCpg() to plot the beta values and then M-values for the
## top four DMPs
par(mfrow = c(1, 2))
for (i in rownames(dmp_m_c[order(dmp_m_c$pval, decreasing = F),
    ])[1:4]) {
    plotCpg(msetSWAN, cpg = i, pheno = cancer.status, type = c("categorical"),
        measure = c("M"))
}
```

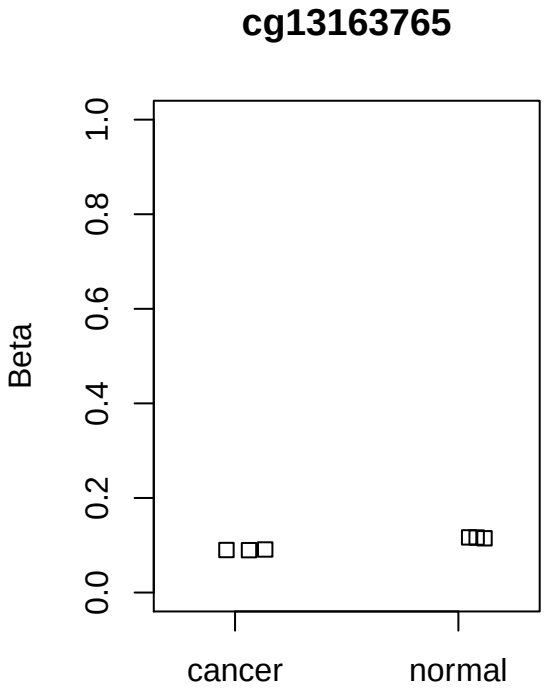


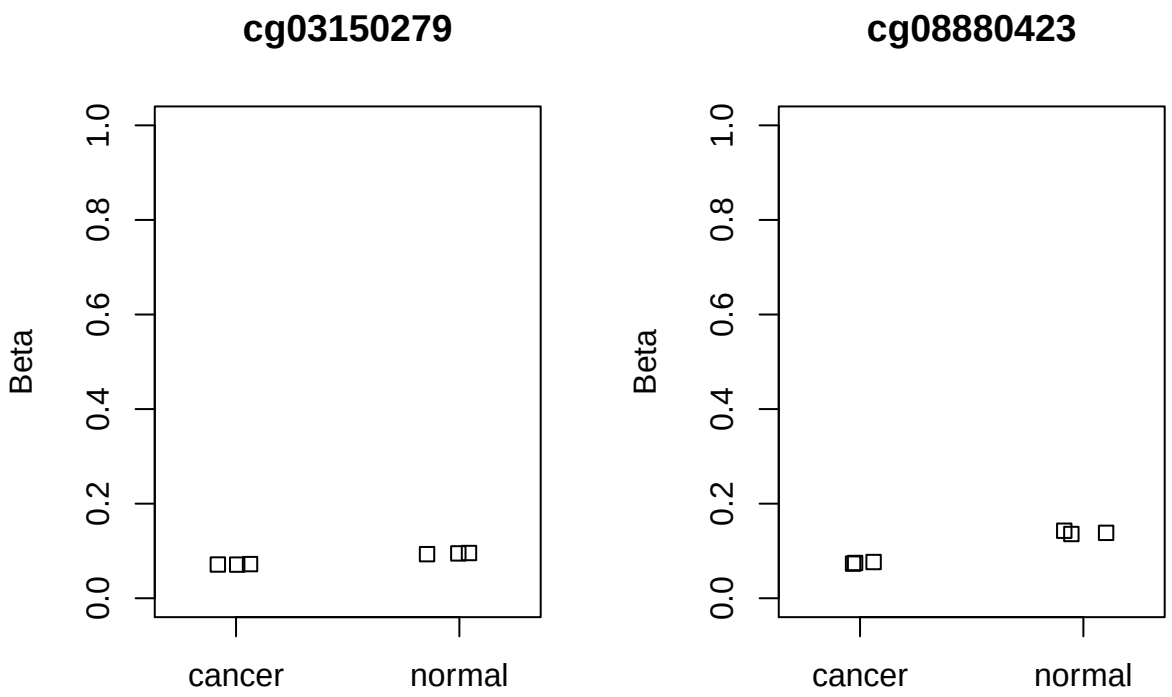





```
for (i in rownames(dmp_m_c[order(dmp_m_c$pval, decreasing = F),
    ])[1:4]) {
    plotCpg(msetSWAN, cpg = i, pheno = cancer.status, type = c("categorical"),
        measure = c("beta"))
}
```








```
## dmp by gender
dmp_m_g <- dmpFinder(mvalue, pheno = groups_sex, type = c("categorical"))
head(dmp_m_g)
```

```
##            intercept        f         pval      qval
## cg10013343 -3.529859 1281.176 3.636446e-06 0.2209973
## cg15479068 -2.780269 1227.899 3.957960e-06 0.2209973
## cg20648899 -1.640258 1186.213 4.240225e-06 0.2209973
## cg17220960  2.645867 1140.153 4.588699e-06 0.2209973
## cg13916877  3.100460 1126.132 4.703333e-06 0.2209973
## cg10492999  1.909465 1067.044 5.236948e-06 0.2209973
```

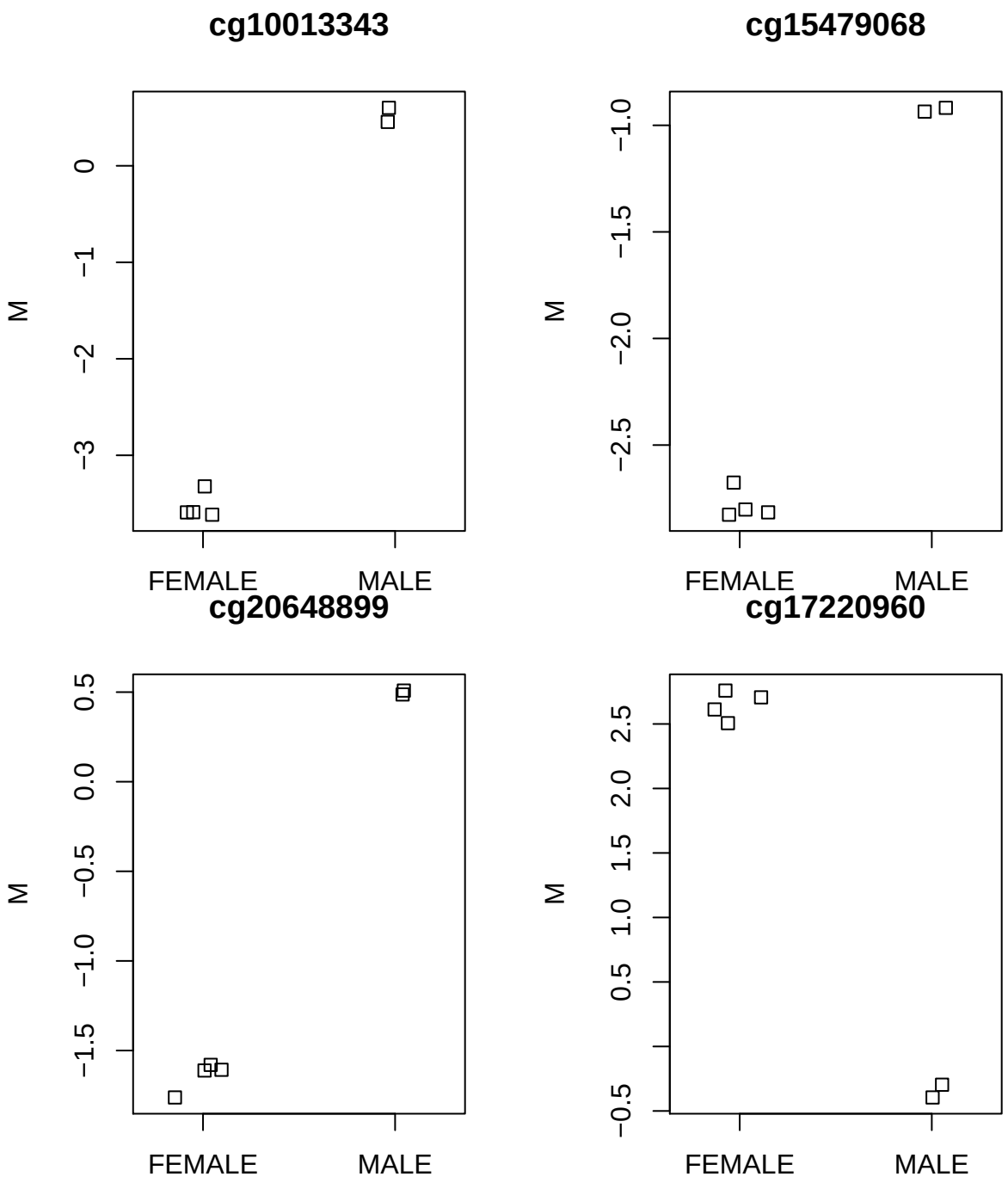
```
head(dmp_m_g[order(dmp_m_g$pval, decreasing = F), ])
```

```
##            intercept        f         pval      qval
## cg10013343 -3.529859 1281.176 3.636446e-06 0.2209973
## cg15479068 -2.780269 1227.899 3.957960e-06 0.2209973
## cg20648899 -1.640258 1186.213 4.240225e-06 0.2209973
## cg17220960  2.645867 1140.153 4.588699e-06 0.2209973
## cg13916877  3.100460 1126.132 4.703333e-06 0.2209973
## cg10492999  1.909465 1067.044 5.236948e-06 0.2209973
```

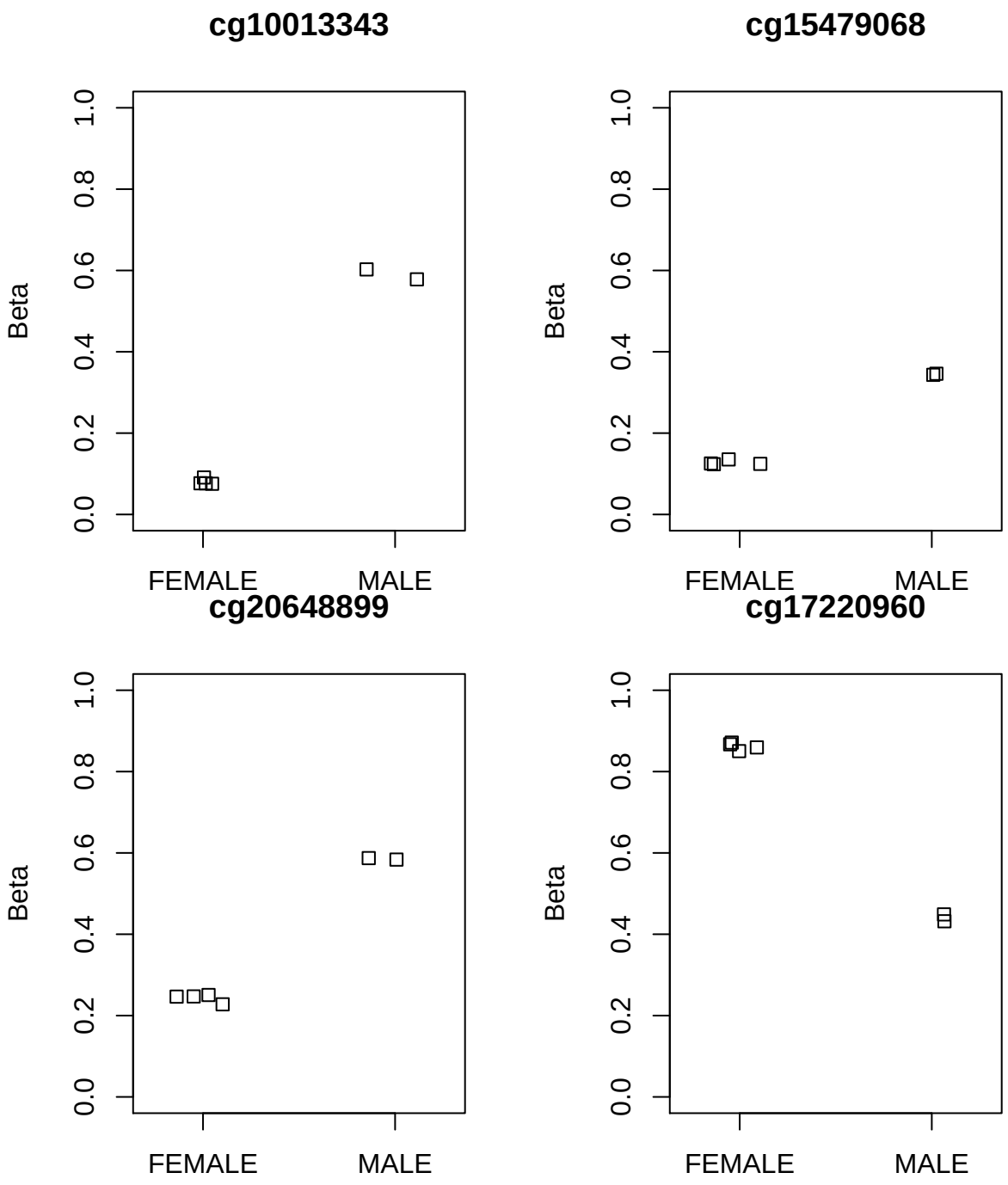
```
dmp_gender <- sum(dmp_m_g$pval <= 1e-05)
dmp_gender
```

```
## [1] 15
```

```
for (i in rownames(dmp_m_g[order(dmp_m_g$pval, decreasing = F),
    ])[1:4]) {
    plotCpg(msetSWAN, cpg = i, pheno = groups_sex, type = c("categorical"),
        measure = c("M"))
}
```




```
for (i in rownames(dmp_m_g[order(dmp_m_g$pval, decreasing = F),
    ])[1:4]) {
    plotCpg(msetSWAN, cpg = i, pheno = groups_sex, type = c("categorical"),
        measure = c("beta"))
}
```



There are no DMPs with q-value 0.10. With a p-value cutoff of 0.00001, there are 6 DMPs show hyper or hypomethylation due to cancer status.

M values:
By the top 4 CpGs, the cancer group is hypomethylated, since the cancer samples have smaller M values. However, the difference between cancer sample and normal sample on the individual CpG level is small, around 0.4 to 1. The Beta values are more close and less obvious, but the trends are the same as the M values.

2.3 (c) Repeat part b) but for DMPs between male and females.

There are no DMPs with q-value 0.10. With a p-value cutoff of 0.00001, there are 15 DMPs show hyper or hypomethylation due to gender.

M values:
By the top 4 CpGs, 3 Female samples are hypomethylated, and one Female sample is hypermethylated. The differences by gender are larger compared with differences by cancer status, generally. To be specific, the differences of top4-p-value CpGs range from 1 to 3. Since here we have greater effect sizes, the trend and effect sizes in Beta values are also obvious.

2.4 (d) Global methylation profiles vary by sex. There is a function addSex() to estimate whether each sample is male or female. Are the predicted and given labels correct for Sex? If not, revisit the MDS plot from part 1e)? Do the new predictions group in the plot? Also repeat the analysis in 2c). Now are there DMPs with q-value 0.10 (or p-value 10−5)?

```
gset = addSex(gset)
cbind(pData(gset)$predictedSex, pData(gset)$Sex)
```

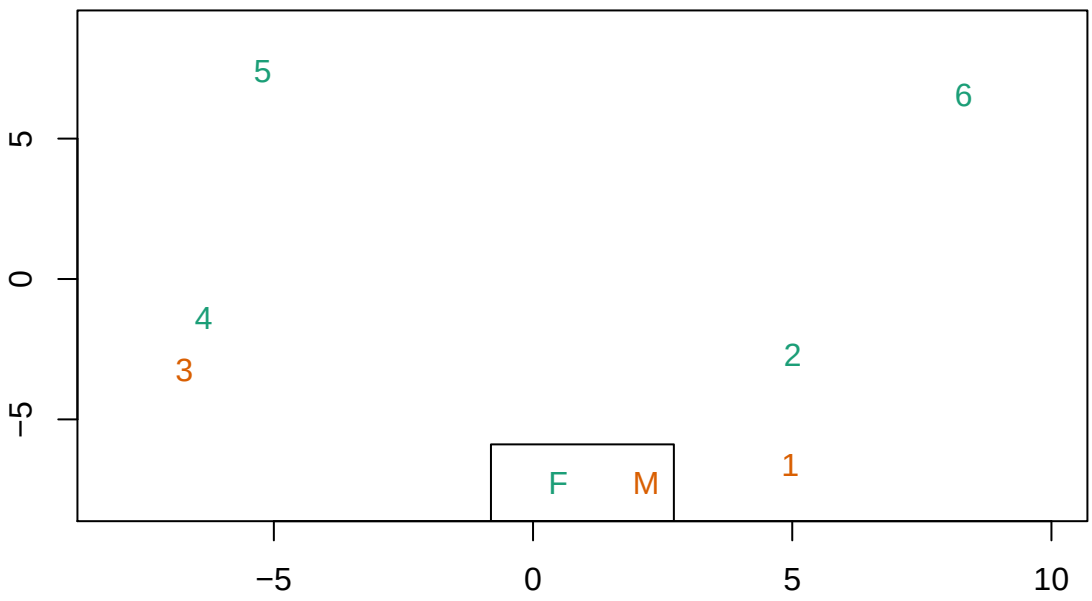
```
##      [,1] [,2]
## [1,] "M"  "MALE"
## [2,] "F"  "MALE"
## [3,] "M"  "FEMALE"
## [4,] "F"  "FEMALE"
## [5,] "F"  "FEMALE"
## [6,] "F"  "FEMALE"
```

```
# mds plot by predictedsex
pre_sex <- pData(gset)$predictedSex

paste("By sex or by cancer status")
```

```
## [1] "By sex or by cancer status"
```

```
mdsPlot(msetSWAN, numPositions = 1000, sampGroups = pre_sex,
    sampNames = sampnames, legendPos = "bottom")
```



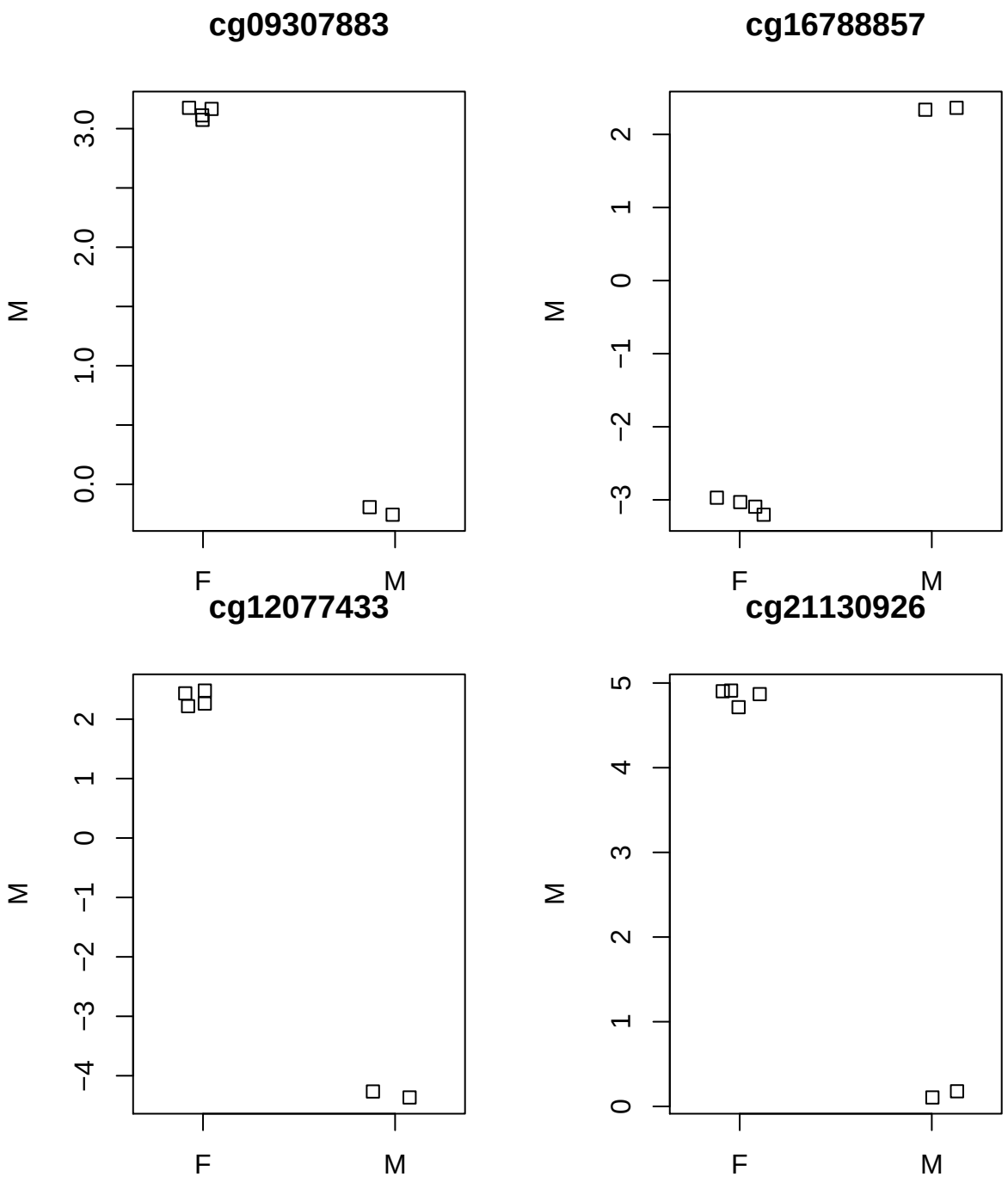

```
## DMP by predicted sex
dmp_pre_sex <- dmpFinder(mvalue, pheno = pre_sex, type = c("categorical"))
head(dmp_pre_sex)
```

```
##            intercept        f         pval       qval
## cg09307883  3.132003 6800.158 1.296247e-07 0.02314273
## cg16788857 -3.074459 5230.012 2.190748e-07 0.02314273
## cg12077433  2.348988 4369.892 3.137239e-07 0.02314273
## cg21130926  4.849165 4241.975 3.329146e-07 0.02314273
## cg07889003  2.402439 4010.672 3.723878e-07 0.02314273
## cg12186981  3.701023 3933.546 3.871213e-07 0.02314273
```

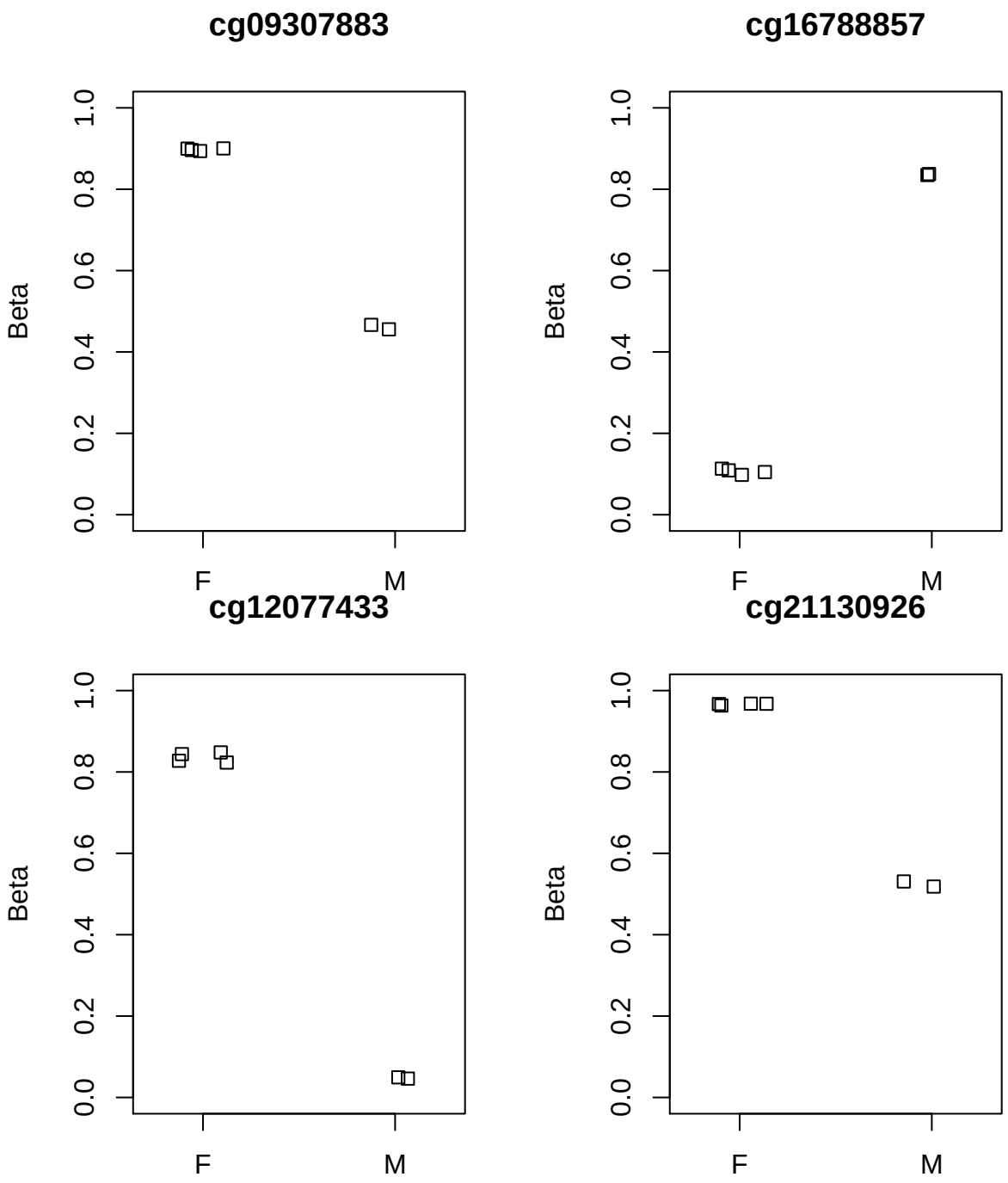
```
n_pre_sex <- sum(dmp_pre_sex$qval <= 0.1)
n_pre_sex
```

```
## [1] 65
```

```
## plots
par(mfrow = c(1, 2))
for (i in rownames(dmp_pre_sex[order(dmp_pre_sex$pval, decreasing = F),
    ])[1:4]) {
    plotCpg(msetSWAN, cpg = i, pheno = pre_sex, type = c("categorical"),
        measure = c("M"))
}
```




```
for (i in rownames(dmp_pre_sex[order(dmp_pre_sex$pval, decreasing = F),
    ])[1:4]) {
    plotCpg(msetSWAN, cpg = i, pheno = pre_sex, type = c("categorical"),
        measure = c("beta"))
}
```



The sex was not predicted correctly. The new gender predictions can be grouped in the mdsplot, and are linear separable. In other words, we can draw a straight roughly horizontal line to separate samples by gender. However, since the sample size is small, the clustering is not good. With new predicted gender, there are 65 CpGs with q-value 0.10. With predicted gender, the trends and effect sizes in M values and Beta values change a lot.

2.5 (e) This sample data set is too small for bumphunter to identify significant regions by performing permutations or bootstrap. However, we can use the getSegment() function to find regions of extreme values for the differences found in part b).

```
diffs <- dmp_m_c$intercept  #NOTE: dmp_m_c is where you saved results from part b)
chr <- annotation$chr
pos <- annotation$pos
cl <- clusterMaker(chr, pos, maxGap = 300)  #cluster probes
# Find regions with a stretch of differences
segs <- getSegments(diffs, f = cl, cutoff = 6)
# To plot one of the regions
ind = segs$dnIndex[[1]]
segs$dnIndex
```

```
## [[1]]
## [1] 27430
##
## [[2]]
## [1] 29918
##
## [[3]]
## [1] 48183
##
## [[4]]
## [1] 53164
##
## [[5]]
## [1] 97337
##
## [[6]]
## [1] 137928
```

```
length(segs$dnIndex)
```

```
## [1] 6
```

```
ind
```

```
## [1] 27430
```

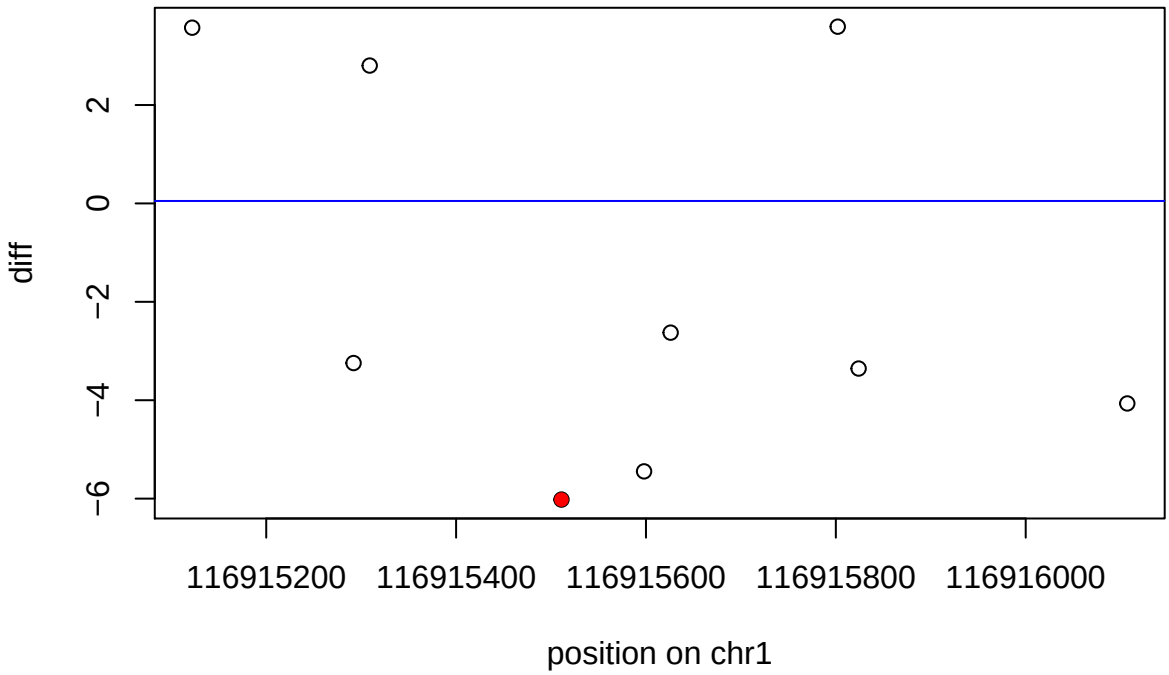
```
index <- which(cl == cl[ind[1]])
ind[1]
```

```
## [1] 27430
```

```
index
```

```
## [1] 27427 27428 27429 27430 27431 27432 27433 27434 27435
```

```
plot(pos[index], diffs[index], xlab = paste("position on", chr[ind[1]]),
    ylab = "diff")
points(pos[ind], diffs[ind], pch = 16, col = 2)
abline(h = 0.05, col = "blue")
```



The getSegments() function identified 6 regions with extreme values. Here I plotted the 1st region. In this region, the cancer group is hypomethylated with M value is smaller by 6. Cancer group is hypomethylated shown by the above plotCpg() function. I also noticed, the results from this function is not reproducible.