DNA Methylation

Contents

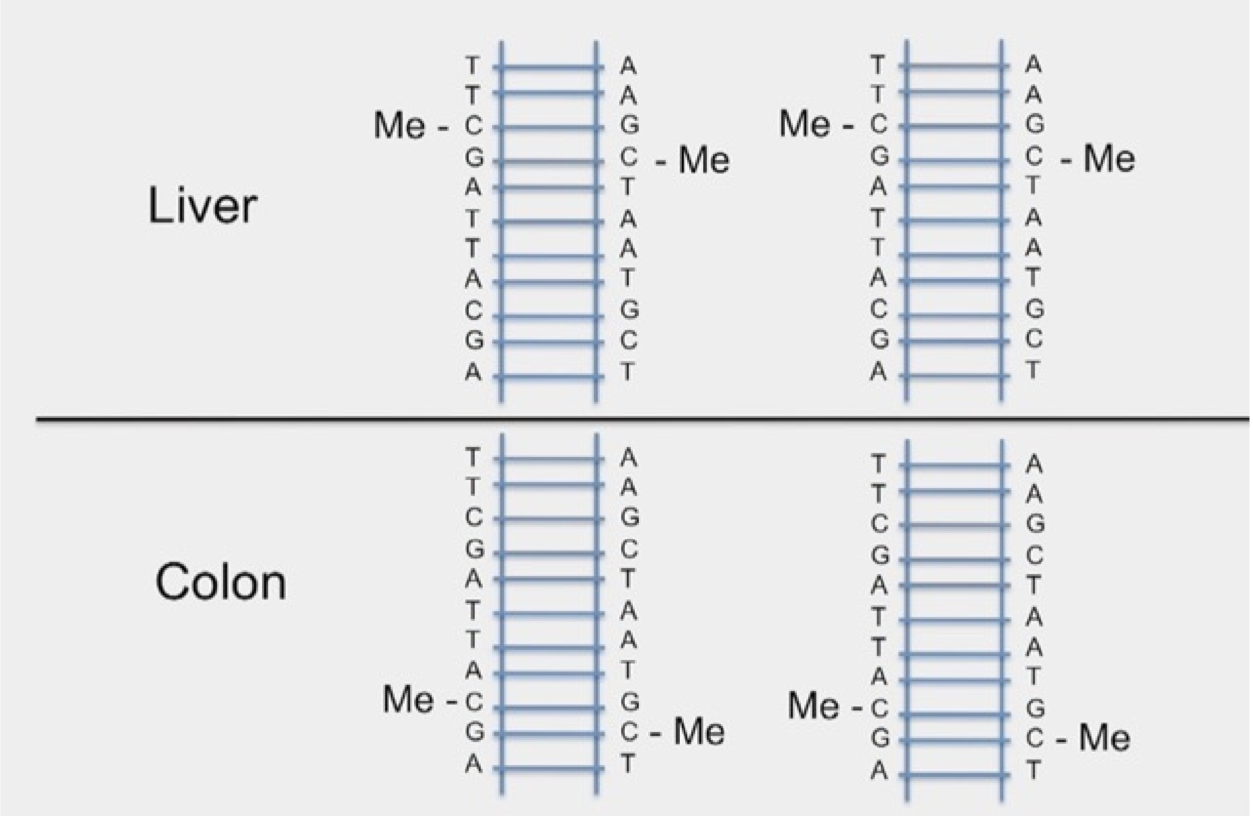
1 Introduction

Epigenetics: Epigenetics is the study of stable phenotypic changes that do not involve alterations in the DNA sequence. Epigenetics most often involves changes that affect gene activity and expression.

Techniques used to study epigenetics:

- ChIP-Seq: A method used to analyze protein interactions with DNA. ChIP-seq combines chromatin immunoprecipitation (ChIP) with massively parallel DNA sequencing to identify the binding sites of DNA-associated proteins.
- MeDIP-Seq: Methylated DNA immunoprecipitation (MeDIP) enables researchers to examine genome-wide changes in DNA methylation patterns.
- ATAC-Seq: The assay for transposase-accessible chromatin with sequencing (ATAC-Seq) is a popular method for determining chromatin accessibility across the genome. By sequencing regions of open chromatin, ATAC-Seq can help you uncover how chromatin packaging and other factors affect gene expression.

Methylation: CpG segments (from 5' to 3' end) are the hotspots for methyl groups to attach to, causing methylated DNA. When the DNA replicates, this methylation characteristic preserves. This methylation is different across different cells and is an epigenetic.



Let's count the CpG segments in a part of human genome:

```
library(BSgenome.Hsapiens.UCSC.hg19)
chr22 <- Hsapiens[["chr22"]]
s <- subseq(chr22, start = 23456789, width = 1000)
s
```

```
## 1000-letter DNAString object
## seq: AGTCACTTGTGCCTGGGTGTGGGGACTAAGCTGTCC...CCTTCCTAGAACAGGAAGGTGGGGTGACCCTGCAGG
```

```r
# Count CpG segments
countPattern("CG", s)
```

```
## [1] 10
```

Focus: We use Bisulfite treatment to assess if a CpG is methylated or not. Bisulfite turns un-methylated CpGs to TG.

CpG Islands: Sometimes in the genome, we see segments which are big clusters of CpGs. These segments are called CpG islands. These islands tend to be close to the promoter of genes. The formal definition is:

- 200 base pairs
- GC-content > 50%
- obs / exp > 0.6

2 Access the CpG Islands in human genome

```r
# Load the data in AnnotationHub package
library(AnnotationHub)
ah <- AnnotationHub(localHub = FALSE)

# Subset to just the databases related to the hg19 genome
ah <- subset(ah, ah$genome == "hg19")

# Retrieve the annotations for CpG Islands
cgi <- ah[["AH5086"]]

# Extract the sequence of each CpG Island
library(BSgenome.Hsapiens.UCSC.hg19)
cgiseq <- getSeq(Hsapiens, cgi)

# Compute the proportion of Cs and Gs for each island
Cs <- letterFrequency(cgiseq, "C", as.prob=TRUE)
Gs <- letterFrequency(cgiseq, "G", as.prob=TRUE)

# Compute the proportion of CpGs we expect to see by chance
Exp <- array(Cs * Gs * width(cgiseq))

# Compute the proportion of CpGs we observe
Obs <- array(vcountPattern("CG", cgiseq))

# Compute the median of the observed to expected ratio
median(Obs / Exp)
```

```
## [1] 0.8316008
```

Note that the CpG observed to expected ratio is below 1 and that few islands actually surpass a ratio of 1 or more. However, for the rest of the genome, the observed to expected ratio is substantially smaller.

3 Differentially methylated regions (DMRs)

Now we will show an example of analyzing methylation data. We will use colon cancer data from TCGA. The data was created with the Illumina 450K array and has already been processed to create matrix with methylation measurements.

```r
library(S4Vectors)
library(coloncancermeth)
data(coloncancermeth)
```

We know have three tables one containing the methylation data, one with information about the samples or columns of the data matrix, and Granges object with the genomic location of the CpGs represetned in the rows of the data matrix.

```r
# Methylation data
meth[1:4, 1:4]
```

```
##                 [,1]       [,2]       [,3]       [,4]
## cg13869341 0.8018963 0.84820056 0.91330487 0.91118459
## cg14008030 0.6147812 0.61560110 0.63670777 0.64317777
## cg12045430 0.1143808 0.03904616 0.04136764 0.09394642
## cg20826792 0.2033524 0.17257738 0.16484079 0.23872555
```

```r
# Sample information
pd[1:4, 20:22]
```

```
## DataFrame with 4 rows and 3 columns
##           sample_type sample_type_id shortest_dimension
##           <character>      <integer>        <character>
## 1       Primary Tumor              1             0.4 cm
## 2 Solid Tissue Normal             11             0.4 cm
## 3 Solid Tissue Normal             11             0.6 cm
## 4       Primary Tumor              1             0.5 cm
```

```r
# Granges object
gr[1:4, ]
```

```
## GRanges object with 4 ranges and 0 metadata columns:
##              seqnames    ranges strand
##                 <Rle> <IRanges>  <Rle>
##   cg13869341     chr1     15865      *
##   cg14008030     chr1     18827      *
##   cg12045430     chr1     29407      *
##   cg20826792     chr1     29425      *
##   -------
##   seqinfo: 24 sequences from hg19 genome; no seqlengths
```

Check the number of cancer vs. non-cancer patients:

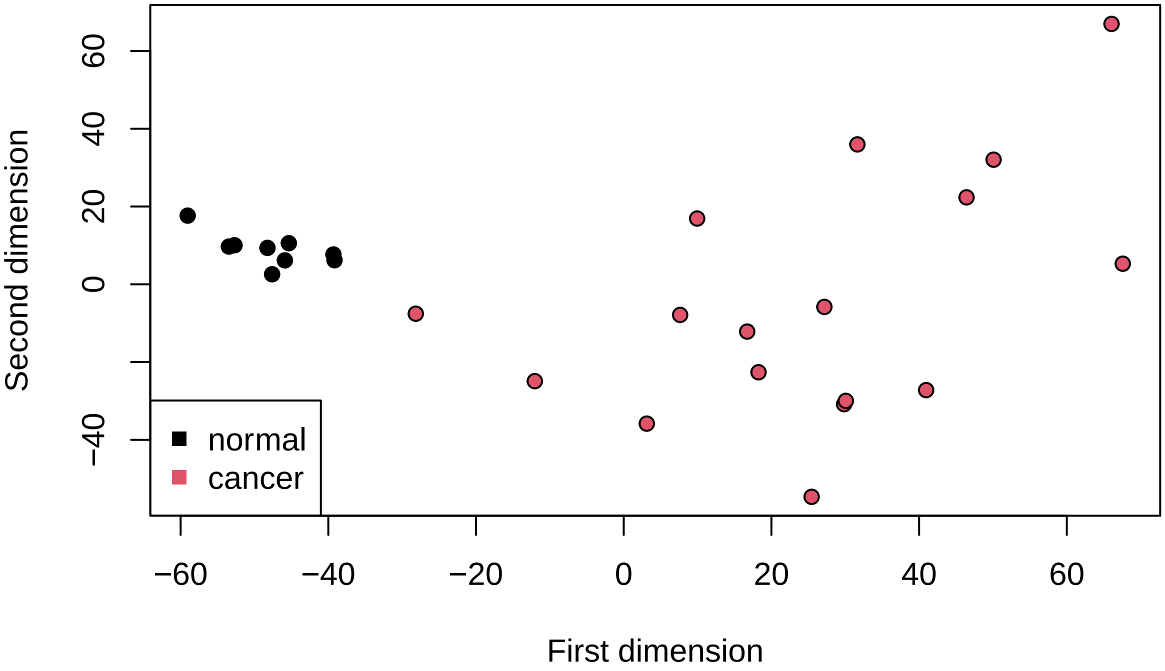
```r
table(pd$Status)
```

```
##
## normal cancer
##      9     17
```

```r
normalIndex <- which(pd$Status == "normal")
cancerlIndex <- which(pd$Status == "cancer")
```

Let's start by creating an MDS plot that graphically shows approximate distances between the samples:

```r
d <- dist(t(meth))
mds <- cmdscale(d)
plot(
  mds[, 1],
  mds[, 2],
  bg = as.numeric(pd$Status),
  pch = 21,
  xlab = "First dimension",
  ylab = "Second dimension"
)
legend("bottomleft",
       levels(pd$Status),
       col = seq(along = levels(pd$Status)),
       pch = 15
)
```

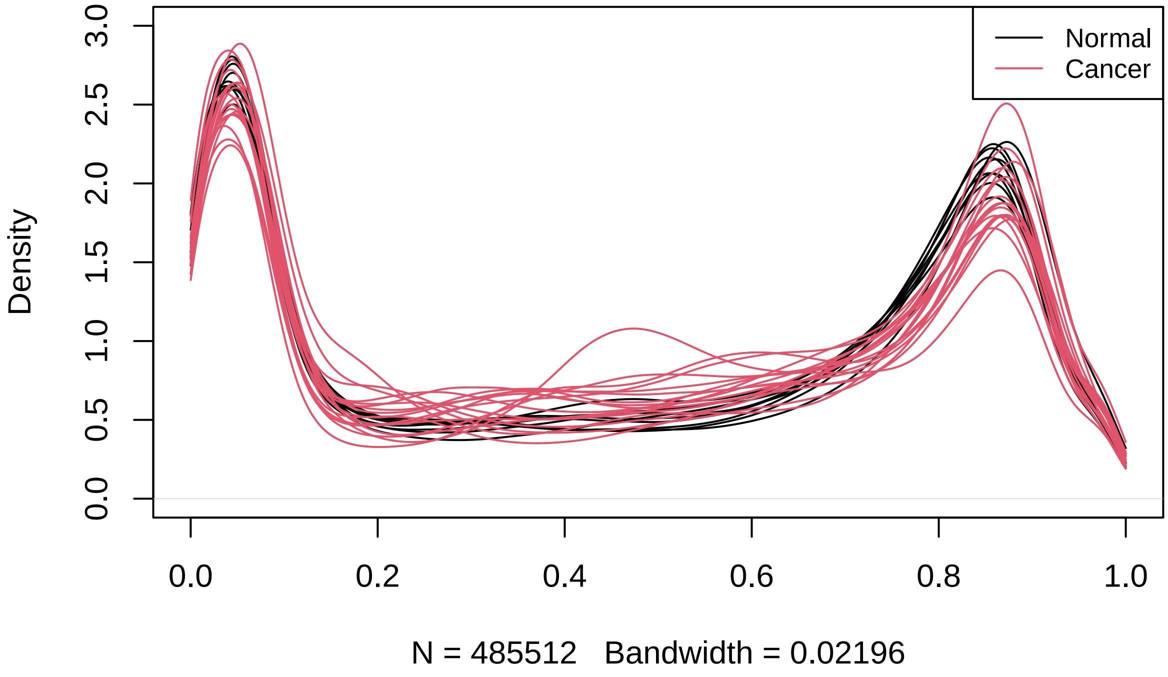


The MDS plot shows separation between cancer and normal samples, but only in the first dimension. The second dimension seems to be associated with a large variability within the cancers.

Now let's take a quick look at the distribution of methylation measurements for the samples:

```
i = normalIndex[1]
plot(
  density(meth[, i], from = 0, to = 1),
  main = "",
  ylim = c(0, 3),
  type = "n"
)

# Add the normal samples
for (i in normalIndex) {
  lines(density(meth[, i], from = 0, to = 1), col = 1)
}

# Add the cancer samples
for (i in cancerlIndex) {
  lines(density(meth[, i], from = 0, to = 1), col = 2)
}
```

```
legend(
  x = "topright",
  legend = c("Normal", "Cancer"),
  col = c(1, 2),
  lty = 1,
  cex = 0.8
)
```

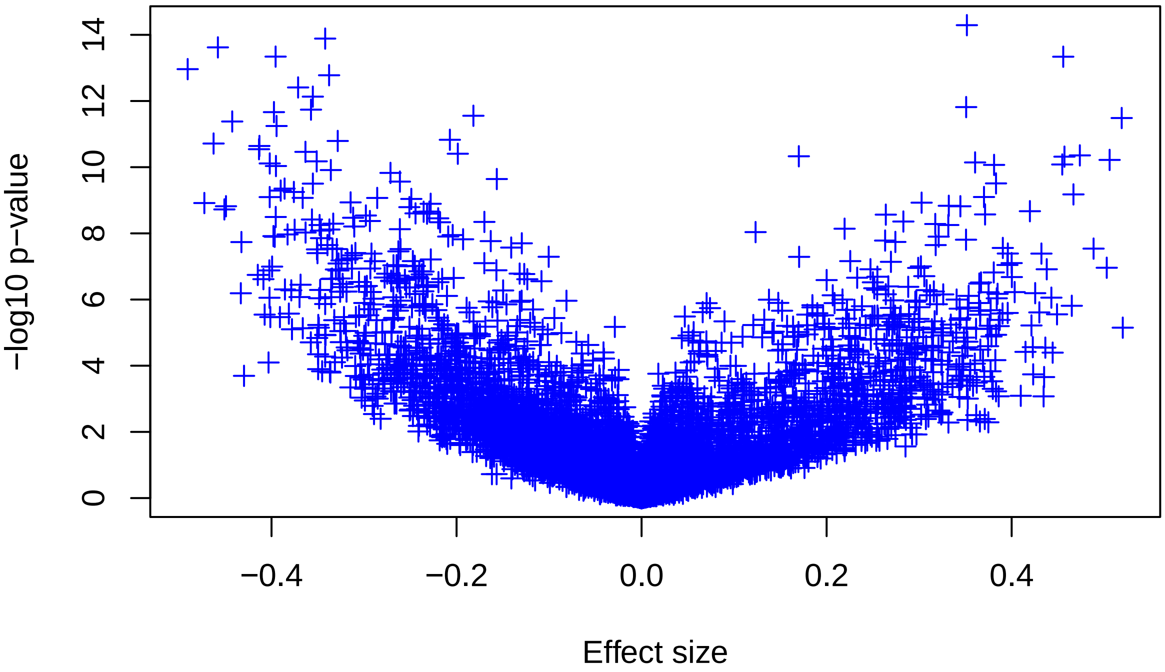


We are interested in finding regions of the genome that are different between cancer and normal samples. Furthermore, we want regions that are consistently different therefore we can treat this as an inference problem. We can compute a t-statistic for each CpG:

```
library(limma)
X <- model.matrix( ~ pd$Status)
fit <- lmFit(meth, X)
eb <- eBayes(fit)
```

A volcano plot reveals many differences:

```
library(rafalib)
splot(fit$coef[, 2],
      -log10(eb$p.value[, 2]),
      xlab = "Effect size",
      ylab = "-log10 p-value",
      col = "blue",
      pch = 3)
```



We can now compute a q-value for each test. If a feature resulted in a p-value of p, the q-value is the estimated pFDR for a list of all the features with a p-value at least as small as p. Let's check the q-values:

```
pvals <- eb$p.value[, 2]
library(qvalue)
res <- qvalue(pvals)
qvals <- res$qvalues
```

What proportion of CpG sites have q-values smaller than 0.05?

```
table(qvals < 0.05)[2] / length(qvals)
```

```
##      TRUE
## 0.2373412
```

What proportion of the CpGs showing statistically significant differences (defined with q-values in the previous question) are, on average, higher in cancer compared to normal samples?

```
index = which(qvals <= 0.05)
diffs = fit$coef[index, 2]
table(diffs > 0)[2] / length(diffs)
```

```
##      TRUE
## 0.4106411
```

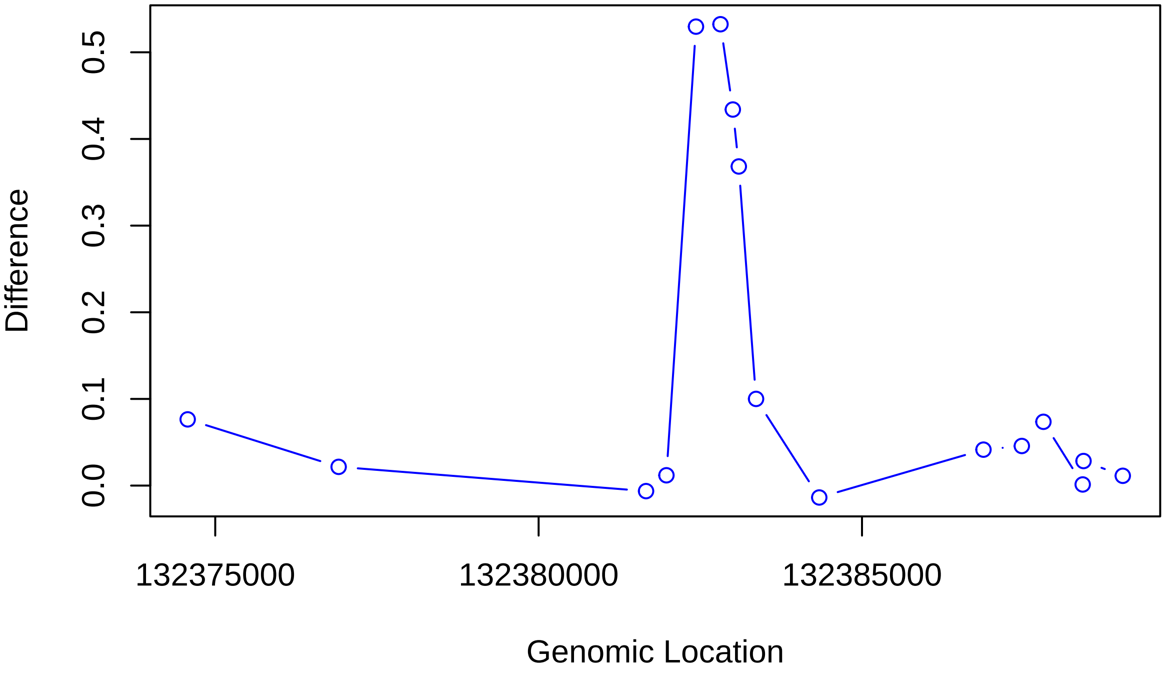
Now let's determine which of the differentially methylated CpGs are in CpG islands.

```
# Redefine CpG islands as cgi
library(AnnotationHub)
ah <- AnnotationHub(localHub = FALSE)
cgi <- ah[["AH5086"]]
index = which(qvals <= 0.05)
table(gr[index] %over% cgi)[2] / length(gr[index])
```

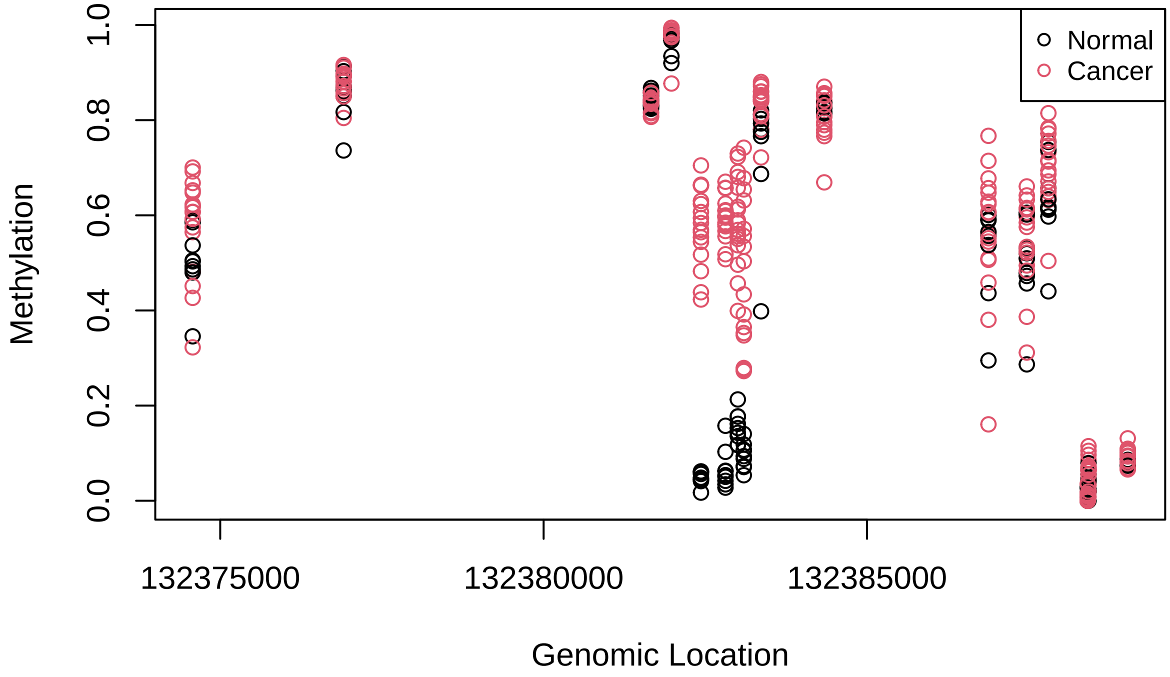
```
##      TRUE
## 0.2626267
```

If we have reason to believe for DNA methylation to have an effect on gene expression a region of the genome needs to be affected, not just a single CpG. Here is plot of the region surrounding the top hit:

```
library(GenomicRanges)
i <- which.min(eb$p.value[, 2])
middle <- gr[i, ]
Index <- gr %over% (middle + 10000)
cols = ifelse(pd$Status == "normal", 1, 2)
chr = as.factor(seqnames(gr))
pos = start(gr)
plot(pos[Index],
     fit$coef[Index, 2],
     type = "b",
     xlab = "Genomic Location",
     ylab = "Difference",
     col = "blue"
)
```



```
matplot(pos[Index],
        meth[Index, ],
        col = cols,
        xlab = "Genomic Location",
        ylab = "Methylation",
        pch = 1)
legend(
  x = "topright",
  legend = c("Normal", "Cancer"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)
```



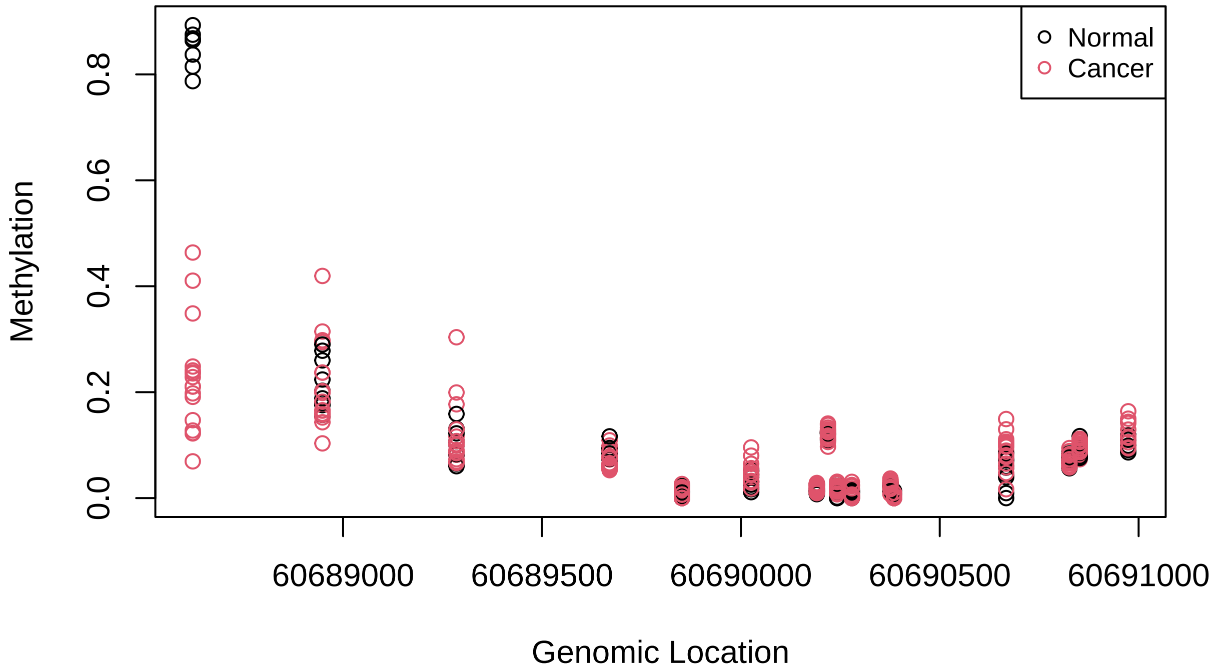
If we are going to perform regional analysis we first have to define a region. But one issue is that not only do we have to separate the analysis by chromosome but that within each chromosome we usually have big gaps creating subgroups of regions to be analyzed. We can create groups in the following way:

```
library(bumphunter)
cl = clusterMaker(chr, pos, maxGap = 500)
# Shows the number of points of difference with 1,2,3, etc.
# The number under each point is the number of clusters with that number
# of points of difference
table(table(cl))[1:4]
```

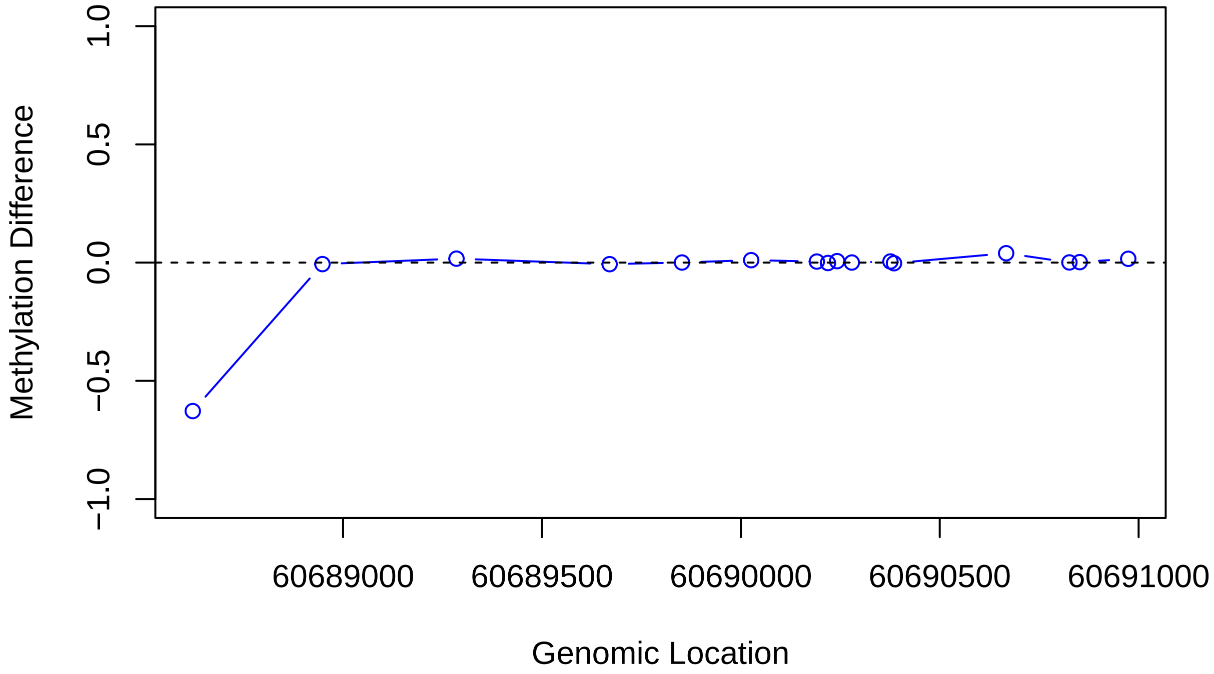
```
##
##      1      2      3      4
## 141457  18071  13227   6473
```

Now let's consider two example regions:

```
# Select the region with the smallest value
Index <- which(cl == cl[which.min(fit$coef[, 2])])
matplot(
  pos[Index],
  meth[Index, ],
  col = cols,
  pch = 1,
  xlab = "Genomic Location",
  ylab = "Methylation"
)
legend(
  x = "topright",
  legend = c("Normal", "Cancer"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)
```

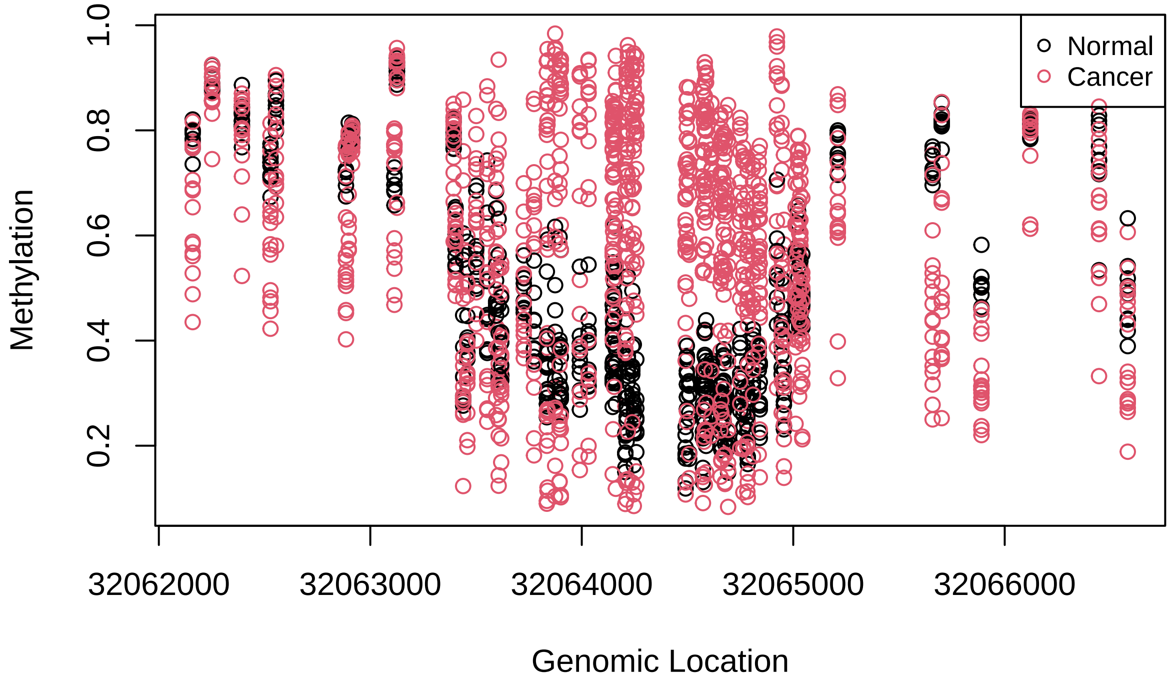


```
x1 = pos[Index]
y1 = fit$coef[Index, 2]
plot(x1,
     y1,
     xlab = "Genomic Location",
     ylab = "Methylation Difference",
     ylim = c(-1, 1),
     col = "blue",
     type = "b"
)
abline(h = 0, lty = 2)
```

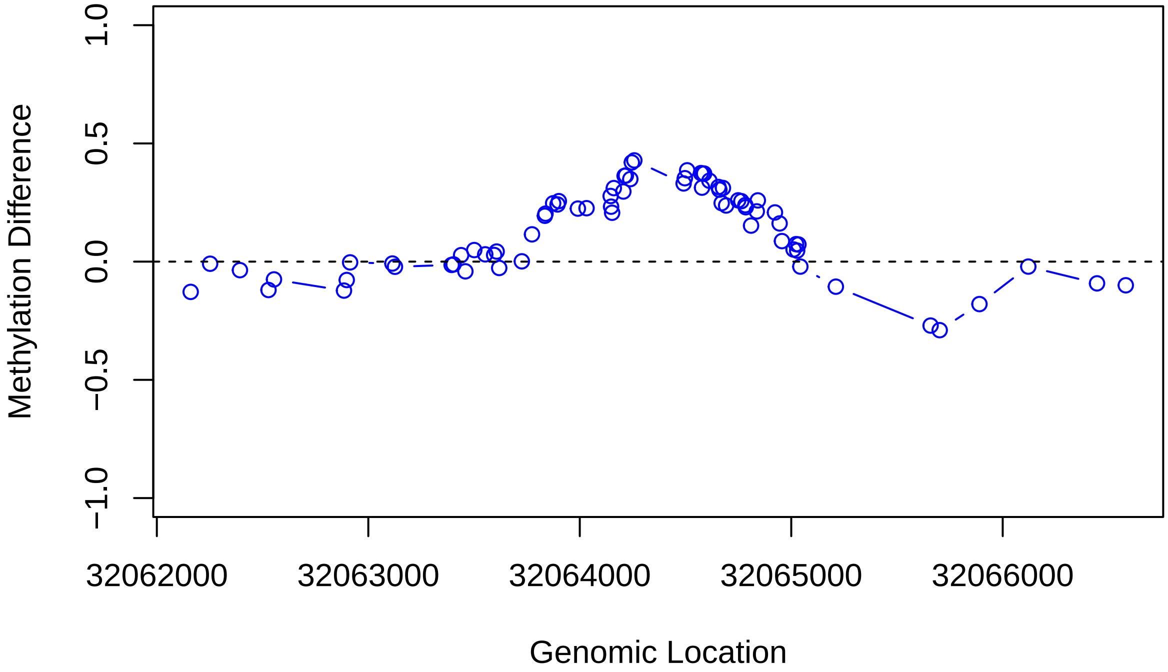


This region shows only a single CpG as different. In contrast, notice this region:

```
Index = which(cl == 72201)
matplot(
  pos[Index],
  meth[Index, ],
  col = cols,
  pch = 1,
  xlab = "Genomic Location",
  ylab = "Methylation"
)
legend(
  x = "topright",
  legend = c("Normal", "Cancer"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)
```

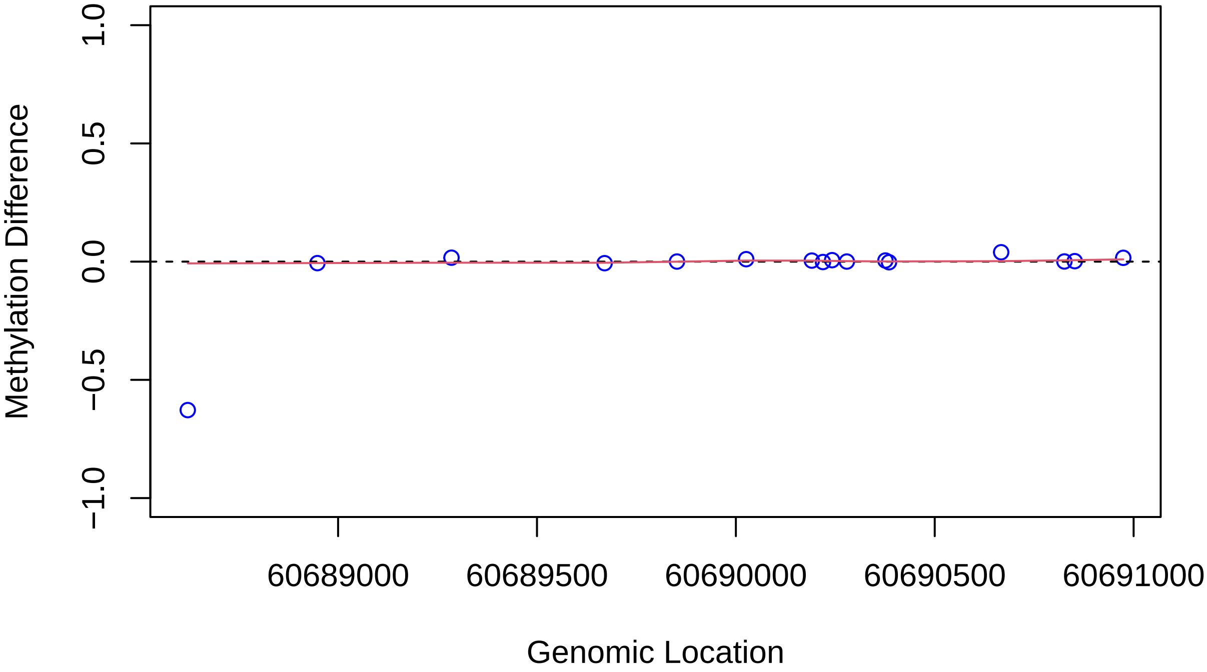


```
x2 = pos[Index]
y2 = fit$coef[Index, 2]
plot(x2,
     y2,
     xlab = "Genomic Location",
     ylab = "Methylation Difference",
     ylim = c(-1, 1),
     col = "blue",
     type = "b"
)
abline(h = 0, lty = 2)
```

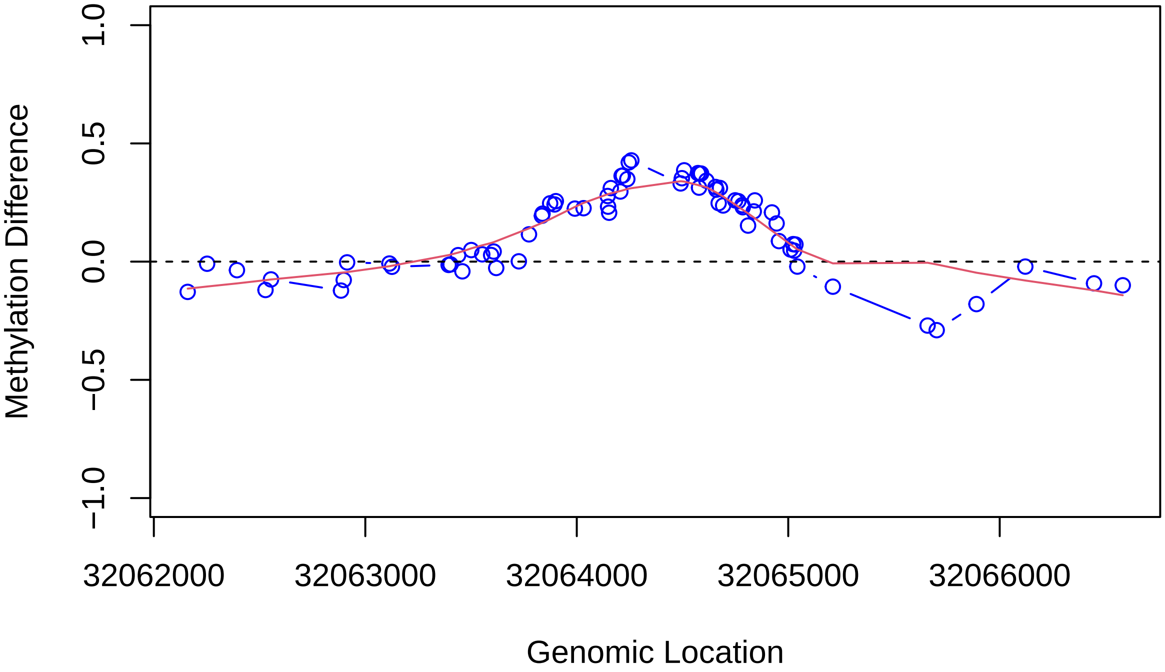


If we are interested in prioritizing regions over single points, we need an alternative approach. If we assume that the real signal is smooth, we could use statistical smoothing techniques such as *loess*. Here is an example two regions above:

```r
# Example 1
lfit <- loess(y1 ~ x1,
              degree = 1,
              family = "symmetric",
              span = 1 / 2)
plot(x1,
     y1,
     xlab = "Genomic Location",
     ylab = "Methylation Difference",
     ylim = c(-1, 1),
     col = "blue"
)
abline(h = 0, lty = 2)
lines(x1, lfit$fitted, col = 2)
```



```
# Example 2
lfit <- loess(y2 ~ x2,
              degree = 1,
              family = "symmetric",
              span = 1 / 2)
plot(x2,
     y2,
     xlab = "Genomic Location",
     ylab = "Methylation Difference",
     ylim = c(-1, 1),
     col = "blue",
     type = "b"
)
abline(h = 0, lty = 2)
lines(x2, lfit$fitted, col = 2)
```



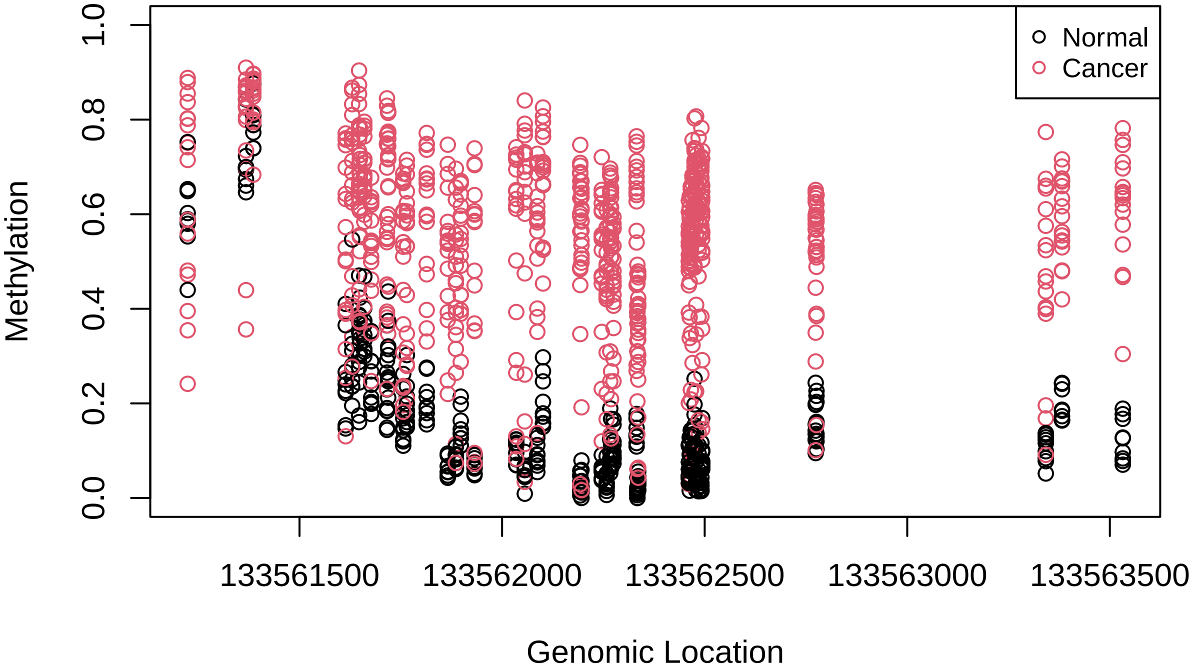
The *bumphunter* automates this procedure of finding DMRs:

```
chr = as.factor(seqnames(gr))
pos = start(gr)
res <- bumphunter(
  meth,
  X,
  chr = chr,
  pos = pos,
  cluster = cl,
  cutoff = 0.1,
  B = 0,
  verbose = FALSE
)
tab <- res$table
tab[1:4, 1:4]
```

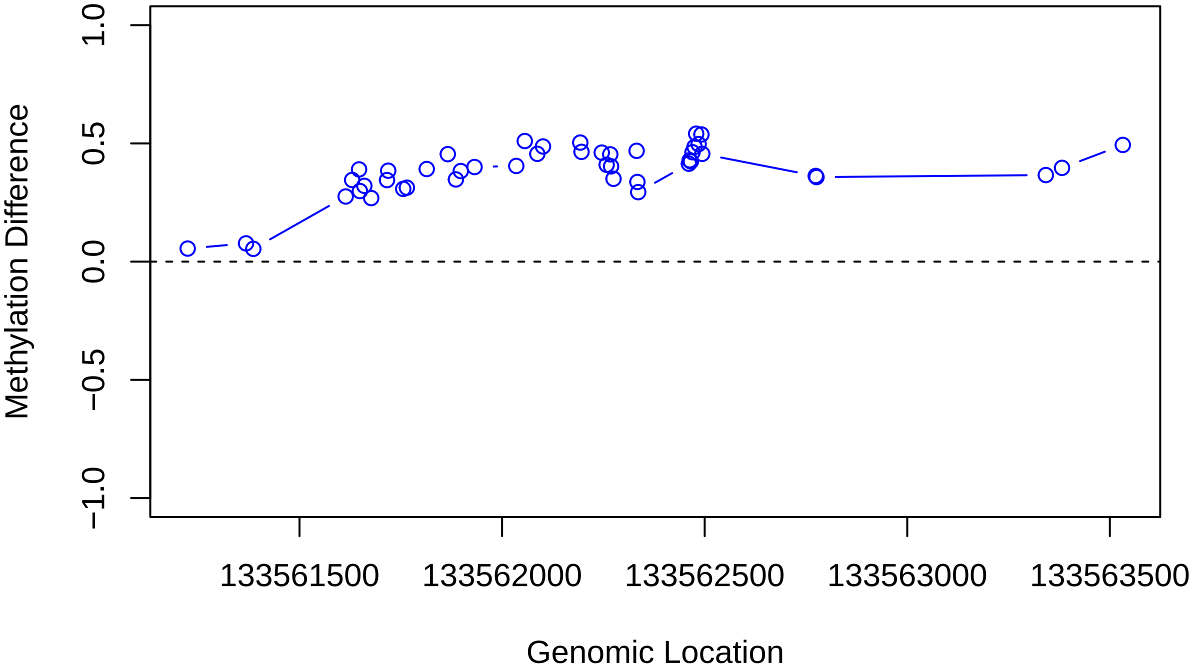
```
##        chr     start       end     value
## 6158  chr6 133561614 133562776 0.4048535
## 6568  chr7  27182493  27185282 0.3023301
## 5566  chr6  29520698  29521803 0.3798166
## 8453 chr10   8094093   8098005 0.2407042
```

We now have a list of regions instead of single points. Here we look at the region with the highest rank if we order by area:

```
Index = (tab[1, 7] - 3):(tab[1, 8] + 3)
matplot(
  pos[Index],
  meth[Index, , drop = TRUE],
  col = cols,
  pch = 1,
  xlab = "Genomic Location",
  ylab = "Methylation",
  ylim = c(0, 1)
)
legend(
  x = "topright",
  legend = c("Normal", "Cancer"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)
```



```
plot(
  pos[Index],
  res$fitted[Index, 1],
  xlab = "Genomic Location",
  ylab = "Methylation Difference",
  ylim = c(-1, 1),
  col = "blue",
  type = "b"
)
abline(h = 0, lty = 2)
```



Let's filter some DMRs by region size:

```
dmrs <- tab[tab$L >= 3, ]

# Convert to GenomicRanges object
dmrs <- makeGRangesFromDataFrame(dmrs)
```

Now let's find the distance to the closest island in *dmrs*:

```
d_island <- distanceToNearest(dmrs, cgi)
```

What proportion of DMRs overlap a CpG island (distance = 0)?

```
d_island <- as.data.frame(d_island)
table(d_island$distance == 0)[2] / length(d_island$distance)
```

```
##      TRUE
## 0.6105276
```

4 Methylation arrays data

The *minfi* package provides tools for analyzing Illumina's Methylation arrays, specifically the 450k and EPIC (also known as the 850k) arrays. In here, we will read idat files from the illumina 450K DNA methylation array:

```
library(minfi)
path <- "Data/idats"
```

Let's start by reading in the csv file, which contains clinical information. This has one row for each sample and one of the columns includes the "basenames" for the files:

```
targets <- read.csv("Data/idats/targets.csv", as.is = TRUE)
names(targets)[102:106]
```

```
## [1] "patient.weight" "Basename"       "Status"         "Tissue"
## [5] "Sex"
```

```
targets$Basename
```

```
## [1] "5775041065_R01C02" "5775041068_R01C02" "5775041065_R04C01"
## [4] "5775041068_R04C01" "5775041068_R06C01" "5775041084_R01C01"
```

To make this script work in any working directory we can edit that column to contain the absolute paths. Then we are ready to read in the raw data:

```
targets$Basename <- file.path(path, targets$Basename)
rgset <- read.metharray(targets$Basename, verbose = FALSE)
pData(rgset) <- as(targets, "DataFrame")
```

How many cancer samples are included in this dataset?

```
table(rgset$Status)
```

```
##
## cancer normal
##      3      3
```

We now have the raw data, red and green intensities which we have access to:

```
getRed(rgset)[1:3, 1:3]
```

```
##          [,1] [,2] [,3]
## 10600313  628  571  697
## 10600322 1315 2139 2186
## 10600328 1746 1700 2137
```

```
getGreen(rgset)[1:3, 1:3]
```

```
##          [,1] [,2]  [,3]
## 10600313  428  282   595
## 10600322 9819 9360 12673
## 10600328 3559 3448  4398
```

Let's use the built in preprocessing algorithm to get an object that gives us access to methylation estimates:

```
mset <- preprocessIllumina(rgset)
```

However, for this to be useful, we want to have the locations of each CpG, and to do that we need map the CpGs to genome. *minfi* keeps this information modular so that when the genome annotation gets updated, one can easily change the mapping:

```
mset <- mapToGenome(mset)
```

Now we are ready to obtain the methylation values and CpG locations:

```
# Methylation values
getBeta(mset, type="Illumina")[1:3, 1:3]
```

```
##                  [,1]       [,2]      [,3]
## cg13869341 0.81958479 0.80491145 0.8018963
## cg14008030 0.63250868 0.59663235 0.6147812
## cg12045430 0.09256506 0.06226323 0.1143808
```

```
# CpG locations
cpgloc <- granges(mset)
cpgloc[1:3]
```

```
## GRanges object with 3 ranges and 0 metadata columns:
```

```
##              seqnames    ranges strand
##                 <Rle> <IRanges>  <Rle>
##   cg13869341     chr1     15865      *
##   cg14008030     chr1     18827      *
##   cg12045430     chr1     29407      *
##   -------
##   seqinfo: 24 sequences from hg19 genome; no seqlengths
```

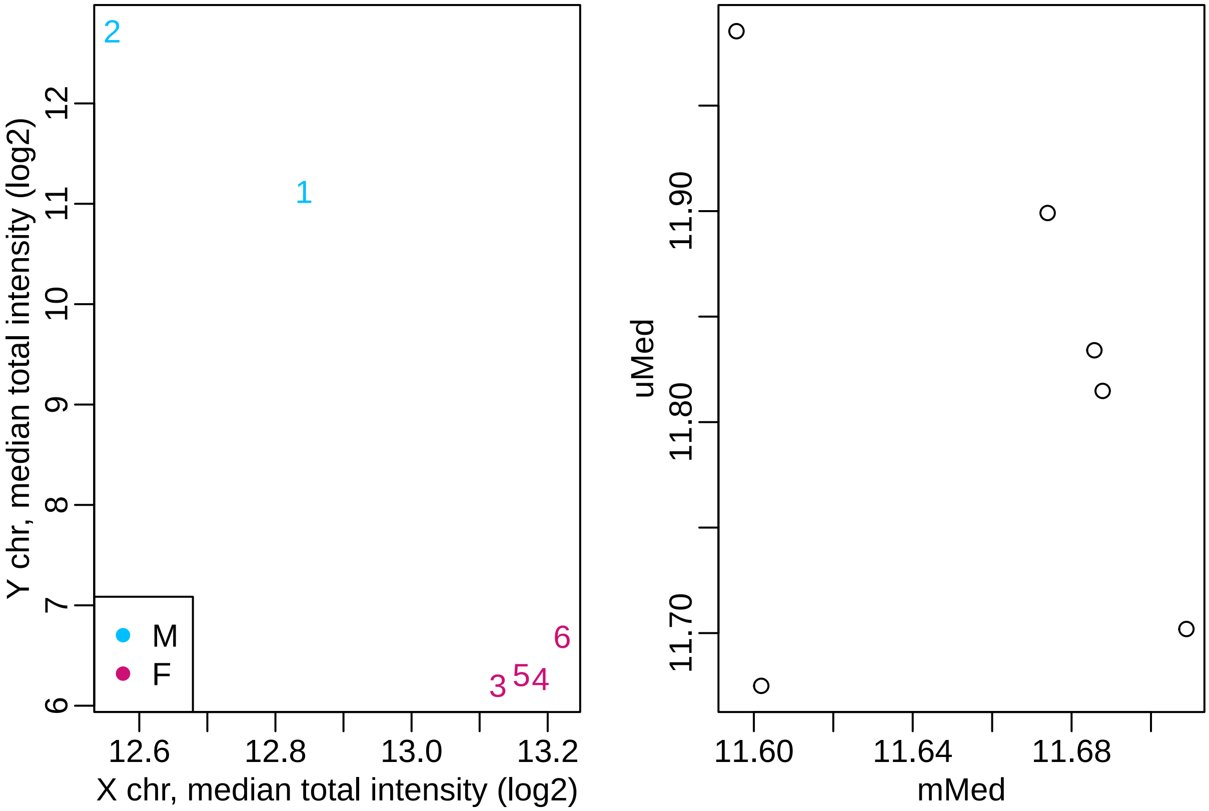
What is the estimated level of methylation for the CpG at location 153807318 on chr4 for sample “5775041068_R04C01”?

```
i <-
  which(seqnames(granges(mset)) == "chr4" &
          start(granges(mset)) == 153807318)
j <- which(rgset$Basename == file.path(path,"5775041068_R04C01"))
getBeta(mset, type = "Illumina")[i, j]
```

```
## cg09689478
##  0.4721712
```

Add other data as well:

```
mypar(1, 2)
# Sex data
colData(mset) <- getSex(mset)
plotSex(mset)

# QC data
plot(as.matrix(getQC(mset)))
```



Now let’s convert *mset* to a *GenomicRatioSet* so we can read it into *bumphunter*:

```r
grset <- ratioConvert(mset, what = "beta", type = "Illumina")
```

Find DMRs between cancer and normal samples:

```r
X = model.matrix( ~ pData(rgset)$Status)
dmrs <- bumphunter(grset, X, cutoff = 0.1, verbose = FALSE)
```

Now we will learn how to run *bumphunter* with smoothing. However to make the code run faster we will only run it on chr22:

```r
# Subset
index <- which(seqnames(grset) == "chr22")
grset2 <- grset[index, ]

# Run bumphunter without smoothing
X <- model.matrix( ~ pData(rgset)$Status)
res <- bumphunter(grset2, X, cutoff = 0.25, verbose = FALSE)

# Run bumphunter with smoothing
res2 <- bumphunter(grset2, X, cutoff = 0.25, smooth = TRUE, verbose = FALSE)
```

Notice that *res* has more DMRs and *res2* has longer DMRs:

```r
# Number of regions in res
nrow(res$table)
```

```
## [1] 180
```

```r
# Number of regions in res2
nrow(res2$table)
```

```
## [1] 21
```

```r
# Mean length of regions in res
mean(res$table$L)
```

```
## [1] 1.277778
```

```r
# Mean length of regions in res2
mean(res2$table$L)
```

```
## [1] 2.952381
```

5 Inference for DNA methylation

First, read in and preprocess the TCGA (The Cancer Genome Atlas program) data:

```r
library(minfi)
library(IlluminaHumanMethylation450kmanifest)
library(doParallel)
library(pkgmaker)
library(rafalib)

path = "Data/tcgaMethylationSubset"
targets = read.delim(file.path (path, "targets.txt"), as.is = TRUE)

# Case-control groups
table(targets$Tissue, targets$Status)
```

```
##
##          cancer normal
##   breast     13     13
##   colon      17     17
##   lung       19     19
```

```
# Subset the normal colon and lung
index = which(targets$Status == "normal" &
                targets$Tissue %in% c("colon", "lung"))
targets = targets[index, ]

# Read and preprocess methylation data
dat = read.metharray.exp(base = path,
                         targets = targets,
                         verbose = FALSE)
dat = preprocessIllumina(dat)
dat = mapToGenome(dat)
dat = ratioConvert(dat, type = "Illumina")

# Check CpG data
granges(dat)[1:4, ]
```

```
## GRanges object with 4 ranges and 0 metadata columns:
##              seqnames    ranges strand
##                 <Rle> <IRanges>  <Rle>
##   cg13869341     chr1     15865      *
##   cg14008030     chr1     18827      *
##   cg12045430     chr1     29407      *
##   cg20826792     chr1     29425      *
##   -------
##   seqinfo: 24 sequences from hg19 genome; no seqlengths
```

```
# Get tissue data
tissue = pData(dat)$Tissue

# Parallelize the processing
library(doParallel)
detectCores()
```

```
## [1] 4
```

```
registerDoParallel(cores = 4)
```

Now, let's build a model for finding DMRs based on tissue type:

```
X = model.matrix( ~ tissue)

# For illustrative purposes let's restrict it to one chromosome only
index = which(seqnames(dat) == "chr22")
dat = dat[index, ]
res = bumphunter(dat, X, cutoff = 0.1, B = 1000, verbose = FALSE)

# Check results
## value = the average height of the region (bump)
## L = the number of CpGs in the region
res$tab[1:4, c(1, 2, 3, 4, 13)]
```

```
##       chr    start      end      value  p.valueArea
## 877 chr22 30476089 30476525 -0.3120382 0.000000e+00
## 177 chr22 24890330 24891166  0.2542534 0.000000e+00
## 565 chr22 44568387 44568913  0.2341722 5.337176e-05
## 406 chr22 38506589 38506781  0.3310945 5.337176e-05
```

```
# How many regions (bumps) were identified?
dim(res$tab)[1]
```

```
## [1] 1084
```

Now, let's evaluate the relationship between these DMRs and CpG islands:

```
library(rafalib)
library(AnnotationHub)

# Load CpG islands
cgi = AnnotationHub(localHub = FALSE)[["AH5086"]]

# Restrict results to those with a FWER < 0.05
tab = res$tab[res$tab$fwer <= 0.05, ]

# Convert to GRanges object
tab = makeGRangesFromDataFrame(tab, keep.extra.columns = TRUE)

# Compute the distance between the nearest CpG island to each DMR
map = distanceToNearest(tab, cgi)
d = mcols(map)$distance

# Check the proportion of distances in categories of 0-1, 1-2000, etc.
prop.table(table(cut(
  as.numeric(d),
  c(0, 1, 2000, 5000, Inf),
  include.lowest = TRUE,
  right = FALSE
)))
```

```
##
##         [0,1)     [1,2e+03) [2e+03,5e+03)   [5e+03,Inf]
##     0.2872340     0.3031915     0.1648936     0.2446809
```

As you can see, most of these DMRs are not within the islands, but are 2000 base pairs far from the islands. We call these regions "CpG island shores". CpGs that are nowhere near an island are called "open sea CpGs".

Those results were for DMRs with a FWER < 0.05. Let's see the distances for all CpGs:

```
nulltab =  granges(dat)
nullmap = distanceToNearest(nulltab, cgi)
nulld = mcols(nullmap)$distance
prop.table(table(cut(
  nulld,
  c(0, 1, 2000, 5000, Inf),
  include.lowest = TRUE,
  right = FALSE
)))
```

```
##
##         [0,1)     [1,2e+03) [2e+03,5e+03)   [5e+03,Inf]
##     0.4168616     0.2657858     0.1460477     0.1713050
```

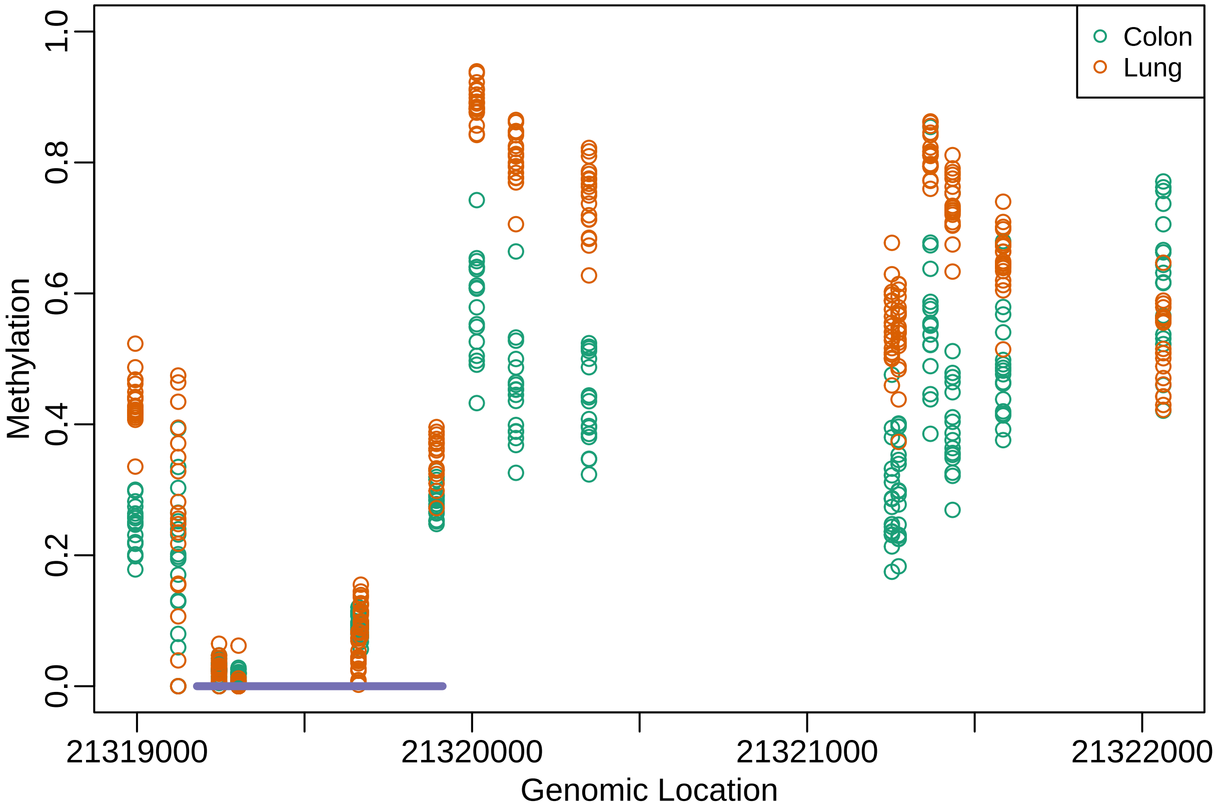
As expected, most CpGs are in the islands. So, the DMRs seem to be more common in the shores than the islands, in contrast to what we see for CpGs. We can also check these information using another function:

```r
# How many CpGs are within islands, shores, etc.?
prop.table(table(getIslandStatus(dat)))
```

```
##
##    Island   OpenSea     Shelf     Shore
## 0.4168616 0.1815949 0.1357577 0.2657858
```

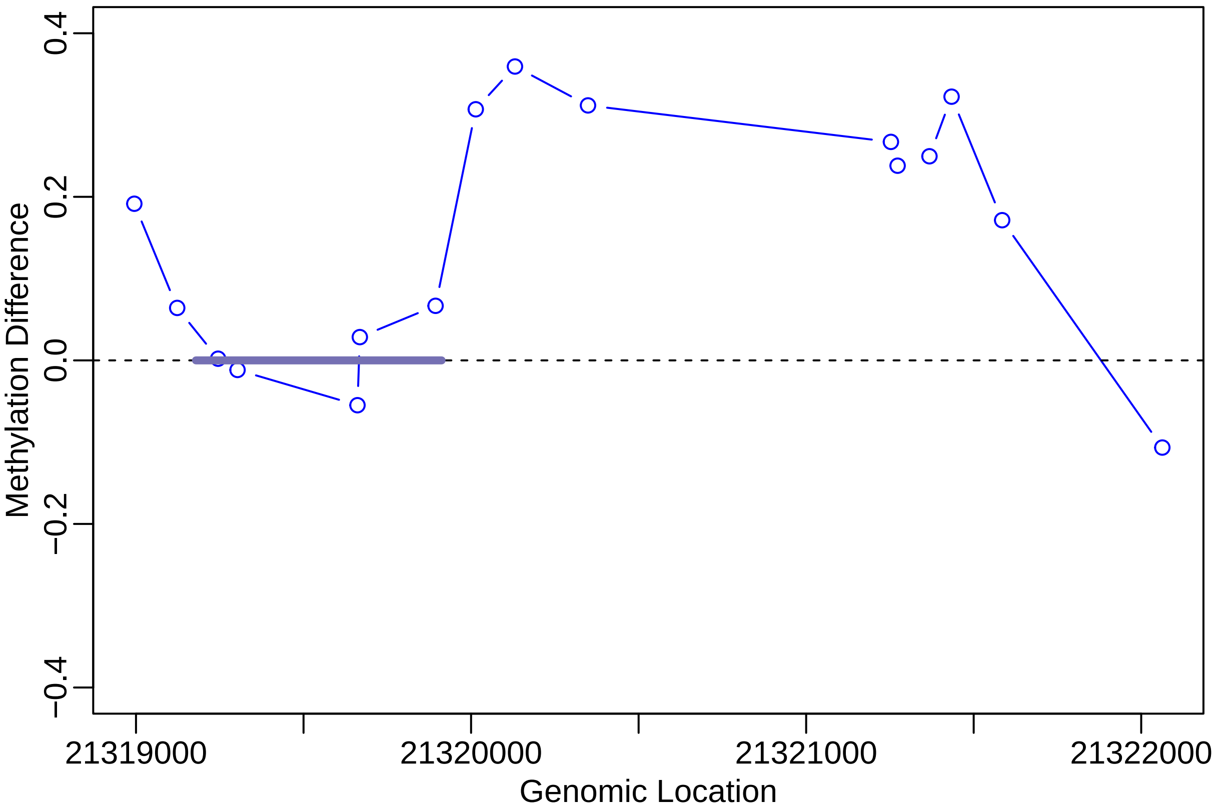
Now, let's plot one of those DMRs:

```r
# Sort DMRs by area
tab = tab[order(-mcols(tab)$area)]

# Add 3000 base pairs to each side
tab = tab + 3000

# Choose a specific DMR
i = 17

# Find all CpGs that are in the region
dataIndex = which(granges(dat) %over% tab[i])
cgiIndex = which(cgi %over% tab[i])
thecgi = cgi[cgiIndex]

# Get the positions of those CpGs
pos = start(dat)[dataIndex]

# Define the limit of the region to include in the plot
xlim = range(c(pos, start(thecgi), end(thecgi)))

# Get the beta values all CpGs
beta = getBeta(dat)

# Get the beta values for the CpGs in our specific region
y = beta[dataIndex, ]

# Color data points by tissue type
cols = as.factor(pData(dat)$Tissue)

# Plot
mypar(1, 1)
matplot(
  pos,
  y,
  col = as.numeric(cols),
  xlim = xlim,
  ylim = c(0, 1),
  xlab = "Genomic Location",
  ylab = "Methylation",
  pch = 1
)
legend(
  x = "topright",
  legend = c("Colon", "Lung"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)

## Show CpG island
apply(cbind(start(thecgi), end(thecgi)), 1, function(x) {
  segments(x[1], 0, x[2], 0, lwd = 4, col = 3)
})
```



```
## NULL
```

```
plot(
  pos,
  res$fitted[dataIndex],
  xlim = xlim,
  ylim = c(-0.4, 0.4),
  xlab = "Genomic Location",
  ylab = "Methylation Difference",
  col = "blue",
  type = "b"
)
abline(h = 0, lty = 2)

## Show CpG island
apply(cbind(start(thecgi), end(thecgi)), 1, function(x) {
  segments(x[1], 0, x[2], 0, lwd = 4, col = 3)
})
```



```
## NULL
```

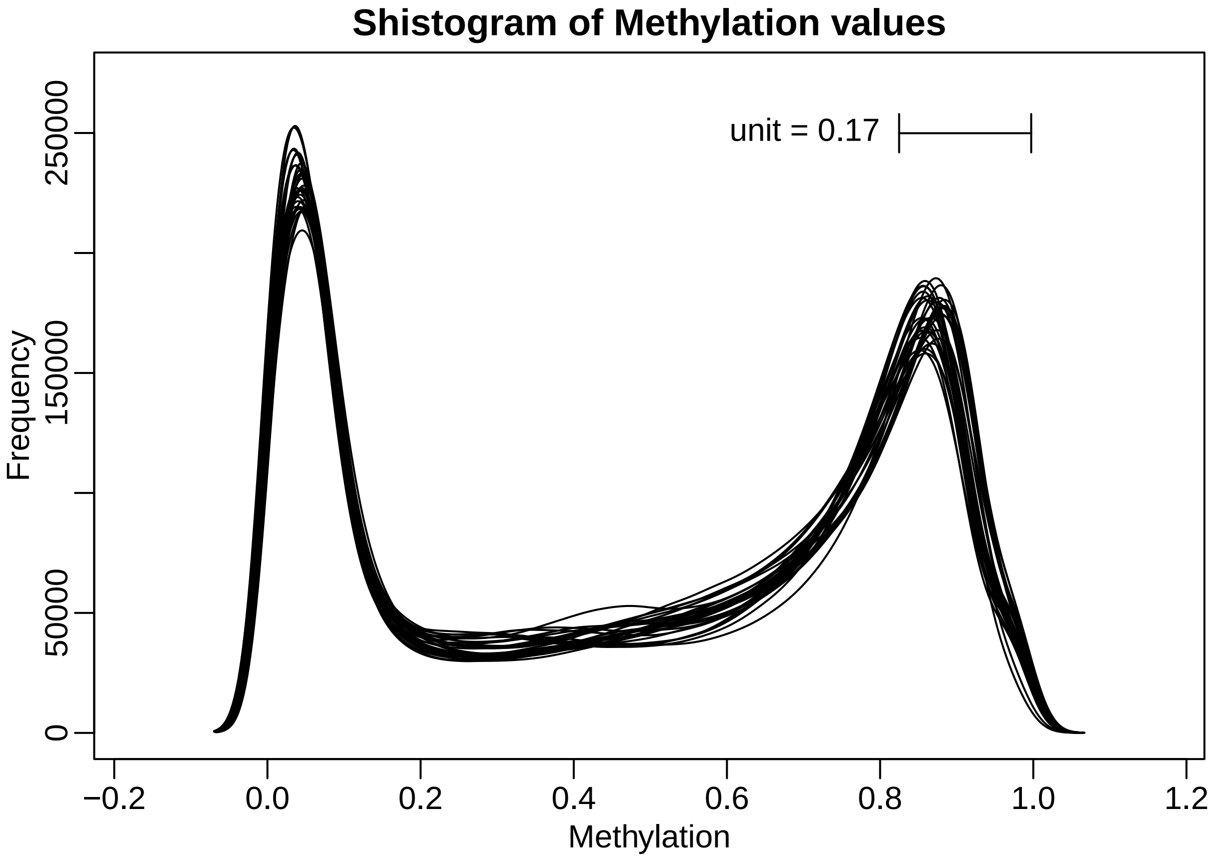
Let's perform another analysis on this data:

```
library(minfi)
library(IlluminaHumanMethylation450kmanifest)
library(IlluminaHumanMethylation450kanno.ilmn12.hg19)

targets = read.delim(file.path (path, "targets.txt"), as.is = TRUE)

# Subset the normal colon and breast samples
index <-
  which(targets$Status == "normal" &
          targets$Tissue %in% c("colon", "breast"))
targets <- targets[index, ]

# Read in methylation data
dat <- read.metharray.exp(base = path,
                          targets = targets,
                          verbose = FALSE)

# Preprocess the data
dat <- preprocessIllumina(dat)

# Assign locations to each CpG
dat <- mapToGenome(dat)
```

```r
# Precompute methylation values from U and M values
dat <- ratioConvert(dat, type = "Illumina")
```

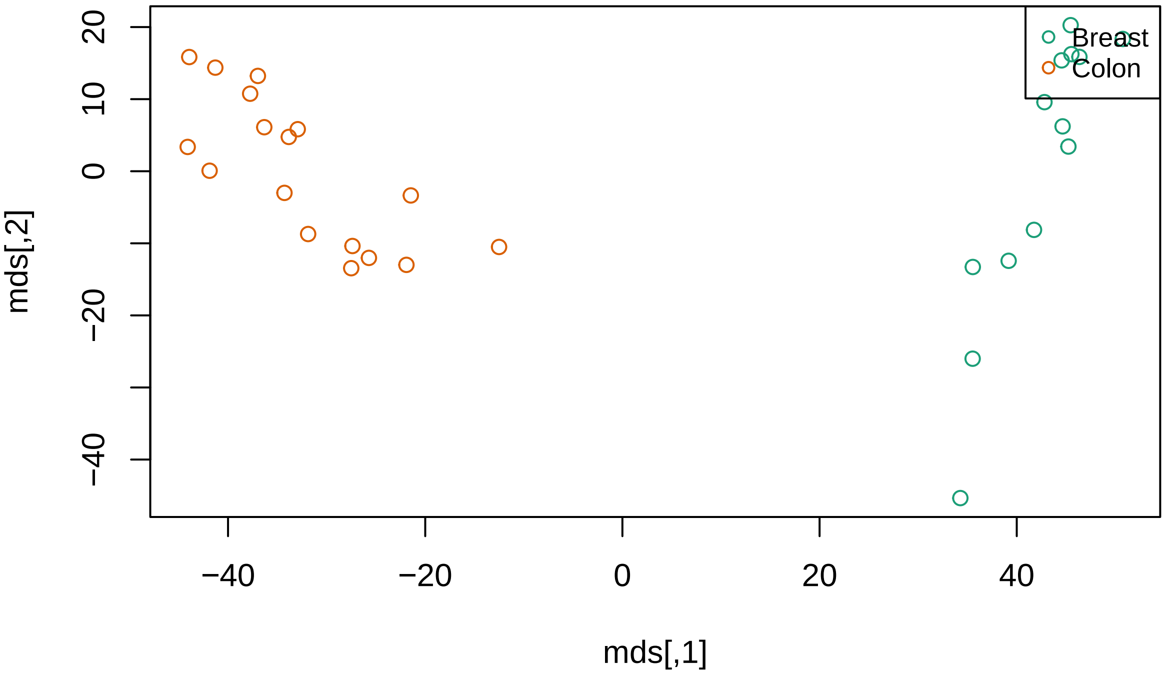
First look at the distribution of each sample:

```r
library(rafalib)

# Extract methylation values
y <- getBeta(dat)

# Plot
mypar(1, 1)
shist(y, xlab = "Methylation", main = "Shistogram of Methylation values")
```



Also, create an MDS plot to search for outlier samples. The first PC splits the data by tissue:

```r
mds <- cmdscale(dist(t(y)))
tissue <- as.factor(pData(dat)$Tissue)
plot(mds, col = tissue)
legend(
  x = "topright",
  legend = c("Breast", "Colon"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)
```



As expected, no sample stands out as an outlier.

Now we are ready to use statistical inference to find DMRs. Let's start by using the limma package to perform a site-by-site analysis:

```
library(limma)

# Create design matrix
tissue = as.factor(pData(dat)$Tissue)
X = model.matrix( ~ tissue)

# Extract methylation values
y = getBeta(dat)

# Obtain effect sizes and pvals with limma
fit = lmFit(y, X)
```

Which CpG has the largest effect size?

```
index = which.max(abs(fit$coef[, 2]))
print(index)
```

```
## cg22365276
##     360649
```

Which chromosome is this CpG on?

```
granges(dat)[index]
```

```
## GRanges object with 1 range and 0 metadata columns:
##              seqnames    ranges strand
##                 <Rle> <IRanges>  <Rle>
##   cg22365276    chr15  60688622      *
##   -------
##   seqinfo: 24 sequences from hg19 genome; no seqlengths
```

Now we will use the *qvalue()* function to determine the q-value for the CpG found in the previous question:

```
library(qvalue)

# Create design matrix
tissue <- as.factor(pData(dat)$Tissue)
X <- model.matrix( ~ tissue)

# Extract methylation values
y <- getBeta(dat)

# Obtain effect sizes and pvals with limma
fit <- lmFit(y, X)
eb <- eBayes(fit)

# Obtain q-values
qvals <- qvalue(eb$p.value[, 2])$qvalue

# What is the q-value for this CpG?
qvals[index]
```

```
##   cg22365276
## 1.269936e-27
```

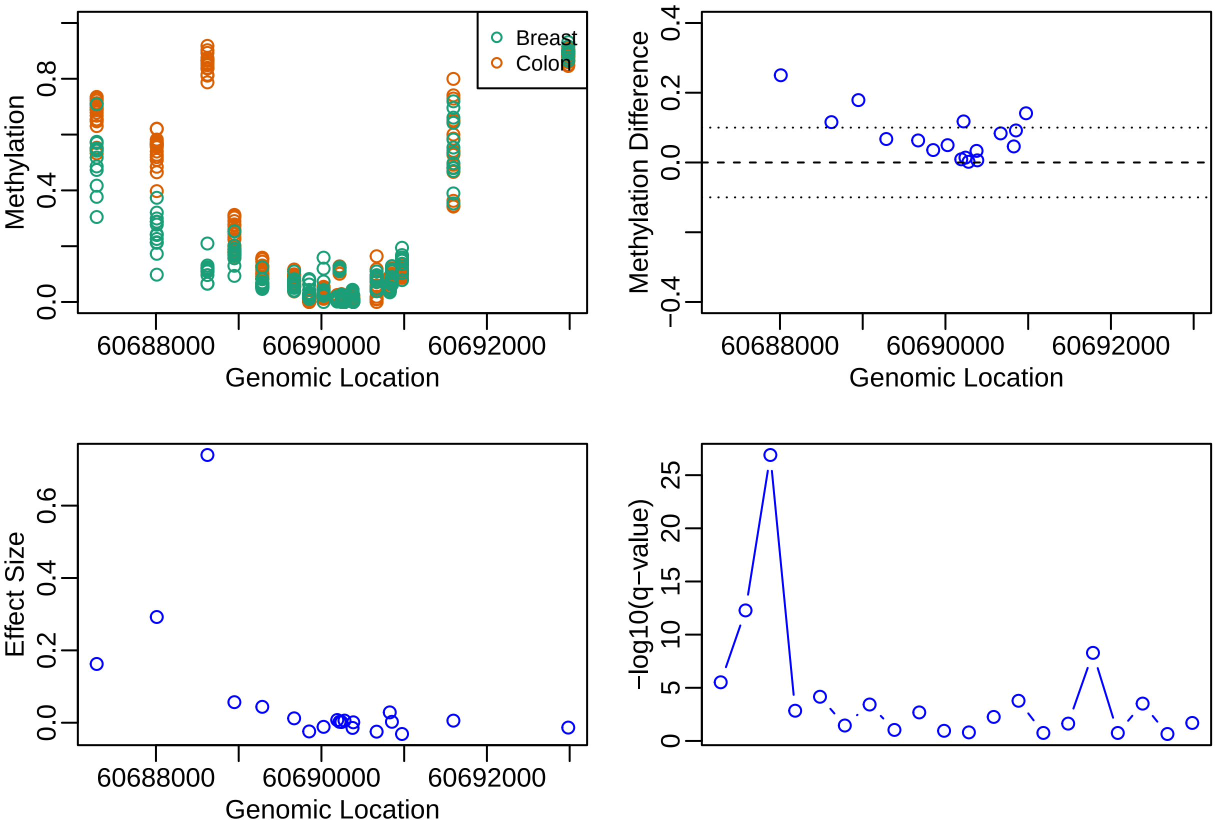
Find all the CpGs within 5000 basepairs of the CpG identified in the previous question:

```
tab = granges(dat)[index] + 5000
dataIndex = which(granges(dat) %over% tab)
```

Create plots showing the methylation values, methylation difference, estimated effect sizes, and q-values across all of the samples at these CpGs:

```
mypar(2, 2)

# Plot of the methylation values
## Get the positions of those CpGs
pos = start(dat)[dataIndex]

## Define the limit of the region to include in the plot
xlim = range(min(pos), max(pos))

## Get the beta values of those CpGs
beta = getBeta(dat[dataIndex])

## Color data points by tissue type
cols = as.factor(pData(dat)$Tissue)

## Plot
matplot(
  pos,
  beta,
  col = as.numeric(cols),
  xlim = xlim,
  ylim = c(0, 1),
  xlab = "Genomic Location",
  ylab = "Methylation",
```

```r
  pch = 1
)
legend(
  x = "topright",
  legend = c("Breast", "Colon"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)

# Plot of the methylation difference
plot(
  pos,
  eb$coefficients[dataIndex],
  xlim = xlim,
  ylim = c(-0.4, 0.4),
  xlab = "Genomic Location",
  ylab = "Methylation Difference",
  col = "blue"
)
abline(h = 0, lty = 2)

## Create a buffer zone for significancy
abline(h = 0.1, lty = 3)
abline(h = -0.1, lty = 3)

# Plot of the estimated effect sizes
plot(pos,
     eb$coefficients[dataIndex, 2],
     xlim = xlim,
     xlab = "Genomic Location",
     ylab = "Effect Size",
     col = "blue")

# Plot of the q-values
plot(
  -log10(qvals[dataIndex]),
  col = "blue",
  type = "b",
  ylab = "-log10(q-value)",
  xlab = "",
  xaxt = 'n'
)
```

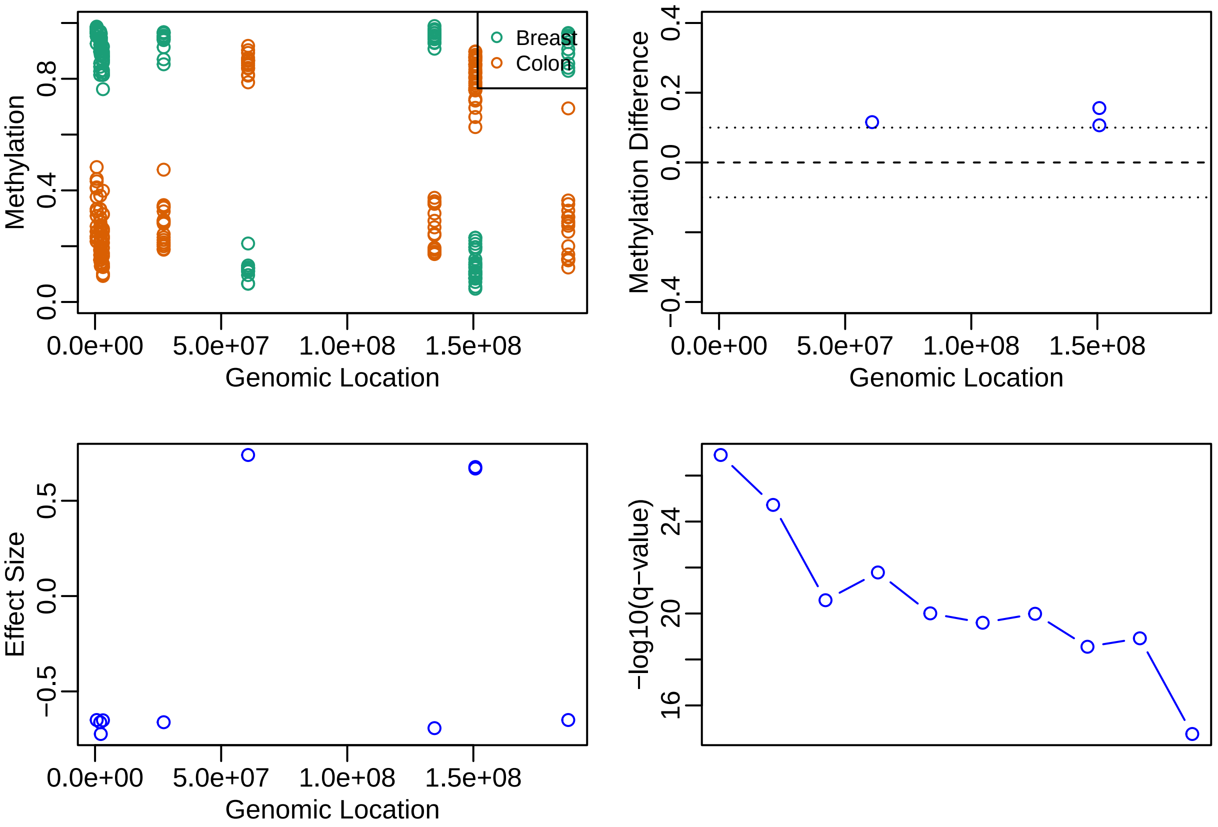


A region of about 1000 base pairs appears to be different.

Let's repeat the above code for the top 10 CpGs ranked by absolute value of the effect size:

```
o <- order(abs(fit$coef[, 2]), decreasing = TRUE)[1:10]

mypar(2, 2)

# Plot of the methylation values
## Get the positions of those CpGs
pos = start(dat)[o]

## Define the limit of the region to include in the plot
xlim = range(min(pos), max(pos))

## Get the beta values of those CpGs
beta = getBeta(dat[o])

## Color data points by tissue type
cols = as.factor(pData(dat)$Tissue)

## Plot
matplot(
  pos,
  beta,
  col = as.numeric(cols),
  xlim = xlim,
  ylim = c(0, 1),
  xlab = "Genomic Location",
```

```r
  ylab = "Methylation",
  pch = 1
)
legend(
  x = "topright",
  legend = c("Breast", "Colon"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)

# Plot of the methylation difference
plot(
  pos,
  eb$coefficients[o],
  xlim = xlim,
  ylim = c(-0.4, 0.4),
  xlab = "Genomic Location",
  ylab = "Methylation Difference",
  col = "blue"
)
abline(h = 0, lty = 2)

## Create a buffer zone for significancy
abline(h = 0.1, lty = 3)
abline(h = -0.1, lty = 3)

# Plot of the estimated effect sizes
plot(pos,
     eb$coefficients[o, 2],
     xlim = xlim,
     xlab = "Genomic Location",
     ylab = "Effect Size",
     col = "blue")

# Plot of the q-values
plot(
  -log10(qvals[o]),
  type = "b",
  col = "blue",
  ylab = "-log10(q-value)",
  xlab = "",
  xaxt = 'n'
)
```



For most plots we see groups of CpGs that are differentially methylated.

Now we are going to explicitly search for DMRs. We will use permutation to assess statistical significance. Because the function is slow, we will restrict our analysis to chromosome 15:

```
index <- which(seqnames(dat) == "chr15")
dat2 <- dat[index, ]

# Create design matrix
tissue <- as.factor(pData(dat)$Tissue)
X <- model.matrix( ~ tissue)

# Extract methylation values
set.seed(1)
res <- bumphunter(dat2, X, cutoff = 0.1, B = 100, verbose = FALSE)
res$tab[1:4, c(1, 2, 3, 4, 9, 13)]
```

```
##       chr    start      end     value  L p.valueArea
## 89  chr15 27215757 27216819 0.4018549 14           0
## 326 chr15 45670478 45671347 0.3161932 15           0
## 86  chr15 27111774 27113511 0.2600836 16           0
## 263 chr15 41793629 41795475 0.3794984  8           0
```

How many regions achieve an FWER lower than 0.05?

```
table(res$table$fwer < 0.05)[2]
```

```
## TRUE
##  420
```

Previously we performed a CpG by CpG analysis and obtained qvalues. Create an index for the CpGs that achieve qvalues smaller than 0.05 and a large effect size larger than 0.5 (in absolute value):

```
index <-
  which(qvals < 0.05 &
          abs(fit$coef[, 2]) > 0.5 & seqnames(dat) == "chr15")
```

Now create a table of the DMRs returned by *bumphunter* that had 3 or more probes and convert the table into GRanges:

```
tab <- res$tab[res$tab$L >= 3, ]
tab <- makeGRangesFromDataFrame(tab)
```

What proportion of the CpGs indexed by “index” are inside regions found in “tab”?

```
overlaps <- findOverlaps(tab, granges(dat[index]))
length(overlaps) / length(index)
```

```
## [1] 0.5714286
```

Now let’s download the table of CpG islands using AnnotationHub:

```
library(AnnotationHub)
cgi <- AnnotationHub(localHub = FALSE)[["AH5086"]]
```

Create a GRanges object from the list of DMRs we computed previously:

```
tab <- res$tab[res$tab$fwer <= 0.05, ]
tab <- makeGRangesFromDataFrame(tab)
```

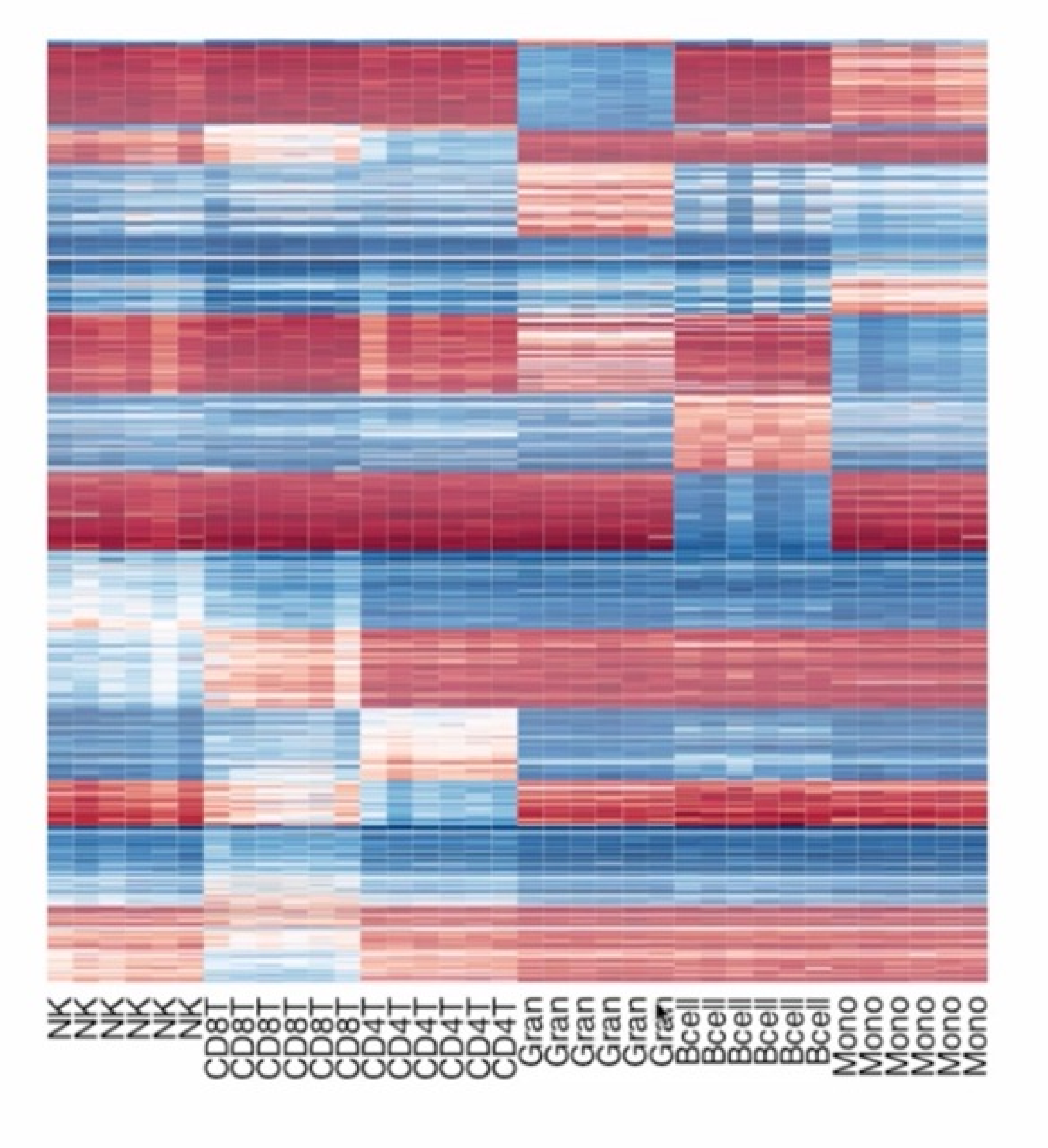
What proportion of the regions represented in “tab” do not overlap islands, but overall CpG islands shores (within 2000 basepairs) ?

```
map = distanceToNearest(tab, cgi)
d = mcols(map)$distance
prop.table(table(cut(
  d,
  c(0, 1, 2000, 5000, Inf),
  include.lowest = TRUE,
  right = FALSE
)))[2]
```

```
## [1,2e+03)
##  0.199536
```

6 Accounting for cell composition

Most of the data we study comes from bulk of tissue which is usually a combination of cell types. Usually, different cell types have very different methylation profiles. In the figure below, you can see the methylation profiles for different cell types in the blood of 6 individuals.



As you can see, the difference between individuals for each cell type is negligible, but the difference between cell types is substantial. This problem could be attacked as an batch effect.

7 Multi-resolution analysis

Until now, we were analyzing data based on regions (resolution) of 10 to 20 kilobases long. Now, we are going to analyze data for bigger resolutions:

```
# Import and preprocess data
library(minfi)
```

```
path = "Data/tcgaMethylationSubset"
targets <- read.delim(file.path (path, "targets.txt"), as.is = TRUE)
index <- which(targets$Tissue == "colon")
targets <- targets[index, ]
dat <- read.metharray.exp(base = path,
                          targets = targets,
                          verbose = FALSE)
dat <- preprocessIllumina(dat)
dat <- mapToGenome(dat)

# Collapse CpG islands
cdat <- cpgCollapse(dat, verbose = FALSE)
```

How many regions are represented in the collapsed object?

```
length(cdat[[1]])
```

```
## [1] 223497
```

We can see the type of regions that are represented in this collapsed object:

```
head(granges(cdat$obj))
```

```
## GRanges object with 6 ranges and 3 metadata columns:
##       seqnames      ranges strand |        id        type blockgroup
##          <Rle>   <IRanges>  <Rle> | <numeric> <character>  <numeric>
##   [1]     chr1       15865      * |         1     OpenSea          1
##   [2]     chr1       18827      * |         2     OpenSea          1
##   [3]     chr1 29407-29435      * |         3      Island         NA
##   [4]     chr1       68849      * |         4     OpenSea          1
##   [5]     chr1       69591      * |         5     OpenSea          1
##   [6]     chr1       91550      * |         6     OpenSea          1
##   -------
##   seqinfo: 24 sequences from an unspecified genome; no seqlengths
```

What proportion of the regions are OpenSea regions?

```
table(granges(cdat$obj)$type == "OpenSea")[2] / length(granges(cdat$obj))
```

```
##      TRUE
## 0.5094341
```

Now let’s find DMRs between cancer and normal:

```
status <- factor(pData(cdat$obj)$Status, level = c("normal", "cancer"))
X <- model.matrix( ~ status)
res <- blockFinder(cdat$obj, X, cutoff = 0.05)

# Take a peek at the blocks
res$table[1:4, c(1, 2, 3, 4, 9)]
```

```
##        chr     start       end      value   L
## 2083 chr11   4388493   6452369 -0.1362436 213
## 2595 chr15  24778439  28263732 -0.1083627 261
## 658   chr2 217454810 219125856 -0.1388161 183
## 308   chr1 151974222 153522342 -0.1493356 165
```

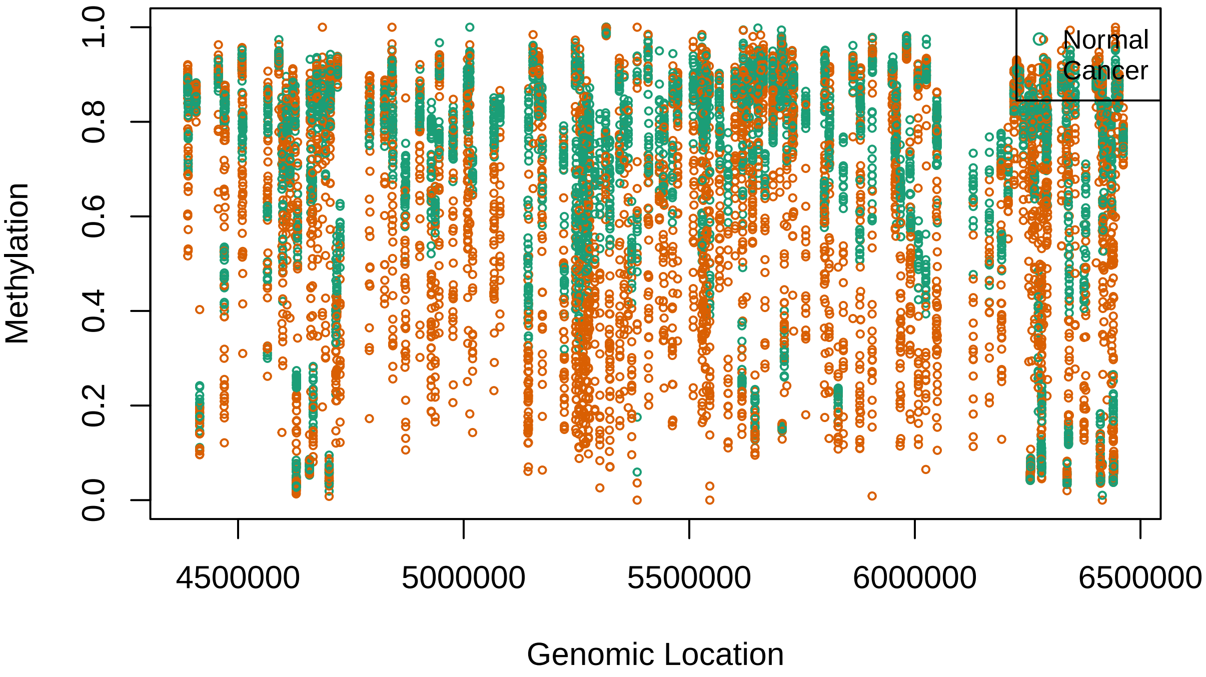
What proportion of the blocks reported in “res$table” are hypomethyated in cancer (lower methylation in cancer versus normal)?

```
table(res$table$value < 0)[2] / length(res$table$value)
```

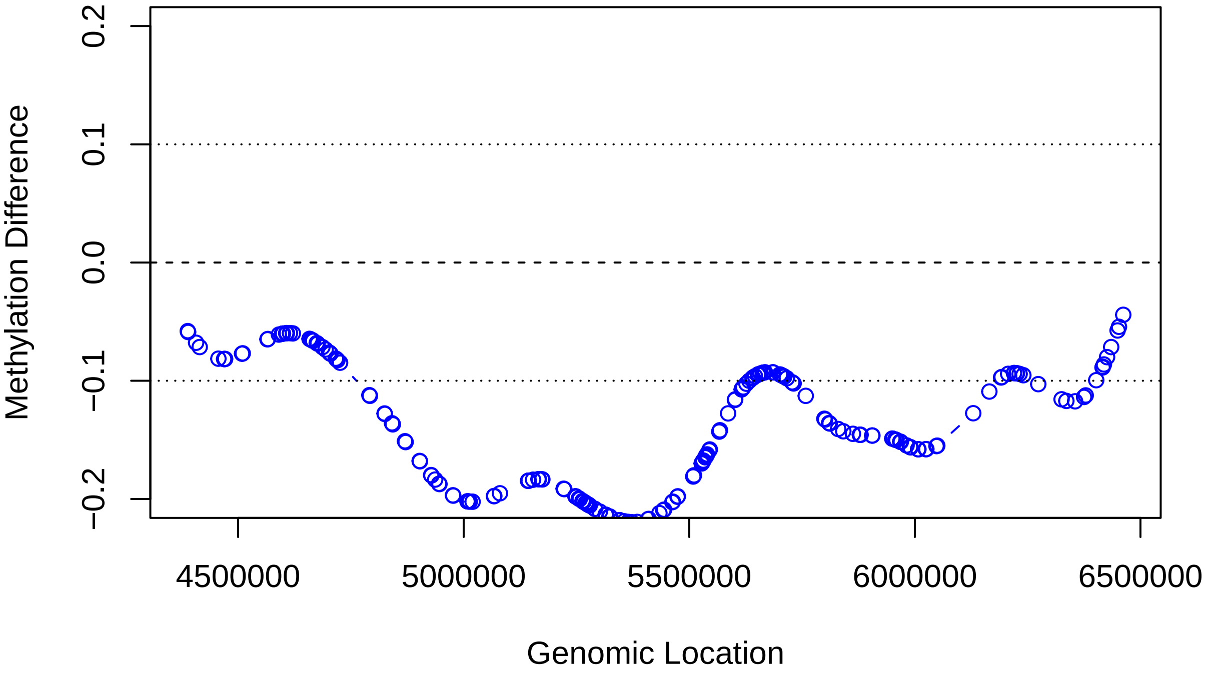
```
##      TRUE
## 0.9475503
```

Let's make figures now:

```
tab = makeGRangesFromDataFrame(res$table)
index = granges(cdat$obj) %over% (tab[1] + 10000)
pos = start(cdat$obj)[index]
col = as.numeric(status)

# Plot for methylation values
matplot(pos,
        getBeta(cdat$obj)[index, ],
        col = col,
        pch = 1,
        cex = 0.5,
        xlab = "Genomic Location",
        ylab = "Methylation"
)
legend(
  x = "topright",
  legend = c("Normal", "Cancer"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)
```



```
# Plot for methylation difference
plot(
  pos,
  res$fitted[index],
  xlab = "Genomic Location",
  ylab = "Methylation Difference",
  col = "blue",
  type = "b",
  ylim = c(-0.2, 0.2)
)
abline(h = 0, lty = 2)

## Create a buffer zone for significancy
abline(h = 0.1, lty = 3)
abline(h = -0.1, lty = 3)
```



8 Whole genome bisulfite sequencing

Reduced Representation Bisulfite Sequencing (RRBS) is an experimental technique widely used to manipulate the regions of the genome we measure. An enzyme is used to cut DNA at CCGG and the general idea is to filter out small or large molecules once DNA is cut. We can use Bioconductor tools to predict the size of these regions.

Let's load the genome package and create an object with the sequence for chr22:

```
library(BSgenome.Hsapiens.UCSC.hg19)
chr22 <- Hsapiens[["chr22"]]
```

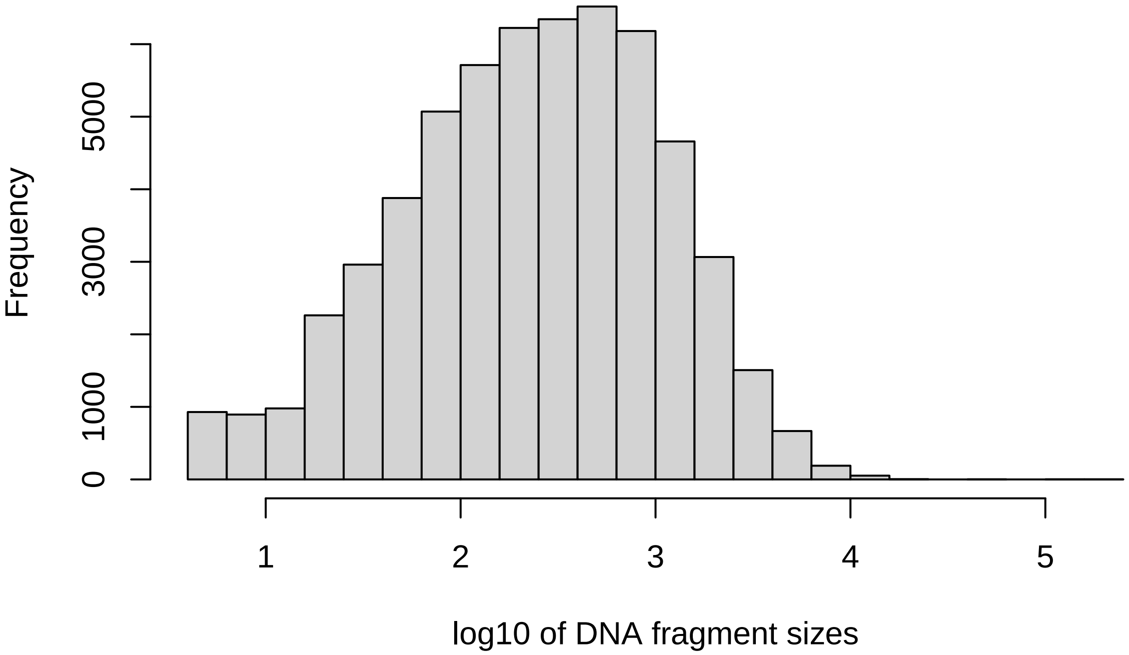
How many CCGG do we find on chr22?

```
CCGG <- matchPattern("CCGG", chr22)
length(CCGG)
```

```
## [1] 58102
```

Plot a histogram of the DNA fragment sizes after we cut at CCGG positions:

```
size = diff(start(CCGG))
hist(log10(size), main = "", xlab = "log10 of DNA fragment sizes")
```



A typical size to filter are DNA regions between 40 and 220 basepairs. What proportion of the fragments created for chr22 are between 40 and 220 basepairs?

```
table(size < 220 & size > 40)[2] / length(size)
```

```
##      TRUE
## 0.3254161
```

If we try to sequence all of chromosome 22, we would need to sequence 51,304,566 bases. If instead we only sequence fragments of sizes between 40 and 220 basepairs, how many bases would we need to sequence?

```
sum(size[size <= 220 & size >= 40])
```

```
## [1] 2203342
```

Whole-genome bisulfite sequencing (WGBS) is another sequencing-based protocol for measuring DNA methylation in samples. Now, we will take a look at WGBS data from a set of paired tumor and normal colon samples:

```
path <- "Data/colonCancerWGBS"
# Read in sample metadata table
targets <-
  read.table(file.path(path, "targets.txt"),
             header = TRUE,
             sep = "    ")
```

The data consists of the following:

1. genomic positions (chromosome and location) for methylation sites
2. M (Methylation) values, the number of reads supporting methylation covering each site
3. Cov (Coverage) values, the total number of reads covering each site

Unzip and prepare data:

```
tar -zxvf Data/colonCancerWGBS/SRR949210.chr22.1.tar.gz
tar -zxvf Data/colonCancerWGBS/SRR949210.chr22.2.tar.gz
mv SRR949210.chr22.1/* Output/
mv SRR949210.chr22.2/* Output/
```

```
rm -rf SRR949210.chr22.1
rm -rf SRR949210.chr22.2
```

```
## x SRR949210.chr22.1/
## x SRR949210.chr22.1/.DS_Store
## x SRR949210.chr22.1/SRR949212.chr22.cov
## x SRR949210.chr22.1/SRR949210.chr22.cov
## x SRR949210.chr22.1/SRR949211.chr22.cov
## x SRR949210.chr22.2/
## x SRR949210.chr22.2/SRR949214.chr22.cov
## x SRR949210.chr22.2/SRR949215.chr22.cov
## x SRR949210.chr22.2/SRR949213.chr22.cov
```

Read data:

```
library(bsseq)
# Turn metadata into DataFrame w/ sample names as rownames
targets <- DataFrame(targets, row.names = as.character(targets$Run))

# Specify path to files in same order as targets table
path <- "Output/"
covfiles <- file.path(path, paste0(rownames(targets), ".chr22.cov"))

# Read coverage files
colonCancerWGBS <- read.bismark(files = covfiles,
                                rmZeroCov = TRUE,
                                colData = targets)

# Check pheno data
pData(colonCancerWGBS)[c(2, 10)]
```

```
## DataFrame with 6 rows and 2 columns
##                          title    characteristics_ch1
##                    <character>            <character>
## SRR949210  Colon_Tumor_Primary disease type: modera..
## SRR949211  Colon_Tumor_Primary disease type: modera..
## SRR949212  Colon_Tumor_Primary disease type: modera..
## SRR949213 Colon_Primary_Normal     disease type: None
## SRR949214 Colon_Primary_Normal     disease type: None
## SRR949215 Colon_Primary_Normal     disease type: None
```

```
# Check geno data
granges(colonCancerWGBS)[1:4]
```

```
## GRanges object with 4 ranges and 0 metadata columns:
##       seqnames    ranges strand
##          <Rle> <IRanges>  <Rle>
##   [1]       22  10514937      *
##   [2]       22  10515170      *
##   [3]       22  10519494      *
##   [4]       22  10519495      *
##   -------
##   seqinfo: 1 sequence from an unspecified genome; no seqlengths
```

Now, extract the coverage and the number of reads with evidence for methylation:

```
cov <- getCoverage(colonCancerWGBS, type = "Cov")
m <- getCoverage(colonCancerWGBS, type = "M")

# Take a peak
m[1:4, ]
```

```
##      SRR949210 SRR949211 SRR949212 SRR949213 SRR949214 SRR949215
## [1,]         1         0         0         0         0         0
## [2,]         0         0         0         0         0         0
## [3,]         0         0         0         0         0         0
## [4,]         0         0         0         0         1         0
```

What proportion of the reported CpGs have some coverage in all sample?

```
index = apply(cov > 0, 1, all)
mean(index)
```

```
## [1] 0.7743644
```

9 Exercise - Epigenetic changes in peripheral blood in acute mania

9.1 Paper

Let's check the paper details first:

Title: Association of DNA Methylation with Acute Mania and Inflammatory Markers

Abstract: In order to determine whether epigenetic changes specific to the manic mood state can be detected in peripheral blood samples we assayed DNA methylation levels genome-wide in serum samples obtained from 20 patients hospitalized for mania and 20 unaffected controls using the Illumina 450K methylation arrays. We identified a methylation locus in the CYP11A1 gene, which is regulated by corticotropin, that is hypo-methylated in individuals hospitalized for mania compared with unaffected controls. DNA methylation levels at this locus appear to be state-related as levels in follow-up samples collected from mania patients six months after hospitalization were similar to those observed in controls. In addition, we found that methylation levels at the CYP11A1 locus were significantly correlated with three inflammatory markers in serum in acute mania cases but not in unaffected controls. We conclude that mania is associated with alterations in levels of DNA methylation and inflammatory markers. Since epigenetic markers are potentially malleable, a better understanding of the role of epigenetics may lead to new methods for the prevention and treatment of mood disorders.

Cohort Demographics:

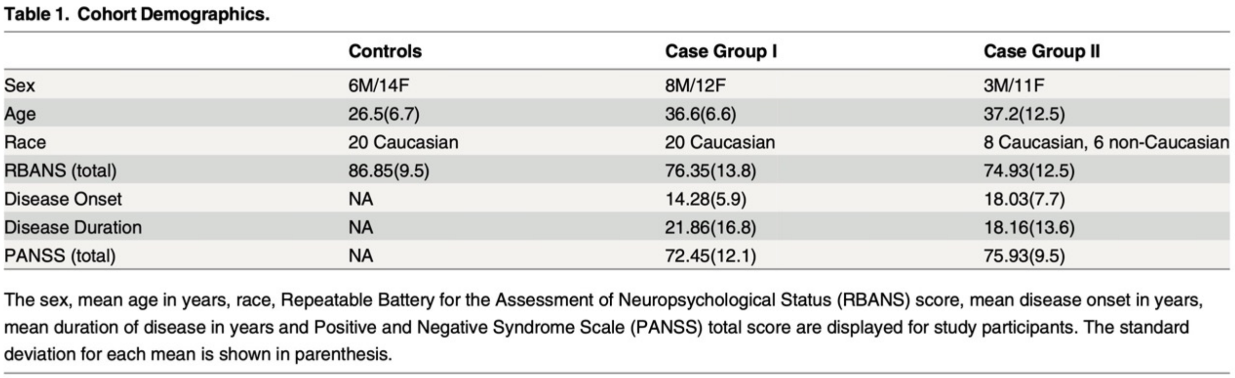
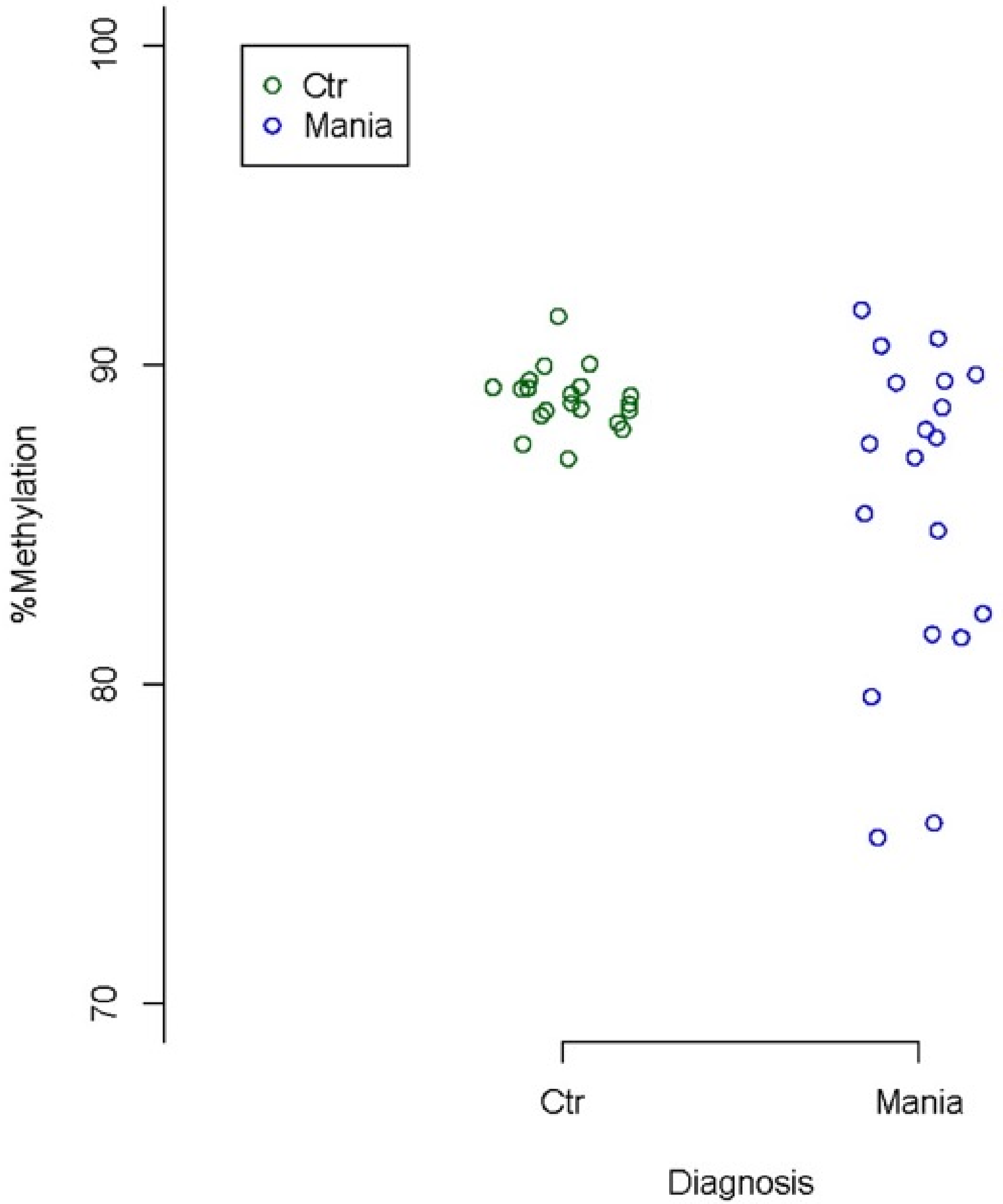


Table 1. Cohort Demographics.

	Controls	Case Group I	Case Group II
Sex	6M/14F	8M/12F	3M/11F
Age	26.5(6.7)	36.6(6.6)	37.2(12.5)
Race	20 Caucasian	20 Caucasian	8 Caucasian, 6 non-Caucasian
RBANS (total)	86.85(9.5)	76.35(13.8)	74.93(12.5)
Disease Onset	NA	14.28(5.9)	18.03(7.7)
Disease Duration	NA	21.86(16.8)	18.16(13.6)
PANSS (total)	NA	72.45(12.1)	75.93(9.5)

The sex, mean age in years, race, Repeatable Battery for the Assessment of Neuropsychological Status (RBANS) score, mean disease onset in years, mean duration of disease in years and Positive and Negative Syndrome Scale (PANSS) total score are displayed for study participants. The standard deviation for each mean is shown in parenthesis.

DNA methylation between cases and controls:



9.2 Import data

Now let's import the data:

```
# Import IDAT files into an RGChannelSet
library(minfi)
rg_68777 = read.metharray.exp("Data/GSE68777/idat", verbose = FALSE)
```

Now let's get the phenotype data:

```
## x GSE68777_series_matrix_1.txt
## x GSE68777_series_matrix_2.txt
## x GSE68777_series_matrix_3.txt
## x GSE68777_series_matrix_4.txt
## x GSE68777_series_matrix_5.txt
## x GSE68777_series_matrix_6.txt
## x GSE68777_series_matrix_7.txt
## x GSE68777_series_matrix_8.txt
## x GSE68777_series_matrix_9.txt
## x GSE68777_series_matrix_10.txt
```

```
library(GEOquery)
library(stringr)
geo <-
  getGEO("GSE68777", filename = "Output/GSE68777_series_matrix.txt.gz", getGPL = FALSE)
pdat = pData(geo)

# Format rownames to match rg_68777 sample names
rownames(pdat) = paste(pdat$geo_accession, pdat$title, sep = "_")

# Extract group and sex metadata
pdat$group = as.factor(str_remove(pdat$characteristics_ch1.1, "^diagnosis: "))
pdat$sex = as.factor(str_remove(pdat$characteristics_ch1.2, "^Sex: "))
pdat = pdat[, c("group", "sex")]

# Make sure rows are in the same order as samples from the rgset
pdat = pdat[sampleNames(rg_68777), ]

# Add pData to rg_68777
pData(rg_68777) = as(pdat, "DataFrame")
```

Let’s check the data:

```
# Check the matrix of counts
assay(rg_68777)[1:4, 1:2]
```

```
##          GSM1681154_5958091019_R03C02 GSM1681155_5935446005_R05C01
## 10600313                          234                          283
## 10600322                         6197                         7000
## 10600328                         2220                         2390
## 10600336                         1326                         1339
```

```
# Check pheno data
pData(rg_68777)[1:4, ]
```

```
## DataFrame with 4 rows and 2 columns
##                                 group      sex
##                              <factor> <factor>
## GSM1681154_5958091019_R03C02    Mania   Female
## GSM1681155_5935446005_R05C01    Mania   Female
## GSM1681156_5958091020_R01C01    Ctr     Male
## GSM1681157_5958091020_R03C02    Ctr     Female
```

9.3 Dataset characteristics

How many features are in *rg_68777*?

```
length(rownames(rg_68777))
```

```
## [1] 622399
```

How many samples are in *rg_68777*?

```
length(colnames(rg_68777))
```

```
## [1] 40
```

How many samples are from patients in the Mania group?

```
table(pData(rg_68777)[, 1])
```

```
##
##   Ctr Mania
##    20    20
```

How many samples from the Mania group are Male?

```
table(pData(rg_68777)[, 1] == "Mania" & pData(rg_68777)[, 2] == "Male")
```

```
##
## FALSE  TRUE
##    32     8
```

9.4 Preprocessing

Let's preprocess the data:

```
rg_68777 = preprocessIllumina(rg_68777)
```

Now let's assign locations to each CpG:

```
rg_68777 = mapToGenome(rg_68777)
```

Compute methylation values:

```
rg_68777 = ratioConvert(rg_68777, type = "Illumina")
```

What is the class of rg_68777 after these processing steps?

```
class(rg_68777)[1]
```

```
## [1] "GenomicRatioSet"
```

9.5 Beta values

Let's get the beta values for *rg_68777*:

```
beta = getBeta(rg_68777)
```

What is the mean value of beta at "cg14008030" across all samples?

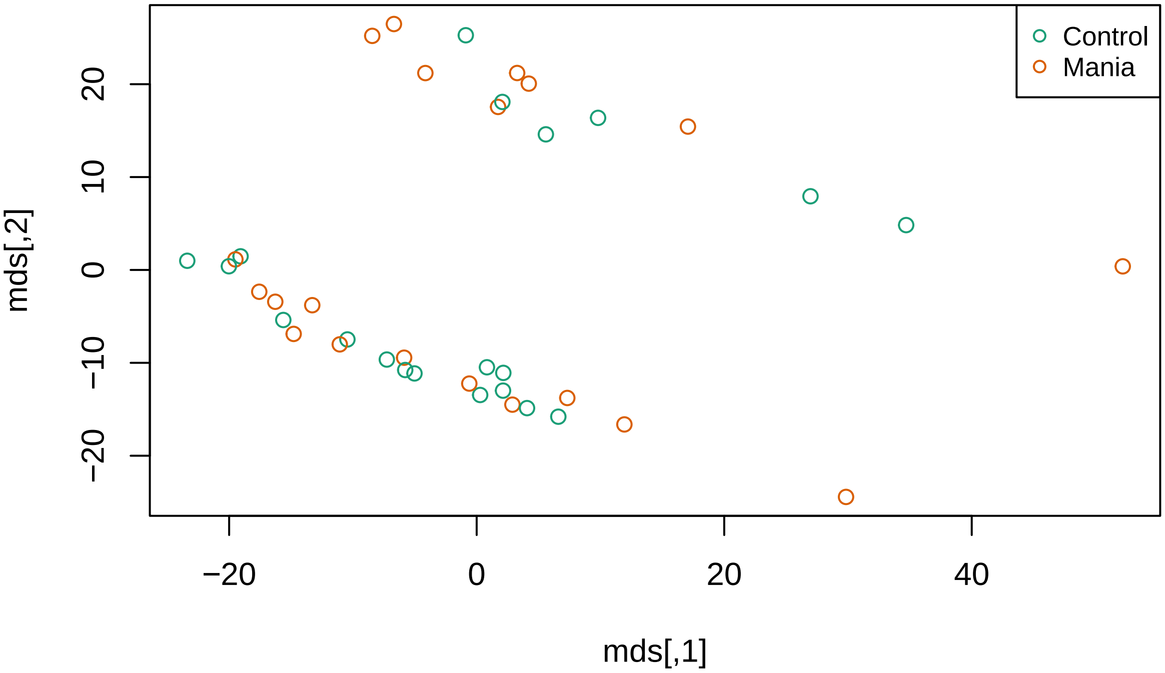
```
mean(beta[which(rownames(beta) == "cg14008030", ), ])
```

```
## [1] 0.6697852
```

9.6 MDS plot

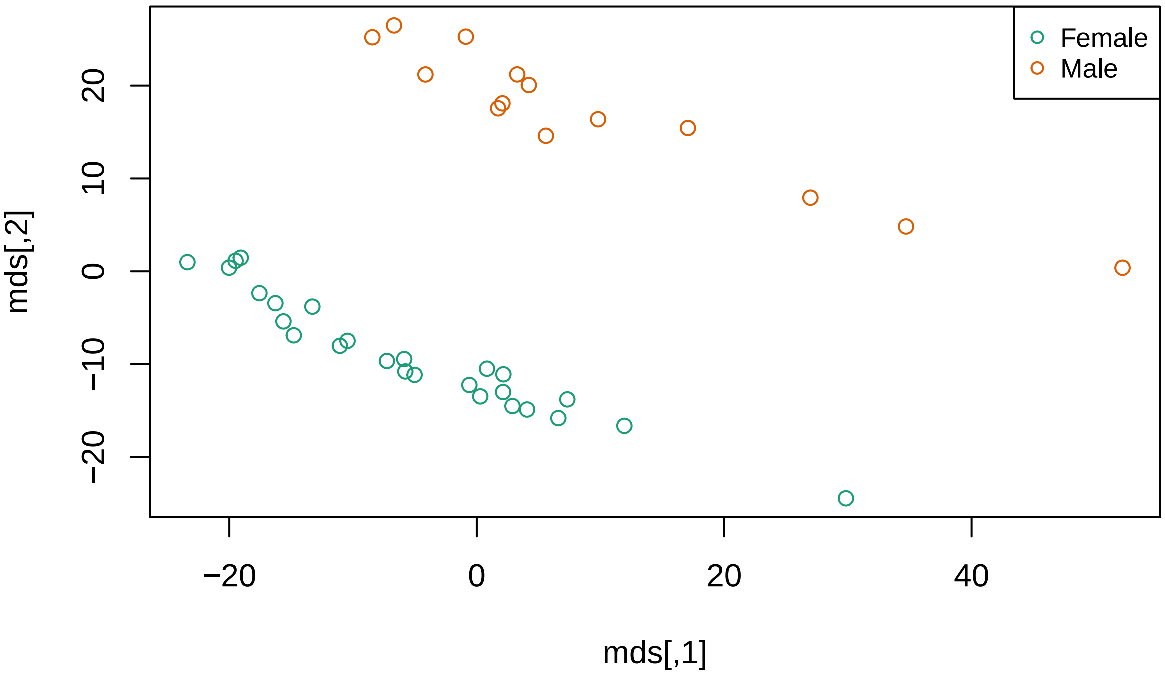
Make an MDS plot of the matrix of beta values, coloring the points by group:

```
mds <- cmdscale(dist(t(beta)))
group <- as.factor(pData(rg_68777)[, 1])
plot(mds, col = group)
legend(
  x = "topright",
  legend = c("Control", "Mania"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)
```



Make an MDS plot of the matrix of beta values, coloring the points by gender:

```
mds <- cmdscale(dist(t(beta)))
gender <- as.factor(pData(rg_68777)[, 2])
plot(mds, col = gender)
legend(
  x = "topright",
  legend = c("Female", "Male"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)
```



As you can see, female samples are clustered together in the lower left. There is no apparent clustering by group.

9.7 limma effect sizes

Let's perform a site-by-site (CpG-by-CpG) analysis for differentially methylated CpGs.

Which group would be considered the reference group?

```
levels(group)[1]
```

```
## [1] "Ctr"
```

Now we do the analysis:

```
library(limma)

# Create design matrix
group = as.factor(pData(rg_68777)[, 1])
X = model.matrix( ~ group)

# Obtain effect sizes and pvals with limma
fit = lmFit(beta, X)
```

Which CpG site has the largest positive effect size for the group variable?

```
index <- which.max(fit$coef[, 2])
print(index)
```

```
## cg12434901
##     410267
```

On which chromosome is this CpG?

```
seqnames(granges(rg_68777)[which(rownames(rg_68777) == names(index))])
```

```
## factor-Rle of length 1 with 1 run
##   Lengths:     1
##   Values : chr17
## Levels(24): chr1 chr2 chr3 chr4 chr5 chr6 ... chr19 chr20 chr21 chr22 chrX chrY
```

In what position on that chromosome is this CpG?

```r
start(granges(rg_68777)[which(rownames(rg_68777) == names(index))])
```

```
## [1] 61602292
```

What is the effect size of the group variable at that CpG?

```r
fit$coef[, 2][index]
```

```
## cg12434901
##  0.3618665
```

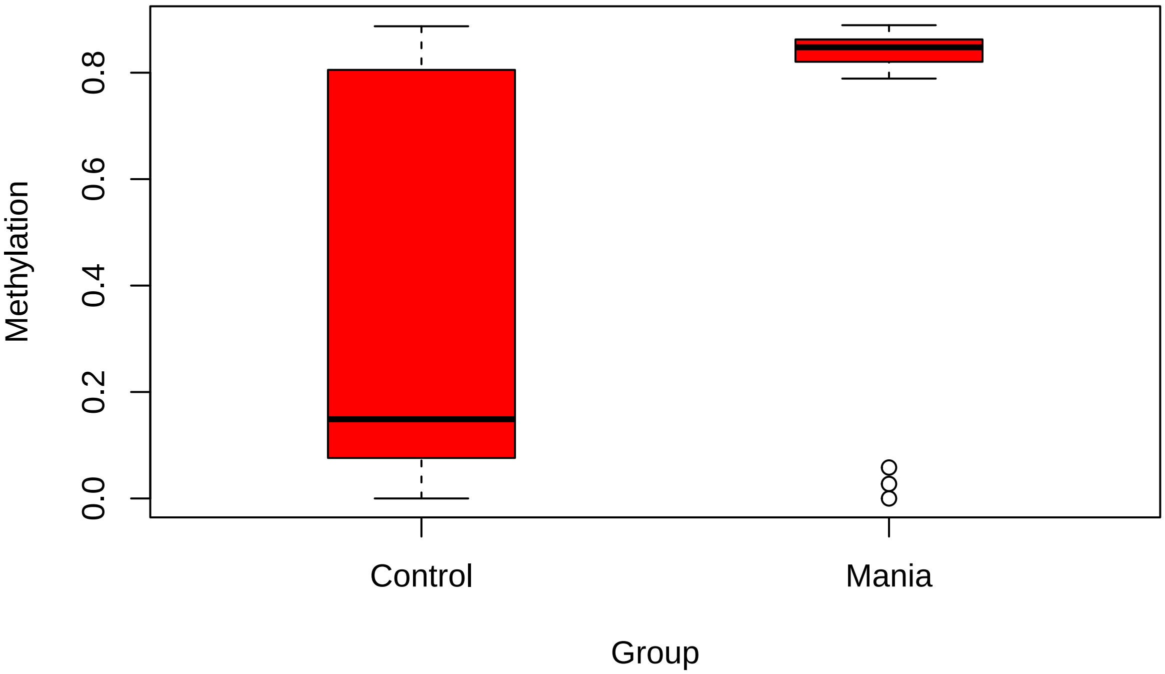
9.8 Boxplot of methylation levels at a specific CpG

Make a boxplot of methylation values at the CpG site with the largest positive effect size, grouping by mania status:

```r
# Check if data order is correct
table(rownames(pData(rg_68777)) == colnames(beta))
```

```
##
## TRUE
##   40
```

```r
# Specify groups
group = as.factor(pData(rg_68777)[, 1])

# Make boxplot
boxplot(
  beta[index,] ~ group,
  xlab = "Group",
  ylab = "Methylation",
  names = c("Control", "Mania"),
  boxwex = 0.4,
  col = "red"
)
```



As you can see, this CpG is more highly methylated in mania samples than in control samples, while control samples have more variable methylation status at this CpG than mania samples.

9.9 limma p-values and q-values

Now let's calculate the p-values for individual CpGs:

```
eb <- eBayes(fit)
pvals <- eb$p.value[, 2]
```

Convert p-values to q-values:

```
library(qvalue)
qvals <- qvalue(pvals)$qvalues
```

How many CpGs genes have q-values smaller than 0.05?

```
table(qvals < 0.05)[2]
```

```
## TRUE
##    9
```

9.10 Comparing limma results to the publication

Get the names of the CpGs with q-values below 0.05:

```
cpg_q <- names(qvals[qvals < 0.05])
```

The original paper reported a single differentially methylated CpG on chromosome 15, but they did not adjust for multiple testing.

How many of the CpGs with a q-value below 0.05 are on chromosome 15?

```
chr = 1:length(cpg_q)
for (i in 1:length(cpg_q)){
  chr[i] = seqnames(granges(rg_68777)[which(rownames(rg_68777) == cpg_q[i])])
}
chr
```

```
## [1] "chr2"  "chr3"  "chr6"  "chr7"  "chr16" "chr16" "chr17" "chr17" "chr19"
```

9.11 Number of nearby CpGs

Find all the CpGs within 10000 basepairs of the location of the CpG site with the largest positive effect size:

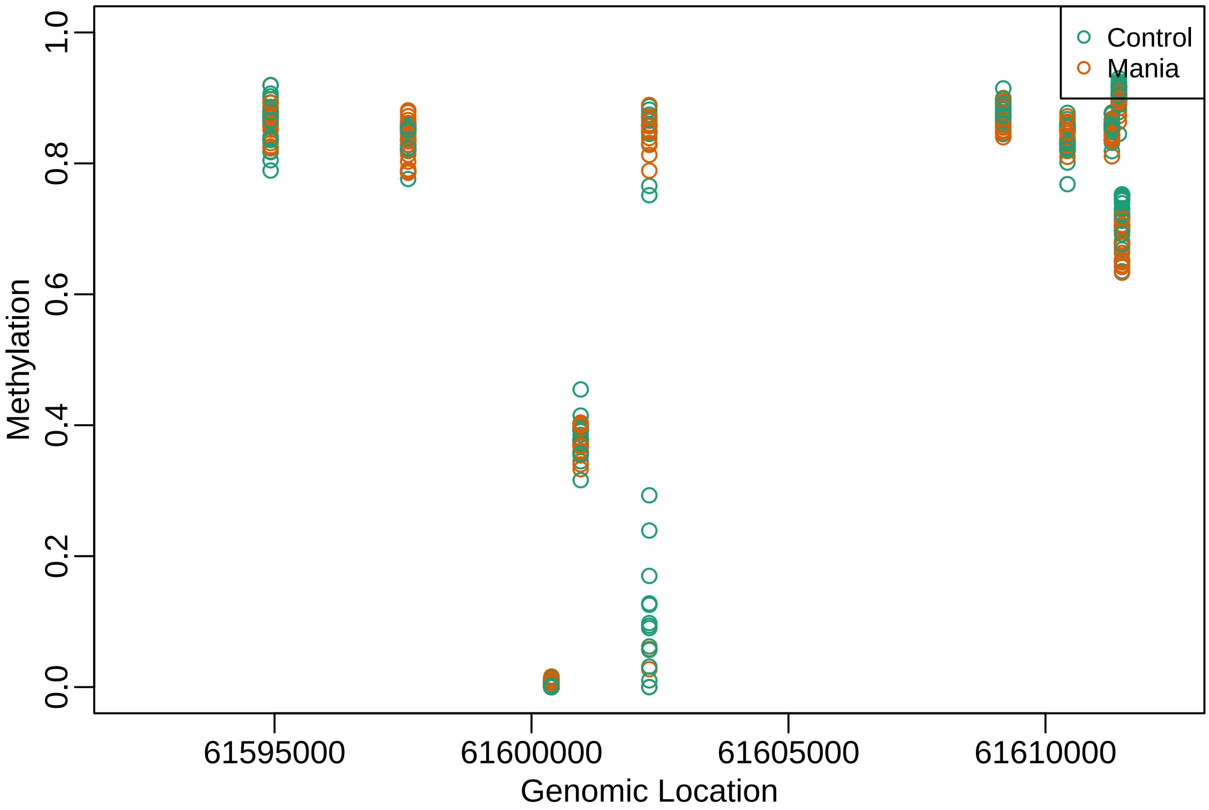
```
r = granges(rg_68777)[index] + 10000
cpg_index = which(granges(rg_68777) %over% r)
length(cpg_index)
```

```
## [1] 11
```

9.12 Plot of CpG methylation at surrounding CpGs

Now create a plot showing the methylation values for all samples for the CpGs identified in previously and use color to distinguish mania from control:

```
# Get the positions of those CpGs
pos = start(granges(rg_68777))[cpg_index]

# Define the limit of the region to include in the plot
xlim = range(c(pos, start(r), end(r)))

# Get the beta values for the CpGs in our specific region
y = beta[cpg_index, ]

# Color data points by tissue type
cols = as.numeric(as.factor(pData(rg_68777)[, 1]))

# Plot
mypar(1, 1)
matplot(
  pos,
  y,
  col = as.numeric(cols),
  xlim = xlim,
  ylim = c(0, 1),
  xlab = "Genomic Location",
  ylab = "Methylation",
  pch = 1
)
legend(
  x = "topright",
  legend = c("Control", "Mania"),
  col = c(1, 2),
  pch = 1,
  cex = 0.8
)
```



As it is obvious, only the CpG with the highest effect size appears differentially methylated, with no apparent differential methylation within 10000 bp.

9.13 DMRs

Now we are going to explicitly search for DMRs. We will use permutation to assess statistical significance. Because the function is slow, we will restrict our analysis to chromosome 17:

```
# Set to chr 17
index <- which(seqnames(rg_68777) == "chr17")
chr17 <- rg_68777[index, ]

# Create design matrix
group <- as.factor(pData(rg_68777)[, 1])
X <- model.matrix( ~ group)

# Extract methylation values
set.seed(1)
res <- bumphunter(chr17, X, cutoff = 0.1, B = 100, verbose = FALSE)
res$tab[1:4, c(1, 2, 3, 4, 9, 13)]
```

```
##      chr    start      end      value L p.valueArea
## 11 chr17 61602292 61602292  0.3618665 1  0.02165507
## 3  chr17  7347123  7347123  0.3213568 1  0.02938902
## 15 chr17   213655   213743 -0.1642609 3  0.01005414
## 23 chr17 64562000 64562000 -0.2398476 1  0.07269915
```

How many bumps (potential DMRs) were found on chromosome 17?

```
dim(res$tab)[1]
```

```
## [1] 27
```

How many DMRs on chromosome 17 have a p-value below 0.05?

```
table(res$tab$p.value <= 0.05)[2]
```

```
## TRUE
##    6
```

How many DMRs on chromosome 17 have a family-wise error rate less than 0.1?

```
table(res$tab$fwer <= 0.1)[2]
```

```
## TRUE
##    1
```