Pipeline for methylation assay

Stefano Roncelli

16/06/20

Student N° 26 Address 10633381 p-value threshold: 0.01 normalization preprocessSWAN Mann_whitney test

Load raw data with minfi and create an object called RGset storing the RGChannelSet object

```
setwd('.')
suppressMessages(library(minfi))
suppressMessages(library(magrittr))
baseDir <- ('./Input_data')
targets <- read.metharray.sheet(baseDir)
```

```
## [1] "./Input_data/Samplesheet_report_2020.csv"
```

```
RGset <- read.metharray.exp(targets = targets)
save(RGset, file = "RGset.RData")
RGset
```

```
## class: RGChannelSet
## dim: 622399 8
## metadata(0):
## assays(2): Green Red
## rownames(622399): 10600313 10600322 ... 74810490 74810492
## rowData names(0):
## colnames(8): 5775278051_R01C01 5775278051_R04C02 ... 5930514035_R04C02
##   5930514035_R06C02
## colData names(7): Sample_Name Group ... Basename filenames
## Annotation
##   array: IlluminaHumanMethylation450k
##   annotation: ilmn12.hg19
```

Step 2

Create the dataframes Red and Green to store the red and green fluorescences respectively

```
Red <- data.frame(getRed(RGset))
Green <- data.frame(getGreen(RGset))
```

Step 3

Fill the following table: what are the Red and Green fluorescences for the address assigned to you? Optional: check in the manifest file if the address corresponds to a Type I or a Type II probe and, in case of Type I probe, report its color.

For the optional procedure we will create two dataframes `type_I` and `type_II`, which will be reused later on for the normalization.

```
type_I <- getProbeInfo(RGset, type = 'I')
type_II <- getProbeInfo(RGset, type = 'II')
type_I[type_I$AddressA == 10633381,]
```

```
## DataFrame with 0 rows and 8 columns
```

```
type_I[type_I$AddressB == 10633381,]
```

```
## DataFrame with 1 row and 8 columns
##          Name    AddressA    AddressB       Color       NextBase
##   <character> <character> <character> <character> <DNAStringSet>
## 1  cg03868159    21656441    10633381         Red               A
##                 ProbeSeqA               ProbeSeqB      nCpG
##            <DNAStringSet>          <DNAStringSet> <integer>
## 1 CTAAACATCC...AACTATACCA CTAAACGTCC...AACTATACCG         2
```

```
type_II[type_II$AddressA == 10633381,]
```

```
## DataFrame with 0 rows and 4 columns
```

```
Red[rownames(Red) == '10633381',]
```

```
##          X5775278051_R01C01 X5775278051_R04C02 X5775278078_R02C01
## 10633381               1852               1694               1354
##          X5775278078_R05C01 X5775278078_R05C02 X5930514034_R01C02
## 10633381               1091               1131                796
##          X5930514035_R04C02 X5930514035_R06C02
## 10633381                894               1149
```

```
Green[rownames(Green) == '10633381',]
```

```
##          X5775278051_R01C01 X5775278051_R04C02 X5775278078_R02C01
## 10633381                458                631                358
##          X5775278078_R05C01 X5775278078_R05C02 X5930514034_R01C02
## 10633381                396                424                302
##          X5930514035_R04C02 X5930514035_R06C02
## 10633381                354                479
```

We can see it's a type I infinium with the Red channel.

Sample	Row	Column	Red Intensity	Green Intensity	Type	Color
5775278051	1	1	1852	458	I	Red
5775278051	4	2	1694	631	I	Red
5775278078	2	1	1354	358	I	Red
5775278078	5	1	1091	396	I	Red
5775278078	5	2	1131	424	I	Red
5930514034	1	2	796	302	I	Red
5930514035	4	2	894	354	I	Red
5930514035	6	2	1149	479	I	Red

Step 4

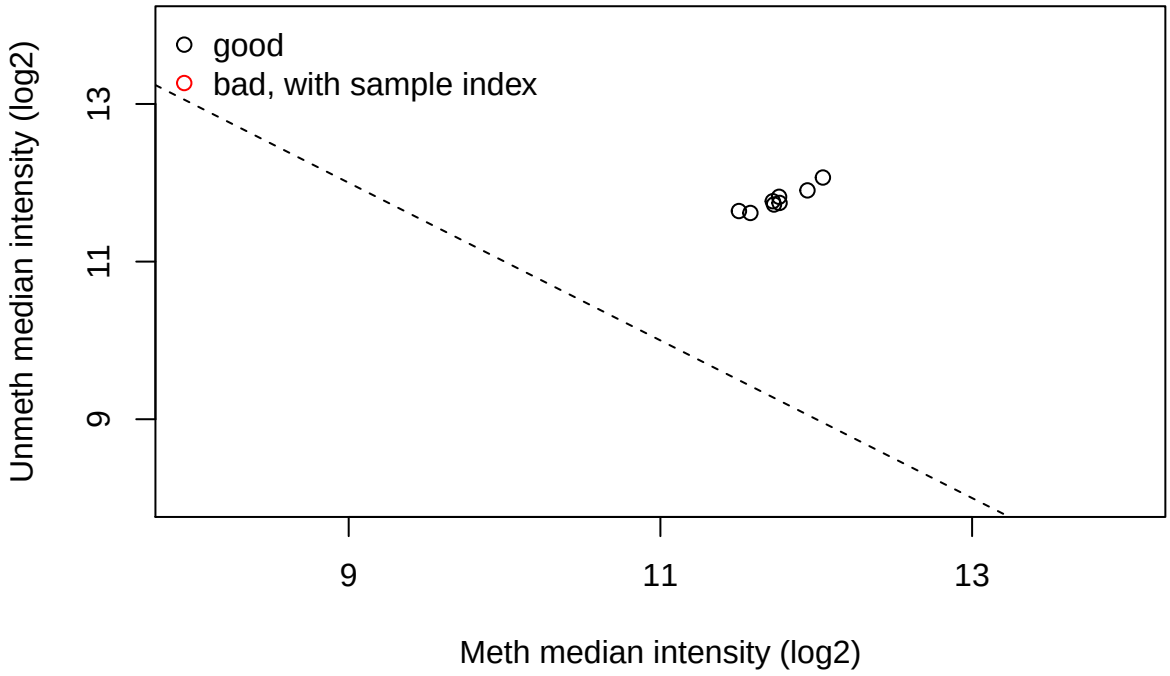
Create the object MSet.raw For the creation of the MSet.raw we use the preprocessRaw function

```
MSet.raw <- preprocessRaw(RGset)
```

Step 5

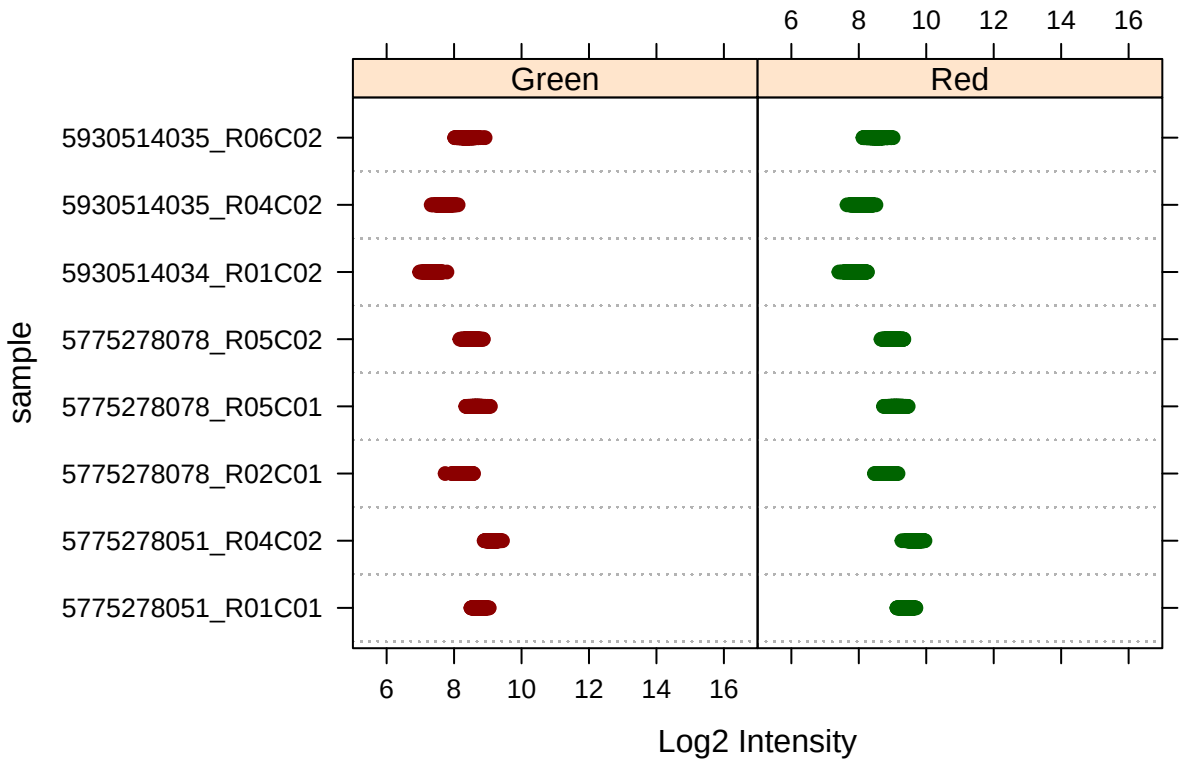
Perform the following quality checks and provide a brief comment to each step: - QCplot

```
qc <- getQC(MSet.raw)
plotQC(qc)
```



- check the intensity of negative controls using minfi

```
controlStripPlot(RGset, controls = "NEGATIVE")
```



- calculate detection pValues; for each sample, how many probes have a detection p-value higher than the threshold assigned to each student? The function to use is detectionP, which takes as input the RGset.

```
detection_p_value <- detectionP(RGset)
dim(detection_p_value)
```

```
## [1] 485512      8
```

```
failed_probes <- detection_p_value > 0.01
table(failed_probes)
```

```
## failed_probes
##   FALSE    TRUE
## 3881919    2177
```

```
summary(failed_probes)
```

```
##  5775278051_R01C01 5775278051_R04C02 5775278078_R02C01 5775278078_R05C01
##  Mode :logical     Mode :logical     Mode :logical     Mode :logical
##  FALSE:485189      FALSE:485252      FALSE:485200      FALSE:485027
##  TRUE :323         TRUE :260         TRUE :312         TRUE :485
##  5775278078_R05C02 5930514034_R01C02 5930514035_R04C02 5930514035_R06C02
##  Mode :logical     Mode :logical     Mode :logical     Mode :logical
##  FALSE:485047      FALSE:485389      FALSE:485452      FALSE:485363
##  TRUE :465         TRUE :123         TRUE :60          TRUE :149
```

The following table summarizes the failed positions

Sample	Group	Slide	Row	Col	Failed probes (p-value > 0.01)
1020	DS	5775278051	1	1	323

Sample	Group	Slide	Row	Col	Failed probes (p-value > 0.01)
1036	DS	5775278051	4	2	260
3038	WT	5775278078	2	1	312
3042	WT	5775278078	5	1	485
3052	WT	5775278078	5	2	465
1016	DS	5930514034	1	2	123
1029	DS	5930514035	4	2	60
3029	WT	5930514035	6	2	149

Step 6

Calculate raw beta and M values and plot the densities of mean methylation values, dividing the samples in DS and WT (suggestion: subset the beta and M values matrixes in order to retain DS or WT subjects and apply the function mean to the 2 subsets).

For the retrieval of the β and M values we use the `getBeta` and `getM` functions respectively.

```
beta_value <- getBeta(MSet.raw)
summary(beta_value)
```

```
##  5775278051_R01C01 5775278051_R04C02 5775278078_R02C01 5775278078_R05C01
##  Min.   :0.01745   Min.   :0.01828   Min.   :0.01128   Min.   :0.01133
##  1st Qu.:0.09198   1st Qu.:0.09763   1st Qu.:0.08523   1st Qu.:0.09360
##  Median :0.60089   Median :0.60543   Median :0.60102   Median :0.60714
##  Mean   :0.47988   Mean   :0.48371   Mean   :0.48459   Mean   :0.49043
##  3rd Qu.:0.79643   3rd Qu.:0.80112   3rd Qu.:0.81373   3rd Qu.:0.81985
##  Max.   :1.00000   Max.   :0.98415   Max.   :1.00000   Max.   :0.98884
##  NA's   :1         NA's   :2         NA's   :3         NA's   :1
##  5775278078_R05C02 5930514034_R01C02 5930514035_R04C02 5930514035_R06C02
##  Min.   :0.01178   Min.   :0.00000   Min.   :0.00000   Min.   :0.008389
##  1st Qu.:0.09452   1st Qu.:0.06721   1st Qu.:0.07456   1st Qu.:0.080286
##  Median :0.60643   Median :0.58693   Median :0.61593   Median :0.616594
##  Mean   :0.49042   Mean   :0.47988   Mean   :0.49289   Mean   :0.494334
##  3rd Qu.:0.81816   3rd Qu.:0.82893   3rd Qu.:0.84495   3rd Qu.:0.843440
##  Max.   :0.98877   Max.   :1.00000   Max.   :1.00000   Max.   :1.000000
##  NA's   :1         NA's   :10        NA's   :7         NA's   :4
```

```
M_value <- getM(MSet.raw)
summary(M_value)
```

```
##  5775278051_R01C01 5775278051_R04C02 5775278078_R02C01 5775278078_R05C01
##  Min.   :-5.8153   Min.   :-5.7467   Min.   :-6.4535   Min.   :-6.4468
##  1st Qu.:-3.3034   1st Qu.:-3.2084   1st Qu.:-3.4241   1st Qu.:-3.2756
##  Median : 0.5903   Median : 0.6177   Median : 0.5911   Median : 0.6280
##  Mean   :    Inf   Mean   :-0.3158   Mean   :    Inf   Mean   :-0.2778
##  3rd Qu.: 1.9680   3rd Qu.: 2.0101   3rd Qu.: 2.1271   3rd Qu.: 2.1861
##  Max.   :    Inf   Max.   : 5.9560   Max.   :    Inf   Max.   : 6.4698
##  NA's   :1         NA's   :2         NA's   :3         NA's   :1
##  5775278078_R05C02 5930514034_R01C02 5930514035_R04C02 5930514035_R06C02
##  Min.   :-6.3903   Min.   :   -Inf   Min.   :   -Inf   Min.   :-6.8851
##  1st Qu.:-3.2600   1st Qu.:-3.7947   1st Qu.:-3.6337   1st Qu.:-3.5180
##  Median : 0.6237   Median : 0.5068   Median : 0.6814   Median : 0.6854
##  Mean   :-0.2818   Mean   :    NaN   Mean   :    NaN   Mean   :    Inf
##  3rd Qu.: 2.1697   3rd Qu.: 2.2767   3rd Qu.: 2.4462   3rd Qu.: 2.4296
```

```
##  Max.   : 6.4600   Max.   :    Inf   Max.   :    Inf   Max.   :    Inf
##  NA's   :1         NA's   :10        NA's   :7         NA's   :4
```

We now subset `beta_value` and `M_value` to obtain the Wild-Type (`WT`) and Down-Syndrome (`DS`).

```
DS_status <- targets$Group == 'DS'
summary(DS_status)
```

```
##    Mode   FALSE    TRUE
## logical       4       4
```

```
beta_value_DS <- beta_value[, DS_status]
beta_value_WT <- beta_value[, !DS_status]
M_value_DS <- M_value[, DS_status]
M_value_WT <- M_value[, !DS_status]
```

Computing the mean for β and M subsets and both groups

```
par(mfrow = c(1, 2))

beta_value_WT %>%
        apply(1, mean, na.rm = T) %>%
        density() %T>%
        plot(main = expression(paste("Density of ", beta, " values"), col = "green"))
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (485512 obs.);   Bandwidth 'bw' = 0.02276
##
##        x                  y
##  Min.   :-0.05486   Min.   :0.000348
##  1st Qu.: 0.22269   1st Qu.:0.303398
##  Median : 0.50025   Median :0.486100
##  Mean   : 0.50025   Mean   :0.899832
##  3rd Qu.: 0.77781   3rd Qu.:1.191970
##  Max.   : 1.05536   Max.   :3.579017
```

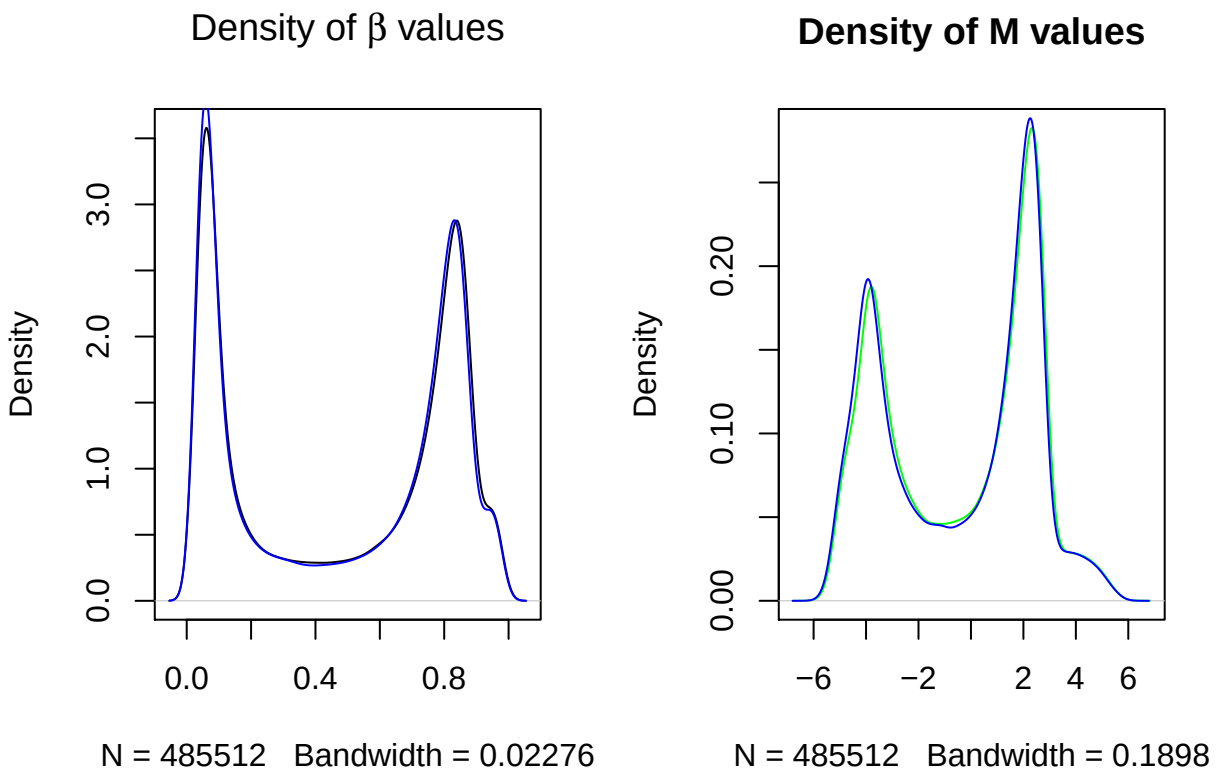
```
beta_value_DS %>%
        apply(1, mean, na.rm = T) %>%
        density() %T>%
        lines(col = "blue")
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (485512 obs.);   Bandwidth 'bw' = 0.02276
##
##        x                  y
##  Min.   :-0.05419   Min.   :0.000367
##  1st Qu.: 0.22279   1st Qu.:0.293884
##  Median : 0.49977   Median :0.476817
##  Mean   : 0.49977   Mean   :0.901706
##  3rd Qu.: 0.77675   3rd Qu.:1.160712
##  Max.   : 1.05373   Max.   :3.808760
```

```
M_value_WT %>%
        apply(1, mean, na.rm = T) %>%
        density() %T>%
        plot(main = "Density of M values", col = "green")
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (485512 obs.);   Bandwidth 'bw' = 0.1898
##
##        x                   y
##  Min.   :-6.78342   Min.   :9.000e-08
##  1st Qu.:-3.37695   1st Qu.:2.048e-02
##  Median : 0.02952   Median :4.978e-02
##  Mean   : 0.02952   Mean   :7.332e-02
##  3rd Qu.: 3.43600   3rd Qu.:1.060e-01
##  Max.   : 6.84247   Max.   :2.826e-01
```

```
M_value_DS %>%
        apply(1, mean, na.rm = T) %>%
        density() %T>%
        lines(col = "blue")
```




```
##
## Call:
##  density.default(x = .)
##
## Data: . (485512 obs.);   Bandwidth 'bw' = 0.1914
##
```

```
##        x                    y
##  Min.   :-6.803023   Min.   :0.0000001
##  1st Qu.:-3.405254   1st Qu.:0.0210921
##  Median :-0.007486   Median :0.0480386
##  Mean   :-0.007486   Mean   :0.0735049
##  3rd Qu.: 3.390282   3rd Qu.:0.1060653
##  Max.   : 6.788050   Max.   :0.2884387
```

Step 7

Normalize the data using the function assigned to each student and compare raw data and normalized data. Produce a plot with 6 panels in which, for both raw and normalized data, you show the density plots of beta mean values according to the chemistry of the probes, the density plot of beta standard deviation values according to the chemistry of the probes and the boxplot of beta values. Provide a short comment regarding the changes you observe.

Producing the plots

```
beta_type_I <- beta_value[rownames(beta_value) %in% type_I$Name, ]
beta_type_II <- beta_value[rownames(beta_value) %in% type_II$Name, ]

# Normalization with preprocessSWAN
beta_norm <-
  RGset %>%
  preprocessSWAN() %>%
    getBeta()

beta_type_I_norm <- beta_norm[rownames(beta_norm) %in% type_I$Name, ]
beta_type_II_norm <- beta_norm[rownames(beta_norm) %in% type_II$Name, ]

par(mfrow=c(2,3))
# Plotting raw beta for type I
beta_type_I %>%
  apply(1, mean, na.rm = T) %>%
  density() %T>%
  plot(main = expression(paste("Raw ", beta)), col = "blue")
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (135476 obs.);   Bandwidth 'bw' = 0.03279
##
##        x                   y
##  Min.   :-0.08424   Min.   :0.001912
##  1st Qu.: 0.20778   1st Qu.:0.181587
##  Median : 0.49980   Median :0.308103
##  Mean   : 0.49980   Mean   :0.855217
##  3rd Qu.: 0.79182   3rd Qu.:1.014959
##  Max.   : 1.08384   Max.   :5.289593
```

```
# Plotting raw beta for type II
beta_type_II %>%
        apply(1, mean, na.rm = T) %>%
        density() %T>%
```

```
        lines(col = "red")
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (350036 obs.);   Bandwidth 'bw' = 0.02233
##
##        x                  y
##  Min.   :-0.04059   Min.   :0.000004
##  1st Qu.: 0.22054   1st Qu.:0.338257
##  Median : 0.48167   Median :0.474378
##  Mean   : 0.48167   Mean   :0.956440
##  3rd Qu.: 0.74280   3rd Qu.:1.292035
##  Max.   : 1.00393   Max.   :3.726014
```

```
# Plotting raw standard deviation for type I
beta_type_I %>%
        apply(1, sd, na.rm = T) %>%
        density() %T>%
        plot(main = expression(paste("Raw standard deviation for ", beta, " values")), col = "blue")
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (135476 obs.);   Bandwidth 'bw' = 0.0008007
##
##        x                   y
##  Min.   :-0.001218   Min.   : 0.00000
##  1st Qu.: 0.099011   1st Qu.: 0.00489
##  Median : 0.199240   Median : 0.02998
##  Mean   : 0.199240   Mean   : 2.49134
##  3rd Qu.: 0.299469   3rd Qu.: 0.41691
##  Max.   : 0.399698   Max.   :82.90717
```

```
# Plotting raw standard deviation for type II
beta_type_II %>%
        apply(1, sd, na.rm = T) %>%
        density() %T>%
        lines(col = "red")
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (350036 obs.);   Bandwidth 'bw' = 0.0008953
##
##        x                    y
##  Min.   :-0.0004834   Min.   : 0.00003
##  1st Qu.: 0.0984307   1st Qu.: 0.01168
##  Median : 0.1973447   Median : 0.04878
##  Mean   : 0.1973447   Mean   : 2.52539
##  3rd Qu.: 0.2962588   3rd Qu.: 0.26989
##  Max.   : 0.3951729   Max.   :35.12776
```

```r
# Plotting bowplot for raw beta
boxplot(beta_value)

# Plotting the normlized beta mean for type
beta_type_I_norm %>%
        apply(1, mean, na.rm = T) %>%
        density() %T>%
        plot(col = "blue", main = "normalized beta")
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (135476 obs.);   Bandwidth 'bw' = 0.03115
##
##        x                 y
##  Min.   :-0.0773   Min.   :0.000853
##  1st Qu.: 0.2091   1st Qu.:0.206406
##  Median : 0.4955   Median :0.343269
##  Mean   : 0.4955   Mean   :0.872076
##  3rd Qu.: 0.7818   3rd Qu.:1.034059
##  Max.   : 1.0682   Max.   :5.584347
```

```r
# Plotting the normalized beta for type II probes
beta_type_II_norm %>%
        apply(1, mean, na.rm = T) %>%
        density() %T>%
        lines(col = "red")
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (350036 obs.);   Bandwidth 'bw' = 0.02388
##
##        x                 y
##  Min.   :-0.0488   Min.   :0.000049
##  1st Qu.: 0.2209   1st Qu.:0.294749
##  Median : 0.4906   Median :0.414114
##  Mean   : 0.4906   Mean   :0.926140
##  3rd Qu.: 0.7602   3rd Qu.:1.152058
##  Max.   : 1.0299   Max.   :4.350685
```

```r
# Plotting the normalized standard deviation for type I probes
beta_type_I_norm %>%
        apply(1, sd, na.rm = T) %>%
        density() %T>%
        plot(col = "blue", main = "normalized standard deviation")
```

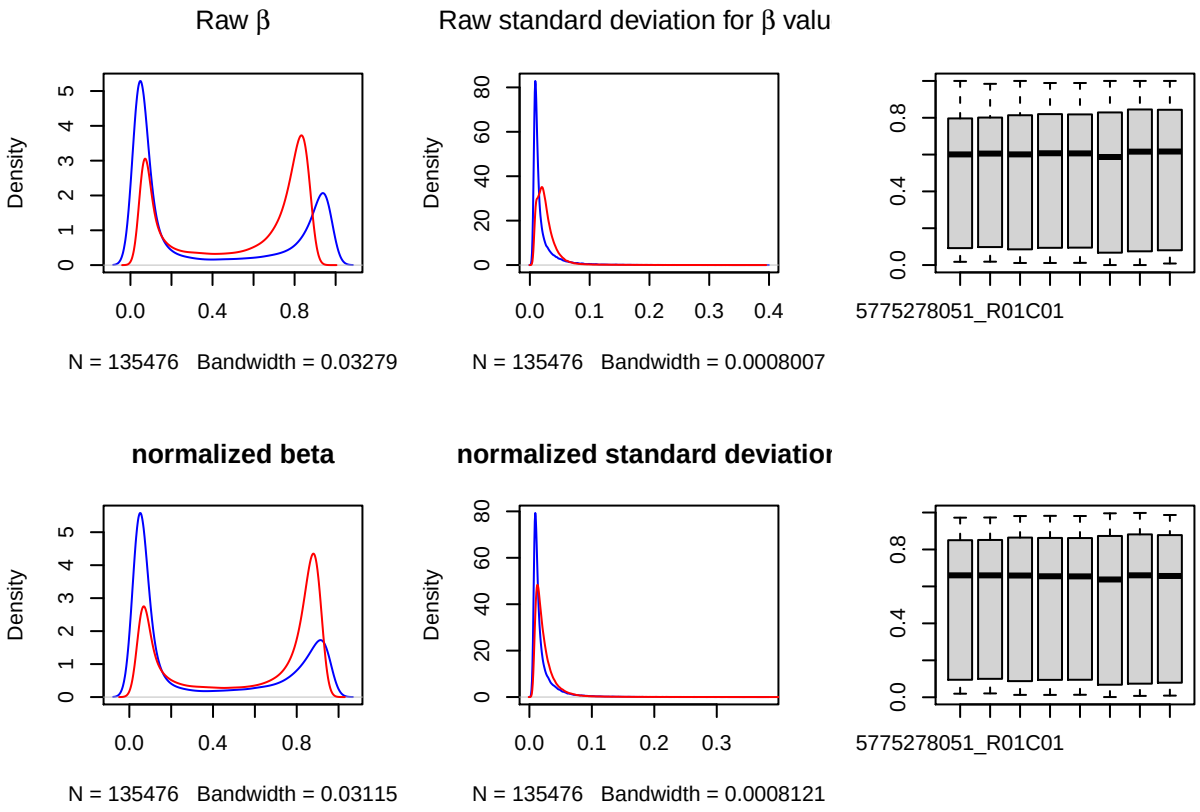
```
##
## Call:
##  density.default(x = .)
##
## Data: . (135476 obs.);   Bandwidth 'bw' = 0.0008121
##
```

```
##        x                    y
##  Min.   :-0.0003721   Min.   : 0.00000
##  1st Qu.: 0.0955699   1st Qu.: 0.00454
##  Median : 0.1915118   Median : 0.02994
##  Mean   : 0.1915118   Mean   : 2.60487
##  3rd Qu.: 0.2874537   3rd Qu.: 0.39154
##  Max.   : 0.3833956   Max.   :79.32855
```

```
# Plotting the normalized standard deviation for type II probes
beta_type_II_norm %>%
        apply(1, sd, na.rm = T) %>%
        density() %T>%
        lines(col = "red")
```

```
##
## Call:
##  density.default(x = .)
##
## Data: . (350036 obs.);   Bandwidth 'bw' = 0.0008485
##
##        x                     y
##  Min.   :-0.001117   Min.   : 0.00003
##  1st Qu.: 0.103050   1st Qu.: 0.01380
##  Median : 0.207216   Median : 0.04005
##  Mean   : 0.207216   Mean   : 2.39719
##  3rd Qu.: 0.311383   3rd Qu.: 0.27216
##  Max.   : 0.415549   Max.   :48.35695
```

```
# Plotting boxplot for normalized beta value
boxplot(beta_norm)
```



Step 8 Perform a PCA on the beta matrix generated in step 7. Comment the plot.

Step 9 Using the matrix of normalized beta values generated in step 7, identify differentially methylated probes between group DS and group WT using the functions assigned to each student. Note; it can take several minutes; if you encounter any problem you can run the differential methylated analysis only on a subset of probes (for example those on chromosome 1, 18 and 21)

Step 10 Apply multiple test correction and set a significant threshold of 0.05. How many probes do you identify as differentially methylated considering nominal pValues? How many after Bonferroni correction? How many after BH correction?

Step 11 Produce an heatmap of the top 100 differentially mehtylated probes

Step 12 Produce a volcano plot and a Manhattan plot of the results of differential methylation analysis

Step 13 (optional) As DS is caused by the trisomy of chromosome 21, try also to plot the density of the methylation values of the probes mapping on chromosome 21. Do you see a very clear difference between the samples? How many differentially methylated probes do you find on chromosome 21?