Working notes

Copy Number alteration estimate with infinium HM-450k and HM-27K platforms (document in preparation!)

P.BADY

16 mars 2015

Résumé

This document exposes a procedure to estimate Copy Number Alteration (CNA, gene concentration) with HM-450K and HM-27K platforms from the company Illmunia.

Table des matières

1 Motivations

The estimation of CNA from Infinium HM-450K platform require some operation to prepare the dataset such as chemistry type correction, normalization and definition of the synthetic reference. When, the calculation of the log-ration between observed and reference total intensity was done, we used simply the classical tools used to analyze the CGH information.

```
library(minfi)
library(minfiData)
library(CGHcall)
library(methyltools)
library(mixOmics)
library(ade4)
```

The procedure that we propose in this document is organized in four step as describe in the figure 1 : preprocessing, segmentation, CNA estimate for markers and CNA estimate for gene or genomic regions.

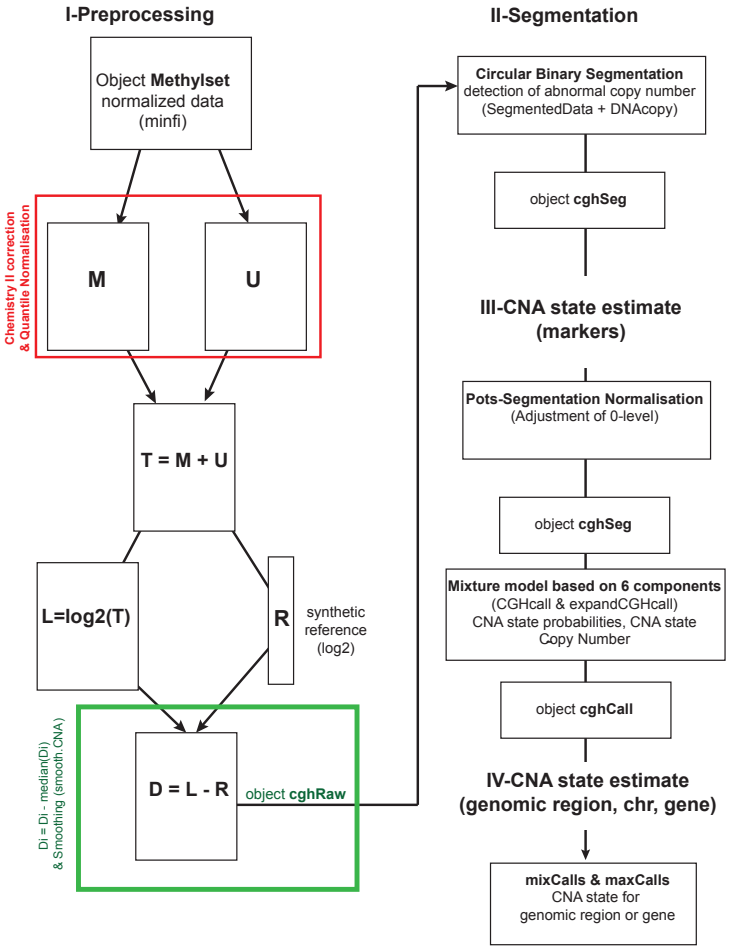

FIGURE 1 – Flow chart showing the CNA estimate based on HM-450K Infinium platform with R packages textttminfi, textttCGHcall and textttmethyltools.

.

2 Preprocessing

The DNA methylation data come from different studies and they are available in different format. When the raw data (.IDat files) were not available, we postulate that the adequate normalization was used to calculate the unmethylated and methylated signal (e.g.[2],[9], [7]). The normalization of reference samples was adjusted for each analysis. When the Raw data were available, the Red and the Green channel into a Methylated and Unmethylated signal were converted by the function preprocessIllumina from R package `minfi`, including a background correction. The reference samples were systematically normalized with the observed samples before the computation of the synthetic reference.

```
dat <- preprocessRaw(RGsetEx)
datIlmn <- preprocessIllumina(RGsetEx)
datIlmnSwan <- preprocessSWAN(RGsetEx, mSet = datIlmn)
```

3 Quality control

The quality of the hybridation are tested and the detection P-values are calculated for each probe and each samples. The results are obtained as follow :

```
pdetect1 <- detectionP(RGsetEx)
```

The Comparison of the methylated and unmethylated medians are computed for each samples and represented on a graphic to investigate the bad quality samples.

```
qc1 <- getQC(datIlmn)
plotQC(qc1)
```

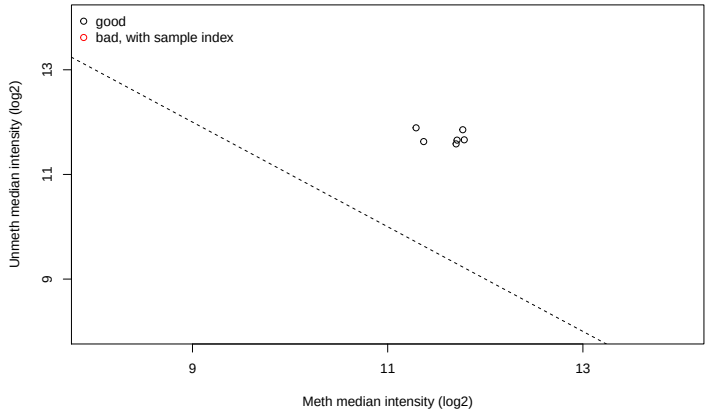

FIGURE 2 – Representation of methylated and unmethylated medians calculated by sample. This graphic is used to detect the bad quality samples.

.

The R packages `minfi` proposes a procedure to compute a predictive gender for each samples based on methylation state of Y and X chromosomes. The comparison of the observed and predicted genders can be interesting quality control.

```
gmSet1 <- mapToGenome(datIlmn)
predsex <- as.data.frame(getSex(gmSet1))
desc <- pData(datIlmn)
contisex <- table(pred=predsex[rownames(desc),3],obs=desc[,"sex"],useNA="always")
contisex
```

```
      obs
pred   F M <NA>
  F    4 0    0
  M    0 2    0
  <NA> 0 0    0
```

```
sum(diag(contisex))/sum(contisex)
```

```
[1] 1
```

```
xx <- as.data.frame(contisex)
xx$ypred <- c(1,2,3)[match(as.character(xx$pred),c(NA,"M","F"))]
xx$xobs <- c(1,2,3)[match(as.character(xx$obs),c(NA,"M","F"))]
plot(xx$xobs,xx$ypred,type="n",panel.first=c(abline(h=1:3,col="lightgray"),
    abline(v=1:3,col="lightgray")),xlab="observed sex",ylab="predicted sex",
    xlim=c(0,4),ylim=c(0,4),axes=FALSE)
symbols(xx$xobs,xx$ypred,circles=xx$Freq,bg="green2",inch=5/(sum(xx$Freq)),
        xlim=c(0,4),ylim=c(0,4),add=TRUE,fg="green2")
text(xx$xobs,xx$ypred,labels=xx$Freq,adj=c(0.5,0.5))
axis(1,at=1:3,labels=c("NA","M","F"))
axis(2,at=1:3,labels=c("NA","M","F"))
box()
title(paste("Predicted versus Observed sex (n=",sum(xx$Freq),")",sep=""))
```

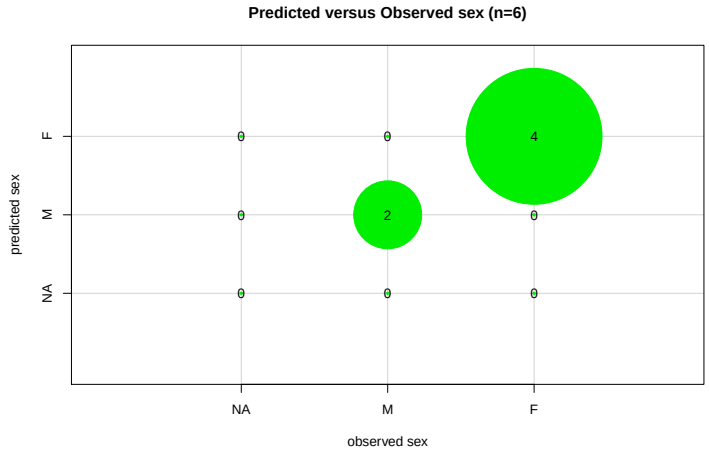

FIGURE 3 – Comparison of observed and predicted gender

.

4 Correction of the Chemistry effect

For the platform Infinium HM-450K, the two chemistry type of the probes can be generated some disturbances in the estimation of the CNA. Indeed, we observed that the variability differ among these two types probes (Figure 2). Consequently, we used a scaling factor approach to adjust the distribution of the two chemistries based directly on unmethylated and methylated preprocessed signals.

```
# boxplot representation
U0 <- getUnmeth(datIlmn)
M0 <- getMeth(datIlmn)
typeI <- data.frame(Name=getProbeInfo(datIlmn, type = "I")[, c("Name")])
typeII <- data.frame(Name=getProbeInfo(datIlmn, type = "II")[, c("Name")])
HMchemistry <- rbind(typeI, typeII)
HMchemistry$type <- c(rep("I",nrow(typeI)),rep("II",nrow(typeII)))
rownames(HMchemistry) <-  HMchemistry$Name
fac <- HMchemistry[rownames(x),"type"]
names(fac) <- rownames(HMchemistry)
fac <- fac[rownames(U0)]
### boxplot
p <- nrow(U0)
par(mfrow=c(2,3))
for(k in 1:6){
vecrnd1 <- sample(1:p,1000)
u0 <- U0[vecrnd1,k]
m0 <- M0[vecrnd1,k]
boxplot(I(u0+m0)~fac[vecrnd1], col=c("green2","orange"),
        main=paste("sample ",k,sep=""))
}
```

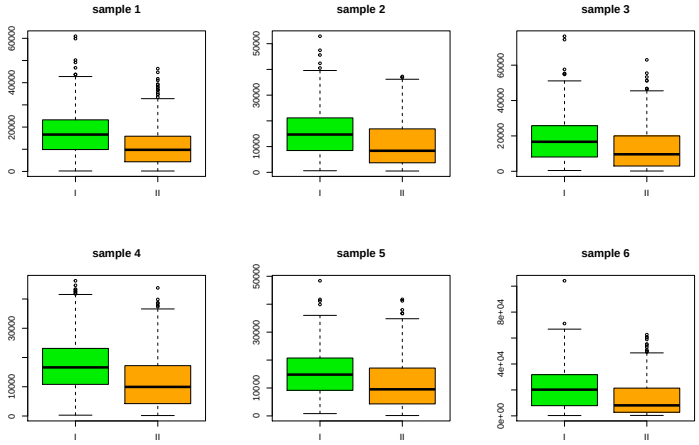

FIGURE 4 – boxplot representation the unmethylated and methylated signals for the chemistry type I and II

.

We proposed to use this strategy of correction because it's no possible to obtain systematically the raw DNA methylation information. When the raw information is available (for example the information contained in the IDat file such as the set of negative control probes used for the definition of the background intensity), the subset-quantile within array normalization (`SWAN`, [3]) is probably more optimized approach (Figure 2). For each sample, a scaling factor was defined by the ratio of the standard deviation observed in each group and it used to modify the scale of the chemistry II probe signals. The formula is given below :

$$x_{1,II} = \frac{x_{0,II}.\sigma_{0,I}}{\sigma_{0,II}}$$

where $x_{0,II}$ and $x_{1,II}$ correspond to the original and corrected intensity values

from chemistry II probes. The terms $\sigma_{0,I}$ and $\sigma_{0,II}$ are the standard deviation estimations for the chemistry I and II respectively. A comparison of three approaches (raw data, SWAN and scaling factor approach) was performed on datasets containing 6 samples from R packages minfiData.

```
### scaliong factor correction
### 2014-05-13 modified by pbady
require("preprocessCore")
M1 <- M0
U1 <- U0
for(k in 1:6){
u0 <- U0[,k]
m0 <- M0[,k]
disperu <- tapply(u0,fac,sd)
disperm <- tapply(m0,fac,sd)
u1 <- u0
u1[fac=="I"] <- u1[fac=="I"]*disperu[2]/disperu[1]
m1 <- m0
m1[fac=="I"] <- m1[fac=="I"]*disperm[2]/disperm[1]
M1[,k] <-m1
U1[,k] <-u1
}
w <- normalize.quantiles(cbind(U1,M1))
U1 <- w[,1:6]
M1 <- w[,7:12]
rownames(U1) <- rownames(M1) <- rownames(U0)
colnames(U1) <- colnames(M1) <- colnames(U0)
```

The representations of the comparison between the three methods of normalization (raw,swan and scaling factor) were given below :

```
par(mfrow=c(3,4))
SM1 <- getMeth(datIlmnSwan)
SU1 <- getUnmeth(datIlmnSwan)
for(k in 1:6){
  vecrnd1 <- sample(1:nrow(U1),1000)
  u1 <- U1[vecrnd1,k]
  m1 <- M1[vecrnd1,k]
  su1 <- SU1[vecrnd1,k]
  sm1 <- SM1[vecrnd1,k]
  u0 <- U0[vecrnd1,k]
  m0 <- M0[vecrnd1,k]
  fac1 <- fac[vecrnd1]
  plot(u1+m1,su1+sm1,bg=ifelse(fac1=="II","orange","green2"),
    pch=21,panel.first=c(grid()),ylab="total intensity (Swan)",
    xlab="total intesnity (scaling factor)",
    main=paste("sample",k,"(scaling factor)"))
  abline(0,1,col="red")
  abline(lm(I(su1+sm1)~I(u1+m1),subset=fac1=="I"),col="green2",lwd=2)
  abline(lm(I(su1+sm1)~I(u1+m1),subset=fac1=="II"),col="orange",lwd=2)
  plot(u0+m0,su1+sm1,bg=ifelse(fac1=="II","orange","green2"),
     pch=21,panel.first=c(grid()),ylab="total intensity (Swan)",
     xlab="total intesnity (raw)",main=paste("sample",k,"(raw)"))
  abline(0,1,col="red")
  abline(lm(I(su1+sm1)~I(u0+m0),subset=fac1=="I"),col="green2",lwd=2)
  abline(lm(I(su1+sm1)~I(u0+m0),subset=fac1=="II"),col="orange",lwd=2)
}
```

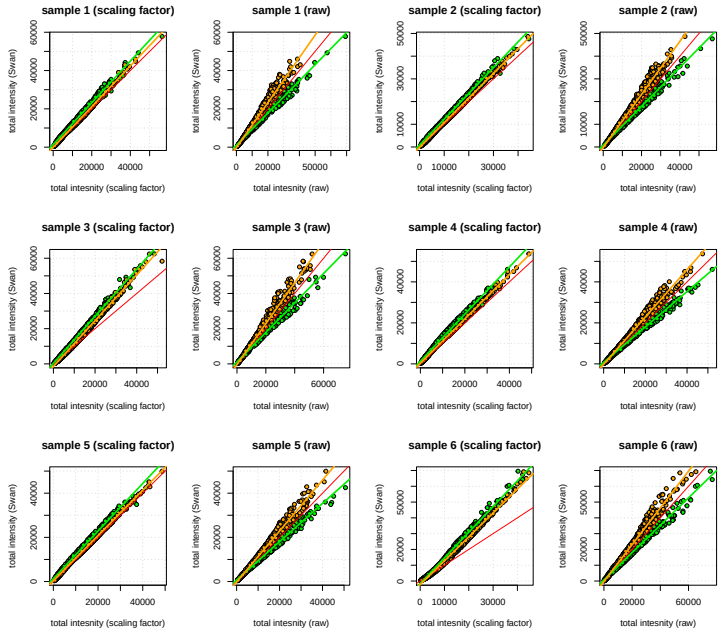

FIGURE 5 – Comparison of correction approaches for the 6 samples

.

```
            used   (Mb) gc trigger   (Mb)  max used   (Mb)
Ncells   8648853  461.9   12387423  661.6   9107316  486.4
Vcells 204811680 1562.6  322676124 2461.9 322674207 2461.9
```

Our correction provided results similar to them obtained by `SWAN` procedure proposed in The R package `minfi` ([3]).

```
require(methyltools)
# scaling and computation of the total intensity
nT1 <- meth2norm.MethySet(datIlmn,scaling=TRUE)
gc()
```

When the chemistry type is unknown, the function textttttmeth2norm automatically uses the function textttgetProbeInfo to extract this information. The comparison between the two estimates (manual and estimate form the function textttmeth2norm) is given below :

```
head(nT1$T)
```

	5723646052_R02C02	5723646052_R04C01	5723646052_R05C02	5723646053_R04C02
cg00050873	15524.9336	191.06031	12327.4285	5.922076
cg00212031	5103.6156	142.19811	1868.5831	33.911056
cg00213748	765.7985	96.44561	312.8934	254.927398
cg00214611	3729.7168	167.75873	1076.9813	183.336376
cg00455876	4061.5189	293.44201	2671.7484	206.877350
cg01707559	9626.0692	627.39144	5051.1521	1143.907583

	5723646053_R05C02	5723646053_R06C02
cg00050873	53.5557	15.29544
cg00212031	157.5056	365.20518
cg00213748	137.1440	463.33526
cg00214611	100.0474	625.79812
cg00455876	207.8675	449.05384
cg01707559	682.4646	1290.01611

```
head(U1+M1)
```

	5723646052_R02C02	5723646052_R04C01	5723646052_R05C02	5723646053_R04C02
cg00050873	15524.9336	191.06031	12327.4285	5.922076
cg00212031	5103.6156	142.19811	1868.5831	33.911056
cg00213748	765.7985	96.44561	312.8934	254.927398
cg00214611	3729.7168	167.75873	1076.9813	183.336376
cg00455876	4061.5189	293.44201	2671.7484	206.877350
cg01707559	9626.0692	627.39144	5051.1521	1143.907583

	5723646053_R05C02	5723646053_R06C02
cg00050873	53.5557	15.29544
cg00212031	157.5056	365.20518
cg00213748	137.1440	463.33526
cg00214611	100.0474	625.79812
cg00455876	207.8675	449.05384
cg01707559	682.4646	1290.01611

NOTA : in the futur version of methyltools, the type-bias correction and computation will be implemented in C program for reducing the time computation.

5 Quantile normalization, Total intensity and Log2-ratio calculation

After extractions of the signals, we used a quantile normalization (QN) strategy to exclure dye bias as proposed in [6] for Illumina Infinium Whole-genome SNP data. The QN is performed individually for each sample using affine normalization intensity (U=unmethylated, M=methylated) from GenomeStudio output or R packages `minfi` and `preprocesscore`. The combined intensity (total intensity), T, was calculated from the QN intensities. Because matched reference samples were not available, $log_2(RR)$ is defined by the difference of intensity between samples and a synthetic reference corresponding to the median profile from a reference data sets :

$$log_2(RR) = log_2(T_{observed} + 1) - log_2(T_{reference} + 1)$$

In our original study, the references data sets contained eight non-tumor brain samples from TCGA ([4]) and EORTC study (e.g. [1]). An additional smoothing procedure was applied to remove the wave bias for more accurate breakpoint

detection in profiles as proposed by ([8]). In this document, we used the median profile based on the 6 samples.

```
# manual calculation
# it's possible to directly use the object 'nT1$T'.
T1 <- U1 + M1
T1 <- log2(T1+1)
Tref <- apply(T1,2,median)
logRR <- T1-Tref
summary(logRR)
rm(list=c("U1","M1"))
gc()
```

6 Circular binary segmentation (CBS)

The CNA data were analysed by circular binary segmentation ([5]) performed on normalized log2(RR) values for each sample. The R packages `DNAcopy` and `CGHcall` were used to performed CBS and to established copy number information. The deletion and amplification events are summarized in using means or median of a given regions (e.g. CHR10, EGFR, CHR7, MGMT, codeletion of 1p and 19q regions).

```
library("FDb.InfiniumMethylation.hg19")
annot450k <- as.data.frame(get450k())
head(annot450k)
annot1 <- annot450k[!is.element(as.character(annot450k$seqnames),c("chrX","chrY")),]
annot1 <- annot1[substring(rownames(annot1),1,2)=="cg"||substring(rownames(annot1),1,2)=="rs",]
annot1$chrom <- as.numeric(gsub("chr","",as.character(annot1$seqnames)))
intersectx <- intersect(rownames(annot1),rownames(logRR))
annot1 <- annot1[intersectx,]
logRR <- logRR[intersectx,]
dim(annot1)
dim(logRR)
rm(annot450k)
gc()
# CGHcall and DNAcopy package
require(CGHcall)
# data preparation
tmp <- data.frame(ID=rownames(annot1),chromStart_hg19=as.numeric(annot1$start),
                  chromEnd_hg19=as.numeric(annot1$end))
tmp$chrom_hg19_num <- annot1$chrom
vecnames <- c("ID","chrom_hg19_num","chromStart_hg19","chromEnd_hg19")
rownames(tmp) <- tmp[,1]
tmp <- cbind(tmp[,vecnames],logRR[rownames(tmp),])
colnames(tmp)[1:4] <- c("ID","Chromosome","Start","End")
tmp$ID <- as.factor(as.character(tmp$ID))
head(tmp)
### preparation object cghRaw
cghtmp <- make_cghRaw(tmp)
pretmp <- CGHcall:::preprocess(cghtmp,nchrom=22)
rm(cghtmp)
gc()
normtmp <- CGHcall:::normalize(pretmp,"median",smooth=TRUE)
rm(pretmp)
gc()
### segmentation to obtain object cghSeg
seg <-  segmentData(normtmp, method = "DNAcopy", nperm=10,undo.splits="sdundo")
```

7 Mixture model and estimation of the CNA state for each CpG

The CNA state is estimated in using mixture model of 6 gaussian distributions (see for example the R package `CGHcall`,[8]). The procedure is described in de-

tails in the documentation of the function. To reduce noise, it's possible to use a post-segmentation normalization before the estimation of the DNA methylation state of the CpG.

```
require(methyltools)
# CGHcallexpand
segnorm <- postsegnormalize(seg)
rm(seg)
gc()
### object cghCall containing amplification status, strat, end, ...
listcalls <- CGHcall(segnorm,nclass=5,robustsig="yes",prior="all",ncpus=1,
                     cellularity=1)
expcalls <- ExpandCGHcall(listcalls,segnorm, divide=3, memeff=FALSE)
```

The CNA profile of the datasets are given in the two following graphical representations :

```
summaryPlot(expcalls[sort(sample(1:dim(expcalls)[1],3000)),])
```

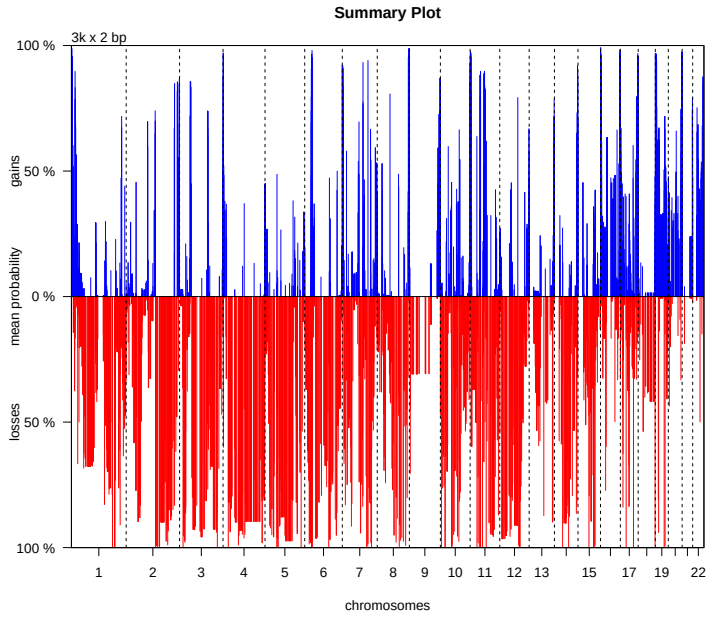

FIGURE 6 – Representation of Copy number alteration based on Inifinium HM-450k platform (we randomly selected 3000 markers for the graphical representation).

```
frequencyPlotCalls(expcalls[sort(sample(1:dim(expcalls)[1],3000)),])
```

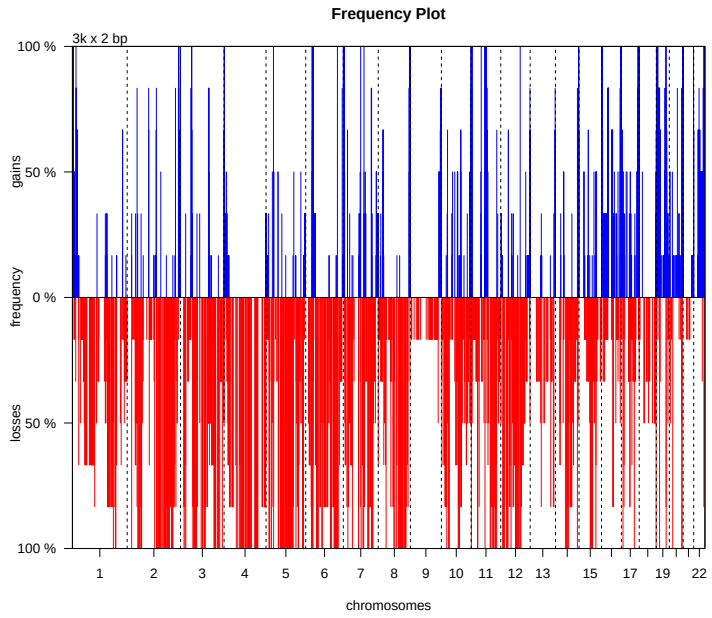

FIGURE 7 – Representation of Copy number alteration based on Inifinium HM-450k platform (we randomly selected 3000 markers for the graphical representation).

8 CNA estimation at the sample scale

8.1 Method "max"

A very simple algorithm is used to define the state of gene or genomic region : 1) Count the number of event by possible states, 2) select the more frequent state. In the package textttmethyltools, a set of functions are specifically dedicated to estimate the CNA with method 'max' for a given region, chromosome or gene for each individual samples (samples scale estimation).

```
maxCNAcalls(X,chrom, start,end,winsize=0,mode=NULL,...)
maxCNAcalls.chr(X,chrom,...)
maxCNAcalls.gene(X,gene,...)
```

The functions returns a list of class textttCNAcalls :

cna : a character vector containing the CNA status for each sample

n : a numeric value corresponding to the number of samples

info : a character value describing the selected genomic regions

winsize : a numerical value used to increase the windows of the selected genomic region (in pb)

location : a list containning information about the location of the selected region (chr, start and end positions)

summary : a numerical vector containing a summarize of the CNA information

maxentropy : maximal entropy (e.g. log(5) for 5 states)

efficiency : index ([0,1], = entropy/maxentropy) used to evaluate the quality of the CNA estimate.

An example of the use of the function `maxCNAcalls` is given below :

```
require(CGHcall)
require(methyltools)
# example: maxCNAcalls for a chromosomes and a gene
egfrmax <- maxCNAcalls.gene(expcalls,gene="EGFR")
mgmtmax <- maxCNAcalls.gene(expcalls,gene="MGMT")
chr7max <- maxCNAcalls.chr(expcalls,chrom="chr7")
# entropy.index, summary and other features
egfrmax
```

```
Estimation of CNA based on mixture model (see CGHcall)
$class:  CNAcalls list
$n:  6
$marker:  54
$info:  EGFR
$winsize 0
$location:
       chr      start        end
    "chr7" "55086725" "55275031"
$summary:
n
6
$call: maxCNAcalls.gene(X = expcalls, gene = "EGFR")
```

```
mgmtmax
```

```
Estimation of CNA based on mixture model (see CGHcall)
$class:  CNAcalls list
$n:  6
$marker:  153
$info:  MGMT
$winsize 0
$location:
        chr       start         end
    "chr10" "131265454" "131565783"
$summary:
g n
2 4
$call: maxCNAcalls.gene(X = expcalls, gene = "MGMT")
```

```
chr7max
```

```
Estimation of CNA based on mixture model (see CGHcall)
$class:  CNAcalls list
$n:  6
$marker:  30017
$info:  chr7
$winsize 0
$location:
   chr
"chr7"
$summary:
g n
2 4
$call: maxCNAcalls.chr(X = expcalls, chrom = "chr7")
```

The standardized entropy index provides information about the quality of the CNA state. The index is equal to one when the entropy is maximal (all states are equally represented for the selected genomic region) and it's equal to zero when of marker of the selected genomic region have the same state.

8.2 Method "mix"

In texttt'methyltools, we propose another way to define CNA state of a genomic region based on cut-off obtained after combination of the CNA state and log2-segmented values defined at the marker scale. Indeed, for each samples, we

defined limits to identify loss,gain or amplification events in using the segmented values and the classification from mixture model proposed in the R package `CGHcall`. In a first step, we compute the mean (or median, etc...) of markers located on the selected region. in a second step, the CNA state is given in using the cut-offs for loss, gain and amplification computed as follow :

$$\begin{cases} cut_{dd} = \frac{(max(x_{dd})+min(x_d))}{2} \\ cut_d = \frac{(max(x_d)+min(x_n))}{2} \\ cut_g = \frac{(max(x_n)+min(x_g))}{2} \\ cut_a = \frac{(max(x_g)+min(x_a))}{2} \end{cases}$$

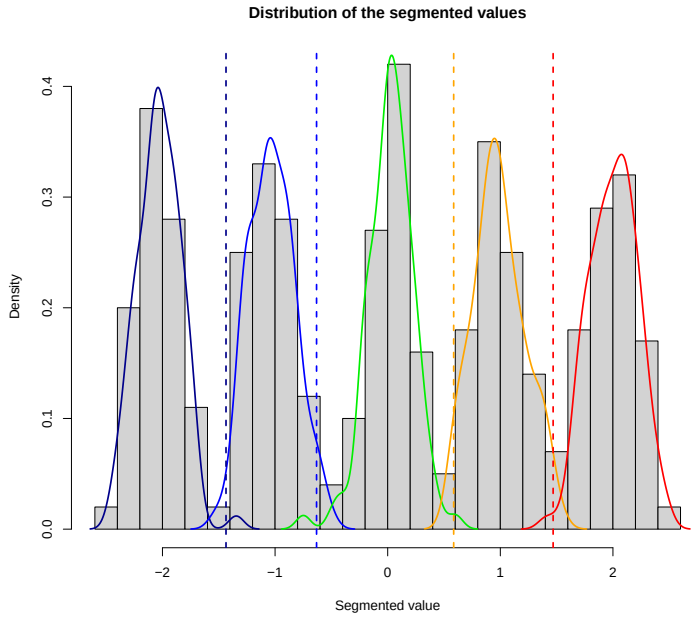

FIGURE 8 – Representation of the CNA limits for simulated datasets.

The function `mixCNAcalls` from R package `methyltools` provides directly the estimation of the CNA for each samples :

```
mixCNAcalls(X,chrom, start,end,winsize=0, statistic= "mean",useout=FALSE,...)
mixCNAcalls.chr(X,chrom,...)
mixCNAcalls.gene(X,gene,...)
```

The functions returns a list of class texttCNAcalls :

cna : a character vector containing the CNA status for each sample

n : a numeric value corresponding to the number of samples

info : a character value describing the selected genomic regions

winsize : a numerical value used to increase the windows of the selected genomic region (in pb)

location : a list containing information about the location of the selected region (chr, start and end positions)

summary : a numerical vector containing a summarize of the CNA information

An example of the use of the function `mixCNAcalls` is given below :

```
 # example: mixCNAcalls for a chromosomes and a gene
 egfrmix <- mixCNAcalls.gene(expcalls,gene="EGFR")
 mgmtmix <- mixCNAcalls.gene(expcalls,gene="MGMT")
 chr7mix <- mixCNAcalls.chr(expcalls,chr="chr7")
 # summary
 egfrmix
Estimation of CNA based on mixture model (see CGHcall)
$class:  mixCNAcalls list
$n:  6
$marker:  54
$info:  EGFR
$winsize 0
$location:
     chr    start      end
       7 55086725 55275031
$summary:
n
6
$call: mixCNAcalls.gene(X = expcalls, gene = "EGFR")
 mgmtmix
Estimation of CNA based on mixture model (see CGHcall)
$class:  mixCNAcalls list
$n:  6
$marker:  153
$info:  MGMT
$winsize 0
$location:
      chr     start       end
       10 131265454 131565783
$summary:
n
6
$call: mixCNAcalls.gene(X = expcalls, gene = "MGMT")
 chr7mix
Estimation of CNA based on mixture model (see CGHcall)
$class:  mixCNAcalls list
$n:  6
$marker:  30017
$info:  chr7
$winsize 0
$location:
   chr
"chr7"
$summary:
n
6
$call: mixCNAcalls.chr(X = expcalls, chrom = "chr7")
```

The comparison between the outputs of mixCNAcalls and `maxCNAcalls` for EGFR, MGMT and chromosome 7 was given below :

```
 # comparison with the results from maxCNAcalls
 table(MAX=egfrmax$cna,MIX=egfrmix$cna)
   MIX
MAX n
  n 6
 table(MAX=mgmtmax$cna,MIX=egfrmix$cna)
   MIX
MAX n
  g 2
  n 4
 table(MAX=chr7max$cna,MIX=egfrmix$cna)
   MIX
MAX n
  g 2
  n 4
```

NOTA : Theoretically, it's possible to compute posterior probability for each sample and CNA state with the parameters (e.g. mean, sd) associated with mixture model. However, in practice this way was not very easy, because the implementation of the function `Calls` are relatively complex (e.g. robust estimate, ...), it's relatively difficult to generalize the estimation of these probabilities. For this additional reason, the two precious solutions appears to be a reasonable compromise.

9 Non-random location of CpG (coverage)

The CpGs of HM-450K platform were not randomly loacted by construction. Indeed, the markers (probes) are loacted in genomic region such as CpG island, miRNA promoter regions, DNase hypersensitive sites, FANTOM 4 promoters, Methylation hotspots in cancer genes, cancer-related targets (see Illumina documentation). The coverage by chromsomes is given below :

```
 chrom <- chromosomes(expcalls)
 table(chrom)
chrom
    1     2     3     4     5     6     7     8     9    10    11    12    13    14
46857 34810 25159 20464 24327 36611 30017 20950  9861 24388 28794 24539 12285 15078
   15    16    17    18    19    20    21    22
15259 21969 27879  5922 25521 10379  4243  8552
 uni.chrom <- names(table(chrom))
 chrom.lengths <- CGHbase:::.getChromosomeLengths("GRCh37")[as.character(uni.chrom)]
 coveragex <- table(chrom)/(chrom.lengths/1000)
```

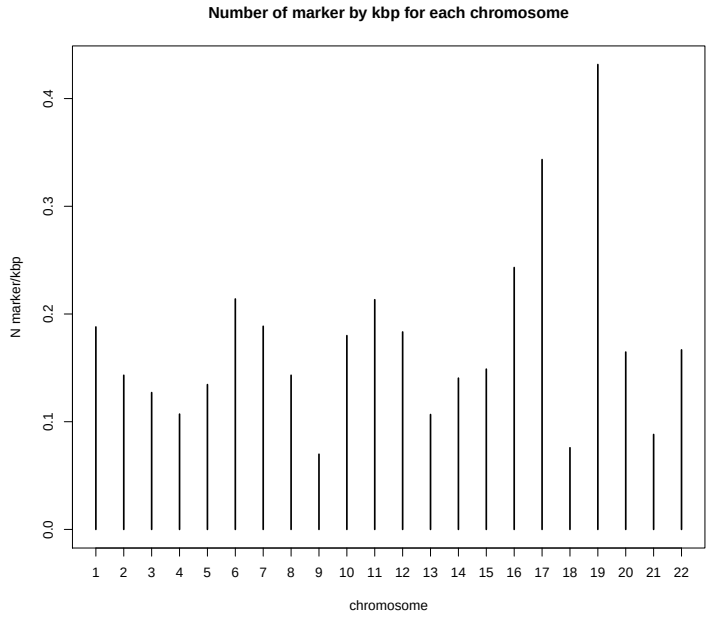

FIGURE 9 – Number of marker by kbp for each chromosome.

10 Effect of CpG concentration on CNA estimate

In this section, we briefly analyze the relationship between the GpC number and the CNA estimate (segmeted log2-RR values). The variable nCpG represents either the number of methylation loci (approx. number of CpGs) on the array or the locus names.

```
library(matrixStats)
seg1 <- CGHbase:::segmented(expcalls)
dim(seg1)
```

```
[1] 473864      6
```

```
m1 <- rowMedians(seg1)
summary(m1)
```

```
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-6.33900 -0.26850 -0.00700 -0.04709  0.22650  1.66800
```

```
# annotation for 450k
typeI <- data.frame(getProbeInfo(datIlmn, type = "I")[, c("Name", "nCpG")])
typeII <- data.frame(getProbeInfo(datIlmn, type = "II")[, c("Name", "nCpG")])
HMchemistry <- rbind(typeI, typeII)
HMchemistry$type <- c(rep("I",nrow(typeI)),rep("II",nrow(typeII)))
rownames(HMchemistry) <-  HMchemistry$Name
HMchemistry <- HMchemistry[rownames(datIlmn),]
table(rownames(HMchemistry)==rownames(datIlmn),useNA="always")
```

```
  TRUE   <NA>
485512      0
```

```
head(HMchemistry)
```

```
                Name nCpG type
cg00050873 cg00050873    2    I
cg00212031 cg00212031    4    I
cg00213748 cg00213748    3    I
cg00214611 cg00214611    5    I
cg00455876 cg00455876    2    I
cg01707559 cg01707559    6    I
```

```
dim(HMchemistry)
```

```
[1] 485512      3
```

```
HMchemistry <- HMchemistry[rownames(seg1),]
table(HMchemistry[,"nCpG"])
```

```
     0      1      2      3      4      5      6      7      8      9     10
147346 120035  85606  58354  35107  17032   7451   2410    464     54      5
```

```
par(mfrow=c(2,1))
hist(HMchemistry[,"nCpG"],proba=TRUE,col="lightgreen",
     main="Distribution of nCpG",xlab="nCpG")
boxplot(m1~HMchemistry[,"nCpG"],main="CNA vs nCpG",xlab="nCpG",
        ylab="segmented value")
abline(h=median(m1),col="red")
# abline(h=mean(m1),col="green2")
```

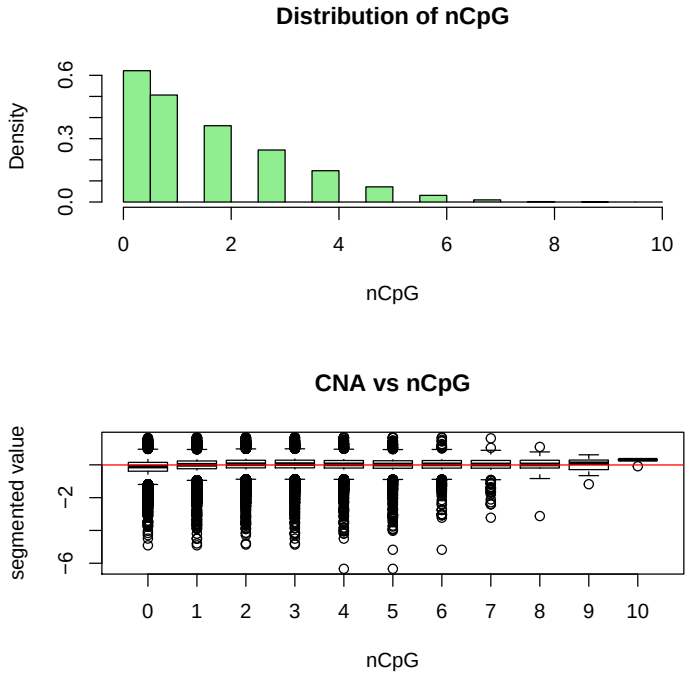

FIGURE 10 – Relationship between Number of CpG and CNA estimate (medians of segmented log2-RR values).

11 Acknowledgments

The results published here are in part based upon data generated by The Cancer TCGA Genome Atlas pilot project established by the NCI and NHGRI. Information about TCGA and the investigators and institutions who constitute the TCGA research network can be found at (http ://cancergenome.nih.gov). The dbGaP accession number to the specific version of the TCGA data set is phs000178.v8.p7.

Références

[1] Pierre Bady, Davide Sciuscio, Annie-Claire Diserens, Jocelyne Bloch, Martin J. van den Bent, Christine Marosi, Pierre-Yves Dietrich, Michael Weller, Luigi Mariani, Frank L. Heppner, David R. McDonald, Denis Lacombe,

Roger Stupp, Mauro Delorenzi, and Monika E. Hegi. Mgmt methylation analysis of glioblastoma on the infinium methylation beadchip identifies two distinct cpg regions associated with gene silencing and outcome, yielding a prediction model for comparisons across datasets, tumor grades, and cimp-status. *Acta Neuropathologica*, 124(4) :547–560, 2012. Times Cited : 6.

[2] Amandine Etcheverry, Marc Aubry, Marie de Tayrac, Elodie Vauleon, Rachel Boniface, Frederique Guenot, Stephan Saikali, Abderrahmane Hamlat, Laurent Riffaud, Philippe Menei, Veronique Quillien, and Jean Mosser. Dna methylation in glioblastoma : impact on gene expression and clinical outcome. *BMC Genomics*, 11(1) :701, 2010.

[3] Jovana Maksimovic, Lavinia Gordon, and Alicia Oshlack. Swan : Subset-quantile within array normalization for illumina infinium humanmethylation450 beadchips. *Genome Biology*, 13(6) :R44, 2012.

[4] The Cancer Genome Atlas Research Network. Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature*, 455(7216) :1061–1068, 2008. 10.1038/nature07385.

[5] Adam B. Olshen, E. S. Venkatraman, Robert Lucito, and Michael Wigler. Circular binary segmentation for the analysis of arrayâbased dna copy number data. *Biostatistics*, 5(4) :557–572, 2004.

[6] Johan Staaf, Johan Vallon-Christersson, David Lindgren, Gunnar Juliusson, Richard Rosenquist, Mattias Hoglund, Ake Borg, and Markus Ringner. Normalization of illumina infinium whole-genome snp data improves copy number estimates and allelic intensity ratios. *BMC Bioinformatics*, 9(1) :409, 2008.

[7] Sevin Turcan, Daniel Rohle, Anuj Goenka, Logan A. Walsh, Fang Fang, Emrullah Yilmaz, Carl Campos, Armida W. M. Fabius, Chao Lu, Patrick S. Ward, Craig B. Thompson, Andrew Kaufman, Olga Guryanova, Ross Levine, Adriana Heguy, Agnes Viale, Luc G. T. Morris, Jason T. Huse, Ingo K. Mellinghoff, and Timothy A. Chan. Idh1 mutation is sufficient to establish the glioma hypermethylator phenotype. *Nature*, 483(7390) :479–483, 2012. 10.1038/nature10866.

[8] Mark A. van de Wiel, Rebecca Brosens, Paul H. C. Eilers, Candy Kumps, Gerrit A. Meijer, Bjorn Menten, Erik Sistermans, Frank Speleman, Marieke E. Timmerman, and Bauke Ylstra. Smoothing waves in array cgh tumor profiles. *Bioinformatics*, 25(9) :1099–1104, 2009.

[9] Martin J. van den Bent, Lonneke A. Gravendeel, Thierry Gorlia, Johan M. Kros, Lariesa Lapre, Pieter Wesseling, Johannes L. Teepen, Ahmed Idbaih, Marc Sanson, Peter A.E. Sillevis Smitt, and Pim J. French. A hypermethylated phenotype is a better predictor of survival than mgmt methylation in anaplastic oligodendroglial brain tumors : A report from eortc study 26951. *Clinical Cancer Research*, 17(22) :7148–7155, 2011.

12 Session

```
print(sessionInfo(),locale=FALSE)
```

```
R version 3.1.2 (2014-10-31)
Platform: x86_64-w64-mingw32/x64 (64-bit)

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

other attached packages:
 [1] matrixStats_0.10.3
 [2] mixOmics_5.0-3
 [3] methyltools_0.4
 [4] ade4_1.6-2
 [5] RPMM_1.20
 [6] cluster_1.15.3
 [7] mixtools_1.0.2
 [8] segmented_0.5-0.0
 [9] MASS_7.3-35
[10] boot_1.3-13
[11] lumi_2.18.0
[12] TxDb.Hsapiens.UCSC.hg19.knownGene_3.0.0
[13] GenomicFeatures_1.18.2
[14] org.Hs.eg.db_3.0.0
[15] RSQLite_1.0.0
[16] DBI_0.3.1
[17] AnnotationDbi_1.28.1
[18] preprocessCore_1.28.0
[19] CGHcall_2.28.0
[20] snowfall_1.84-6
[21] snow_0.3-13
[22] CGHbase_1.26.0
[23] marray_1.44.0
[24] limma_3.22.1
[25] DNAcopy_1.40.0
[26] impute_1.40.0
[27] minfiData_0.7.1
[28] IlluminaHumanMethylation450kanno.ilmn12.hg19_0.2.1
[29] IlluminaHumanMethylation450kmanifest_0.4.0
[30] minfi_1.12.0
[31] bumphunter_1.6.0
[32] locfit_1.5-9.1
[33] iterators_1.0.7
[34] foreach_1.4.2
[35] Biostrings_2.34.1
[36] XVector_0.6.0
[37] GenomicRanges_1.18.3
[38] GenomeInfoDb_1.2.3
[39] IRanges_2.0.0
[40] S4Vectors_0.4.0
[41] lattice_0.20-29
[42] Biobase_2.26.0
[43] BiocGenerics_0.12.1
[44] markdown_0.7.4
[45] knitr_1.8

loaded via a namespace (and not attached):
 [1] affy_1.44.0             affyio_1.34.0           annotate_1.44.0
 [4] base64_1.1              base64enc_0.1-2         BatchJobs_1.5
 [7] BBmisc_1.9              beanplot_1.2            BiocInstaller_1.16.1
[10] BiocParallel_1.0.3      biomaRt_2.22.0          bitops_1.0-6
[13] brew_1.0-6              checkmate_1.5.0         codetools_0.2-9
[16] colorspace_1.2-4        digest_0.6.4            doRNG_1.6
[19] evaluate_0.5.5          fail_1.2                formatR_1.0
[22] genefilter_1.48.1       GenomicAlignments_1.2.1 grid_3.1.2
[25] igraph_0.7.1            illuminaio_0.8.0        KernSmooth_2.23-13
[28] lme4_1.1-7              Matrix_1.1-4            mclust_4.4
[31] methylumi_2.12.0        mgcv_1.8-3              minqa_1.2.4
[34] multtest_2.22.0         nleqslv_2.5             nlme_3.1-118
[37] nloptr_1.0.4            nor1mix_1.2-0           pheatmap_0.7.7
[40] pkgmaker_0.22           plyr_1.8.1              quadprog_1.5-5
[43] R.methodsS3_1.6.1       RColorBrewer_1.0-5      Rcpp_0.11.3
[46] RCurl_1.95-4.3          registry_0.2            reshape_0.8.5
[49] RGCCA_2.0               rgl_0.95.1158           rngtools_1.2.4
[52] Rsamtools_1.18.2        rtracklayer_1.26.2      sendmailR_1.2-1
[55] siggenes_1.40.0         splines_3.1.2           stringr_0.6.2
[58] survival_2.37-7         tools_3.1.2             XML_3.98-1.1
[61] xtable_1.7-4            zlibbioc_1.12.0
```