BIOS 7659 Homework 8

Tim Vigers

22 November 2020

# 1. Cell-Type Composition

Read in the data:

baseDir1 = "blood/plate1"  
targets1 = read.metharray.sheet(baseDir1)

## [1] "blood/plate1/selected\_plate1.csv"

baseDir2 = "blood/plate2"  
targets2 = read.metharray.sheet(baseDir2)

## [1] "blood/plate2/selected\_plate2.csv"

targets = rbind(targets1, targets2)  
rgSet = read.metharray.exp(targets=targets,extended=T)  
sampleNames(rgSet) = rgSet[[1]]  
getManifest(rgSet)

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

clindat = read.table("blood/demographic.txt", sep="\t", header=T)  
pData(rgSet)$Sample\_Group = clindat$Exposure  
pData(rgSet)$child\_sex = clindat$child\_sex

## a) Find differentially methylated positions based on exposure status

# SWAN normalization  
msetSWAN = preprocessSWAN(rgSet)  
dmp = dmpFinder(msetSWAN,pheno = msetSWAN$Sample\_Group,type = "continuous")

Here the unexposed group is the reference, so a negative value of beta in the DMP table indicates that the probe was hypomethylated in the exposed group (and a positive value indicates hypermethylation). There are no probes significant at the level, but there are 4 significant at the level:

dmp %>% rownames\_to\_column(var="Probe") %>% filter(pval < 1e-5) %>% mutate(Direction = ifelse(sign(beta)==-1,"Hypomethylated","Hypermethylated")) %>% select(Probe,beta,Direction,pval) %>% rename("Beta"=beta,"P Value"=pval) %>% flextable(.) %>% footnote(.,i=1,j=3,value = as\_paragraph("With respect to exposure"),ref\_symbols = "\*",part = "header") %>% set\_table\_properties(.,layout = "autofit",width = 0.75)

| Probe | Beta | Direction\* | P Value |
| --- | --- | --- | --- |
| cg25761397 | -0.3997037 | Hypomethylated | 1.332702e-06 |
| cg00244352 | -0.3431909 | Hypomethylated | 4.254079e-06 |
| cg15614119 | -0.2460500 | Hypomethylated | 5.528681e-06 |
| cg21088165 | -0.4816779 | Hypomethylated | 7.535189e-06 |
| \*With respect to exposure | | | |

## b) Cell-type composition

Cell-type composition is important in methylation studies because methylation is a key factor in converting cells from a pluripotent progenitor to specific cell-type. As a result, differentially methylated regions (DMRs) can distinguish cell-types with high sensitivity and specificity [1]. In other words, if you are comparing two samples with different cell-type composition without some sort of adjustment, many DMRs that distinguish between them will likely be the result of cell-type composition rather than the exposure of interest. There are many ways to adjust for cell-type composition when cell-type isn’t directly measured, but the approach in Houseman et al. [1] is implemented in estimateCellCounts() in the minfi package, and that paper is frequently cited. The approach in Jaffe & Irizarry [2] is simply an adaptation of the Houseman algorithm for the 450K array.

The Houseman algorithm takes an vector of methylation values for a homogenous cell population, and proposes the linear model where is a covariate vector indicating cell-type and is an matrix. Also, indexes the specimen and is an error vector. Next, they propose a similar model for the data without cell-type measurements: with the linking regression model where indexes samples of interest and is a matrix of errors. The math gets fairly complicated after this, so I won’t go into any more detail, but the points is that the cell-type mixture coefficients can be “recovered” from by estimating and via linear models using ordinary least squares, mixed effects models, limma, or surrogate variable analysis. The authors also recommend bootstrap procedures for estimating standard errors.

## c) Estimate cell counts

rgSetCell = read.metharray.exp(targets=targets,extended=F)

## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls  
  
## Warning in readChar(con, nchars = n): truncating string with embedded nuls

cells = estimateCellCounts(rgSetCell,meanPlot = T,compositeCellType = "Blood",cellTypes = c("CD8T","CD4T","NK","Bcell","Mono","Gran"))

## [estimateCellCounts] Combining user data with reference (flow sorted) data.

## Warning in DataFrame(sampleNames = c(colnames(rgSet),  
## colnames(referenceRGset)), : 'stringsAsFactors' is ignored

## [estimateCellCounts] Processing user and reference data together.

## [preprocessQuantile] Mapping to genome.

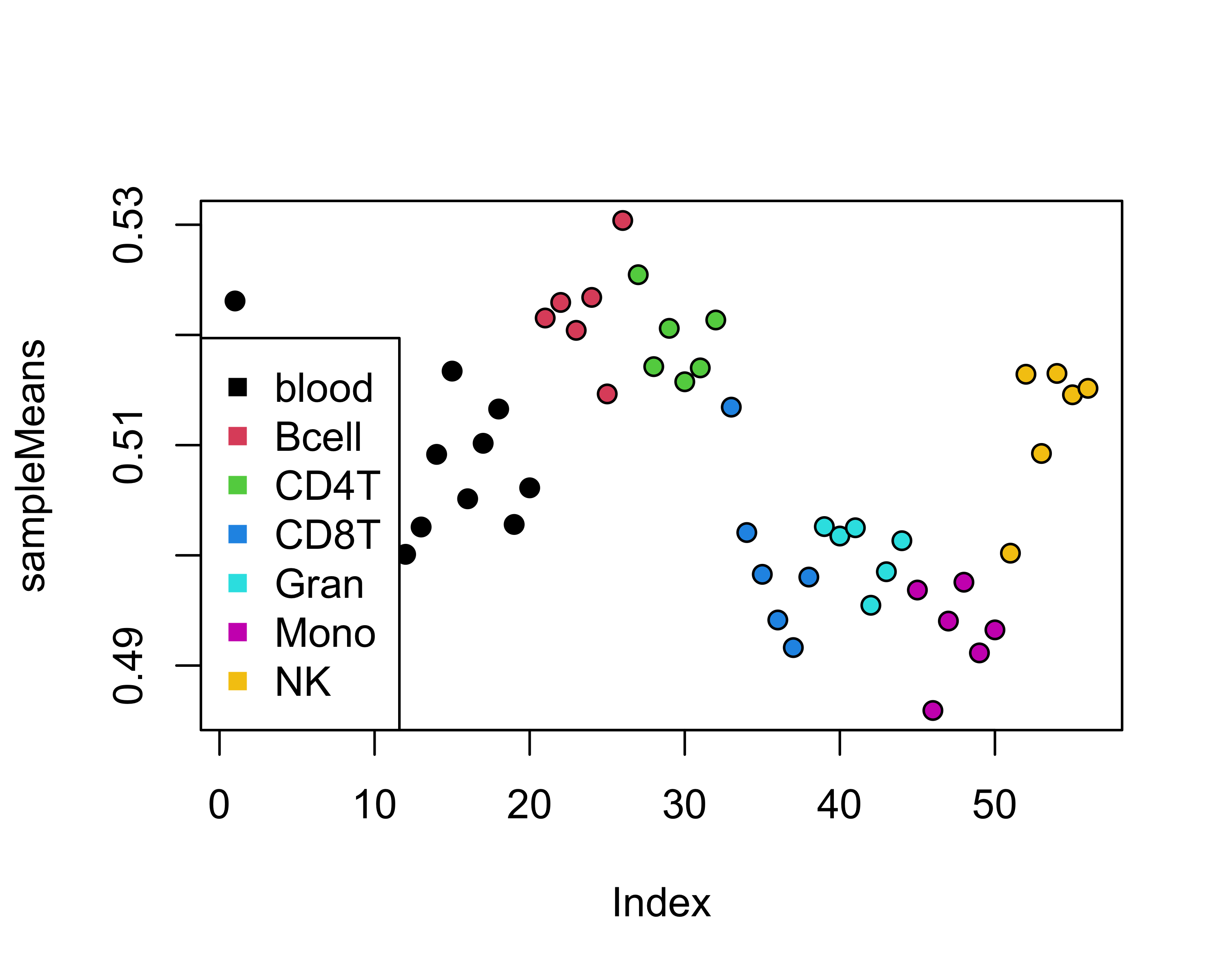
## Loading required package: IlluminaHumanMethylation450kanno.ilmn12.hg19

## [preprocessQuantile] Fixing outliers.

## [preprocessQuantile] Quantile normalizing.

## [estimateCellCounts] Picking probes for composition estimation.

## [estimateCellCounts] Estimating composition.



The plot above shows the average methylation across cell-type differentiating probes, for our data and the reference data set (“FlowSorted.Blood.450k”). The means of the two datasets should be in a similar range. If not, this suggests large batch effects in the data and the composition estimates should not be trusted (the function automatically performs quantile normalization between the two datasets to reduce potential batch effects). This plot looks reasonable, so we can probably trust these cell-type estimates and repeat part a) with adjustment for cell-type composition:

# 2. ChIP-Seq

Read in the data:

data(yeastCDS)   
allData = dataReading("./hw8files/tup1\_IP.txt", "./hw8files/mock\_IP.txt",  
 yeastSpecies = yeastCDS$Saccharomyces.cerevisiae)

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

# References

1. Houseman EA, Accomando WP, Koestler DC, et al. DNA methylation arrays as surrogate measures of cell mixture distribution. BMC Bioinformatics. 2012;13(1):86. <doi:10.1186/1471-2105-13-86>
2. Jaffe AE, Irizarry RA. Accounting for cellular heterogeneity is critical in epigenome-wide association studies. Genome Biol. 2014;15(2):R31. <doi:10.1186/gb-2014-15-2-r31>