TOPMed CRA DNA methylation data QC and sample cleaning

reprk: PKachroo
CDNM, BWH

May 9, 2021

Contents

1 Setup

```
# restart R session
#.rs.restartR()
rm(list=ls())

options(mc.cores=5)
system("hostname")
print(Sys.Date())
```

```
## [1] "2021-05-08"
```

```
print(Sys.time())
```

```
## [1] "2021-05-08 15:16:39 EDT"
```

```
# To generate document:
# Change working directory to code directory
# Run this code on toques using:
# module load R/4.0.3
#R -e 'library(knitr);knit("TOPMed_CRA_DNAm_processing.Rnw")'
# pdflatex TOPMed_CRA_DNAm_processing.tex

# merging with WGS, uses hg38?
## load libraries
libs <- c("IlluminaHumanMethylationEPICanno.ilm10b4.hg19",
          "IlluminaHumanMethylationEPICmanifest", "minfi")

for (l in libs) {
  if (require(l, character.only = T)) {
    print(paste0(l, " loaded successfully"))
  } else {
    install.packages(l)
    require(l, character.only = T)
    print(paste0(l, " installed and loaded successfully"))
  }
}
```

```
## [1] "IlluminaHumanMethylationEPICanno.ilm10b4.hg19 loaded successfully"
## [1] "IlluminaHumanMethylationEPICmanifest loaded successfully"
## [1] "minfi loaded successfully"
```

```
sig_digits <- 2
sum_sd <- function(data, varname) {
    eval(parse(text = str_c("data[, round(summary(", varname, "), digits=2)] %>% print()")))
    eval(parse(text = str_c("print(str_c('SD: ', data[, sd(", varname, ", na.rm = T) %>%
                                round(sig_digits)]))")))
}
```

1.1 Packages, Data locations and loading

```
qc.dir = "/proj/regeps/regep00/studies/CRA"
cra.dir = file.path(qc.dir,"data/epigenetic/methylation/TopMed/data/freezes/20200117")
RGSet.cra = readRDS(file=file.path(cra.dir, "LEVEL2/RGSet")) # 1237 samples
dim(RGSet.cra) # 1008711
```

```
## [1] 1008711    1237
```

```
RGSet.cra
```

```
## class: RGChannelSet
## dim: 1008711 1237
## metadata(0):
## assays(2): Green Red
## rownames(1008711): 1600101 1600111 ... 99810978 99810992
## rowData names(0):
## colnames(1237): TOE909374-BIS-v01_R08C01 TOE239310-BIS-v01_R01C01 ...
##   TOE786315-BIS-v01_R07C01 TOE294448-BIS-v01_R06C01
## colData names(7): Basename S_SAMPLEID ... S_STUDYID filenames
## Annotation
##   array: IlluminaHumanMethylationEPIC
##   annotation: ilm10b4.hg19
```

```
manifest = getManifest(RGSet.cra)
manifest
```

```
## IlluminaMethylationManifest object
## Annotation
##   array: IlluminaHumanMethylationEPIC
## Number of type I probes: 142262
## Number of type II probes: 724574
## Number of control probes: 635
## Number of SNP type I probes: 21
## Number of SNP type II probes: 38
```

```
data(IlluminaHumanMethylationEPICanno.ilm10b4.hg19)
data("Manifest")
table(Manifest$Type)
```

```
##
##      I     II
## 142137 723722
```

```
length(grep("^cg.", rownames(Manifest), value=TRUE)) # 862927 CG probes
```

```
## [1] 862927
```

```
length(grep("^ch.", rownames(Manifest), value=TRUE)) # 2932 CH probes
```

```
## [1] 2932
```

```
length(grep("^rs.", rownames(Manifest), value=TRUE)) # 0
```

```
## [1] 0
```

```
# Downloaded Illumina manifest file
festV1 <- read.csv("/proj/rerefs/reref00/Illumina/MethylationEPIC-v1-0-B4/lib/MethylationEPIC_v-1-0_B
                   skip=7,as.is=TRUE, sep=",", stringsAsFactors=FALSE)

# loading rest of the libraries
libs <- c("limma", "wateRmelon", "minfi", "gplots", "ggplot2", "knitr", "R.utils", "impute",
          "stats", "tidyverse", "data.table", "here", "e1071", "GGally", "ggrepel", "ENmix",
          "meffil", "data.table", "robustbase", "stringi", "geneplotter", "RColorBrewer",
          "colorRamps", "lumi", "ggrepel")

for (l in libs) {
  if (require(l, character.only = T)) {
    print(paste0(l, " loaded successfully"))
  } else {
    install.packages(l)
    require(l, character.only = T)
    print(paste0(l, " installed and loaded successfully"))
  }
}
```

```
## [1] "limma loaded successfully"
## [1] "wateRmelon loaded successfully"
## [1] "minfi loaded successfully"
## [1] "gplots loaded successfully"
## [1] "ggplot2 loaded successfully"
## [1] "knitr loaded successfully"
## [1] "R.utils loaded successfully"
## [1] "impute loaded successfully"
## [1] "stats loaded successfully"
## [1] "tidyverse loaded successfully"
## [1] "data.table loaded successfully"
## [1] "here loaded successfully"
## [1] "e1071 loaded successfully"
## [1] "GGally loaded successfully"
## [1] "ggrepel loaded successfully"
## [1] "ENmix loaded successfully"
## [1] "meffil loaded successfully"
## [1] "data.table loaded successfully"
## [1] "robustbase loaded successfully"
## [1] "stringi loaded successfully"
## [1] "geneplotter loaded successfully"
## [1] "RColorBrewer loaded successfully"
## [1] "colorRamps loaded successfully"
## [1] "lumi loaded successfully"
## [1] "ggrepel loaded successfully"
```

```
plots.dir = file.path(qc.dir,"analysis/reprk/methylation/plots")
results.dir = file.path(plots.dir,"../results")
meff.dir = file.path(qc.dir,"analysis/reprk/meffil_850K")

# modified RCP code
source("/udd/reprk/projects/TOPMed/scripts/RCP_mod.R")

pca.betas <- function (beta, npc = 50)
{
    if (!is.matrix(beta)) {
        stop("beta is not a data matirx")
    }
    cat("Analysis is running, please wait...!", "\n")
    npc <- min(ncol(beta), npc)
    svd <- prcomp(t(beta), center = TRUE, scale = TRUE, retx = TRUE)
    eigenvalue <- svd[["sdev"]]^2
    prop <- (sum(eigenvalue[1:npc])/sum(eigenvalue)) * 100
    cat("Top ", npc, " principal components can explain ", prop,
        "% of data \n    variation", "\n")
    save(svd, eigenvalue, prop, file=file.path(results.dir,"pca_betas_auto.RData"))
}

setwd("/udd/reprk/projects/TOPMed/scripts")
cra.pheno <- read.csv(file=file.path(qc.dir, "data/phenotype/CRA_Phenotype_Data/COS_TRIO_pheno_1165.c
                as.is=TRUE, sep=",", stringsAsFactors=FALSE)

samplesheet.cra <- read.csv(file=file.path(cra.dir, "LEVEL1/SampleSheet.csv"),
                            as.is=TRUE, sep = ",", fill=T, stringsAsFactors=FALSE)

sex.mismatch <- read.table(file=file.path(cra.dir, "LEVEL2/sex_mismatch.txt"),
                           sep="\t", header=F,stringsAsFactors=FALSE)

# cra chanmine issues
```

```
# https://chanmine.bwh.harvard.edu/issues/20974
# https://chanmine.bwh.harvard.edu/issues/21321
# https://chanmine.bwh.harvard.edu/issues/20731

# fam file format
#A text file with no header line, and one line per sample with the following six fields:

#    Family ID ('FID')
#    Within-family ID ('IID'; cannot be '0')
#    Within-family ID of father ('0' if father isn't in dataset)
#    Within-family ID of mother ('0' if mother isn't in dataset)
#    Sex code ('1' = male, '2' = female, '0' = unknown)
#    Phenotype value ('1' = control, '2' = case, '-9'/'0'/non-numeric = missing data if case/control
```

```
cra.fam <- str_c(qc.dir, "/metadata/CRA.fam")
cra.fam <- fread(cra.fam)
colnames(cra.fam) <- c("FID","IID","FatherID","MotherID","Sex","Phenotype")
cra.fam$sex[cra.fam$Sex==1]<-"M"; cra.fam$sex[cra.fam$Sex==2]<-"F"
dim(cra.fam) # 5117; 2410 F, 2700 M, 7 NAs
```

```
## [1] 5117    7
```

```
# Save result files with timeStamp
timeStamp <- as.character(round(unclass(Sys.time())))
print(timeStamp)
```

```
## [1] "1620501723"
```

```
# Resource: https://github.com/markgene/maxprobes
cross_probes_file = paste(cra.dir, "/LEVEL2/cross_reactive_probes.txt",
                          sep = "")
if (!file.size(cross_probes_file) == 0){
    cross_probes = read.table(cross_probes_file, sep = "\t",
                              header = F, quote = "\"", fill = T)
    colnames(cross_probes) = c("sample")
    n_cross_probes = nrow(cross_probes)
    n_cross_probes
} else {
    n_cross_probes = 0
}
```

```
## [1] 44570
```

```
n_cross_probes # 44,570
```

```
## [1] 44570
```

```
fail.samps <- read.table(file=file.path(meff.dir, "qc/cra_failed_samples_metrics_hg19_1617744049.txt"
                         sep="\t", header=T,stringsAsFactors=FALSE)
```

2 Data preprocessing and filtering

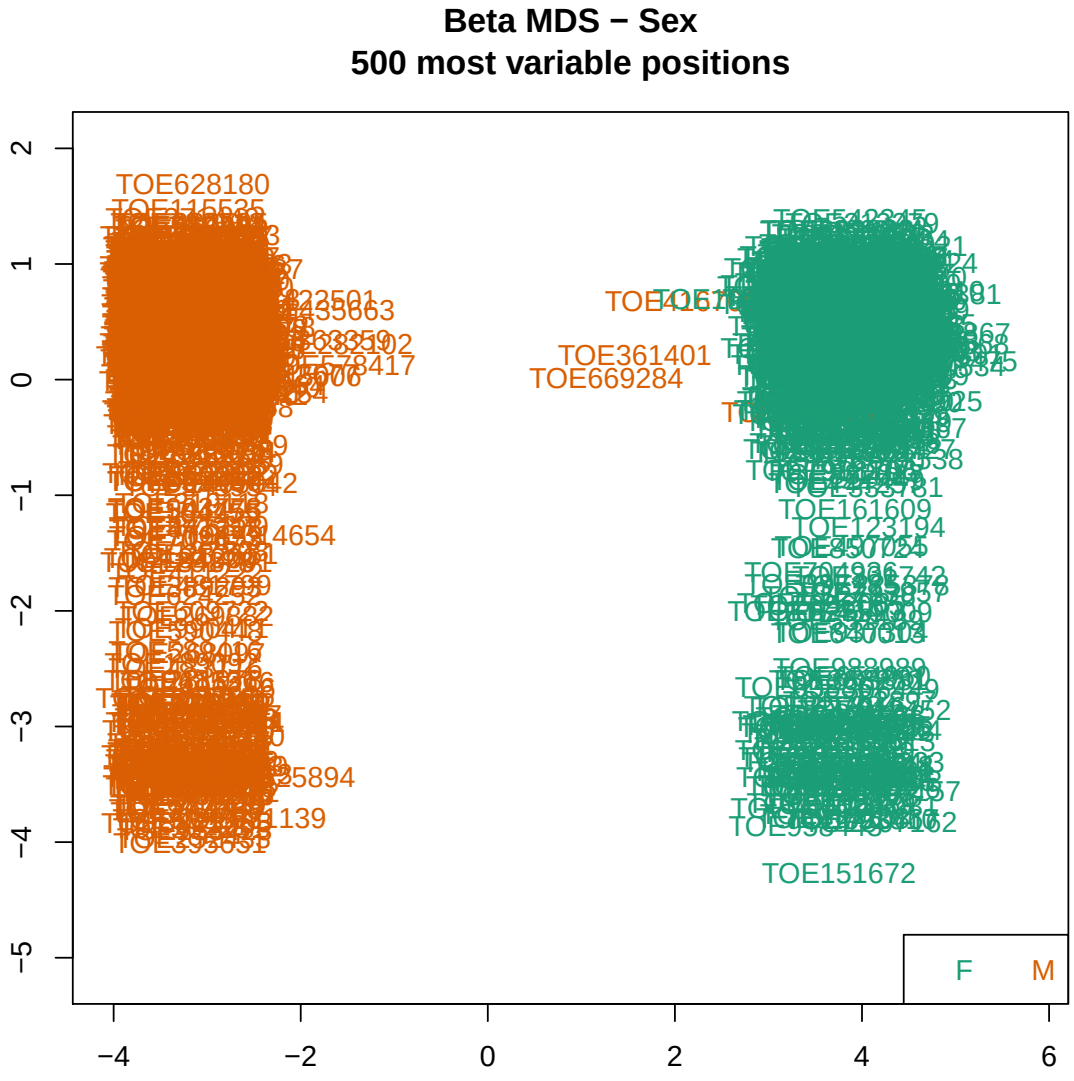
2.1 Failed Samples filtering and sex mismatches

```
################################
# CRA final analysis from RGSet
################################
```

```
sex.mismatch <- sex.mismatch$V1; length(sex.mismatch)
```

```
## [1] 6

sex.mismatch

## [1] "TOE692745-BIS-v01_R06C01" "TOE939842-BIS-v01_R06C01"
## [3] "TOE778267-BIS-v01_R02C01" "TOE194624-BIS-v01_R07C01"
## [5] "TOE890170-BIS-v01_R03C01" "TOE969516-BIS-v01_R06C01"

RGSet.cra=RGSet.cra[,!colnames(RGSet.cra) %in% sex.mismatch]

#first six samples are sex mismatches so removed
# samples with mixed genotype distributions on the measured SNP probes (59 SNP probes),
# indicating possible sample contamination (n=3) or sample mix ups
rem <- c("TOE283252-BIS-v01_R02C01",
         "TOE176389-BIS-v01_R08C01",
         "TOE722209-BIS-v01_R06C01")

RGSet.cra=RGSet.cra[,!colnames(RGSet.cra) %in% rem]
intersect(sex.mismatch, rem)

## character(0)

################################################
# Remove failed samples identified using meffil
################################################

# loaded this file in file loading section
# selected samples to exclude based on QC report
index <- fail.samps$issue %in% c("Control probe (dye.bias)",
                                 "Methylated vs Unmethylated",
                                 "Control probe (bisulfite1)",
                                 "Control probe (bisulfite2)",
                                 "Control probe (hybe.21771417)",
                                 "Control probe (hybe.28684356)",
                                 "Control probe (hybe.39782321)")

outlier <- fail.samps[index,]
dim(outlier) # 38

## [1] 38  2

failed.ids <- unique(outlier$sample.name) # 21
length(failed.ids); failed.ids # finally samples that will be removed

## [1] 21
##  [1] "TOE139227-BIS-v01_R02C01" "TOE156021-BIS-v01_R02C01"
##  [3] "TOE188467-BIS-v01_R02C01" "TOE306813-BIS-v01_R05C01"
##  [5] "TOE312709-BIS-v01_R08C01" "TOE360057-BIS-v01_R01C01"
##  [7] "TOE362104-BIS-v01_R03C01" "TOE429305-BIS-v01_R03C01"
##  [9] "TOE454440-BIS-v01_R08C01" "TOE526362-BIS-v01_R07C01"
## [11] "TOE666713-BIS-v01_R07C01" "TOE684725-BIS-v01_R04C01"
## [13] "TOE708113-BIS-v01_R01C01" "TOE716281-BIS-v01_R04C01"
## [15] "TOE751718-BIS-v01_R03C01" "TOE753420-BIS-v01_R01C01"
## [17] "TOE763963-BIS-v01_R01C01" "TOE881024-BIS-v01_R05C01"
## [19] "TOE912733-BIS-v01_R08C01" "TOE915473-BIS-v01_R01C01"
## [21] "TOE939080-BIS-v01_R02C01"

# checking overlap of 9 sex mismatches + genotype issues with failed meffil samples
intersect(sex.mismatch, failed.ids);intersect(failed.ids, rem)
```

```
## character(0)
## character(0)

RGSet.cra=RGSet.cra[,!colnames(RGSet.cra) %in% failed.ids]
dim(RGSet.cra)

## [1] 1008711    1207

betas.chk <- getBeta(RGSet.cra)
pData.cra <- pData(RGSet.cra)
ann850k <- getAnnotation(RGSet.cra)
xychr = rownames(betas.chk) %in% ann850k$Name[ann850k$chr %in% c("chrX","chrY")]
betas.xy = betas.chk[xychr,]

# shows the 3 sex outliers
mdsPlot(as.matrix(betas.chk), numPositions=500, main=sprintf("Beta MDS - Sex\n%d most variable positi
```




```
mdsPlot(as.matrix(betas.xy), numPositions=19627, main=sprintf("Beta MDS - Sex\n%d all sex chr positio
```


```
pdf(file = file.path(plots.dir, "MDS_sex_out_500pos_names.pdf"), width = 6, height = 6)
mdsPlot(as.matrix(betas.chk), numPositions=500, main=sprintf("Beta MDS - Sex\n%d most variable positi
dev.off()

## pdf
##   2

pdf(file = file.path(plots.dir, "MDS_sex_chr_out_19627pos_names.pdf"), width = 6, height = 6)
mdsPlot(as.matrix(betas.xy), numPositions=19627, main=sprintf("Beta MDS - Sex\n%d all sex chr positio
dev.off()

## pdf
##   2

# further removed the 3 sex outliers identified as above and through meffil norm
# report we don't remove all the sex outliers as they did not look concerning
sex.out <- c("TOE768143-BIS-v01_R01C01", "TOE473872-BIS-v01_R02C01", "TOE631417-BIS-v01_R02C01")
RGSet.cra=RGSet.cra[,!colnames(RGSet.cra) %in% sex.out]
dim(RGSet.cra) # 1204

## [1] 1008711    1204
```

```
# Do any overlap with those failed samples identified before?
intersect(rem, sex.out);intersect(failed.ids, sex.out)
```

```
## character(0)
## character(0)
```

```
# probably not needed anymore as these plots are also generated using meffil
#library(ENmix)
#jpeg(file = file.path(plots.dir, "ENmixcontrol_plots_CRA_bisulfite.jpg"),
#    width = 750, height = 1500)
#plotCtrl(RGSet.cra)
#dev.copy(jpeg,'ENmixcontrol_plots_CRA_bisulfite.jpg')
#dev.off()
```

2.2 detP calculation

```
##############
# Detection P
##############

detP.cra <- detectionP(RGSet.cra, type="m+u")

save(detP.cra, file=file.path(results.dir,paste0("detP.cra_hg19_", timeStamp,".RData")))
print(table(detP.cra>0.05))
```

```
##
##      FALSE       TRUE
## 1041583898     910338
```

```
print(table(detP.cra>0.01))
```

```
##
##      FALSE       TRUE
## 1041205731    1288505
```

```
# to check whether there are any probes with detP>0.05 across each sample,
# not doing this anymore, this is slightly different to what we are doing below
#M2<-matrix(runif(36),nrow=6);M2;M2.f<-M2>0.50;M2.f;colMeans(M2.f);rowMeans(M2.f);
#M3 <- M2;M3[M3>0.50] <- NA;M3;M3[rowMeans(M2.f)>0.50] <- NA;M3;M4 <- M2 + M3
#detP.cra[detP.cra > 0.05] <- NA
#detP.cra[detP.cra < 0.05] <- 0

#####################
# sample-wise thresh.
#####################
# colMedians(detP.cra) similar# robustbase r package
dim(detP.cra[,colMeans(detP.cra)>0.001])
```

```
## [1] 865859     33
```

```
dim(detP.cra[,colMeans(detP.cra)>0.002])
```

```
## [1] 865859     10
```

```
dim(detP.cra[,colMeans(detP.cra)>0.003])
```

```
## [1] 865859      6
```

```
dim(detP.cra[,colMeans(detP.cra)>0.004])
```

```
## [1] 865859      2
```

```
dim(detP.cra[,colMeans(detP.cra)>0.005])
```

```
## NULL
```

```
dim(detP.cra[,colMeans(detP.cra)>0.05]) # none
```

```
## [1] 865859      0
```

```
dim(detP.cra[,colMeans(detP.cra)>0.01]) # none
```

```
## [1] 865859      0
```

```
#########################
# Failed detP probes minfi
#########################
# Threshold of detP 0.01 in more than 25% of the samples using minfi stats
failed.01<-detP.cra > 0.01
#colMeans(failed.01) # Fraction of failed positions per sample
sum(colMeans(failed.01)>0.20) # >20% probes failed per sample
```

```
## [1] 0
```

```
sum(rowMeans(failed.01)>0.05)
```

```
## [1] 4204
```

```
sum(rowMeans(failed.01)>0.10)
```

```
## [1] 2325
```

```
sum(rowMeans(failed.01)>0.15)
```

```
## [1] 1621
```

```
sum(rowMeans(failed.01)>0.20) # should be same as length(failedProbes)
```

```
## [1] 1242
```

```
sum(rowMeans(failed.01)>0.25)
```

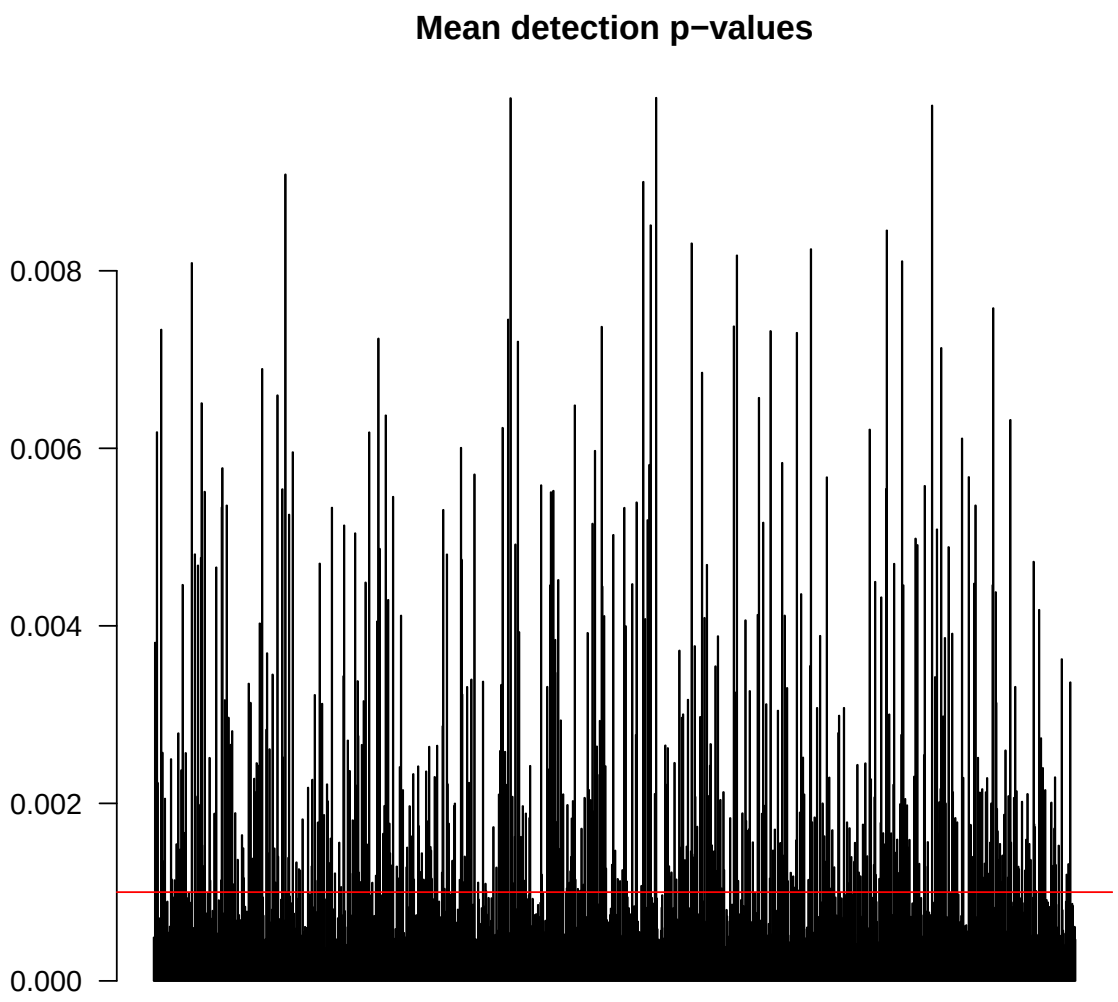
```
## [1] 993
```

```
# How many positions failed in >20% of samples?
failedProbes <- rownames(failed.01)[rowMeans(failed.01)>0.20]
length(failedProbes)
```

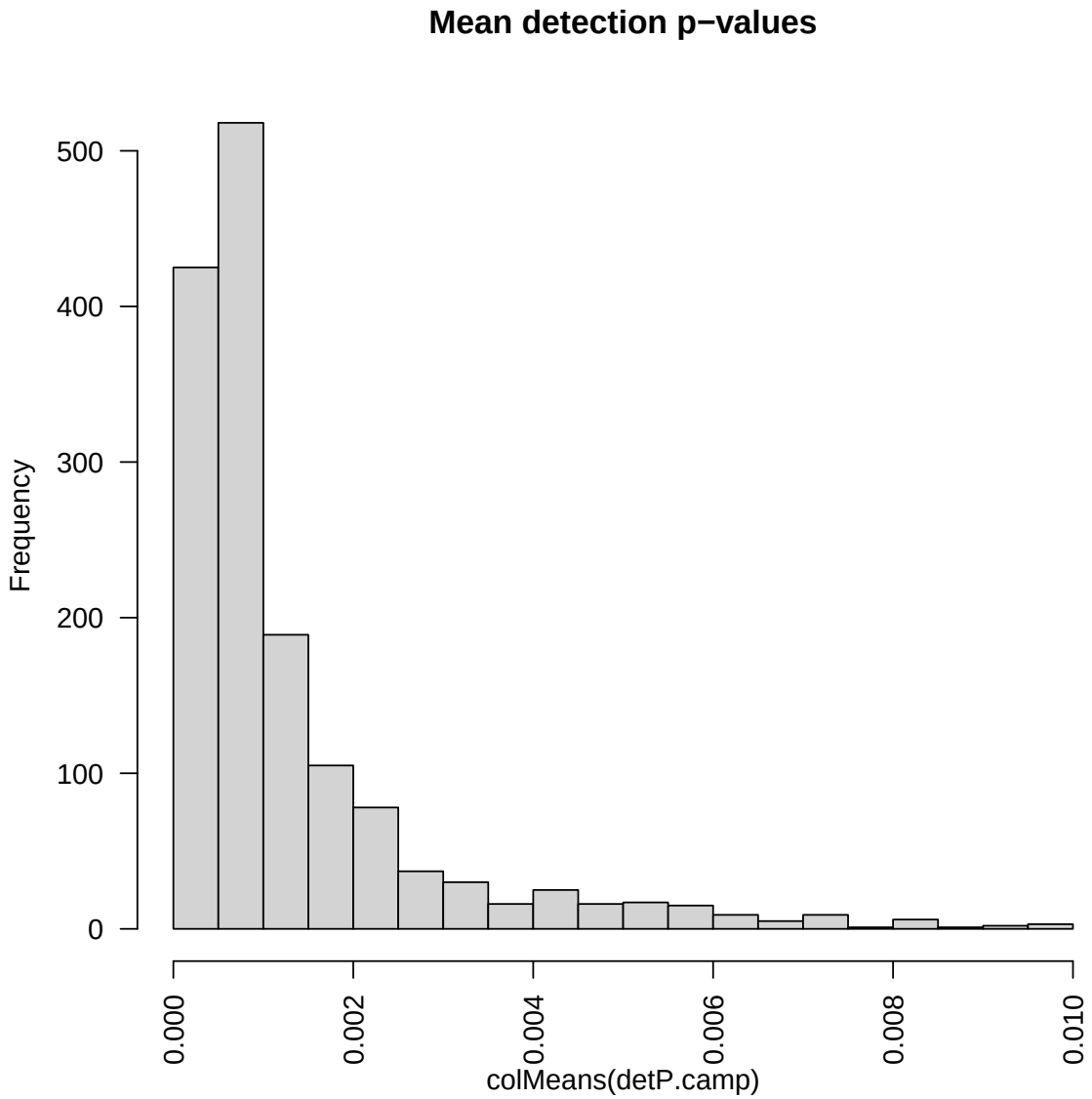
```
## [1] 1242
```

```
save(failedProbes, file=file.path(results.dir,paste0("failedProbes_CRA_hg19_", timeStamp,".RData")))
```

```
# plots
barplot(colMeans(detP.cra), las=2, axisnames = FALSE, main="Mean detection p-values")
abline(h=0.001,col="red")
```




```
hist(colMeans(detP.cra), las=2, main="Mean detection p-values", breaks=30)
```




```
pdf(file = file.path(plots.dir, "colMeans_detP_check_failed_samples_CRA.pdf"),
     width = 10, height = 5)
barplot(colMeans(detP.cra), las=2, axisnames = FALSE, main="Mean detection p-values")
abline(h=0.001,col="red")
dev.off()

## pdf
##   2

pdf(file = file.path(plots.dir, "colMeans_detP_all_samples_hist_CRA.pdf"),
     width = 6, height = 5)
hist(colMeans(detP.cra), las=2, main="Mean detection p-values", breaks=30)
dev.off()

## pdf
##   2

# plotting every sample, Not printing this because the loop will run for all
# and will likely print a huge plot
pdf(file = file.path(plots.dir, "detP_all_samples_hist_freq_CRA.pdf"),
     width = 156, height = 136)
```

```
par(mfrow=c(32, 38))
colnames <- dimnames(detP.cra)[[2]]
for (i in 1:1204) {
#print(i)
hist(log10(detP.cra[,i]), las=2, breaks=50, main=colnames[i], col="gray", border="white")
}
dev.off()
```

```
## pdf
##   2
```

2.3 svas and cell type count estimation

```
# generate surrogate variables derived based on intensity data
# for non-negative internal control probes
# this step is pretty quick
csva<-ctrlsva(RGSet.cra)
```

```
## 12  surrogate variables explain  95.12225 % of
##     data variation
```

```
save(csva, file=file.path(results.dir,paste0("CRA_svas_rawdata_hg19_", timeStamp, ".RData")))
```

```
# cell count estimates
library("FlowSorted.Blood.EPIC")
library(ExperimentHub)
hub <- ExperimentHub()
#> snapshotDate(): 2020-10-02
epicref <- query(hub, "FlowSorted.Blood.EPIC")
epicref; epicref$title
```

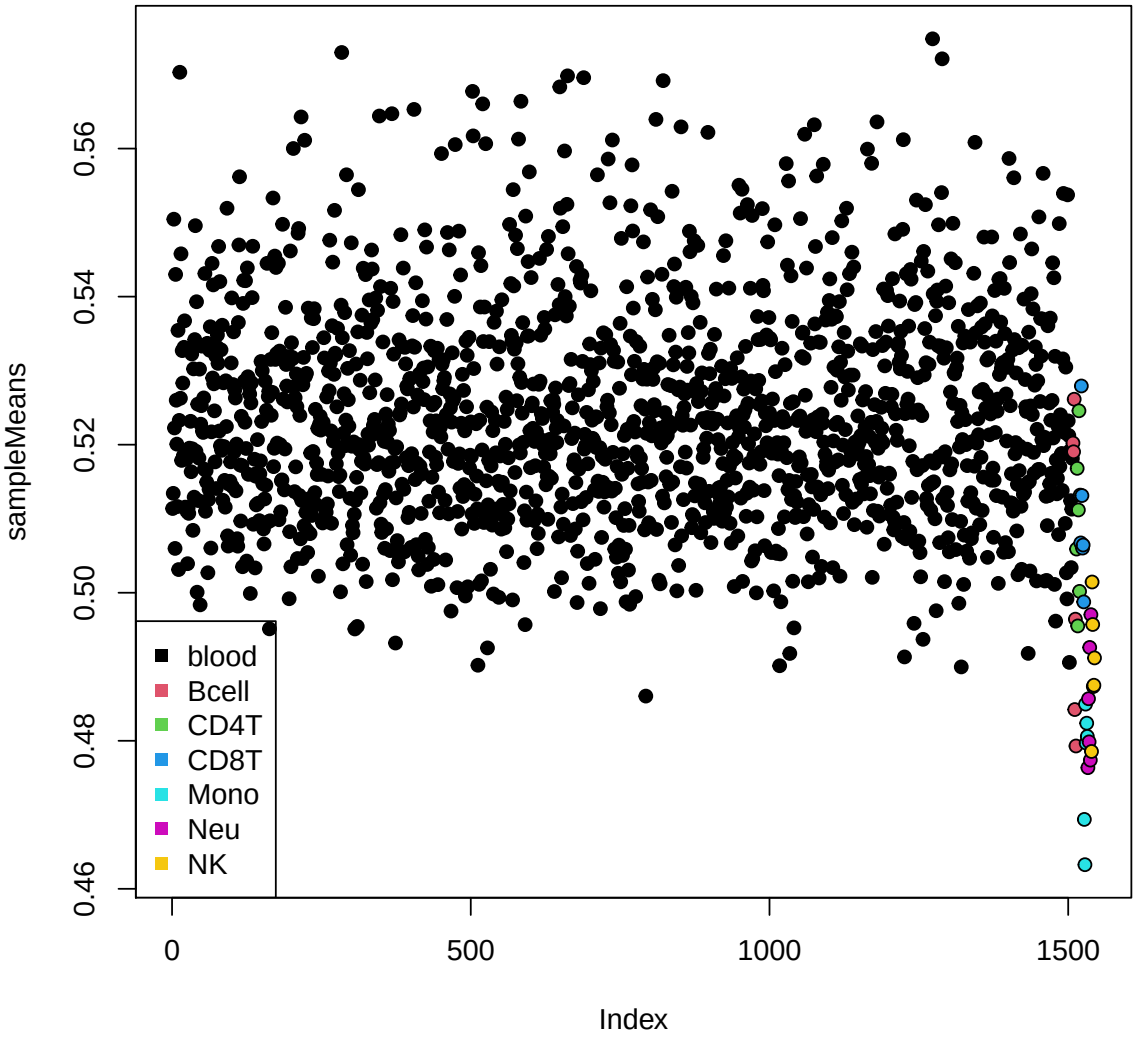
```
## ExperimentHub with 1 record
## # snapshotDate(): 2020-10-27
## # names(): EH1136
## # package(): FlowSorted.Blood.EPIC
## # $dataprovider: GEO
## # $species: Homo sapiens
## # $rdataclass: RGChannelSet
## # $rdatadateadded: 2018-04-20
## # $title: FlowSorted.Blood.EPIC: Illumina Human Methylation data from EPIC o...
## # $description: The FlowSorted.Blood.EPIC package contains Illumina HumanMet...
## # $taxonomyid: 9606
## # $genome: hg19
## # $sourcetype: tar.gz
## # $sourceurl: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE110554
## # $sourcesize: NA
## # $tags: c("ExperimentData", "Homo_sapiens_Data", "Tissue",
## #   "MicroarrayData", "Genome", "TissueMicroarrayData",
## #   "MethylationArrayData")
## # retrieve record with 'object[["EH1136"]]'
## [1] "FlowSorted.Blood.EPIC: Illumina Human Methylation data from EPIC on immunomagnetic sorted adu
```

```
FlowSorted.Blood.EPIC.ref <- epicref[[1]]
FlowSorted.Blood.EPIC.ref
```

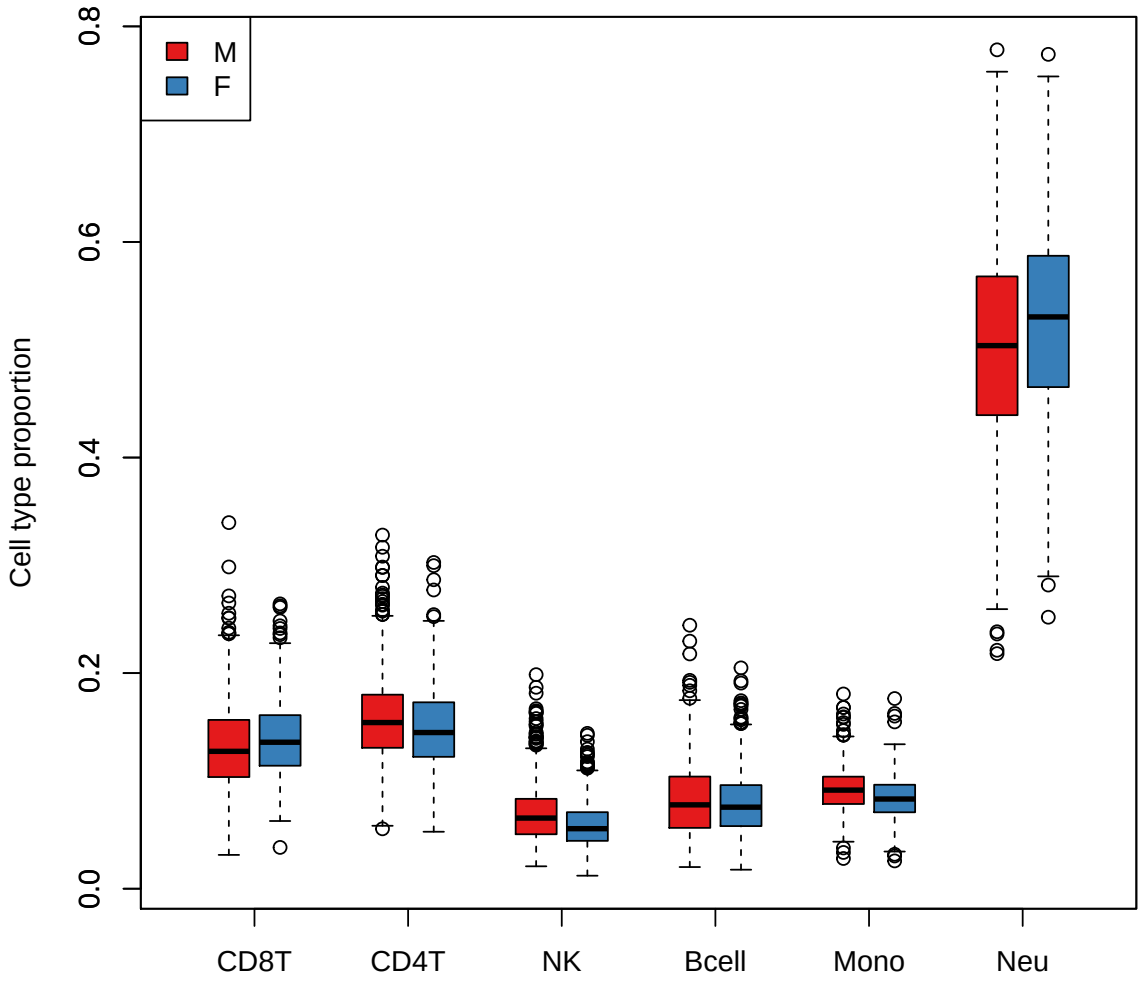
```
## class: RGChannelSet
## dim: 1051815 49
## metadata(0):
## assays(2): Green Red
```

```
## rownames(1051815): 1600101 1600111 ... 99810990 99810992
## rowData names(0):
## colnames(49): 201868500150_R01C01 201868500150_R03C01 ...
##   201870610111_R06C01 201870610111_R07C01
## colData names(32): Sample_Plate Sample_Well ... filenames normalmix
## Annotation
##   array: IlluminaHumanMethylationEPIC
##   annotation: ilm10b4.hg19
```

```
# meanPlot=TRUE: Whether to plots the average DNA methylation
#across the cell-type discrimating probes within the mixed
# and sorted samples.
if (memory.limit()>8000){
  countsEPIC<-estimateCellCounts2(RGSet.cra, compositeCellType = "Blood",
                                  processMethod = "preprocessFunnorm",
                                  cellTypes = c("CD8T", "CD4T", "NK", "Bcell",
                                                "Mono", "Neu"),
                                  referencePlatform =
                                      "IlluminaHumanMethylationEPIC",
                                  referenceset = "FlowSorted.Blood.EPIC.ref",
                                  IDOLOptimizedCpGs =NULL,
                                  returnAll = TRUE,
                                  meanPlot = TRUE,
                                  verbose = TRUE)
}
```



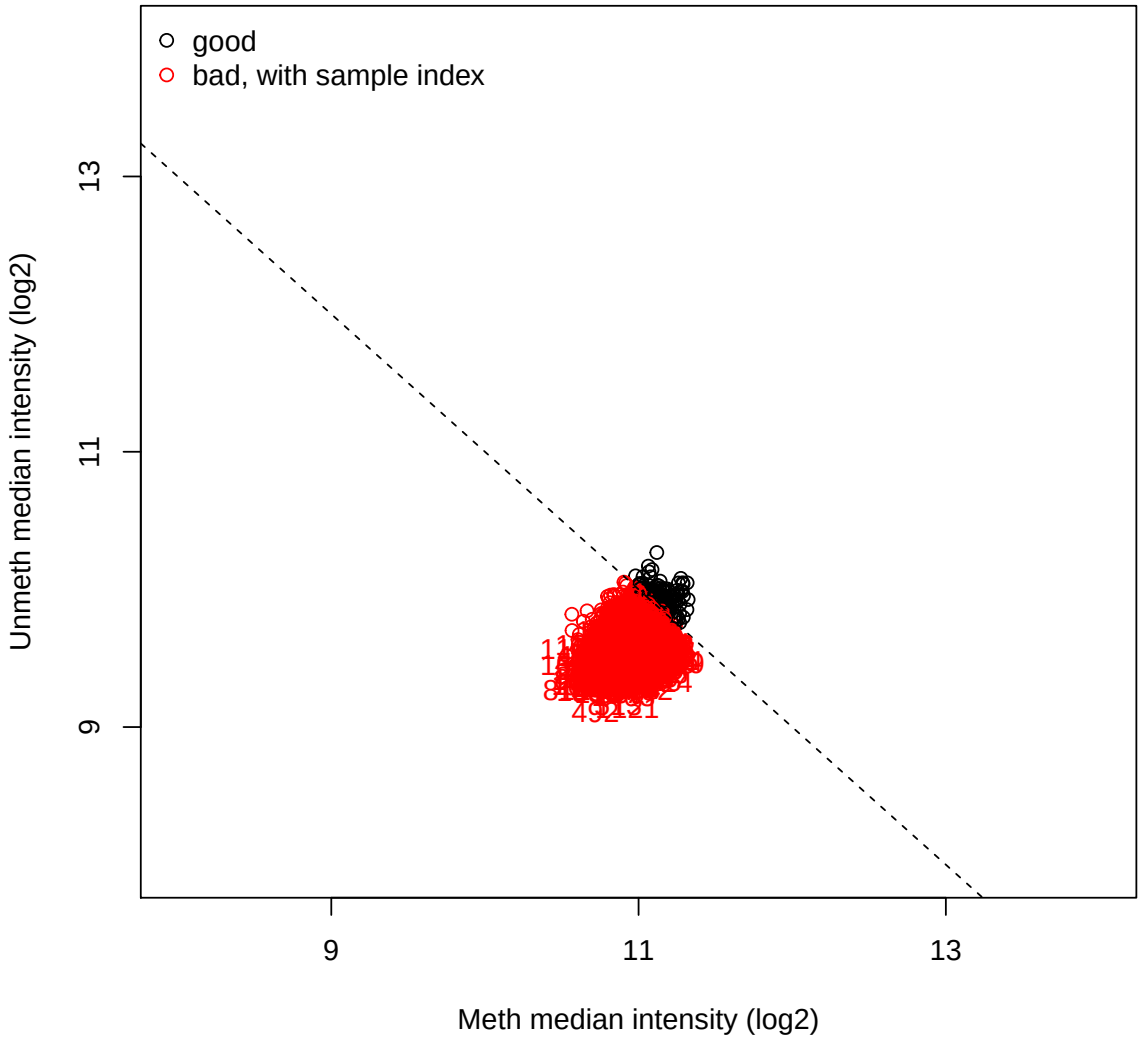

```
save(countsEPIC, file=file.path(results.dir,paste0("CRA_EPIC_estimatecellcounts_hg19_",
                                                   timeStamp,".RData")))
celltype.est.2 <- countsEPIC$counts
save(celltype.est.2, file=file.path(results.dir, paste0("CRA_EPIC_estimatecellcounts2_result_hg19_",
                                                        timeStamp, ".RData")))

TOE <- data.frame(do.call('rbind', strsplit(as.character(samplesheet.cra$Basename),'/',fixed=TRUE)))
samplesheet.cra$TOE_RC <- TOE$X14
ct.sampsheet <- merge(celltype.est.2, samplesheet.cra, by.x="row.names", by.y="TOE_RC", sort=F)
#cra.pheno.sel=cra.pheno[,c("cra","S_SUBJECTID"), drop=FALSE]
#ct.pheno <- merge(ct.sampsheet, cra.pheno.sel, by="S_SUBJECTID", sort=F)
par(mfrow=c(1,1));sex.pal <- brewer.pal(8,"Set1")
a = celltype.est.2[ct.sampsheet$Gender == "M",]
b = celltype.est.2[ct.sampsheet$Gender == "F",]
boxplot(a, at=0:5*3 + 1, xlim=c(0, 18), ylim=range(a, b), xaxt="n",
        col=sex.pal[1], main="", ylab="Cell type proportion")
boxplot(b, at=0:5*3 + 2, xaxt="n", add=TRUE, col=sex.pal[2])
axis(1, at=0:5*3 + 1.5, labels=colnames(a), tick=TRUE)
legend("topleft", legend=c("M","F"), fill=sex.pal)
```



2.4 Noob normalization and funnorm

```
# clearing up some objects from memory no longer needed
rm(countsEPIC)
rm(celltype.est.2)
rm(detP.cra)
rm(csva)
######################
# NOOB Normalization
######################

MSet.noob.cra <- preprocessNoob(RGSet.cra, offset = 15, dyeCorr = TRUE, verbose = TRUE)

#############################################################################
# This object is sample cleaned but probe filtering has not been performed for this,
# check probe filtering steps below to identify which probes need to be set to NA
# if you use this object for any further downstream analysis
#############################################################################
```

```
save(MSet.noob.cra, file=file.path(results.dir,paste0("Mset.noob.cra_hg19_",
                                                      timeStamp,".RData")))
MSet.noob.cra <- NULL

################################################################################
# shows that phenotype file and LIMS/samplesheet genders match for the probands
################################################################################
pData.cra <- pData(RGSet.cra)
pData.pheno <- merge(pData.cra, cra.pheno, by="S_SUBJECTID", sort=F)
table(pData.pheno$Gender, pData.pheno$gender)

##
##       F   M
##   F 324   0
##   M   0 479

pData.cra$Sex[pData.cra$Gender=="F"]<-0
pData.cra$Sex[pData.cra$Gender=="M"]<-1
sex <- pData.cra$Sex

# Runs for few hours, I wanted the output to be in mset format, otherwise, I can not extract methyla
mset.cra.funnorm <- preprocessFunnorm(RGSet.cra, nPCs=3, sex=sex, ratioConvert = FALSE,
                                      bgCorr=TRUE, dyeCorr=TRUE, verbose=TRUE)
mset.cra.funnorm <- addSex(mset.cra.funnorm)

############################################
# Dataset after functional normalization
############################################
save(mset.cra.funnorm, file=file.path(results.dir,paste0("mset.cra.funnorm_hg19_",
                                                         timeStamp,".RData")))

betas <- getBeta(mset.cra.funnorm)
ann850k <- getAnnotation(mset.cra.funnorm)
#pData.cra <- pData(mset.cra.funnorm)

# median meth and unmeth intensities plot
qc <- getQC(mset.cra.funnorm)
meds <- (qc$mMed + qc$uMed)/2
mMed <- qc@listData$mMed
uMed <- qc@listData$uMed
qc.cra <- data.frame(mMed, uMed, meds)
rownames(qc.cra) <- qc@rownames
dim(qc.cra[qc.cra$meds<10.5,])

## [1] 0 3

plotQC(qc)
```




```
# Plots
pdf(file = file.path(plots.dir, "QC_samples_minfi_detP0.01_20per_samples_CRA.pdf"),
      width = 5, height = 5)
plotQC(qc)
dev.off()
```

```
## pdf
##   2
```

3 Session information

[1] "2021-05-09" [1] "2021-05-09 01:19:09 EDT"

- R version 4.0.3 (2020-10-10), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=en_US.UTF-8`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Running under: `CentOS Linux 7 (Core)`

- Matrix products: default
- BLAS: `/app/R-4.0.3@i86-rhel7.0/lib64/R/lib/libRblas.so`
- LAPACK: `/app/R-4.0.3@i86-rhel7.0/lib64/R/lib/libRlapack.so`
- Base packages: base, datasets, graphics, grDevices, methods, parallel, stats, stats4, utils
- Other packages: annotate 1.68.0, AnnotationDbi 1.52.0, AnnotationHub 2.22.1, Biobase 2.50.0, BiocFileCache 1.14.0, BiocGenerics 0.36.1, BiocParallel 1.24.1, Biostrings 2.58.0, bumphunter 1.32.0, Cairo 1.5-12.2, colorRamps 2.3, data.table 1.14.0, dbplyr 2.1.0, DNAcopy 1.64.0, doParallel 1.0.16, dplyr 1.0.3, e1071 1.7-6, ENmix 1.26.10, ExperimentHub 1.16.1, fastICA 1.2-2, FDb.InfiniumMethylation.hg19 2.2.0, FlowSorted.Blood.EPIC 1.8.0, forcats 0.5.1, foreach 1.5.1, gdsfmt 1.26.1, genefilter 1.72.1, geneplotter 1.68.0, GenomeInfoDb 1.26.7, GenomicFeatures 1.42.3, GenomicRanges 1.42.0, GGally 2.1.0, ggplot2 3.3.3, ggrepel 0.9.1, gplots 3.1.1, gridExtra 2.3, here 1.0.1, IlluminaHumanMethylation450kanno.ilmn12.hg19 0.6.0, IlluminaHumanMethylationEPICanno.ilm10b4.hg19 0.6.0, IlluminaHumanMethylationEPICmanifest 0.3.0, illuminaio 0.32.0, impute 1.64.0, IRanges 2.24.1, isva 1.9, iterators 1.0.13, JADE 2.0-3, knitr 1.33, lattice 0.20-44, limma 3.46.0, lme4 1.1-26, locfit 1.5-9.4, lumi 2.42.0, markdown 1.1, MASS 7.3-54, Matrix 1.3-3, MatrixGenerics 1.2.1, matrixStats 0.58.0, meffil 1.1.1, methylumi 2.36.0, mgcv 1.8-35, minfi 1.36.0, multcomp 1.4-17, mvtnorm 1.1-1, nlme 3.1-152, org.Hs.eg.db 3.12.0, plyr 1.8.6, preprocessCore 1.52.1, purrr 0.3.4, quadprog 1.5-8, qvalue 2.22.0, R.methodsS3 1.8.1, R.oo 1.24.0, R.utils 2.10.1, RColorBrewer 1.1-2, readr 1.4.0, reshape2 1.4.4, robustbase 0.93-7, ROC 1.66.0, RSpectra 0.16-0, S4Vectors 0.28.1, scales 1.1.1, SmartSVA 0.1.3, statmod 1.4.35, stringi 1.5.3, stringr 1.4.0, SummarizedExperiment 1.20.0, survival 3.2-11, sva 3.38.0, TH.data 1.0-10, tibble 3.1.1, tidyr 1.1.3, tidyverse 1.3.0, TxDb.Hsapiens.UCSC.hg19.knownGene 3.2.2, wateRmelon 1.34.0, XML 3.99-0.6, XVector 0.30.0
- Loaded via a namespace (and not attached): affy 1.68.0, affyio 1.60.0, askpass 1.1, assertthat 0.2.1, backports 1.2.1, base64 2.0, beanplot 1.2, BiocManager 1.30.12, BiocVersion 3.12.0, biomaRt 2.46.3, bit 4.0.4, bit64 4.0.5, bitops 1.0-7, blob 1.2.1, boot 1.3-28, broom 0.7.6, cachem 1.0.4, caTools 1.18.2, cellranger 1.1.0, class 7.3-19, cli 2.5.0, clue 0.3-59, cluster 2.1.2, codetools 0.2-18, colorspace 2.0-1, compiler 4.0.3, crayon 1.4.1, curl 4.3.1, DBI 1.1.1, DelayedArray 0.16.3, DelayedMatrixStats 1.12.3, DEoptimR 1.0-8, digest 0.6.27, doRNG 1.8.2, dynamicTreeCut 1.63-1, edgeR 3.32.1, ellipsis 0.3.2, evaluate 0.14, fansi 0.4.2, fastmap 1.1.0, fs 1.5.0, generics 0.1.0, GenomeInfoDbData 1.2.4, GenomicAlignments 1.26.0, GEOquery 2.58.0, glue 1.4.2, grid 4.0.3, gtable 0.3.0, gtools 3.8.2, haven 2.4.1, HDF5Array 1.18.1, highr 0.9, hms 1.0.0, htmltools 0.5.1.1, httpuv 1.6.0, httr 1.4.2, interactiveDisplayBase 1.28.0, irr 0.84.1, jsonlite 1.7.2, KernSmooth 2.23-20, later 1.2.0, lifecycle 0.2.0, lpSolve 5.6.15, lubridate 1.7.10, magrittr 2.0.1, mclust 5.4.7, memoise 2.0.0, mime 0.10, minqa 1.2.4, modelr 0.1.8, multtest 2.46.0, munsell 0.5.0, nleqslv 3.3.2, nloptr 1.2.2.2, nor1mix 1.3-0, openssl 1.4.4, pillar 1.6.0, pkgconfig 2.0.3, prettyunits 1.1.1, progress 1.2.2, promises 1.2.0.1, proxy 0.4-25, ps 1.6.0, R6 2.5.0, rappdirs 0.3.3, Rcpp 1.0.6, RCurl 1.98-1.3, readxl 1.3.1, reprex 2.0.0, reshape 0.8.8, rhdf5 2.34.0, rhdf5filters 1.2.1, Rhdf5lib 1.12.1, rlang 0.4.9, rngtools 1.5, RPMM 1.25, rprojroot 2.0.2, Rsamtools 2.6.0, RSQLite 2.2.3, rstudioapi 0.13, rtracklayer 1.50.0, rvest 0.3.6, sandwich 3.0-0, scrime 1.3.5, shiny 1.6.0, siggenes 1.64.0, sparseMatrixStats 1.2.1, splines 4.0.3, tidyselect 1.1.1, tools 4.0.3, utf8 1.2.1, vctrs 0.3.6, withr 2.4.2, xfun 0.22, xml2 1.3.2, xtable 1.8-4, yaml 2.2.1, zlibbioc 1.36.0, zoo 1.8-9