TOPMed CRA DNA methylation betas cleaning

reprk: PKachroo
CDNM, BWH

May 12, 2021

Contents

1 Setup

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

options(mc.cores=4)
system("hostname")
print(Sys.Date())
```

```
## [1] "2021-05-12"
```

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

```
## [1] "2021-05-12 10:37:36 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_betas_clean.Rnw")'
# pdflatex TOPMed_CRA_betas_clean.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")

# 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")

# scripts/code directory
setwd("/udd/reprk/projects/TOPMed/scripts")
# modified RCP code
source("RCP_mod.R")
source("LociWithSnps.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_CRA.RData"))
}

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)

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

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

```
## [1] "1620830664"
```

```
# 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
```

2 Mset loading

2.1 Failed probes loading

```
##############################
# CRA funnorm-normalized mset
##############################
load(file=file.path(results.dir, "mset.cra.funnorm_hg19_1620501723.RData"))

##############################
# Failed probes based on detP
# see QC code for details
##############################
load(file=file.path(results.dir, "failedProbes_CRA_hg19_1620501723.RData"))
length(failedProbes)
```

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

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

3 Probe level cleaning and rcp

```
########################
# Probe filtering stats
########################

########################################
# Failed probes identified using meffil
# not using them though, just to stay
# within the minfi framework for this
########################################
#dp <- meffil.load.detection.pvalues(qc.objects)
fail.cgs <- read.table(file=file.path(meff.dir, "qc/cra_failed_cgs_hg19_1617744049.txt"),
                       sep="\t", header=T,stringsAsFactors=FALSE)
fail.cgs <- fail.cgs$x
length(fail.cgs)
```

```
## [1] 747
```

```
length(intersect(failedProbes, fail.cgs))
```

```
## [1] 670
```

```
# we aren't dropping probes anymore
# drop non CpG probes and probes that measure SNPs
#mset.cra.funnorm <- dropMethylationLoci(mset.cra.funnorm, dropRS = TRUE, dropCH = TRUE)
#dim(mset.cra.funnorm_fil)
#mset.cra.funnorm <- dropLociWithSnps(mset.cra.funnorm, snps=c("SBE","CpG"), maf=0.05)
#dim(mset.cra.funnorm)

# Remove sex chr or Pull data on sex chromosomes if needed
# to remove
#xychr = !(featureNames(mset.cra.funnorm) %in% ann850k$Name[ann850k$chr %in% c("chrX","chrY")])
# mset.cra.funnorm = mset.cra.funnorm[xychr,]

# both sex chromosomes
xychr = (featureNames(mset.cra.funnorm) %in% ann850k$Name[ann850k$chr %in% c("chrX","chrY")])
table(xychr)
```

```
## xychr
##  FALSE   TRUE
## 846232  19627
```

```
auto = !(featureNames(mset.cra.funnorm) %in% ann850k$Name[ann850k$chr %in% c("chrX","chrY")])
mset.auto = mset.cra.funnorm[auto,]
auto.probes <- featureNames(mset.auto)
length(auto.probes) # used later to extract betas from autosomes
```

```
## [1] 846232
```

```
rm(mset.auto) # to clear out memory

# count sex chromosomes individually
dim(ann850k[ann850k$chr=="chrX",])
```

```
## [1] 19090    46
```

```
dim(ann850k[ann850k$chr=="chrY",])
```

```
## [1] 537  46
```

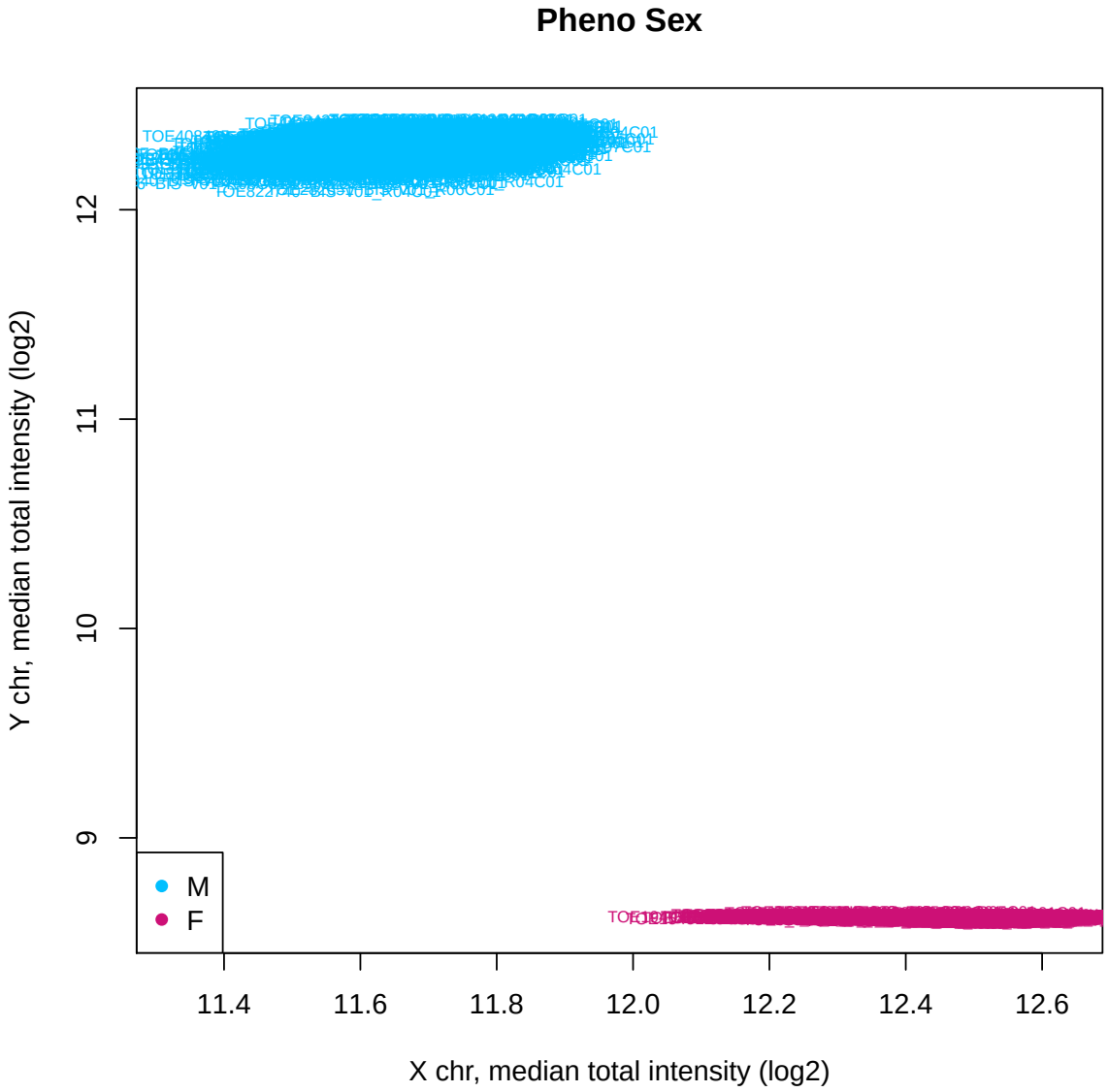
```
#Gender check plot using median total intensities X and Y chr
# predictedSex <- getSex(mset.cra.funnorm, cutoff = -2)$predictedSex
xy <- getSex(mset.cra.funnorm, cutoff = -2)
xy$sex <- pData.cra$Gender
head(xy)
```

```
## DataFrame with 6 rows and 4 columns
##                              xMed      yMed predictedSex         sex
##                         <numeric> <numeric>  <character> <character>
## TOE909374-BIS-v01_R08C01   12.7689   8.73861            F           F
## TOE239310-BIS-v01_R01C01   12.6702   8.72893            F           F
## TOE665077-BIS-v01_R07C01   12.4876   8.73639            F           F
## TOE971486-BIS-v01_R03C01   11.9102  12.59374            M           M
## TOE204245-BIS-v01_R07C01   12.7321   8.74032            F           F
## TOE535942-BIS-v01_R04C01   11.9055  12.55428            M           M
```

```
table(xy$sex, xy$predictedSex) # should be same
```

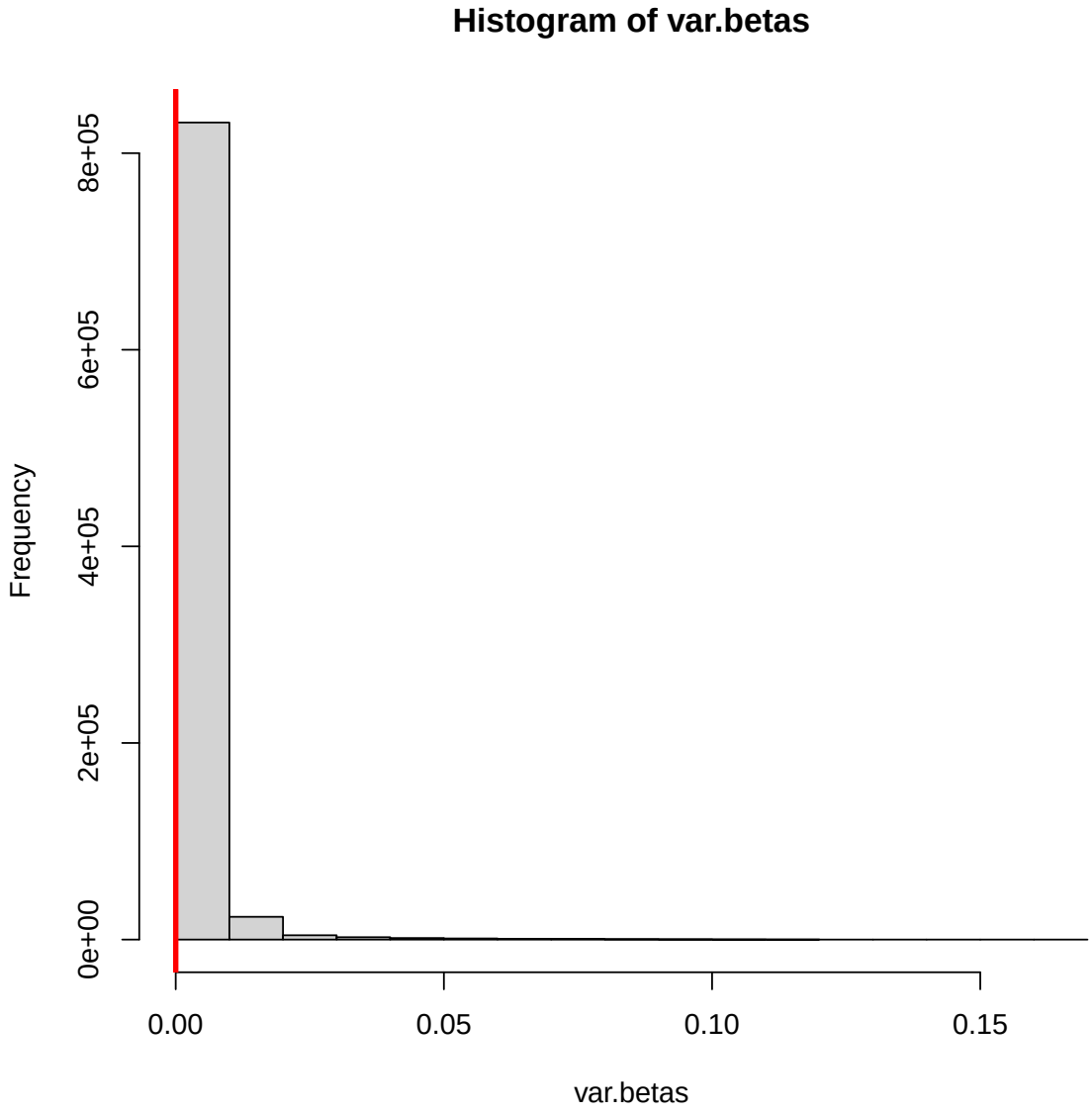
```
##
##       F   M
##   F 525   0
##   M   0 679
```

```
# Plots based on phenotype sex
plot(xy$xMed, xy$yMed, type = "n", main="Pheno Sex",
     xlab = "X chr, median total intensity (log2)",
     ylab = "Y chr, median total intensity (log2)")
id <- rownames(xy)
colors=c("deeppink3","deepskyblue")
text(xy$xMed, xy$yMed, id, col=colors[as.factor(xy$sex)], cex=0.6)
legend("bottomleft", c("M", "F"),
       col = c("deepskyblue", "deeppink3"), pch = 16)
```




```
# there shouldn't be any sex mismatches at this stage
pdf(file=file.path(plots.dir, "gender_check_xy_sex_CRA.pdf"),
    width = 5, height = 5)
plot(xy$xMed, xy$yMed, type = "n", main="Pheno Sex",
     xlab = "X chr, median total intensity (log2)",
     ylab = "Y chr, median total intensity (log2)")
id <- rownames(xy)
colors=c("deeppink3","deepskyblue")
text(xy$xMed, xy$yMed, id, col=colors[as.factor(xy$sex)], cex=0.6)
legend("bottomleft", c("M", "F"),
       col = c("deepskyblue", "deeppink3"), pch = 16)
dev.off()

## pdf
##   2

################################################
# This block of code would only be needed if we do rcp on methylset,
# but since we are doing all cleaning on betas, I have applied modified
# RCP code on betas
################################################
```

```
#head(getMeth(mset.cra.funnorm)[,1:3])
#head(getUnmeth(mset.cra.funnorm)[,1:3])
#meth <- getMeth(mset.cra.funnorm)
#unmeth <- getUnmeth(mset.cra.funnorm)
#methset <- MethylSet(Meth = meth, Unmeth = unmeth)
#methset@annotation <- RGSet.cra@annotation
#betas.rcp.cra <- rcp(methset)
################################################

################################
# Minimum variance pruning (just checking)
################################
var.betas <- rowVars(betas)
cutoff <- quantile(var.betas,0.01, na.rm=TRUE)
print(table(var.betas > cutoff))

##
##  FALSE   TRUE
##   8659 857200

hist(var.betas)
abline(v=cutoff,col='red',lwd=3)
```




```
#betas.use <- betas[var.betas > cutoff,]
#dim(betas.use)
########################################

# cross reactive probes and CH/rs probes
crossprobes <- cross_probes$sample
ch <- grep("^ch.", rownames(ann850k), value=TRUE); length(ch)
```

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

```
rs <- grep("^rs", rownames(ann850k), value=TRUE); length(rs)
```

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

```
# probes with SNPs at the single base extension (minor allele frequency (MAF) >5%), probes containing
cpg.snpsUP <- LociWithSnps(mset.cra.funnorm, snps=c("SBE","CpG"), maf=0.05)
dim(cpg.snpsUP);head(cpg.snpsUP)
```

```
## [1] 11681     7
## DataFrame with 6 rows and 7 columns
##               Probe_rs Probe_maf      CpG_rs   CpG_maf      SBE_rs   SBE_maf
##            <character> <numeric> <character> <numeric> <character> <numeric>
```

```
## cg09139287          NA        NA   rs2905055  0.336368   rs2905055  0.336368
## cg05321646          NA        NA  rs74714520  0.428571  rs74714520  0.428571
## cg13692836          NA        NA  rs13303328  0.145985          NA        NA
## cg06624358          NA        NA  rs56024075  0.070023          NA        NA
## cg10644916          NA        NA   rs3813184  0.087352   rs3813184  0.087352
## cg10625579          NA        NA  rs76233940  0.066406  rs76233940  0.066406
##               rownos
##            <integer>
## cg09139287       103
## cg05321646       133
## cg13692836       208
## cg06624358       210
## cg10644916       288
## cg10625579       380
```

```
save(cpg.snpsUP, file=file.path(results.dir,
            paste0("cpg_snpsUP_", timeStamp,".RData")))
```

```
cpg.snpsUP <- rownames(cpg.snpsUP)
length(cpg.snpsUP)
```

```
## [1] 11681
```

```
# just checking stats for 2 of the SUPs
summary(betas["cg09139287",])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1152  0.3336  0.4221  0.4475  0.5517  0.8622
```

```
summary(betas["cg05321646",])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.6133  0.6901  0.7121  0.7134  0.7340  0.8525
```

```
rm(mset.cra.funnorm) # clear memory, would not need this further in the code
```

```
################################################
# Probe filtering-cleaning/setting to missing
################################################
```

```
dim(betas)
```

```
## [1] 865859   1204
```

```
betas[(rownames(betas) %in% cpg.snpsUP),] <- NA
# gives an idea about how many probes set to missing at each step
dim(na.omit(betas))
```

```
## [1] 854178   1204
```

```
# detP>0.01 threshold in more than 20% of the samples
betas[(rownames(betas) %in% failedProbes),] <- NA
dim(na.omit(betas))
```

```
## [1] 853218   1204
```

```
betas[(rownames(betas) %in% crossprobes),] <- NA
dim(na.omit(betas))
```

```
## [1] 809852   1204
```

```
betas[(rownames(betas) %in% ch),] <- NA
dim(na.omit(betas))
```

```
## [1] 808573   1204
```

```
betas[(rownames(betas) %in% rs),] <- NA
dim(na.omit(betas))
```

```
## [1] 808573   1204
```

```
##############################
# RCP on noob normalized and
# cleaned betas
##############################
dim(betas)
```

```
## [1] 865859   1204
```

```
dim(ann850k)
```

```
## [1] 865859     46
```

```
norm.betas.rcp <- rcp.mod(betas, ann850k)
```

```
#########################################################################
# Normalized betas including parents and probands and all chromosomes
#########################################################################
save(norm.betas.rcp, ann850k, file=file.path(results.dir,
      paste0("norm.betas.cra_rcp_hg19_clean_allchr_", timeStamp,".RData")))

# frequency distribution plots for funnorm betas before RCP
# again this takes a while so not printing in report, but saving it
beta1=betas[ann850k$Type=="I",]
beta2=betas[ann850k$Type=="II",]

jpeg(file = file.path(plots.dir, "freq_distribution_norm_betas_CRA.jpg"),
    width = 750, height = 1500)
#jpeg("distributions_CAMP.jpg",height=900,width=500)
par(mfrow=c(3,1))
multifreqpoly(betas,main="Multifreqpoly",xlab="Beta value")
multifreqpoly(beta1,main="Multifreqpoly: Infinium I", xlab="Beta value")
multifreqpoly(beta2,main="Multifreqpoly: Infinium II", xlab="Beta value")
dev.off()
```

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

```
# frequency distribution plots for funnorm betas after RCP
beta1=norm.betas.rcp[ann850k$Type=="I",]
beta2=norm.betas.rcp[ann850k$Type=="II",]

jpeg(file = file.path(plots.dir, "freq_distribution_norm_betas_rcp_CRA.jpg"),
    width = 750, height = 1500)
par(mfrow=c(3,1))
multifreqpoly(norm.betas.rcp,main="Multifreqpoly",xlab="Beta value")
multifreqpoly(beta1,main="Multifreqpoly: Infinium I", xlab="Beta value")
multifreqpoly(beta2,main="Multifreqpoly: Infinium II", xlab="Beta value")
dev.off()
```

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

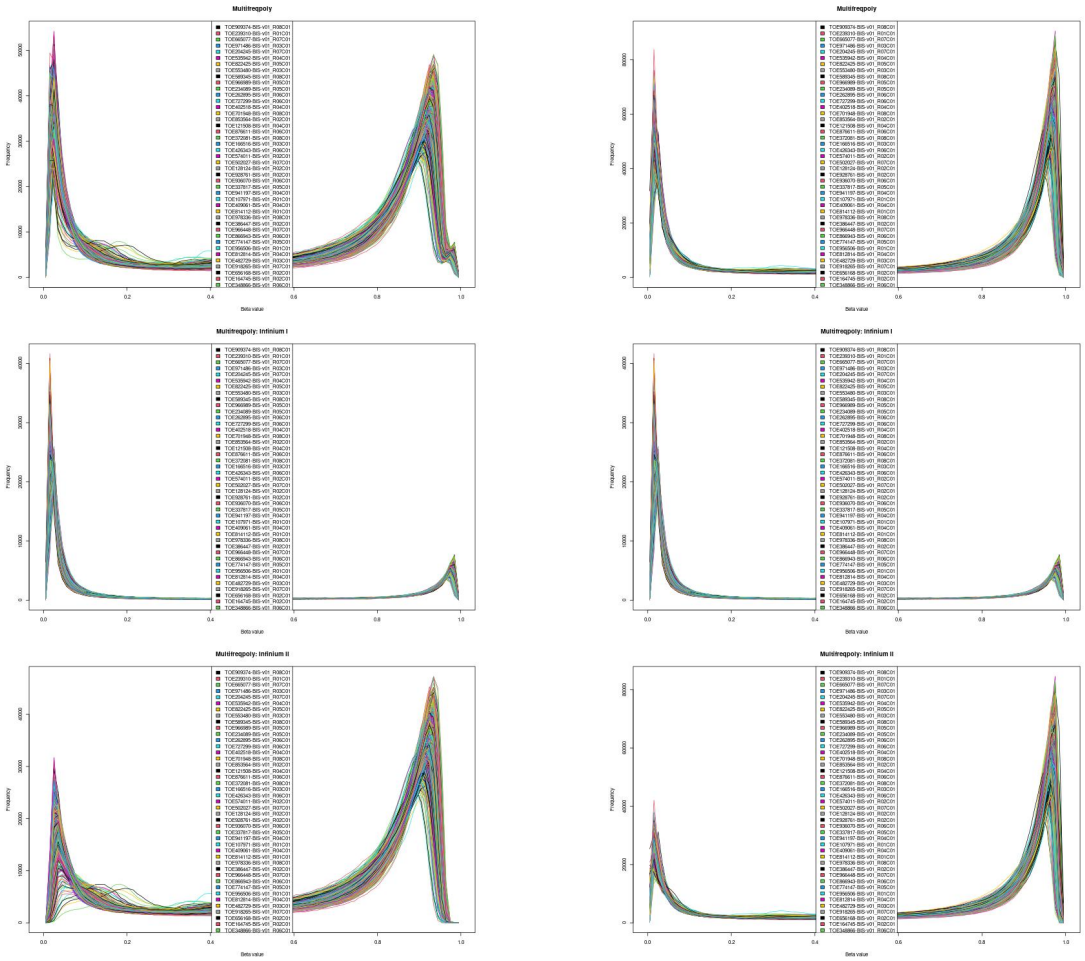


Figure 1: a) Normalized Distribution before rcp b) Normalized Distribution after rcp

3.1 Beta to m-value conversion

```
###################
# Beta to M values
###################
rm(betas) # clearing some memory
mvals <- beta2m(norm.betas.rcp)
save(mvals, file=file.path(results.dir,paste0("norm.mvals.cra_hg19_clean_allchr_",
                                              timeStamp,".RData")))

head(pData.cra)
```

```
## DataFrame with 6 rows and 10 columns
##                                       Basename  S_SAMPLEID S_SUBJECTID
##                                    <character> <character> <character>
## TOE909374-BIS-v01_R08C01 LEVEL1//TOE909374-BI.. S-000663478 ST-00068220
## TOE239310-BIS-v01_R01C01 LEVEL1//TOE239310-BI.. S-000673909 ST-00068476
## TOE665077-BIS-v01_R07C01 LEVEL1//TOE665077-BI.. S-000667801 ST-00047538
## TOE971486-BIS-v01_R03C01 LEVEL1//TOE971486-BI.. S-001397772 ST-00067117
## TOE204245-BIS-v01_R07C01 LEVEL1//TOE204245-BI.. S-000622937 ST-00065799
## TOE535942-BIS-v01_R04C01 LEVEL1//TOE535942-BI.. S-000633015 ST-00045177
##                                Gender LEVEL1.TOEID   S_STUDYID
##                             <character>  <character> <character>
## TOE909374-BIS-v01_R08C01           F    TOE909374         CRA
## TOE239310-BIS-v01_R01C01           F    TOE239310         CRA
## TOE665077-BIS-v01_R07C01           F    TOE665077         CRA
## TOE971486-BIS-v01_R03C01           M    TOE971486         CRA
## TOE204245-BIS-v01_R07C01           F    TOE204245         CRA
## TOE535942-BIS-v01_R04C01           M    TOE535942         CRA
##                                      filenames      xMed      yMed
##                                    <character> <numeric> <numeric>
## TOE909374-BIS-v01_R08C01 LEVEL1//TOE909374-BI..   12.7689   8.73861
## TOE239310-BIS-v01_R01C01 LEVEL1//TOE239310-BI..   12.6702   8.72893
## TOE665077-BIS-v01_R07C01 LEVEL1//TOE665077-BI..   12.4876   8.73639
## TOE971486-BIS-v01_R03C01 LEVEL1//TOE971486-BI..   11.9102  12.59374
## TOE204245-BIS-v01_R07C01 LEVEL1//TOE204245-BI..   12.7321   8.74032
## TOE535942-BIS-v01_R04C01 LEVEL1//TOE535942-BI..   11.9055  12.55428
##                             predictedSex
##                              <character>
## TOE909374-BIS-v01_R08C01           F
## TOE239310-BIS-v01_R01C01           F
## TOE665077-BIS-v01_R07C01           F
## TOE971486-BIS-v01_R03C01           M
## TOE204245-BIS-v01_R07C01           F
## TOE535942-BIS-v01_R04C01           M
```

```
# checking the clustering
pdf(file = file.path(plots.dir, "MDS_sex_500pos.pdf"), width = 6, height = 6)
mdsPlot(as.matrix(norm.betas.rcp), numPositions=500,
        main=sprintf("Beta MDS - Sex\n%d most variable positions", 500),
        pch=19,  legendNCol=5, sampGroups=pData.cra$Gender,
        legendPos="topleft", pal=c(brewer.pal(8, "Dark2"),
                                   brewer.pal(12, "Paired")))
dev.off()
```

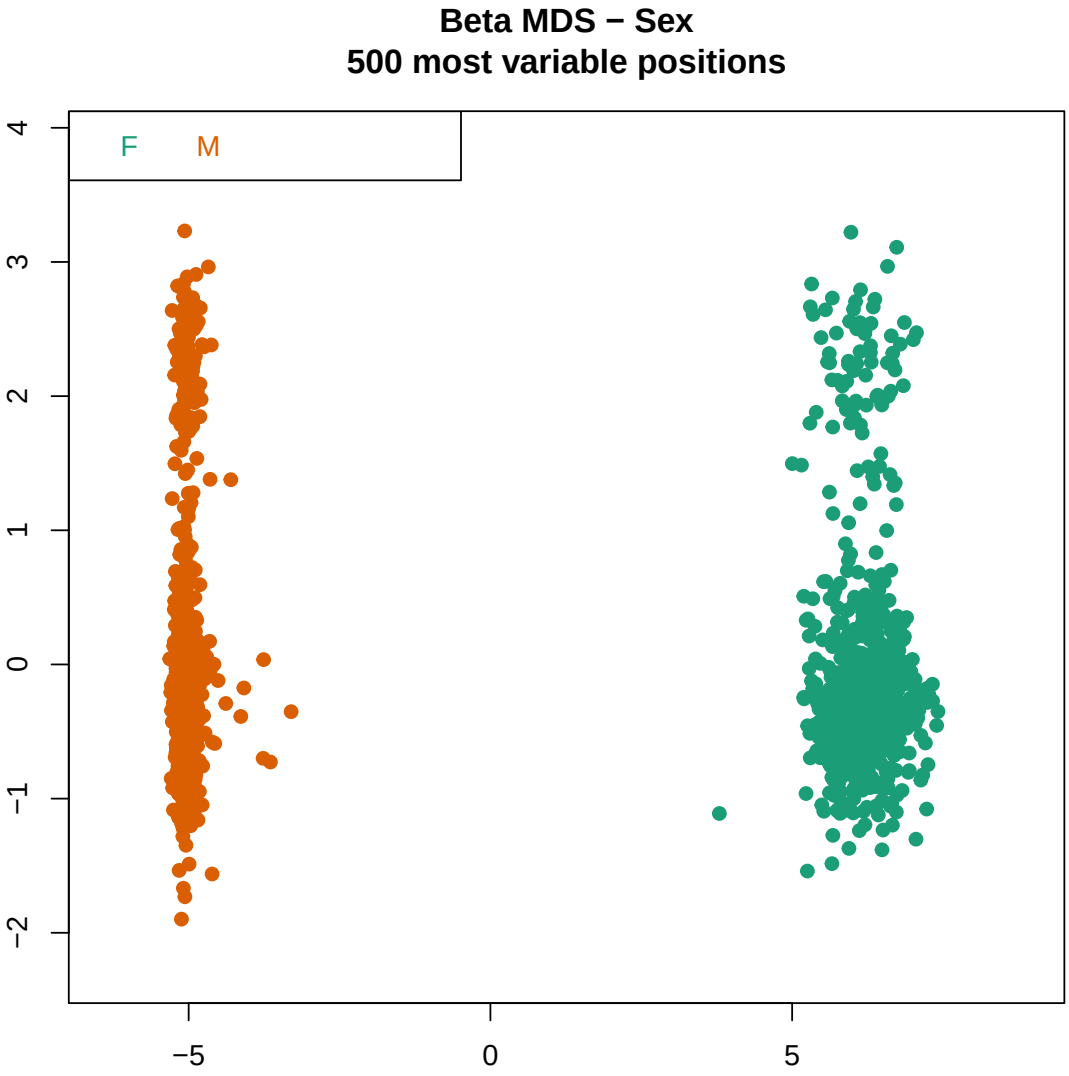
```
## pdf
##   2
```

```
# select 1000 rows at random and then plot instead of 500 most variable
betas.rand <- norm.betas.rcp[sample(nrow(norm.betas.rcp), 1000), ]
```

```
pdf(file = file.path(plots.dir, "MDS_sex_1000pos_rand.pdf"), width = 6, height = 6)
mdsPlot(as.matrix(betas.rand), numPositions=1000,
        main=sprintf("Beta MDS - Sex\n%d random positions", 1000),
        pch=19,  legendNCol=5, sampGroups=pData.cra$Gender,
        legendPos="topleft", pal=c(brewer.pal(8, "Dark2"),
                                   brewer.pal(12, "Paired")))
dev.off()
```

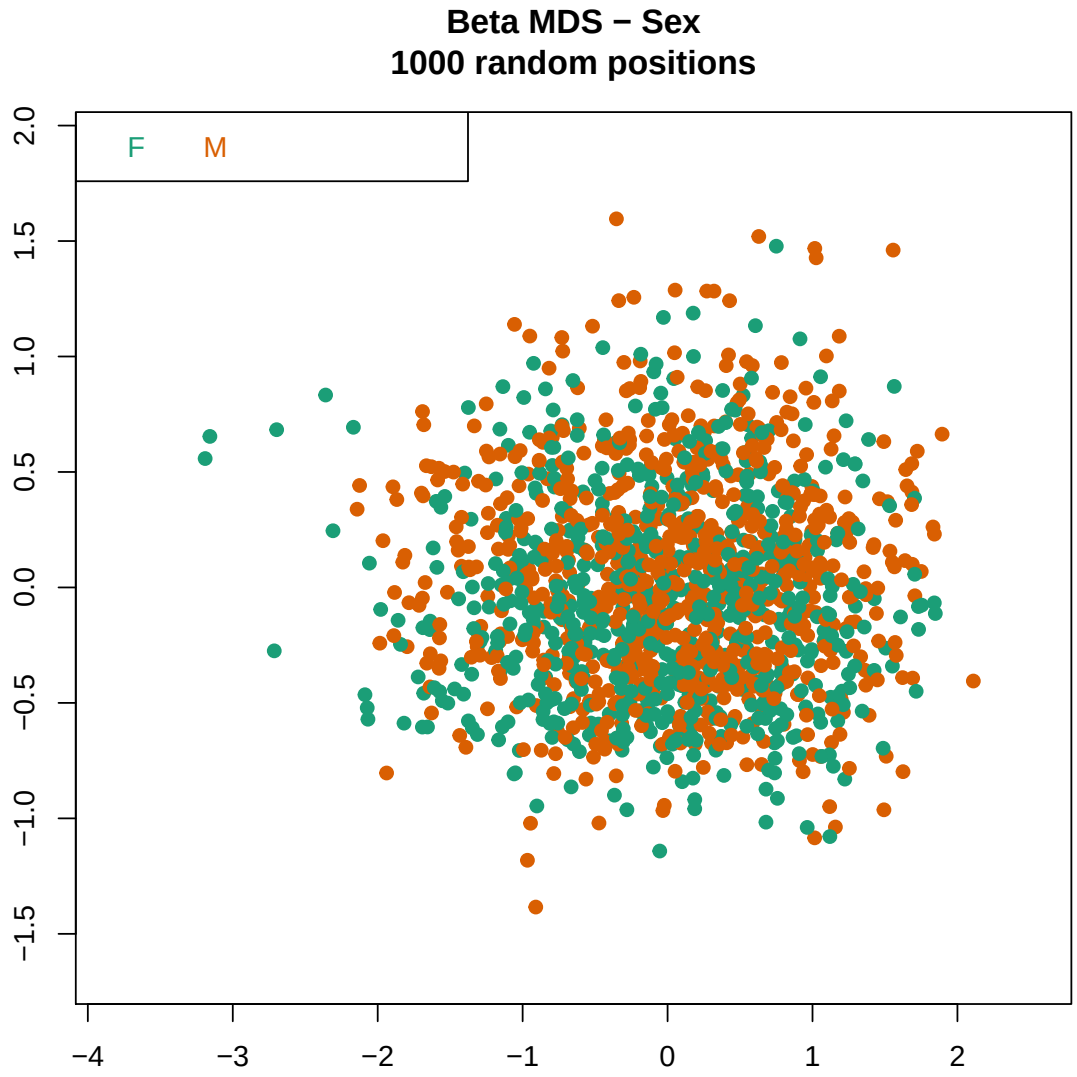
```
## pdf
##   2
```

```
mdsPlot(as.matrix(norm.betas.rcp), numPositions=500,
        main=sprintf("Beta MDS - Sex\n%d most variable positions", 500),
        pch=19,  legendNCol=5, sampGroups=pData.cra$Gender,
        legendPos="topleft", pal=c(brewer.pal(8, "Dark2"),
                                   brewer.pal(12, "Paired")))
```




```
mdsPlot(as.matrix(betas.rand), numPositions=1000,
        main=sprintf("Beta MDS - Sex\n%d random positions", 1000),
        pch=19,  legendNCol=5, sampGroups=pData.cra$Gender,
```

```
        legendPos="topleft", pal=c(brewer.pal(8, "Dark2"),
                                   brewer.pal(12, "Paired")))
```



4 PCAs on autosomes

```
# remove non asthmatic samples
dim(pDat.cra); dim(cra.pheno)
```

```
## Error in eval(expr, envir, enclos):  object 'pDat.cra' not found
```

```
## [1] 1165  514
```

```
table(cra.pheno$Dr_Dx_Asthma)
```

```
##
##    1    2
##   31 1134
```

```
cra.pheno.ast <- cra.pheno[cra.pheno$Dr_Dx_Asthma==2,]
```

```
pData.cra$toe_ids <- rownames(pData.cra)
pData.pheno.cra <- merge(pData.cra, cra.pheno.ast, by="S_SUBJECTID", sort=F) # 788 samples
dim(pData.pheno.cra)
```

```
## [1] 788 524
```

```
norm.betas.rcp.prob <- norm.betas.rcp[,colnames(norm.betas.rcp) %in% pData.pheno.cra$toe_ids]
dim(norm.betas.rcp.prob)
```

```
## [1] 865859    788
```

```
#####################################################################
# Normalized betas from probands and all chromosomes
# remove missing probes if needed before any downstream analysis
#####################################################################

# number of probes remaining after removing missing/failed probes
dim(na.omit(norm.betas.rcp.prob))
```

```
## [1] 808449    788
```

```
save(norm.betas.rcp.prob,
     file=file.path(results.dir,paste0("norm.betas.cra_hg19_clean_allchr_probands_",
                                       timeStamp,".RData")))

rm(norm.betas.rcp.prob) # cleaning memory

# pca on autosomes
# autosomal.sites <- meffil.get.autosomal.sites("epic")
# length(autosomal.sites)
# autosomal.sites <- intersect(autosomal.sites, rownames(norm.betas.rcp))
norm.betas.rcp.auto <- norm.betas.rcp[rownames(norm.betas.rcp) %in% auto.probes,]
dim(norm.betas.rcp.auto) # 846232 probes, 1204 samples
```

```
## [1] 846232   1204
```

```
norm.betas.rcp.auto <- na.omit(norm.betas.rcp.auto) # 787354

norm.betas.rcp.auto.prob <- norm.betas.rcp.auto[,colnames(norm.betas.rcp.auto)
                                                %in% pData.pheno.cra$toe_ids]
dim(norm.betas.rcp.auto.prob)
```

```
## [1] 790798    788
```

```
rm(norm.betas.rcp)
rm(norm.betas.rcp.auto)

#####################################################################
# this function will automatically save pcas in results directory
# Top  10  principal components can explain  43.11407 % of data variation
#####################################################################
pca.betas(norm.betas.rcp.auto.prob, n=10)
```

```
## Analysis is running, please wait...!
## Top  10  principal components can explain  43.11407 % of data
##     variation
```

```
#####################################################################
# Normalized betas from probands and autosomes (no sex chr)
# removed missing probes removed as above for EWAS
#####################################################################
save(norm.betas.rcp.auto.prob, file=file.path(results.dir,
                              paste0("norm.betas.cra_hg19_clean_NOsexchr_probands_",
                                     timeStamp,".RData")))
```

5 Session information

[1] "2021-05-12" [1] "2021-05-12 14:28:03 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, Biobase 2.50.0, 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, DNAcopy 1.64.0, doParallel 1.0.16, dplyr 1.0.3, e1071 1.7-6, ENmix 1.26.10, fastICA 1.2-2, FDb.InfiniumMethylation.hg19 2.2.0, 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, AnnotationHub 2.22.1, askpass 1.1, assertthat 0.2.1, backports 1.2.1, base64 2.0, beanplot 1.2, BiocFileCache 1.14.0, 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, dbplyr 2.1.0, 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, ExperimentHub 1.16.1, 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