```
################################################################
## This is an R script to accompany Weigelt et al
## titled "Metaplastic breast carcinomas display genomic and
transcriptomic heterogeneity"
##
## Author: Charlotte K Y Ng
## Date: 6th June 2014
## Files required to run this script are available at
## https://dl.dropboxusercontent.com/u/15115364/
Weigelt_et_al_Metplastic_subtyping.zip
#################################################################

library(beadarray)
library(limma)
library(snow)

source("pickProbesForNormalization.R")
source("createTargetDistribution.R")
source("normalizeArrayToTarget.R")
source("classifyFunctions.R")
source("classifyFunctions_claudinlow.R")

## files provided ##
UNNORM_FILE = "20131126_discoveryset_metaplastic_unnorm_39424.RData"
SNP6_FILE = "20131126_metaplastic_SNP6_LRR_pergene.RData"
PAM50_CENTROIDS_FILE = "PAM50_ILMN_centroids_METABRICcode.txt"
CLAUDIN_LOW_CENTROIDS_FILE =
"ClaudinLow.centroids_ILMN_METABRIC.txt"
ICLUSTER_CNPROBES_FILE= "CNprobes_CurtisClassifier.csv"
ICLUSTER_EXPPROBES_FILE= "EXPprobes_CurtisClassifier.csv"
ICLUSTER_CENTROIDS_FILE = "table_S42_CurtisCentroids.txt"

## files to be generated ##
UNNORM_TNBC_FILE =
"20131126_discoveryset_TNBC_metaplastic_unnorm_39424.RData"
TARGET_DIST_FILE =
"20131126_targetDist_discoveryset_metaplastic_39424.RData"
TARGET_DIST_TNBC_FILE =
"20131126_targetDist_discoveryset_TNBC_metaplastic_39424.RData"
NORM_FILE =
"20131126_discoveryset_metaplastic_39424_normalised.RData"
NORM_TNBC_FILE =
"20131126_discoveryset_TNBC_metaplastic_39424_normalised.RData"
PAM50_CALLS_FILE =
"20131126_discoveryset_metaplastic_39424_normalised_pam50.RData"
CLAUDIN_LOW_CALLS_FILE =
"20131126_discoveryset_metaplastic_39424_normalised_claudinlow.RData
"
INTRINSIC_CALLS_FILE =
"20131126_discoveryset_metaplastic_39424_normalised_intrinsicsubtype
.RData"
ICLUSTER_CALLS_FILE =
"20131126_discoveryset_metaplastic_39424_normalised_icluster.RData"
```

```
## temp directories ##
TEMP_DIR = "discoveryset_metaplastic_39424"
TEMP_TNBC_DIR = "discoveryset_TNBC_metaplastic_39424"

################# NORMALIZATION ####################
## Normalization is performed according to Curtis et al.
## The metaplasic cases are normalized with the discovery
## cohort of METABRIC
####################################################

load(UNNORM_FILE)

probes_for_target_dist <- pickProbesForNormalization()

targetDist <- createTargetDistribution(exprs(mergedeset), erSplit=T,
probeList=probes_for_target_dist, outputName = TARGET_DIST_FILE,
ER=pData(mergedeset)$ER_IHC_status)

outdir <- TEMP_DIR
if (!file.exists(outdir)) { dir.create(outdir) }

cl <- makeCluster(8)
parLapply(cl, sampleNames(mergedeset), normalizeArrayToTarget,
outdir)
stopCluster(cl)

rda <- paste(outdir, "/", sampleNames(mergedeset),
"_Normalized.rda", sep="")

load(rda[1])
BSData <- bsd
for (i in rda[-1]) {
        print (i)
        load(i)
        BSData <- do.call(combine, list(BSData, bsd))
}

save(BSData, file=NORM_FILE)

################# PAM50 subtyping ####################
## PAM50 subtyping is performed according to Curtis et al.
####################################################

load(NORM_FILE)

nTrials = 100
callMat = matrix(nrow = ncol(exprs(BSData)), ncol=nTrials)

for(i in 1:nTrials){
        print(i)
        callMat[,i] = calls.BSData.Scale.erWeight <-
classifySubtype(exprs(BSData), centroidFiles=PAM50_CENTROIDS_FILE,
                 medianCentre=TRUE, zScore = FALSE, iqrFilter = 0,
ER = pData(BSData)$ER_IHC_status,writeOutput=FALSE)[[1]][[1]]
}

rownames(callMat) = colnames(exprs(BSData))
calls.BSData = vector(length = length(rownames(callMat)))
names(calls.BSData) = rownames(callMat)

calls.BSData[rownames(callMat)]  = apply(callMat, 1, function(x)
names(which.max(table(x))))

save(calls.BSData, callMat, file=PAM50_CALLS_FILE)

### check agreement with original classification

table(as.data.frame(cbind(pData(BSData)$Pam50[grep("MB",
sampleNames(BSData))], calls.BSData[grep("MB",
sampleNames(BSData))])))

metabric_pheno <- cbind(pData(BSData)[grep("MB",
sampleNames(BSData)),], calls.BSData[grep("MB",
sampleNames(BSData))])

colnames(metabric_pheno)[ncol(metabric_pheno)] <- "Pam50"

################# Claudin-low subtyping ####################
## Claudin-low subtyping is performed by modifying the functions
## used for PAM50 subtyping. Specifically, instead of correlation,
## distance is calculated by Euclidean distance, as described in
## the UNC guide.
#############################################################

load(NORM_FILE)

nTrials = 100
callMat = matrix(nrow = ncol(exprs(BSData)), ncol=nTrials)

for(i in 1:nTrials){
        print(i)
        callMat[,i] = calls.BSData.Scale.erWeight <-
classifySubtype_claudinlow(exprs(BSData),
centroidFiles=CLAUDIN_LOW_CENTROIDS_FILE,
                medianCentre=TRUE, zScore = FALSE, iqrFilter = 0,
ER = pData(BSData)$ER_IHC_status,
percentagePos=0.58,writeOutput=FALSE)[[1]][[1]]

}

rownames(callMat) = colnames(exprs(BSData))
calls.BSData = vector(length = length(rownames(callMat)))
names(calls.BSData) = rownames(callMat)

calls.BSData[rownames(callMat)]  = apply(callMat, 1, function(x)
names(which.max(table(x))))
```

```
save(calls.BSData, callMat, file=CLAUDIN_LOW_CALLS_FILE)

############# Intrinsic subtyping ###################
## Intrinsic subtping is performed according to the
## UNC guide by combining PAM50 and Claudin-low subtyping.
#####################################################

load(PAM50_CALLS_FILE)
intrinsic_subtype <- calls.BSData
load(CLAUDIN_LOW_CALLS_FILE)
intrinsic_subtype[which(calls.BSData=="ClaudinLow")] <- "ClaudinLow"

save(intrinsic_subtype, file=INTRINSIC_CALLS_FILE)

################# TNBC subtyping ###############
## Normalization is done using only TNBC cases,
## as recommended by the TNBC subtyping website
#################################################

load(UNNORM_FILE)

mergedeset <- mergedeset[,c(which(pData(mergedeset)$ER.Expr=="-" &
pData(mergedeset)$PR.Expr=="-" & pData(mergedeset)$Her2.Expr=="-"),
grep("META", sampleNames(mergedeset)))]

save(mergedeset, file=UNNORM_TNBC_FILE)

load(UNNORM_TNBC_FILE)
probes_for_target_dist <- pickProbesForNormalization()

targetDist <- createTargetDistribution(exprs(mergedeset), erSplit=F,
probeList=probes_for_target_dist, outputName =
TARGET_DIST_TNBC_FILE)

outdir <- TEMP_TNBC_DIR
if (!file.exists(outdir)) { dir.create(outdir) }

cl <- makeCluster(16)
parLapply(cl, sampleNames(mergedeset), normalizeArrayToTarget,
outdir)
stopCluster(cl)

rda <- paste(outdir, "/", sampleNames(mergedeset),
"_Normalized.rda", sep="")

load(rda[1])
BSData <- bsd

for (i in rda[-1]) {
        print (i)
        load(i)
        BSData <- do.call(combine, list(BSData, bsd))
}
```

```
## summarise to per-gene expresison, then
## remove cases with ESR1 expression within the top 25% and
## repeat normalization without them

tnbc <- exprs(BSData)
rownames(tnbc) <- fData(BSData)$ILMN_Gene

tnbc_uniq <- matrix(nrow=0, ncol=ncol(tnbc))
colnames(tnbc_uniq) <- colnames(tnbc)
for (u in unique(rownames(tnbc))) {
        subset <- tnbc[which(rownames(tnbc)==u),]
        if (is.null(dim(subset))) {
                tnbc_uniq <- rbind(tnbc_uniq, subset)
        }else {
                tnbc_uniq <- rbind(tnbc_uniq,
subset[which.max(unlist(apply(subset, 1, var))),])
        }
        rownames(tnbc_uniq)[nrow(tnbc_uniq)] <- u
}

discov_TNBC_ERneg_meta <-
names(which(apply(cbind(tnbc_uniq[which(rownames(tnbc_uniq)=="ESR1")
,], unlist(apply(tnbc_uniq[-which(rownames(tnbc_uniq)=="ESR1"),], 2,
quantile, 0.75))), 1, function(x){x[1]>x[2]})==F))

load(UNNORM_TNBC_FILE)

mergedeset <- mergedeset[,which(sampleNames(mergedeset) %in%
discov_TNBC_ERneg_meta)]

targetDist <- createTargetDistribution(exprs(mergedeset), erSplit=F,
probeList=probes_for_target_dist, outputName =
TARGET_DIST_TNBC_FILE)

outdir <- TEMP_TNBC_DIR
if (!file.exists(outdir)) { dir.create(outdir) }

cl <- makeCluster(16)
parLapply(cl, sampleNames(mergedeset), normalizeArrayToTarget,
outdir)
stopCluster(cl)

rda <- paste(outdir, "/", sampleNames(mergedeset),
"_Normalized.rda", sep="")

load(rda[1])
BSData <- bsd

for (i in rda[-1]) {
        print (i)
        load(i)
        BSData <- do.call(combine, list(BSData, bsd))
}
```

```
tnbc <- exprs(BSData)
rownames(tnbc) <- fData(BSData)$ILMN_Gene

tnbc_uniq <- matrix(nrow=0, ncol=ncol(tnbc))
colnames(tnbc_uniq) <- colnames(tnbc)
for (u in unique(rownames(tnbc))) {
        subset <- tnbc[which(rownames(tnbc)==u),]
        if (is.null(dim(subset))) {
                tnbc_uniq <- rbind(tnbc_uniq, subset)
        }else {
                tnbc_uniq <- rbind(tnbc_uniq,
subset[which.max(unlist(apply(subset, 1, var))),])
        }
        rownames(tnbc_uniq)[nrow(tnbc_uniq)] <- u
}

# this file is exported as text file and uploaded
# for TNBC subtyping
write.table(tnbc_uniq, file="TNBCsubtyping.txt", sep=",",
col.names=NA, quote=F)

save(BSData, file=NORM_TNBC_FILE)

################## Integrated clustering ##########
## Integrative clustering is performed as described
## by Curtis et al.
###################################################

load(SNP6_FILE)
gnames.cn=rownames(metaplastic.cn)
metaplastic.cn=apply(metaplastic.cn,2,as.numeric)

load(NORM_FILE)
metaplastic.exp <- exprs(BSData)
metaplastic.exp <- metaplastic.exp[,grep("META",
colnames(metaplastic.exp))]
gnames.exp=rownames(metaplastic.exp)
metaplastic.exp=apply(metaplastic.exp,2,as.numeric)
metaplastic.exp=scale(metaplastic.exp,center=T,scale=T)
metaplastic.exp=scale(t(metaplastic.exp),center=T,scale=T)
metaplastic.exp=t(metaplastic.exp)

cnprobes=read.csv(file=ICLUSTER_CNPROBES_FILE,header=T)

idx=match(as.character(cnprobes[,2]),gnames.cn)
which.na=which(is.na(idx))
idx=idx[-which.na]
cn=metaplastic.cn[idx,]
rownames(cn)=paste("CN",cnprobes[-which.na,1],sep="-")

expprobes=read.csv(file=ICLUSTER_EXPPROBES_FILE,header=T)

idx=match(as.character(expprobes[,1]),gnames.exp)
exp=metaplastic.exp[idx,]
```

```
rownames(exp)=paste("Exp",expprobes[,1],sep="-")

centroids=read.delim(file=ICLUSTER_CENTROIDS_FILE,header=T,sep='\t',
as.is=T)
centroids.id=centroids[,1]
centroids=apply(centroids[,-1],2,as.numeric)

newdata=rbind(cn,exp[,colnames(cn)])

idx=match(rownames(newdata),centroids.id)
m=centroids[idx,]

corr=cor(newdata,m)
class=apply(corr,1,which.max)
max.corr=apply(corr,1,max)

save(corr, class, file=ICLUSTER_CALLS_FILE)
```