NetBAS for tumor genes

HBG

11/4/2018

This script perform GO enrichment using NetBAS

for 51 PDAC cell high BF genes

```r
panc.file <- read.csv("../rnf43.csv",header=TRUE,stringsAsFactors=F)
panc.gene <- panc.file$gene
panc.panc <- panc.file$panc.mean
panc.np <- panc.file$nonpanc.mean
quant <- quantile(panc.panc, probs = seq(0,1,1/20))
quant.np <- quantile(panc.np, probs=seq(0,1,1/20))

# top 10% BF genes (1718) in tumor cells
panc.top10 <- panc.gene[which(panc.panc > quant[19])]
length(panc.top10)
```

```
## [1] 1718
```

```r
# genes (8458) of 0% to 50% BF factors in normal cells
np.bottom50 <- panc.gene[which(panc.np < quant.np[11])]
length(np.bottom50)
```

```
## [1] 8458
```

```r
# the overlap genes between top 10% tumor and (0-50%) normal cells
gene.list <- panc.gene[which(panc.top10 %in% np.bottom50)]
length(gene.list)
```

```
## [1] 51
```

```r
gene.list
```

```
##  [1] "FZD5"     "WLS"      "HNF1A"    "RBM15"    "PPCS"     "PORCN"
##  [7] "TPK1"     "ADRBK1"   "NANS"     "PPARG"    "SLC2A1"   "STK40"
## [13] "MPI"      "ALG3"     "WNT3"     "TRIB1"    "STX4"     "SLC25A1"
## [19] "WDR26"    "VPS4B"    "MMACHC"   "PPP5C"    "KATNA1"   "ARHGEF39"
## [25] "GALE"     "MOGS"     "PCBD1"    "PGM3"     "PRR12"    "AP2M1"
## [31] "PTCH2"    "RIF1"     "TFB1M"    "DHX29"    "MTR"      "NPC1"
## [37] "PPP2R4"   "ALG9"     "PPCDC"    "GOLGA7"   "KLK3"     "EDC3"
## [43] "LAMA3"    "ACO2"     "MTRR"     "RANBP17"  "NDE1"     "FAM221B"
## [49] "FDFT1"    "DPF2"     "ELMSAN1"
```

```r
## read the original network
network <- read.csv("../Data/human.pin.csv", header=T, stringsAsFactors=F)
geneA <- network$geneA
geneB <- network$geneB

GOcategory.file <- read.csv("../Data/human.mf.term.csv",header=TRUE, stringsAsFactors=F)
mf.go.cat <- GOcategory.file$GO.term
mf.dim <- length(mf.go.cat)

GOterm.file <- read.csv("../Data/human.mf.gene.term.csv", header=T, stringsAsFactors=F)
mf.GO.gene <- GOterm.file$gene  #it should be changed to System for yeast pin
mf.GO.term <- GOterm.file$GO.term

vec <- numeric(length=mf.dim)

for (i in 1:length(gene.list)) {
    orf <- as.character(gene.list[i])
    intA <- geneB[which(geneA %in% orf)]
    for (j in 1:length(intA)) {
        mfA <- mf.GO.term[which(mf.GO.gene %in% intA[j])]
        for (k in 1:length(mfA)) {
           na <- which(mf.go.cat %in% mfA[k])
           vec[na] <- vec[na] + 1
        }
    }

    intB <- geneA[which(geneB %in% orf)]
    for (s in 1:length(intB)) {
        mfB <- mf.GO.term[which(mf.GO.gene %in% intB[s])]
        for (t in 1:length(mfB)) {
           nb <- which(mf.go.cat %in% mfB[t])
           vec[nb] <- vec[nb] + 1
        }
    }
}

write.table(vec, file="hs.rnf43.list.mf.txt", col.names=F, row.names=F, quote=F)
```

```
# Now the ms02star permutations
for (p in 1:100) {
permutation.file <- paste("../ms02star/human/", "ms02.", p, ".csv", sep="")
permutation <- read.csv(permutation.file, header=T, stringsAsFactors = F)
geneA <- permutation$id1
geneB <- permutation$id2

vecp <-  numeric(length = mf.dim)
for (i in 1:length(gene.list)) {
    orf <- as.character(gene.list[i])
    intA <- geneB[which(geneA %in% orf)]
    for (j in 1:length(intA)) {
        mfA <- mf.GO.term[which(mf.GO.gene %in% intA[j])]
        for (k in 1:length(mfA)) {
           na <- which(mf.go.cat %in% mfA[k])
           vecp[na] <- vecp[na] + 1
        }
    }

    intB <- geneA[which(geneB %in% orf)]
    for (s in 1:length(intB)) {
        mfB <- mf.GO.term[which(mf.GO.gene %in% intB[s])]
        for (t in 1:length(mfB)) {
```

```r
            nb <- which(mf.go.cat %in% mfB[t])
            vecp[nb] <- vecp[nb] + 1
        }
    }
}

output <- paste("ms02.human", "/", "rnf43.list", "/", "ms02.", p, ".mf.matrix.csv", sep="")

write.table(vecp, file = output, col.names=F, row.names=F, quote=F)
}
```

```r
library("microbenchmark")
library("matrixStats")

conn.dim <- 1

hspin <- matrix(as.numeric(unlist(read.table("hs.rnf43.list.mf.txt", header=F, sep=","))), nrow=mf.dim,
obs <- c(hspin)

perm <- c()
for (i in 1:100) {
    name <- paste("ms02.human", "/", "rnf43.list", "/", "ms02.", i, ".mf.matrix.csv", sep="")
    mat <- matrix(as.numeric(unlist(read.table(name, header=F, sep=","))), nrow=mf.dim, ncol=conn.dim)
    perm <- rbind(perm, c(mat))
}

mean <- colMeans(perm)
std <- colSds(perm)

zscore <- round((obs - mean)/std, 3)

z <- matrix(zscore, nrow=mf.dim, ncol=conn.dim)

write.table(z, file="hs.rnf43.list.mf.z.csv", sep=",", row.names=F, col.names=F, quote=F)
```

```r
library('gplots')
```

```
##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

```r
library('GO.db')
```

```
## Loading required package: AnnotationDbi

## Loading required package: stats4

## Loading required package: BiocGenerics

## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
```

```
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind,
##     colMeans, colnames, colSums, dirname, do.call, duplicated,
##     eval, evalq, Filter, Find, get, grep, grepl, intersect,
##     is.unsorted, lapply, lengths, Map, mapply, match, mget, order,
##     paste, pmax, pmax.int, pmin, pmin.int, Position, rank, rbind,
##     Reduce, rowMeans, rownames, rowSums, sapply, setdiff, sort,
##     table, tapply, union, unique, unsplit, which, which.max,
##     which.min
```

```
## Loading required package: Biobase
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
##
## Attaching package: 'Biobase'
```

```
## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

```
## Loading required package: IRanges
```

```
## Loading required package: S4Vectors
```

```
##
## Attaching package: 'S4Vectors'
```

```
## The following object is masked from 'package:gplots':
##
##     space
```

```
## The following object is masked from 'package:base':
##
##     expand.grid
```

```
##
```

```
order <- order(z)
mf.go.cat <- mf.go.cat[order]
z <- z[order]

z <- t(z)
colnames(z) <- mf.go.cat

enriched.list <- mf.go.cat[which(z >= 3)]
```

```
enriched <- c("GO.ID", "GO.Term", "Z-score")
for (i in 1:length(enriched.list)) {
  term <- Term(GOID(as.character(enriched.list[i])))
  enriched <- rbind(enriched, c(as.character(enriched.list[i]),term,
                                z[which(mf.go.cat %in% enriched.list[i])]))
}

print(enriched)
```

```
##
## enriched "GO.ID"
##          "GO:0005096"
##          "GO:0031210"
##          "GO:0042626"
##          "GO:0019899"
##          "GO:0046966"
##          "GO:0031072"
##          "GO:0001968"
##          "GO:0005102"
##          "GO:0005267"
##          "GO:0019904"
##          "GO:0008237"
##          "GO:0004879"
##          "GO:0008013"
##          "GO:0003714"
##          "GO:0004222"
##          "GO:0005484"
##          "GO:0016887"
##          "GO:0001102"
##          "GO:0008022"
##          "GO:0003705"
##          "GO:0003777"
##          "GO:0050681"
##          "GO:0005509"
##          "GO:0030170"
##          "GO:0042623"
##          "GO:0001618"
##          "GO:0001786"
##          "GO:0005125"
##          "GO:0004857"
##          "GO:0017075"
##          "GO:0030276"
##          "GO:0030374"
##          "GO:0000149"
##          "GO:0004867"
##          "GO:0051787"
##          "GO:0003707"
##          "GO:0044183"
##          "GO:0003713"
##          "GO:0015301"
##          "GO:0004252"
##          "GO:0016491"
##          "GO:0008017"
##          "GO:0001540"
##          "GO:0035257"
##          "GO:0016922"
##          "GO:0005178"
##          "GO:0042803"
##          "GO:0019905"
##          "GO:0005496"
##          "GO:0001972"
##          "GO:0017147"
##          "GO:0048018"
##          "GO:0005109"
##          "GO:0015020"
##          "GO:0008194"
##          GO:0005096
## enriched "GO.Term"
##          "GTPase activator activity"
##          "phosphatidylcholine binding"
##          "ATPase activity, coupled to transmembrane movement of substances"
##          "enzyme binding"
##          "thyroid hormone receptor binding"
##          "heat shock protein binding"
##          "fibronectin binding"
##          "signaling receptor binding"
##          "potassium channel activity"
##          "protein domain specific binding"
##          "metallopeptidase activity"
##          "nuclear receptor activity"
##          "beta-catenin binding"
##          "transcription corepressor activity"
##          "metalloendopeptidase activity"
##          "SNAP receptor activity"
##          "ATPase activity"
##          "RNA polymerase II activating transcription factor binding"
##          "protein C-terminus binding"
##          "transcription factor activity, RNA polymerase II distal enhancer sequence-specific binding
##          "microtubule motor activity"
##          "androgen receptor binding"
##          "calcium ion binding"
##          "pyridoxal phosphate binding"
##          "ATPase activity, coupled"
##          "virus receptor activity"
##          "phosphatidylserine binding"
##          "cytokine activity"
##          "enzyme inhibitor activity"
##          "syntaxin-1 binding"
##          "clathrin binding"
##          "nuclear receptor transcription coactivator activity"
##          "SNARE binding"
##          "serine-type endopeptidase inhibitor activity"
##          "misfolded protein binding"
##          "steroid hormone receptor activity"
##          "protein binding involved in protein folding"
##          "transcription coactivator activity"
##          "anion:anion antiporter activity"
##          "serine-type endopeptidase activity"
##          "oxidoreductase activity"
##          "microtubule binding"
##          "amyloid-beta binding"
##          "nuclear hormone receptor binding"
##          "nuclear receptor binding"
##          "integrin binding"
##          "protein homodimerization activity"
##          "syntaxin binding"
##          "steroid binding"
##          "retinoic acid binding"
##          "Wnt-protein binding"
##          "receptor ligand activity"
##          "frizzled binding"
##          "glucuronosyltransferase activity"
##          "UDP-glycosyltransferase activity"
##
## enriched "Z-score"
##          "3.017"
##          "3.118"
##          "3.13"
##          "3.243"
##          "3.244"
##          "3.278"
##          "3.286"
##          "3.338"
##          "3.385"
##          "3.398"
##          "3.41"
##          "3.426"
##          "3.441"
##          "3.448"
##          "3.472"
##          "3.478"
##          "3.5"
##          "3.569"
##          "3.58"
##          "3.606"
##          "3.629"
##          "3.686"
##          "3.7"
##          "3.838"
##          "3.913"
##          "3.927"
##          "3.932"
##          "4.042"
##          "4.117"
##          "4.209"
##          "4.284"
##          "4.32"
##          "4.41"
##          "4.436"
##          "4.459"
##          "4.522"
##          "4.619"
##            "4.624"
##            "4.723"
##            "4.764"
##            "4.799"
##            "4.823"
##            "4.909"
##            "5.407"
##            "5.566"
##            "5.592"
##            "5.677"
##            "6.061"
##            "6.487"
##            "10.525"
##            "12.912"
##            "17.239"
##            "21.91"
##            "22.16"
##            "24.135"
```

```
write.table(enriched, file="hs.rnf43.list.mf.enriched.csv", row.names=F, col.names=F, quote=F, sep="\t"

###No suppressed terms have been found
#sup.list <- mf.go.cat[which(z <= -3)]
#sup <- c("GO.ID", "GO.Term", "Z-score")
#for (i in 1:length(sup.list)) {
#  term <- Term(GOID(as.character(sup.list[i])))
#  sup <- rbind(sup, c(as.character(sup.list[i]),term,
#                z[which(mf.go.cat %in% sup.list[i])]))
#}

#print(sup)

#write.table(sup, file="human.pdcd1.mf.suppressed.csv", row.names=F, col.names=F, quote=F, sep="\t")
###

#Note that there may be "inf" Z-scores owing to lack of sampling (i.e., zero in standard deviations)
#We can also extract the GO-terms for the gene for comparison
```