Blood Module Preservation in ASD Cortical Tissue

Module Preservation ASD Blood Leukocytes, ASD Post-Mortem Cortical Tissue

Setup and read in data

```
library(easypackages)
libraries("WGCNA","here")
```

```
## ==========================================================================
## *
## *  Package WGCNA 1.63 loaded.
## *
## *    Important note: It appears that your system supports multi-threading,
## *    but it is not enabled within WGCNA in R.
## *    To allow multi-threading within WGCNA with all available cores, use
## *
## *          allowWGCNAThreads()
## *
## *    within R. Use disableWGCNAThreads() to disable threading if necessary.
## *    Alternatively, set the following environment variable on your system:
## *
## *          ALLOW_WGCNA_THREADS=<number_of_processors>
## *
## *    for example
## *
## *          ALLOW_WGCNA_THREADS=24
## *
## *    To set the environment variable in linux bash shell, type
## *
## *           export ALLOW_WGCNA_THREADS=24
## *
## *     before running R. Other operating systems or shells will
## *     have a similar command to achieve the same aim.
## *
## ==========================================================================
```

```
options(stringsAsFactors=FALSE)

# Read in ASD brain data
load(here("data","tidy","asd_brain_data.Rdata"))

# Read in ASD blood data
load(here("data","processed","exprDataAdj.Rdata"))

# Read in WGCNA ASD blood results
wgcna_res = read.csv(here("WGCNAresults","wgcna_results_summary.csv"))
blood_colors = wgcna_res$moduleColors
asd_blood_data = exprDataAdj
asd_blood_geneAnno = geneInfo
```

Find common genes amongst the two datasets

```
gene_mask = is.element(asd_blood_geneAnno$geneSymbol, asd_brain_geneAnno$hgnc_symbol)
genes2use = asd_blood_geneAnno$geneSymbol[gene_mask]
asd_blood_geneAnno_subset = asd_blood_geneAnno[gene_mask,]

asd_blood_data_subset = asd_blood_data[gene_mask,]
rownames(asd_blood_data_subset) = asd_blood_geneAnno_subset$geneSymbol
blood_colors_subset = blood_colors[gene_mask]

geneAnno2 = subset(asd_brain_geneAnno,
                   is.element(asd_brain_geneAnno$hgnc_symbol,genes2use))
asd_brain_data_subset = subset(asd_brain_data,
                               is.element(asd_brain_geneAnno$hgnc_symbol, genes2use))
rownames(asd_brain_data_subset) = geneAnno2$hgnc_symbol
asd_brain_data_subset = asd_brain_data_subset[rownames(asd_blood_data_subset),]
```

Run modulePreservation

```
setLabels = c("ASDBlood","ASDBrain")
multiExpr = list(ASDBlood = list(data = t(asd_blood_data_subset)),
                 ASDBrain = list(data = t(asd_brain_data_subset)))
multiColor = list(ASDBlood = blood_colors_subset)

# Calculate module preservation stats
corFnc2use = "bicor"
networkType = "signed"
nperm = 1000
rand_seed = 1
mp = modulePreservation(multiExpr,
                        multiColor,
                        networkType = networkType,
                        corFnc= corFnc2use,
                        maxGoldModuleSize = 1000,
                        referenceNetworks = 1,
                        nPermutations = nperm,
                        randomSeed = rand_seed,
                        quickCor = 0,
                        verbose = 0)
```

Show module preservation results

```
ref = 1
test = 2
statsObs = cbind(mp$quality$observed[[ref]][[test]][,-1],
                 mp$preservation$observed[[ref]][[test]][,-1])
statsZ = cbind(mp$quality$Z[[ref]][[test]][,-1],
               mp$preservation$Z[[ref]][[test]][,-1])

# compute p-values from log10 pvalues in mp
```

```
Zsummary.log10pvals = mp$preservation$log.p$ref.ASDBlood$inColumnsAlsoPresentIn.ASDBrain$log.psummary.pr
Zsummary.pvals = 10^Zsummary.log10pvals
Zsummary.fdr = p.adjust(Zsummary.pvals, method = "fdr")

sumTable = cbind(statsObs[, c("medianRank.pres", "medianRank.qual")],
             signif(statsZ[, c("Zsummary.pres", "Zsummary.qual")], 2),
             Zsummary.pvals, Zsummary.fdr)

ModCols = c("black","blue","brown","cyan","darkred","gold","green",
            "greenyellow","grey","grey60","lightcyan","lightgreen",
            "lightyellow","magenta","midnightblue","pink","purple",
            "red","royalblue","salmon","tan","turquoise","yellow")
ModNums = c("M7","M2","M3","M14","M21",NA,"M5","M11","M0","M16","M17",
            "M18","M19","M9","M15","M8","M10","M6","M20","M13","M12",
            "M1","M4")
modinfo = data.frame(moduleColors = ModCols, moduleLabels = ModNums)
sumTable = cbind(moduleLabels = modinfo$moduleLabels, sumTable)
sumTable
```

```
##              moduleLabels medianRank.pres medianRank.qual Zsummary.pres
## black                  M7             17            18.0         0.240
## blue                   M2              2            11.0         8.100
## brown                  M3             19            10.0        -0.940
## cyan                  M14              9            15.0         0.900
## darkred               M21             16             5.0        -0.190
## gold                 <NA>             14            22.0         5.500
## green                  M5              9            18.0         1.300
## greenyellow           M11              3             1.0         4.100
## grey                   M0             18            23.0        -0.054
## grey60                M16             20             7.0        -1.200
## lightcyan             M17             18            17.0        -0.380
## lightgreen            M18              9             4.0         0.810
## lightyellow           M19             14            12.0        -0.190
## magenta                M9              1            12.0         6.500
## midnightblue          M15             14             3.5        -0.160
## pink                   M8              4             6.0         5.400
## purple                M10             11            20.0         1.700
## red                    M6             13             9.5        -0.940
## royalblue             M20             14             2.0         0.340
## salmon                M13             15            18.0        -0.240
## tan                   M12              5             8.0         2.500
## turquoise              M1             12            13.5         1.500
## yellow                 M4             10            21.0         2.400
##              Zsummary.qual Zsummary.pvals Zsummary.fdr
## black                   25   1.588561e-01 2.810530e-01
## blue                    56   3.402800e-19 7.826441e-18
## brown                   80   6.023294e-01 6.297080e-01
## cyan                    23   1.225806e-01 2.563049e-01
## darkred                 48   4.955699e-01 6.297080e-01
## gold                     1   2.483746e-10 1.904205e-09
## green                   22   7.677112e-02 1.765736e-01
## greenyellow             84   4.107692e-07 1.889538e-06
## grey                   -14   1.767056e-01 2.903021e-01
## grey60                  41   8.646665e-01 8.646665e-01
```

```
## lightcyan               15   5.928797e-01 6.297080e-01
## lightgreen              55   1.413842e-01 2.709863e-01
## lightyellow             19   5.709222e-01 6.297080e-01
## magenta                 48   1.341995e-13 1.543294e-12
## midnightblue            62   5.573078e-01 6.297080e-01
## pink                    78   6.224280e-09 3.578961e-08
## purple                  19   2.293958e-02 5.862338e-02
## red                     53   4.692526e-01 6.297080e-01
## royalblue               58   3.449277e-01 5.288892e-01
## salmon                  21   5.660540e-01 6.297080e-01
## tan                     47   6.357133e-03 2.088772e-02
## turquoise               86   1.643211e-02 4.724230e-02
## yellow                  20   6.254077e-03 2.088772e-02
```

```r
# Plot results
modColors = rownames(mp$preservation$observed[[ref]][[test]])
moduleSizes = mp$preservation$Z[[ref]][[test]][,1]
plotMods = !(modColors %in% c("grey", "gold"));
text = modColors[plotMods]
plotData = cbind(mp$preservation$observed[[ref]][[test]][,2],
                 mp$preservation$Z[[ref]][[test]][,2])
mains = c("Preservation Median rank", "Preservation Zsummary")
par(mfrow = c(1,2))
par(mar = c(4.5,4.5,2.5,1))
for (p in 1:2)
{
  min = min(plotData[, p], na.rm = TRUE)
  max = max(plotData[, p], na.rm = TRUE)
  # Adjust ploting ranges appropriately
  if (p==2)
  {
    if (min > -max/10) min = -max/10
    # ylim = c(min - 0.1 * (max-min), max + 0.1 * (max-min))
    ylim = c(-2,12)
  } else
    ylim = c(max + 0.1 * (max-min), min - 0.1 * (max-min))
    # ylim = c(-2,12)
  plot(moduleSizes[plotMods],
       plotData[plotMods, p],
       col = 1,
       bg = modColors[plotMods],
       pch = 21,
       main = mains[p],
       cex = 2.4,
       ylab = mains[p],
       xlab = "Module size",
       log = "x",
       ylim = ylim,
       xlim = c(10, 2000),
       cex.lab = 1.2,
       cex.axis = 1.2,
       cex.main =1.4)
  # labelPoints(moduleSizes[plotMods],
  #             plotData[plotMods, p],
```

```
  #             text,
  #             cex = 1,
  #             offs = 0.08);

  # For Zsummary, add threshold lines
  if (p==2) {
    abline(h=0)
    abline(h=2, col = "blue", lty = 2)
    abline(h=10, col = "darkgreen", lty = 2)
  }
}
```