Tutorial for the WGCNA package for R
II. Consensus network analysis of liver expression data, female and male mice

3. Relating consensus modules to female set-specific modules

Peter Langfelder and Steve Horvath

February 13, 2016

Contents

0 Preliminaries: setting up the R session **1**

3 Relating consensus modules to female set-specific modules **1**

0 Preliminaries: setting up the R session

Here we assume that a new R session has just been started. We load the WGCNA package, set up basic parameters and load data saved in the parts 1 and 2 of the tutorial.

```
# Display the current working directory
getwd();
# If necessary, change the path below to the directory where the data files are stored.
# "." means current directory. On Windows use a forward slash / instead of the usual \.
workingDir = ".";
setwd(workingDir);
# Load the WGCNA package
library(WGCNA)
# The following setting is important, do not omit.
options(stringsAsFactors = FALSE);
# Load the data saved in the first part
lnames = load(file = "Consensus-dataInput.RData");
#The variable lnames contains the names of loaded variables.
lnames
# Load the results of network analysis, tutorial part 2.a
lnames = load(file = "Consensus-NetworkConstruction-auto.RData");
lnames
```

We have loaded the variables `multiExpr` and `Traits` containing the expression and trait data, respectively. Further, expression data dimensions are stored in `nGenes` and `nSamples`.

3 Relating consensus modules to female set-specific modules

In this section we assume the reader has worked through the one-step network construction and module detection in the female mouse liver data (Section 2.a of the female mouse liver analysis tutorial). We first load the female

data and rename them so that they do not conflict with the consensus data:

```
lnames = load("../Mouse-Female/FemaleLiver-02-networkConstruction-auto.RData")
lnames
# Rename variables to avoid conflicts
femaleLabels = moduleLabels;
femaleColors = moduleColors;
femaleTree = geneTree;
femaleMEs = orderMEs(MEs, greyName = "ME0");
```

Next we load the results of the consensus module identification:

```
lnames = load("Consensus-NetworkConstruction-auto.RData")
lnames
```

The consensus network analysis results are represented by the variables `consMEs`, `moduleLabels`, `moduleColors`, and `consTree`. We are now ready to relate the female modules to the consensus modules. We calculate the overlaps of each pair of female-consensus modules, and use the Fisher's exact test (also known as hypergeometric test) to assign a p-value to each of the pairwise overlaps.

```
# Isolate the module labels in the order they appear in ordered module eigengenes
femModuleLabels = substring(names(femaleMEs), 3)
consModuleLabels = substring(names(consMEs[[1]]$data), 3)
# Convert the numeric module labels to color labels
femModules = labels2colors(as.numeric(femModuleLabels))
consModules = labels2colors(as.numeric(consModuleLabels))
# Numbers of female and consensus modules
nFemMods = length(femModules)
nConsMods = length(consModules)
# Initialize tables of p-values and of the corresponding counts
pTable = matrix(0, nrow = nFemMods, ncol = nConsMods);
CountTbl = matrix(0, nrow = nFemMods, ncol = nConsMods);
# Execute all pairwaise comparisons
for (fmod in 1:nFemMods)
  for (cmod in 1:nConsMods)
  {
    femMembers = (femaleColors == femModules[fmod]);
    consMembers = (moduleColors == consModules[cmod]);
    pTable[fmod, cmod] = -log10(fisher.test(femMembers, consMembers, alternative = "greater")$p.value);
    CountTbl[fmod, cmod] = sum(femaleColors == femModules[fmod] & moduleColors ==
                    consModules[cmod])
  }
```

To display the p-value and count tables in an informative way, we create a color-coded table of the intersection counts. The colors will indicate the p-value significance:

```
# Truncate p values smaller than 10^{-50} to 10^{-50}
pTable[is.infinite(pTable)] = 1.3*max(pTable[is.finite(pTable)]);
pTable[pTable>50 ] = 50 ;
# Marginal counts (really module sizes)
femModTotals = apply(CountTbl, 1, sum)
consModTotals = apply(CountTbl, 2, sum)
# Actual plotting
sizeGrWindow(10,7 );
#pdf(file = "Plots/ConsensusVsFemaleModules.pdf", wi = 10, he = 7);
par(mfrow=c(1,1));
par(cex = 1.0);
par(mar=c(8, 10.4, 2.7, 1)+0.3);
# Use function labeledHeatmap to produce the color-coded table with all the trimmings
```

```
labeledHeatmap(Matrix = pTable,
               xLabels = paste(" ", consModules),
               yLabels = paste(" ", femModules),
               colorLabels = TRUE,
               xSymbols = paste("Cons ", consModules, ": ", consModTotals, sep=""),
               ySymbols = paste("Fem ", femModules, ": ", femModTotals, sep=""),
               textMatrix = CountTbl,
               colors = greenWhiteRed(100)[50:100],
               main = "Correspondence of Female set-specific and Female-Male consensus modules",
               cex.text = 1.0, cex.lab = 1.0, setStdMargins = FALSE);
dev.off();
```

The resulting color-coded table is shown in Fig. 1. The table indicates that most female set-specific modules have a consensus counterpart. This indirectly shows that the module structure in the male liver expression data is very similar to the female data. Interestingly, there are two female modules (labeled by grey60 and lightgreen colors) that have no consensus counterpart; almost all genes in the two female modules are labeled "grey", that is unassigned, in the consensus network.

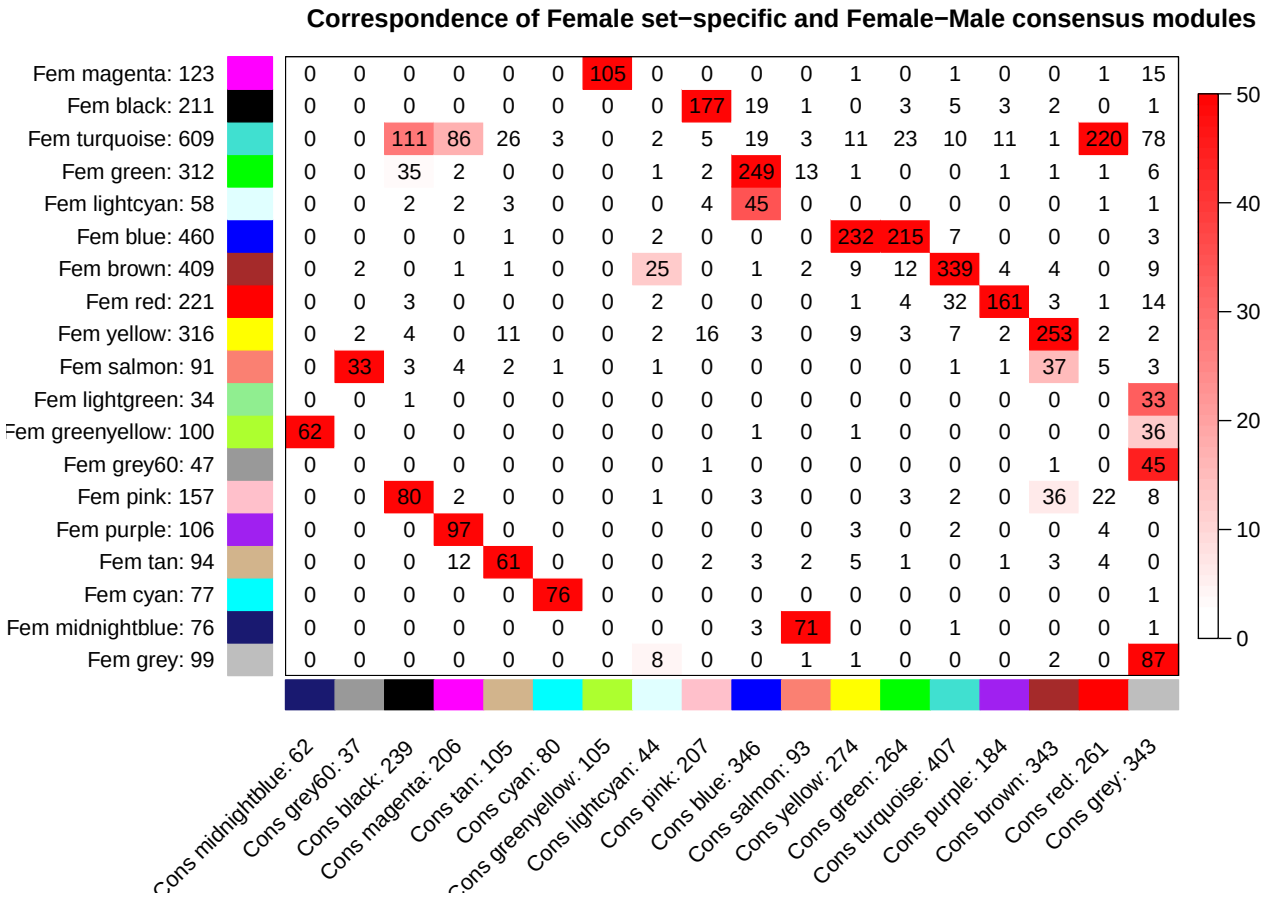

Figure 1: Correspondence of female set-specific modules and the female-male consensus modules. Each row of the table corresponds to one female set-specific module (labeled by color as well as text), and each column corresponds to one consensus module. Numbers in the table indicate gene counts in the intersection of the corresponding modules. Coloring of the table encodes $-\log(p)$, with p being the Fisher's exact test p-value for the overlap of the two modules. The stronger the red color, the more significant the overlap is. The table indicates that most female set-specific modules have a consensus counterpart.