main

July 12, 2021

1 WGCNA

- @author = 'Apua Paquola'
- Edits by K.J. Benjamin
- Edits2 by Arthur S. Feltrin
  - New scale-free plots, export data to create network on cytoscape/igraph and format for jupyter notebook (05/2019)
  - Conversion from Rscript to jupyter notebook

Final edits by K.J. Benjamin for publication

[1]:
```
PARAM_NETWORK_TYPE = 'signed'
```

1.1 Prepare Data and Traits Table

[2]:
```
filter_outliers = function(expression, z_threshold = 2.5)
{
    # Input: an expression matrix
    # Output: an expression matrix with outliers removed
    # Remove samples with z normalized total distance from other samples >␣
 ↪z_threshold

    sample_distance = dist(expression)
    dist_z = scale(colSums(as.matrix(sample_distance)))
    stopifnot(all(rownames(dist_z) == rownames(expression)))

    keepSamples = dist_z < z_threshold

    new_expression = expression[keepSamples,]
    new_expression
}
```

[3]:
```
prepare_data=function()
{
  suppressMessages(library(dplyr))
  # Load sample data
  load("../../../differential_analysis/hippocampus/_m/genes/voomSVA.RData")
  sample_table = v$design %>% as.data.frame %>% select(-Intercept) %>%
```

```
    rename("Ancestry"="EA", "Sex"="Male")

  # Load residualized expression
  vsd <- data.table::fread(paste0("../../../differential_analysis/hippocampus/",
                                  "_m/genes/residualized_expression.tsv")) %>%
    replace(is.na(.), "") %>% tibble::column_to_rownames("V1")
  print(dim(vsd))

  # Keep only the columns and rows that are present in
  # both the sample table and vsd file
  samples = intersect(colnames(vsd), rownames(sample_table))
  vsd = vsd[,samples]
  sample_table = sample_table[samples,]

  # WGCNA data import
  suppressMessages(library(WGCNA))
  options(stringsAsFactors = FALSE)
  datExpr0 = t(vsd)

  # Remove offending genes and samples from the data
  gsg = goodSamplesGenes(datExpr0, verbose = 3);
  if (!gsg$allOK)
  {
    datExpr0 = datExpr0[gsg$goodSamples, gsg$goodGenes]
  }
  datExpr=datExpr0
  # Remove outliers
  datExpr = filter_outliers(datExpr0, z_threshold = 2.5)
  rm(datExpr0)
  # Clean data
  samples = intersect(rownames(datExpr), rownames(sample_table))
  sample_table = sample_table[samples,]
  datExpr = datExpr[samples,]
  print(dim(datExpr))
  save(datExpr, sample_table, file = '00.RData')
}
```

1.2 Create Sample Dendogram Based on Distance (h)

[4]:
```
prepare_traits = function()
{
    lnames = load('00.RData')
    # Associate traits with samples
    traitRows = match(rownames(datExpr), rownames(sample_table))
    datTraits = sample_table[traitRows,]
    # Diagnostic plot: Sample dendrogram and trait heatmap
    pdf(file='sample_dendrogram_and_trait_heatmap.pdf',height=16,width = 22)
```

```
    sampleTree2 = hclust(dist(datExpr), method = "average")
    # Convert traits to a color representation: white means
    # low, red means high, grey means missing entry
    traitColors = numbers2colors(traitRows, signed=FALSE);
    # Plot the sample dendrogram and the colors underneath.
    plotDendroAndColors(sampleTree2, traitColors, groupLabels="Avg. Counts",
                        main = "Sample dendrogram and trait heatmap",
                        cex.dendroLabels=0.7)
    dev.off()
    # Print output
    plotDendroAndColors(sampleTree2, traitColors, groupLabels="Avg. Counts",
                        main = "Sample dendrogram and trait heatmap",
                        cex.dendroLabels=0.75)
    save(datExpr, sample_table, datTraits, file = "01.RData")
}
```

1.3 Calculate Scale-Free Topology

[5]:
```
plot_power_parameter=function(datExpr, plot_filename)
{
    # Choose a set of soft-thresholding powers
    powers = seq(from = 1, to=30, by=1)
    # Call the network topology analysis function
    sft = pickSoftThreshold(datExpr, networkType = PARAM_NETWORK_TYPE,
                            powerVector = powers, verbose = 5)
    # Plot the results:
    pdf(file=plot_filename)
    par(mfcol = c(2,2));
    par(mar = c(4.2, 4.5 , 2.2, 0.5),oma=c(0,0,2,0))
    cex1 = 0.7;
    # Scale-free topology fit index as a function of the
    # soft-thresholding power
    plot(sft$fitIndices[,1], -sign(sft$fitIndices[,3])*sft$fitIndices[,2],
         xlab="Soft Threshold (power)",
         ylab="Scale Free Topology Model Fit,signed R^2",type="n",
         main = paste("Scale independence"))
    text(sft$fitIndices[,1], -sign(sft$fitIndices[,3])*sft$fitIndices[,2],
         labels=powers,cex=cex1,col="blue");
    # this line corresponds to using an R^2 cut-off of h
    abline(h=0.80,col="red")
    # Mean connectivity as a function of the soft-thresholding power
    plot(sft$fitIndices[,1], sft$fitIndices[,5],
         xlab="Soft Threshold (power)", ylab="Mean Connectivity",
         type="n", main = paste("Mean connectivity"))
    text(sft$fitIndices[,1], sft$fitIndices[,5], labels=powers,␣
↪cex=cex1,col="blue")
    #####
```

```
    plot(sft$fitIndices[,1], sft$fitIndices[,6],
         xlab="Soft Threshold (power)", ylab="Median Connectivity",
         type="n", main = paste("Median connectivity"))
    text(sft$fitIndices[,1], sft$fitIndices[,6], labels=powers,␣
↪cex=cex1,col="blue")
    #####
    plot(sft$fitIndices[,1], sft$fitIndices[,7],
         xlab="Soft Threshold (power)", ylab="Max Connectivity",
         type="n", main = paste("Max connectivity"))
    text(sft$fitIndices[,1], sft$fitIndices[,7], labels=powers,␣
↪cex=cex1,col="blue")
    dev.off()
    ####plot on jupyter
    par(mfcol = c(2,2));
    par(mar = c(4.2, 4.5 , 2.2, 0.5),oma=c(0,0,2,0))
    cex1 = 0.7;
    # Scale-free topology fit index as a function of the
    # soft-thresholding power
    plot(sft$fitIndices[,1], -sign(sft$fitIndices[,3])*sft$fitIndices[,2],
         xlab="Soft Threshold (power)",
         ylab="Scale Free Topology Model Fit,signed R^2",type="n",
         main = paste("Scale independence"))
    text(sft$fitIndices[,1], -sign(sft$fitIndices[,3])*sft$fitIndices[,2],
         labels=powers,cex=cex1,col="blue");
    # this line corresponds to using an R^2 cut-off of h
    abline(h=0.80,col="red")
    # Mean connectivity as a function of the soft-thresholding power
    plot(sft$fitIndices[,1], sft$fitIndices[,5],
         xlab="Soft Threshold (power)", ylab="Mean Connectivity",
         type="n", main = paste("Mean connectivity"))
    text(sft$fitIndices[,1], sft$fitIndices[,5], labels=powers,␣
↪cex=cex1,col="blue")
    #####
    plot(sft$fitIndices[,1], sft$fitIndices[,6],
         xlab="Soft Threshold (power)", ylab="Median Connectivity",
         type="n", main = paste("Median connectivity"))
    text(sft$fitIndices[,1], sft$fitIndices[,6], labels=powers,␣
↪cex=cex1,col="blue")
    #####
    plot(sft$fitIndices[,1], sft$fitIndices[,7],
         xlab="Soft Threshold (power)",ylab="Max Connectivity", type="n",
         main = paste("Max connectivity"))
    text(sft$fitIndices[,1], sft$fitIndices[,7], labels=powers,␣
↪cex=cex1,col="blue")
}
```

[6]:
```
figure_out_power_parameter=function()
{
    library(WGCNA)
    options(stringsAsFactors = FALSE);
    #enableWGCNAThreads(nThreads=16)
    lnames = load(file = '01.RData')
    plot_power_parameter(datExpr, 'power_parameter_selection.pdf')
}
```

1.4 Build the Network

[7]:
```
construct_network=function()
{
  library(WGCNA)
  options(stringsAsFactors = FALSE);
  enableWGCNAThreads(nThreads=16)
  lnames = load(file = "01.RData")

  # softPower value from previous plot power_parameter_selection.pdf
  softPower = 14; #check this value, it changes accordingly to your data! You␣
 ↪should
  # ALWAYS choose a value equal or above (better) 0.8
  cor <- WGCNA::cor
  net = blockwiseModules(datExpr, #mergeCutHeight = 0.2,
                         power = softPower,
                         networkType = PARAM_NETWORK_TYPE,
                         TOMType = PARAM_NETWORK_TYPE,
                         numericLabels = TRUE,
                         corType = "bicor",
                         saveTOMs = TRUE, saveTOMFileBase = "TOM",
                         verbose = 3, maxBlockSize=30000)

  moduleLabels = net$colors
  moduleColors = labels2colors(net$colors)
  MEs = net$MEs;
  geneTree = net$dendrograms[[1]];
  save(net, MEs, moduleLabels, moduleColors, geneTree, softPower, file = "02.
 ↪RData")
}
#cyt = exportNetworkToCytoscape(modTOM,
```

1.5 Use Topology Overlap Matrix (TOM) to cluster the genes on the networks into different modules

[8]:
```
plot_cluster_dendrogram=function()
{
    library(WGCNA)
    options(stringsAsFactors = FALSE);
    enableWGCNAThreads(nThreads=16)
    load(file = "02.RData")
    pdf(file="cluster_dendrogram.pdf",height=16,width = 22)
    mergedColors = labels2colors(net$colors)
    plotDendroAndColors(net$dendrograms[[1]], mergedColors[net$blockGenes[[1]]],
                        "Module Colors", dendroLabels = FALSE, hang = 0.03,
                        addGuide = TRUE, guideHang = 0.05, cex.dendroLabels=0.3)
    dev.off()
    # Print output
    plotDendroAndColors(net$dendrograms[[1]], mergedColors[net$blockGenes[[1]]],
                       "Module Colors", dendroLabels = FALSE, hang = 0.03,
                       addGuide = TRUE, guideHang = 0.05, cex.dendroLabels=0.3)
}
```

1.6 Use Pearson Correlation to measure the correlation between each module eigenvalue (kME) and the various sample traits

[9]:
```
correlate_with_traits=function()
{
    library(WGCNA)
    options(stringsAsFactors = FALSE)
    enableWGCNAThreads(nThreads=16)
    lnames = load(file = "01.RData")
    lnames = load(file = "02.RData")
    # Define numbers of genes and samples
    nGenes = ncol(datExpr);
    nSamples = nrow(datExpr);
    # Recalculate MEs with color labels
    MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes
    MEs = orderMEs(MEs0)
    moduleTraitCor = cor(MEs, datTraits, use = "p");
    moduleTraitPvalue = corPvalueStudent(moduleTraitCor, nSamples);
    # Plot
    pdf(file="module_trait_relationships.pdf", height=22,width = 26)
    # Will display correlations and their p-values
    textMatrix = paste(signif(moduleTraitCor, 2), "\n(",
                       signif(moduleTraitPvalue, 1), ")", sep = "");
    dim(textMatrix) = dim(moduleTraitCor)
    par(mar = c(6, 8.5, 3, 3));
    # Display the correlation values within a heatmap plot
```

```
    labeledHeatmap(Matrix = moduleTraitCor,
                   xLabels = names(datTraits),
                   yLabels = names(MEs),
                   ySymbols = names(MEs),
                   colorLabels = FALSE,
                   naColor = "grey",
                   colors = blueWhiteRed(50),
                   textMatrix = textMatrix,
                   setStdMargins = FALSE,
                   cex.text = 0.9,
                   zlim = c(-1,1),
                   main = paste("Module kME-Trait Correlation"))
    dev.off()
    # Print output
    textMatrix = paste(signif(moduleTraitCor, 2), "\n(",
                       signif(moduleTraitPvalue, 1), ")", sep = "");
    dim(textMatrix) = dim(moduleTraitCor)
    par(mar = c(12, 6.5, 3, 0.5));
    # Display the correlation values within a heatmap plot
    labeledHeatmap(Matrix = moduleTraitCor, xLabels = names(datTraits),
                   yLabels = names(MEs), ySymbols = names(MEs),
                   colorLabels = FALSE, naColor = "grey",
                   colors = blueWhiteRed(50), textMatrix = textMatrix,
                   setStdMargins = FALSE, cex.text = 0.55, zlim = c(-1,1),
                   main = paste("Module kME-Trait Correlation"))
}
```

1.7 Export the main results

[10]:
```
export_eigengene_tables = function()
{
    library(WGCNA)
    options(stringsAsFactors = FALSE)
    lnames = load(file = "01.RData")
    lnames = load(file = "02.RData")
    # Define numbers of genes and samples
    nGenes = ncol(datExpr)
    nSamples = nrow(datExpr)
    # Recalculate MEs with color labels
    MEs0 = moduleEigengenes(datExpr, moduleColors)$eigengenes
    rownames(MEs0) = rownames(datExpr)
    write.csv(MEs0, 'eigengenes.csv')
    # Write modules
    modules = data.frame(row.names=colnames(datExpr), module=moduleColors)
    write.csv(modules, 'modules.csv')
    save(datExpr,softPower,moduleColors, file = "cytoscapenetwork.Rdata")
}
```

1.8 Run the functions and plot the results

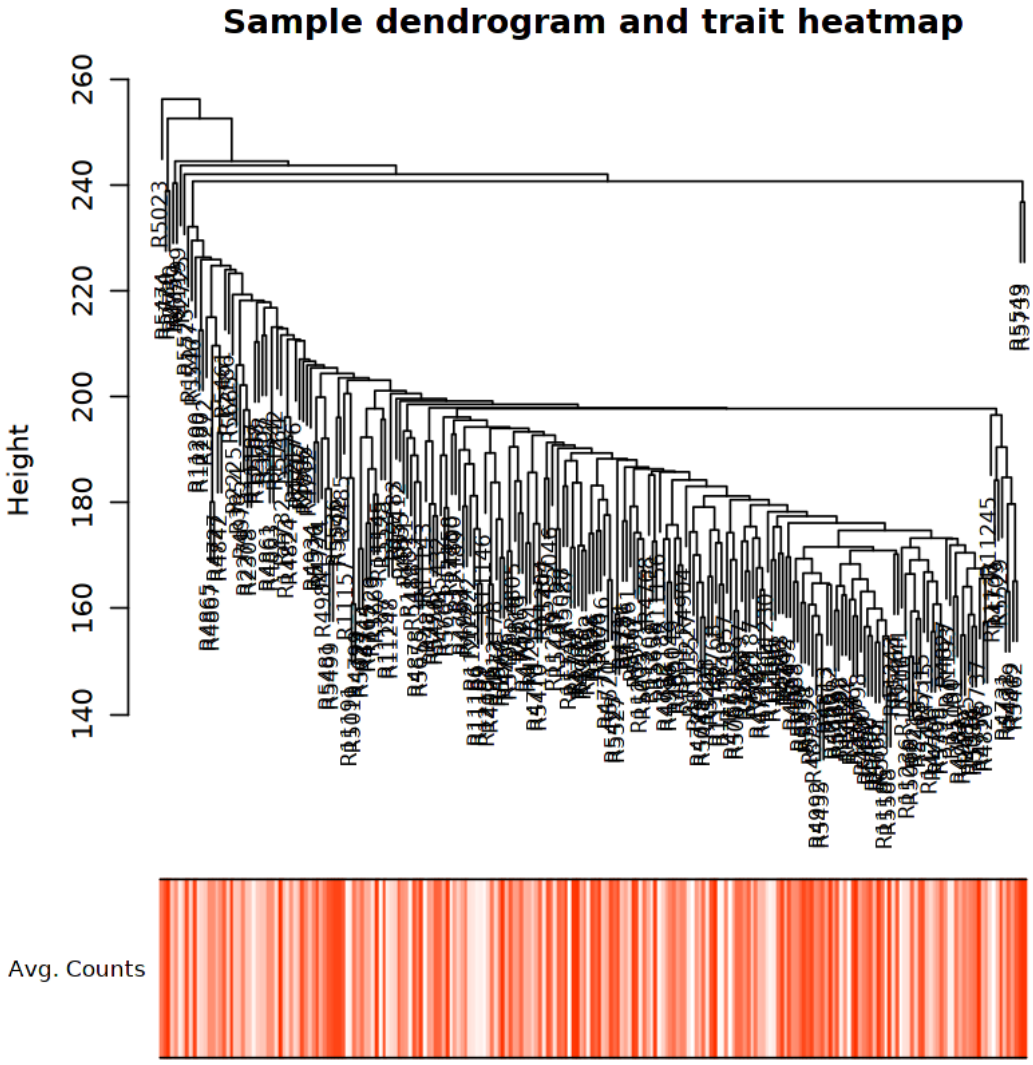
[11]:
```
prepare_data()
```

```
Loading required package: limma

Warning message in
data.table::fread(paste0("../../../differential_analysis/hippocampus/", :
"Detected 242 column names but the data has 243 columns (i.e. invalid file).
Added 1 extra default column name for the first column which is guessed to be
row names or an index. Use setnames() afterwards if this guess is not correct,
or fix the file write command that created the file to create a valid file."

[1] 22269   242
 Flagging genes and samples with too many missing values…
  ..step 1
[1]   233 22269
```

[12]:
```
# 1 - Sample dendrogram and trait heatmap
prepare_traits()
```



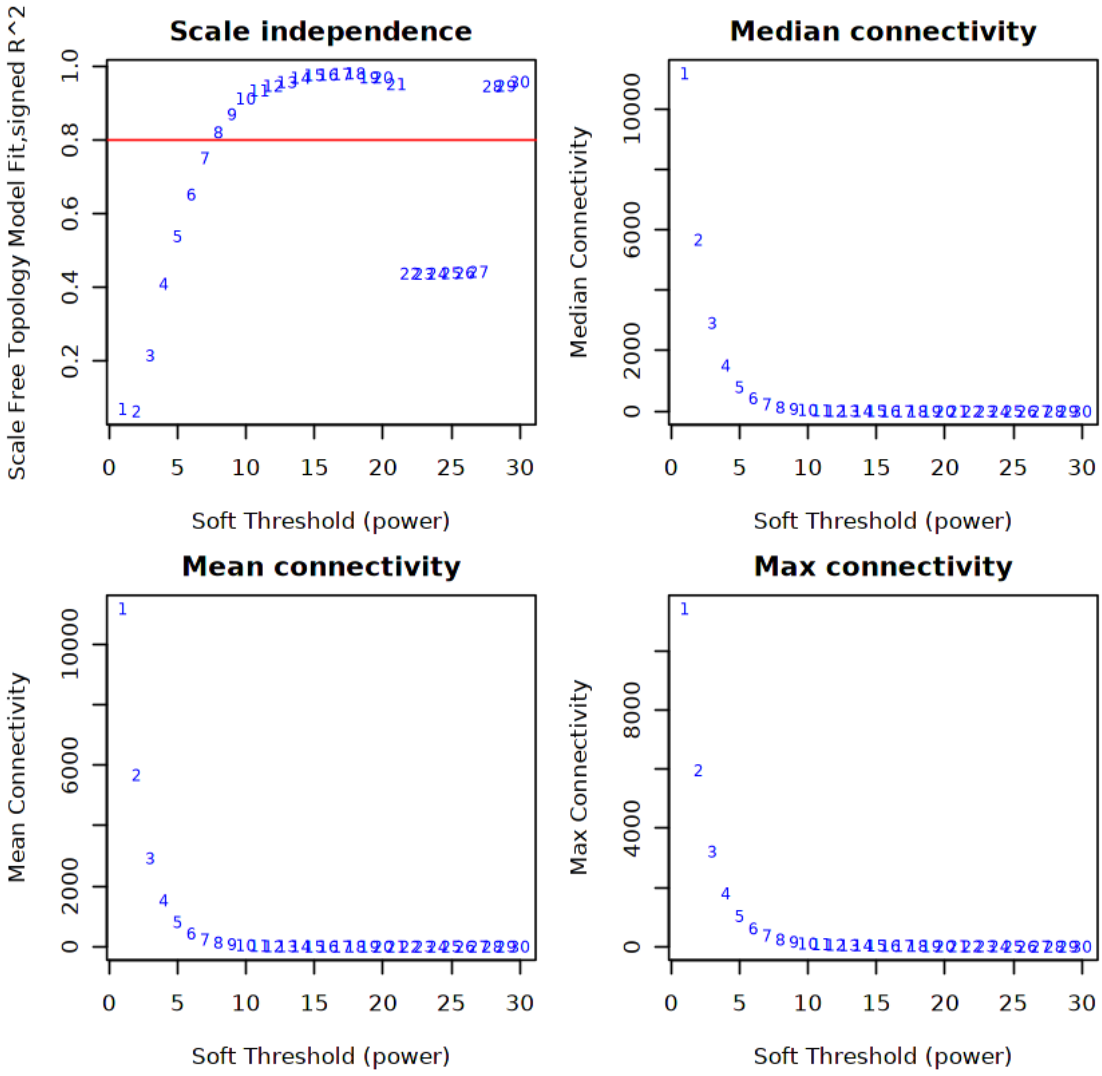
[13]:
```
# 2 - Scale Free Topology Model Fit
figure_out_power_parameter()
```

```
pickSoftThreshold: will use block size 2009.
 pickSoftThreshold: calculating connectivity for given powers…
   ..working on genes 1 through 2009 of 22269
```

```
Warning message:
"executing %dopar% sequentially: no parallel backend registered"
```

```
   ..working on genes 2010 through 4018 of 22269
   ..working on genes 4019 through 6027 of 22269
   ..working on genes 6028 through 8036 of 22269
   ..working on genes 8037 through 10045 of 22269
```

```
   ..working on genes 10046 through 12054 of 22269
   ..working on genes 12055 through 14063 of 22269
   ..working on genes 14064 through 16072 of 22269
   ..working on genes 16073 through 18081 of 22269
   ..working on genes 18082 through 20090 of 22269
   ..working on genes 20091 through 22099 of 22269
   ..working on genes 22100 through 22269 of 22269
   Power SFT.R.sq  slope truncated.R.sq  mean.k. median.k.   max.k.
1      1    0.070 -44.90          0.974 1.12e+04  1.12e+04 11400.00
2      2    0.064 -18.20          0.931 5.67e+03  5.67e+03  5960.00
3      3    0.215 -18.40          0.936 2.92e+03  2.91e+03  3220.00
4      4    0.409 -15.40          0.944 1.52e+03  1.52e+03  1800.00
5      5    0.539 -11.80          0.950 8.05e+02  7.97e+02  1040.00
6      6    0.652  -9.36          0.964 4.31e+02  4.24e+02   622.00
7      7    0.751  -7.67          0.980 2.35e+02  2.29e+02   384.00
8      8    0.821  -6.43          0.990 1.29e+02  1.25e+02   248.00
9      9    0.870  -5.38          0.995 7.26e+01  6.86e+01   165.00
10    10    0.912  -4.50          0.994 4.15e+01  3.83e+01   113.00
11    11    0.934  -4.17          0.996 2.41e+01  2.16e+01    86.20
12    12    0.947  -3.87          0.996 1.43e+01  1.23e+01    69.40
13    13    0.957  -3.56          0.997 8.69e+00  7.08e+00    57.60
14    14    0.968  -3.22          0.998 5.40e+00  4.12e+00    48.90
15    15    0.976  -2.93          0.997 3.44e+00  2.43e+00    42.20
16    16    0.977  -2.69          0.993 2.25e+00  1.45e+00    36.90
17    17    0.979  -2.49          0.992 1.51e+00  8.71e-01    32.60
18    18    0.980  -2.32          0.993 1.04e+00  5.30e-01    28.90
19    19    0.970  -2.20          0.980 7.39e-01  3.26e-01    25.90
20    20    0.969  -2.08          0.980 5.37e-01  2.02e-01    23.20
21    21    0.951  -2.00          0.958 3.99e-01  1.27e-01    20.90
22    22    0.437  -2.56          0.391 3.03e-01  8.01e-02    19.00
23    23    0.437  -2.46          0.396 2.35e-01  5.10e-02    17.30
24    24    0.437  -2.37          0.401 1.85e-01  3.29e-02    15.80
25    25    0.439  -2.30          0.405 1.49e-01  2.13e-02    14.50
26    26    0.440  -2.23          0.406 1.21e-01  1.39e-02    13.30
27    27    0.442  -2.17          0.412 9.94e-02  9.16e-03    12.20
28    28    0.946  -1.64          0.962 8.27e-02  6.07e-03    11.20
29    29    0.947  -1.62          0.963 6.95e-02  4.07e-03    10.40
30    30    0.957  -1.58          0.972 5.89e-02  2.72e-03     9.57
```



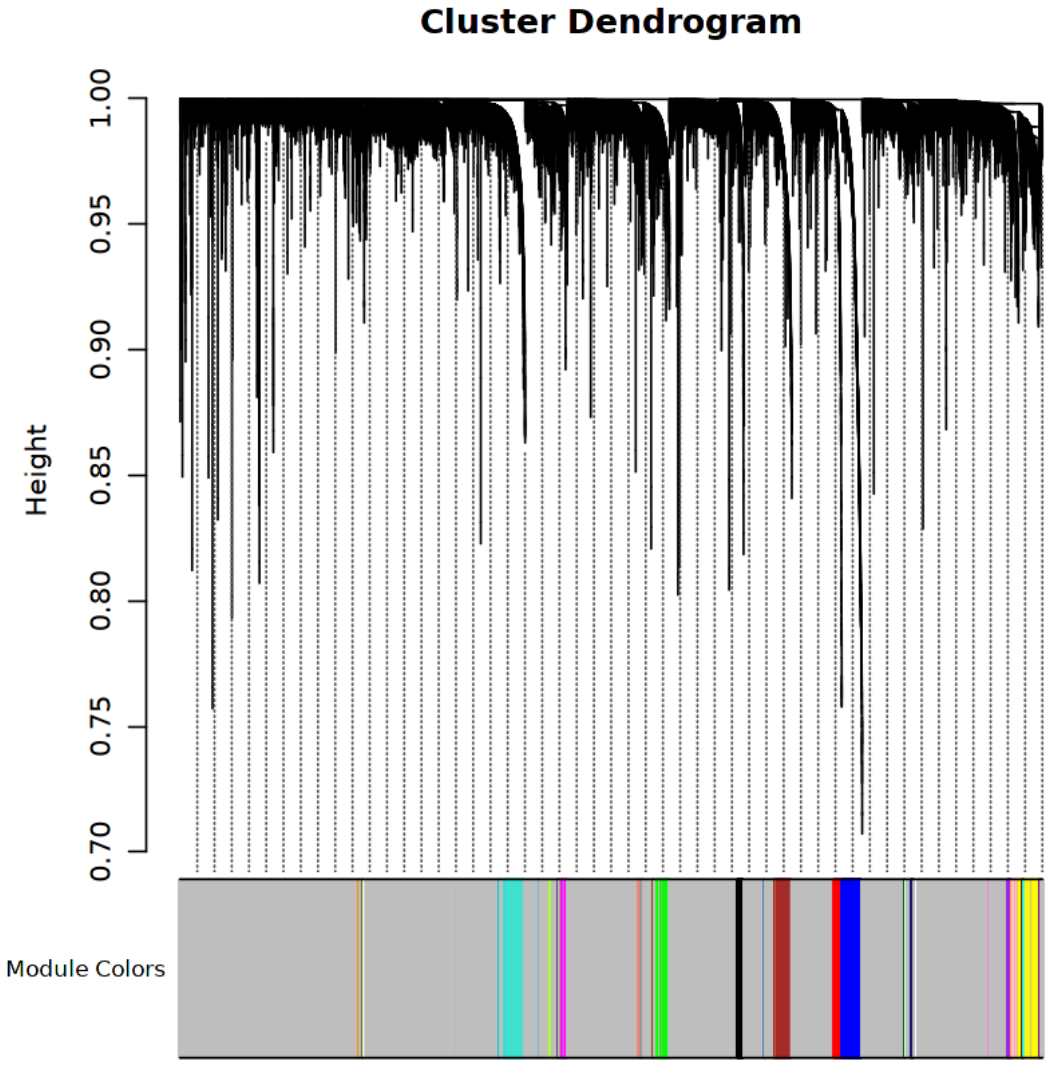
[14]:
```
construct_network()
```

```
Allowing parallel execution with up to 16 working processes.
 Calculating module eigengenes block-wise from all genes
   Flagging genes and samples with too many missing values…
    ..step 1
 ..Working on block 1 .
    TOM calculation: adjacency..
    ..will use 16 parallel threads.
     Fraction of slow calculations: 0.000000
    ..connectivity..
    ..matrix multiplication (system BLAS)..
    ..normalization..
```

```
   ..done.
  ..saving TOM for block 1 into file TOM-block.1.RData
…clustering..
…detecting modules..
…calculating module eigengenes..
…checking kME in modules..
    ..removing 1 genes from module 2 because their KME is too low.
    ..removing 1 genes from module 5 because their KME is too low.
 ..reassigning 6 genes from module 1 to modules with higher KME.
 ..reassigning 4 genes from module 2 to modules with higher KME.
 ..reassigning 8 genes from module 3 to modules with higher KME.
 ..reassigning 23 genes from module 4 to modules with higher KME.
 ..reassigning 13 genes from module 5 to modules with higher KME.
 ..reassigning 10 genes from module 7 to modules with higher KME.
 ..reassigning 5 genes from module 9 to modules with higher KME.
 ..reassigning 3 genes from module 10 to modules with higher KME.
 ..reassigning 3 genes from module 11 to modules with higher KME.
 ..reassigning 2 genes from module 13 to modules with higher KME.
 ..reassigning 1 genes from module 14 to modules with higher KME.
 ..reassigning 4 genes from module 15 to modules with higher KME.
 ..reassigning 3 genes from module 16 to modules with higher KME.
 ..reassigning 1 genes from module 17 to modules with higher KME.
 ..reassigning 2 genes from module 20 to modules with higher KME.
 ..reassigning 1 genes from module 21 to modules with higher KME.
 ..reassigning 1 genes from module 23 to modules with higher KME.
 ..reassigning 2 genes from module 26 to modules with higher KME.
 ..reassigning 1 genes from module 28 to modules with higher KME.
..merging modules that are too close..
    mergeCloseModules: Merging modules whose distance is less than 0.15
      Calculating new MEs…
```

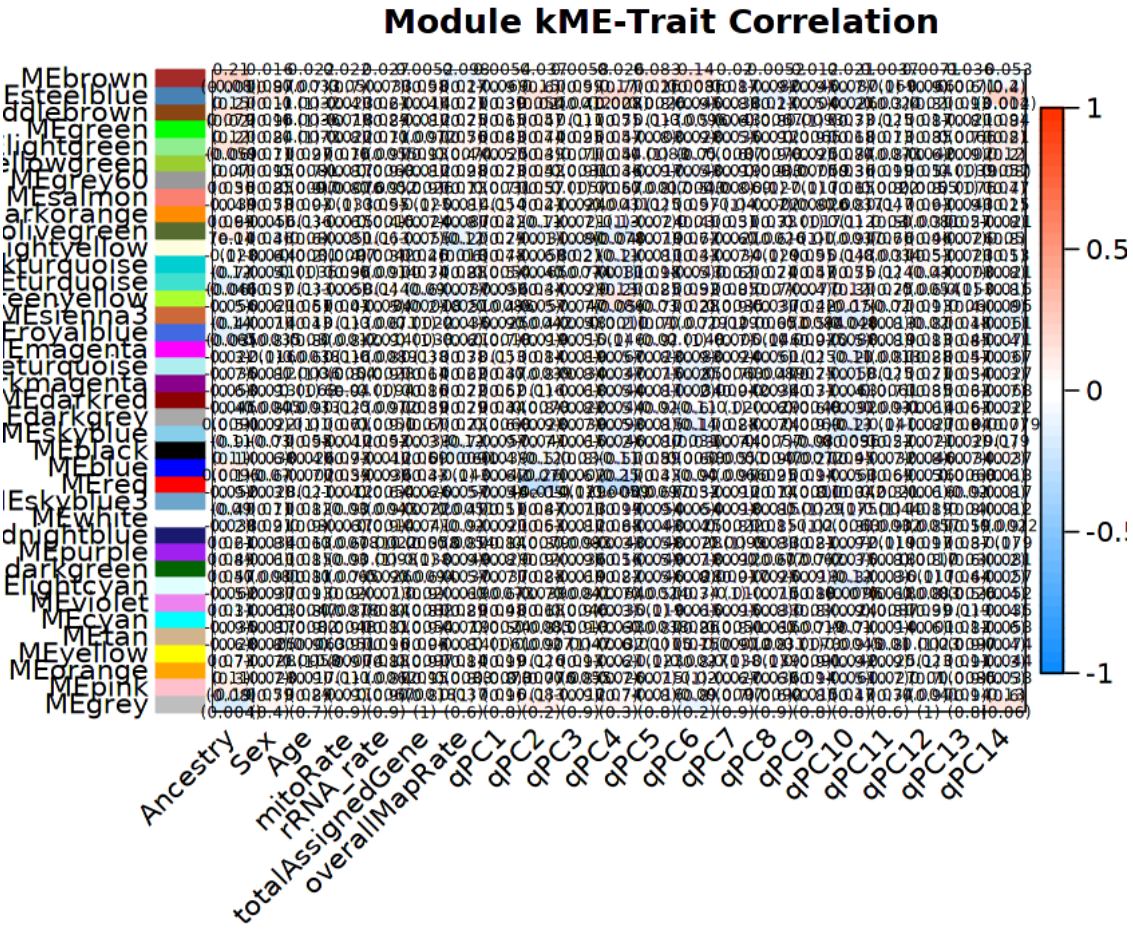
[15]:
```
#3 - TOM Dendogram
plot_cluster_dendrogram()
```

```
Allowing parallel execution with up to 16 working processes.
```



[16]:
```
#4 - Module Eigenvalue Correlation with sample's traits
correlate_with_traits()
export_eigengene_tables()
```

```
Allowing parallel execution with up to 16 working processes.
```



1.9 Repreducibility Information

[17]:
```
Sys.time()
proc.time()
options(width = 120)
sessioninfo::session_info()
```

```
[1] "2021-07-12 10:46:37 EDT"
```

```
    user   system  elapsed
4092.798 1198.755  630.204
```

```
 Session info
 setting  value
```

```
version  R version 4.0.3 (2020-10-10)
os       Arch Linux
system   x86_64, linux-gnu
ui       X11
language (EN)
collate  en_US.UTF-8
ctype    en_US.UTF-8
tz       America/New_York
date     2021-07-12

 Packages
package        * version  date       lib source
AnnotationDbi    1.52.0   2020-10-27 [1] Bioconductor
assertthat       0.2.1    2019-03-21 [1] CRAN (R 4.0.2)
backports        1.2.1    2020-12-09 [1] CRAN (R 4.0.2)
base64enc        0.1-3    2015-07-28 [1] CRAN (R 4.0.2)
Biobase          2.50.0   2020-10-27 [1] Bioconductor
BiocGenerics     0.36.1   2021-04-16 [1] Bioconductor
bit              4.0.4    2020-08-04 [1] CRAN (R 4.0.2)
bit64            4.0.5    2020-08-30 [1] CRAN (R 4.0.2)
blob             1.2.1    2020-01-20 [1] CRAN (R 4.0.2)
cachem           1.0.5    2021-05-15 [1] CRAN (R 4.0.3)
Cairo            1.5-12.2 2020-07-07 [1] CRAN (R 4.0.2)
checkmate        2.0.0    2020-02-06 [1] CRAN (R 4.0.2)
cli              3.0.0    2021-06-30 [1] CRAN (R 4.0.3)
cluster          2.1.0    2019-06-19 [2] CRAN (R 4.0.3)
codetools        0.2-16   2018-12-24 [2] CRAN (R 4.0.3)
colorspace       2.0-2    2021-06-24 [1] CRAN (R 4.0.3)
crayon           1.4.1    2021-02-08 [1] CRAN (R 4.0.3)
data.table       1.14.0   2021-02-21 [1] CRAN (R 4.0.3)
DBI              1.1.1    2021-01-15 [1] CRAN (R 4.0.2)
digest           0.6.27   2020-10-24 [1] CRAN (R 4.0.2)
doParallel       1.0.16   2020-10-16 [1] CRAN (R 4.0.3)
dplyr          * 1.0.7    2021-06-18 [1] CRAN (R 4.0.3)
dynamicTreeCut * 1.63-1   2016-03-11 [1] CRAN (R 4.0.3)
ellipsis         0.3.2    2021-04-29 [1] CRAN (R 4.0.3)
evaluate         0.14     2019-05-28 [1] CRAN (R 4.0.2)
fansi            0.5.0    2021-05-25 [1] CRAN (R 4.0.3)
fastcluster    * 1.2.3    2021-05-24 [1] CRAN (R 4.0.3)
fastmap          1.1.0    2021-01-25 [1] CRAN (R 4.0.2)
foreach          1.5.1    2020-10-15 [1] CRAN (R 4.0.2)
foreign          0.8-80   2020-05-24 [2] CRAN (R 4.0.3)
Formula          1.2-4    2020-10-16 [1] CRAN (R 4.0.2)
generics         0.1.0    2020-10-31 [1] CRAN (R 4.0.2)
ggplot2          3.3.5    2021-06-25 [1] CRAN (R 4.0.3)
glue             1.4.2    2020-08-27 [1] CRAN (R 4.0.2)
GO.db            3.12.1   2021-04-08 [1] Bioconductor
gridExtra        2.3      2017-09-09 [1] CRAN (R 4.0.2)
```

```
gtable           0.3.0    2019-03-25 [1] CRAN (R 4.0.2)
Hmisc            4.5-0    2021-02-28 [1] CRAN (R 4.0.3)
htmlTable        2.2.1    2021-05-18 [1] CRAN (R 4.0.3)
htmltools        0.5.1.1  2021-01-22 [1] CRAN (R 4.0.2)
htmlwidgets      1.5.3    2020-12-10 [1] CRAN (R 4.0.2)
impute           1.64.0   2020-10-27 [1] Bioconductor
IRanges          2.24.1   2020-12-12 [1] Bioconductor
IRdisplay        1.0      2021-01-20 [1] CRAN (R 4.0.2)
IRkernel         1.2      2021-05-11 [1] CRAN (R 4.0.3)
iterators        1.0.13   2020-10-15 [1] CRAN (R 4.0.2)
jpeg             0.1-8.1  2019-10-24 [1] CRAN (R 4.0.2)
jsonlite         1.7.2    2020-12-09 [1] CRAN (R 4.0.2)
knitr            1.33     2021-04-24 [1] CRAN (R 4.0.3)
lattice          0.20-41  2020-04-02 [2] CRAN (R 4.0.3)
latticeExtra     0.6-29   2019-12-19 [1] CRAN (R 4.0.2)
lifecycle        1.0.0    2021-02-15 [1] CRAN (R 4.0.3)
limma          * 3.46.0   2020-10-27 [1] Bioconductor
magrittr         2.0.1    2020-11-17 [1] CRAN (R 4.0.2)
Matrix           1.3-4    2021-06-01 [1] CRAN (R 4.0.3)
matrixStats      0.59.0   2021-06-01 [1] CRAN (R 4.0.3)
memoise          2.0.0    2021-01-26 [1] CRAN (R 4.0.2)
munsell          0.5.0    2018-06-12 [1] CRAN (R 4.0.2)
nnet             7.3-14   2020-04-26 [2] CRAN (R 4.0.3)
pbdZMQ           0.3-5    2021-02-10 [1] CRAN (R 4.0.3)
pillar           1.6.1    2021-05-16 [1] CRAN (R 4.0.3)
pkgconfig        2.0.3    2019-09-22 [1] CRAN (R 4.0.2)
png              0.1-7    2013-12-03 [1] CRAN (R 4.0.2)
preprocessCore   1.52.1   2021-01-08 [1] Bioconductor
purrr            0.3.4    2020-04-17 [1] CRAN (R 4.0.2)
R6               2.5.0    2020-10-28 [1] CRAN (R 4.0.2)
RColorBrewer     1.1-2    2014-12-07 [1] CRAN (R 4.0.2)
Rcpp             1.0.7    2021-07-07 [1] CRAN (R 4.0.3)
repr             1.1.3    2021-01-21 [1] CRAN (R 4.0.2)
rlang            0.4.11   2021-04-30 [1] CRAN (R 4.0.3)
rpart            4.1-15   2019-04-12 [2] CRAN (R 4.0.3)
RSQLite          2.2.7    2021-04-22 [1] CRAN (R 4.0.3)
rstudioapi       0.13     2020-11-12 [1] CRAN (R 4.0.2)
S4Vectors        0.28.1   2020-12-09 [1] Bioconductor
scales           1.1.1    2020-05-11 [1] CRAN (R 4.0.2)
sessioninfo      1.1.1    2018-11-05 [1] CRAN (R 4.0.2)
stringi          1.6.2    2021-05-17 [1] CRAN (R 4.0.3)
stringr          1.4.0    2019-02-10 [1] CRAN (R 4.0.2)
survival         3.2-7    2020-09-28 [2] CRAN (R 4.0.3)
tibble           3.1.2    2021-05-16 [1] CRAN (R 4.0.3)
tidyselect       1.1.1    2021-04-30 [1] CRAN (R 4.0.3)
utf8             1.2.1    2021-03-12 [1] CRAN (R 4.0.3)
uuid             0.1-4    2020-02-26 [1] CRAN (R 4.0.2)
vctrs            0.3.8    2021-04-29 [1] CRAN (R 4.0.3)
```

```
 WGCNA          * 1.70-3   2021-02-28 [1] CRAN (R 4.0.3)
 withr            2.4.2    2021-04-18 [1] CRAN (R 4.0.3)
 xfun             0.24     2021-06-15 [1] CRAN (R 4.0.3)

[1] /home/jbenja13/R/x86_64-pc-linux-gnu-library/4.0
[2] /usr/lib/R/library
```