Gaussian Graphical Models in Metabolomics

Raji Balasubramanian (UMass-Amherst) and Denise Scholtens (Northwestern Feinberg School of Medicine)

Monday, March 15, 2021

Graphical models in medicine

Network Medicine

- **Fundamental principle**: “.. a disease is rarely a consequence of an abnormality in a single effector gene product. Instead, the disease phenotype is a reflection of various pathobiological processes that interact in a complex network. A corollary of this hypothesis is that the interdependencies among a cell’s molecular components lead to deep functional, molecular, and causal relationships among apparently distinct phenotypes.”
- **Reference**: Barabasi, A. et al. (2011), Network Medicine: A Network-based Approach to Human Disease, Nature Reviews Genetics.

Metabolites as networks

Metabolites are naturally represented as networks:

- **Nodes**: represent individual metabolites.
- **Edges (undirected)**: denote pairwise metabolite relationships.

Example network

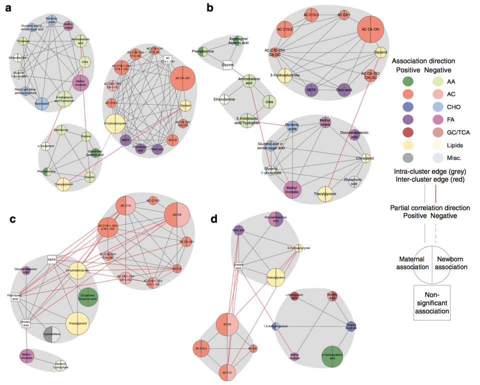


Fig. 3 Subnetworks of maternal metabolites associated with both maternal BMI adjusted for maternal glucose and newborn outcomes. Plotting features are as described in Fig. 1. Subnetwork of maternal (**a**) fasting and (**c**) 1 h metabolites associated with both maternal BMI and newborn SSF. Subnetwork of maternal (**b**) fasting and (**d**) 1 h metabolites associated with both maternal BMI and BW. These subnetworks indicate correlated metabolites that are jointly associated with both maternal BMI and newborn outcomes

Figure 1: Maternal BMI and newborn SSF associated metabolite networks from Sandler, V.,Reisetter, A. C. et. al., Diabetologia, 2017.

Correlation networks

- Correlation networks are established methods for constructing metabolite networks.
- Edges in correlation networks depict pairwise correlations between metabolite pairs.
- Networks are often created by thresholding on a correlation cut-off.

- **Recent example from literature:** A network analysis of biomarkers for Type 2 Diabetes in the Nurses Health Study. [1]

[1] Huang, T., Glass, K. et al., Diabetes, 2018.

Correlation networks

- **Drawback:** Correlations between metabolite pairs can be driven by direct and indirect relationships.
- Drivers of high correlation include shared or common enzymatic activities. [2]
- Large number of non-zero pairwise correlations are usually observed.
- Absence of an edge results from satisfying a **strong** criterion of marginal independence between metabolite pairs. [3]

[2] Su and Clish, Metabolomics and Network Medicine, 2017

[3] Strimmer, K., Notes on Gaussian Graphical Models. http://www.strimmerlab.org/notes/ggm.html

Gaussian graphical models (GGM)

- **Model:** Metabolites are multivariate Gaussian with mean μ and covariance matrix Σ.
- The precision (concentration) matrix $\Omega = \Sigma^{-1}$.
- If $\Omega_{jk} = 0$, **if and only** if the ith metabolite is independent of the jth metabolite, given all other variables.

GGM estimation

- **Meinshausen and Buhlmann (2006)**: estimates $\Omega_{jk} = 0$ by fitting a lasso to each metabolite, using all others as predictors.
- $\hat{\Omega}_{jk} \neq 0$: if the estimated coefficients of metabolite i on j AND vice-versa are non-zero.

- **Friedman et al. (2007)**: Glasso and variants for exact maximization of the penalized log-likelihood.

Model selection

- Gaussian graphical model estimation involves a process to estimate the **optimal regularization parameter (λ)**.
- Large values of λ correspond to increasing sparsity of the resulting graph.

- Stability approach for regularization selection (StARS): uses a subsampling approach to estimate the optimal λ.
- Rotation information criterion (RIC): uses a permutation approach to estimate λ.

Correlation network versus GGM

- **Correlation network:** An edge between metabolite pairs can result from both direct AND indirect relationships.
- **GGM:** An edge exists ONLY if the metabolite pair is dependent after accounting for all other indirect relationships.

Data

HAPO Metabolomics

- **Hyperglycemia and Adverse Pregnancy Outcome (HAPO) Study** conducted during 2000 - 2006 at 15 international field centers.
- Blood samples were obtained during a 75-g oral glucose tolerance test (OGTT) between 24 and 32 weeks gestation.
- Metabolites were measured in maternal fasting and 1-h serum samples from **400** mothers in each ancestry group (Afro-Caribbean, Mexican American, Northern European, Thai).
- Mothers were sampled to span the range of maternal glucose and BMI.

HAPO Metabolomics

Data Format:

- **Column 1**: ID
- **Column 2**: Ancestry Group
- **Column 3**: Fasting glucose
- **Columns 4-54**: 51 metabolites

HAPO Metabolomics

Loading data ..

```r
mydat <- read.csv(file = "Data/hapo_metabolomics_2020.csv")
print(mydat[1:3,1:10])
```

```
##       id anc_gp  fpg    mt1_1     mt1_2    mt1_3    mt1_4     mt1_5
## 1 hm0001    ag3 75.6 218.2223  76.99525 19.06366 14.23091  86.75162
## 2 hm0002    ag3 84.6 292.6314 136.41320 43.14854 17.77549 120.17344
## 3 hm0003    ag4 79.2 361.1135  79.98370 22.15848 13.05497  74.75441
##      mt1_7
## 1 64.00578
## 2 91.30156
## 3 83.67878
```

HAPO Metabolomics

Three groups of metabolites:

- Prefix **mt1**: Amino Acids (AA)
- Prefix **mt2**: Acyl carnitines (AC)
- Prefix **mt3**: Other

HAPO Metabolomics

Let's take a look at the numbers by **ancestry group**:

```
ag <- mydat[,2]
table(ag)
```

```
## ag
## ag1 ag2 ag3 ag4
## 400 400 400 400
```

HAPO Metabolomics

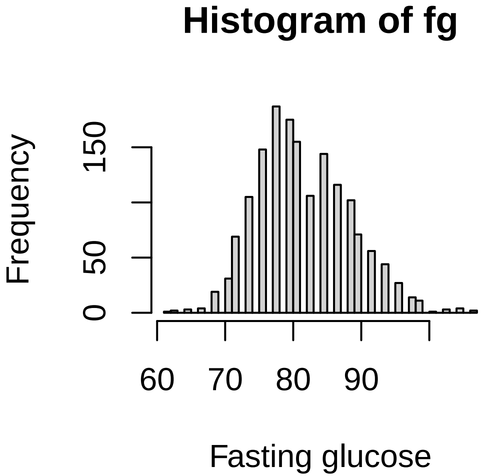
Let's take a look at the distribution of **fasting glucose**:

```
fg <- mydat[,3]
summary(fg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   61.20   77.40   81.00   81.63   86.40  106.20
```

HAPO Metabolomics

Let's take a look at the distribution of **fasting glucose**:



Introduction to network analysis in R

Preliminaries

- **igraph R** package: provides a way of representing graphs and various tools for working with graphs.

Preliminaries

- Let's work with a small (p=6) set of metabolites sampled from the HAPO dataset.
- As an example, we start with a simple correlation network of 6 metabolites

```
mx <- mydat[,-c(1:3)]
mx.1 <- mx[ag == "ag1", c(1,2,16,17,34,35)]
cor.1 <- round(cor(mx.1, use="pairwise.complete.obs"), digits=2)

### Create an adjacency matrix using a threshold of 0.1
adj.1 <- matrix(0, nrow(cor.1), nrow(cor.1))
adj.1[abs(cor.1) > 0.1] <- 1
colnames(adj.1) <- rownames(adj.1) <- colnames(cor.1)
```

Defining network objects in R

Let p denote the number of metabolites in our network.

- **Adjacency matrix**: $p \times p$ matrix, where i, j element is 1 if there is an edge between metabolite i and metabolite j, and 0 otherwise.

```
### Adjacency matrix
print(adj.1)
```

```
##       mt1_1 mt1_2 mt2_1 mt2_2 mt3_1 mt3_2
## mt1_1     1     1     1     1     0     0
## mt1_2     1     1     0     0     0     1
## mt2_1     1     0     1     1     0     0
## mt2_2     1     0     1     1     0     0
## mt3_1     0     0     0     0     1     0
## mt3_2     0     1     0     0     0     1
```

igraph R package

We can convert an adjacency matrix to an igraph object.

```
library(igraph)
igraph.obj <- graph.adjacency(adj.1,mode="undirected",weighted=NULL,diag=FALSE)

## Extracting nodes and edges from igraph object
V(igraph.obj)
```

```
## + 6/6 vertices, named, from 7c85825:
## [1] mt1_1 mt1_2 mt2_1 mt2_2 mt3_1 mt3_2
```

```
E(igraph.obj)
```

```
## + 5/5 edges from 7c85825 (vertex names):
## [1] mt1_1--mt1_2 mt1_1--mt2_1 mt1_1--mt2_2 mt1_2--mt3_2 mt2_1--mt2_2
```

Visualizing our network

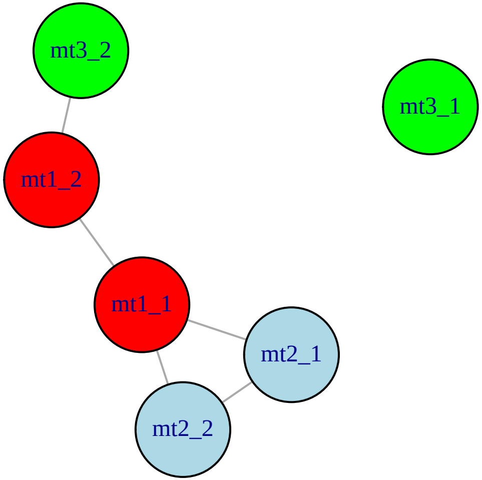
Let's assign metabolite class to each of our nodes and an associated color.

```
### Assigning attributes to the list of nodes

V(igraph.obj)$MxClass <- c(rep("AA",2), rep("AC", 2), rep("Oth",2))
V(igraph.obj)$color <- c(rep("red", 2), rep("light blue",2), rep("green",2))
V(igraph.obj)$size <- 50
V(igraph.obj)$label.cex <- 0.75
```

Visualizing our network

Visualize the network..

```
### Visualizing network
plot.igraph(igraph.obj,vertex.label=colnames(adj.1),layout=layout.fruchterman.reingold)
```



Changing node attributes

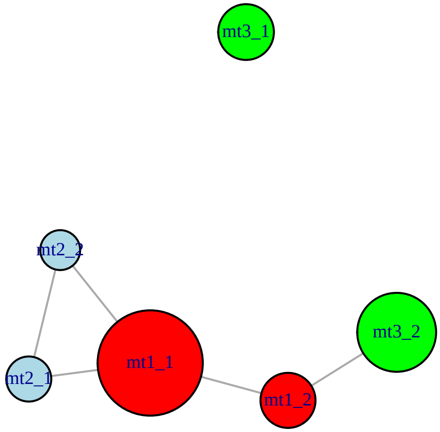
Let's change node size in proportion to significance of association with fasting glucose..

```
### Changing the node size to match the level

### of signficance with outcome (fasting glucose)

myfun <- function(metabolite, outcome){
    mymod <- lm(outcome ~ metabolite)
    minuslogp <- -log(summary(mymod)$coef[2,4])
    return(minuslogp)
}

fg1 <- fg[ag == "ag1"]
vals <- apply(mx.1, 2, myfun, fg1)

### scaling the node size
### changing the font fize
### of the vertex label

V(igraph.obj)$size <- vals*3+20
V(igraph.obj)$label.cex <- 0.6
```

Visualizing our network

Visualize the network after changing node attributes..

```
### Visualizing network
plot.igraph(igraph.obj,vertex.label=colnames(adj.1),layout=layout.fruchterman.reingold)
```



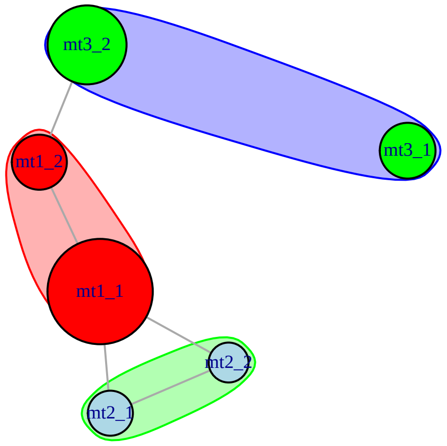
Grouping nodes

We can also visually depict metabolite classes (Amino acids, Acyl carnitines, Other) in our network ..

```
### Visualizing network with node groups
mylist <- list(c("mt1_1","mt1_2"), c("mt2_1","mt2_2"), c("mt3_1","mt3_2"))
```

Grouping nodes

```
plot.igraph(igraph.obj,vertex.label=colnames(adj.1),

            layout=layout.fruchterman.reingold, mark.groups=mylist)
```



Networks in R

There are a myriad of options available for visualizing networks. For more, see help associated with plot.igraph() in the igraph package.

```
### Other layouts (Kamada-Kawai)

### For other options -- Check ?plot.igraph

l <- layout_with_kk(igraph.obj)
plot.igraph(igraph.obj,vertex.label=colnames(adj.1),layout=l, mark.groups=mylist)
```

Gaussian Graphical Models (GGM) in R

GGM in R

We illustrate estimation of the Gaussian graphical model using the R package huge.

To keep in mind:

- Missing values of metabolite levels need to be imputed prior to invoking the functions in **huge**.
- Each metabolite should be standardized to render them of unit variance.

Preliminaries

We prepare metabolite data in ancestry group ag1 for graphical model estimation.

```
### Prepping data for GGM
### Impute missing values
### Standardize

standardizeMetabolite = function(x)
{
  x[x == Inf] <- NA
  x[is.na(x)] <- min(x, na.rm=T)/2
  return((x-mean(x, na.rm=T))/sd(x, na.rm=T))
}

mx.1 <- mx[ag == "ag1",]
mx1.s <- apply(mx.1, 2, standardizeMetabolite)

summary(apply(mx1.s,2,sd))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1       1       1       1       1       1
```

GGM estimation

The key functions involved are:

- **huge:** estimates GGM over a range of penalty parameters (can be left unspecified).
- **huge.select:** implements regularization parameter selection.
  Reference: T. Zhao and H. Liu (2012). The huge Package for High-dimensional Undirected Graph Estimation in R. Journal of Machine Learning Research.

GGM estimation

Regularization parameter selection options include:

- StARS: tends to overselects edges.
- RIC: more computationally efficient, tends to underselect edges.
- **Reference**: T. Zhao and H. Liu (2012). The huge Package for High-dimensional Undirected Graph Estimation in R. Journal of Machine Learning Research.

GGM ESTIMATION

Let's estimate the GGM network for our data..

```
library(huge)
### creates the GGM model object
mbModel <- huge(mx1.s, method="mb")
```

```
## Conducting Meinshausen & Buhlmann graph estimation (mb)....done
```

```
### Optimal parameter selection using ric
 mbOptRIC = huge.select(mbModel, criterion="ric")
```

```
## Conducting rotation information criterion (ric) selection....done
## Computing the optimal graph....done
```

```
### extract the graph corresponding to optimal param
mbOptRICGraph = mbOptRIC$refit
```

GGM

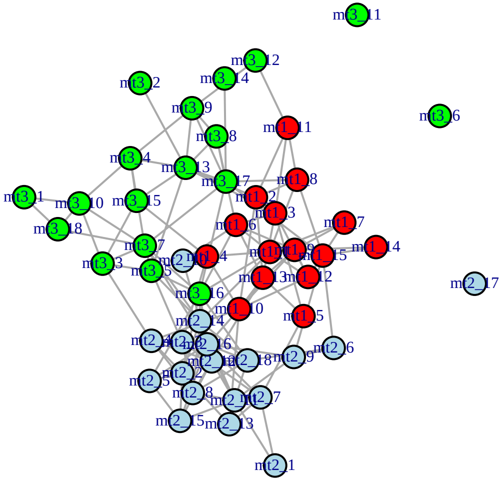
Visualize our estimated GGM ..

Let's estimate the GGM network for our data..

```
myg <- graph_from_adjacency_matrix(mbOptRICGraph, mode="undirected")
### Assigning attributes to the list of nodes

V(myg)$MxClass <- c(rep("AA",15), rep("AC", 18), rep("Oth",18))
V(myg)$color <- c(rep("red", 15), rep("light blue",18), rep("green",18))
V(myg)$size <- 10
V(myg)$label.cex <- 0.5
```

GGM

```
### Visualizing network
plot.igraph(myg,vertex.label=colnames(mx.1),layout=layout.fruchterman.reingold)
```



Other options

- **Method:** can be changed to glasso; huge(.., method="glasso").
- **Selecting λ:** in huge.select(.., criterion="stars").
- **Relaxing Gaussian assumption:** using nonparanormal (npn) transformation; huge.npn() will return a transformed data matrix.

Next ..

Telling stories with GGMs

- Detecting communities within networks
- Differential networks
- Case studies

References

- Su, J. and Clish, C. (2018). Metabolomics and Network Medicine, Network Medicine: Complex Systems in Human Disease and Therapeutics, Harvard University Press.
- Go, KI, Cusick, ME, Valle, D, Childs B, Vidal M, Barabási AL (2007).The human disease network, PNAS, 104(21):8685-90.
- Sandler, V., Reisetter, A. C., Bain, J.R., ..., Scholtens, D.M., Lowe, W.L.Jr (2018) Associations of maternal BMI and insulin resistance with the maternal metabolome and newborn outcomes, Diabetologia, 60(3):518-530.
- Meinshausen, N. and Buhlmann, P. (2006). High-dimensional graphs and variable selection with the Lasso, Annals of Statistics, Vol. 34, No. 3, 1436-1462.
- Friedman, J., Hastie, T. and Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso, Biostatistics, 9(3):432-441.
- Roeder, K., Lafferty, J., Wasserman, L., Zhao, T., Liu, H. (2012) The huge package for high-dimensional undirected graph estimation in R. Journal of Machine Learning Research, (13):1059–1062.

Gaussian Graphical Models in Metabolomics - Part 2

Raji Balasubramanian (UMass-Amherst) and Denise Scholtens (Northwestern Feinberg School of Medicine)

Monday March 15, 2021

1) Subnetworks associated with phenotype

2) Differential network analysis

Beyond simple networks

- Graphical lasso identifies conditional dependence between pairs of metabolites and applies a node-and-edge graph representation of these dependencies
- While estimating conditional dependencies among metabolite pairs is interesting, for most investigations, these dependencies are not of primary interest.

- More complex questions:
  - Which subnetworks are associated with a phenotype?
  - Do networks vary across groups?

1) Subnetworks associated with phenotype

Subnetworks associated with phenotype

- Prior to network analyses, investigators often perform per-metabolite association analyses with a phenotype of interest
- How can per-metabolite and network analyses be linked?
- Some existing approaches:
  - Dittrich et al. (2008) *Bioinformatics*. Identifying functional modules in protein–protein interaction networks: an integrated exact approach.
  - Ben-Hamo et al. (2014) *Bioinformatics*. PhenoNet: identification of key networks associated with disease phenotype.
  - Soul et al. (2015) *Scientific Reports*. PhenomeExpress: A refined network analysis of expression datasets by inclusion of known disease phenotypes.

Subnetworks associated with phenotype

- A simple approach using graphical lasso
  - Identify a set of metabolites, $\mathcal{M}_p$, associated with phenotype
  - Identify additional metabolites, $\mathcal{M}_c$, with Pearson correlation exceeding some threshold (say 0.25) with at least one member of $\mathcal{M}_p$
  - Run graphical lasso on $\mathcal{M}_p \cup \mathcal{M}_c$

Ancestry-specific networks assoc with FPG

In case you'd like to start a new R session, let's reload the libraries and set the working directory.

```
library(igraph)
```

```
## Warning: package 'igraph' was built under R version 4.0.2
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.0.2
```

```
library(iDINGO)
```

```
## Warning: package 'iDINGO' was built under R version 4.0.2
```

```
library(huge)
```

```
## Warning: package 'huge' was built under R version 4.0.2
```

Ancestry-specific networks assoc with fpg

Now read in the data and review some simple descriptors.

```
mydat <- read.csv("Data/hapo_metabolomics_2020.csv")
rownames(mydat) <- mydat$id
dim(mydat)
```

```
## [1] 1600   54
```

```
head(colnames(mydat))
```

```
## [1] "id"     "anc_gp" "fpg"    "mt1_1"  "mt1_2"  "mt1_3"
```

```
table(mydat$anc_gp)
```

```
##
## ag1 ag2 ag3 ag4
## 400 400 400 400
```

Ancestry-specific networks assoc with FPG

Perform simple ancestry-group specific mean imputation of missing metabolite values.

```
hapo_ag <- split(mydat,f=mydat$anc_gp)
length(hapo_ag)
```

```
## [1] 4
```

```
sapply(hapo_ag,FUN=dim)
```

```
##      ag1 ag2 ag3 ag4
## [1,] 400 400 400 400
## [2,]  54  54  54  54
```

```
hapo_ag_m_i <- lapply(hapo_ag,
        FUN=function(x) apply(x[,grep("mt",colnames(x),value=TRUE)],
        MARGIN=2,
        FUN=function(y) ifelse(is.na(y),mean(y,na.rm=TRUE),y)))
```

Ancestry-specific networks assoc with fpg

Check to make sure imputation worked as planned.

```
hapo_m_i <- do.call("rbind",hapo_ag_m_i)
hapo_i <- data.frame(mydat[rownames(hapo_m_i),c("id","anc_gp","fpg")],
                     hapo_m_i)
tapply(mydat[,"mt3_4"],INDEX=mydat$anc_gp,FUN=mean,na.rm=TRUE)
```

```
##      ag1      ag2      ag3      ag4
## 18.11342 22.06506 20.54547 19.95429
```

```
tapply(mydat[,"mt3_12"],INDEX=mydat$anc_gp,FUN=mean,na.rm=TRUE)
```

```
##      ag1      ag2      ag3      ag4
## 26.41744 29.66998 29.01828 26.97278
```

Ancestry-specific networks assoc with FPG

Check to make sure imputation worked as planned.

```
mydat[c(1,2,3,6),c("anc_gp","mt3_4","mt3_12")]
```

```
##         anc_gp    mt3_4   mt3_12
## hm0001     ag3 20.50824 29.37834
## hm0002     ag3       NA 29.51101
## hm0003     ag4 19.89055 27.85653
## hm0006     ag4 20.04486       NA
```

```
hapo_i[rownames(mydat)[c(1,2,3,6)],c("anc_gp","mt3_4","mt3_12")]
```

```
##         anc_gp    mt3_4   mt3_12
## hm0001     ag3 20.50824 29.37834
## hm0002     ag3 20.54547 29.51101
## hm0003     ag4 19.89055 27.85653
## hm0006     ag4 20.04486 26.97278
```

Ancestry-specific networks assoc with fpg

Find subset of metabolites within each ancestry associated with fpg.

```
myfun <- function(metabolite,outcome){
    mymod <- lm(outcome~metabolite)
    minuslogp <- -log(summary(mymod)$coef[2,4])
    return(minuslogp)
}

hapo_i_ag <- split(hapo_i,f=hapo_i$anc_gp)

m_fpg_p_ag <- lapply(hapo_i_ag,
            FUN=function(x){
                x_m <- x[,grep("mt",colnames(x))]
                ans <- apply(x_m,MARGIN=2,FUN=myfun,outcome=x$fpg)
                return(ans)
                })
```

Ancestry-specific networks assoc with fpg

Find subset of metabolites within each ancestry associated with fpg.

```
sig_m_ag <- lapply(m_fpg_p_ag,
        FUN=function(x) names(x[which(x>-log(.05))]))
sig_m_ag
```

```
## $ag1
##  [1] "mt1_1"  "mt1_2"  "mt1_3"  "mt1_5"  "mt1_11" "mt1_12" "mt2_3"  "mt2_8"
##  [9] "mt2_11" "mt3_1"  "mt3_2"  "mt3_3"  "mt3_4"  "mt3_5"  "mt3_10" "mt3_15"
##
## $ag2
##  [1] "mt1_1"  "mt1_2"  "mt1_3"  "mt1_5"  "mt1_11" "mt1_12" "mt2_10" "mt3_4"
##  [9] "mt3_6"  "mt3_9"  "mt3_13" "mt3_16"
##
## $ag3
##  [1] "mt1_1"  "mt1_2"  "mt1_3"  "mt1_5"  "mt1_8"  "mt1_11" "mt1_12" "mt1_15"
##  [9] "mt2_4"  "mt2_8"  "mt2_13" "mt2_14" "mt3_1"  "mt3_6"  "mt3_10" "mt3_13"
##
## $ag4
## [1] "mt1_1"  "mt1_5"  "mt1_12" "mt1_15" "mt2_2"  "mt2_8"  "mt2_14" "mt3_5"
## [9] "mt3_12"
```

Ancestry-specific networks assoc with FPG

Find other metabolites correlated with significant metabolites.

```r
m_cor_ag <- lapply(hapo_ag_m_i,FUN=cor,use="pairwise.complete.obs")
sig_cor_ag <- vector("list",length=4)
names(sig_cor_ag) <- names(sig_m_ag)
for (i in 1:4){
    sig_m_cor_pairs <- m_cor_ag[[i]][sig_m_ag[[i]],]
    sig_m_cor <- names(which(colSums(abs(sig_m_cor_pairs)>=.25)>0))
    sig_m_cor_vals <- hapo_ag_m_i[[i]][,sig_m_cor]
    sig_m_cor_vals_s <- apply(sig_m_cor_vals,MARGIN=2,FUN=scale)
    sig_cor_ag[[i]] <- sig_m_cor_vals_s
}
sapply(sig_cor_ag,FUN=dim)
```

```
##      ag1 ag2 ag3 ag4
## [1,] 400 400 400 400
## [2,]  42  40  44  31
```

Ancestry-specific networks assoc with FPG

Now apply graphical lasso for these subsets of metabolites.

```
mbModel_ag <- lapply(sig_cor_ag,FUN=huge,method="mb")
```

```
## Conducting Meinshausen & Buhlmann graph estimation (mb)....done
## Conducting Meinshausen & Buhlmann graph estimation (mb)....done
## Conducting Meinshausen & Buhlmann graph estimation (mb)....done
## Conducting Meinshausen & Buhlmann graph estimation (mb)....done
```

```
mb_opt_ag <- lapply(mbModel_ag,FUN=huge.select,criterion="ric")
```

```
## Conducting rotation information criterion (ric) selection....done
## Computing the optimal graph....done
## Conducting rotation information criterion (ric) selection....done
## Computing the optimal graph....done
## Conducting rotation information criterion (ric) selection....done
## Computing the optimal graph....done
## Conducting rotation information criterion (ric) selection....done
## Computing the optimal graph....done
```

Ancestry-specific networks assoc with fpg

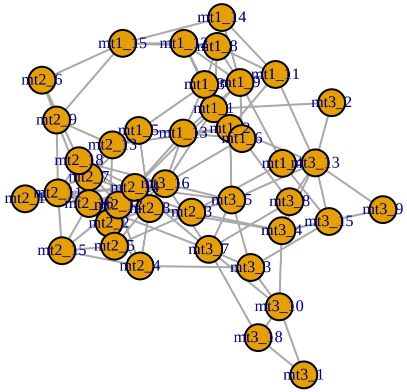
Generate the igraph objects.

```r
ggm_ag_mat <- lapply(mb_opt_ag,FUN=function(x) x$refit)
ggm_ag_g <-lapply(ggm_ag_mat,
                  FUN=graph_from_adjacency_matrix,
                  mode="undirected")

for (i in 1:4){
    V(ggm_ag_g[[i]])$label <- colnames(sig_cor_ag[[i]])
}
```

Ancestry-specific networks assoc with fpg

Now plot the graphs - Ancestry group 1 (layout may vary)

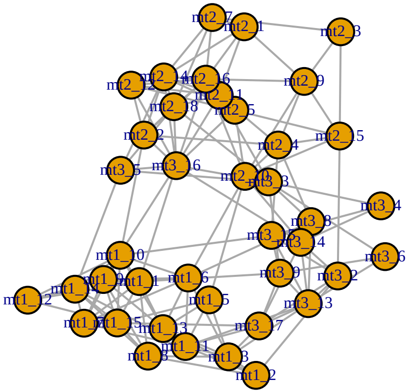
```r
plot(ggm_ag_g[["ag1"]],vertex.label=V(ggm_ag_g[["ag1"]])$label,
     vertex.label.cex=.5)
```



Ancestry-specific networks assoc with fpg

Ancestry group 2

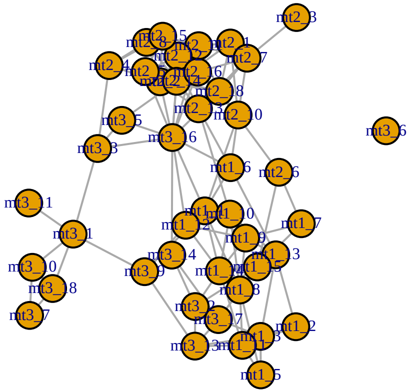
```
plot(ggm_ag_g[["ag2"]],vertex.label=V(ggm_ag_g[["ag2"]])$label,
     vertex.label.cex=.5)
```



Ancestry-specific networks assoc with FPG

Ancestry group 3 - note the singleton node

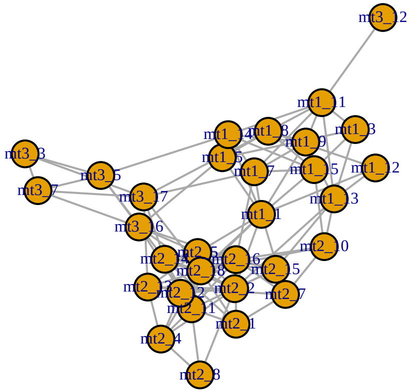
```
plot(ggm_ag_g[["ag3"]],vertex.label=V(ggm_ag_g[["ag3"]])$label,
     vertex.label.cex=.5)
```



Ancestry-specific networks assoc with fpg

Ancestry group 4 - note the singleton node

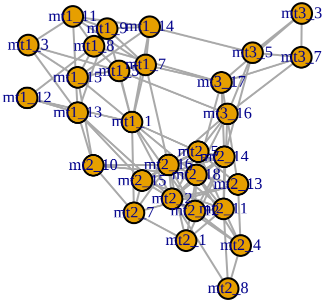
```r
plot(ggm_ag_g[["ag4"]],vertex.label=V(ggm_ag_g[["ag4"]])$label,
     vertex.label.cex=.5)
```



Ancestry-specific networks assoc with FPG

Drop the singletons.

```
ggm_ag_g[[3]] <- delete_vertices(ggm_ag_g[[3]],
                           which(V(ggm_ag_g[[3]])$label=="mt3_6"))
ggm_ag_g[[4]] <- delete_vertices(ggm_ag_g[[4]],
                           which(V(ggm_ag_g[[4]])$label=="mt3_12"))
plot(ggm_ag_g[["ag4"]],vertex.label=V(ggm_ag_g[["ag4"]])$label,
     vertex.label.cex=.5)
```



Community detection

- Visual inspection and biological interpretation of these networks is challenging
- Pick out pairwise relationships? Then what?
- Community detection helps tell a story
- *igraph* package
    - cluster_spinglass (Newman and Girvan, 2004)
    - cluster_fast_greedy
    - cluster_label_prop
    - cluster_walktrap
    - etc.

Community detection

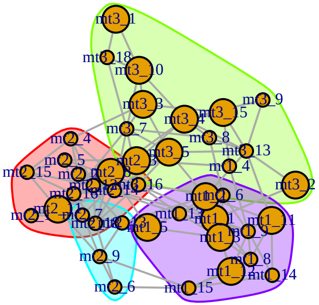
Spinglass clustering on all four graphs

```
ggm_ag_g_spg <- lapply(ggm_ag_g,FUN=cluster_spinglass)
```

Community detection

Spinglass clustering - ancestry group 1

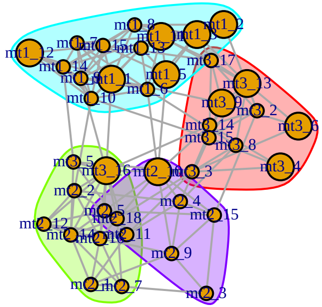
```
plot(ggm_ag_g[["ag1"]],
    vertex.label=V(ggm_ag_g[["ag1"]])$label,
    vertex.label.cex=.5,
    mark.groups=ggm_ag_g_spg[["ag1"]],
    vertex.size=ifelse(V(ggm_ag_g[["ag1"]])$label %in%
         sig_m_ag[["ag1"]],20,10))
```



Community detection

Spinglass clustering - ancestry group 2

```
plot(ggm_ag_g[["ag2"]],
    vertex.label=V(ggm_ag_g[["ag2"]])$label,
    vertex.label.cex=.5,
    mark.groups=ggm_ag_g_spg[["ag2"]],
    vertex.size=ifelse(V(ggm_ag_g[["ag2"]])$label %in%
        sig_m_ag[["ag2"]],20,10))
```



Example from HAPO Metabolomics

- Investigation of associations between maternal metabolites at 28 weeks gestation with newborn phenotypes at birth
- Examined associations within and across four ancestry groups – Afro-Caribbean, European, Mexican-American, Thai
- Used a similar approach to that described here
- For graphical lasso, used residuals from a linear model for each metabolite with predictors for covariates of interest

- Kadakia et al. (2019) *Diabetologia* Maternal metabolites during pregnancy are associated with newborn outcomes and hyperinsulimaemia across ancestries.

Example from HAPO Metabolomics

Maternal fasting metabolites associated with newborn sum of skinfolds under 2 covariate adjustment models

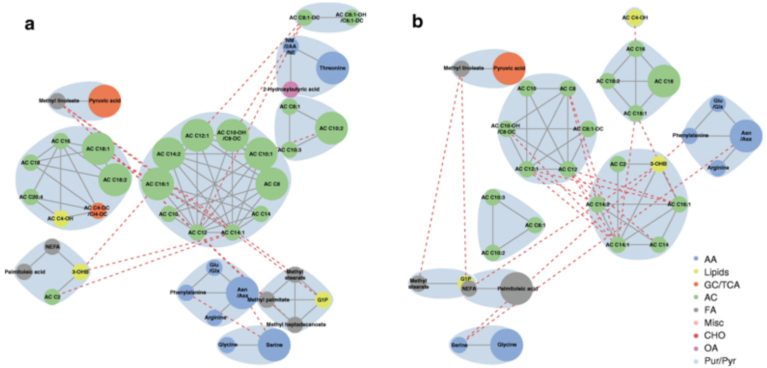


Figure 1: Kadakia et al. (2019)

2) Differential network analysis

Differential network analysis

- Visual inspection suggests there are differences in the ancestry-specific networks we just generated
- But are the differences ‘statistically significant’?
- One approach to differential network analysis:
  - *iDINGO* R package
  - Ha et al. *Bioinformatics* (2015) DINGO: differential network analysis in genomics.
  - Class et al. *Bioinformatics* (2018) iDINGO - integrative differential network analysis in genomics with *Shiny* application.

Differential network analysis

- DINGO estimates a 'global' component of the network, $\mathcal{G}$, that represents edges that are common across groups
- DINGO also estimates 'local' group-specific components, $\mathcal{L}(x)$, that represent unique relationships in each group depending on the value of a categorical variable x.
- For two groups, group-specific edges are identified using a *Differential Score:*

$$\delta_{ab}^{(12)} = \frac{\hat{\phi}_{ab}^{(1)} - \hat{\phi}_{ab}^{(2)}}{s_{ab}^{B}}$$

where $\hat{\phi}_{ab}^{(1)}$ and $\hat{\phi}_{ab}^{(2)}$ are Fisher's Z transformation of the estimates of group-specific partial correlations between metabolites a and b in groups 1 and 2, and s_{ab}^{B} is the bootstrap estimate of the standard error.

Differential network analysis

Let's work with the first two ancestry groups.

```
hapo_2ag <- subset(hapo_i,anc_gp %in% c("ag1","ag2"))
hapo_2ag <- droplevels(hapo_2ag)
hapo_2ag_mt <- hapo_2ag[,grep("mt",colnames(hapo_2ag),value=TRUE)]
dim(hapo_2ag)
```

```
## [1] 800  54
```

```
dim(hapo_2ag_mt)
```

```
## [1] 800  51
```

Differential network analysis

The commented code below would perform the DINGO algorithm. The bootstrapping takes a long time. So we will just load an R object of the results that should be in your working directory.

```
#hapo_2ag_dn <- dingo(hapo_2ag_mt,x=hapo_2ag$anc_gp,B=50)
load("Data/hapo_2ag_dn_B50.rda")
```

Differential network analysis

Let's look at the various components of the output.

```
names(hapo_2ag_dn)
```

```
##  [1] "genepair"   "levels.x"   "R1"         "R2"         "boot.diff"
##  [6] "diff.score" "p.val"      "rho"        "P"          "Q"
## [11] "Psi"        "step.times"
```

```
head(hapo_2ag_dn$genepair)
```

```
##   gene1 gene2
## 1 mt1_1 mt1_2
## 2 mt1_1 mt1_3
## 3 mt1_2 mt1_3
## 4 mt1_1 mt1_4
## 5 mt1_2 mt1_4
## 6 mt1_3 mt1_4
```

```
dim(hapo_2ag_dn$genepair)
```

```
## [1] 1275    2
```

Differential network analysis

More components of the output.

```r
hapo_2ag_dn$levels.x
```

```
## [1] ag1 ag2
## Levels: ag1 ag2
```

```r
length(hapo_2ag_dn$R1)
```

```
## [1] 1275
```

```r
length(hapo_2ag_dn$R2)
```

```
## [1] 1275
```

```r
dim(hapo_2ag_dn$boot.diff)
```

```
## [1] 1275   50
```

Differential network analysis

More components of the output.

```r
length(hapo_2ag_dn$diff.score)
```

```
## [1] 1275
```

```r
length(hapo_2ag_dn$p.val)
```

```
## [1] 1275
```

Differential network analysis

Create a data frame of some of the output

```
hapo_2ag_dn_df <- data.frame(gene1=hapo_2ag_dn$genepair$gene1,
        gene2=hapo_2ag_dn$genepair$gene2,
        genepair=paste(as.character(hapo_2ag_dn$genepair$gene1),
                as.character(hapo_2ag_dn$genepair$gene2),sep=":"),
        R1=hapo_2ag_dn$R1,
        R2=hapo_2ag_dn$R2,
        diff.score=hapo_2ag_dn$diff.score,
        p.val=hapo_2ag_dn$p.val)
```

Differential network analysis

Create a data frame of some of the output.

```
head(hapo_2ag_dn_df)
```

```
##   gene1 gene2    genepair          R1          R2   diff.score     p.val
## 1 mt1_1 mt1_2 mt1_1:mt1_2  0.07809638  0.07832990 -0.016040253 0.9986491
## 2 mt1_1 mt1_3 mt1_1:mt1_3  0.01951538  0.02186509 -0.135858912 0.8148208
## 3 mt1_2 mt1_3 mt1_2:mt1_3  0.40212136  0.40158482  0.047007443 0.9039632
## 4 mt1_1 mt1_4 mt1_1:mt1_4 -0.25814119 -0.25921713  0.092097027 0.8351067
## 5 mt1_2 mt1_4 mt1_2:mt1_4  0.29185984  0.29178087  0.005269656 0.9683637
## 6 mt1_3 mt1_4 mt1_3:mt1_4 -0.24110807 -0.24006858 -0.076546532 0.9051876
```

Differential network analysis

Identify extremely different scores with diff.score>5 or <-5.

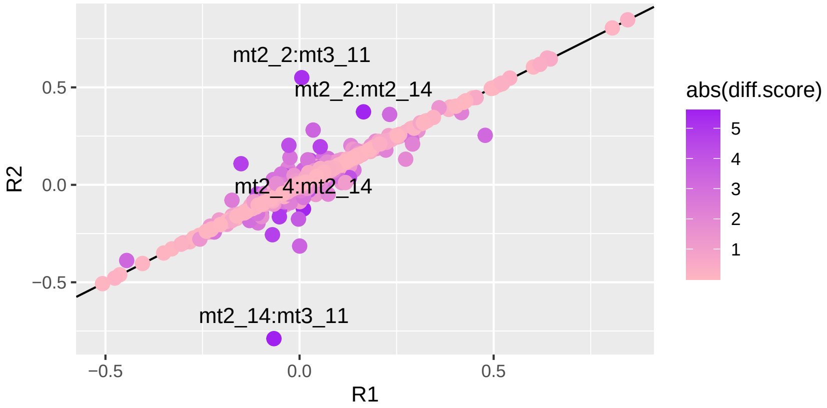
```
hapo_2ag_dn_df$high_ds <- ifelse(abs(hapo_2ag_dn_df$diff.score)>5,
                                 as.character(hapo_2ag_dn_df$genepair),"")
hapo_2ag_dn_df[which(!hapo_2ag_dn_df$high_ds==""),]
```

```
##      gene1  gene2      genepair           R1         R2 diff.score p.val
## 395  mt2_2 mt2_14  mt2_2:mt2_14  0.164750400  0.3744255  -5.521262     0
## 397  mt2_4 mt2_14  mt2_4:mt2_14  0.010156435 -0.1238160   5.522768     0
## 920  mt2_2 mt3_11  mt2_2:mt3_11  0.005775354  0.5498016  -5.201985     0
## 932 mt2_14 mt3_11 mt2_14:mt3_11 -0.065983987 -0.7890289   5.598957     0
##          high_ds
## 395  mt2_2:mt2_14
## 397  mt2_4:mt2_14
## 920  mt2_2:mt3_11
## 932 mt2_14:mt3_11
```

Differential network analysis

Compare R1 and R2, colored by diff.score.

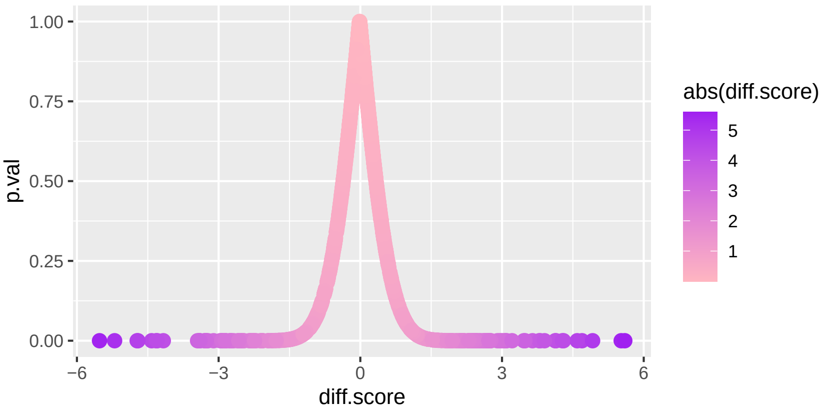
```
ggplot(hapo_2ag_dn_df,aes(x=R1,y=R2)) +
        geom_abline(intercept=0,slope=1) +
        geom_point(aes(color=abs(diff.score)),size=3) +
        geom_text(label=hapo_2ag_dn_df$high_ds,vjust=-1) +
        scale_color_gradient(low="lightpink",high="purple")
```



Differential network analysis

Plot of diff.score by p.val, colored by diff.score.

```
ggplot(hapo_2ag_dn_df,aes(x=diff.score,y=p.val)) +
        geom_point(aes(color=abs(diff.score)),size=3) +
        scale_color_gradient(low="lightpink",high="purple")
```



Differential network analysis

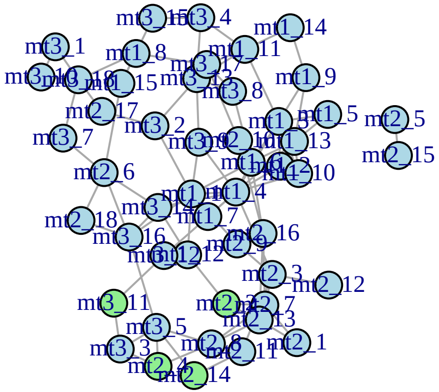
Explore the global component of the dingo graph.

```r
dingo_rho_thresh <- .20
hapo_2ag_dn_df$global <- ifelse(
          (abs(hapo_2ag_dn_df$R1)>dingo_rho_thresh) &
                    (abs(hapo_2ag_dn_df$R2)>dingo_rho_thresh) &
                    (sign(hapo_2ag_dn_df$R1>dingo_rho_thresh)==
                    sign(hapo_2ag_dn_df$R2>dingo_rho_thresh)),1,0)
global_g <- graph_from_edgelist(
  as.matrix(hapo_2ag_dn_df[which(hapo_2ag_dn_df$global==1),
  c("gene1","gene2")]),directed=FALSE)
```

Differential network analysis

Explore the global component of the dingo graph.

```
V(global_g)$color <- rep("light blue",length(V(global_g)))
V(global_g)$color[which(names(V(global_g)) %in%
        c("mt2_2","mt2_4","mt2_14","mt3_11"))] <-"light green"
V(global_g)$size <- 15
V(global_g)$label.cex <- .75
plot(global_g,layout=layout_nicely(global_g))
```



Differential network analysis

Explore the local components of the dingo graphs.

```
hapo_2ag_dn_df$local_ag1 <- ifelse(
          (abs(hapo_2ag_dn_df$R1)>dingo_rho_thresh) &
                    (abs(hapo_2ag_dn_df$R2)<dingo_rho_thresh) &
                    (hapo_2ag_dn_df$p.val<.05),1,0)

hapo_2ag_dn_df$local_ag2 <- ifelse(
          (abs(hapo_2ag_dn_df$R2)>dingo_rho_thresh) &
                    (abs(hapo_2ag_dn_df$R1)<dingo_rho_thresh) &
                    (hapo_2ag_dn_df$p.val<.05),1,0)

table(hapo_2ag_dn_df$local_ag1,hapo_2ag_dn_df$local_ag2)
```

```
##
##          0    1
##    0 1259   11
##    1    5    0
```

Differential network analysis

Explore the local components of the dingo graphs.

```r
local_g_ag1 <- graph_from_edgelist(
  as.matrix(hapo_2ag_dn_df[which((hapo_2ag_dn_df$global+
                                    hapo_2ag_dn_df$local_ag1)==1),
                           c("gene1","gene2")]),directed=FALSE)
local_g_ag2 <- graph_from_edgelist(
  as.matrix(hapo_2ag_dn_df[which((hapo_2ag_dn_df$global+
                                    hapo_2ag_dn_df$local_ag2)==1),
                           c("gene1","gene2")]),directed=FALSE)
```

Differential network analysis

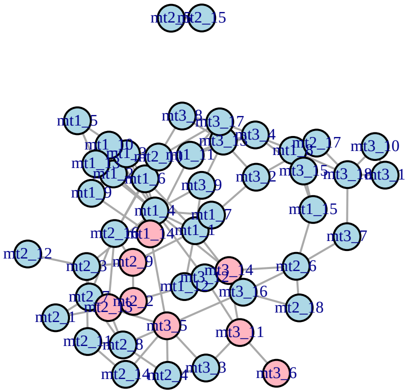
Explore the local components of the dingo graphs.

```r
local_ag1_nodes <- unique(c(as.character(hapo_2ag_dn_df[which(hapo_2ag_dn_df$local_ag1==1),"gene1"]),
                            as.character(hapo_2ag_dn_df[which(hapo_2ag_dn_df$local_ag1==1),"gene2"])))
local_ag2_nodes <- unique(c(as.character(hapo_2ag_dn_df[which(hapo_2ag_dn_df$local_ag2==1),"gene1"]),
                            as.character(hapo_2ag_dn_df[which(hapo_2ag_dn_df$local_ag2==1),"gene2"])))

V(local_g_ag1)$color <- rep("light blue",length(V(local_g_ag1)))
V(local_g_ag1)$color[which(names(V(local_g_ag1)) %in% local_ag1_nodes)] <- "light pink"

V(local_g_ag2)$color <- rep("light blue",length(V(local_g_ag2)))
V(local_g_ag2)$color[which(names(V(local_g_ag2)) %in% local_ag2_nodes)] <- "light green"
```

Differential network analysis

Local component for ancestry group 1

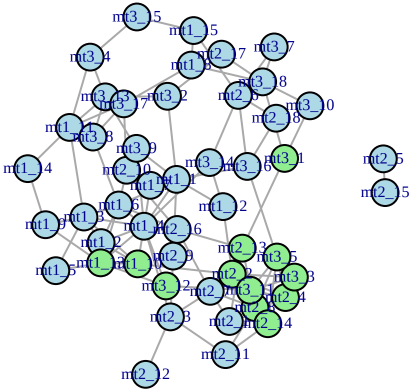
```r
plot(local_g_ag1,vertex.label.cex=.5)
```



Differential network analysis

Local component for ancestry group 2

```r
plot(local_g_ag2,vertex.label.cex=.5)
```



Summary

- Networks are very helpful for 'story telling' in metabolomics (and other omics) settings
- Graphical lasso and related methods focus on conditional dependence
- Gives some assurance that edges aren't simply an artifact of sharing common correlations between a pair of nodes with a third node
- Focusing on subnetworks related to phenotype can place per-metabolite associations into context
- Differential network analyses based on graphical models can point to meaningful differences between groups
- Graphics take a while. . . be patient and use Google!

Acknowledgements

- **Thanks to...**
- **Kate Shutta**, **Octavious Talbot**, **Alan Kuang** and **Nathan Gill** for workshop assistance
- **HAPO Metabolomics** investigators (**PI: William Lowe Jr. MD**) for example data