Gene set enrichment and Network analysis

Yongjin Park

15 February, 2022

What's next after differential expression analysis?

- Differential expression analysis of ~20k genes identified tens or hundreds of significant genes
- Can we evaluate our findings from a systems biology perspective?

What you will learn

1. Gene Set Enrichment Analysis
2. Biological network analysis
3. Learning structures in biological networks

What is Gene Set Analysis?

(Discrete) Gene Set Analysis

Input:

1. A dictionary of gene sets that map genes to sets (gene to gene set mapping)
2. A list of **top** XX number of genes identified in our own study (after FDR control)

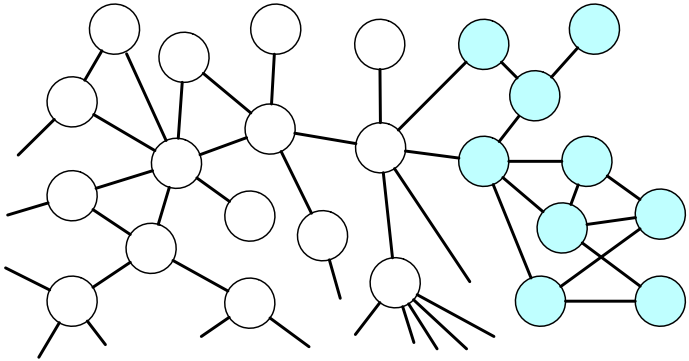
(Rank-based) Gene Set Enrichment

Input:

1. A dictionary of gene sets that map genes to sets
2. A **full** list of gene-level **scores** (e.g., p-values, z-scores)

Output: A table of scores for all the gene sets in the dictionary.

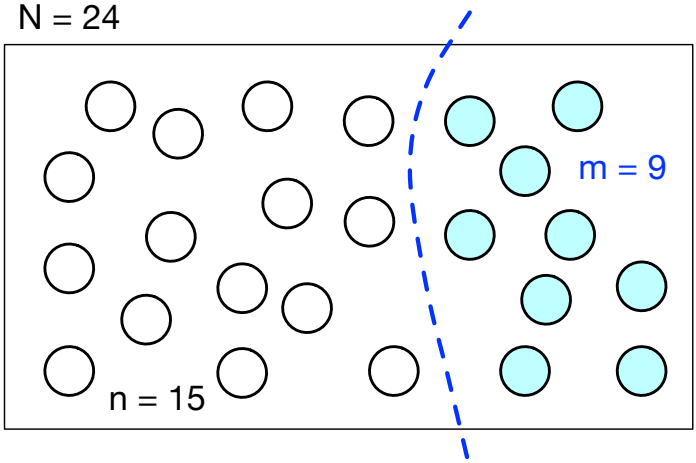
What have we learned from our differential expression analysis?



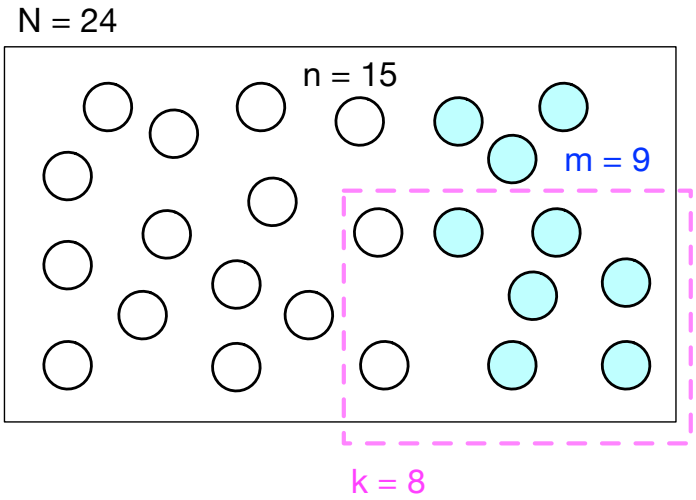
→ **coloured:** some pathway of interest

Let's drop the edges in biological networks

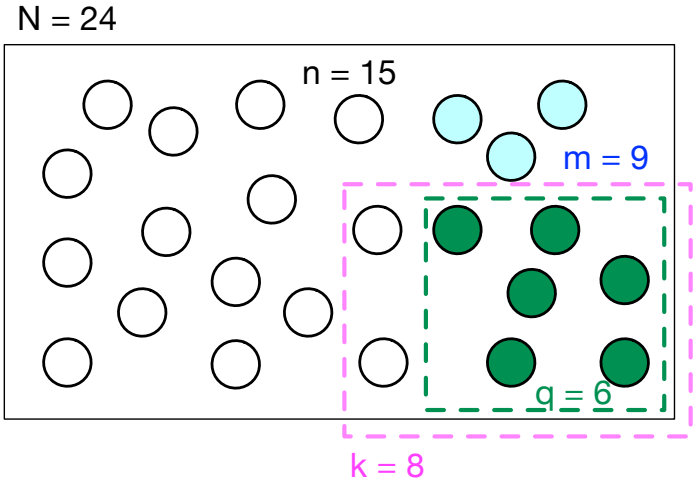
Prior Knowledge of biological pathways define a set of genes (coloured)



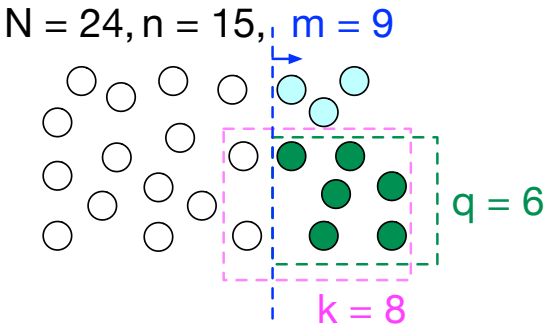
Does this pathway overlap with our DEG list?



Say that we found q genes are overlapping with this pathway



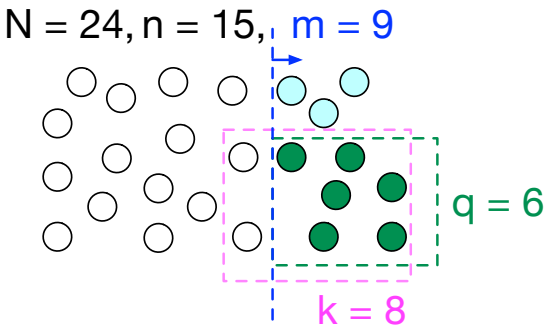
Gene Set Analysis testing over-representation of DEGs



What are the numbers to count?

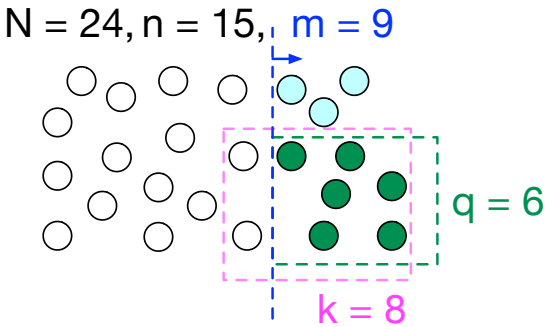
- N: # genes in this universe
- m: # genes in this set
- n: # genes *not* in this gene set
- k: # DEGs in our analysis
- q: # DEGs (of k) overlapping with the set of m genes

Is this overlap of $q = 6$ of $k = 8$ genes significant?



- $N = 24$ # a total of 24 genes
- $m = 9$ genes in this set
- $n = N - m = 15$
- $k = 8$ DEGs
- $q = 6$ out of $k = 8$ overlap

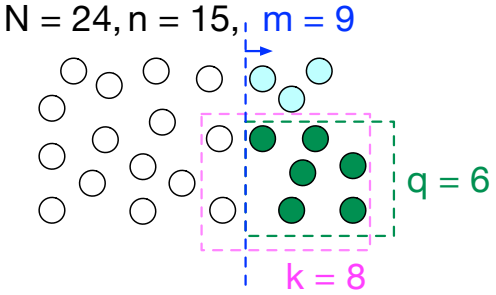
Is this overlap of $q = 6$ of $k = 8$ genes significant?



Questions:

- Is it meaningful enough to report?
- Is it surprising enough that we recapitulated 6/9 (~67 %)?
- What is the null distribution?
- What is the generative/simulation scheme?

How do we find q out of k DEGs overlapping with a gene set of m genes?



Under the null of hypergeometric distribution

1. Sample k DEGs out of N genes
2. Of these k genes, q overlap with a gene set consisting of m genes
3. The rest $k - q$ genes overlap with genes outside of the gene set $N - m$

Review of binomial coefficient: Let's see if we can estimate the null distribution by counting

- How many all possible ways to select $k = 8$ out of $N = 24$ genes, ignoring the order of k selected genes and $N - k$ *not* selected genes?
- We can think of this as three steps: (1) enumerating N genes, (2) partition them into the first k genes and the rest, (3) ignore the order within each partition.

$$\binom{24}{8} = \frac{\{\text{all possible ways to enumerate } 24 \text{ genes}\}}{\{\text{enumerating } 8 \text{ genes}\}\{\text{enumerating } 16 \text{ genes}\}} = \frac{24!}{8!16!}$$

Hypergeometric distribution

1. What is the probability of selecting k DEGs out of N genes?
2. How many possible ways of finding q DEGs overlapping with m genes in the gene set?
3. How many possible ways of finding the rest $k - q$ DEGs overlapping with $(N - m = n)$ genes in the gene set?

Hypergeometric distribution

$$P_0(q|N,m,k) = \sum_{\substack{\#\text{ possible ways of selecting } q \text{ out of } m \\ \#\text{ possible ways of selecting } (k-q) \text{ out of } N-m}} \text{the probability of the gene set}$$

$$= \overbrace{\binom{m}{q}}^{\substack{\#\text{ ways of choosing} \\ q\text{overlap out of } m}} \overbrace{\binom{N-m}{k-q}}^{\substack{\#\text{ ways of choosing} \\ (k-q) \text{ out of } N-m}} \underbrace{\frac{1}{\binom{N}{k}}}_{\text{equal probability of the gene set}}$$

What is the probability of k overlapping DEGs?

Hypergenometric PMF

$$p(x|N,m,k) = \frac{\binom{m}{x}\binom{N-m}{k-x}}{\binom{N}{k}}$$

Hypergeometric CDF

$$p(q|N,m,k) = \sum_{x=0}^{q} \frac{\binom{m}{x}\binom{N-m}{k-x}}{\binom{N}{k}}$$

Hypergeometric test for testing significant overlap

$$H_0 : x \leq q \quad \text{vs.} H_1 : x > q$$

We may observe overlap q genes by random sampling of k genes **without** replacement.

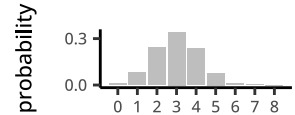
Therefore, we can calculate the p-value:

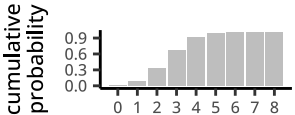
$$P(x > q|n, m, k) = 1 - \sum_{x=0}^{q} \frac{\binom{m}{x}\binom{n}{k-x}}{\binom{n+m}{k}}$$

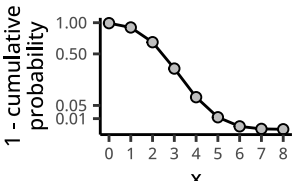
```
phyper(q=6, m=9, n=15, k=8, lower.tail=FALSE)
```

How significant is q overlap in our discovery?

Fixing $m = 9$, $n = 15$, and $k = 8$,





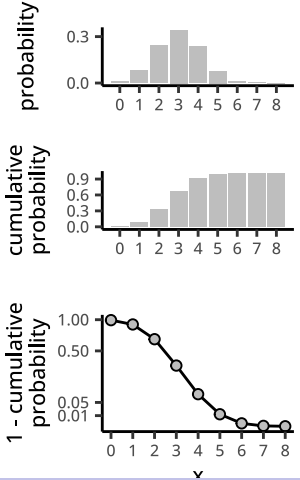


If $q = 3$:

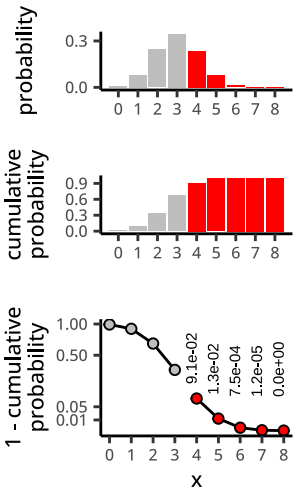
If $q = 6$:

How significant is q overlap in our discovery?

Fixing $m = 9$, $n = 15$, and $k = 8$,



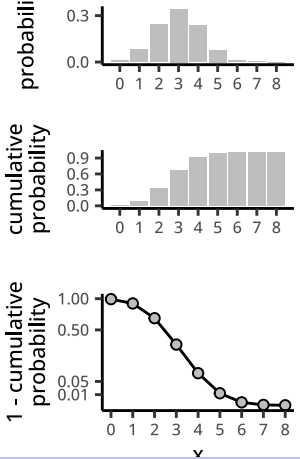
If $q = 3$:



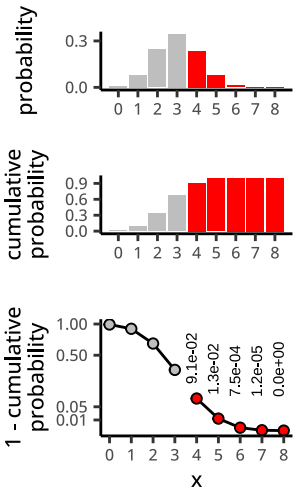
If $q = 6$:

How significant is q overlap in our discovery?

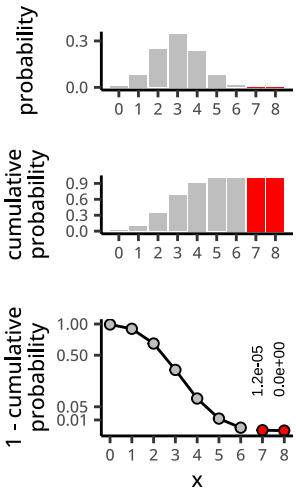
Fixing $m = 9$, $n = 15$, and $k = 8$,



If $q = 3$:

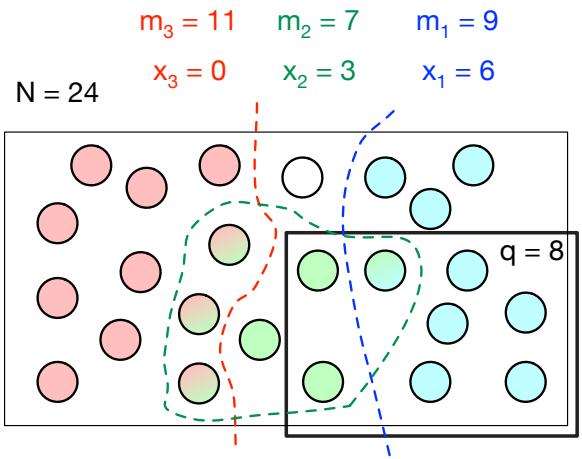


If $q = 6$:



Multiple gene sets, multiple hypothesis testing

Which one is significantly overlapping with $k = 8$ genes?



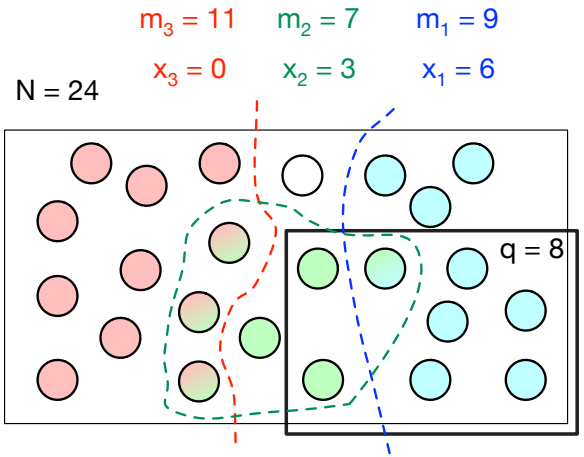
Measure the tail probability (p-value):

```
mm <- c(9, 7, 11)          # |gene set|
xx <- c(6, 3, 0)           # |overlap|
N <- 24                    # N. total
pval <- phyper(xx,         # overlap
               m=mm,       # inside
               n=(N-mm),   # outside
               k=8,        # significant
               lower.tail=FALSE)
```

Which one is significant?

Multiple gene sets, multiple hypothesis testing

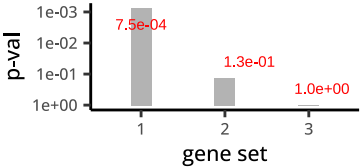
Which one is significantly overlapping with $k = 8$ genes?



Measure the tail probability (p-value):

```
mm <- c(9, 7, 11)          # |gene set|
xx <- c(6, 3, 0)           # |overlap|
N <- 24                    # N. total
pval <- phyper(xx,         # overlap
               m=mm,       # inside
               n=(N-mm),   # outside
               k=8,        # significant
               lower.tail=FALSE)
```

Which one is significant?



Summary of the gene set analysis by hypergeometric test

When will it work?

- A routine to construct a set of differentially expressed genes by handling multiple hypothesis testing
- Gene sets are of similar sizes and *nearly* disjoint/independent from one another
- Genes are *nearly* independent (there is no overwhelmingly favourite genes)

When will it fail?

- **Don't** have a good way to make a set:
  - Our discovery data may lack statistical power, i.e., no (or a few) significant genes left after multiple hypothesis correction
- There is a hidden factor that can affect two steps: (1) gene set selection (annotations/knowledge) and (2) differential expression calling

Complete `seminar-07` to better understand GSA

- We will take a look at real-world gene sets–MSigDB pathways and NHGRI-EBI GWAS catalogue.
- We will also discuss `goseq` method.
- Gene ontology analysis for RNA-seq: accounting for selection bias.
- We will also demonstrate one kind of gene-level bias induced by gene length.

Rank-based Gene Set Enrichment Analysis method (Subramanian *et al.* 2005)

GSEA input

- A collection of gene-sets: $\mathcal{C}_1, \ldots, \mathcal{C}_K$
- A vector of gene-level scores (G genes): $z_1, \ldots, z_G$
- Each z_g could come from differential expression analysis

GSEA algorithm

- For each k, compute a set-level score $S_k(\mathbf{z}, \mathcal{C}_k)$
- E.g., Kolmogorov-Smirnov statistic comparing the distribution $\{z_g : g \in \mathcal{C}_k\}$ vs. $\{z_g : g \notin \mathcal{C}_k\}$

Gene Set Enrichment Analysis method (Subramanian *et al.* 2005)

- Construct the null distribution of $S_1, \ldots, S_K$ by sample label (case-control) or gene-to-set membership permutation
- Using the null distribution by permutation, estimate p-values and false discovery rates
- If we knew the null distribution, we would not need expensive permutations.

Good:

- No cutoff/assumptions needed to estimate the null distribution
- Aggregate scores across many genes! (boost the power)

Bad:

- What is an appropriate statistic?
- What should be permuted? For how long?

Let's simulate some gene set data to understand GSEA

Simulation (Efron and Tibshirani 2007)

1. Generate basal gene expression

$$X_{i,g} \sim \mathcal{N}(0, 1)$$

2. Sample case vs. control membership (the rows of X) uniformly at random
3. Sample membership gene to gene set uniformly at random
4. For the first gene set, select a certain fraction of genes to perturb
5. For the selected genes g^*, add some Δ value to X_{i,g^*} if the sample i belongs to the control group

Let's simulate some gene set data (Efron and Tibshirani 2007)

```
simulate.data <-
    function(G = 1000,          # genes
             K = 150,           # gene sets
             n.samp = 100,      # sample size
             delta = .4,        # perturbation
             p.perturb = 1) { # Pr of petrub
  case.control <- sample(0:1, n.samp, TRUE)
  S <- sample(K, G, TRUE)        # gene sets
  X <- .rnorm(n.samp, G)         # All the other genes
  ## Perturbation of the first gene set
  .genes.1 <- which(S == 1)
  n1 <- length(.genes.1)
  n.perturb <- max(floor(n1 * p.perturb),1)
  .genes.1 <- sample(.genes.1, n.perturb)
  .case <- case.control == 1
  X[.case, .genes.1] <- X[.case, .genes.1] + delta
  require(Matrix)
  .membership <- sparseMatrix(j=1:G, i=S, x=rep(1,G))
  list(X=X, S=.membership, Y=case.control)
}
```

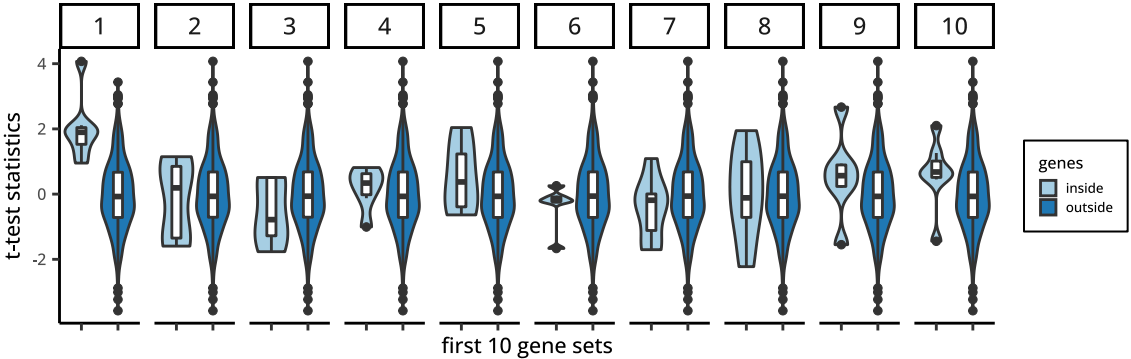
```
set.seed(1)
dat <- simulate.data()
```

Let's run t-test for each gene:

```
run.t.test <- function(X, Y){
  .case <- Y == 1
  .ctrl <- Y == 0
  .fun <- function(x){
    t.test(x[.case],
           x[.ctrl])$statistic
  }
  apply(X, 2, .fun)
}
```

What will be a proper gene set score?

Gene-level scores across gene sets:



What will be a proper gene set score?

Can we simply aggregate gene-level z-scores (or t-statistics) within each set?

Irizarry *et al.* (2009), using Stouffer Z-score

$$S_k = \sum_{g \in \mathcal{C}_k} z_g / \sqrt{|\mathcal{C}_k|} \sim \mathcal{N}(0,1)$$

if $Z_g \sim \mathcal{N}(0,1)$, $\forall g$

```
geneset.score <- function(X, S, Y) {
    z.genes <- run.t.test(X, Y)
    n.sets <- apply(S, 1, sum)
    z.sets <- (S %*% z.genes /
               sqrt(n.sets))
}
```

```
z.sets <- geneset.score(dat$X, dat$S, dat$Y)
```

What will be a proper gene set score?

Can we simply aggregate gene-level z-scores (or t-statistics) within each set?

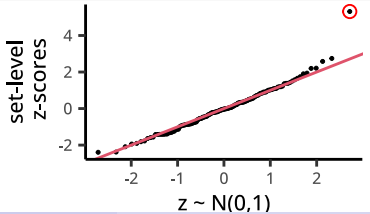
Irizarry *et al.* (2009), using Stouffer Z-score

$$S_k = \sum_{g \in \mathcal{C}_k} z_g / \sqrt{|\mathcal{C}_k|} \sim \mathcal{N}(0,1)$$

if $Z_g \sim \mathcal{N}(0,1)$, $\forall g$

```
geneset.score <- function(X, S, Y) {
    z.genes <- run.t.test(X, Y)
    n.sets <- apply(S, 1, sum)
    z.sets <- (S %*% z.genes /
               sqrt(n.sets))
}
```

```
z.sets <- geneset.score(dat$X, dat$S, dat$Y)
```



What will be a proper gene set score?

Can we simply aggregate gene-level z-scores (or t-statistics) within each set?

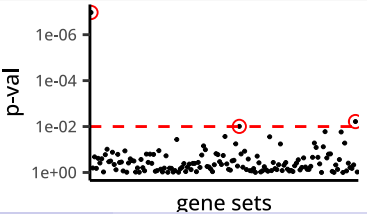
Irizarry *et al.* (2009), using Stouffer Z-score

$$S_k = \sum_{g \in \mathcal{C}_k} z_g / \sqrt{|\mathcal{C}_k|} \sim \mathcal{N}(0,1)$$

if $Z_g \sim \mathcal{N}(0,1)$, $\forall g$

```
geneset.score <- function(X, S, Y) {
    z.genes <- run.t.test(X, Y)
    n.sets <- apply(S, 1, sum)
    z.sets <- (S %*% z.genes /
               sqrt(n.sets))
}
```

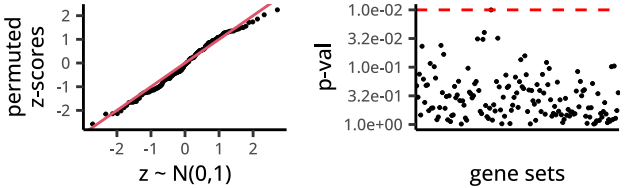
```
z.sets <- geneset.score(dat$X, dat$S, dat$Y)
```



Constructing the null distribution by gene permutation

What if we don't know the distribution of set-wise scores?

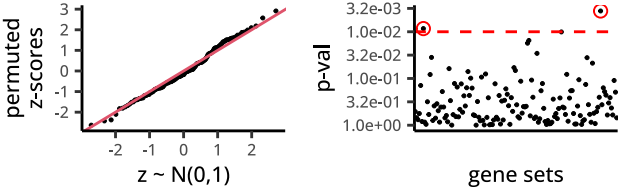
```
S.perm <- t(apply(dat$S, 1, sample))
z.perm <- geneset.score(dat$X, S.perm, dat$Y)
```



- Repeat the permutation of gene set membership matrix while preserving the number of genes within each set
- Compute set-level z-scores (or a similar kind) and construct the null distribution
- Calculate p-values by counting the frequency of observed $S_k^* > S_k^{\text{perm}}$

Constructing the null distribution by sample permutation

```
Y.perm <- sample(dat$Y)
z.perm <- geneset.score(dat$X, S.perm, Y.perm)
```



- Repeat the permutation of case-control labels while preserving the same number of cases and controls
- Compute set-level z-scores (or a similar kind) and construct the null distribution
- Calculate p-values by counting the frequency of observed $S_k^* > S_k^{\text{perm}}$

Again, complete `seminar-07` to better understand GSEA

- When we have little confidence on the null distribution of gene-set scores, we usually rely on many, many steps of permutations
- A naive permutation scheme is highly inefficient, so we will use fast GSEA (`fgsea`) and approximate p-value calculation
- Check out this preprint: https://www.biorxiv.org/content/10.1101/060012v3

What you will learn

1. Gene Set Enrichment Analysis
2. Biological network analysis
3. Learning structures in biological networks

What is network? Some graph theory terms

- Network is a graph, which is a set of sets
- We define a graph as a tuple $G = (V, E)$ if undirected:
  - V (or $V(G)$): a set of vertices, $V = \{v_1, \dots, v_n\}$
  - E (or $E(G)$): a set of edges, $E = \{(v_i, v_j) : \text{if vertices } i, j \text{ are conected}\}$
- $n(G) = |V(G)|$ number of the vertices in G, the size of the set V (or simply n)
- $m(G) = |E(G)|$ number of the edges (connected pairs) in G, the size of the set E (or simply m)
- $d_v(G) =$ degree of a vertex v's neighbours in G (# how many friends?)

We can use igraph in R

```
library(igraph)
```

- We can use igraph in R or Python or C++.

Let's take a look at some classic examples

Download some from Mark Newman's website:

```
#' @param .src www location
#' @param .dir downloading local directory
#' @param .zip zip file name
#' @param .tgt target gml file name
.read.online <- function(.src,
                         .dir="Data/",
                         .zip = str_c(.dir,basename(.src)),
                         .tgt = str_replace(.zip, "zip$", "gml")){
    if(!file.exists(.tgt)){
        .gml <- basename(.tgt)                    # remove the dir name
        download.file(.src, .zip)                 # download from the web
        unzip(.zip, files=.gml, exdir = .dir)     # unzip
    }
    return(igraph::read_graph(.tgt, format="gml")) # read them into memory
}
```

```
karate <- .read.online("http://www-personal.umich.edu/~mejn/netdata/karate.zip")
polblog <- .read.online("http://www-personal.umich.edu/~mejn/netdata/polblogs.zip")
netsci <- .read.online("http://www-personal.umich.edu/~mejn/netdata/netscience.zip")
football <- .read.online("http://www-personal.umich.edu/~mejn/netdata/football.zip")
condmat <- .read.online("http://www-personal.umich.edu/~mejn/netdata/cond-mat-2005.zip")
```

Handy functions for visualizing a network

We want to match graph vertices with the layout coordinates found by `igraph::layout_nicely` and make it easy to show in `ggplot` or something else.

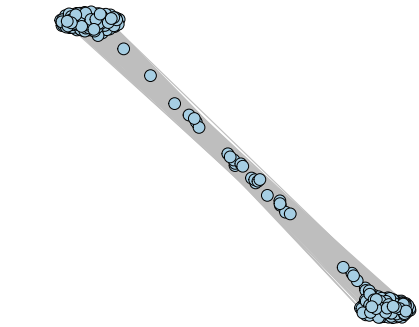
```
#' @param G igraph object
#' @param layout.file local file
find.nice.layout <-
    function(G, layout.file) {

    if(!file.exists(layout.file)){
        layout.dt <-
            layout_nicely(G) %>%
            as.data.table()
        saveRDS(layout.dt, layout.file)
    }
    readRDS(layout.file)
}
```

```
#' @param G igraph object
#' @param .name file header
apply.layout.dt <- function(G, .name){
  .out.file <-
    str_c(fig.dir,"/",.name,".rds")

  .out <-
    find.nice.layout(G, .out.file) %>%
    dplyr::mutate(v = 1:dplyr::n()) # index

  as_long_data_frame(G) %>%
    as.data.table %>%
    merge(.out, by.x= "from", by.y="v") %>%
    merge(.out, by.x= "to", by.y="v",
        suffixes=c(".from",".to"))
}
```

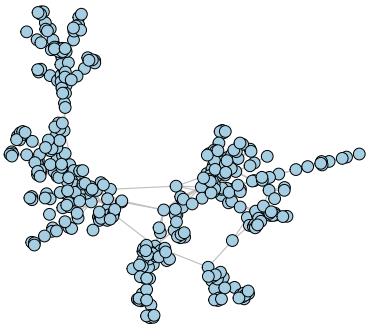
Plotting a graph with points and segments in ggplot

```
plot.layout.dt <- function(.dt, .palette="Paired", .palette.e = "Set2", .esize = .5, .vsize = 5, .add.lab=TRUE){
  if(!("k.from" %in% colnames(.dt))) .dt[, k.from := 1]                    # add membership
  if(!("k.to" %in% colnames(.dt))) .dt[, k.to := 1]                        # add membership
  .rename <- function(xx) { colnames(xx) <- c("v","V1","V2","k"); xx }      # Find a consolidated
  .dt.v <-                                                                 # set of vertices
    rbind(.dt[, .(`from`, `V1.from`, `V2.from`, `k.from`)] %>% .rename(),
        .dt[, .(`to`, `V1.to`, `V2.to`, `k.to`)] %>% .rename()) %>%
    dplyr::mutate(k = as.factor(k)) %>% unique

  .aes.e <- aes(V1.from, V2.from, xend=V1.to, yend=V2.to)   # edge colours
  if("k.edge" %in% colnames(.dt)) {
    .aes.e <- aes(V1.from, V2.from, xend=V1.to, yend=V2.to, colour=as.factor(k.edge))
  }
  plt <- ggplot() + theme_void()                                           # show no axis
  if("k.edge" %in% colnames(.dt)) {
    plt <- plt +
      geom_segment(.aes.e, data = .dt, size = .esize) +                    # draw edges
      scale_colour_brewer(palette=.palette.e, na.value="grey", guide="none")
  } else {
    plt <- plt +
      geom_segment(.aes.e, data = .dt, size = .esize, colour="gray")       # draw edges
  }
  plt <- plt +
    geom_point(aes(V1, V2, fill=k), data = .dt.v, pch=21, stroke=.1, size = .vsize) + # vertices
    scale_fill_brewer(palette = .palette, guide="none")                    # vertex colouring

  if(.add.lab){ plt <- plt + geom_text(aes(V1, V2, label=v), data = .dt.v, size = 3) }
  return(plt)
}
```

Social network examples

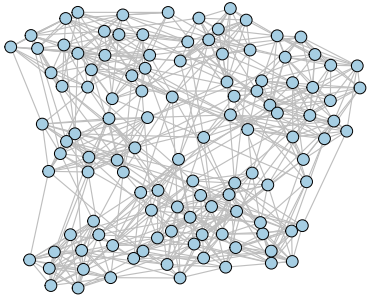


- L. A. Adamic and N. Glance, "The political blogosphere and the 2004 US Election", in Proceedings of the WWW-2005 Workshop on the Weblogging Ecosystem (2005).

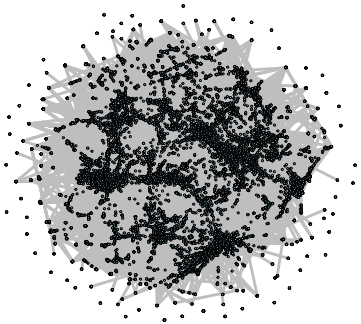


- Coauthorship network of scientists working on network theory and experiment
- Showing only the largest component
- M. Newman in May 2006.

More examples



- Network of American football games between Division IA colleges during regular season Fall 2000.
- M. Girvan and M. E. J. Newman, Proc. Natl. Acad. Sci. USA 99, 7821-7826 (2002).



- Co-authorships between scientists posting preprints on the Condensed Matter E-Print Archive. M. Newman, PNAS (2001).
- Showing densely-connected part (iterative degree cutoff $>= 5$)

Biological network: Human Reference Protein-Protein interaction map

- Download the data from `interactome-atlas.org`

```
db.url <- "http://www.interactome-atlas.org"
ppi.net.file <- "Data/HI-union.tsv.gz"
if(!file.exists(ppi.net.file)){
    .file <- str_remove(ppi.net.file, "[.]gz$")
    .url <- str_c(db.url,"/data/HI-union.tsv")
    download.file(.url, destfile = .file)
    R.utils::gzip(.file)
}
```

- Build a network from a list of interacting pairs

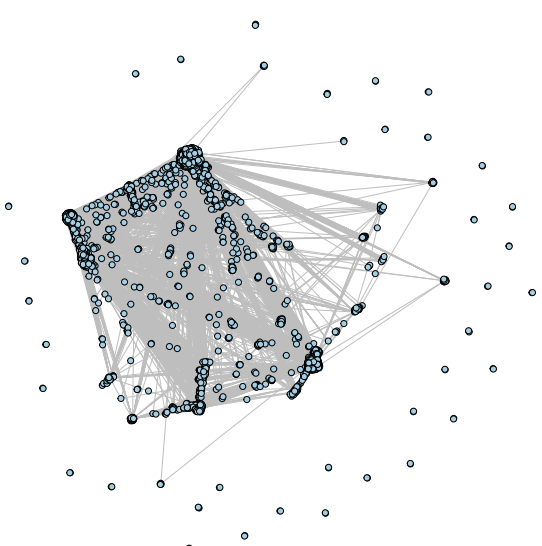
```
ppi.pairs <-
  fread(ppi.net.file, header=FALSE, col.names=c("v1","v2")) %>%
  filter(v1 != v2) %>% as.matrix
G.ppi <- graph_from_edgelist(ppi.pairs, directed = FALSE)
```

- Vertices: genes (or the resulting proteins)
- Edges: two or multiple genes work together (could have many different meanings)

```
n <- length(V(G.ppi))
m <- length(E(G.ppi))
```

- n = 9060 vertices
- m = 63242 edges

Biological network: Human Reference Protein-Protein interaction map



- Vertices: genes (or the resulting proteins)
- Edges: two or multiple genes work together (could have many different meanings)

```
n <- length(V(G.ppi))
m <- length(E(G.ppi))
```

- n = 9060 vertices
- m = 63242 edges

Sources of biological networks:

- High-throughput experiments
- Co-expression analysis
- Literature-based curation

Degree distribution (How many friends?)

Vertex degree

For each vertex $v \in V(G)$: $d_v = \sum_u A_{uv}$

- It is simple to calculate in `igraph`:

```
deg.dt <- degree(G.ppi) %>%
    (function(.d)
        data.table(d=.d, v = names(.d))
    )
```

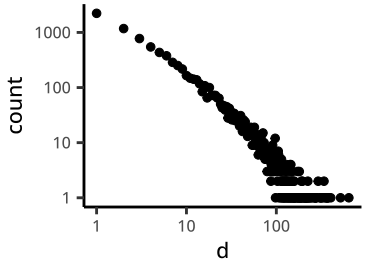
- Degree distributions may teach us the underlying generative scheme
- Power-law (almost linear slope in a log-log plot) may indicate a "rich-get-richer" phenomenon.
- The majority with < 10 neighbours vs. hub nodes with > 100 neighbours
- Poisson distribution if edges are drawn uniformly at random

Degree distribution (How many friends?)

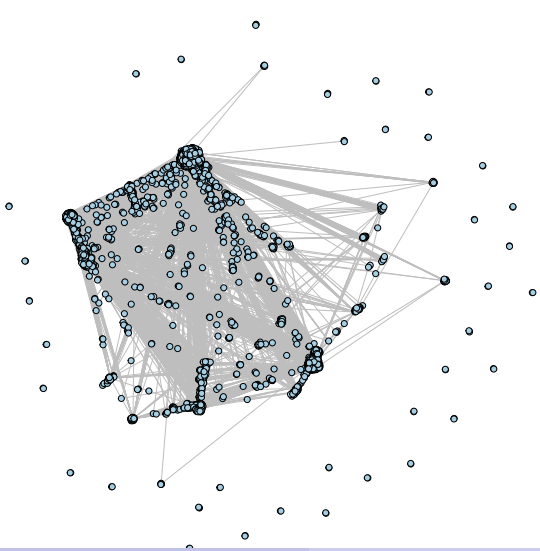
A handy one-liner:

```
.dt <- deg.dt[, .(count=.N), by=.(d)]
```

Log-log plot:



Connected components and a giant component



A (connected) component = a set of vertices reachable by hopping through edges.

```
.comp <- components(G.ppi)
```

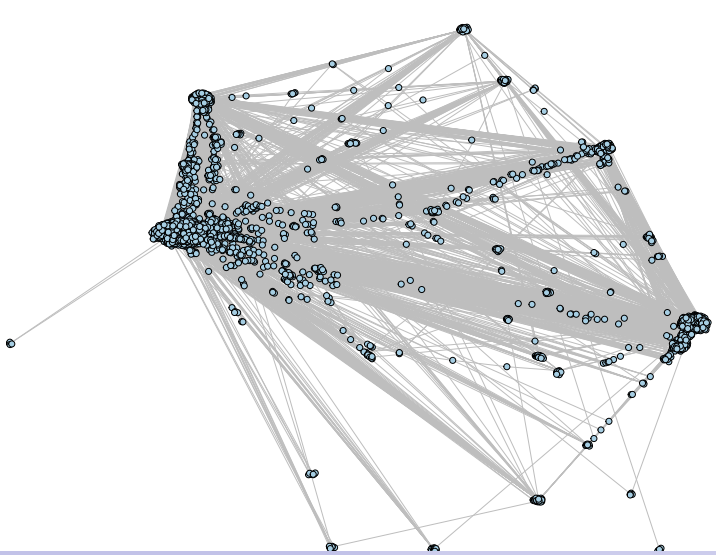
Found 36 connected components

n	count
8986	1
2	31
3	4

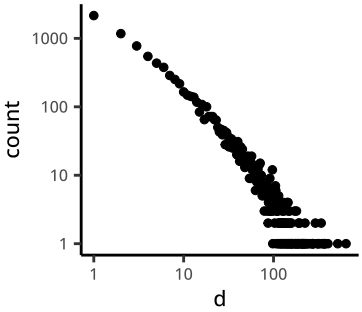
- Can we take the giant one?

```
k <- which.max(.comp$csize)
.sub <- names(.comp$membership)[
    .comp$membership==k]
G.ppi.sub <- induced_subgraph(G.ppi, .sub)
```

A subgraph induced by genes/proteins in the giant component



- $n(G)$: 8986 (vs. 9060)
- $m(G)$: 63203 (vs. 63242)
- Degree distribution:



Iterative degree cutoff to remove dangling vertices

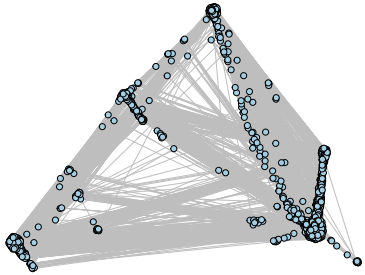
We may consider this as a Q/C step:

```
degree.cutoff <- function(G, .cutoff = 3) {
    G.sub <- G
    n.remove <- sum(degree(G.sub) < .cutoff)
    while(n.remove > 0){
        vv <- V(G.sub)
        .retain <- vv[degree(G.sub) >= .cutoff]
        G.sub <- induced_subgraph(G.sub, .retain)
        n.remove <- sum(degree(G.sub) < .cutoff)
    }
    return(G.sub)
}
G.dc <- degree.cutoff(G.ppi.sub, 3)
```

Comparison with the raw PPI data:

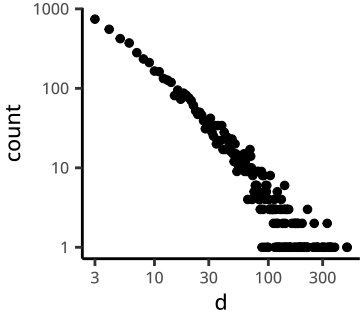
- $n(G)$: 5525 (vs. 9060)
- $m(G)$: 58548 (vs. 63242)

Will it look nicer?



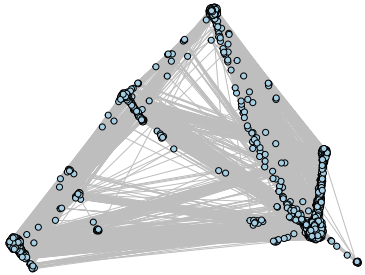
Iterative degree cutoff to remove dangling vertices

Degree distribution:



- $n(G)$: 5525 (vs. 9060)
- $m(G)$: 58548 (vs. 63242)
- Note: d starts from 3

Will it look nicer?



- Rationale: no genes act alone. If some genes have significantly fewer edges connected, high-throughput experiments had not covered them sufficiently.

Let's use this PPI network to understand disease mechanisms

We can download public data mapping SNPs (genes) to diseases/phenotypes from the NHGRI-EBI GWAS Catalog

- GWAS: genome-wide association study (will cover later)
- Again, `seminar-07` also covers GWAS catalogue.

Let's retrieve a list of Multiple Sclerosis (MS)-related genes

```
.gwas.dt <- fread(.gwas.catalog.file, sep="\t", quote="")
is.ms.trait <- function(x) str_detect(x, "[Mm]ultiple sclerosis")
ms.genes <-
 .gwas.dt[is.ms.trait(`DISEASE/TRAIT`),] %>%
 filter(str_length(`MAPPED_GENE`) > 2) %>%
 select(`MAPPED_GENE`) %>% na.omit %>% unlist %>%
 str_split(pattern="[ ,;]+" ) %>% unlist %>% unique
```

Gene prioritization: a toy example of multiple sclerosis GWAS

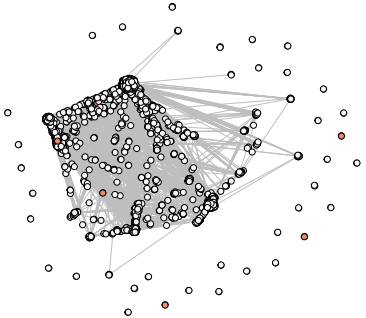
How many of them overlap with the vertices in the PPI network?

```
vtot <- V(G.ppi)
v.dt <- data.table(v=vtot,ensembl_gene_id=names(vtot))
.dt <- merge(v.dt, gene.info)
overlap.dt <-
  .dt[hgnc_symbol %in% ms.genes,
      .(v, ensembl_gene_id, hgnc_symbol)
    ] %>% unique
```

- 194 genes (of 588 MS genes) recapitulated in the network
- That is a small fraction compared to the total 9060 genes in the network
- What if no mutations occurred in other relevant genes? No mutation $\rightarrow$ no association.

Mapping disease genes onto a gene-gene interaction network

Revisiting the gene-gene (protein-protein) interaction network

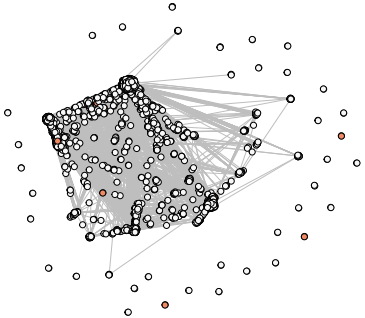


- Where are my GWAS genes in the network?

- Genes immediately hit by GWAS are probably *not* comprehensive enough.
- It seems like that we can extend the set by propagating across the edges

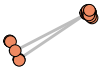
Mapping disease genes onto a gene-gene interaction network

Revisiting the gene-gene (protein-protein) interaction network



- Where are my GWAS genes in the network?

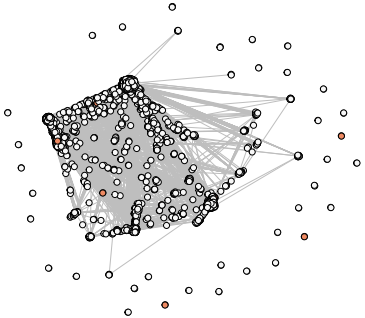
194 MS GWAS genes (2%)



- Genes immediately hit by GWAS are probably *not* comprehensive enough.
- It seems like that we can extend the set by propagating across the edges

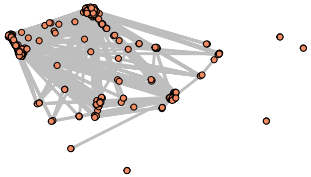
Mapping disease genes onto a gene-gene interaction network

Revisiting the gene-gene (protein-protein) interaction network



- Where are my GWAS genes in the network?

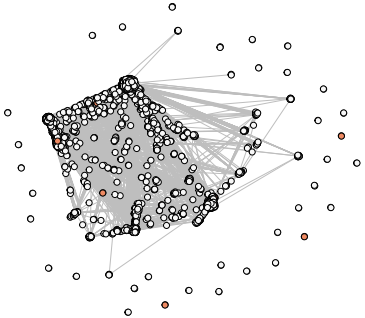
1841, MS genes+ immediate neighbours (20%)



- Genes immediately hit by GWAS are probably *not* comprehensive enough.
- It seems like that we can extend the set by propagating across the edges

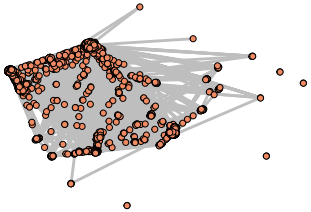
Mapping disease genes onto a gene-gene interaction network

Revisiting the gene-gene (protein-protein) interaction network



- Where are my GWAS genes in the network?

7779 MS genes
neighbours in two hops
(86%)



- Genes immediately hit by GWAS are probably *not* comprehensive enough.
- It seems like that we can extend the set by propagating across the edges

Can we prioritize disease-relevant genes using PPI network?

- GWAS might cover too few candidate genes...
- Naively expanding the set by all the neighbourhood would be too many...
- Not all the neighbours are affected by the disease...
- Who are the relevant neighbours?
- **Network $\approx$ Information flow diagram**
- Spreading "labels" in the information network

Random walk probability: What are the probabilities of visiting neighbours?

A wrapper function for `igraph`'s `random_walk()`

```
.rand.walk <- function(vv, nn){
  .fun <-
    function(v)
      random_walk(G.ppi, v, nn, mode="all")
  lapply(vv, .fun) %>%
      do.call(what = c) %>%
      unique
}
```

Let's take five steps from the MS genes:

```
v0 <- overlap.dt$v # seed
v.1 <- .rand.walk(v0,1)
v.2 <- .rand.walk(v0,2)
v.3 <- .rand.walk(v0,3)
```

Some vertices are more frequently visited when we randomly traverse in the network.

n=194 / 194

Random walk probability: What are the probabilities of visiting neighbours?

A wrapper function for `igraph`'s `random_walk()`

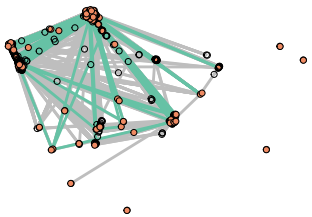
```
.rand.walk <- function(vv, nn){
  .fun <-
    function(v)
      random_walk(G.ppi, v, nn, mode="all")
  lapply(vv, .fun) %>%
      do.call(what = c) %>%
      unique
}
```

Let's take five steps from the MS genes:

```
v0 <- overlap.dt$v # seed
v.1 <- .rand.walk(v0,1)
v.2 <- .rand.walk(v0,2)
v.3 <- .rand.walk(v0,3)
```

Some vertices are more frequently visited when we randomly traverse in the network.

n=371 / 1684



Random walk probability: What are the probabilities of visiting neighbours?

A wrapper function for `igraph`'s `random_walk()`

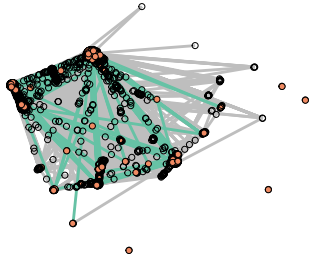
```
.rand.walk <- function(vv, nn){
  .fun <-
    function(v)
      random_walk(G.ppi, v, nn, mode="all")
  lapply(vv, .fun) %>%
      do.call(what = c) %>%
      unique
}
```

Let's take five steps from the MS genes:

```
v0 <- overlap.dt$v # seed
v.1 <- .rand.walk(v0,1)
v.2 <- .rand.walk(v0,2)
v.3 <- .rand.walk(v0,3)
```

Some vertices are more frequently visited when we randomly traverse in the network.

n=532 / 7725



What are the probabilities of visits from the disease genes?

- We can represent the network as an adjacency matrix A, where $A_{ij} = 1$ if vertices i and j are connected.
- Define a weight matrix $W_{ij} = Pr(j \to i)$ with equal probabilities, i.e., $W_{ij} \leftarrow A_{ij} / \sum_i A_{ij}$.

$$\mathbf{p}^{(t)} \leftarrow \gamma W \mathbf{p}^{(t-1)} + (1-\gamma)\mathbf{p}^{(0)}$$

- We initialize $\mathbf{p}^{(0)}$ by the disease genes, setting $p_i = 1/\#$ disease genes if a gene i is in the MS GWAS.

What are the probabilities of visits from the disease genes?

The code is actually simpler than explanation:

Initialization:

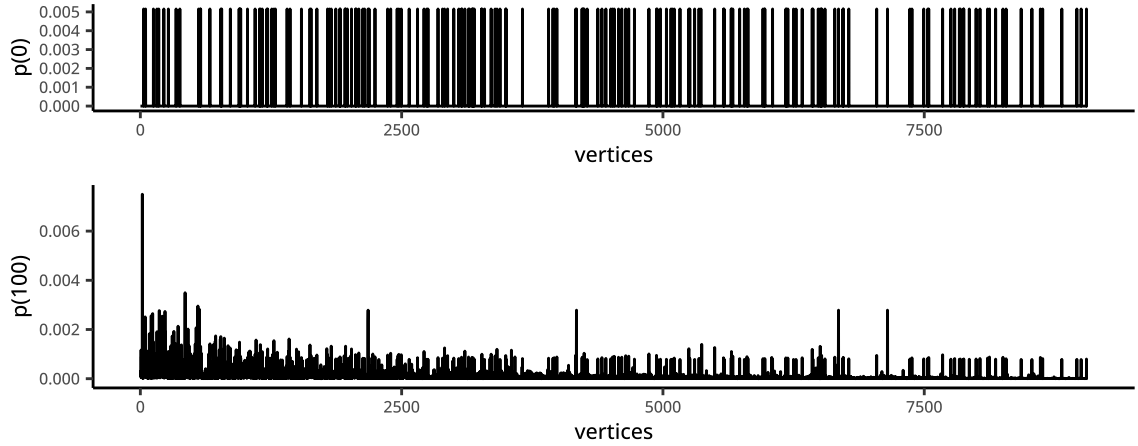
```
A <- as_adjacency_matrix(G.ppi)
d <- degree(G.ppi)
W <- sweep(A, 2, d, `/`)

.v <- as.integer(overlap.dt$v)
n <- length(V(G.ppi))
pr <- matrix(0, n, 1)
pr[.v, 1] <- 1
pr.0 <- pr / sum(pr)
```

Take 100 random walks:

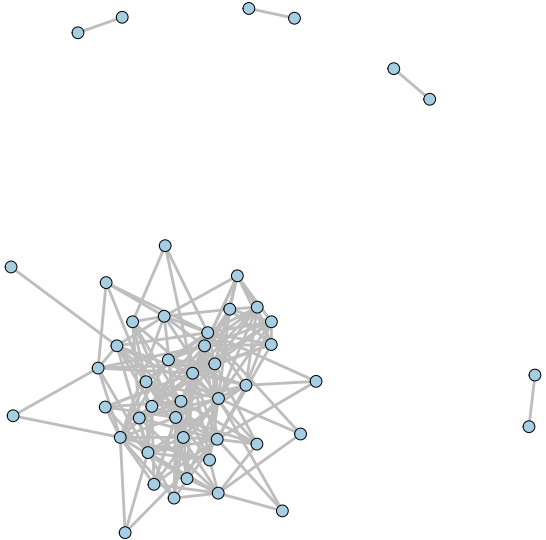
```
pr <- pr.0; r <- 0.85
P <- matrix(NA, n, 100)
P[,  1] <-  as.numeric(pr)
for(tt in 1:99) {
    pr <- r * W %*% pr + (1 - r) * pr.0
    pr <- pr / sum(pr)
    P[, tt + 1] <-  as.numeric(pr)
}
```

What are the probabilities of visits from the disease genes?

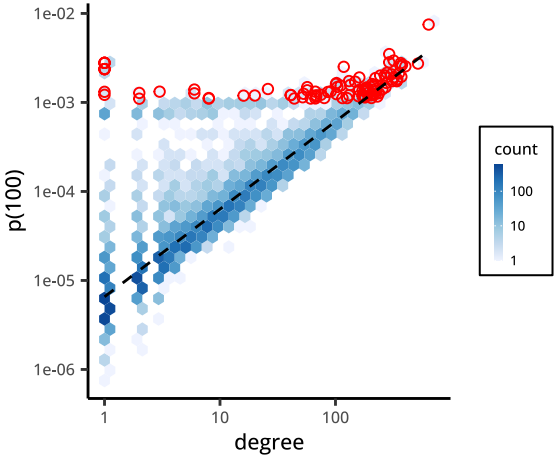
What are the probabilities of visits from the disease genes?



Top 50 genes identified by random walk from MS GWAS



Are simply we recapitulating high-degree vertices?



Red dots: top 100 genes.

What you will learn

1. Gene Set Enrichment Analysis
2. Biological network analysis
3. Learning structures in biological networks

Some useful packages in R

We can use igraph in R or Python or C++.

```
library(igraph)
```

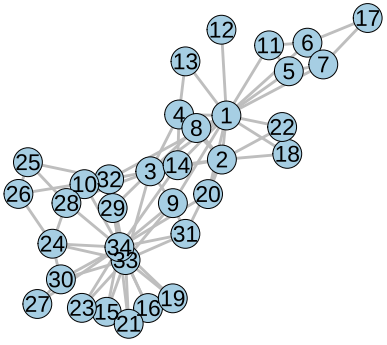
This is my own R/C++ package that can find a stochastic block model in an adjacency matrix.

```
remotes::install_github("YPARK/hsblock")
```

```
library(hsblock)
```

Community detection in Zachary's karate club (motivating example)

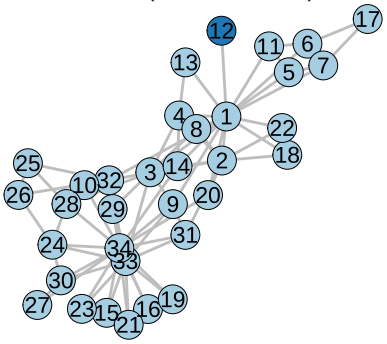
- Social network of friendships between 34 members of a karate club at a US university in the 1970s.
- W. W. Zachary, An information flow model for conflict and fission in small groups, Journal of Anthropological Research 33, 452-473 (1977).
- **Vertex**: a member in the karate club; **edge**: friendship
- How many samples?
- What is the dimensionality?



A graph-theoretic approach: Community detection by discrete optimization

- Consider two groups
- Find a set of edges to induce two disconnected components
- Not so ideal for a sparse, irregular graph

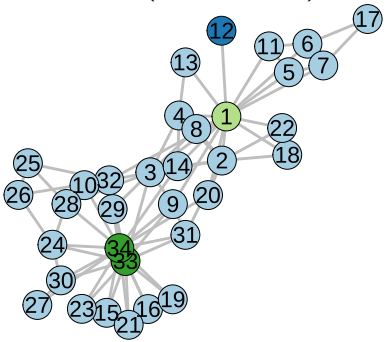
A trivial solution (min cut = 1)



Who are the “influencers” in the karate club?

- Network is not just a mathematical object
- Vertices and edges have a meaning
- Member **1**: the instructor (“Mr. Hi”), probably the founder?
- Member **34**: the president (John A)
- Member **33**: perhaps working with the present (similar interaction patterns)

A trivial solution (min cut = 1)



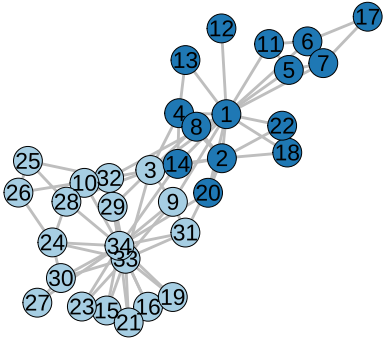
Finding min-cuts from the source to target vertices

```
.cuts <- as.directed(karate) %>%
    st_min_cuts(source = "1", target = c("33","34"))
.sets <- .cuts[["partition1s"]] %>%
    lapply(as.character)
```

- Multiple solutions with the same min-cut=5 edges.
- A partition from “1”:

```
str_c(sort(.cuts$partition1s[[1]]), collapse=", ")
```

[1] “1, 2, 4, 5, 6, 7, 8, 11, 12, 13, 14, 17, 18, 20, 22”

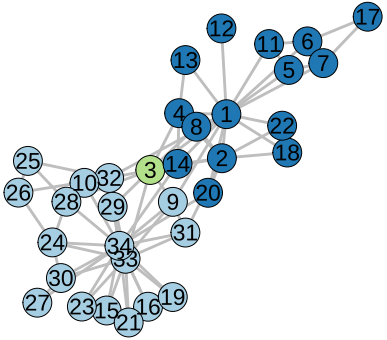


Finding min-cuts from the source to target vertices – 2

- Multiple solutions with the same min-cut=5 edges.
- A partition from "1":

```
str_c(sort(.cuts$partition1s[[2]]), collapse=", ")
```

[1] "1, 2, 3, 4, 5, 6, 7, 8, 11, 12, 13, 14, 17, 18, 20, 22"



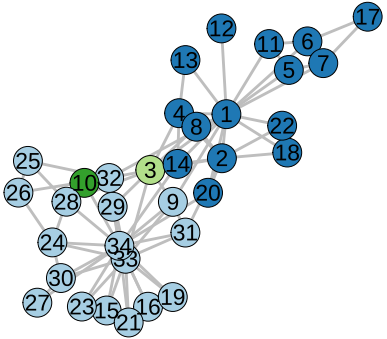
Finding min-cuts from the source to target vertices – 3

- Multiple solutions with the same min-cut=5 edges.
- A partition from "1":

```
str_c(sort(.cuts$partition1s[[3]]), collapse=", ")
```

[1] "1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 17, 18, 20, 22"

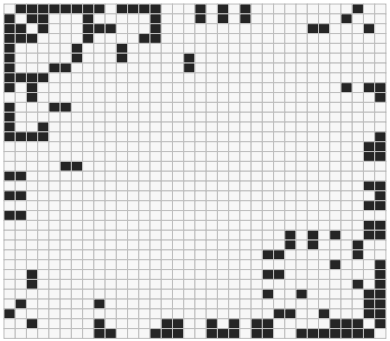
- Can we identify group/block structures without the vertex labels (e.g., source and target)?



A generative modelling approach: Stochastic block model (network is an adjacency matrix)

(vertex x vertex)

```
A <- as_adjacency_matrix(karate)
.matshow(A, .scale=FALSE, .lab = 0)
```



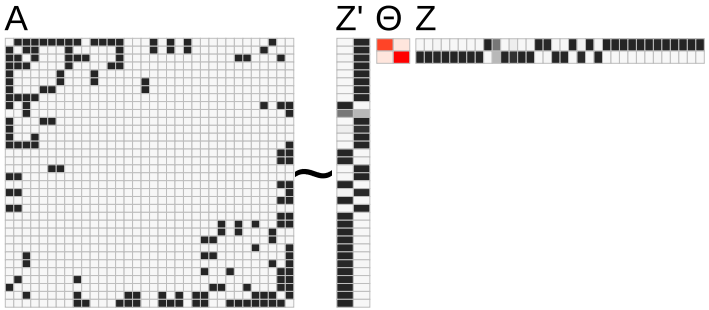
- **Row/column**: a member in the karate club
- **non-zero element**: friendship between the row and column
- How many samples? What is the dimensionality?
- Can you see group structures?
- Most of $A_{ij} = 0$. Otherwise, there is no reason to consider this matrix as a network.

Stochastic block model: membership Z and parameter Θ

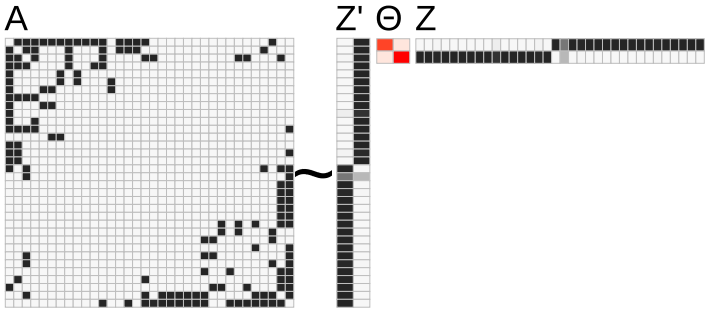
```
.out <- fit.hsblock(A, "bernoulli", vbiter=100, inner.iter=10, verbose=FALSE)
Z <- as.matrix(.out$Z)          # Cluster x vertex
n <- ncol(Z)                    # Number of vertices
Edges <- 0.5 * Z %*% A %*% t(Z) # Num of edges within/between clusters
.ones <- matrix(1,n,n)          # Total num of pairs
diag(.ones) <- 0                # without cycles
Tot <- Z %*% .ones %*% t(Z)/2   #
Pr <- Edges / Tot               # Probability of edges within/between clusters
```

- Latent var.: $Z_{ki} = 1$ iff a vertex $i \rightarrow$ a cluster k
- Model parameters: $\theta_{kk'} \in (0, 1)$: pr of edges between k and k'

Stochastic block model: membership Z and parameter Θ



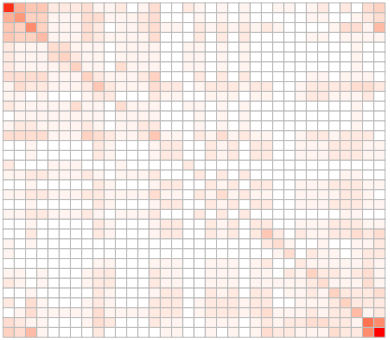
Stochastic block model: membership Z and parameter Θ



A shared neighbourhood matrix (A friend of a friend $\approx$ a friend)

- $S \leftarrow A^\top A$
- $S_{ij} = \sum_k A_{ki} A_{kj}$, number of shared neighbours
- How many samples? Dimensionality?
- Can you see group structures?

```
.matshow(A %*% A, .scale=TRUE, .lab = 0)
```



Brighter red = more frequently sharing neighbours

Network is an (edge) incidence matrix (vertex x edge)

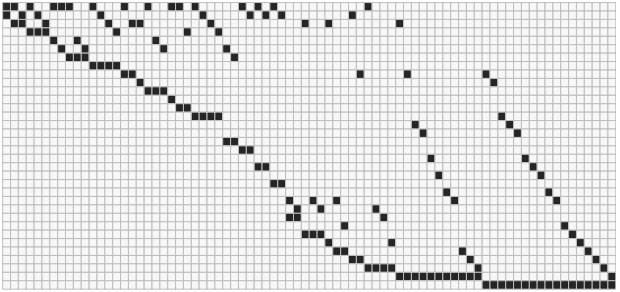
Incidence matrix

$M_{k,(i,j)} = 1$ if $A_{ik} = 1$ and $A_{jk} = 1$

```
library(Matrix)
G <- karate;
.list <- ends(G, E(G));
m <- nrow(.list)
ii <- c(.list[,1],.list[,2])
jj <- rep(1:m,2)
xx <- rep(1,2*m)
M <- sparseMatrix(ii, jj, x=xx)
```

Network is an (edge) incidence matrix (vertex x edge)

$M_{k,(i,j)} = 1$ if $A_{ik} = 1$ and $A_{jk} = 1$



- How many non-zero elements for each column?
- How many samples? Dimensionality? Can you see group structures?

Enumerating shared neighbours for each edge

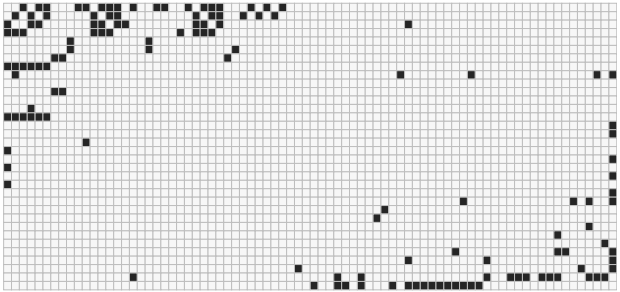
Shared neighbour incidence matrix

$$M_{k,(i,j)} = A_{ki}A_{kj}$$

```
ee <- ends(G, E(G));
ii <- ee[,1]
jj <- ee[,2]
M2 <- A[,ii] * A[,jj]
```

Enumerating shared neighbours for each edge

$$M_{k,(i,j)} = A_{ki}A_{kj}$$



- How many samples? Dimensionality? Can you see group structures?

Another way to define the edge incidence matrix

Feature loading matrix

$$M_{k,(i,j)} = A_{ki} + A_{kj}$$

```
ee <- ends(G, E(G));
ii <- ee[,1]
jj <- ee[,2]
M3 <- A[,ii] + A[,jj]
```

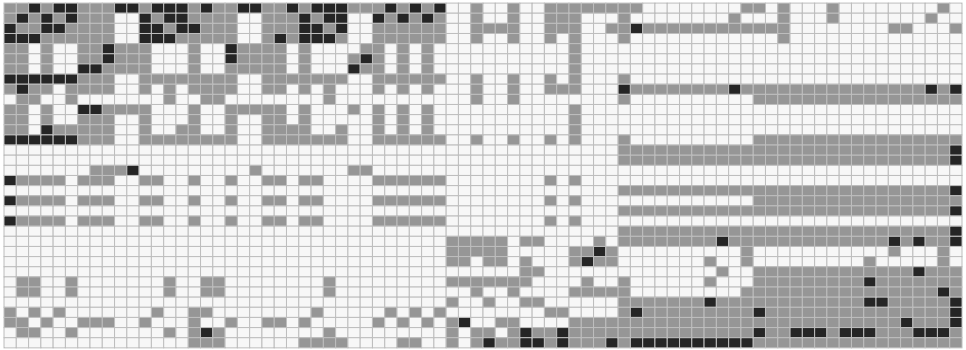
We can simply apply an out-of-the-shelf clustering method to this matrix

```
.feat <- t(as.matrix(M3))
.out <- kmeans(.feat, centers=5, nstart=100)
```

Another way to define the edge incidence matrix

$$M_{k,(i,j)} = A_{ki} + A_{kj}$$

34 x 78

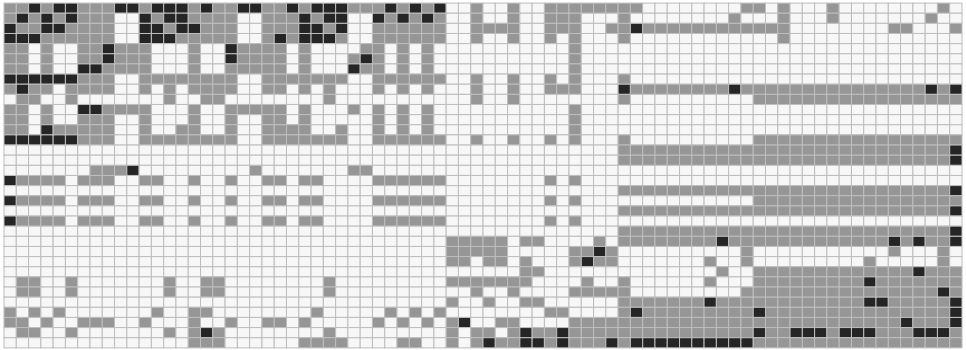


k-means clustering can recover groups of edges

Another way to define the edge incidence matrix

$$M_{k,(i,j)} = A_{ki} + A_{kj}$$

34 x 78

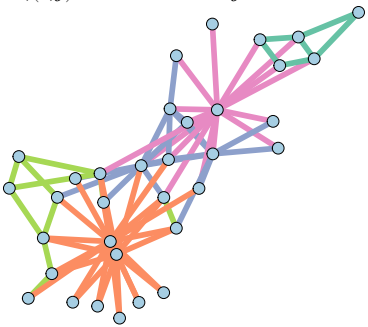


k-means clustering can recover groups of edges

5 clusters x 78 edges

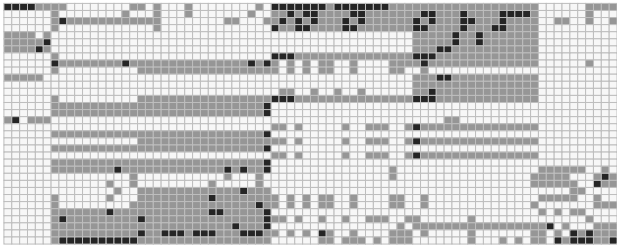
We may find overlapping membership by clustering the edges

$$M_{k,(i,j)} = A_{ki} + A_{kj}$$



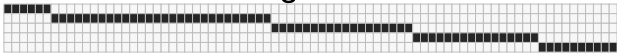
Sorted edges by the group membership:

34 x 78

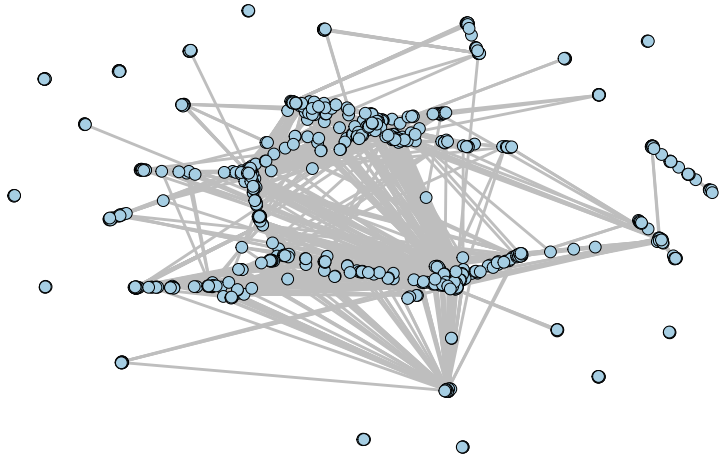


k-means clustering can recover groups of edges

5 clusters x 78 edges



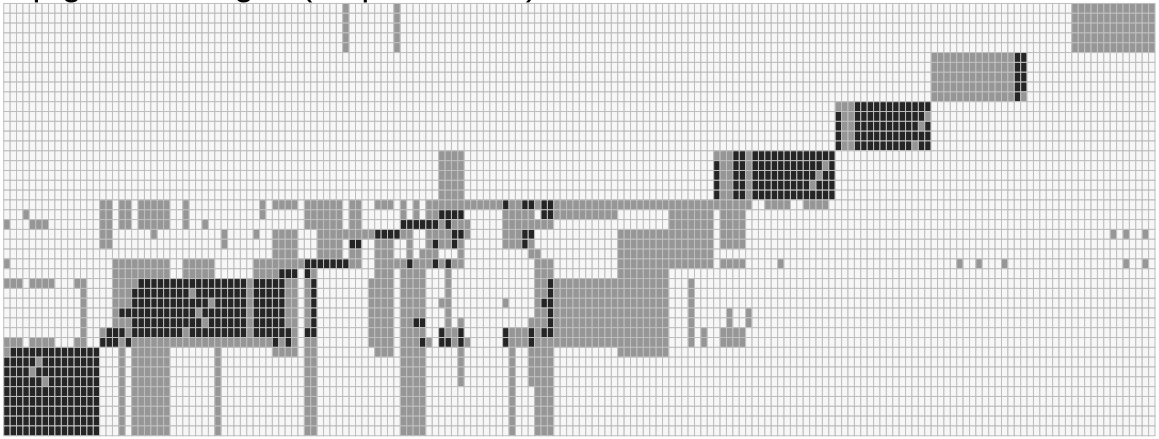
Reactome functional interaction network



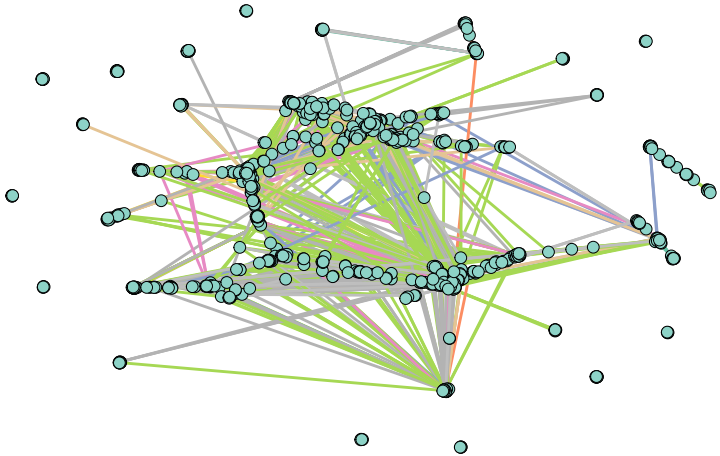
We only took edges with “reaction” for simplicity.

We can do the same edge clustering

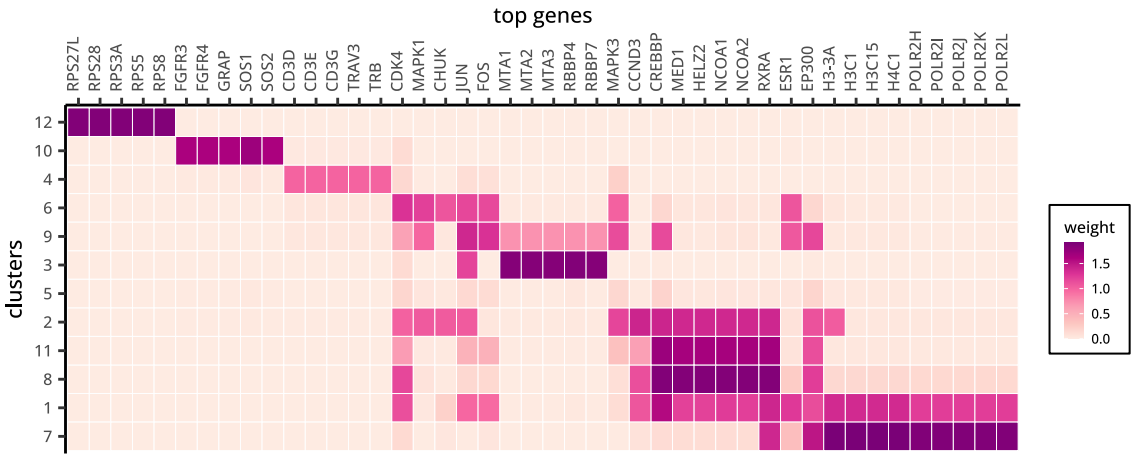
top genes x edges (15 per cluster)



A new type of gene sets that visualization could not find



A new type of gene sets that visualization could not find



Summary

- Gene set analysis (ignoring edges, only vertices)
  - `Similar-07` is your friend
- Diffusion-based network analysis (propagating info via edges)
- Stochastic block model to find sets in a network