SeqNet: an R package for generation RNA-Seq data from regulatory networks

Tyler Grimes

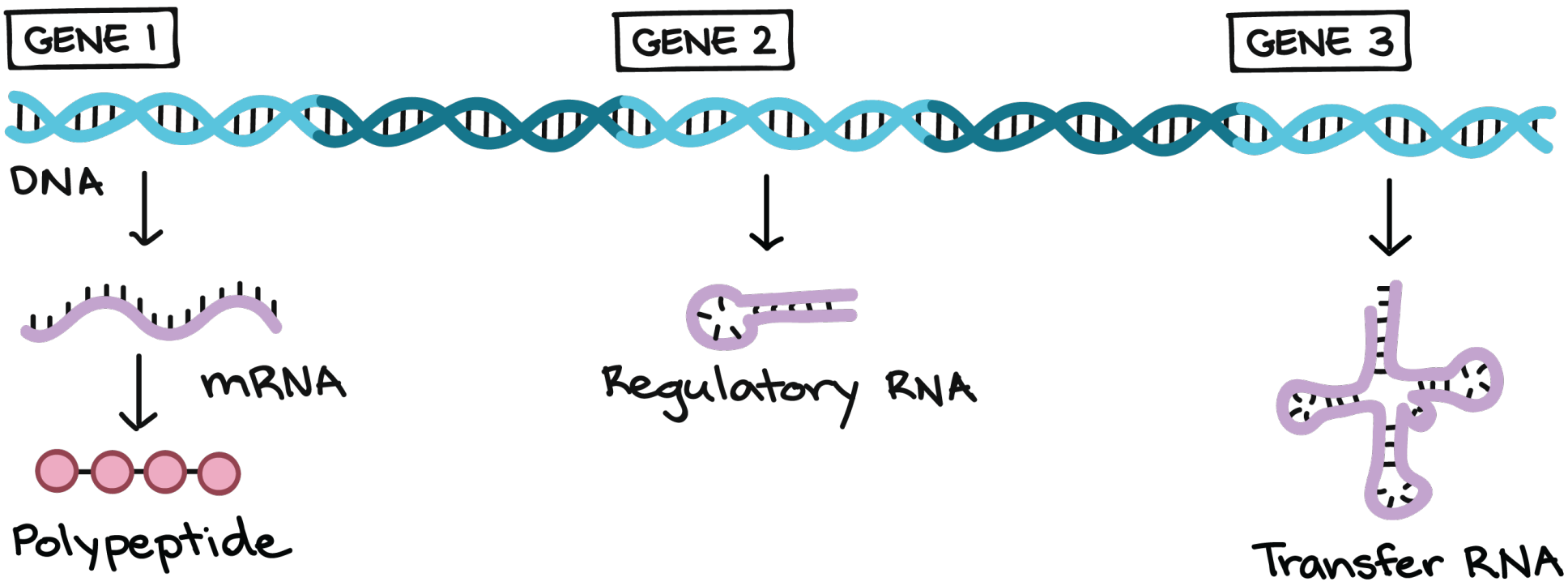
December 4, 2017

Contents

- Introduction
- Modeling
- Algorithm
- Examples

Introduction

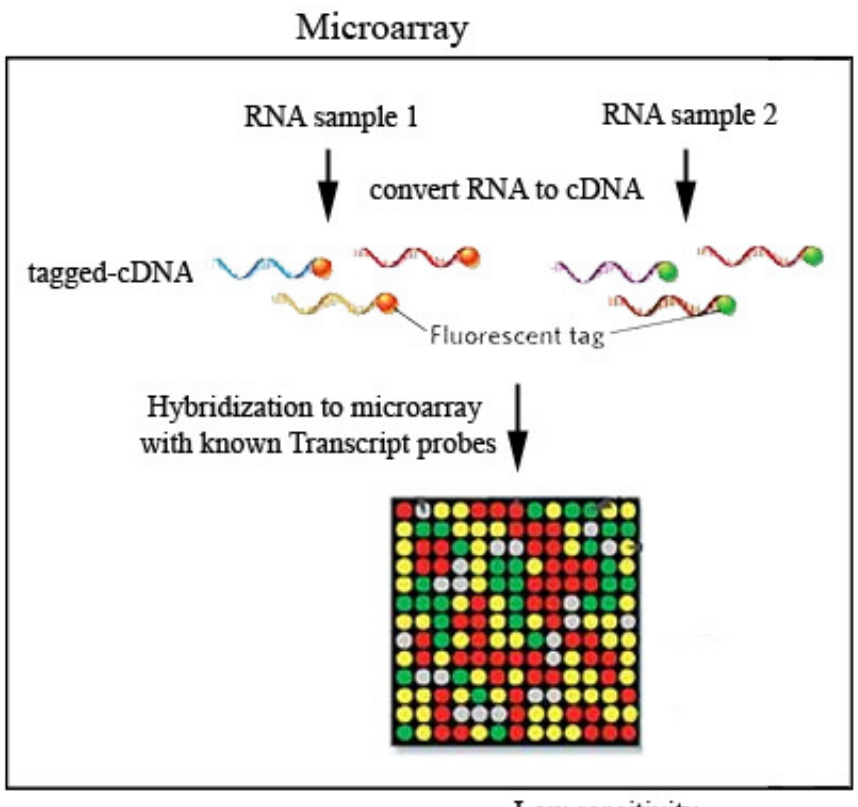
Gene expression



Gene expression and different roles of RNA.

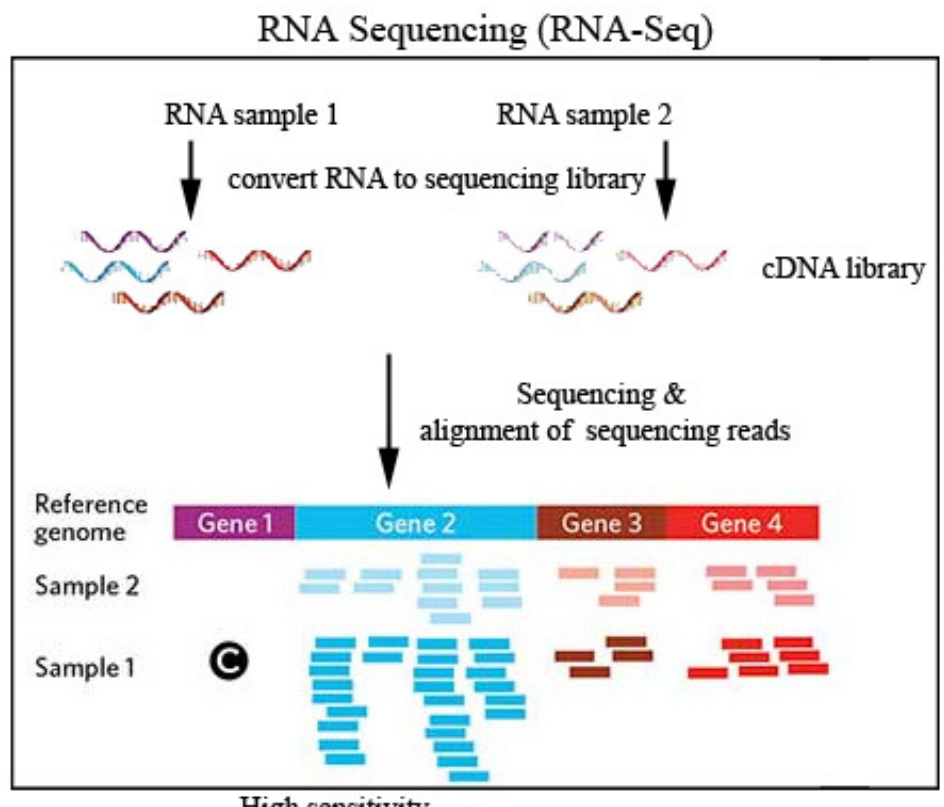
Introduction

Next-generation sequencing (RNA-Seq)



relative intensity
=
expression levels

Low sensitivity
Low dynamic range
known transcript only
No alternative splicing information
lower cost



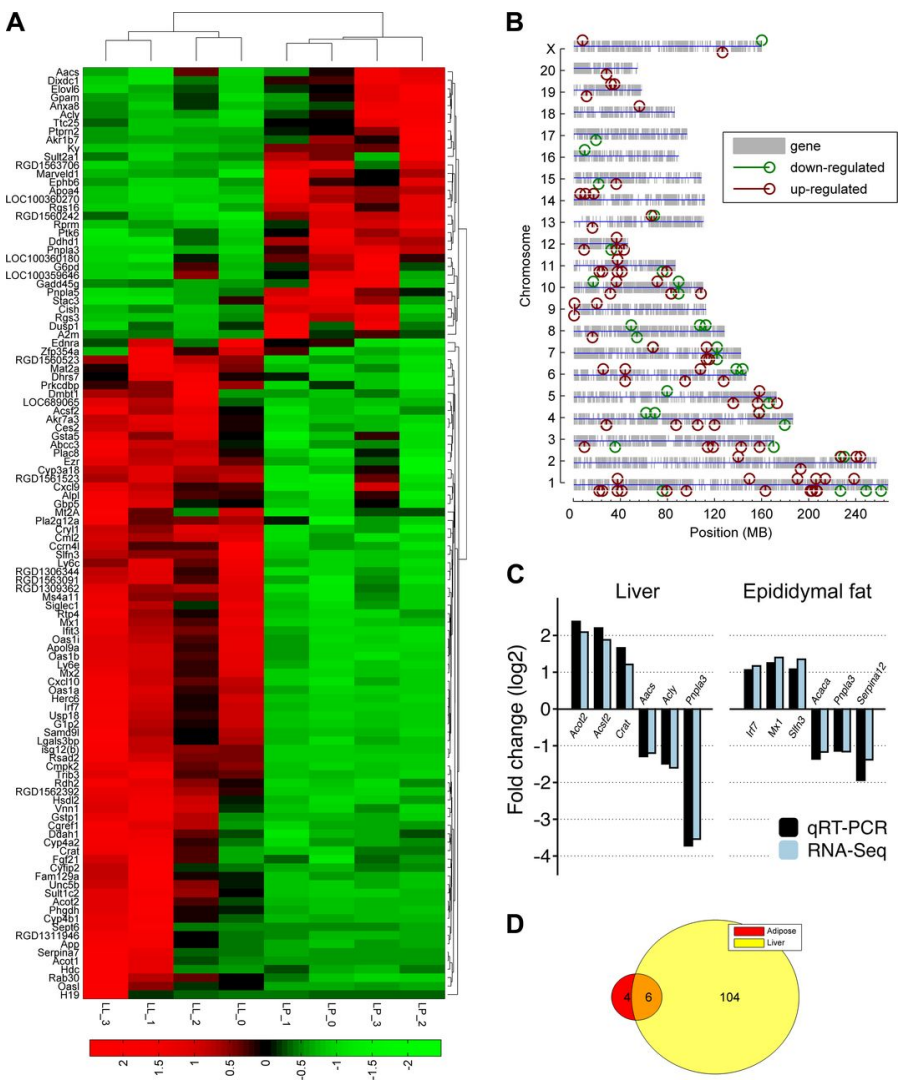
High sensitivity
High dynamic range
Novel transcripts sequences identified
structural variation & alternative splicing revealed
unlimited sample comparisons

Sequencing Reads
=
expression levels

Comparison of microarray to RNA-sequencing.

Introduction

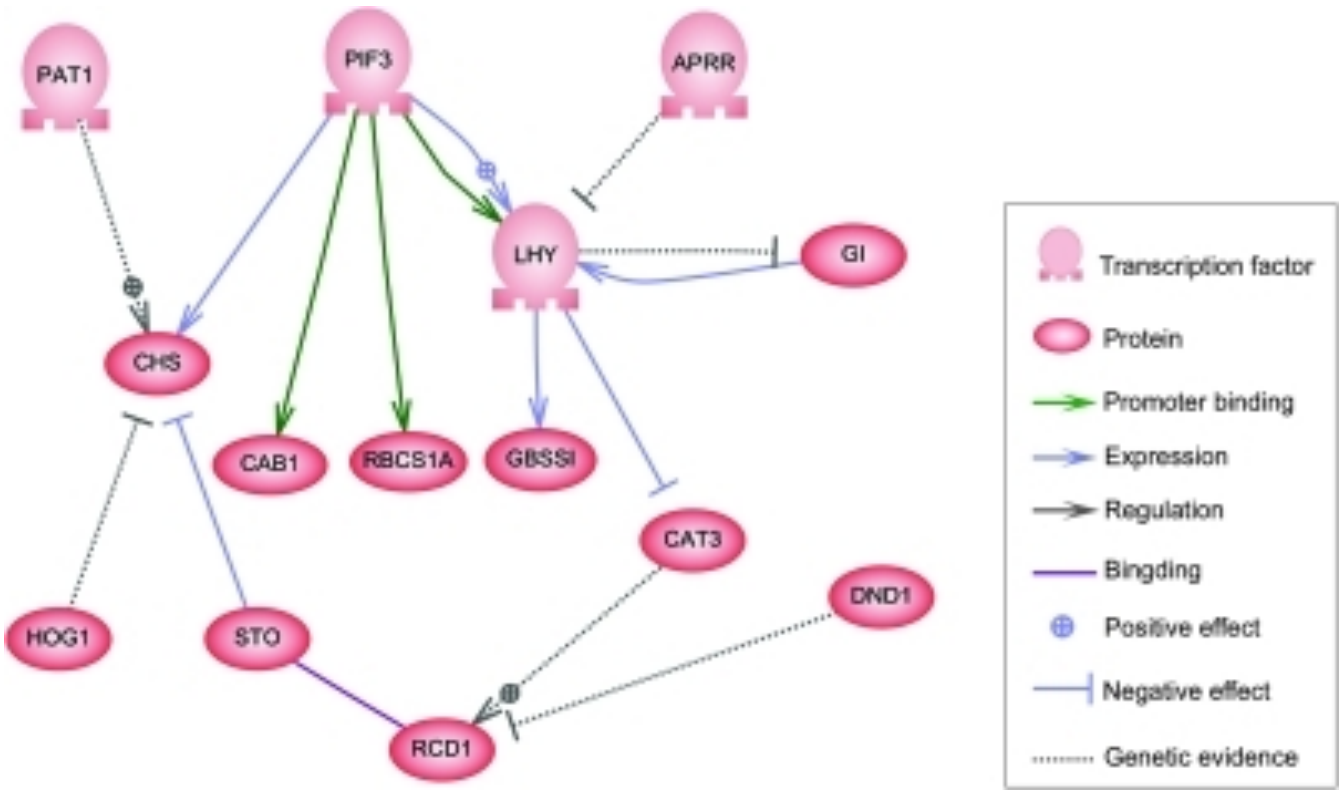
Differential expression



110 genes found to be differentially expressed between two groups ($n = 4$ in both group).

Introduction

Regulatory networks

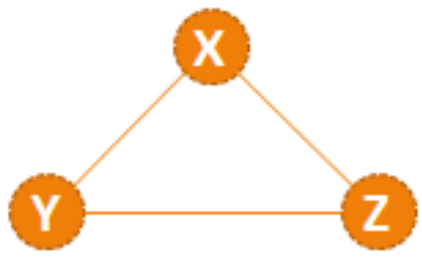


Example of a regulatory network.

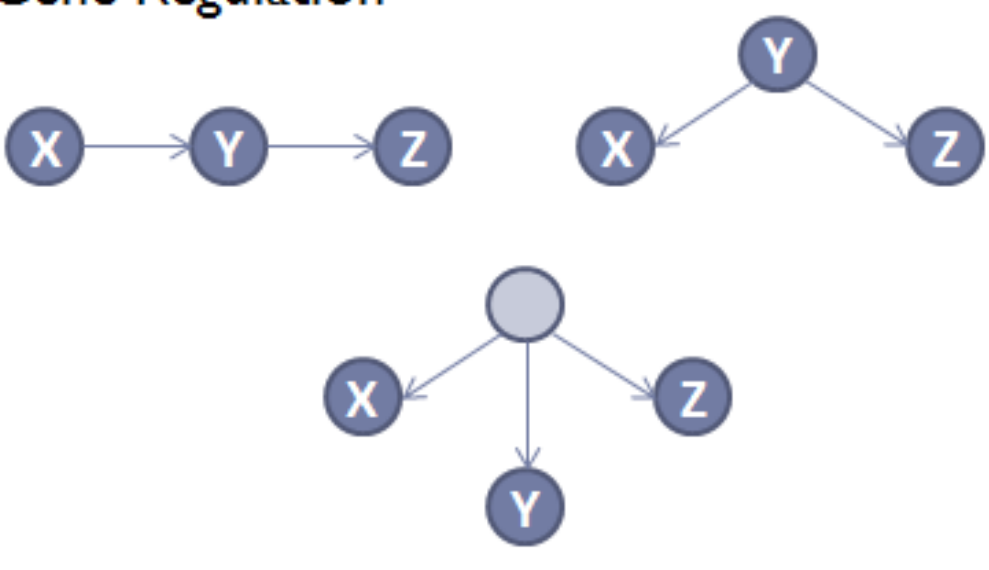
Introduction

Co-expression networks (Guilt-by-association)

Gene Co-expression



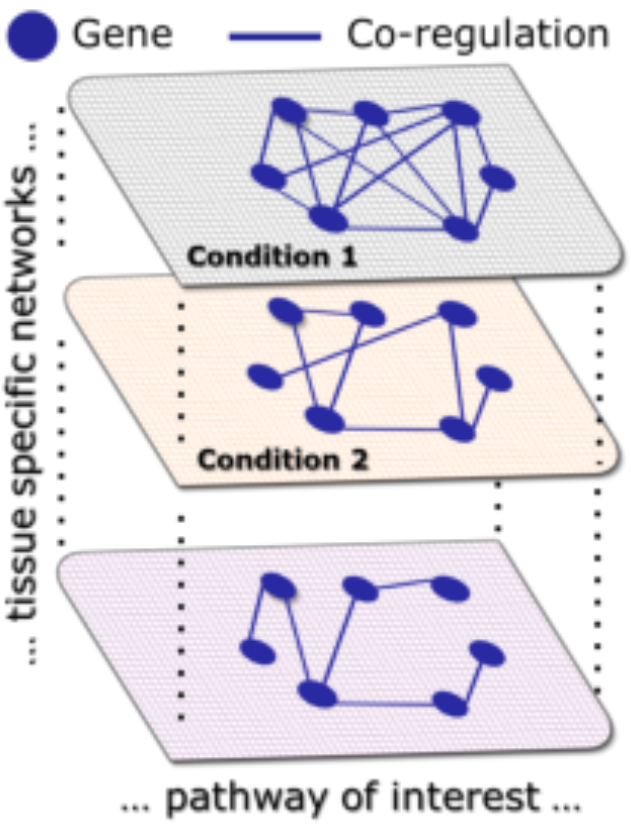
Gene Regulation



Co-expression networks analysis.

Introduction

Differential network analysis



Comparing co-expression networks amoung various groups.

Methodology is developed to detect different topological changes in the networks.

Introduction

Simulations to compare methods

- Start with regulatory network
- Generate RNA-seq data samples
- Run differential network analysis

Modeling

Distribution assumptions

Gaussian distribution for **microarrays**

- Continuous intensity scores

Negative-binomial for **RNA-Seq**

- Discrete counts

Modeling

Gamma-Poisson mixture

To generate values from a negative-binomial, we use the mixture

$$\theta \sim \text{Gamma}(\text{shape} = r,\ \text{scale} = \beta)$$
$$X \sim \text{Poisson}(\theta)$$

The marginal distribution of X will be **negative-binomial** with

$$E(X) = r\beta$$
$$Var(X) = r\beta(1 + \beta)$$

Modeling

Gamma-Poisson mixture (cont.)

For a desired average count μ, we can set

$$r = \mu/\gamma$$
$$\beta = \gamma$$

where γ is an overdispersion parameter.

Can have better control on tail-behavior by adding a third parameter

$$r = \mu^{2-k}/\gamma$$
$$\beta = \mu^{k-1}\gamma$$

Algorithm

Setting up the network

- Not just adjacency matrix
- Specify cliques, hubs, and other modules

Algorithm

Setting up the network (cont.)

- cliques: regulated by some latent variable
- hubs: hub gene regulating its connected genes
- modules: for construction of other pathways

Algorithm

Creating an adjacency matrix from structures

Let $A \in \{0, 1\}^{p \times p}$ and $A_{ij} = A_{ji} = 1$ if gene i and j are connected in any of the structures, and $A_{ij} = A_{ji} = 0$ otherwise.

Algorithm

Creating a weight matrix

Each edge in the adjacency graph needs to be assigned a weight. This weight determines the strength of the connection.

- Cliques: sample $w \sim N(0, \sigma)$
- Hubs: sample $w \sim N(0, \sigma)$ for each connection
- Modules: sample $w \sim N(0, \sigma)$ for each connection
- Add up any overlapping weights.

Different weights are computed **for each sample**.

Algorithm

Gamma-Poisson mixture

Given a weight matrix $W \in \mathbb{R}^{p \times p}$ containing the total weight of each edge, and baseline means $\mu \in \mathbb{R}^p$

1. Let $i = 1$
2. Set $\mu_i = \mu_i \exp(\sum_j W_{ji})$
3. Sample $\theta \sim \text{Gamma}(\mu^{2-k}/\gamma, \mu^{k-1}\gamma)$
4. Set $x_i \sim \text{Poisson}(\theta)$
5. Repeat steps 2 - 4 for $i = 1, \ldots, p$.

Examples R code

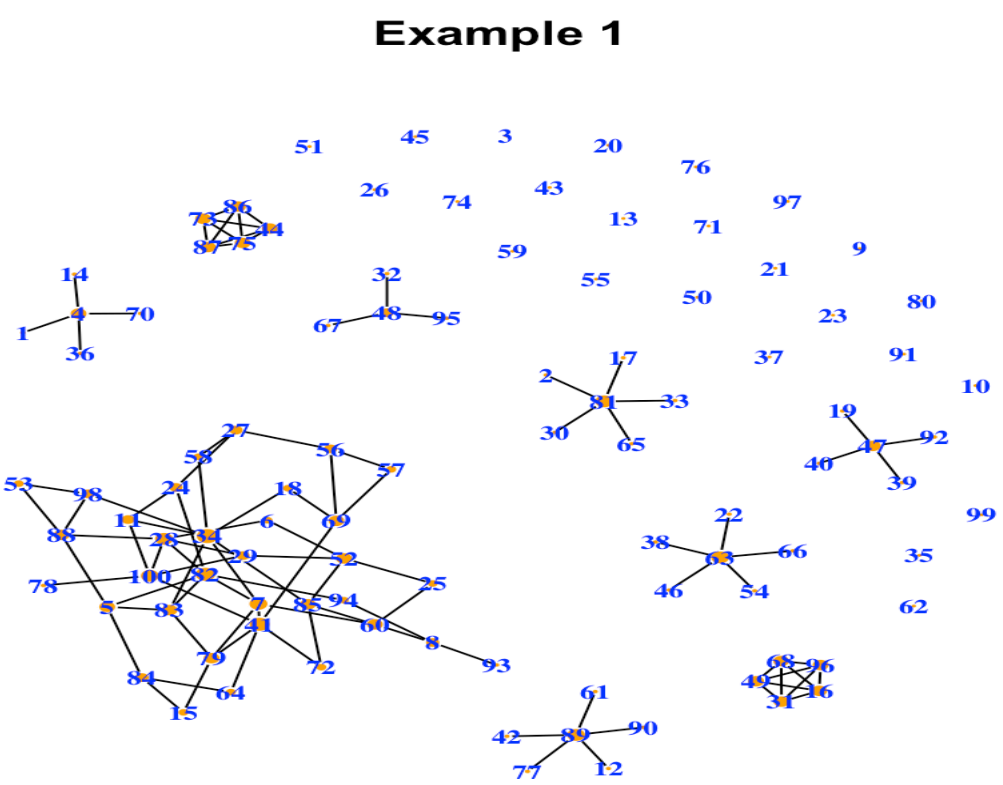
Installing and loading

```
devtools::install_github("tgrimes/SeqNet")
library(SeqNet)
```

Examples R code

Creating a network and plotting

```
set.seed(12345)
p <- 100
network <- create_network(p = p,
                          clique_size = c(5, 5),
                          hub_size = c(5, 5, 4, 6, 6, 6),
                          module_size = c(35),
                          nonoverlapping = TRUE)
plot(network, main = "Example 1")
```

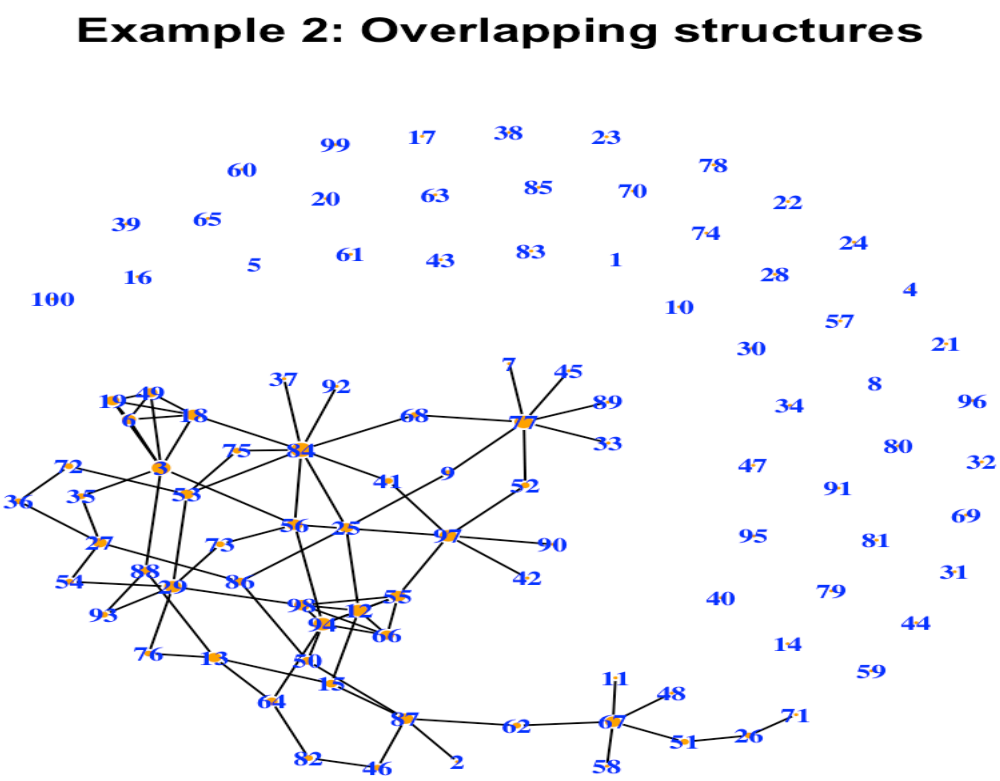



```
## IGRAPH 77a1077 UN-- 100 102 --
## + attr: name (v/c), size (v/n), frame.color (v/c), width (e/n)
## + edges from 77a1077 (vertex names):
##  [1] 1 --4   2 --81  4 --14  4 --36  4 --70  5 --82  5 --83  5 --84
##  [9] 5 --88  6 --28  6 --52  7 --34  7 --41  7 --60  7 --79  7 --82
## [17] 8 --60  8 --93  8 --94  11--24  11--34  11--100 12--89  15--79
## [25] 15--84  16--31  16--49  16--68  16--96  17--81  18--34  18--69
## [33] 19--47  22--63  24--27  24--82  25--52  25--60  27--56  27--58
## [41] 28--29  28--82  28--88  28--100 29--52  29--100 30--81  31--49
## [49] 31--68  31--96  32--48  33--81  34--58  34--83  34--85  34--98
## [57] 38--63  39--47  40--47  41--64  41--69  41--72  41--79  41--100
## + ... omitted several edges
```

Examples R code

Creating a network and plotting (cont.)

```
network <- create_network(p = p,
                          clique_size = c(5, 5),
                          hub_size = c(5, 5, 4, 6, 6, 6),
                          module_size = c(35),
                          nonoverlapping = FALSE)
plot(network, main = "Example 2: Overlapping structures")
```

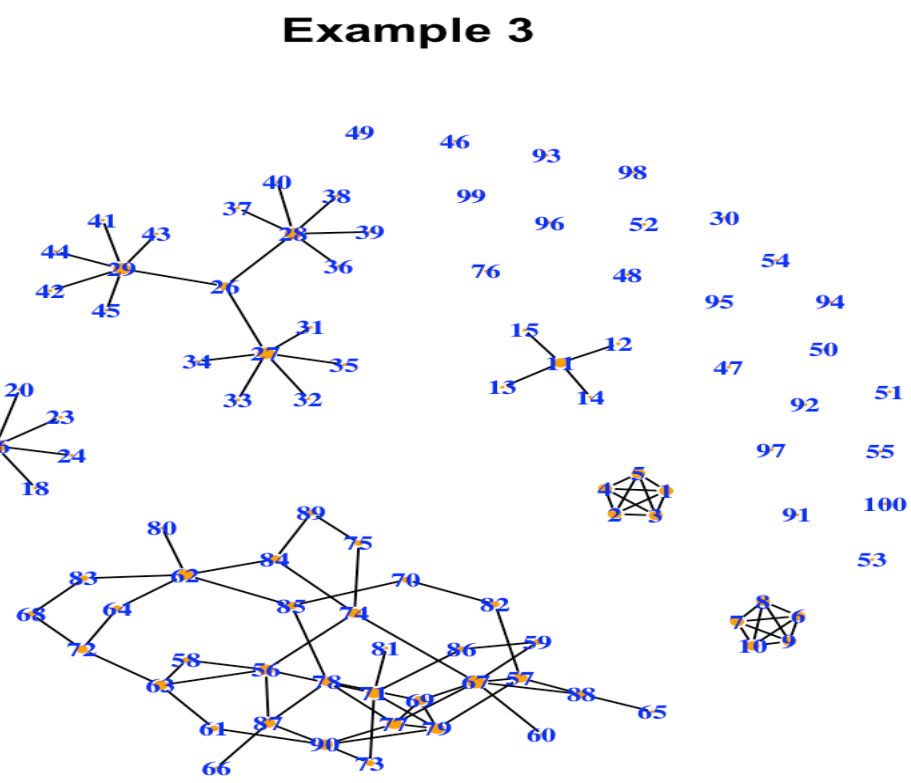



```
## IGRAPH cff353d UN-- 100 86 --
## + attr: name (v/c), size (v/n), frame.color (v/c), width (e/n)
## + edges from cff353d (vertex names):
##  [1] 2 --87 3 --6  3 --18 3 --19 3 --35 3 --49 3 --56 3 --88 6 --18 6 --19
## [11] 6 --49 7 --77 9 --25 9 --77 11--67 12--15 12--25 12--55 12--66 12--94
## [21] 12--98 13--15 13--64 13--76 13--88 15--87 18--19 18--49 18--84 19--49
## [31] 25--84 25--86 26--51 26--71 27--35 27--36 27--54 27--86 29--53 29--54
## [41] 29--73 29--76 29--93 29--98 33--77 36--72 37--84 41--84 41--97 42--97
## [51] 45--77 46--82 46--87 48--67 50--86 50--87 50--94 51--67 52--77 52--97
## [61] 53--72 53--75 53--84 55--66 55--94 55--97 55--98 56--73 56--84 56--94
## [71] 56--97 58--67 62--67 62--87 64--82 64--94 66--94 66--98 68--77 68--84
## + ... omitted several edges
```

Examples R code

Creating a network and plotting (cont.)

```
network <- create_network(p = p,
                          cliques = list(1:5, 6:10),
                          hubs = list(11:15, 16:25, 26:29,
                                      c(27, 31:35), c(28, 36:40), c(29, 41:45)),
                          modules = list(56:90))
plot(network, main = "Example 3")
```




```
## IGRAPH 29ca511 UN-- 100 102 --
## + attr: name (v/c), size (v/n), frame.color (v/c), width (e/n)
## + edges from 29ca511 (vertex names):
##  [1] 1 --2  1 --3  1 --4  1 --5  2 --3  2 --4  2 --5  3 --4  3 --5  4 --5
## [11] 6 --7  6 --8  6 --9  6 --10 7 --8  7 --9  7 --10 8 --9  8 --10 9 --10
## [21] 11--12 11--13 11--14 11--15 16--17 16--18 16--19 16--20 16--21 16--22
## [31] 16--23 16--24 16--25 26--27 26--28 26--29 27--31 27--32 27--33 27--34
## [41] 27--35 28--36 28--37 28--38 28--39 28--40 29--41 29--42 29--43 29--44
## [51] 29--45 56--58 56--63 56--71 56--74 56--87 57--67 57--79 57--82 57--88
## [61] 58--63 59--67 59--86 60--67 61--63 61--90 62--64 62--80 62--83 62--84
## [71] 62--85 63--72 64--72 65--88 66--87 67--69 67--74 67--77 67--88 68--72
## + ... omitted several edges
```

Examples R code

Co-expression networks $n = 50$

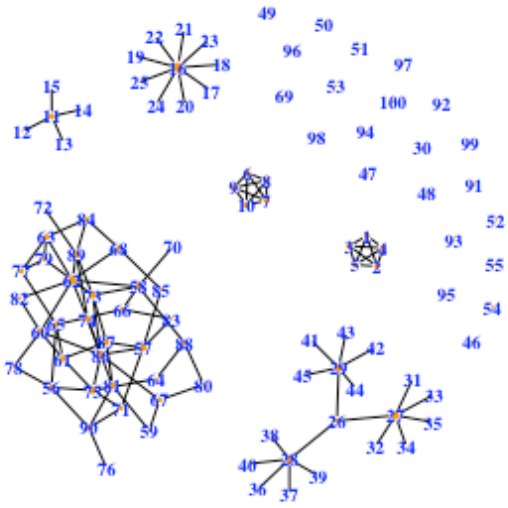
```
n <- 50
overdispersion <- 1
intensity <- 1
k <- 1.5
network <- add_sign_to_network(network)
mu <- sample(get_reference_count_means(), p, replace = TRUE)
df <- gen_gamma_poisson(n, network, mu,
                        overdispersion= overdispersion,
                        intensity = intensity, k = k)
cpls <- run_cpls(df$x, v = 3)$scores
corr <- run_corr(df$x)$scores
wgcna <- run_wgcna(df$x)$scores
n_top <- 100
adj_matrix_using_top_scores <- function(scores) {
    scores[abs(scores) < sort(abs(scores), decreasing = TRUE)[n_top]] <- 0
    scores[scores != 0] <- 1
    return(scores)
}
```

Examples R code

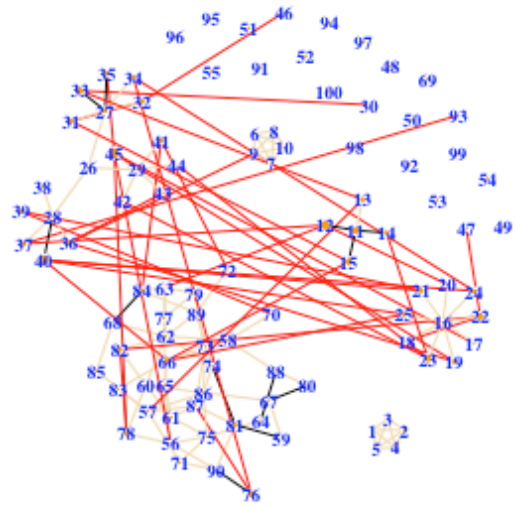
Co-expression networks $n = 50$ (cont.)

```
par(mfrow = c(2, 2))
g <- plot(network, main = "Regulatory Network")
plot_network(adj_matrix_using_top_scores(cpls), g, main = "cPLS")
plot_network(adj_matrix_using_top_scores(corr), g, main = "cor")
plot_network(adj_matrix_using_top_scores(wgcna), g, main = "WGCNA")
```

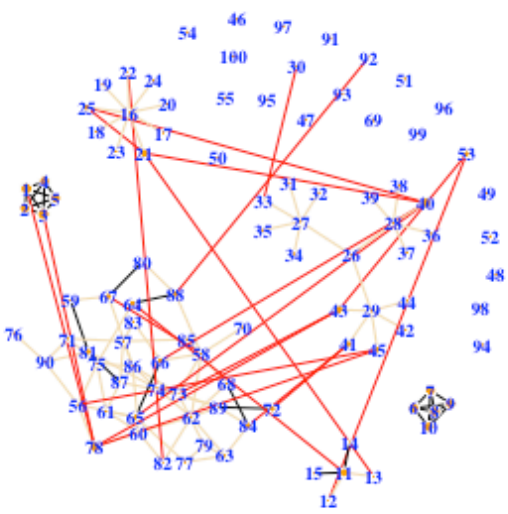
Regulatory Network



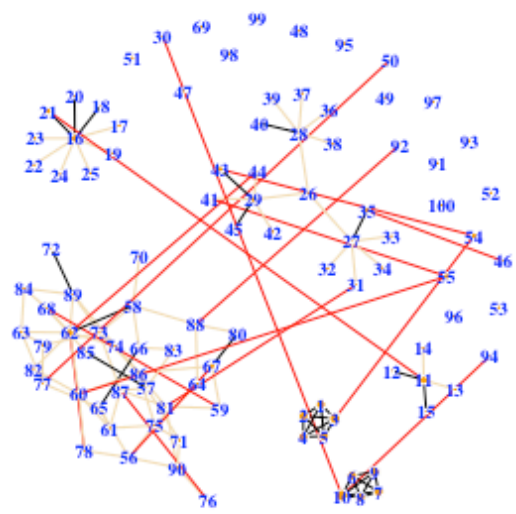
cPLS



cor



WGCNA



Examples R code

Co-expression networks $n = 200$

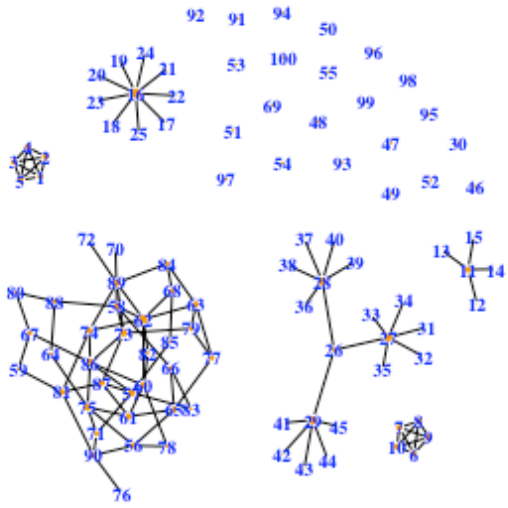
```
n <- 200
overdispersion <- 1
intensity <- 1
k <- 1.5
network <- add_sign_to_network(network)
mu <- sample(get_reference_count_means(), p, replace = TRUE)
df <- gen_gamma_poisson(n, network, mu,
                        overdispersion= overdispersion,
                        intensity = intensity, k = k)
cpls <- run_cpls(df$x, v = 3)$scores
corr <- run_corr(df$x)$scores
wgcna <- run_wgcna(df$x)$scores
n_top <- 100
adj_matrix_using_top_scores <- function(scores) {
    scores[abs(scores) < sort(abs(scores), decreasing = TRUE)[n_top]] <- 0
    scores[scores != 0] <- 1
    return(scores)
}
```

Examples R code

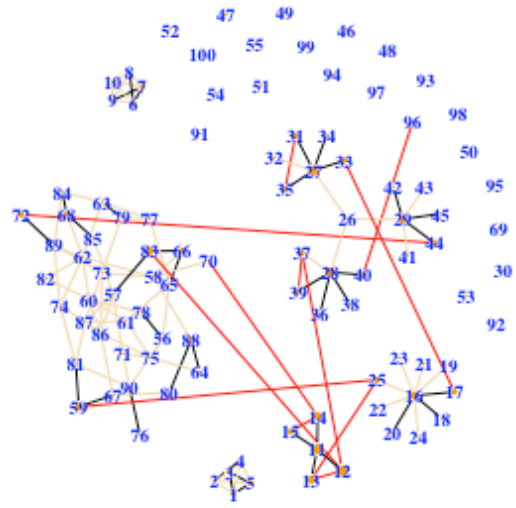
Co-expression networks $n = 200$ (cont.)

```
par(mfrow = c(2, 2))
g <- plot(network, main = "Regulatory Network")
plot_network(adj_matrix_using_top_scores(cpls), g, main = "cPLS")
plot_network(adj_matrix_using_top_scores(corr), g, main = "cor")
plot_network(adj_matrix_using_top_scores(wgcna), g, main = "WGCNA")
```

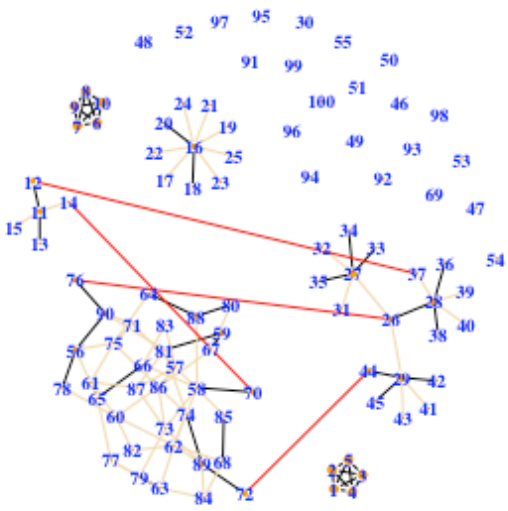
Regulatory Network



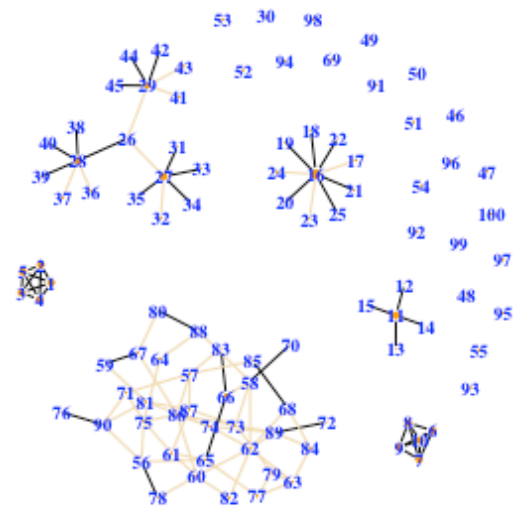
cPLS



cor



WGCNA



Examples R code

Co-expression networks $n = 1000$

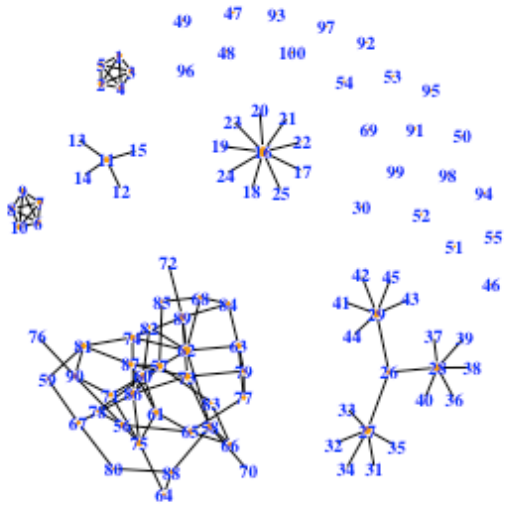
```
n <- 1000
overdispersion <- 1
intensity <- 1
k <- 1.5
network <- add_sign_to_network(network)
mu <- sample(get_reference_count_means(), p, replace = TRUE)
df <- gen_gamma_poisson(n, network, mu,
                        overdispersion= overdispersion,
                        intensity = intensity, k = k)
cpls <- run_cpls(df$x, v = 3)$scores
corr <- run_corr(df$x)$scores
wgcna <- run_wgcna(df$x)$scores
n_top <- 200
adj_matrix_using_top_scores <- function(scores) {
    scores[abs(scores) < sort(abs(scores), decreasing = TRUE)[n_top]] <- 0
    scores[scores != 0] <- 1
    return(scores)
}
```

Examples R code

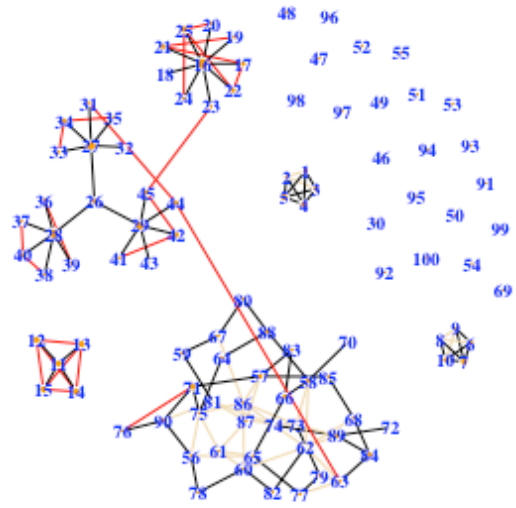
Co-expression networks $n = 1000$ (cont.)

```
par(mfrow = c(2, 2))
g <- plot(network, main = "Regulatory Network")
plot_network(adj_matrix_using_top_scores(cpls), g, main = "cPLS")
plot_network(adj_matrix_using_top_scores(corr), g, main = "cor")
plot_network(adj_matrix_using_top_scores(wgcna), g, main = "WGCNA")
```

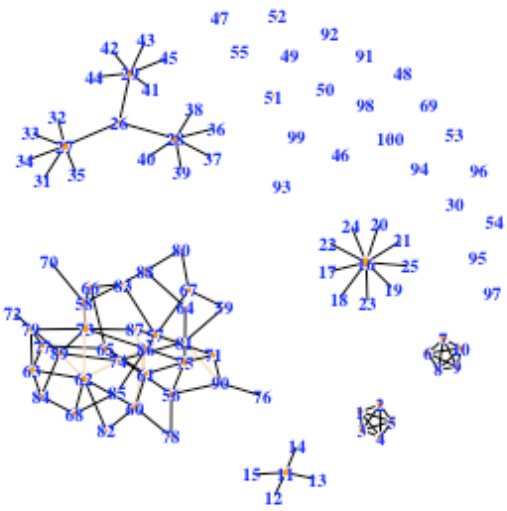
Regulatory Network



cPLS



cor



WGCNA