# Using Cooccurrance Networks

Jim Brunner

Los Alamos National Laboratory

# Coincidence Network

Constructing a coincidence network. I map the abundances according to

$$a(r_{ji}) = \begin{cases} \lfloor \left( \frac{r_{ji}}{\max_{s_k}(r_{jk})} \right) n \rfloor + 1 & \frac{r_{ji}}{\max_{s_k}(r_{jk})} \geq m \\ 0 & \frac{r_{ji}}{\max_{s_k}(r_{jk})} < m \end{cases}$$

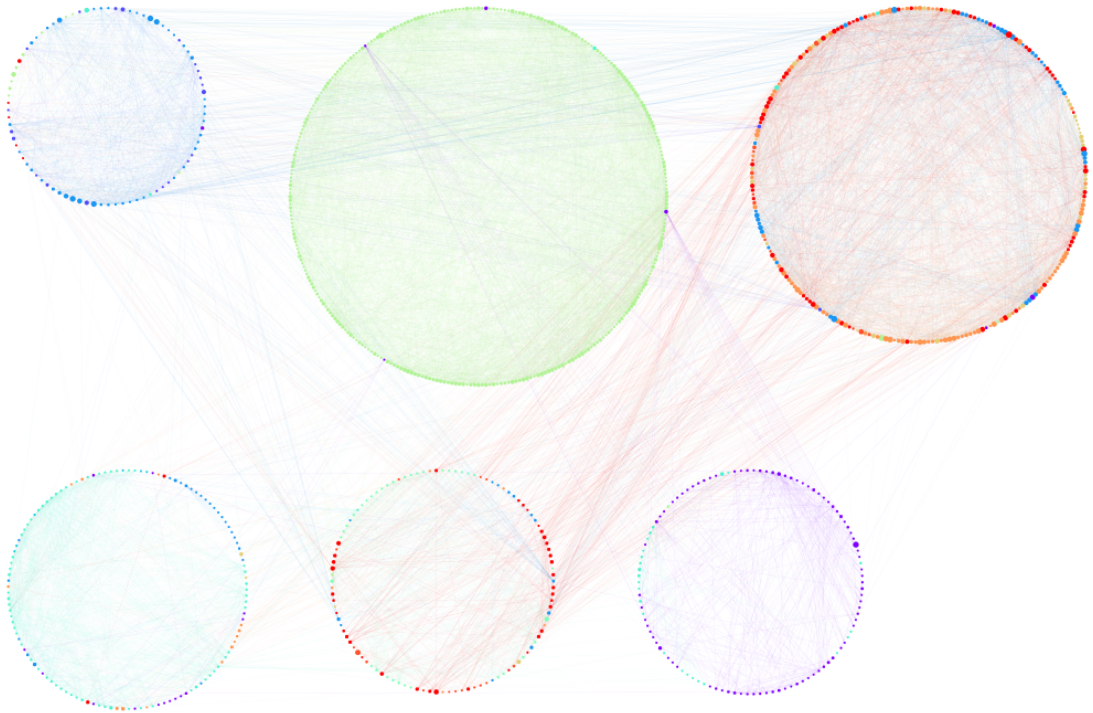
into "bins" relative to the maximum that taxa appears. Then count

$$w_{jk}^1 = \frac{\|\{i : a(r_{ji}) = a(r_{ki}) \neq 0\}\|}{S}$$

how often two organisms appear in the same bin.

# Cooccurrance Network

Same idea but now edges weights are compared to a random graph (null model). So we only keep edges that have a higher than “random” weight.



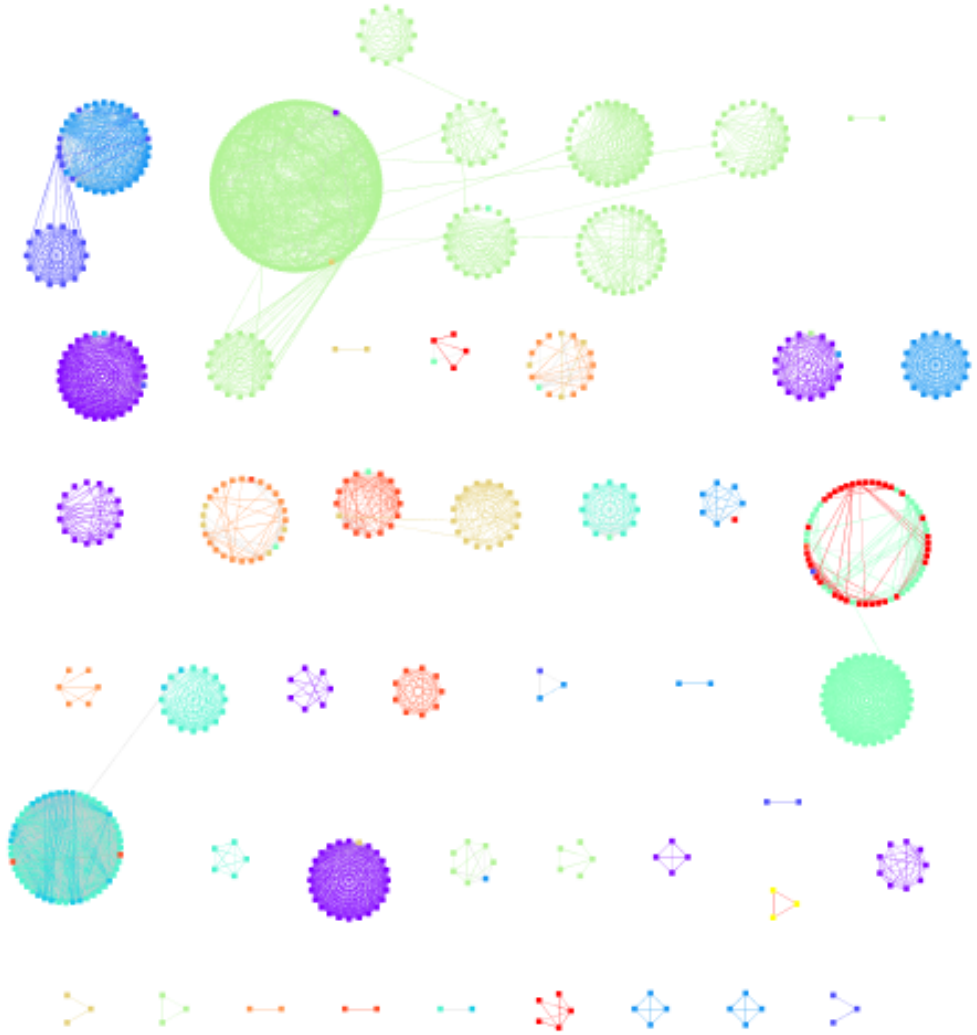
# Cooccurrance Network - Pearson Correlation

We can also compute the Pearson correlation coefficient between taxa across the samples.

$$\rho_{xy} = \frac{1}{N} \frac{(\boldsymbol{x} - \mu_x \mathbf{1}) \cdot (\boldsymbol{y} - \mu_y \mathbf{1})}{\sigma_x \sigma_y}$$

Then, we keep an edge if $\rho > 0.8$ and $p < 0.05$, where $p$ is the chance of correlation higher than $\rho_{xy}$ in a random model. The random model assigns abundances as a binomial with parameters determined by sample and taxa. The $p$ values are calculated using a Monte Carlo simulation with 1000 trials. The species level network using this method had $\sim 1/2$ as many edges.

# Cooccurrance Network - Pearson Correlation



# Clustering

We can cluster the network to attempt to determine or evaluate meaningful groups of taxa.

- Community clustering
  - Minimize a function based on "edge betweeness". Determines edges that are between clusters.
- Spectral clustering
  - Performs a random walk on the network.

# Analyzing a sample

- Filter GOTTCHA results - try to determine probability of seeing groups of organisms - Random Markov Field
- Diffusion on graph

$$\frac{\partial}{\partial t}u(v,t) = -Lu(v,t)$$

where $v$ takes values in the vertex set of the graph. Then, we can encode "known" information in three ways: initial values, boundary values, or a forcing vector.

# Analyzing a sample - Diffusion Process

$$\frac{\partial}{\partial t}u(v,t) = -Lu(v,t)$$

where $v$ takes values in the vertex set of the graph. Then, we can encode sample information in three ways: initial values, boundary values, or a forcing vector.

Related to spectral clustering - spectral clustering groups by "distance" in firsk $k$ eigenmodes of diffusion process. This process gives "distance" in weighted sum of eigenmodes of diffusion process.

**Highest ranked nodes are "closest" to the sample data**

# Initial Values

## Initial Value Problem

Let $u_i(t)$ be the solution at node $v_i$ to the discete diffusion problem

$$\frac{d}{dt}\boldsymbol{u}(t) = -L\boldsymbol{u}$$

where $L$ is the graph laplacian with initial conditions determined by sample information.
Then, if $\boldsymbol{K}$ is the information "known" from the sample and the values of $v_k$ and $v_l$ are unknown,

$$\int_0^\infty u_k(t)dt - \int_0^\infty u_l(t)dt > 0 \Rightarrow P(v_k = 1|\boldsymbol{K}) > P(v_l = 1|\boldsymbol{K})$$

# Boundary Values

## Boundary Value Problem

Let $u_i(t)$ be the solution at node $v_i$ to the discete diffusion problem

$$\frac{d}{dt}\boldsymbol{u}(t) = -L\boldsymbol{u}$$

where $L$ is the graph laplacian with fixed values (which can be regarded as boundary values) $u_i = 1$ if node $v_i$ is known to be "on", $u_j = 0$ if $v_j$ is known to be "off".
Then, if $\boldsymbol{K}$ is the information "known" and the values of $v_k$ and $v_l$ are unknown, and $\tilde{\boldsymbol{u}}$ is the equilibrium solution to the diffusion problem,

$$\tilde{u}_k dt > \tilde{u}_l \Leftrightarrow P(v_k = 1|\boldsymbol{K}) > P(v_l = 1|\boldsymbol{K})$$

# Forcing Function

## Forced Problem

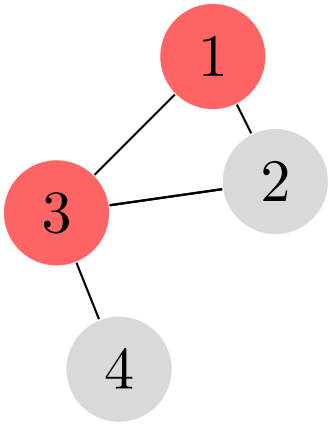
Let $u_i(t)$ be the solution at node $v_i$ to the discete diffusion problem

$$\frac{d}{dt}\boldsymbol{u}(t) = -L\boldsymbol{u} + f$$

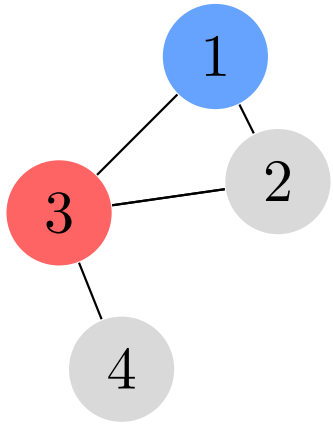
where $L$ is the graph laplacian and $f$ a forcing vector with $f_i = \alpha_{cc}$ if node $v_i$ is known to be "on", $f_j = -\beta_{cc}$ if $v_j$ is known to be "off", where $cc$ denotes a connected component of the graph. We choose $\alpha_{cc}$ and $\beta_{cc}$ so that on any connected component $cc$, $\sum \alpha_{cc} = \sum \beta_{cc} = 1$. Then, if $\boldsymbol{K}$ is the information "known" and the values of $v_k$ and $v_l$ are unknown, and $\tilde{\boldsymbol{u}}$ is the equilibrium solution to the diffusion problem,

$$\tilde{u}_k dt > \tilde{u}_l \Leftrightarrow P(v_k = 1|\boldsymbol{K}) > P(v_l = 1|\boldsymbol{K})$$

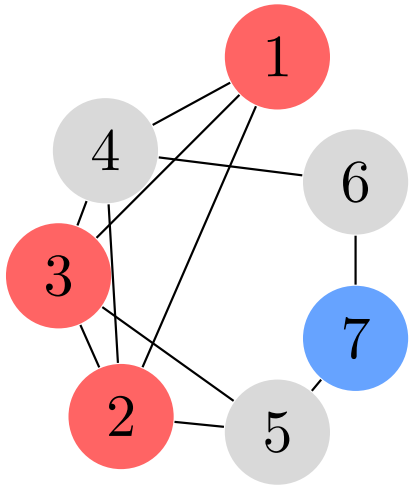
# Small Network Examples




(a) Small network, two on.




(b) small network, one on one off




(c) Larger network

Figure: Test networks. Gray nodes were "unknown", blue nodes "off", and red "on".

# Small Network Examples

| Configuration | Method | Ranking | Ties |
|---|---|---|---|
| (a) Small Network, two on. | IVP | 2, 4 | none |
| | BVP | 4, 2 | 4, 2 |
| | Forcing | 2, 4 | none |
| (b) Small Network, one on one off | IVP | 4, 2 | none |
| | BVP | 4, 2 | none |
| | Forcing | 4, 2 | none |
| (c) Larger network | IVP | 4, 5, 6 | none |
| | BVP | 4, 5, 6 | none |
| | Forcing | 4, 5, 6 | none |

# Network Examples

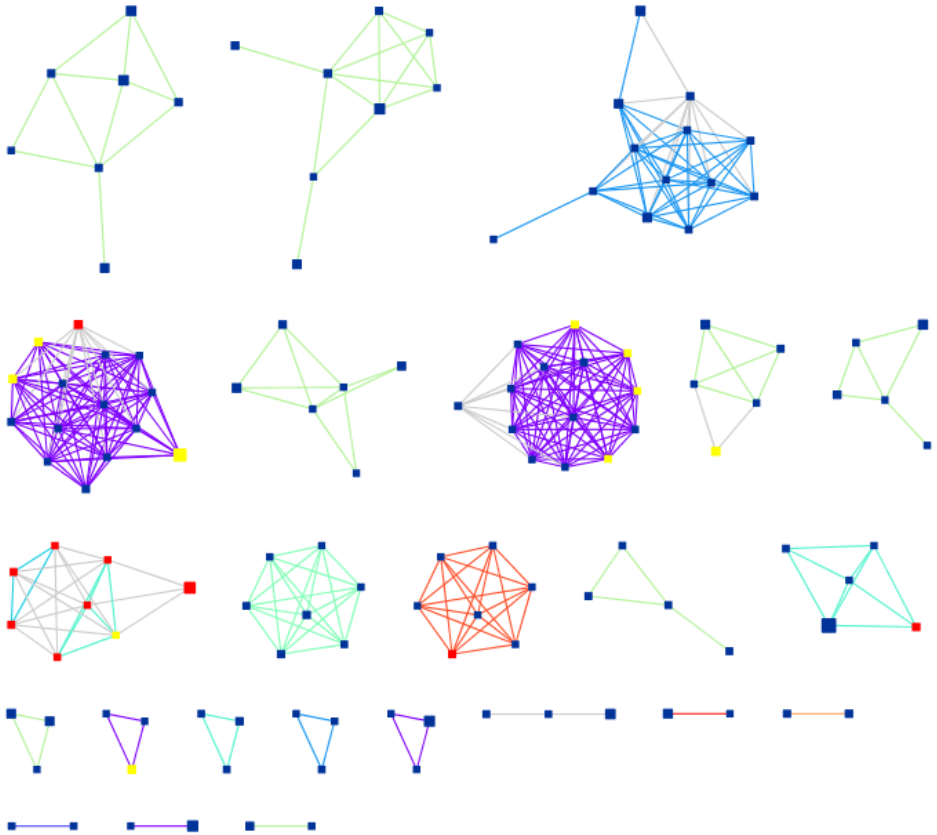


Figure: The “known” set - blue is off and red is on, while yellow is unknown.

# Network Examples

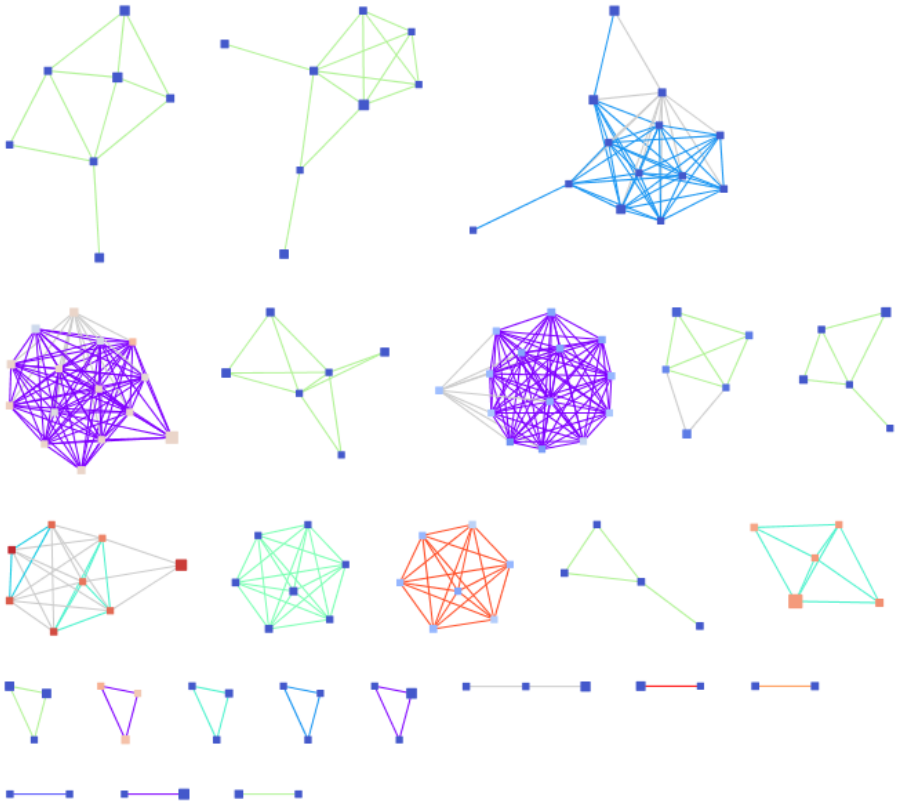


Figure: Result of initial value problem, with all nodes analyzed. Hotter colors indicate higher likelihood.

# Network Examples

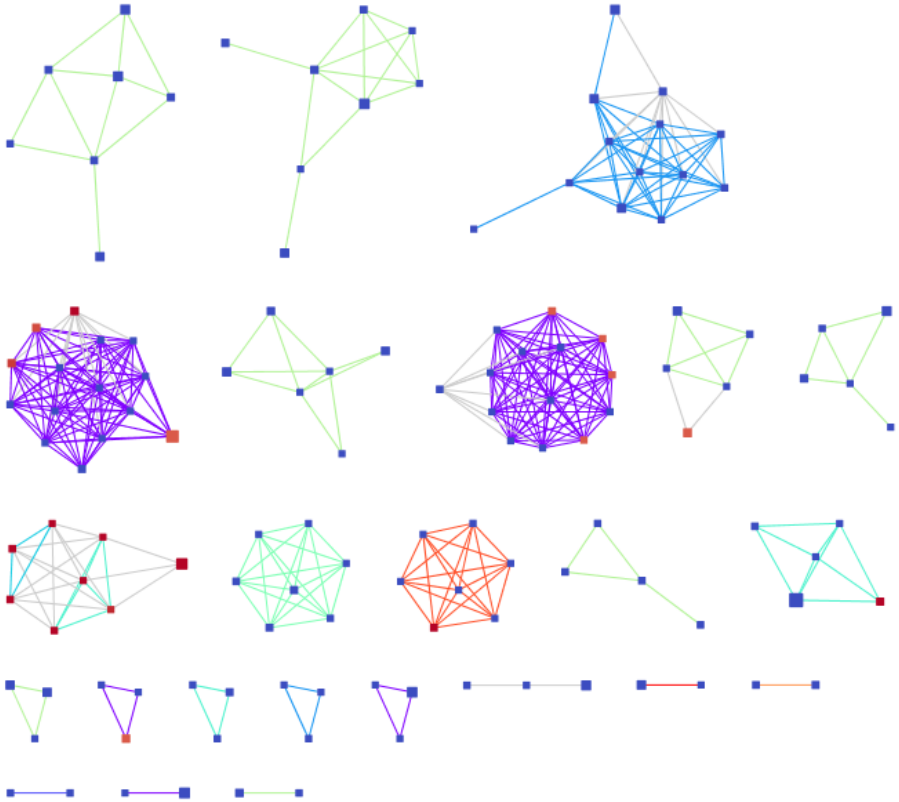


Figure: Result of boundary value problem. Hotter colors indicate higher likelihood, with red indicating assumed "on".

# Assigning a sample

Given a sample, we get ranking from a network $N_j$. Let $\boldsymbol{r}^j(\boldsymbol{s})$ be the ranking given by diffusion on network $j$. Then,

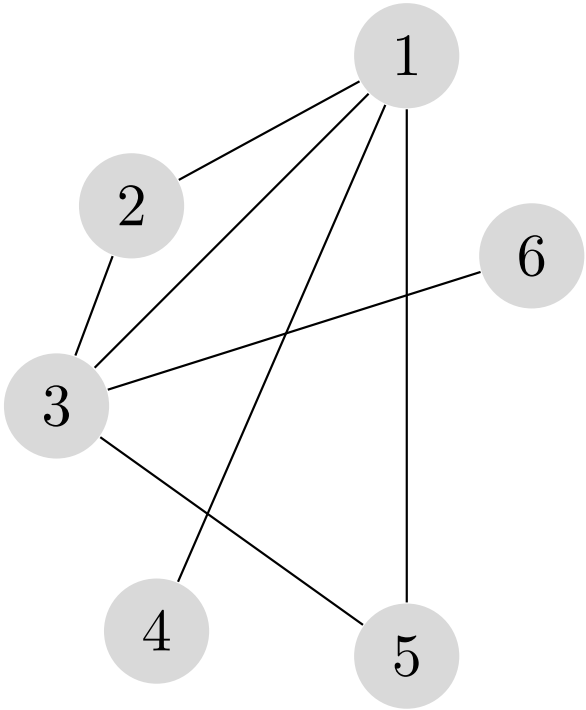
$$F_j(\boldsymbol{s}) = F(\boldsymbol{s}, \boldsymbol{r}^j(\boldsymbol{s})) = \sum_{i=1}^{n} c^{r_i^j} \frac{s_i}{\|\boldsymbol{s}\|_1}$$

and we can attempt to optimize over the networks we have (if there's only a few that's easy). The conjecture is then

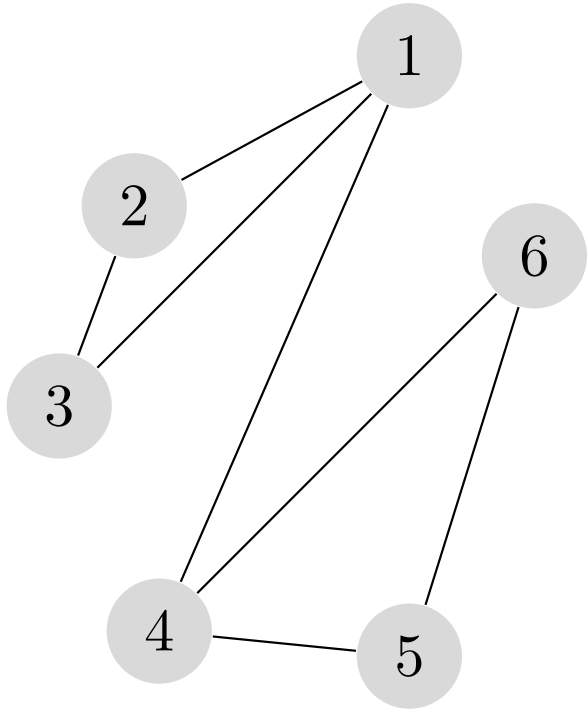
**Conjecture**

Assume that $F_j(\boldsymbol{s}) > F_k(\boldsymbol{s})$. Then $P(\boldsymbol{s}|N_j) > P(\boldsymbol{s}|N_k)$.

# Assigning a sample



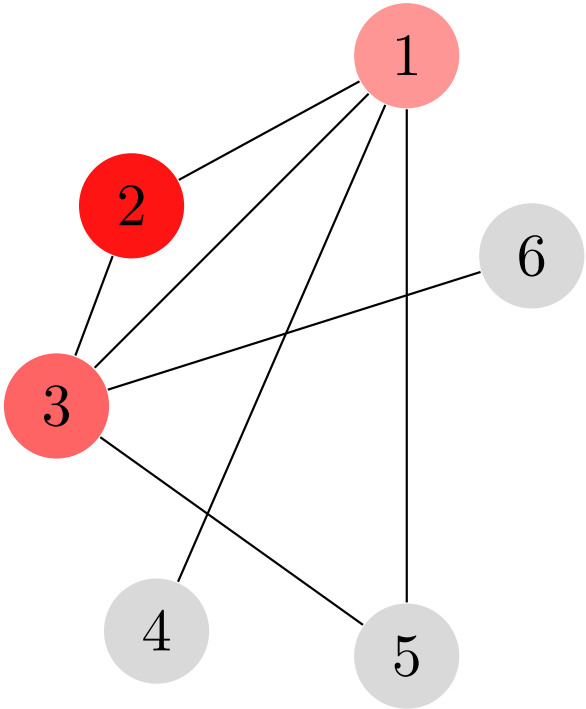

(a) Network $A_1$



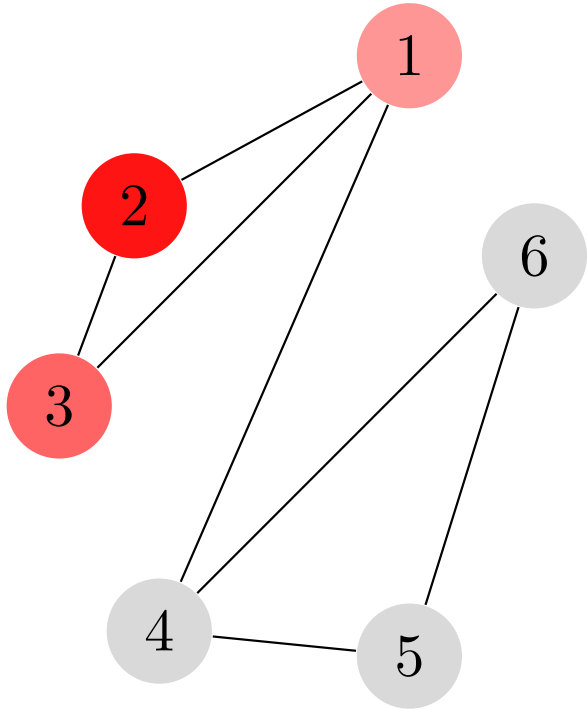

(b) Network $A_2$

Figure: Test Networks

# Assigning a sample



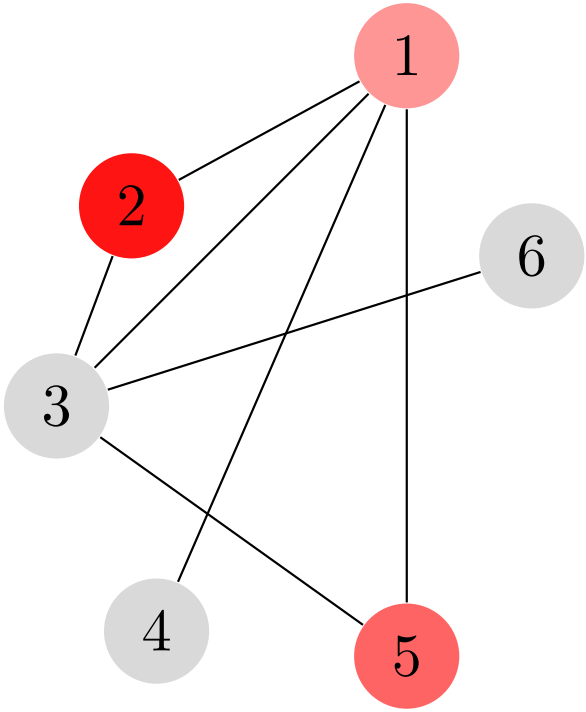

(a) Network $A_1$



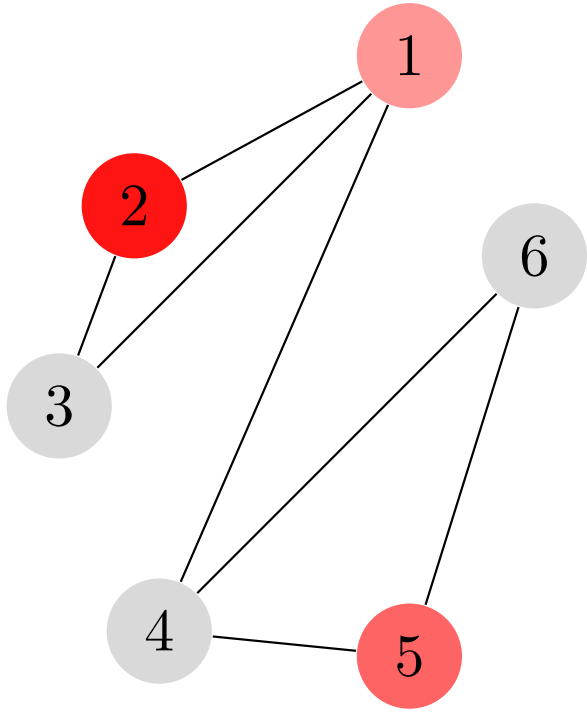

(b) Network $A_2$

Figure: "Sample" $u_1 = (1/6, 1/2, 1/3, 0, 0, 0)$

# Assigning a sample



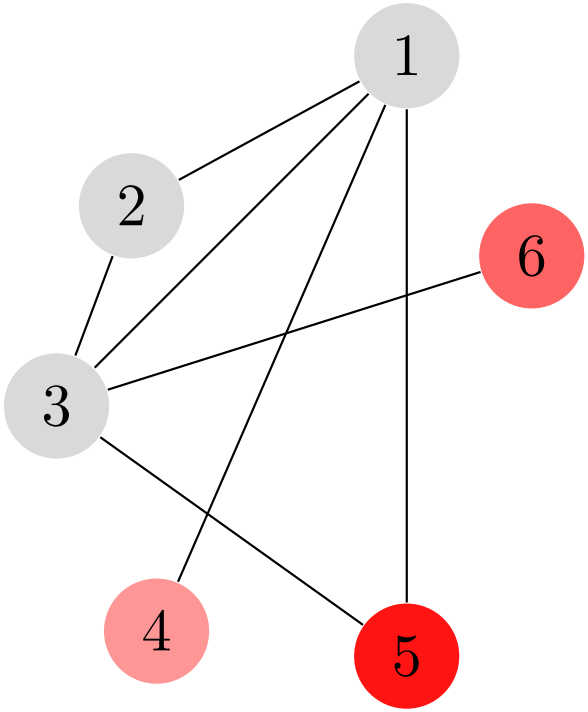

(a) Network $A_1$



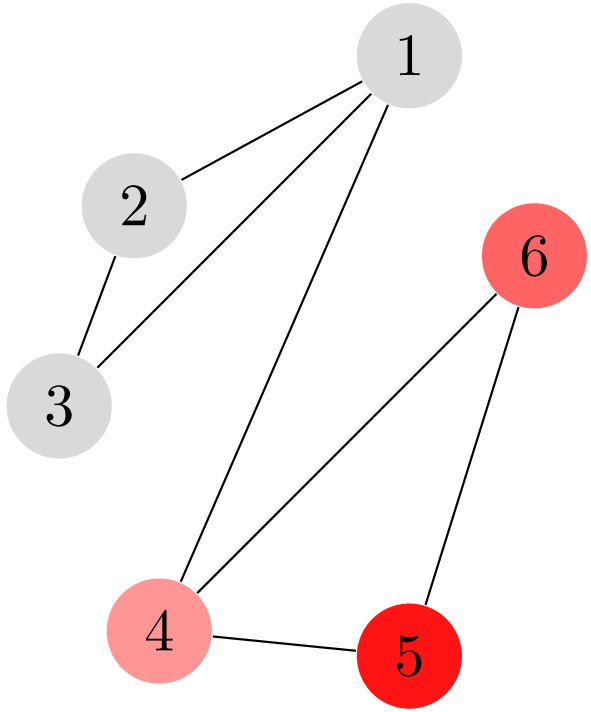

(b) Network $A_2$

Figure: "Sample" $u_2 = (1/6, 1/2, 0, 0, 1/3, 0)$

# Assigning a sample



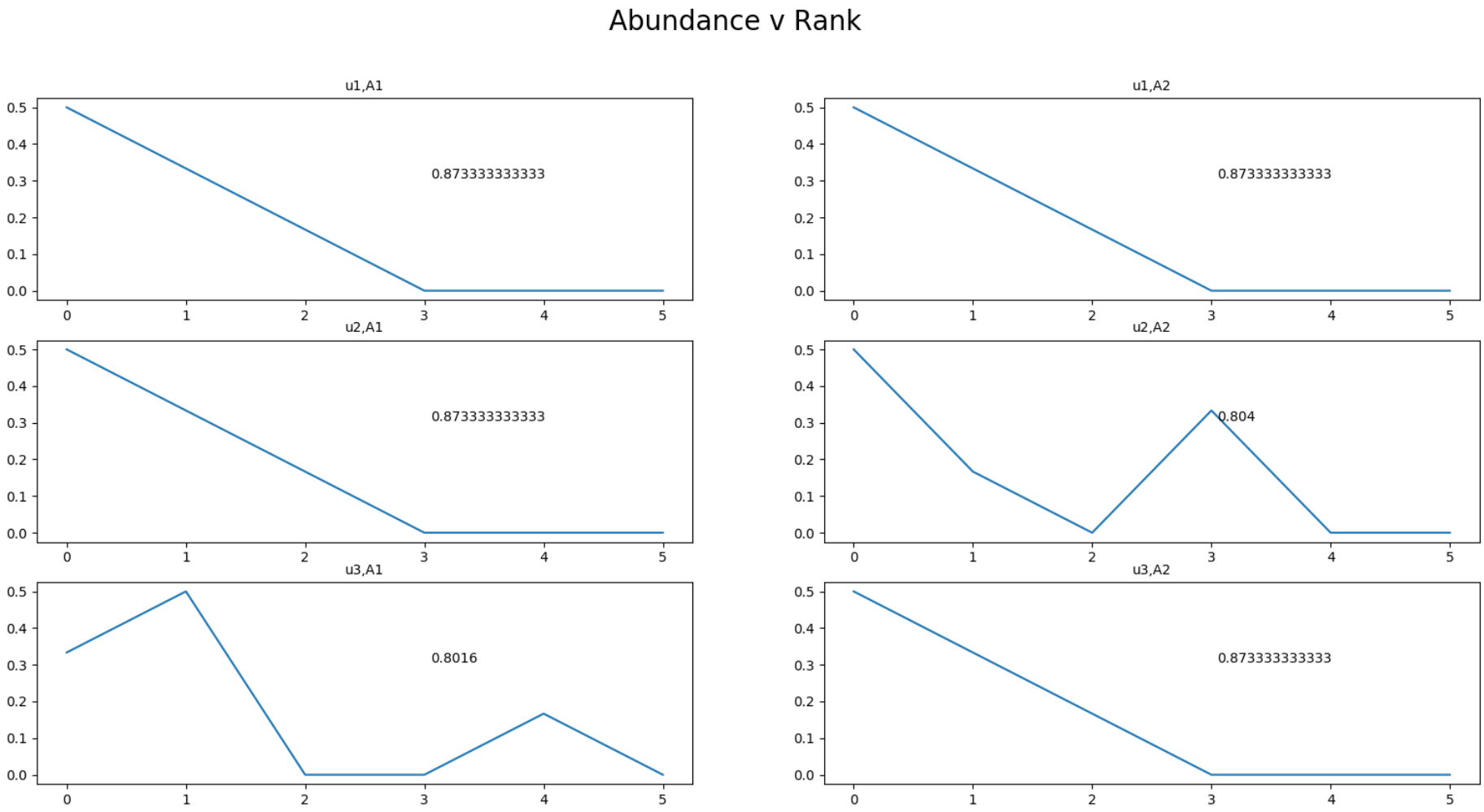

(a) Network $A_1$




(b) Network $A_2$

Figure: "Sample" $u_3 = (0, 0, 0, 1/6, 1/2, 1/3)$

# Assigning a sample



# Assigning a sample