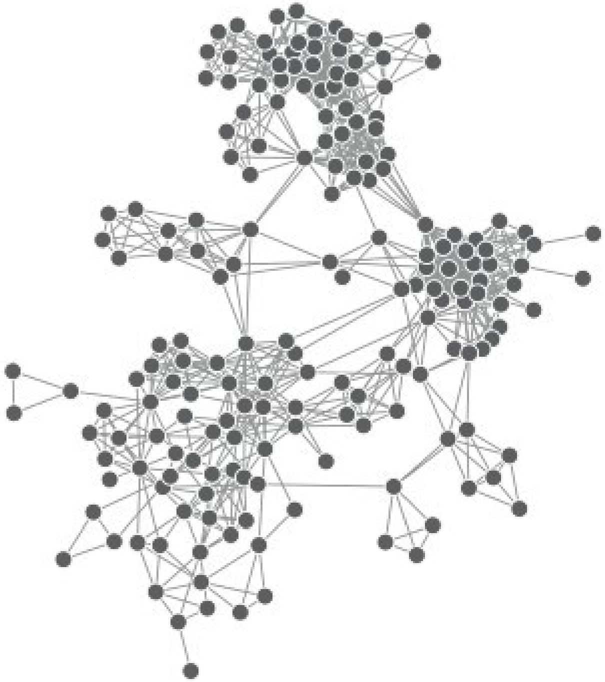
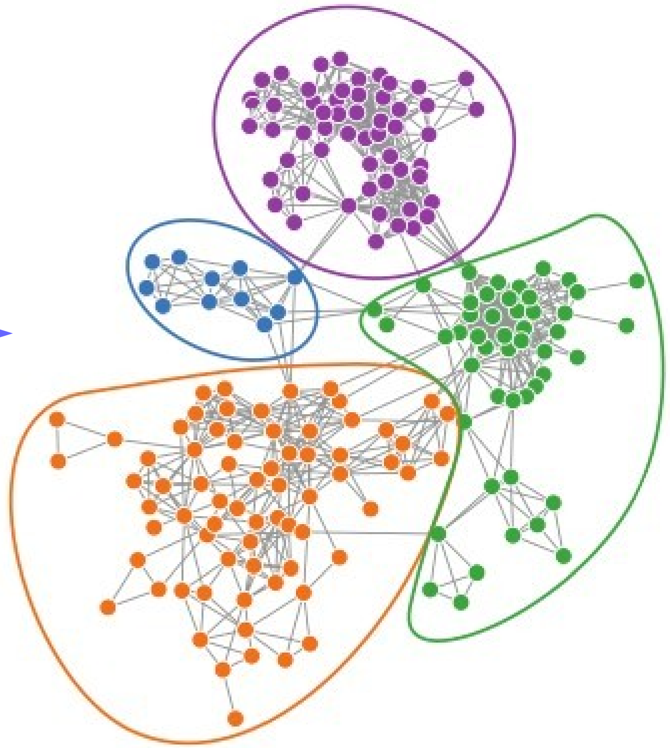
Community detection

Leto Peel
l.peel@maastrichtuniversity.nl
@PiratePeel

Community detection



Network



Communities

Supervised vs unsupervised

Supervised vs unsupervised

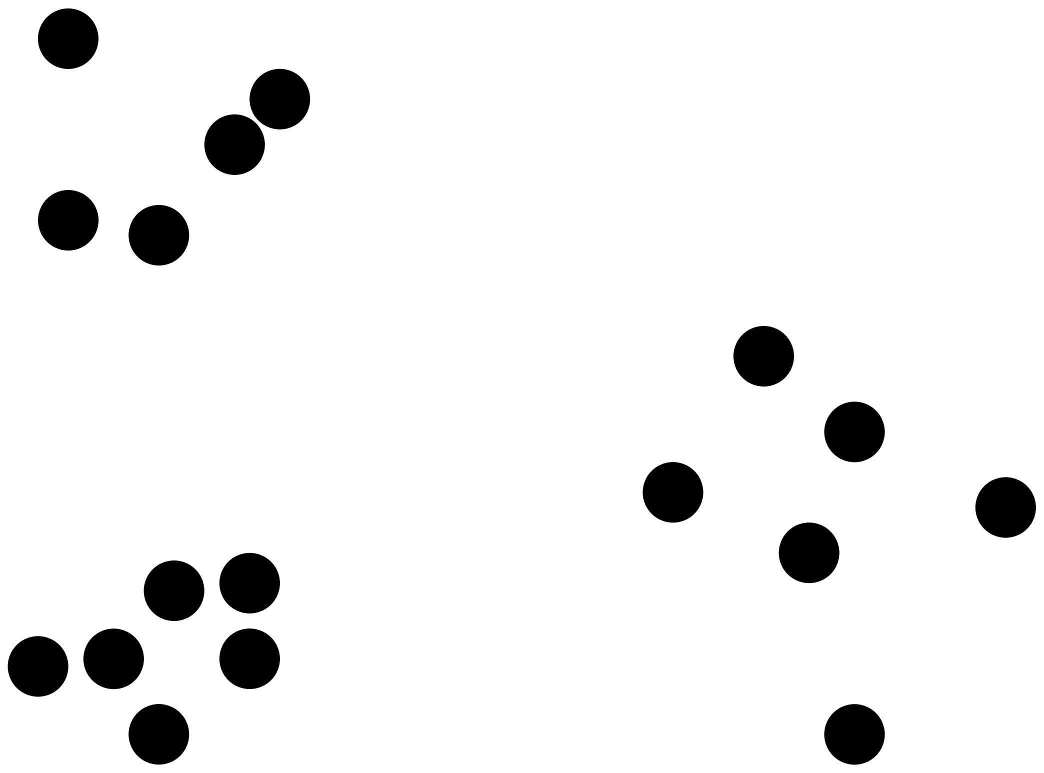
The supervised learner doesn't know much

Supervised vs unsupervised

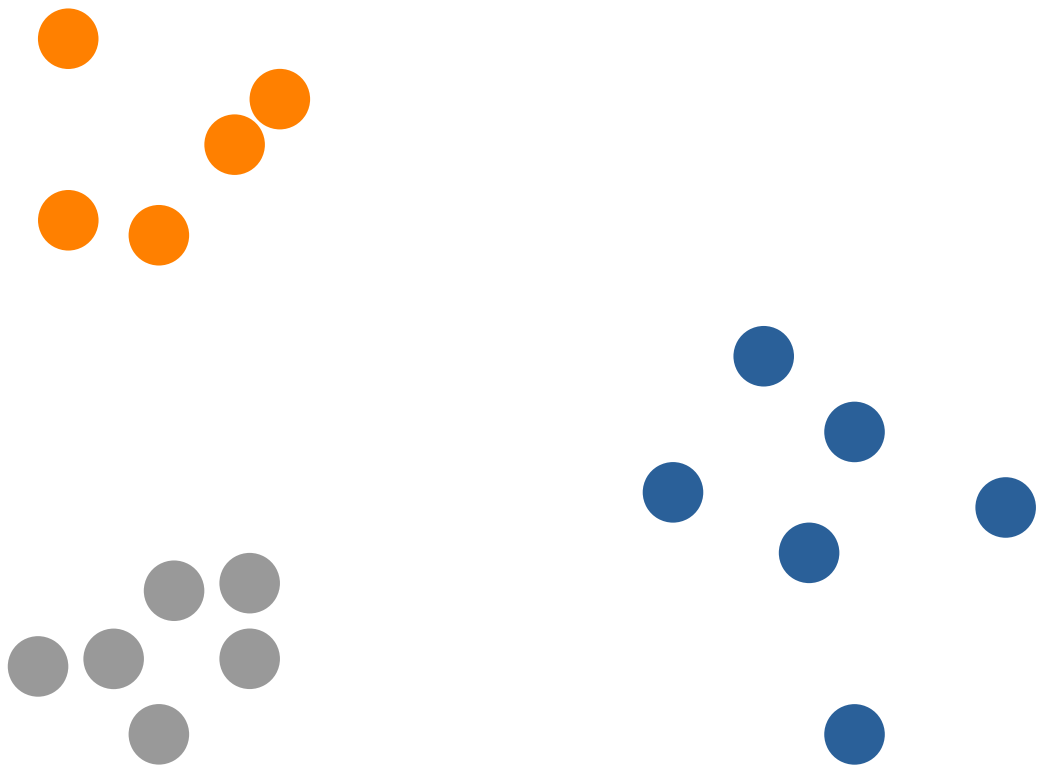
The supervised learner doesn't know much

The unsupervised learner knows what it is doing

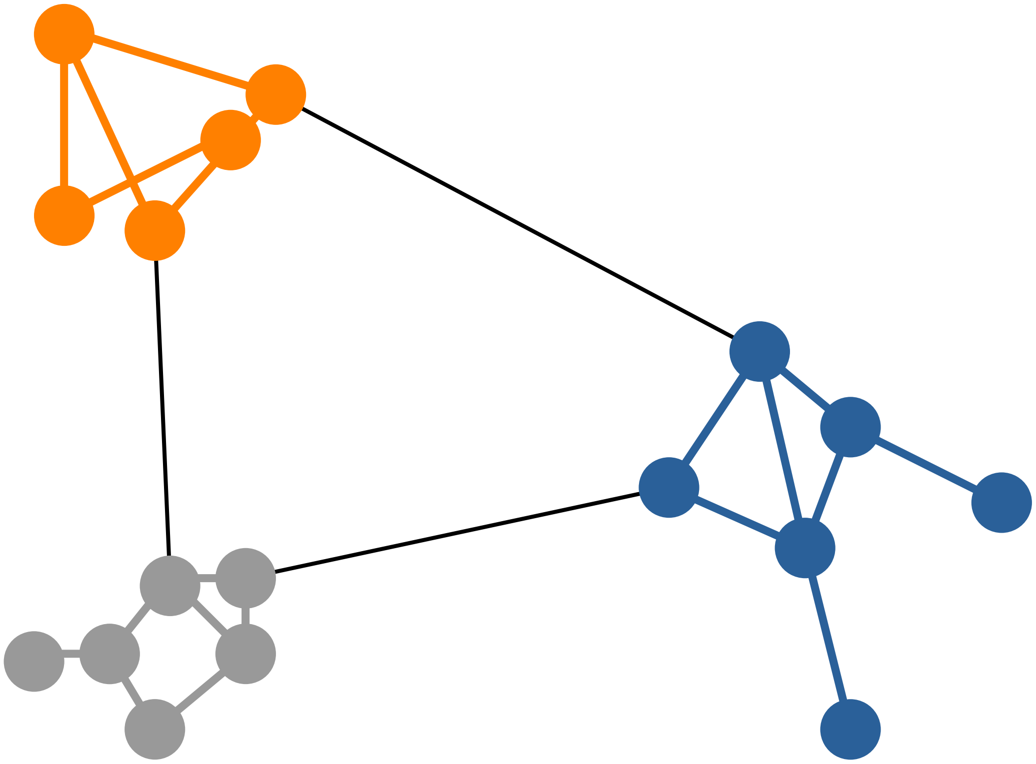
Clustering



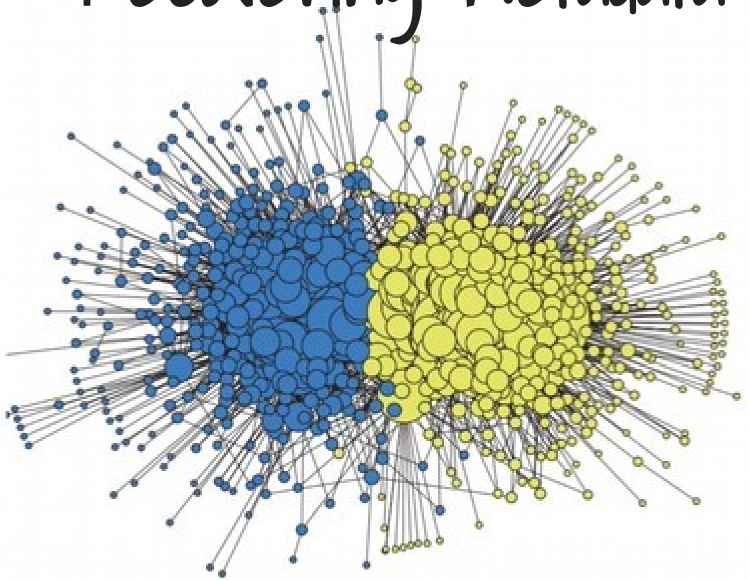
Clustering



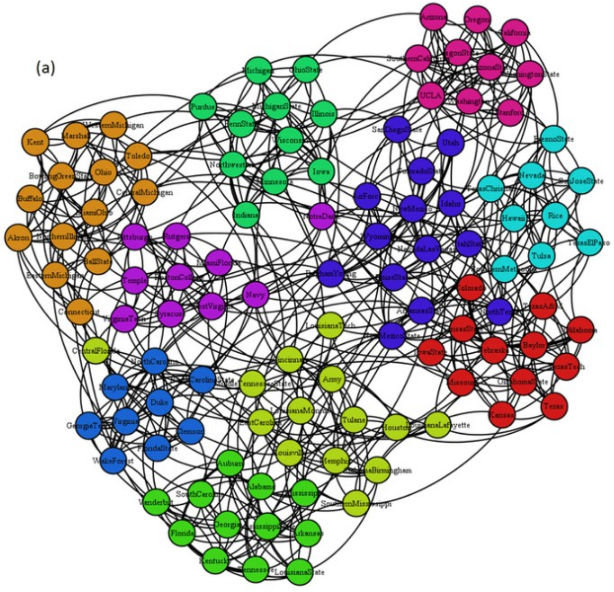
Community detection



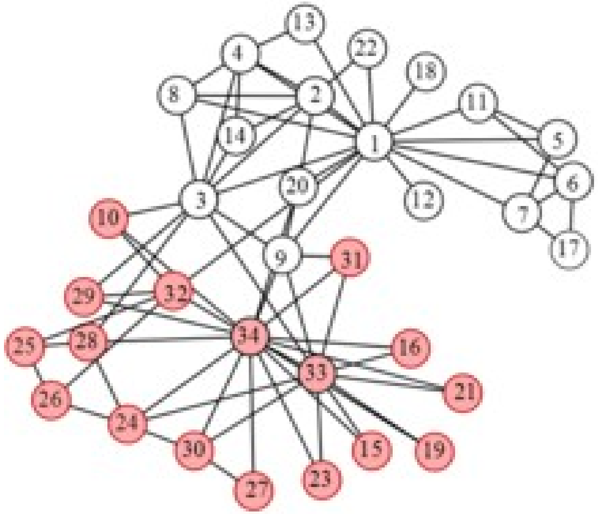
Recovering metadata implies sensible methods



US political blogs



American football



karate club*

*this network is so popular for community detection it has its own prize associated with it, see http://networkkarate.tumblr.com/

"Cluster" these objects

"Cluster" these objects

Red

Not Red

"Cluster" these objects

Cannot Fly

Can Fly

"Cluster" these objects

Transport

Not Transport

"Cluster" these objects

Not Alive

Alive

There are no ground truth communities!

Peel, L., Larremore, D. B., & Clauset, A. (2017). The ground truth about metadata and community detection in networks. Science advances, 3(5), e1602548.

Modularity

A partition better than random chance

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(x_i, x_j)$$

Modularity

A partition better than random chance

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(x_i, x_j)$$

Value of the
adjacency matrix

Modularity

A partition better than random chance

Node degree

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(x_i, x_j)$$

Value of the
adjacency matrix

Modularity

A partition better than random chance

Node degree

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(x_i, x_j)$$

2 time the number of edges in the network

Value of the adjacency matrix

Modularity

A partition better than random chance

Node degree

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(x_i, x_j)$$

2 time the number of edges in the network

Value of the adjacency matrix

Delta function: equals 1 if nodes are in the same community, or 0 if not.

Modularity

A partition better than random chance

$$Q = \sum_i (e_{ii} - a_i^2)$$

Modularity

A partition better than random chance

$$Q = \sum_i (e_{ii} - a_i^2)$$

Fraction of links
inside community *i*

Modularity

A partition better than random chance

$$Q = \sum_i (e_{ii} - a_i^2)$$

Fraction of links inside community *i*

Expected fraction of links in community *i* if the network had no community structure

Modularity

A partition better than random chance

$$a_i = \sum_j e_{ij}$$

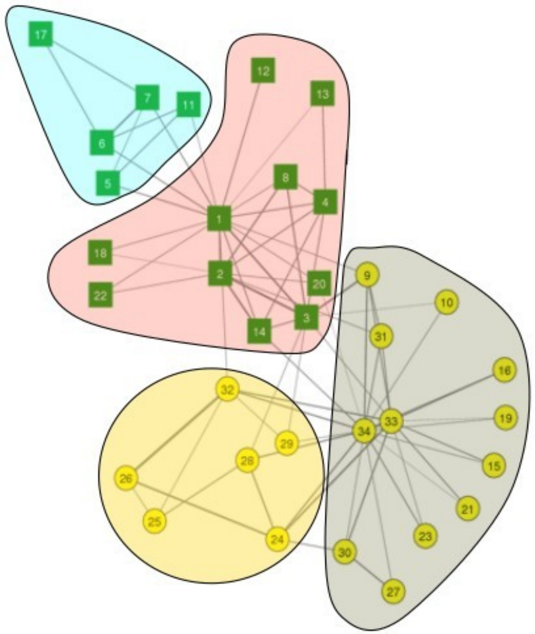
fraction of edges that go to community *i*

$$Q = \sum_i (e_{ii} - a_i^2)$$

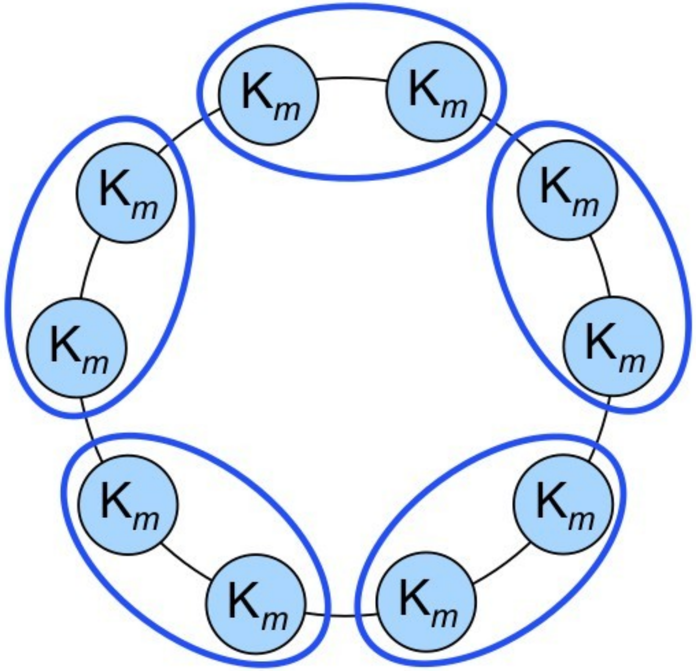
Fraction of links inside community *i*

Expected fraction of links in community *i* if the network had no community structure

Problems with modularity



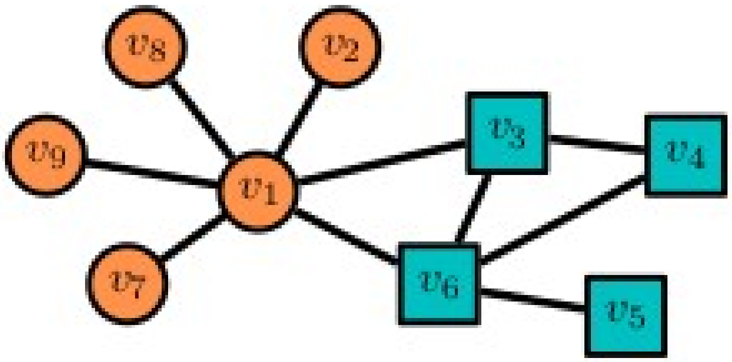
Finds spurious communities
(overfitting)



Resolution limit
(underfitting)

Calculate the modularity

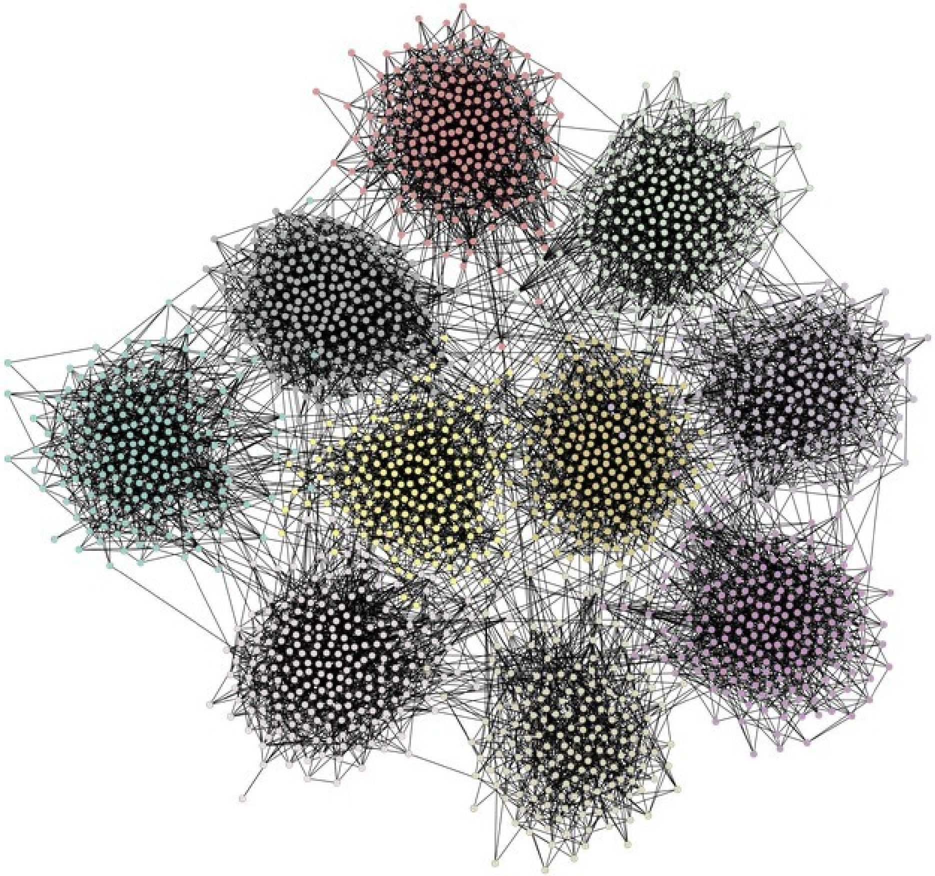
$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(x_i, x_j)$$



$$Q = \sum_i (e_{ii} - a_i^2)$$

Practical Q1 and Q2

Stochastic Block Models



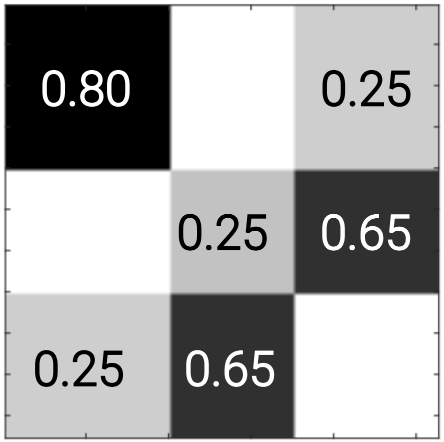
Generating a network using the SBM

Step 1 : Assign each node to a group

Generating a network using the SBM

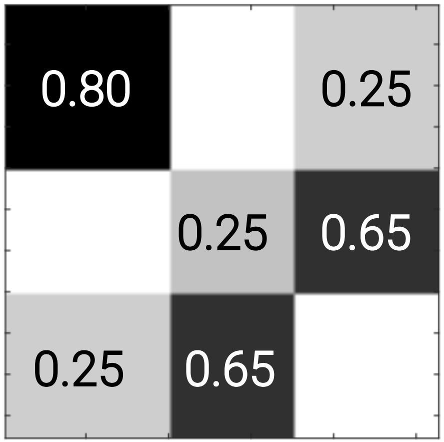
Step 1 : Assign each node to a group

- **Step 2** : Select some connection probabilities (mixing matrix)



Mixing Matrix

Generating a network using the SBM



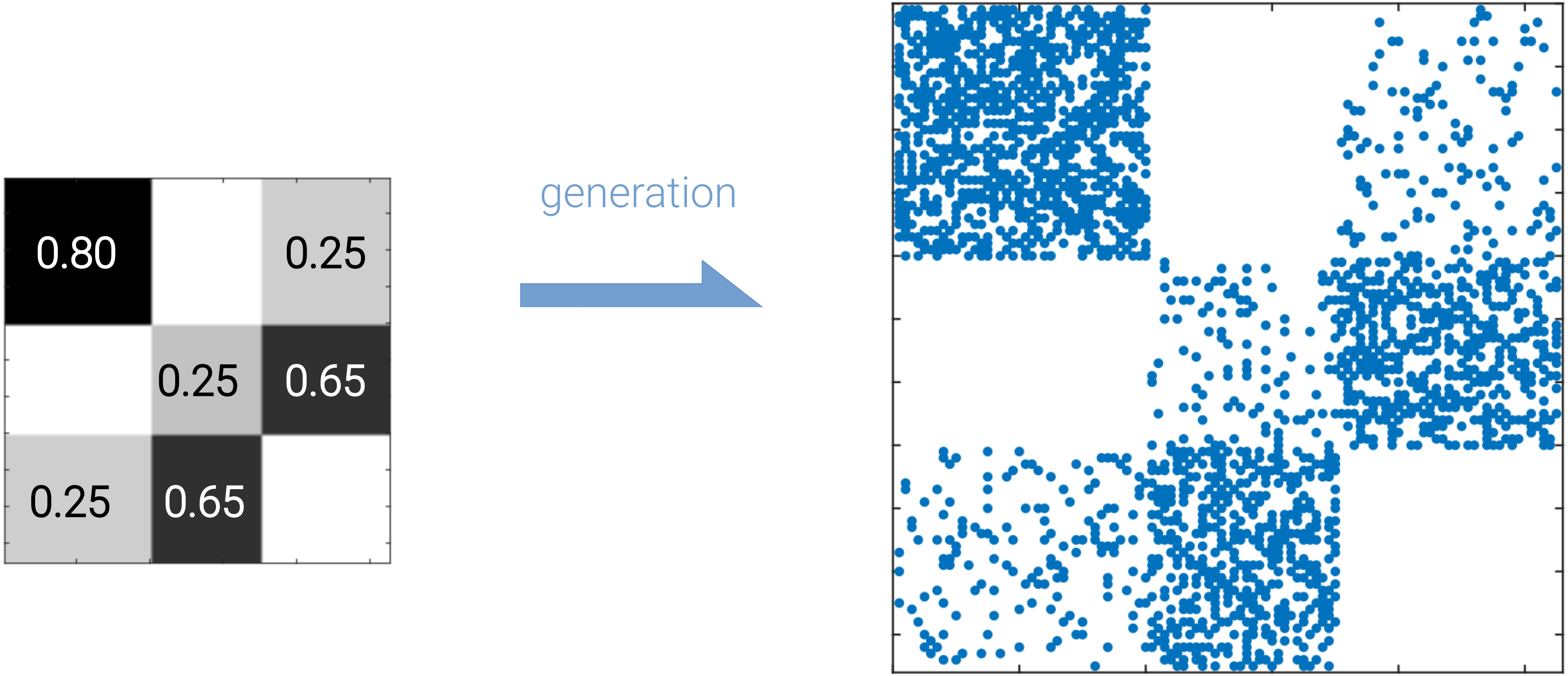
Mixing Matrix

Step 1 : Assign each node to a group

- Step 2 : Select some connection probabilities (mixing matrix)

- Step 3 : For each pair of nodes, add an edge with probability according to the group memberships

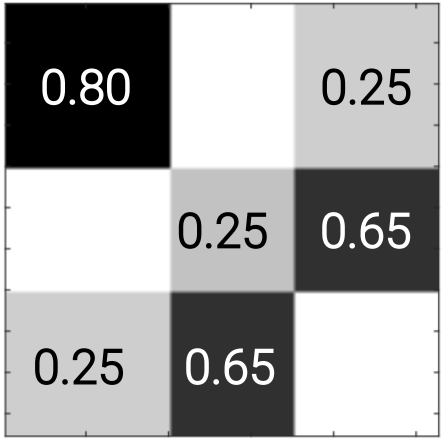
Generating a network using the SBM



Mixing Matrix

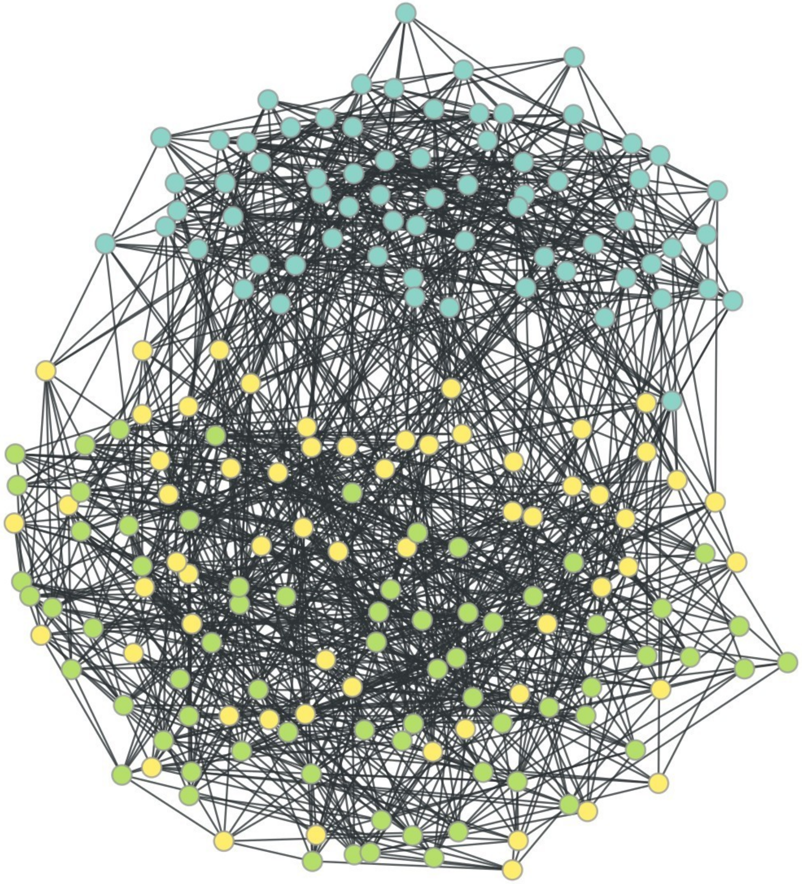
Adjacency Matrix

Generating a network using the SBM

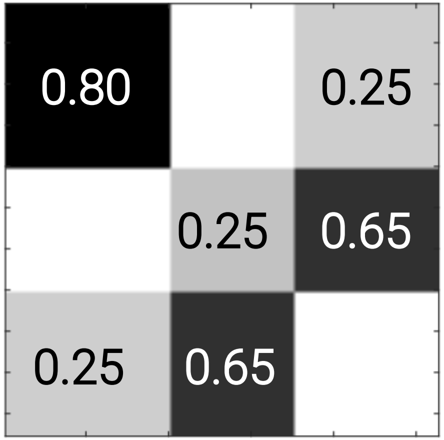


Mixing Matrix

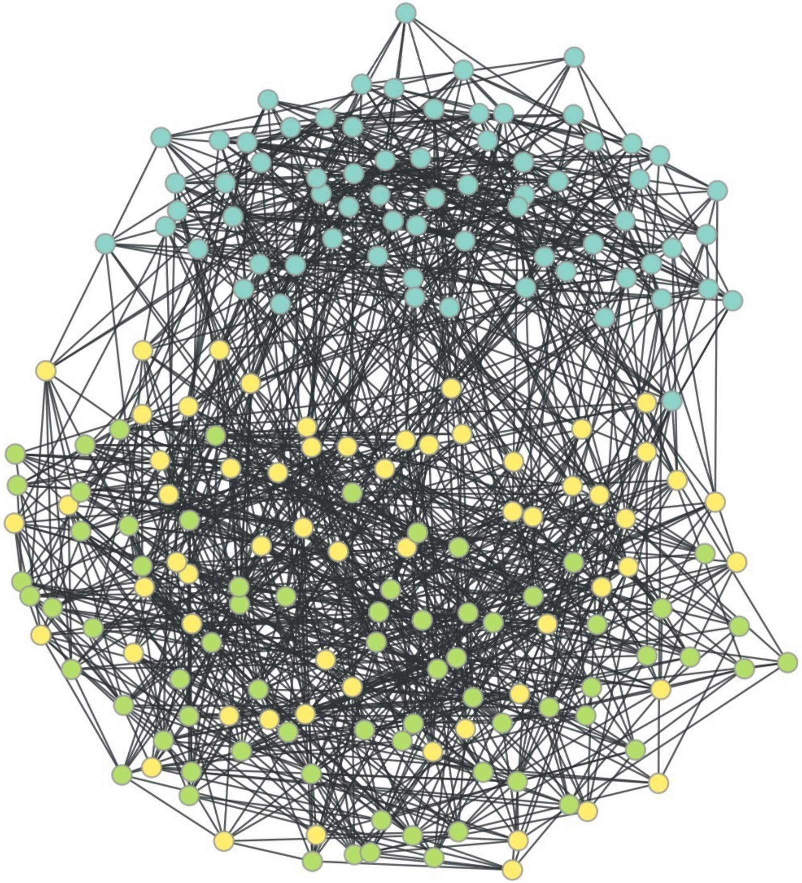
generation



Generating a network using the SBM

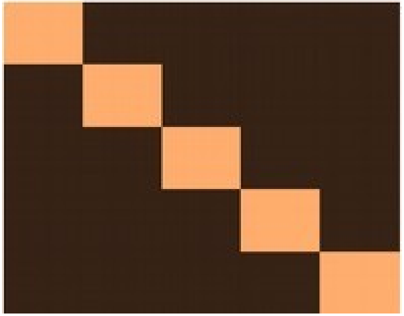


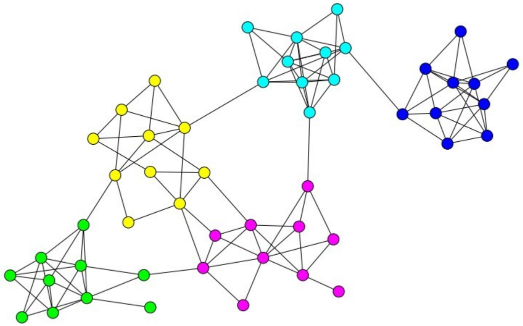
Mixing Matrix



Different types of structure

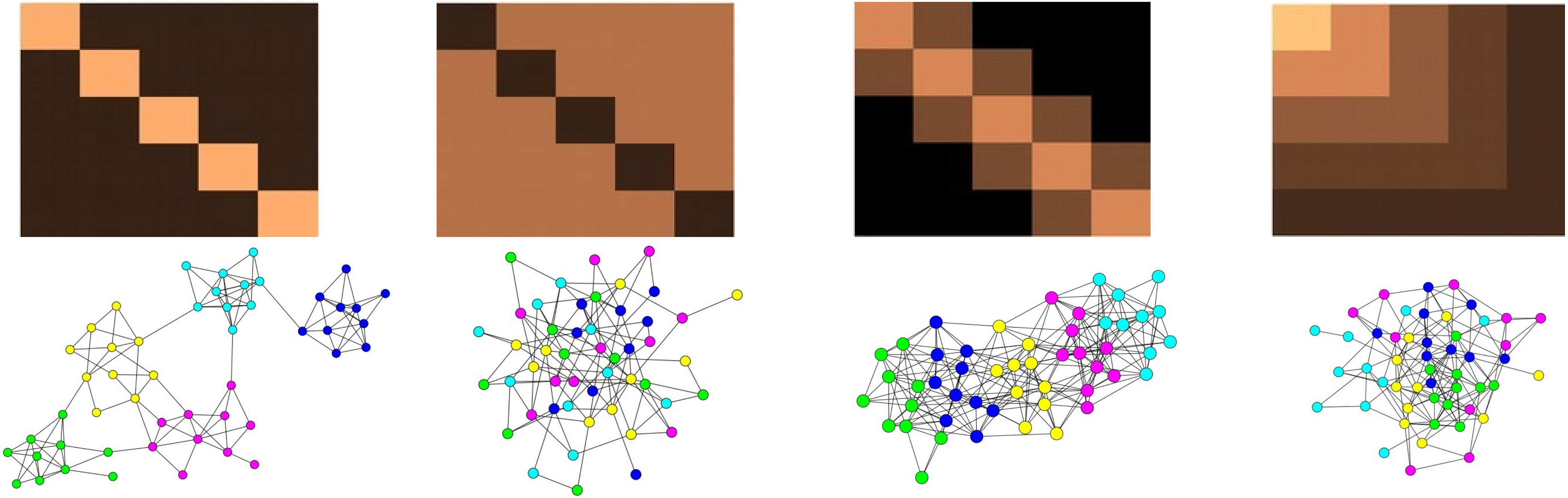
assortative





Different types of structure

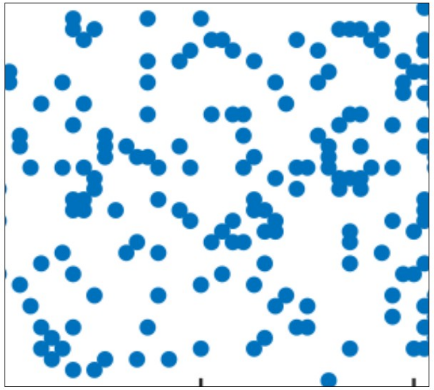
assortative | disassortative | ordered | core-periphery



Erdos-Renyi random graphs

Flip biased coin for every node pair

$$\Pr(A|\theta) = \prod_{ij} \theta^{A_{ij}} (1-\theta)^{(1-A_{ij})}$$



Adjacency Matrix

Heads or Tails?

Heads or Tails?

[0 1 0 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 1 1 1 1 1 1 0 1 1 0 0]

is just as likely as:

[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

What is the likelihood that we will observe this sequence of events?

s = [0 1 0 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 1 1 1 1 1 1 0 1 1 0 0]

What is the likelihood that we will observe this sequence of events?

s = [0 1 0 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 1 1 1 1 1 1 0 1 1 0 0]

$$\Pr(s_i = 1) = \theta = 0.5$$

(1-0.5) * 0.5 * (1-0.5) * 0.5 * (1-0.5) * 0.5 * (1-0.5) * (1-0.5) . . .

What is the likelihood that we will observe this sequence of events?

s = [0 1 0 1 0 1 0 0 1 0 0 0 0 0 1 0 0 0 1 1 1 1 1 1 1 0 1 1 0 0]

$$\Pr(s_i = 1) = \theta = 0.5$$

(1-0.5) * 0.5 * (1-0.5) * 0.5 * (1-0.5) * 0.5 * (1-0.5) * (1-0.5) . . .

$$\Pr(s|\theta) = \prod_i \theta^{s_i}(1-\theta)^{(1-s_i)}$$

What is the probability of heads?

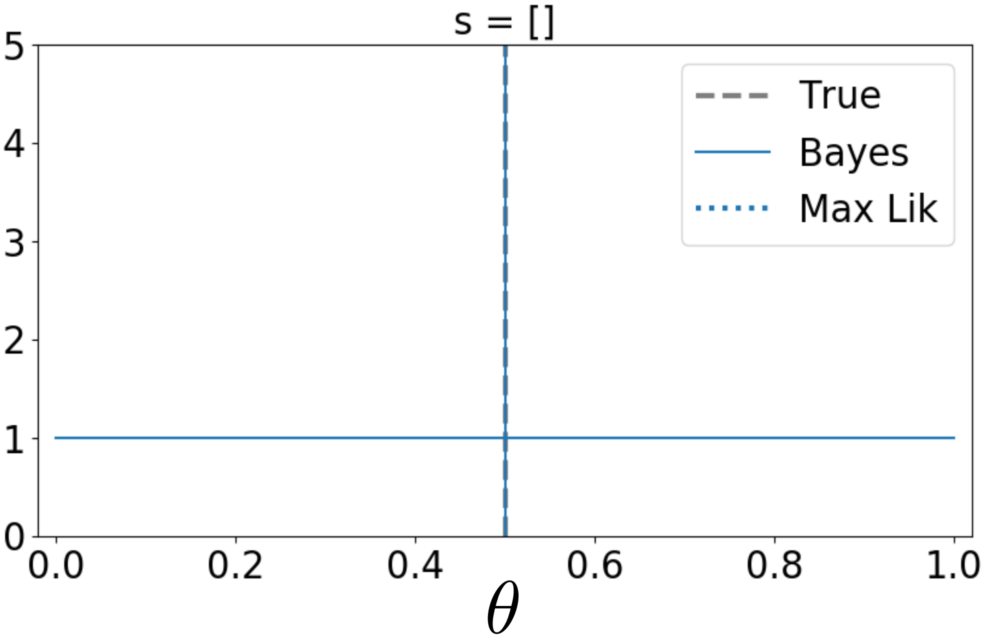
Probabilistic generative model

$$\theta_{\mathrm{ML}} = \max_{\theta} \mathrm{Pr}(s|\theta)$$

$$\theta_{\mathrm{Bayes}} \propto \mathrm{Pr}(s|\theta)\mathrm{Pr}(\theta)$$

What is the probability of heads?

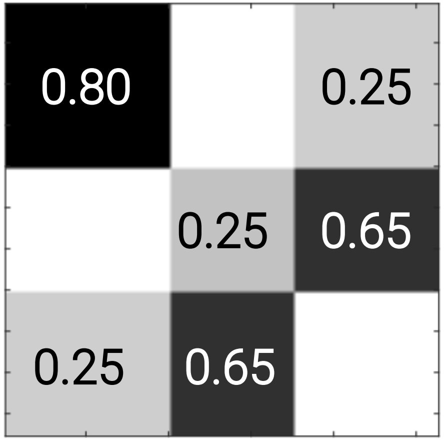
Probabilistic generative model



“I’ve never seen a coin before”

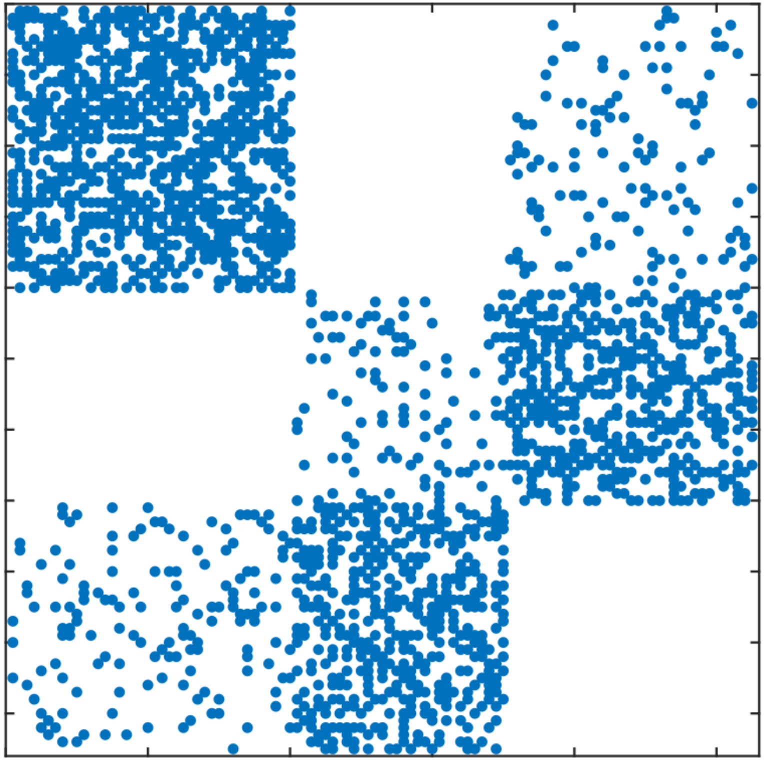
The stochastic block model

Just a bunch of coins!



Mixing Matrix

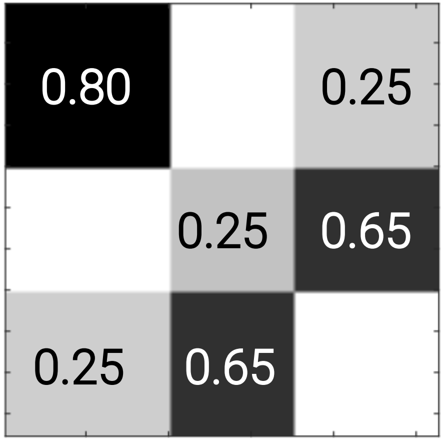
generation



Adjacency Matrix

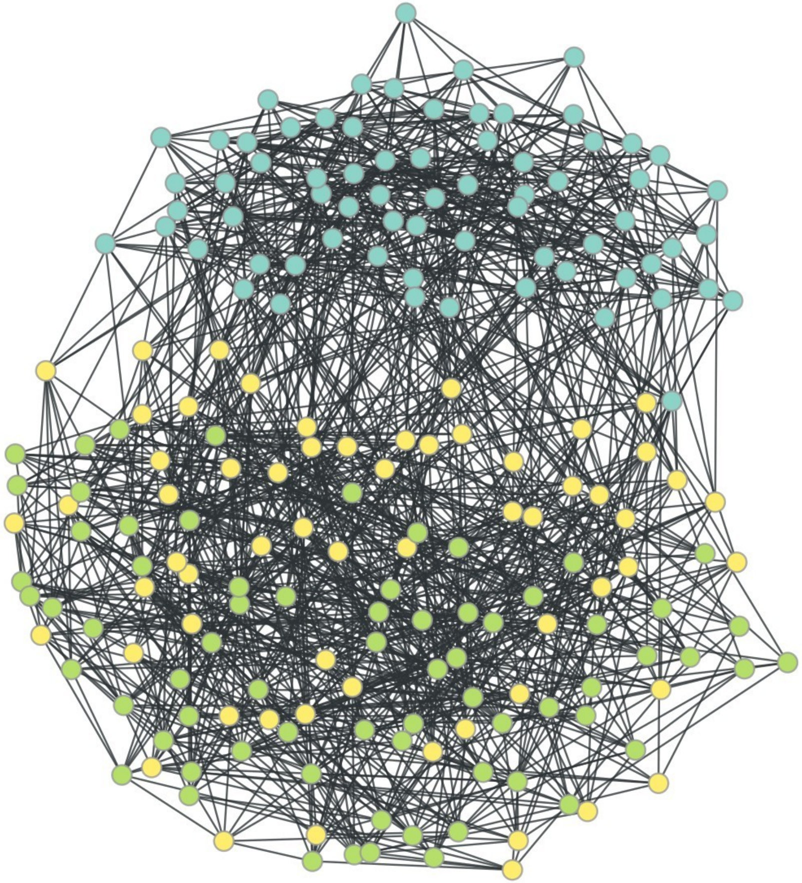
The stochastic block model

Just a bunch of coins!



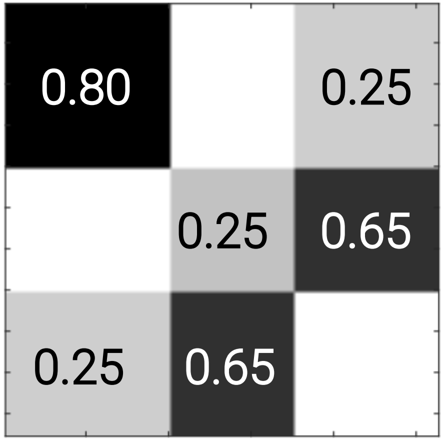
Mixing Matrix

generation



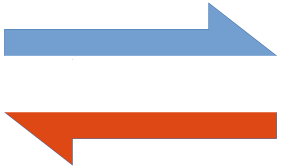
The stochastic block model

Just a bunch of coins!

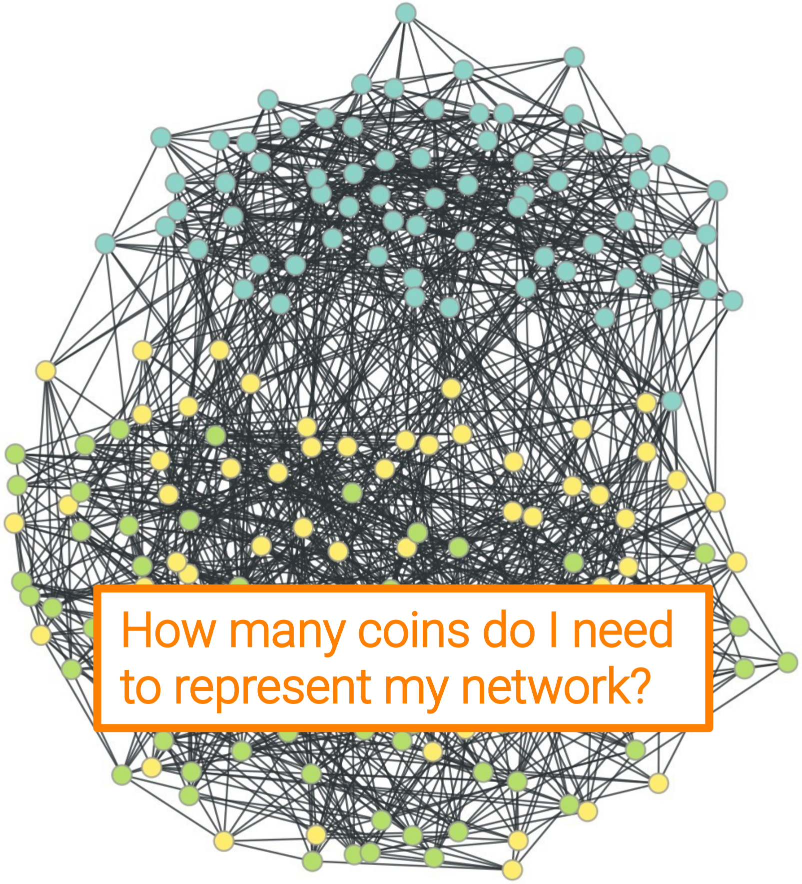


Mixing Matrix

generation



inference



How many coins do I need to represent my network?

Minimum description length

Description length
(entropy)

$$\Sigma = -\ln P(\boldsymbol{A}|\boldsymbol{\theta}, \boldsymbol{b}) - \ln P(\boldsymbol{\theta}, \boldsymbol{b})$$

Minimum description length

Description length
(entropy)

$$\Sigma = \underbrace{-\ln P(\boldsymbol{A}|\boldsymbol{\theta}, \boldsymbol{b})}_{\text{data description length}} - \ln P(\boldsymbol{\theta}, \boldsymbol{b})$$

Minimum description length

Description length (entropy)

$$\Sigma = \underbrace{-\ln P(\boldsymbol{A}|\boldsymbol{\theta}, \boldsymbol{b})}_{\text{data description length}} \underbrace{- \ln P(\boldsymbol{\theta}, \boldsymbol{b})}_{\text{model description length}}$$

Minimum description length

Description length (entropy)

$$\Sigma = -\underbrace{\ln \overbrace{P(\boldsymbol{A}|\boldsymbol{\theta},\boldsymbol{b})}^{\text{likelihood}}}_{\text{data description length}} - \underbrace{\ln \overbrace{P(\boldsymbol{\theta},\boldsymbol{b})}^{\text{prior}}}_{\text{model description length}}$$

Practical Q3

Many extensions and applications...

Link prediction

Network reconstruction

Many extensions

Many extensions and applications...

Link prediction

Network reconstruction

Many extensions

- degree correction
- mixed membership
- hierarchical
- edge weights/types
- node metadata
- temporal models

Many extensions and applications...

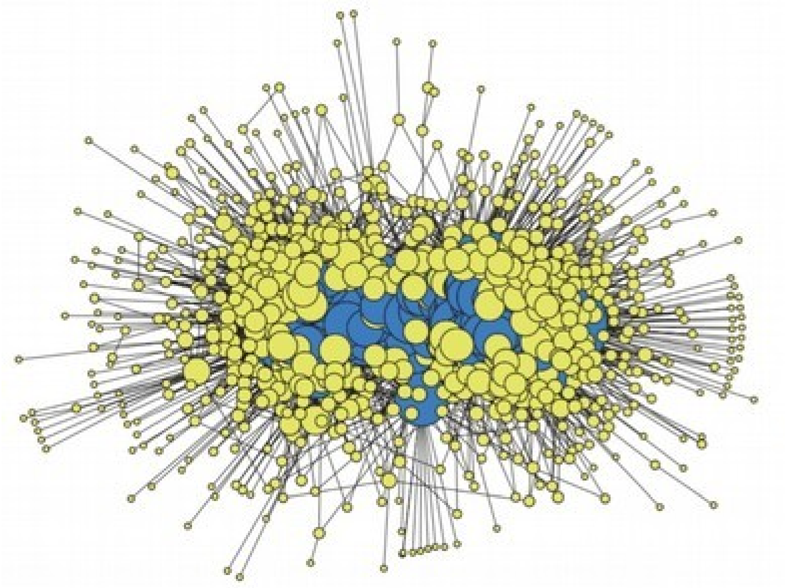
Link prediction

Network reconstruction

$\boxed{\text{Many extensions}}$

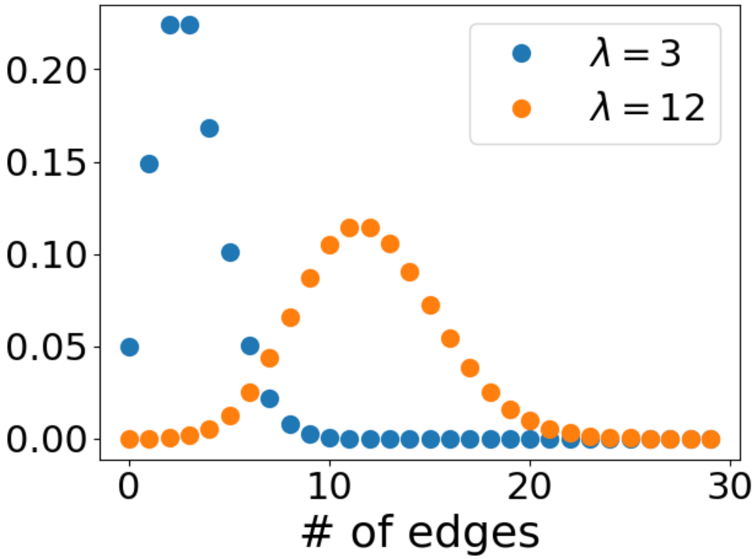
- degree correction
- mixed membership
- hierarchical
- edge weights/types
- node metadata
- temporal models

Degree-corrected SBM



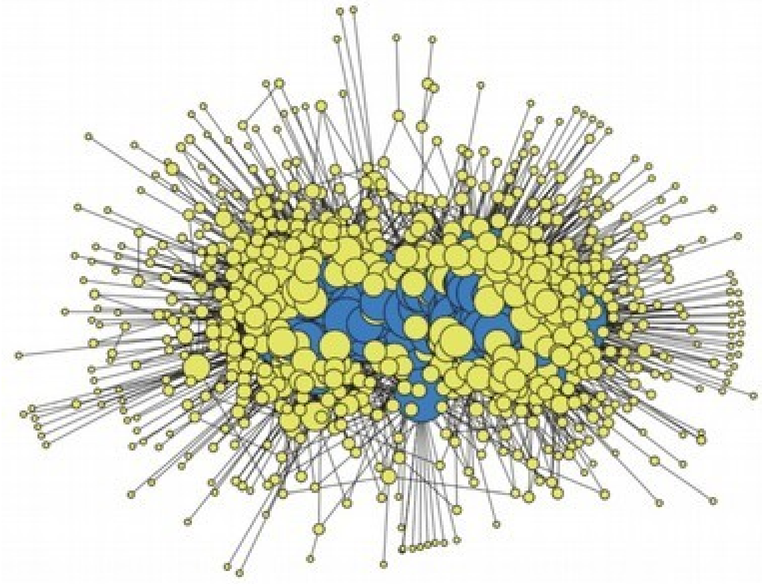
stochastic block model

SBM assumes Poisson distributed degr

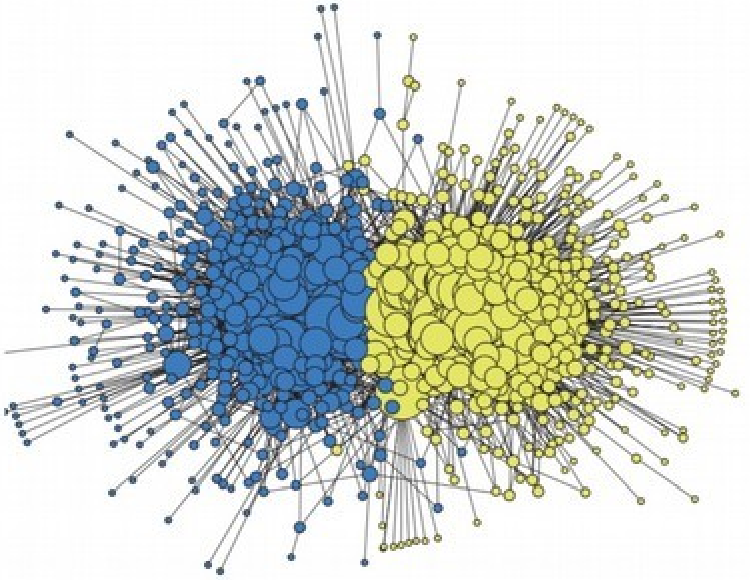


Karrer, Newman. Stochastic blockmodels and community structure in networks. Phys. Rev. E 83, 016107 (2011).
Adamic, Glance. The political blogosphere and the 2004 US election: divided they blog. 36–43 (2005).

Degree-corrected SBM

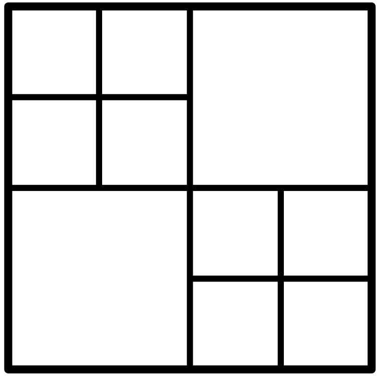


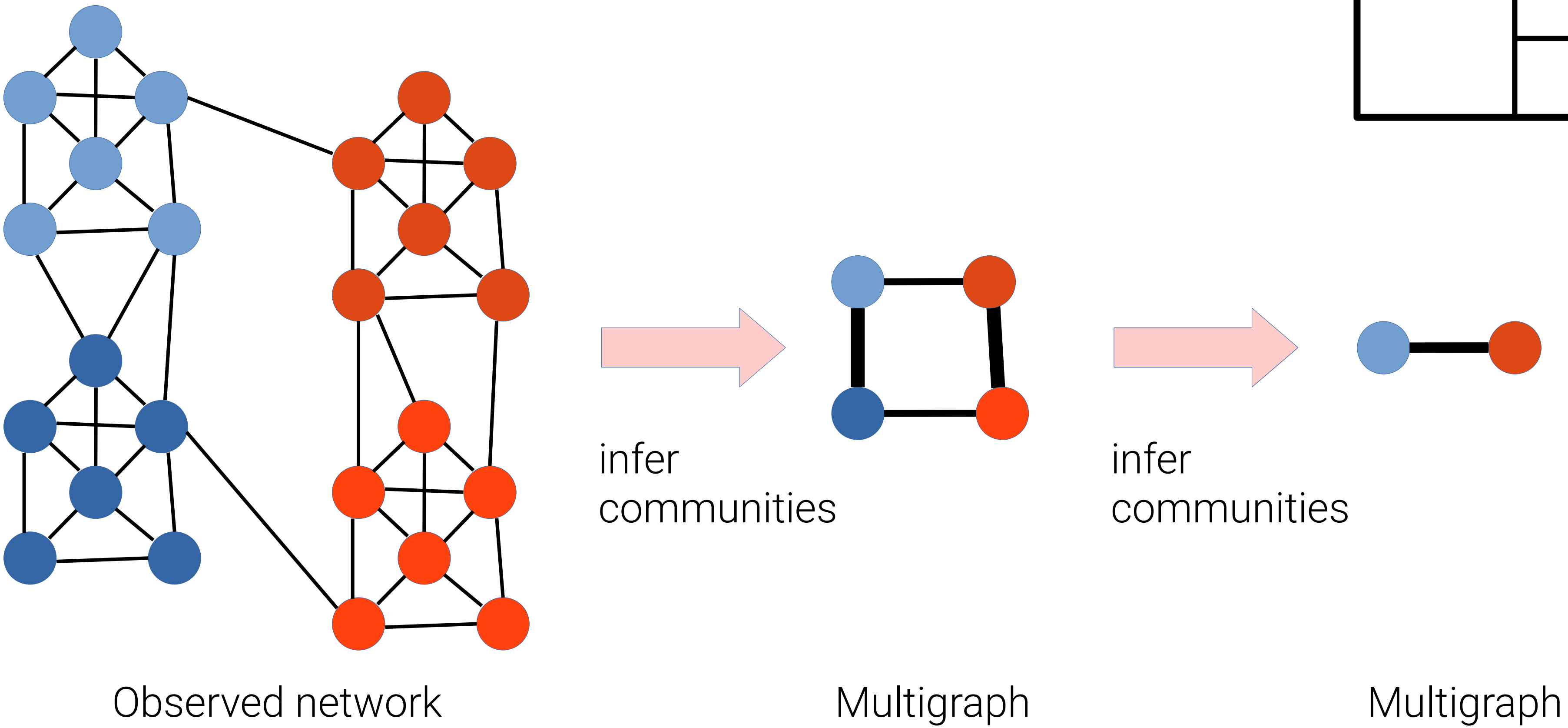
stochastic block model

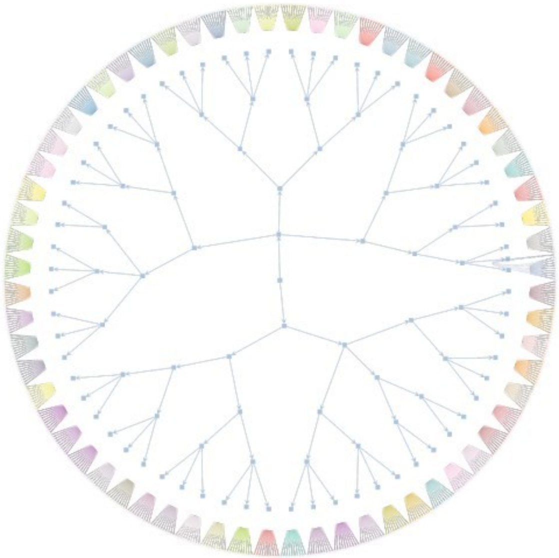


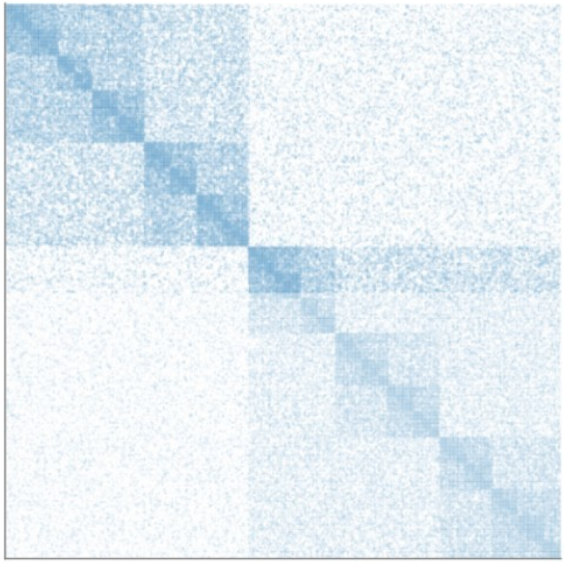
Karrer, Newman. Stochastic blockmodels and community structure in networks. Phys. Rev. E 83, 016107 (2011).
Adamic, Glance. The political blogosphere and the 2004 US election: divided they blog. 36–43 (2005).

Building the hierarchy



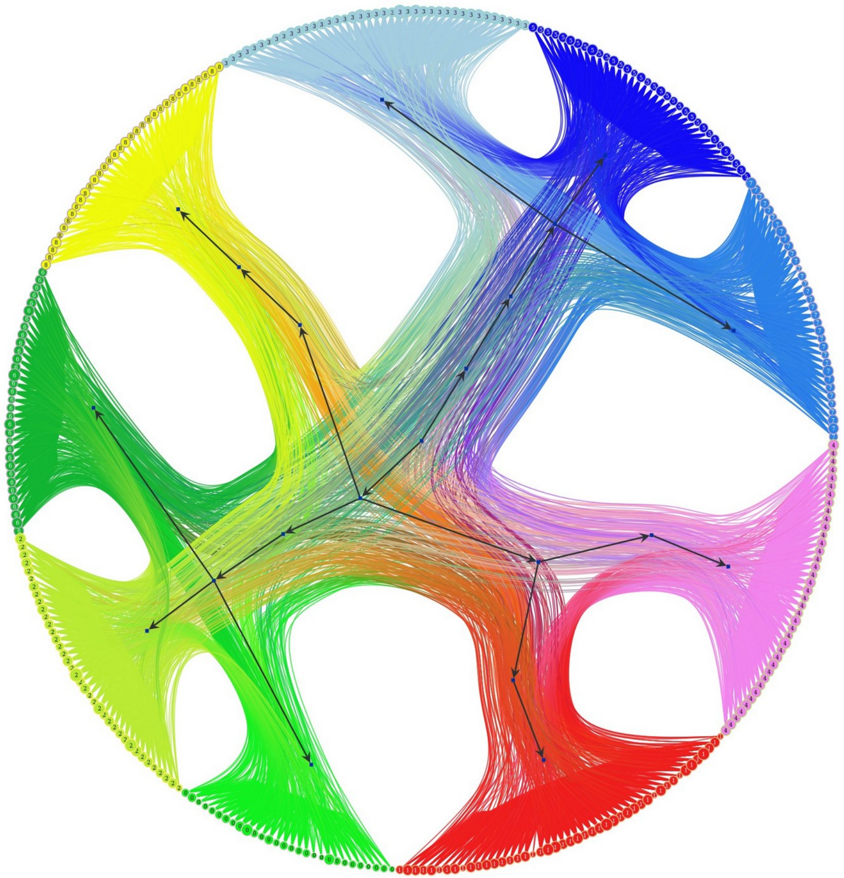




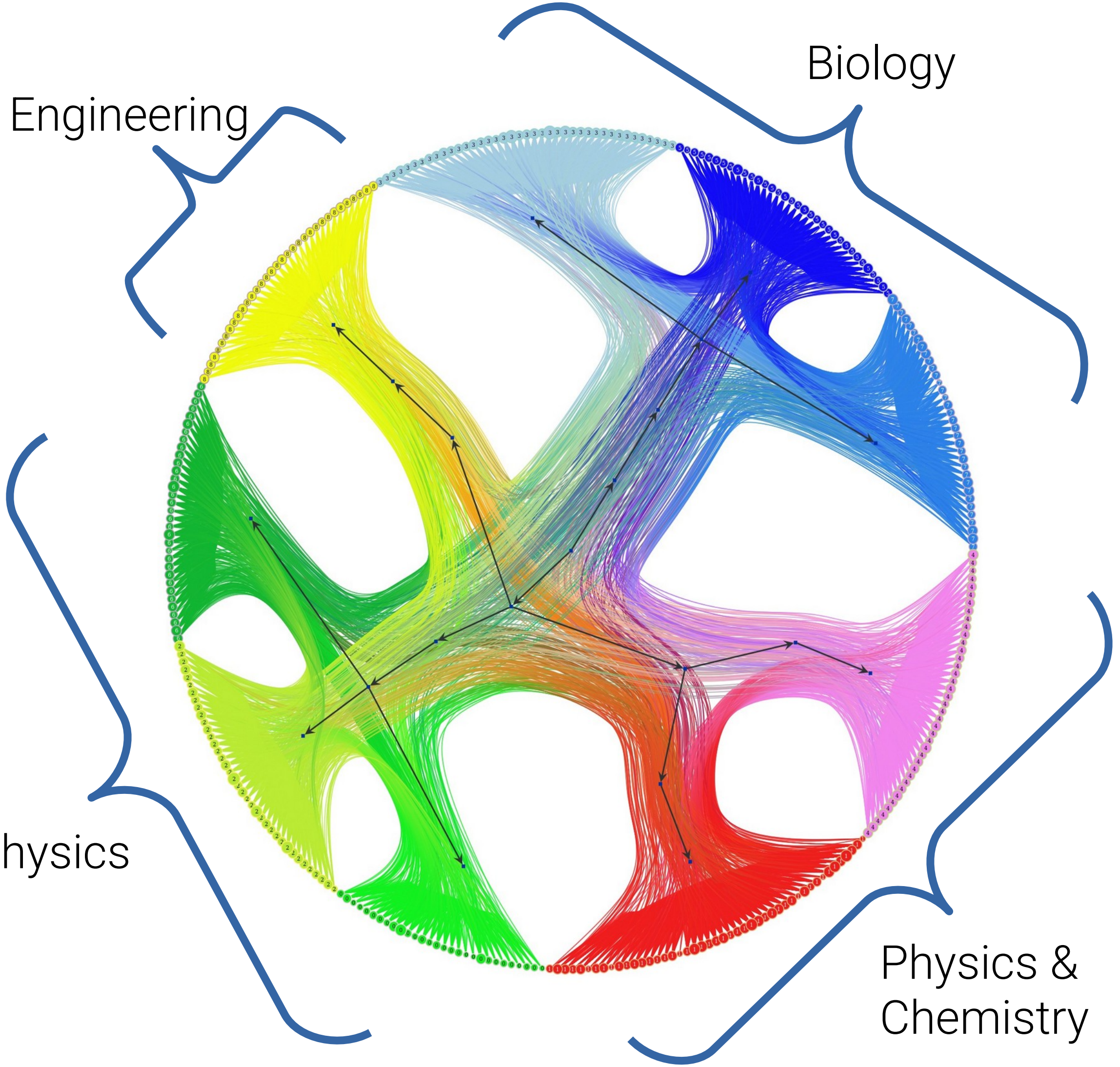


Adjacency matrix with hierarchy

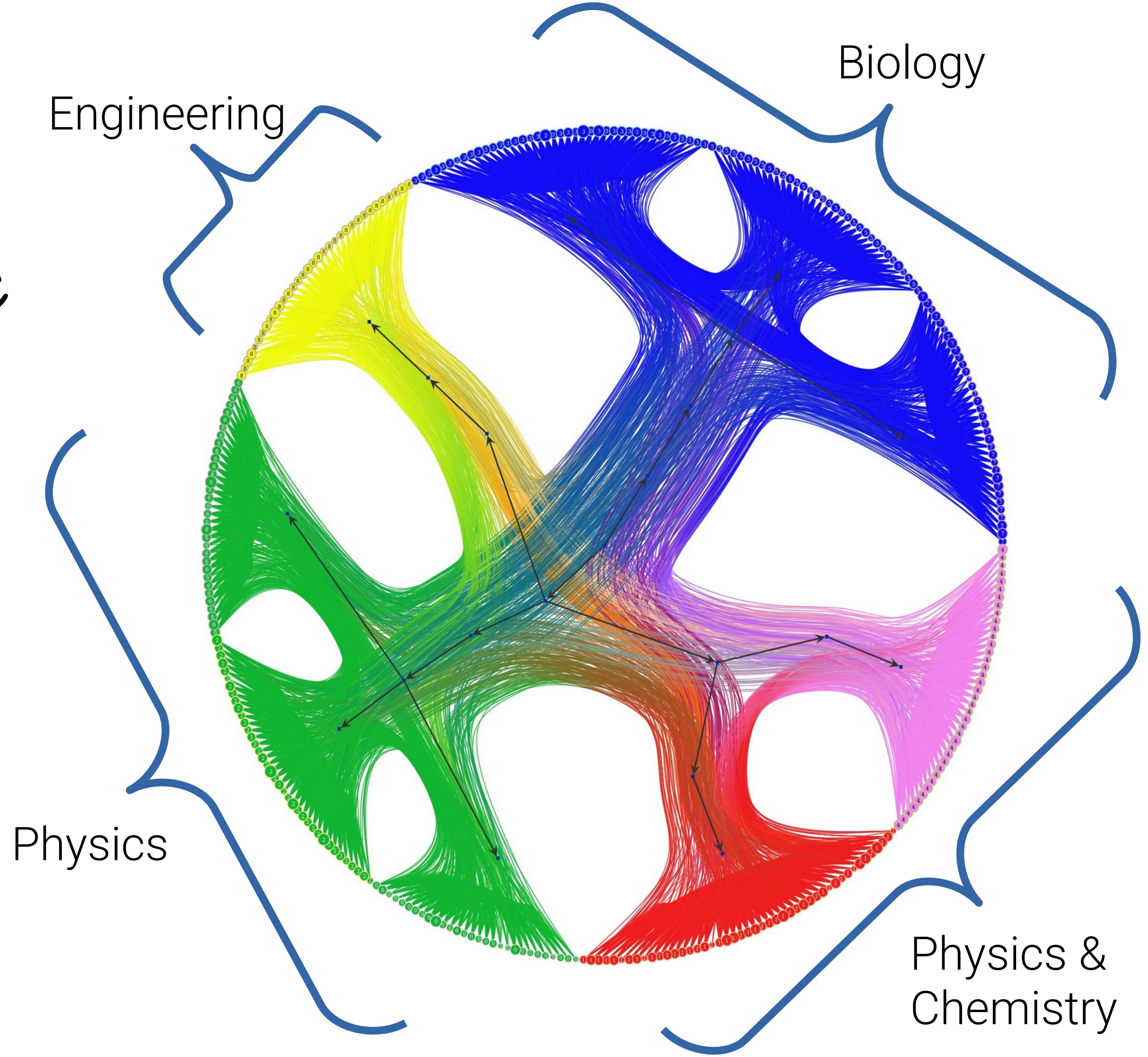
Face-to-face contacts



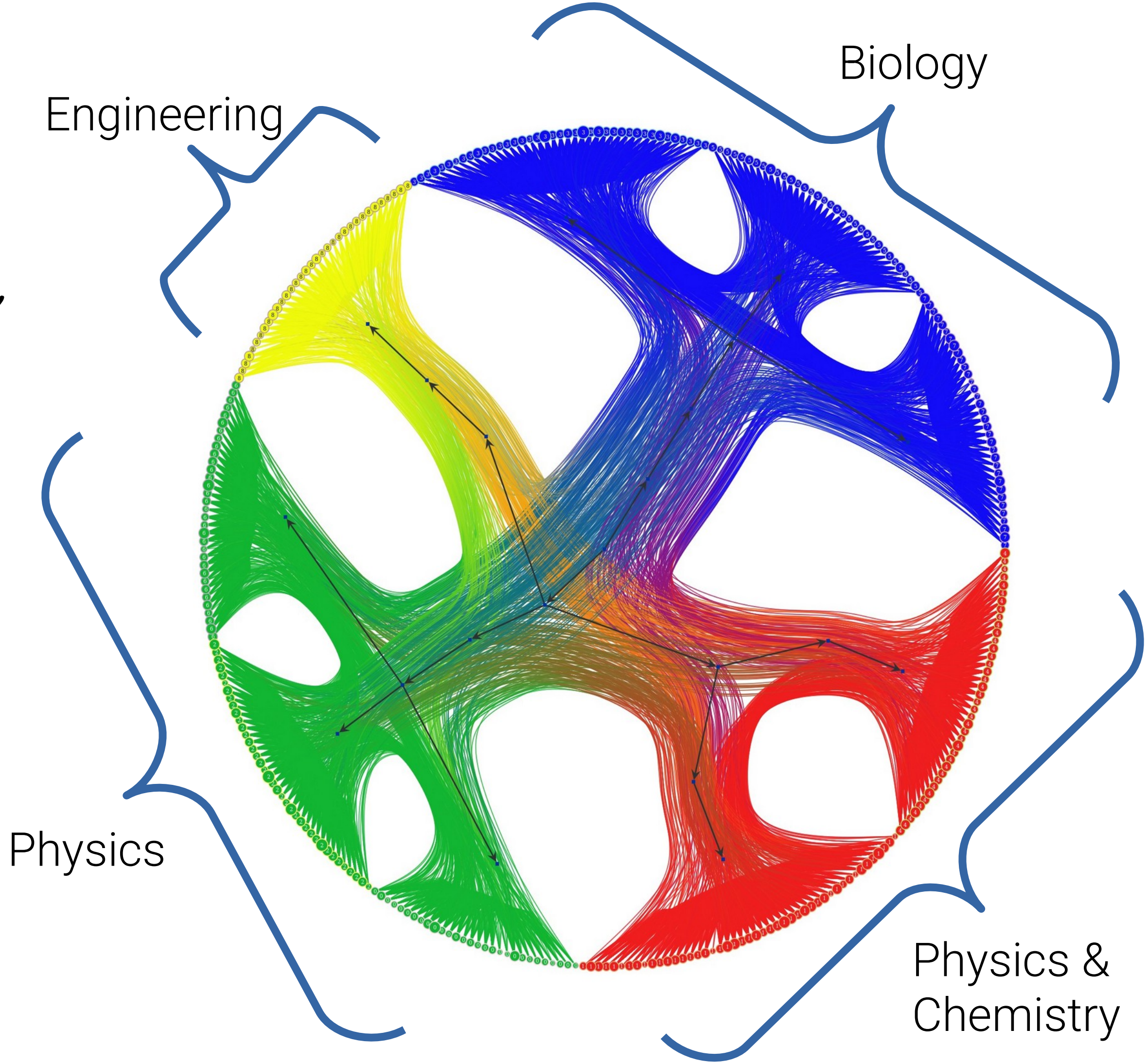
Face-to-face contacts



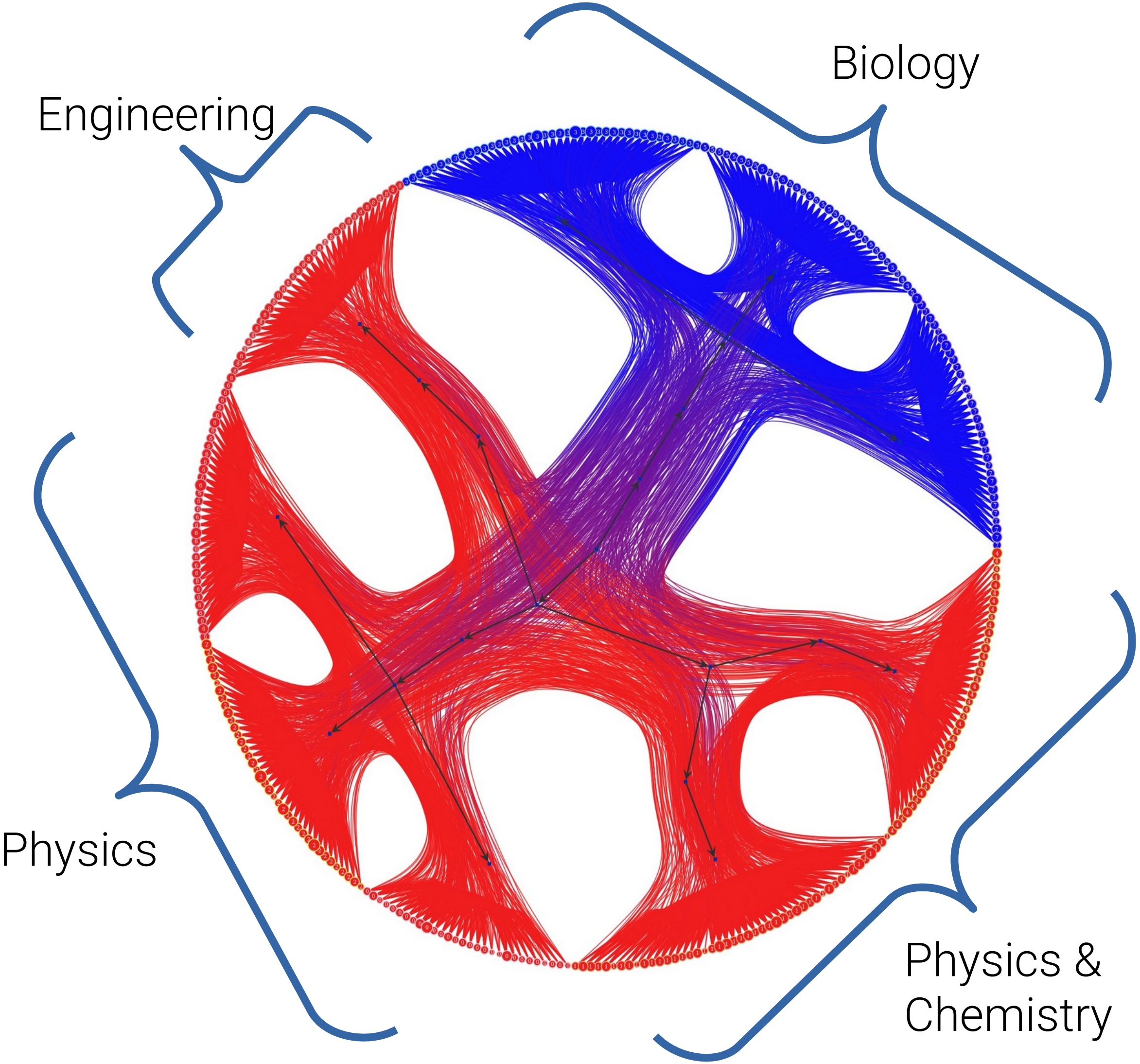
Face-to-face contacts



Face-to-face contacts



Face-to-face contacts

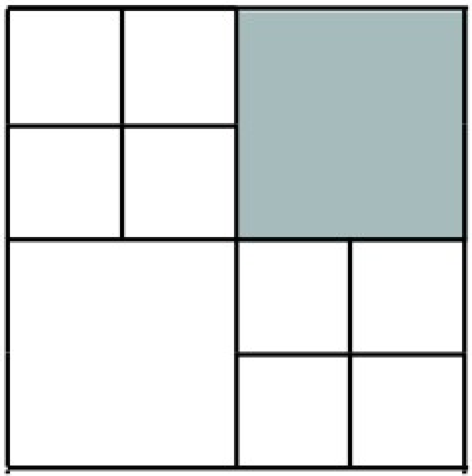


Practical Q4 and Q5

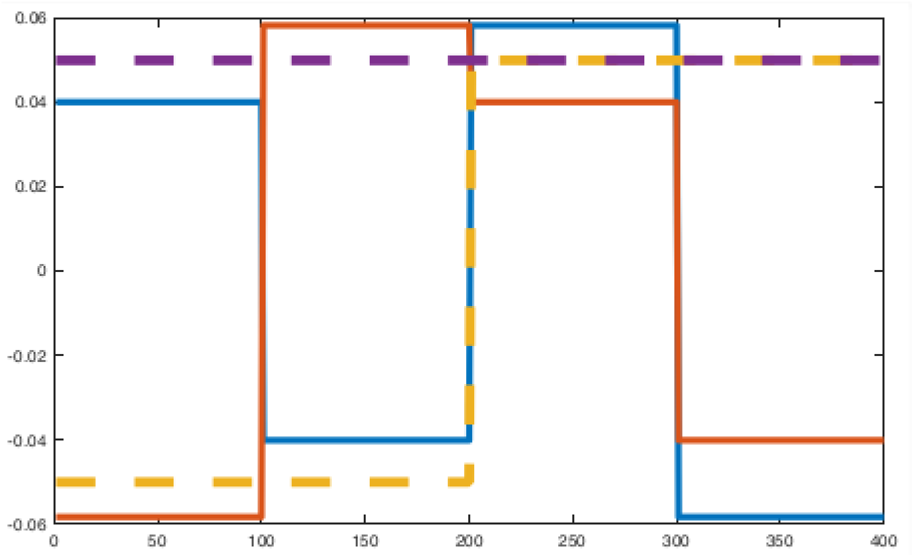
Spectral clustering

Spectral properties

$\mathbb{E}[A]$

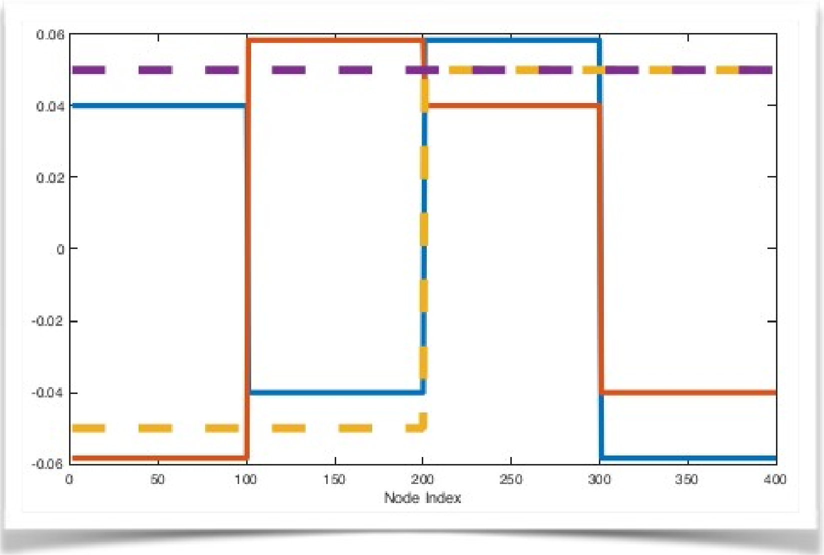


First 4 Eigenvectors of the Laplacian

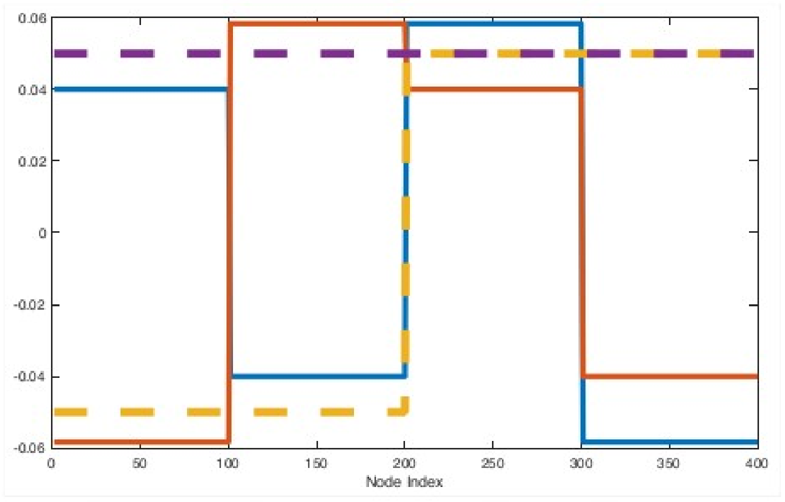


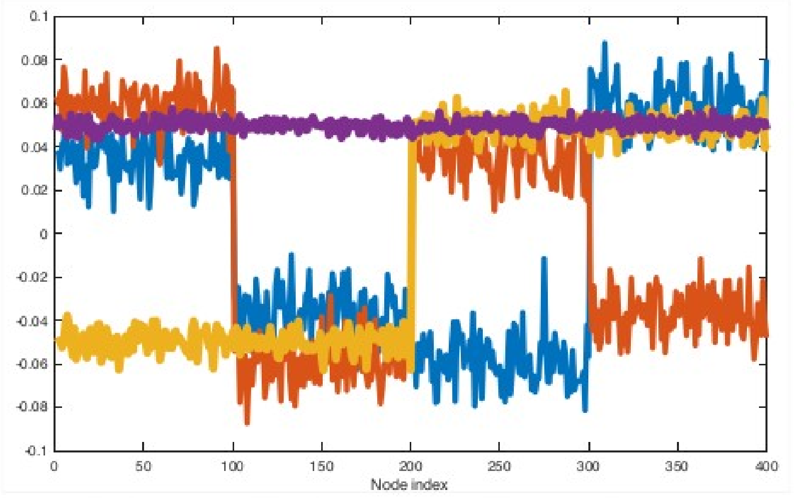
Node index

If we could just “see” the expected adjacency matrix, then we could just look for constant eigenvectors



If we could just “see” the expected adjacency matrix, then we could just look for constant eigenvectors





K-means clustering

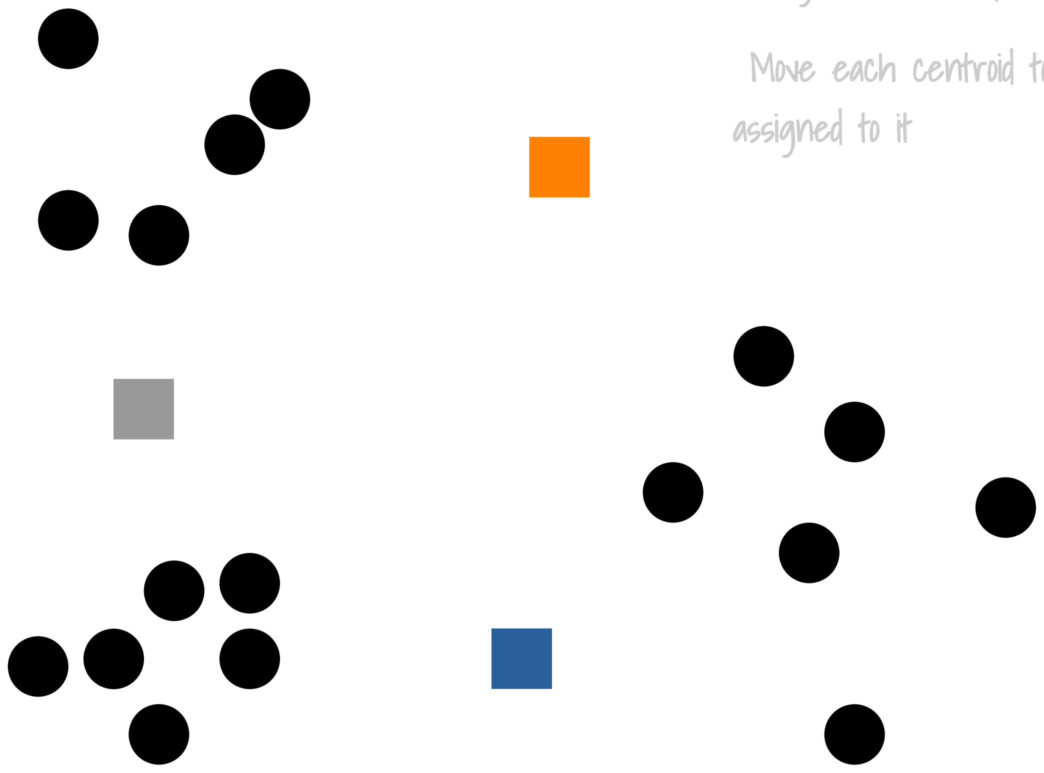
1) Randomly initialise centroids (one per cluster)
2) Iterate:
    a) Assign each data point to the nearest centroid
    b) Move each centroid to the mean of the data points assigned to it

Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it

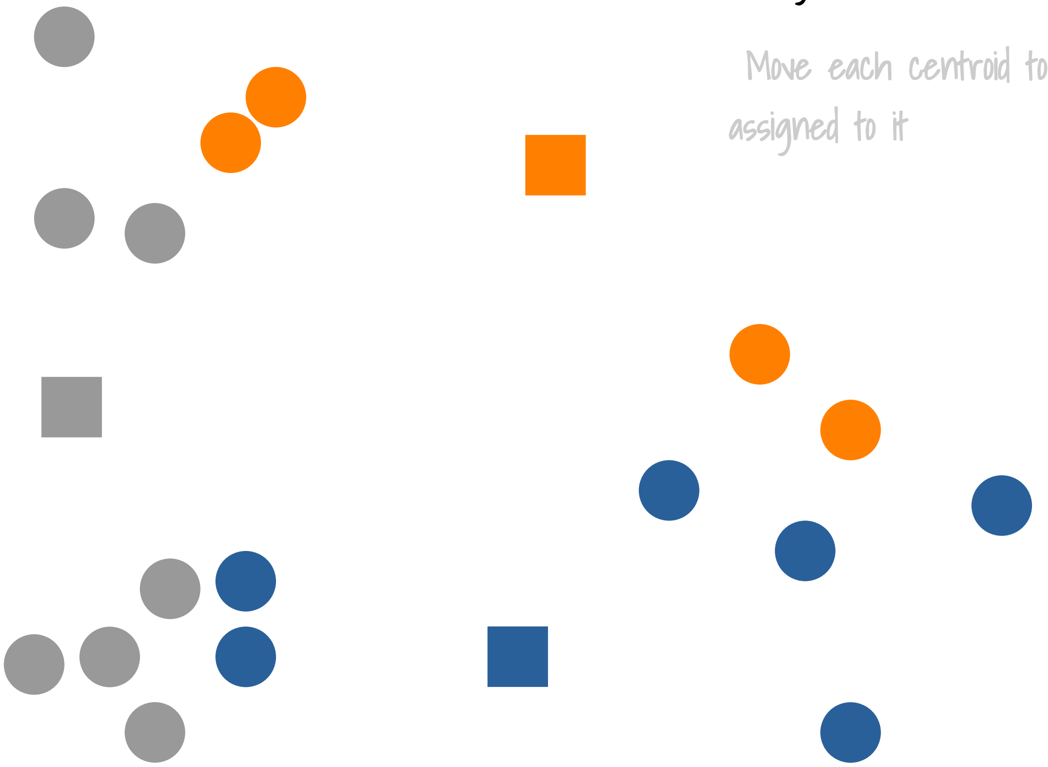


Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it

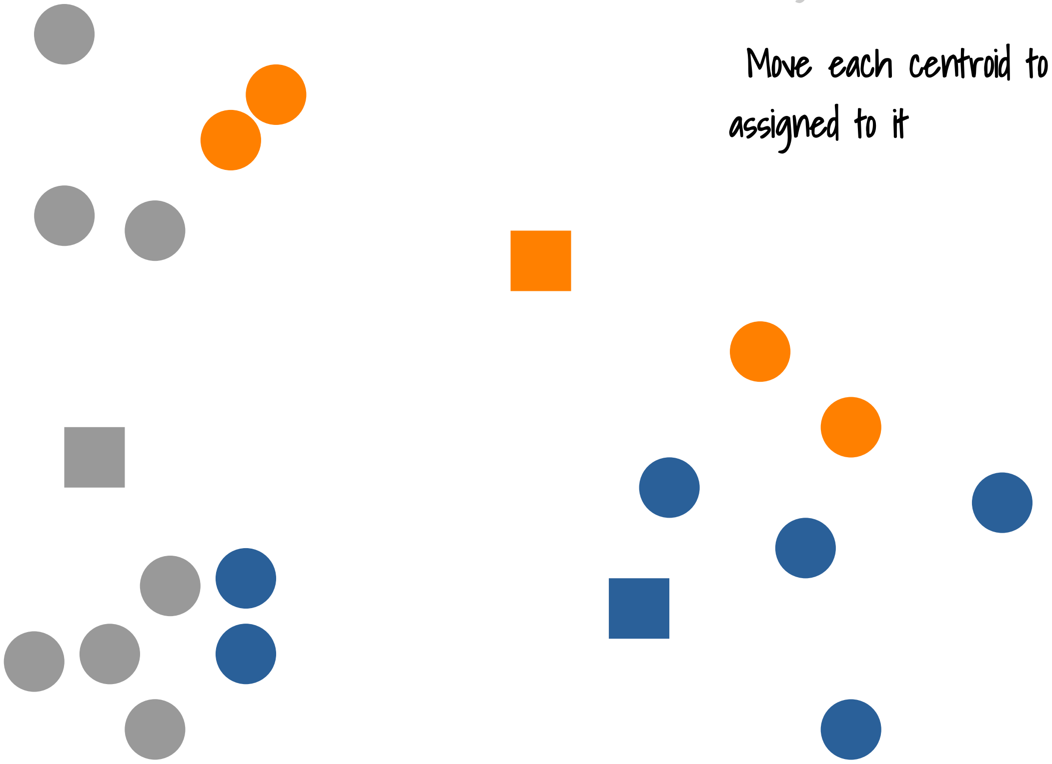


Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it

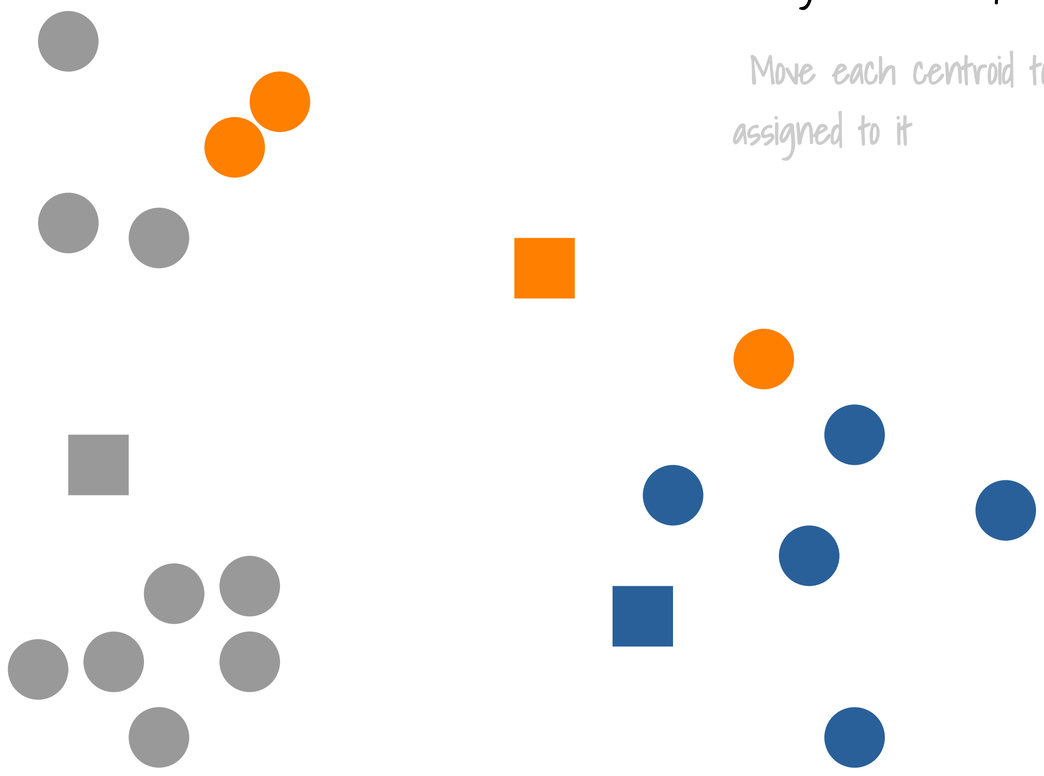


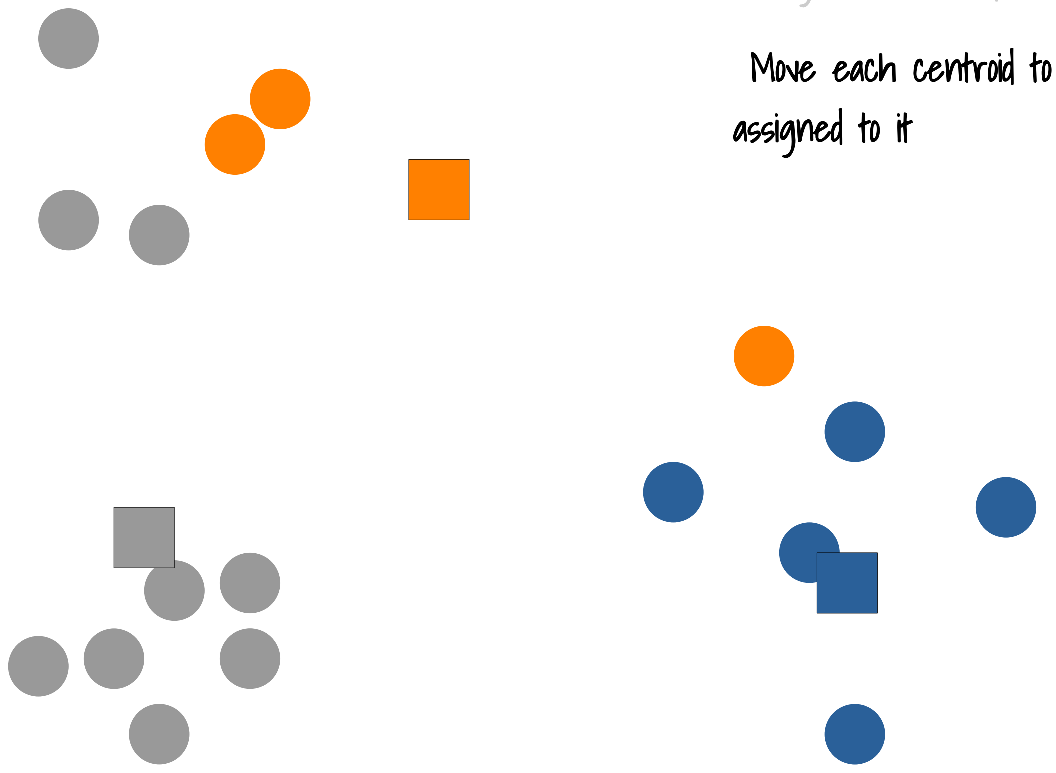
Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it



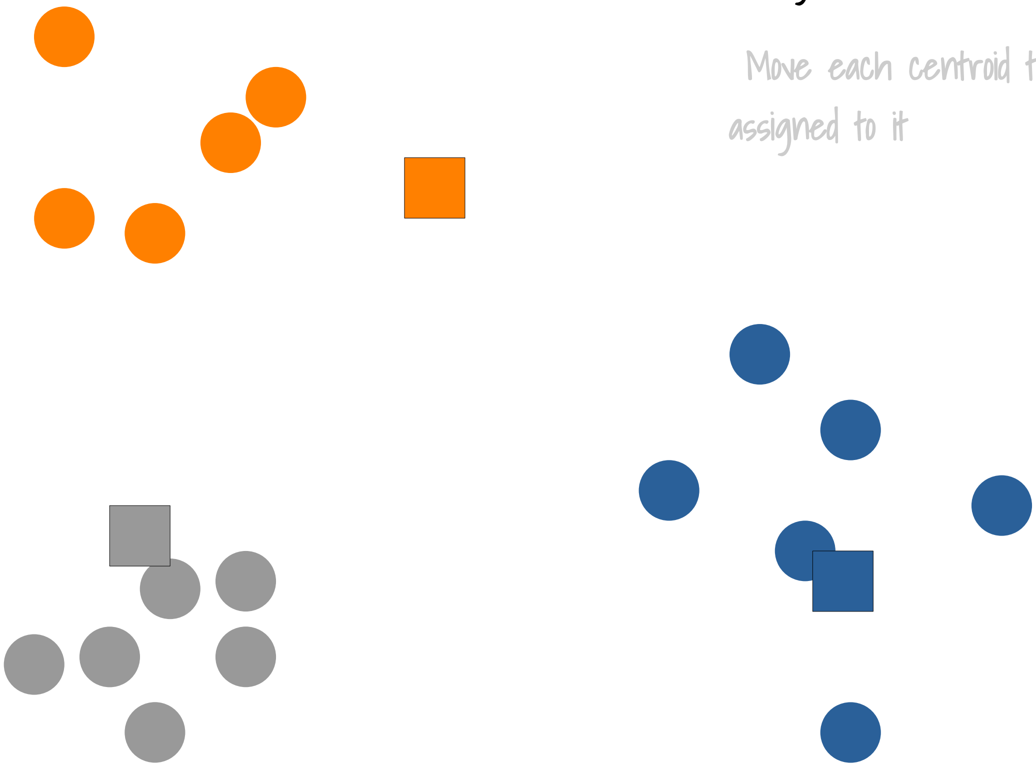


Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it



Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

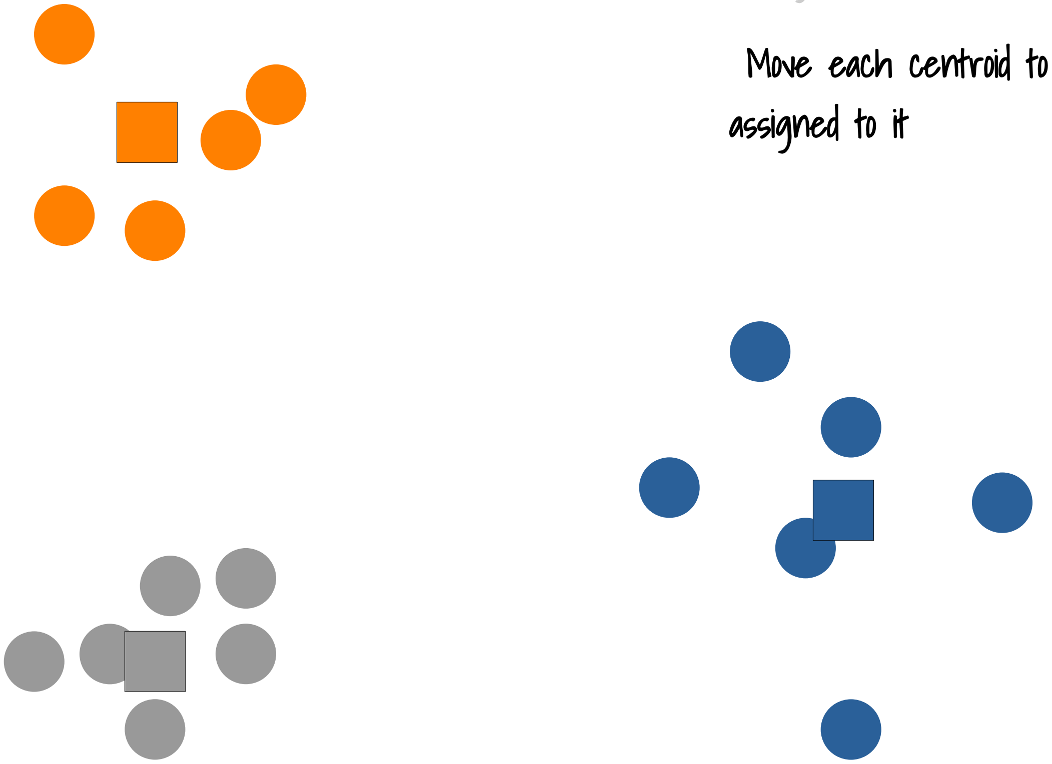
Move each centroid to the mean of the data points assigned to it

Randomly initialise centroids (one per cluster)

Iterate:

Assign each data point to the nearest centroid

Move each centroid to the mean of the data points assigned to it



Summary...

Summary...

Community detection finds nodes with similar connectivity patterns

- Nodes often have similar properties or functions

Summary...

Community detection finds nodes with similar connectivity patterns

- Nodes often have similar properties or functions

There can be multiple good ways to partition a network

Summary...

Community detection finds nodes with similar connectivity patterns

- Nodes often have similar properties or functions

There can be multiple good ways to partition a network

It's unsupervised!

- Your model must know how to partition the network