Quick Review

Community detection: identifying relatively dense regions in a network

→ # edges internal vs. external to the community

{[Issue: global max is one b.g comm.

Algs: agglomerative vs. divisive

→ Modularity

- One of the most widely-used approaches for C.D.
- Explicitly captures a metric of how "modular" a network is with regards to a random

→ usually used for optimization or relative comparison

Basic hypothesis: random networks lack inherent community structure

↳ so, we can measure how clustered our network is relative to what's randomly expected (null hypothesis)

How do we calculate modularity?

$$M = \frac{1}{2m} \sum_{\forall u,v \in C} \left(A_{uv} - \frac{d(u)d(v)}{2m} \right)$$

↑ random graph attachment probabilities

M = modularity

$m = |E(G)|$

$\forall u,v \in C$ = vertex pairs in a community

A_{uv} = # edges between u, v or sum of the weights of edges

$d(u)$ = degree of u

$\frac{d(u)d(v)}{2m}$ = expected # edges between u and v in random graph

↳ good approx. for loopy multigraphs

↳ good approx. for loopy multigraphs

NOT so good for simple graphs
(more later)

Modularity Maximization

We can maximize modularity as a community detection algorithm

→ Usually: greedy agglomerative

Newman Algorithm

- Greedy agglomerative algorithm
- Initially: all vertices in unique communities
- Iterate while #comms > 1:
  - Merge community pair that maximizes modularity gain

Pros: Capture the hierarchy
good quality in practice
good theoretical basis
can be fast

Cons: there are issues with modularity calculations in practice (discussed soon)

Louvain Algorithm
- same as above, but we do explicit edge contractions
- can be faster

Note: these are not "optimal"

Note 2: most networks don't have a modularity peak
↳ more of a plateau

Issues with modularity

Resolution limit: we can't resolve relatively small communities
↳ How small?

Change in modularity by combining communities A and B

$$\Delta M = \frac{l_{AB}}{} - \frac{k_A k_B}{}$$

$$\Delta M = \frac{l_{AB}}{m} - \frac{k_A k_B}{2m^2}$$

l_{AB} = edges between A and B

k_A = sum of degrees of vertices in A

Consider:

$$\frac{l_{AB}}{m} = \frac{k_A k_B}{2m^2}$$

$$l_{AB} = \frac{k_A k_B}{2m}$$

if $l_{AB} > \frac{k_A k_B}{2m}$ => we gain from merging A, B

assume $k_A = k_B = k$

$$l_{AB} = 1$$

$$1 > \frac{k^2}{2m}$$

$$2m > k^2$$

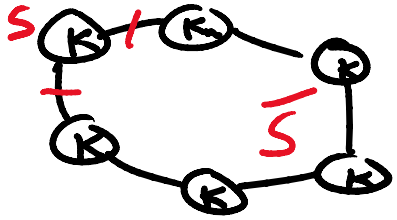
$$\sqrt{2m} > k$$

so we merge A and B if $k \leq \sqrt{2m}$

if $K \leq \sqrt{2m}$

This is a lower bound on the size of community that modularity maximization can find

→ The "ring of cliques" graph highlights this quite well



Issue: goes against intuition of what a community

Why this isn't huuuge problem:

- Most algorithms are hierarchical
  → so we're still likely to have small communities we can observe
- A wide spread in real community sizes could be problematic
  ↳ we have "multi-resolution" approaches

The other problem: $\frac{d(w)\,d(v)}{2m}$

- Most social nets are simple

$2m$

- Most social nets are simple
- The above approximation can be bad
  * Especially problematic for dense graphs, skewed, or dense subgraph
- What happens when $2m << d(v)d(w)$
  → in multigraph → $\frac{d(v)d(w)}{2m}$ is expected # edges
  → in simple → nonsense

Takeaway: our modularity value can be meaningless in skewed or dense networks

Fix: we could use actual attachment probabilities

Issue: No closed-form way to calculate

(100,000 pt Bonus: figure the above out)

Issue: empirically calculating in practice is computationally difficult

Conductance

Generally: a measure of how quickly a random walk converges to a stationary state

Lower conductance => more defined communities

We generally conductance relative to some edge cut, S and $\bar{S}$

$$\text{conductance}(S) = \frac{\text{cut}(S)}{\min(K_S, K_{\bar{S}})}$$

$\text{cut}(S)$ = number of edges in cut

K_S = sum of degrees in S

How to apply to C.D.:

Issue: we only define conductance relative to a single

→ we can consider local cuts

* Spectral methods (more later)
* Personalized PageRank

* Personalized PageRank

Challenge: we need good seed vertices

challenge 2: we need community sizes

Note: we aren't explicitly optimizing conductance