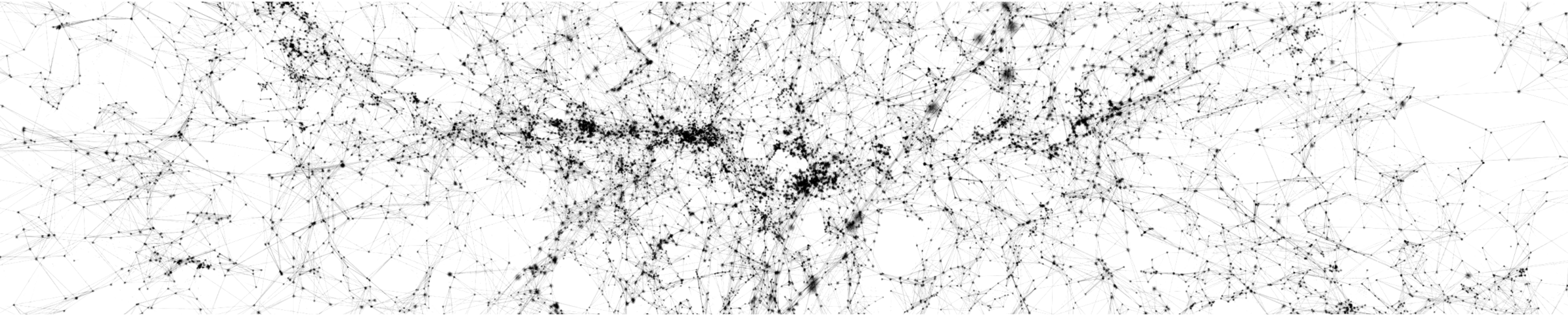
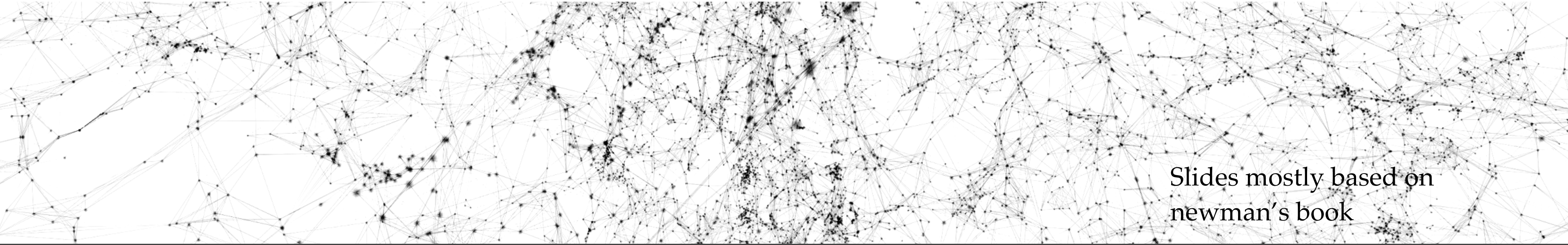


Modules

Analysis of complex interconnected data



Slides mostly based on newman's book

Quick Recap of Centrality Measures

- Degree Centrality
    - count the number of neighbours, ignores their importance

$$x_i = \sum_{j \in N(i)} 1$$

- Eigenvalue Centrality
    - consider importance but gives zero to nodes not in scc or its out component, in extreme case of an acyclic networks, e.g. citation networks, all nodes get zero score

$$x_i = \alpha \sum_{j \in N(i)} x_j \text{ , } \alpha = \frac{1}{\lambda^*(A)}$$

, $\lambda^*(A)$: largest eigenvalue of A

- Katz Centrality
    - avoid zeros by giving everyone a basic importance

$$x_i = \alpha \sum_{j \in N(i)} x_j + 1 \text{ , } \alpha < \frac{1}{\lambda^*(A)}$$

- PageRank
    - divide importance on how many connections it is passed over to

$$x_i = \alpha \sum_{j \in N(i)} \frac{x_j}{d_j} + 1 \text{ , } \alpha < \frac{1}{\lambda^*(A)}$$

- HITS
    - consider two types of importance, hubs and authorities

$$x_i = \alpha \sum_{j \in N(i)} y_j \text{ , } y_i = \beta \sum_{j \in N(i)} x_j \text{ , } \alpha\beta = \frac{1}{\lambda^*(A\,A^\top)}$$

- Closeness centrality
    - average how close you are to the rest

$$x_i = \frac{1}{n-1} \sum_j \frac{1}{s_{ij}}$$

, s_{ij}: length of shortest path from i to j

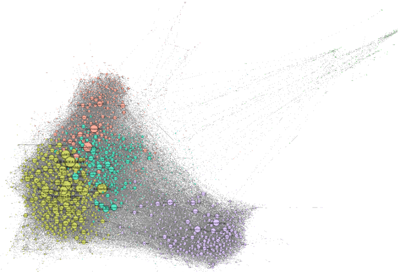
- Betweenness centrality
    - count what fraction of shortest paths pass through you

$$x_i = \frac{1}{n^2} \sum_{jk} \frac{|i \in s_{jk}^i|}{|s_{jk}|}$$

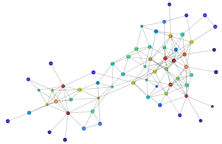
, s_{ij}: set of shortest path from i to j

Outline

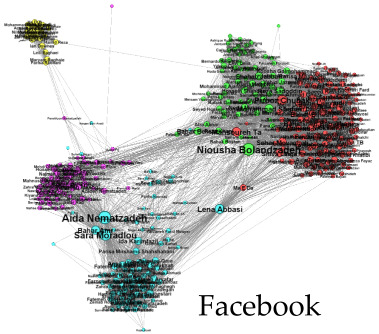
- Quick Recap of Centrality Measures
- **Modules**
    - Real graphs are modular
    - Spectral clustering
    - Objectives for quality of a module
    - TopLeaders
    - Using Betweenness Centrality
    - Modularity Optimization, FastModularity & Louvain
    - Resolution limits of Modularity
    - Link clustering
    - Evaluating clustering results



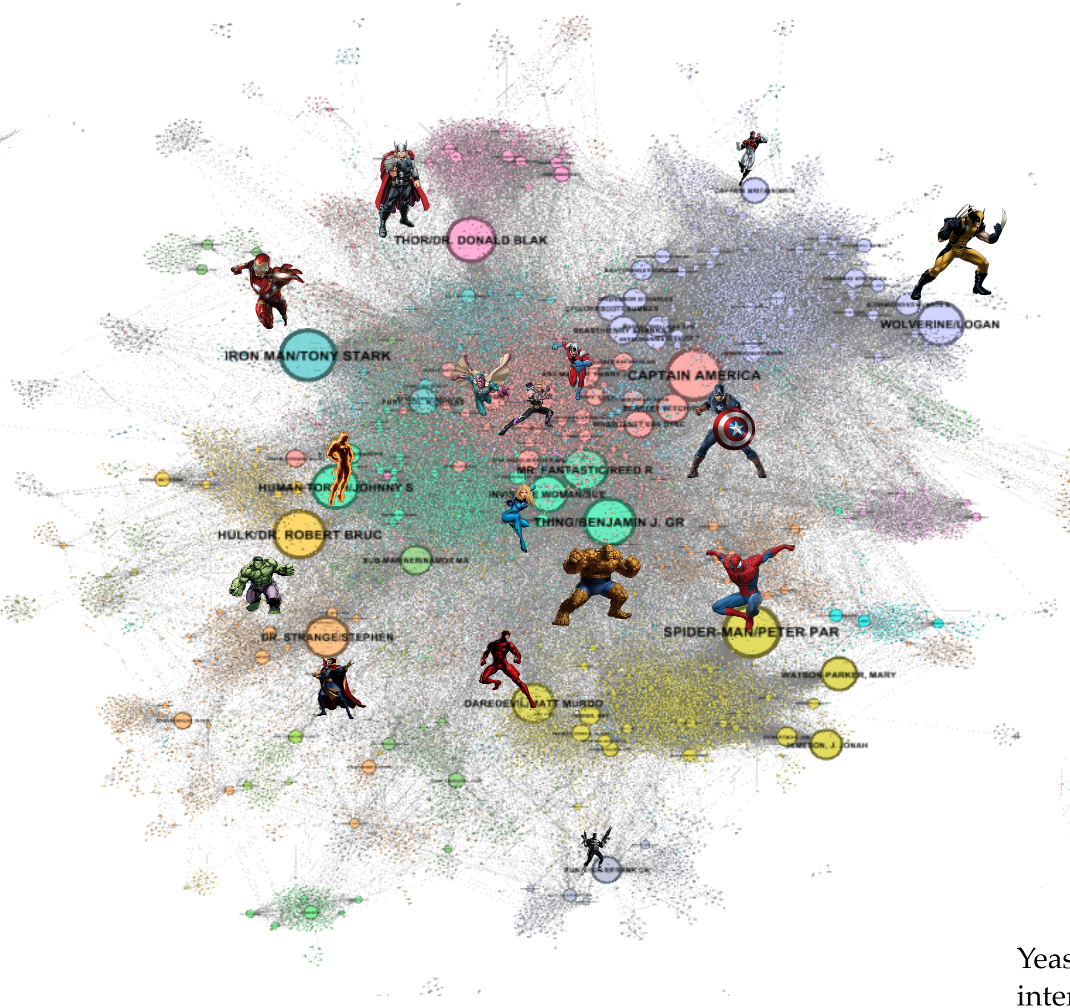
Twitter

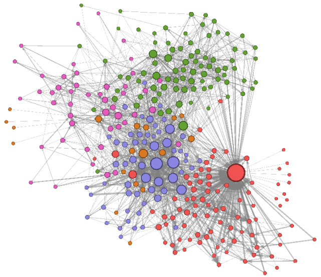


Dolphins

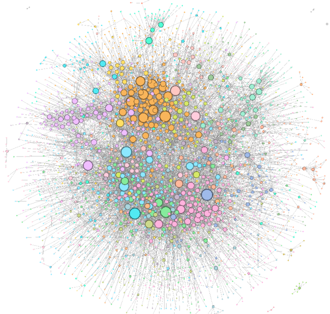


Facebook



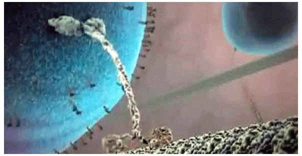


C. elegans neural network



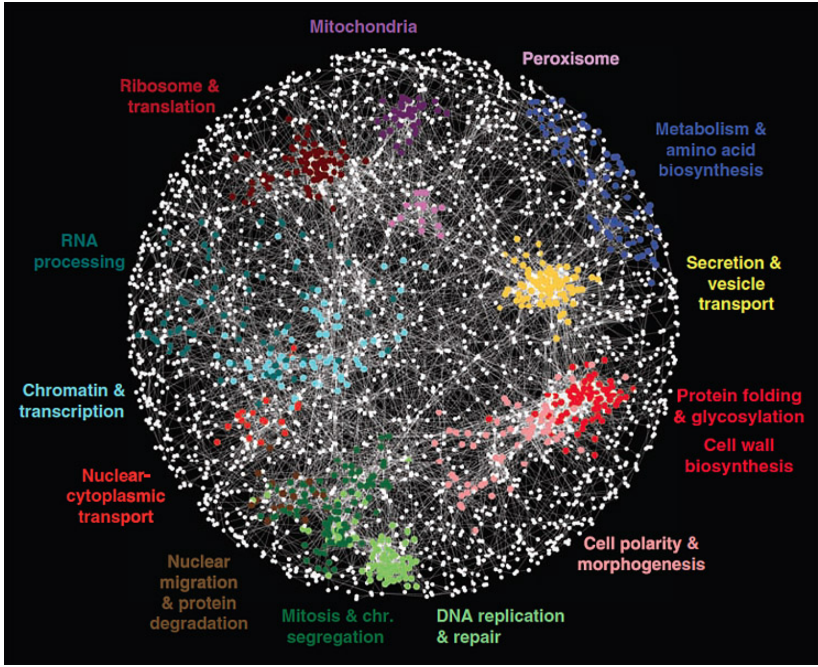
Yeast protein protein interaction networks

Example Applications



Module identification in biological networks

- Protein complexes and functional modules in PPI networks (Spirin & Mirny, PNAS 2003)
  - protein complexes: proteins that interact to carry out a task as a single complex unit, e.g., RNA splicing
  - functional units: proteins that bind at different time to participate in a cellular process, e.g., communicating a signal from the surface of the cell to the nucleus
- Representation of the metabolic networks (R Guimerà & Amaral, Nature 2005)
  - ultra-peripheral metabolites (that have all their connections inside their modules) have the highest evolutionary loss rate, whereas connector hubs (that connect to most of the other modules) are the most conserved across the species

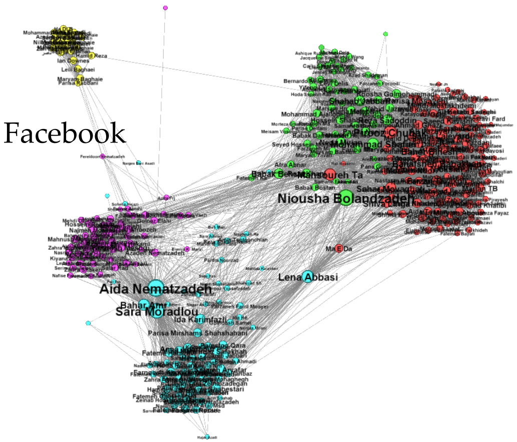

Mitochondria
Peroxisome
Ribosome & translation
Metabolism & amino acid biosynthesis
RNA processing
Secretion & vesicle transport
Chromatin & transcription
Protein folding & glycosylation
Cell wall biosynthesis
Nuclear-cytoplasmic transport
Cell polarity & morphogenesis
Nuclear migration & protein degradation
Mitosis & chr. segregation
DNA replication & repair

Modules as Coarse Representation

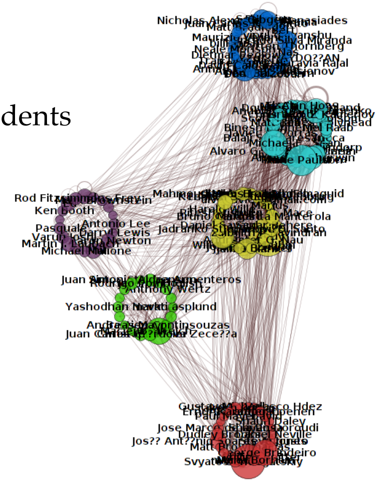
Modules give a coarse-grained representation of the structure

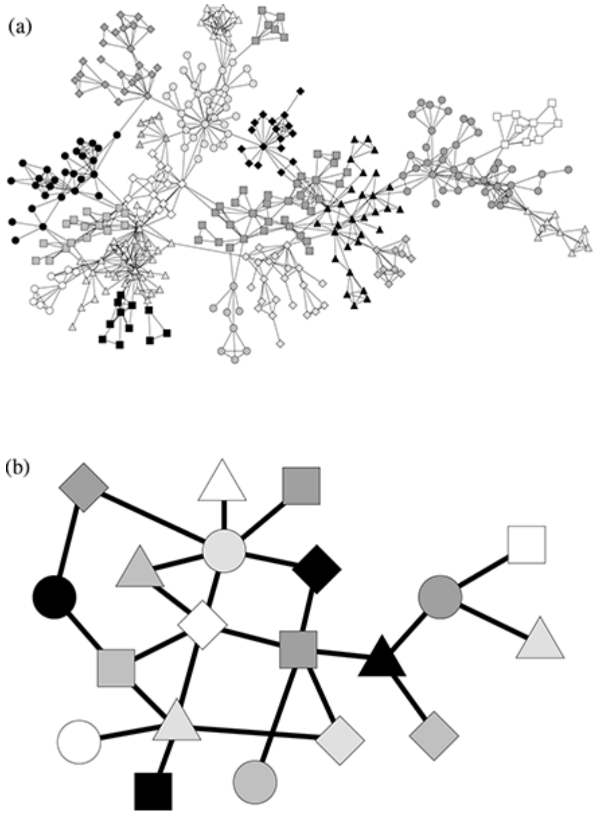
Also referred to as meso-scale, cluster, communities, etc.

Facebook



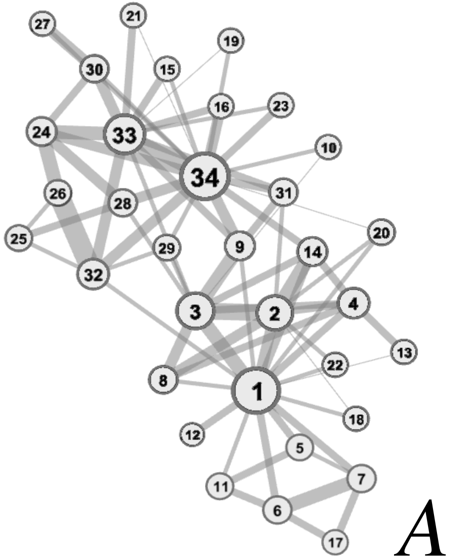
Students



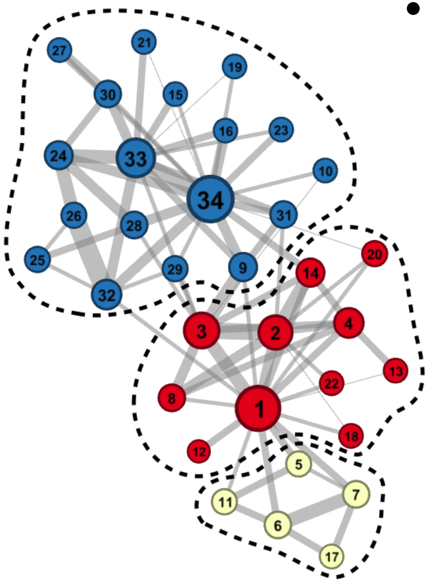


Clustering a.k.a Community Detection

Given a graph, how to cluster the nodes into modules?



Community detection algorithm



Common formulations:

- vector or a function:

  $C \in [1 \ldots k]^n$

  $C_i \in [1..k]$ gives cluster index of node i

- Set of disjoint sets:

  $C = \{C_1, C_2 \ldots C_k\}$

  C_i gives set of nodes belonging to cluster i

  $C_i \cap C_j = \emptyset \forall i \neq j$

  $\cup_1^k C_i = V$: set of all nodes

Outline

- Quick Recap of Centrality Measures
- **Modules**
  - Real graphs are modular
  - **Spectral clustering**
  - Objectives for quality of a module
  - TopLeaders
  - Using Betweenness Centrality
  - Modularity Optimization, FastModularity & Louvain
  - Resolution limits of Modularity
  - Link clustering
  - Evaluating clustering results

Spectral clustering: **Laplacian Matrix**

Uses the relation between connectivity & Laplacian matrix

Recall:

Laplacian Matrix: $L = D - A$

A: adjacency matrix

D: diagonal matrix of degrees

```
[[ 3 -1 -1 -1  0]
 [-1  3 -1  0 -1]
 [-1 -1  4 -1 -1]
 [-1  0 -1  2  0]
 [0  -1 -1  0  2]]
```

L

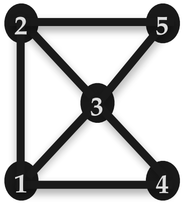
example

```
[[3 0 0 0 0]
 [0 3 0 0 0]
 [0 0 4 0 0]
 [0 0 0 2 0]
 [0 0 0 0 2]]
```

D

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

A



L is symmetric & positive-semidefinite

Spectral clustering: **Laplacian Spectrum**

Uses the relation between connectivity & Laplacian matrix

- $Lu = \lambda u$: Eigenvalues of Laplacian Matrix
- We have n eigenvalues which we call **Laplacian Spectrum**:

$$0 = \lambda_0 \leq \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$$

- λ_0 is always zero since we have $L(1,1...1) = 0$: why?
- $0 = \lambda_0 = \lambda_1 = \lambda_2 = \ldots = \lambda_k \Rightarrow$ k is number of connected components
- Largest is bounded by twice the maximum degree in G
- $E = \frac{1}{2}\sum_i d_i = \frac{1}{2}Tr(L) = \frac{1}{2}\sum_i \lambda_i$
- Spectral gap: smallest nonzero eigenvalue
- Fiedler vector: eigenvector corresponding to the spectral gap
- Spectral ordering: Fiedler vector sorted
- Laplacian Spectrum relates to graph connectivity & clustering

Spectral clustering: **Laplacian Matrix & Smoothness**

For any function on a graph we have

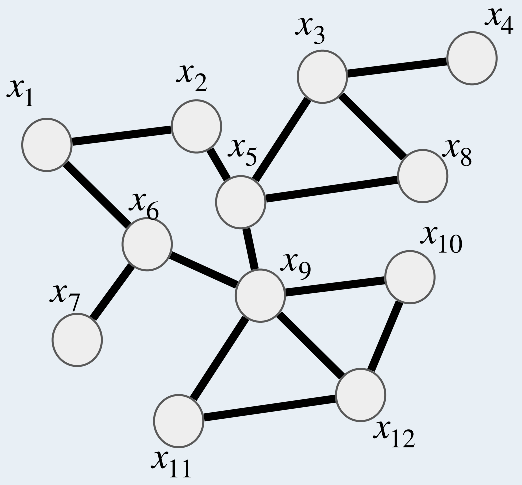
$$x \in \mathbb{R}^n \Rightarrow x^\top L x = \frac{1}{2} \sum_{ij} A_{ij}(x_i - x_j)^2$$

Measures how much the value of f is smooth over edges, i.e. the difference of values for connecting nodes

How to find modules?

Find x that give smoothest results, i.e, minimizes this

Consider function $x : i \mapsto \mathbb{R}$ that maps vertices to a value



$$x = [x_1, x_2, \ldots, x_n]$$

$$x^\top L x = x^\top D x - x^\top A x = \sum_i d_i x_i^2 - \sum_{ij} x_i x_j A_{ij} = \frac{1}{2}[\sum_i d_i x_i^2 - 2\sum_{ij} x_i x_j A_{ij} + \sum_i d_i x_i^2] = \frac{1}{2}\sum_{ij} A_{ij}(x_i - x_j)^2$$

See this for more details.

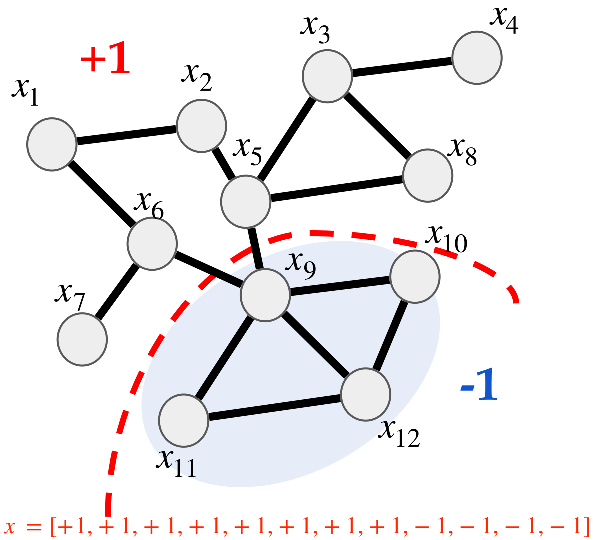
Spectral clustering: **Graph Cut**

Minimize: $$x^\top L x = \frac{1}{2}\sum_{ij} A_{ij}(x_i - x_j)^2$$

- Cut edges $= \frac{1}{4} x^\top L x$, *why?*
- How to enforce balanced clusters?

  Minimize given $x_i \in \{+1, -1\}$, $\sum_i x_i = 0$

  That is having the same number of nodes in each cluster



$x = [+1, +1, +1, +1, +1, +1, +1, +1, -1, -1, -1, -1]$

See this for more details.

x_i **+1** if $i \in 1$ else **-1**

Spectral clustering: **Graph Ratio Cut**

Minimize: $x^\top L x = \frac{1}{2}\sum_{ij} A_{ij}(x_i - x_j)^2$

Given $x_i \in \{+1, -1\}$, $\sum_i x_i = 0$

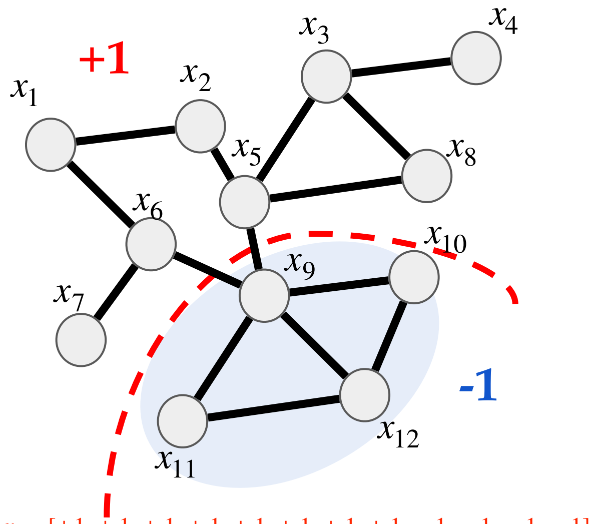
Relax $\Rightarrow x_i \in \mathbb{R}$, $\sum_i x_i^2 = n$, then

Courant Fischer Minimax Theorem

$$Min \frac{1}{4} x^\top L x = \frac{1}{4} n v_1^\top L v_1 = \frac{1}{4} n \lambda_1$$

- Second smallest **eigenvalue**
  $\Rightarrow$ **sparsest ratio cut**

- Signs of corresponding **eigenvector**



$x = [+1, +1, +1, +1, +1, +1, +1, +1, -1, -1, -1, -1]$

See this for more details.

Spectral clustering: **Normalized Cut**

Spectral clustering with unnormalized Laplacian optimizes the RatioCut = $\sum_{i}^{k} \frac{cut(C_i, \bar{C}_i)}{|C_i|}$, $cut(C_i, \bar{C}_i) = \sum_{j \in C_i\, k \notin C_i} A_{jk}$

We can use normalized Laplacian matrix which optimizes **normalized cut** = $\sum_{i}^{k} \frac{cut(C_i, \bar{C}_i)}{vol(C_i)}$, $vol(C_i) = \sum_{j \in C_i} d_j$ *number of* **edges** *in the clusters*

Random walk normalization: $L_{rw} = D^{-1}L = I - D^{-1}A$

- used for spectral clustering by **Shi and Malik** (2000)

Symmetric normalization: $L_{sym} = D^{-1/2}LD^{-1/2} = I - D^{-1/2}AD^{-1/2}$

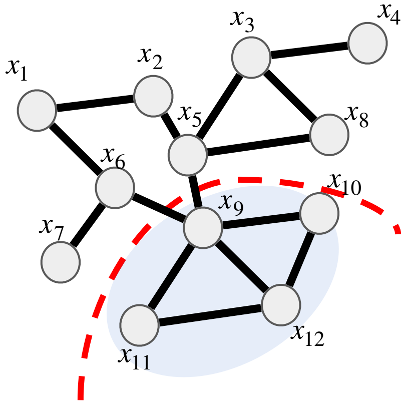
- used for spectral clustering by **Ng, Jordan, and Weiss** (2002)

L_{sym} & L_{rw} are positive semi-definite and have n non-negative real-valued eigenvalues

- Multiplicity of zero eigenvalues in both still gives the number of connected components
- Their eigenvalues are the same, and eigenvectors related

Cut? 2

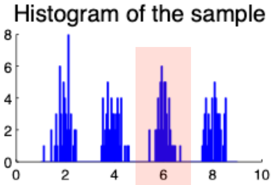
RatioCut? $\frac{2}{4} + \frac{2}{8}$ NormCut? $\frac{2}{12} + \frac{2}{18}$



K Clusters?

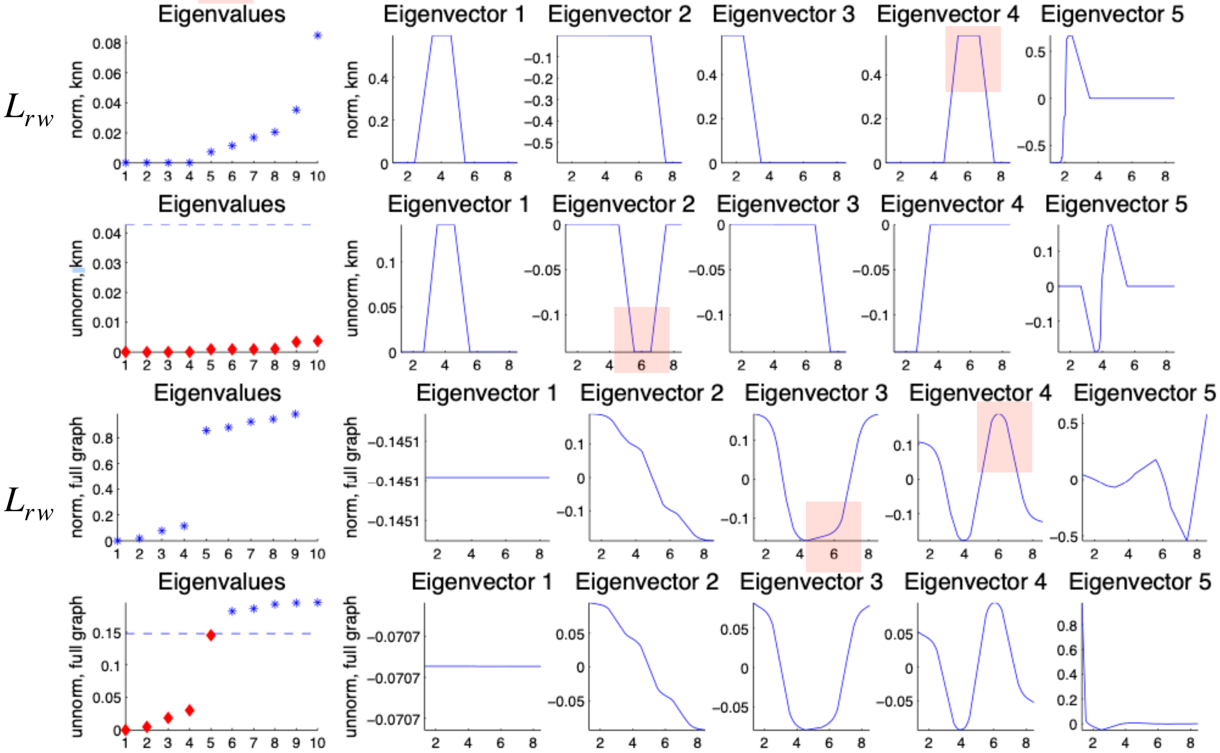
Use k-means on first (nontrivial) k eigenvectors (each node is represented with k features)

Further reading? See this



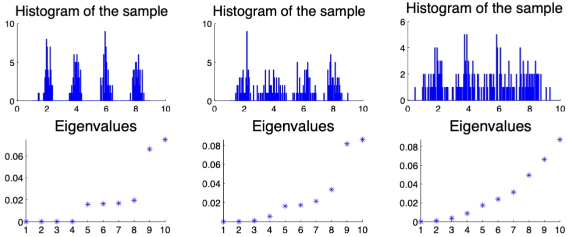
[example] Eigenvectors as indicator vectors of clusters

a random sample of 200 points drawn according to a mixture of four Gaussians: $x_1, \ldots, x_{200} \in R \Rightarrow$ similarity graph with knn or complete graph



Knn: the first four eigenvalues are 0, and the corresponding eigenvectors are cluster indicator vectors since clusters form disconnected parts in the k-nearest neighbor graph

fully connected graph: wighted by similarity, the first eigenvector is the constant vector. The following eigenvectors carry the information about the clusters.



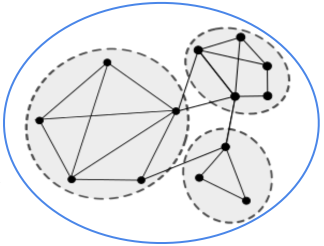
Further reading? See this

Outline

- Quick Recap of Centrality Measures
- **Modules**
  - Real graphs are modular
  - Spectral clustering
  - **Objectives for quality of a module**
  - TopLeaders
  - Using Betweenness Centrality
  - Modularity Optimization, FastModularity & Louvain
  - Resolution limits of Modularity
  - Link clustering
  - Evaluating clustering results

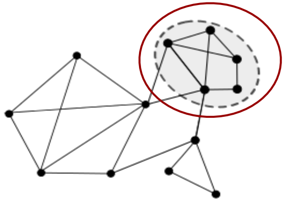
Objectives for quality of a community

We can define the community detection either **globally** or **locally** and have global or local algorithms. Local algorithms are the choice when really with graphs that do not fit in memory.

$Q($  $)$

Globally-defined quality function to partition the whole network

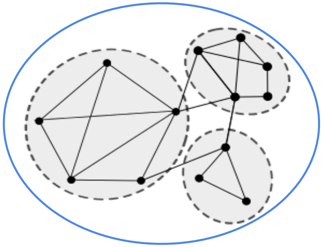
Gives sets of sets, set of all clusters, usually disjoint and covering the full data

$f($  $)$

Locally defined quality function for one subset of nodes in a network

Gives one set of nodes belonging to the same cluster

Objectives for quality of a community

$Q($  $)$

C: sets of sets, set of all clusters, usually disjoint and covering the full data

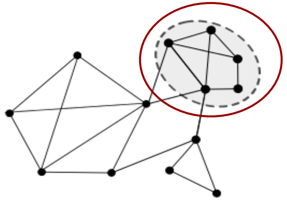
Globally-defined quality function to partition the whole network

- **Q-modularity** (Newman 2003)

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{d_i d_j}{2m}\right) \delta(C_i, C_j)$$

m= total edges in the graph
d_i: degree of nodes i
C_i: cluster index that node i belongs to
$\delta(x, y) = 1 \iff i = j$

Most common representatives

$f($  $)$

S: a set of nodes in one cluster

Locally defined quality function for one subset of nodes in a network

- **Conductance** (Sinclair & Jerrum 1989)

$$f(S) = \frac{c_S}{2m_S + c_S}$$

- Normalized Cut (Shi & Malik 2000)

$$f(S) = \frac{c_S}{2m_S + c_S} + \frac{c_S}{2(m - m_S) - c_S}$$

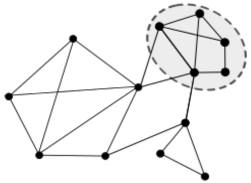
c_S = cut size: number of edges going out of module
m_S= module size: number of edges inside module
$2m_S + c_S$ = vol (S) = sum of degrees for nodes in S

In the example above

= 3/(2*7+3)

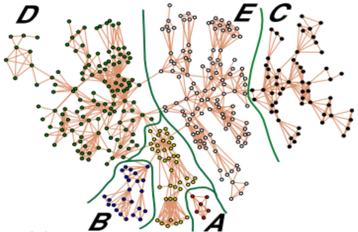
= 3/(2*7+3)
+ 3/(2*(12)+3)

Locally defined objectives

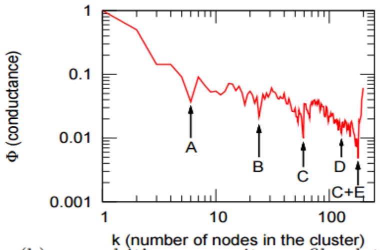


Defining and evaluating network communities based on ground-truth (Yang, J., Leskovec, J., Knowledge and Information Systems, 2015)

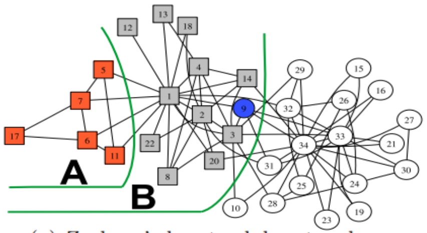
- Community detection from a seed node
  - Measure proximity of nodes from seed using random walk
  - Expand from the closest node $(\frac{r_i}{d_i})$, and compute the objective for every first k nodes
  - Local optima of objective (e.g. conductance) correspond to detected communities



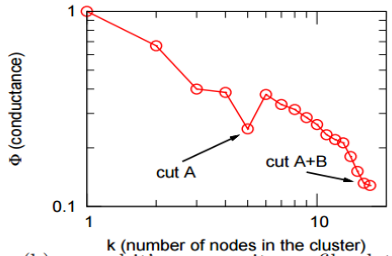
(g) Network science network . . .



(h) . . . and it's community profile plot



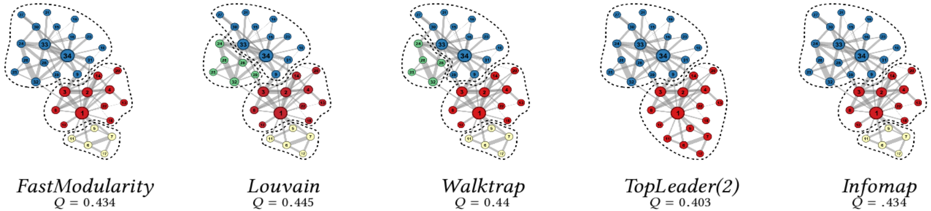
(a) Zachary's karate club network . . .



(b) . . . and it's community profile plot

Defining the Global Modular Structure of Networks

- **Number of links between them is more than chance**
  - Modularity Q (Newman & Grivan, Phys Rev E, 2004)
    - FastModularity (Clauset, Phys Rev E 2005); Louvain (Blondel et al., J Stat Mech Theory Exp, 2008)
- **Within them a random walk is more likely to trap**
  - Walktrap (Pons & Latapy, ISCIS 2005)
- **Coding gives efficient compression of any random walk**
  - Infomap (Rosvall & Bergstrom, PNAS 2008; PloS One 2010)
- **Follow their closest leader**
  - TopLeader



FastModularity $Q = 0.434$

Louvain $Q = 0.445$

Walktrap $Q = 0.44$

TopLeader(2) $Q = 0.403$

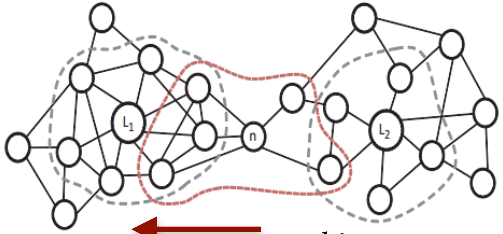
Infomap $Q = .434$

Outline

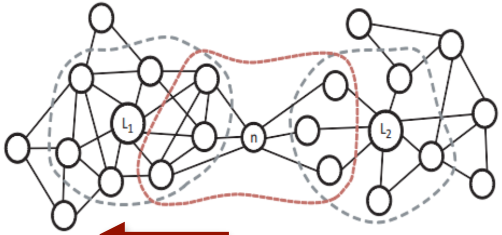
- Quick Recap of Centrality Measures
- **Modules**
  - Real graphs are modular
  - Spectral clustering
  - Objectives for quality of a module
  - **TopLeaders**
  - Using Betweenness Centrality
  - Modularity Optimization, FastModularity & Louvain
  - Resolution limits of Modularity
  - Link clustering
  - Evaluating clustering results

TopLeaders: K-medoid for graphs

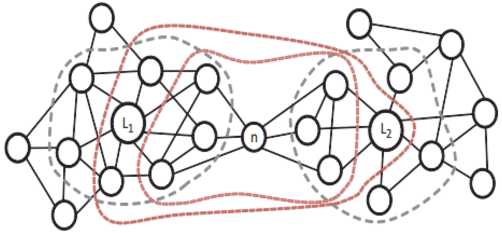
- Iteratively assigns nodes to leaders, selects leaders
  - Leader: central member in community
  - Community: set of followers surrounding a leader
  - Assigning followers to closest leader based on neighbourhoods
- Initialization requires k (central nodes with few neighbours in common)



More neighbours



More cohesive neighbours



More extended neighbours

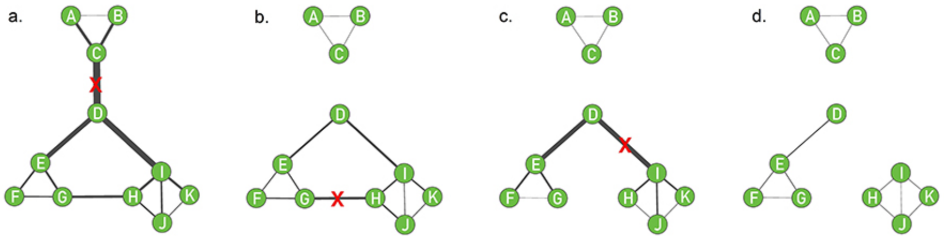
- Also identifies outliers and hubs in the network
- Closeness measure based on diffusion of innovation

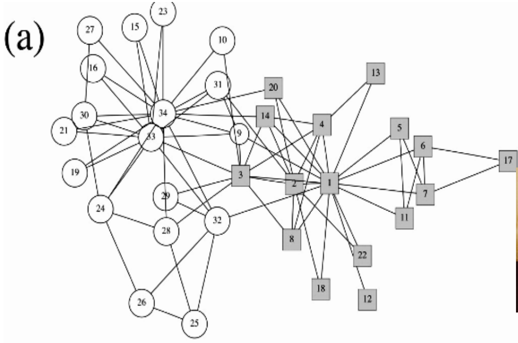
First community detection algorithm

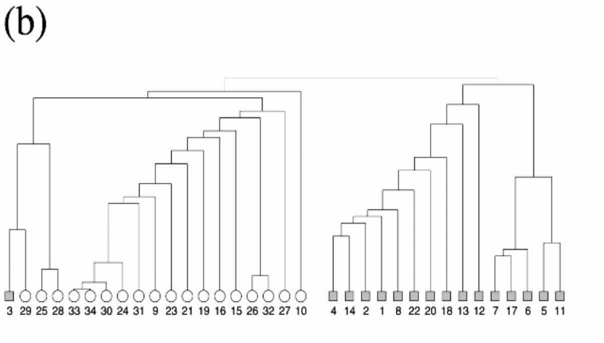
A divisive hierarchical clustering

(Girvan and Newman, PNAS 2002)

1. Calculate the betweenness for all edges in the network
2. Remove the edge with the highest betweenness
3. Recalculate betweennesses for all edges affected by the removal
4. Repeat from step 2 until no edges remain
5. When to stop? Where to cut the dendrogram?

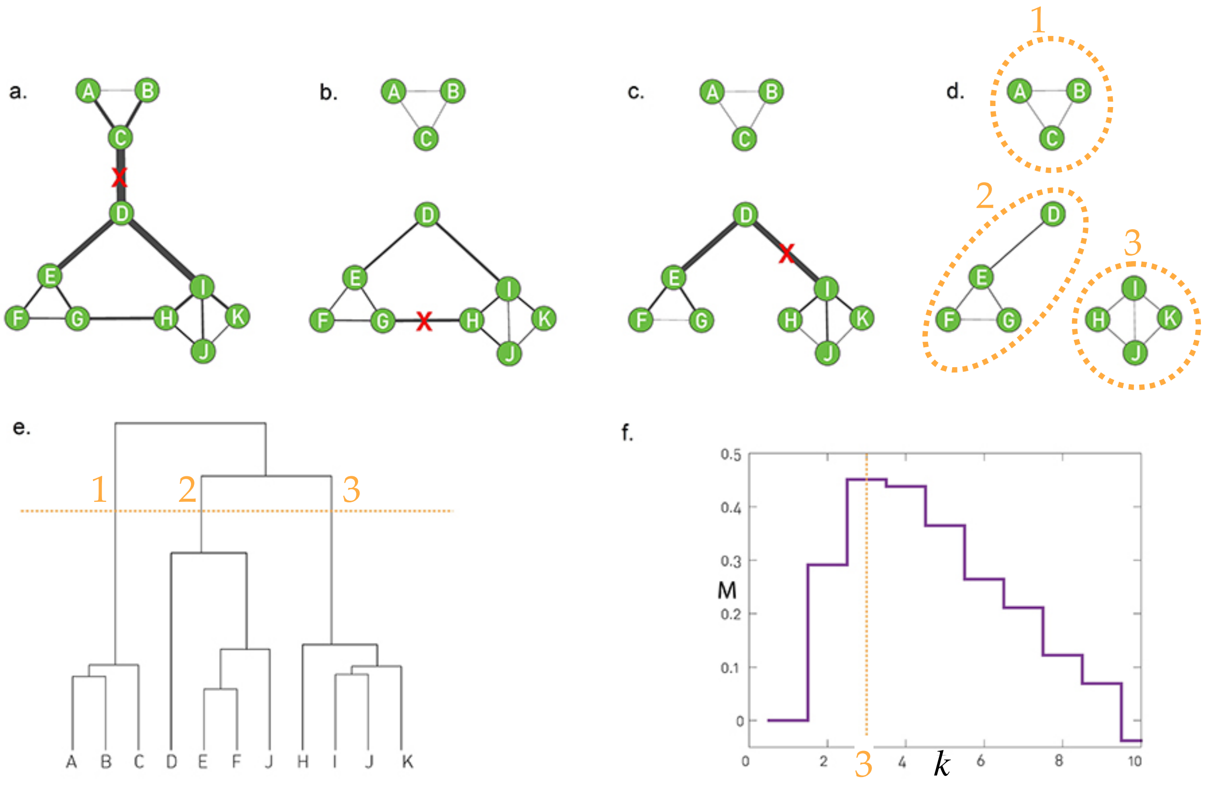






the infamous Karate club dataset
https://networkkarate.tumblr.com/

A divisive hierarchical clustering



Recursively remove **bridges**, edges with high edge-betweenness

In the resulted dendrogram, evaluate M for flat modules obtained at different levels

How to define M?

Q-modularity: goodness of a global partition

Originally proposed to know where to cut the dendrogram, but we optimize this directly in practice

Measure the difference between the fraction of edges that are within the clusters and the expected such fraction if the edges were randomly distributed when degrees are fixed, i.e. using the configuration model as the null model

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \delta(C_i, C_j)$$

Only nonzero if *i* and *j* are in the same cluster

$$Q = \sum_{k} \sum_{ij \in C_k} \frac{A_{ij}}{2m} - \frac{d_i}{2m} \frac{d_j}{2m}$$

Sum over all pairs of nodes in the same cluster

probability of an edge in configuration model

m= total edges in the graph
d_i: degree of nodes I
C_i: cluster index that node i belongs to
$\delta(x, y) = 1 \iff i = j$ {Kronecker delta}

$$Q \leq 1$$

Rule of thumb: $Q > 0.3$ indicates strong communities

Q-modularity: goodness of a global partition

Measure the difference between the fraction of edges that are within the clusters and the expected such fraction if the edges were randomly distributed when degrees are fixed, i.e. using the configuration model as the null model

$$Q = \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{d_i d_j}{2m}\right)\delta(C_i, C_j)$$

m= total edges in the graph
d_i: degree of nodes i
C_i: cluster index that node i belongs to
$\delta(x, y) = 1 \iff i = j$ {Kronecker delta}

$$Q = \sum_k \sum_{ij\in C_k} \frac{A_{ij}}{2m} - \frac{d_i}{2m}\frac{d_j}{2m}$$

fraction of edges between cluster *k* and *l*

$$Q = \sum_k E_{kk} - E_k^2 = Tr[E] - \|E^2\| \quad \text{where} \quad E_{kl} = \sum_{i\in c_k,\ j\in c_l} \frac{A_{ij}}{2m}$$

Fraction of edges within clusters

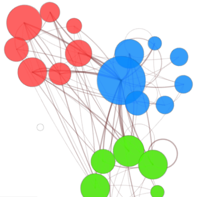
$E_k = \sum_l E_{kl}$

Expected fraction of edges within clusters by chance, i.e. in configuration model

$= \sum_{kl} E_{ij}^2$

Outline

- Quick Recap of Centrality Measures
- **Modules**
    - Real graphs are modular
    - Spectral clustering
    - Objectives for quality of a module
    - TopLeaders
    - Using Betweenness Centrality
    - **Modularity Optimization, FastModularity & Louvain**
    - Resolution limits of Modularity
    - Link clustering
    - Evaluating clustering results



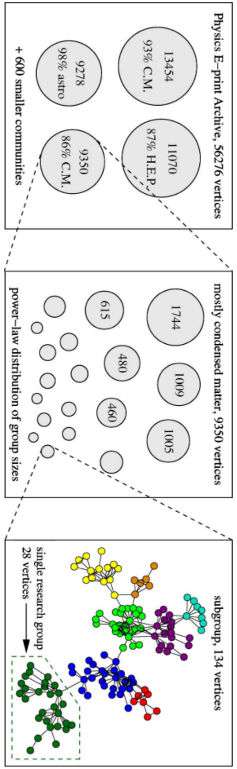
Modularity optimization: a an agglomerative hierarchical clustering

(Newman, Phys. Rev. E 2004)

1. Start from every node a cluster
2. Initialize E as the adjacency matrix
3. Merge two cluster that give the highest gain in Q:

$$\Delta Q = 2(E_{ij} - E_i E_j)$$

1. Update the E by merging together the rows and columns corresponding to the joined communities
2. Go to step 3 until no increase in Q


Physics E-print Archive, 56276 vertices
13454
93% C.M.
11070
87% H.E.P.
9278
98% astro
9350
86% C.M.
+ 600 smaller communities
mostly condensed matter, 9350 vertices
1744
615
1009
480
460
1005
power−law distribution of group sizes
subgroup, 134 vertices
single research group
28 vertices

Modularity optimization

- Divisive hierarchical clustering (Girvan and Newman, PNAS 2002)
  - Removes the edge with highest betweenness
  - All pairs shortest paths: expensive to compute
  - can be approximated but still not scalable

- Agglomerative hierarchical clustering (Newman, Phys. Rev. E 2006)
  - Start from every node a cluster, and merge
  - $\mathcal{O}(n(m+n))$: n, m: number of nodes and edges
  - With heap based data structure ⇒ $\mathcal{O}(m \log n)$ (Clauset et al., 2004)

⇒ **FastModularity**

Louvain, another agglomerative method

Agglomerative method tends to produce super-communities => go Louvain

$$Q = \sum_k \sum_{ij \in C_k} \frac{A_{ij}}{2m} - \frac{d_i}{2m}\frac{d_j}{2m} \quad => \quad Q = \sum_k \sum_{ij \in C_k} W_{ij} - W_{i.}W_{j.} \quad \text{where} \quad W = \frac{1}{2m}A$$

W_{ij} : normalized weight of the edge from node *i* to node j

Each node its own cluster

Move nodes to neighbouring cluster (through the links) with maximum gain

Aggregate clusters as nodes

Repeat

Gain of adding node *i* to community k is $\Delta Q = 2 \sum_{j \in k} W_{ij} - W_{i.}W_{j.}$

$\mathcal{O}(n \log n)$: very fast and can be used for large graphs

(Blondel et al. Journal of Statistical Mechanics, 2008)

Outline

- Quick Recap of Centrality Measures
- **Modules**
  - Real graphs are modular
  - Spectral clustering
  - Objectives for quality of a module
  - TopLeaders
  - Using Betweenness Centrality
  - Modularity Optimization, FastModularity & Louvain
  - **Resolution limits of Modularity**
  - Link clustering
  - Evaluating clustering results

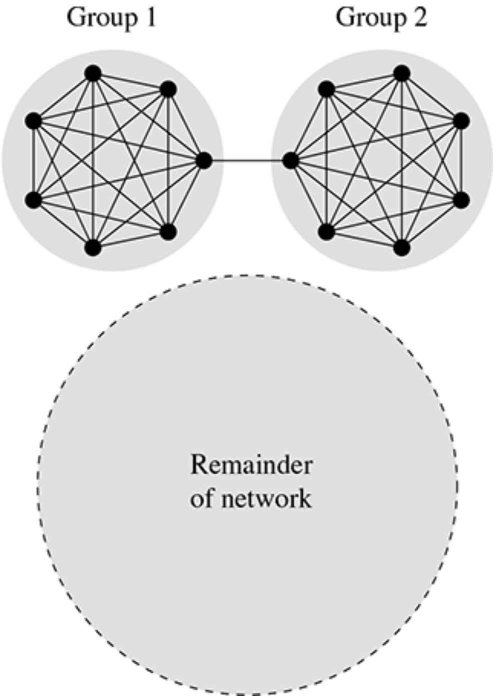
Modularity optimization: limitations

very different divisions of the network can have the same Q modularity

Resolution limit, the inability to see communities in a network if they are too small, relative to the size of the network as a whole

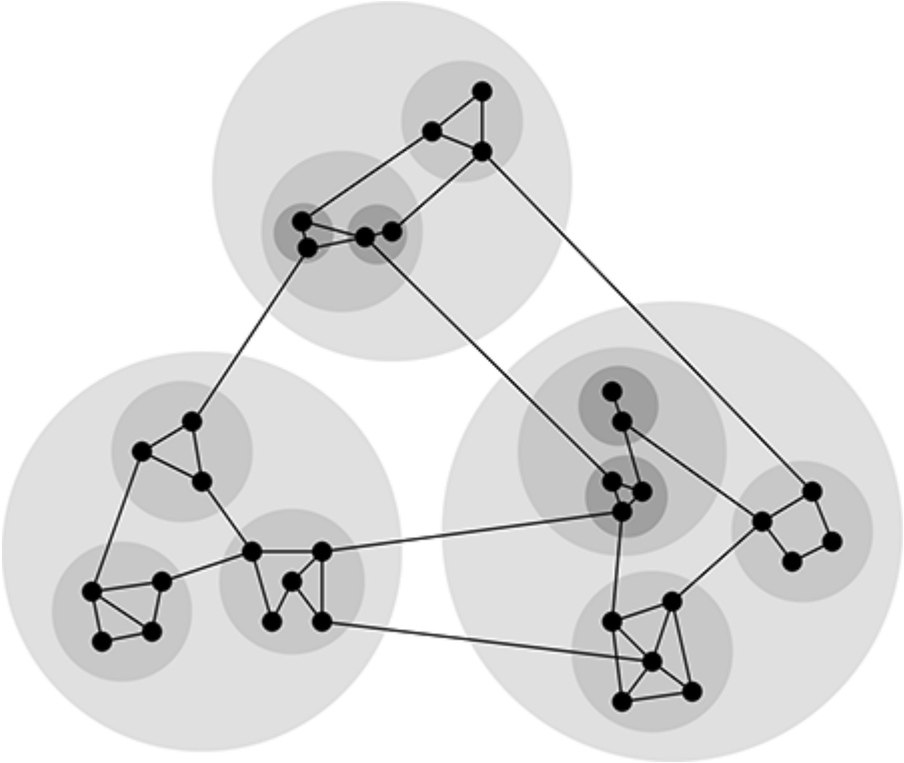
$$\Delta Q = \frac{1}{2m} - E_i E_j > 0 \iff E_i E_j > 2m$$

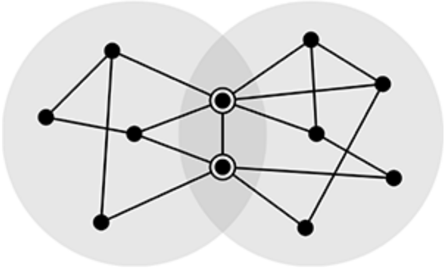
modularity maximization will fail to distinguish these groups as separate communities if the product of the sums of their degrees is less than twice the number of edges in the entire network

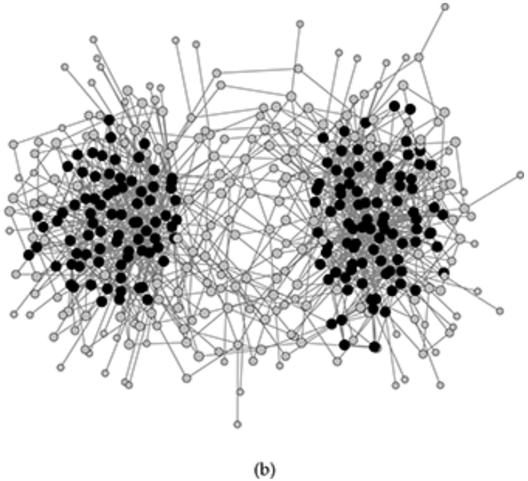


E.g. 5000 edges, can not detect degree sum less than 100

Overlap, hierarchy, periphery





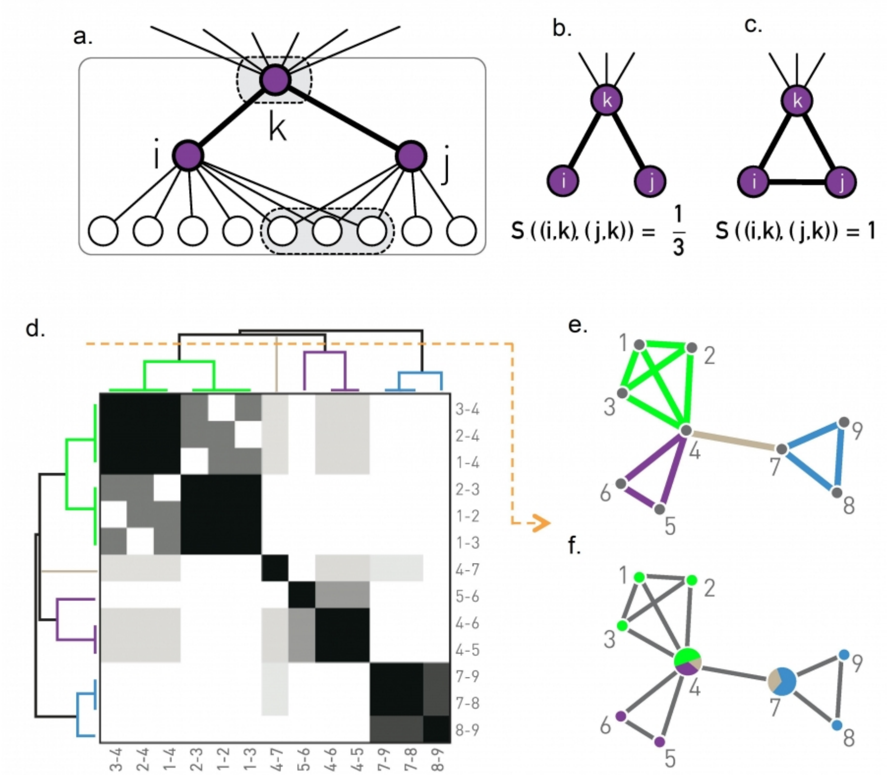


Link Clustering

Find overlapping clusters naturally by clustering edges instead of nodes

The similarity of a link pair is determined by the neighbourhood of the nodes connected by them.

Ahn YY, Bagrow JP, Lehmann S. Link communities reveal multiscale complexity in networks. nature. 2010 Aug;466(7307):761-4.

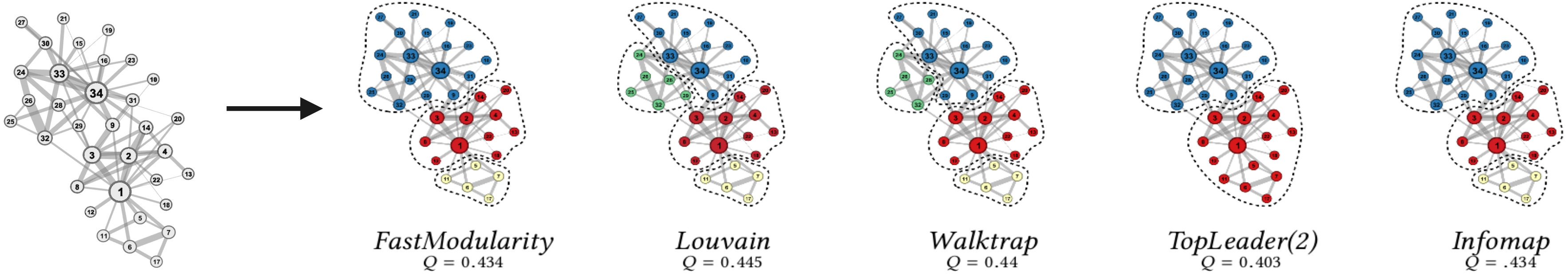


Outline

- Quick Recap of Centrality Measures
- **Modules**
  - Real graphs are modular
  - Spectral clustering
  - Objectives for quality of a module
  - TopLeaders
  - Using Betweenness Centrality
  - Modularity Optimization, FastModularity & Louvain
  - Resolution limits of Modularity
  - Link clustering
  - **Evaluating clustering results**

Evaluating the Modular Structure of Networks

*Given different algorithms **which one** to choose?*



FastModularity $Q = 0.434$

Louvain $Q = 0.445$

Walktrap $Q = 0.44$

TopLeader(2) $Q = 0.403$

Infomap $Q = .434$

We can do external and/or internal/relative evaluation

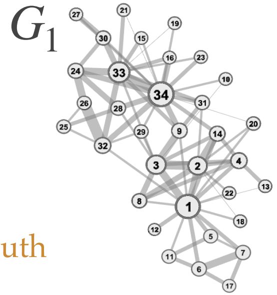
- External Evaluation: Compare performances on benchmark datasets with known true clusters
  - useful for designing algorithms
- Internal Evaluation: use a quality index to pick an algorithm for datasets, e.g. Q-modularity
  - useful when applied to real world graph with unknown clusters
  - see here for comparison of Q with some alternatives

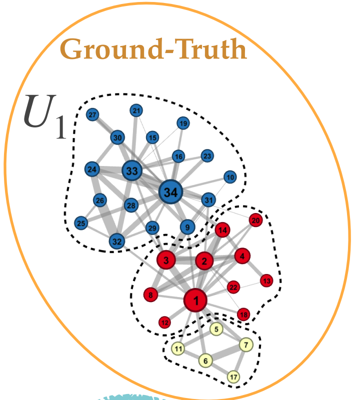
External Evaluation of Community Detection: a common practice

Validate on graph benchmarks with known true partition

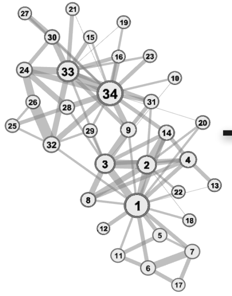
(G_1, U_1)
(G_2, U_2)
(G_3, U_3)
⋮

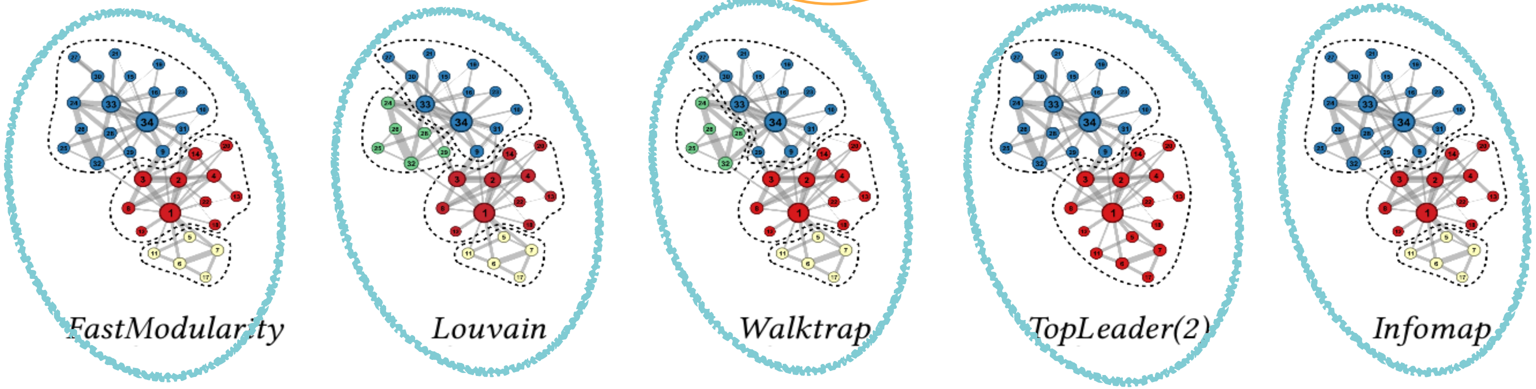
For each benchmark compare results of each algorithm with ground-truth





Assumption:
average performance on benchmarks predicts how well the algorithm would be in practice when applied to data with unknown clusters



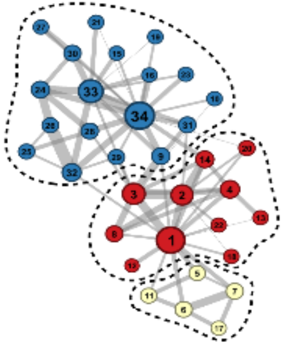


For this comparison we need an agreement measure that gets two clusterings and quantifies how much they agree.

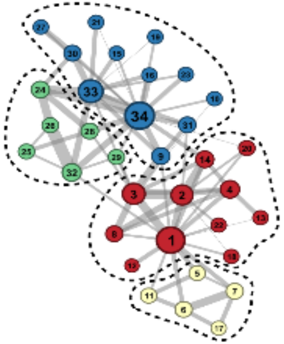
Clusterings Agreement Measures

A measure of agreement for clusterings quantifies how much they agree.

A(



,



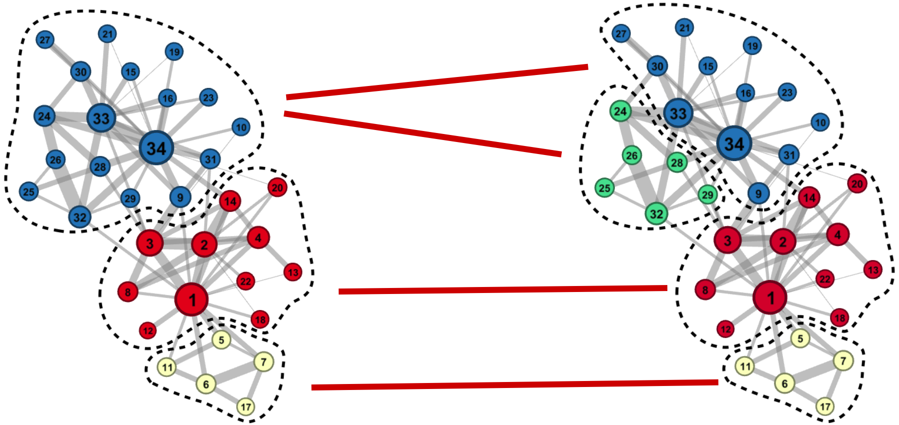
)

There are 3 main families:

- Set matching
- Information theoretic
- Pair counting

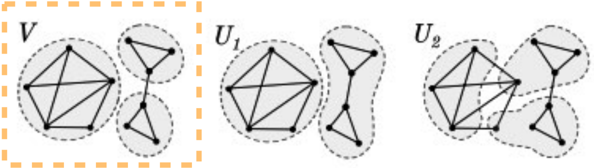
Clusterings Agreement Measures: Set Matching family

Based on a one-2-one matching between clusters in the two partitioning



"problem of matching" since it only compares the best matched clusters

example:



Which one of U_1 and U_2 better agrees with V:

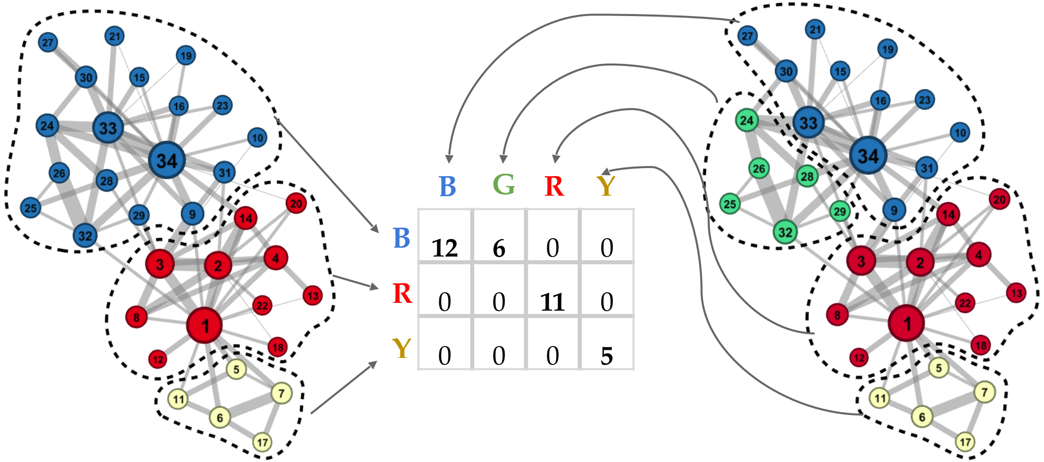
$$A(U_1, V) > A(U_2, V)$$

Read more here

Clusterings Agreement Measures: Information theoretic family

Examples: *Variation of Information (VI), Normalized Mutual Information (NMI)*

Consider all the **pairwise overlaps** between clusters as a joint distribution then define joint entropy and mutual information.



	B	G	R	Y
B	12	6	0	0
R	0	0	11	0
Y	0	0	0	5

n_{ij} : overlap size between cluster i and j

$$H(U,V) = -\sum_k \sum_r \frac{n_{ij}}{n} \log \frac{n_{ij}}{n}$$

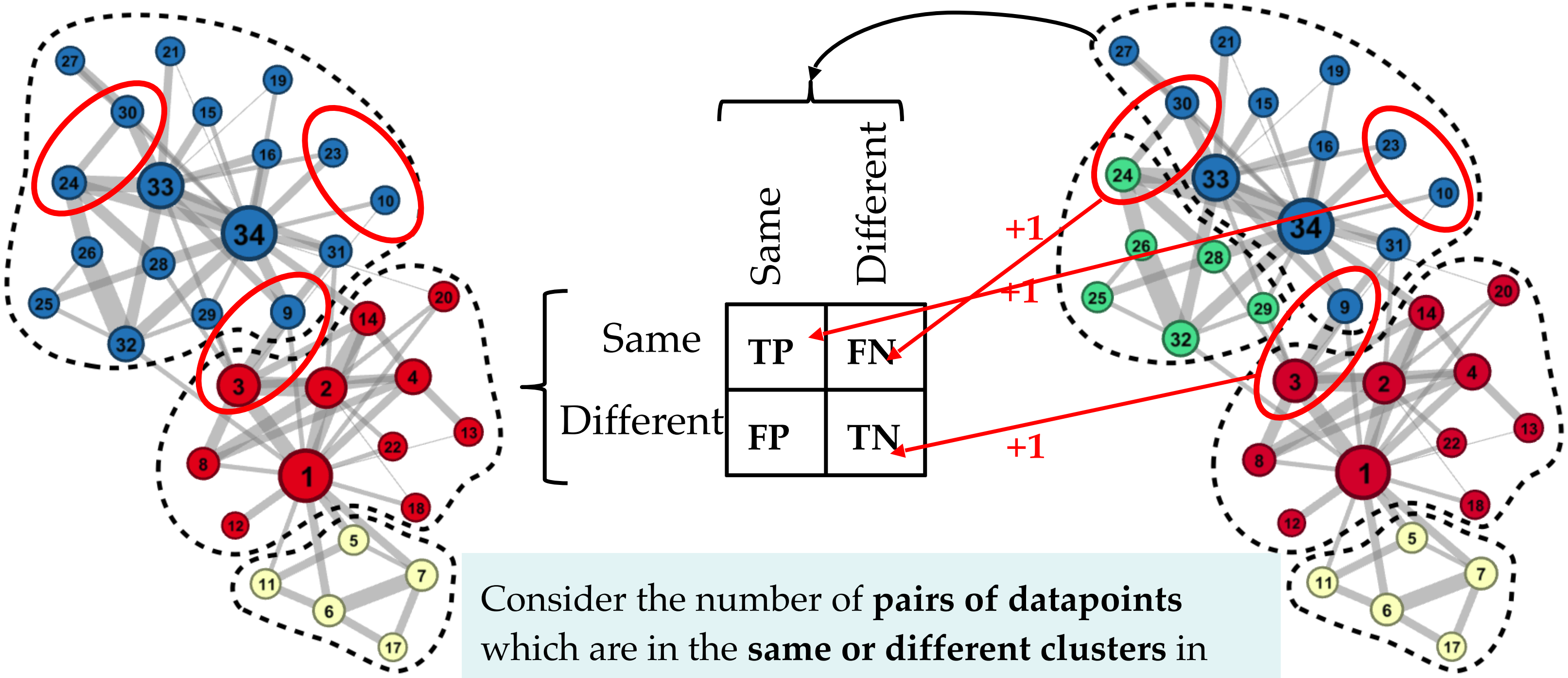
$$H(U) = -\sum_k \frac{n_{i.}}{n} \log \frac{n_{i.}}{n}$$

$$NMI = \frac{I(U,V)}{\frac{1}{2}[H(U)+H(V)]}$$

$$NMI = \frac{H(U,V)-H(U)-H(V)}{\frac{1}{2}[H(U)+H(V)]}$$

Clusterings Agreement Measures: Pair counting family

Examples: *Jaccard, Rand Index, F-measure, Adjusted Rand Index (ARI)*



	Same	Different
Same	TP	FN
Different	FP	TN

Consider the number of **pairs of datapoints** which are in the **same or different clusters** in the two clusterings then compute F-measure

Clusterings Agreement Measures: Pair counting family

Examples: *Jaccard, Rand Index, F-measure, Adjusted Rand Index (ARI)*

Consider the number of **pairs of datapoints** which are in the **same or different clusters** in the two clusterings then compute F-measure

	Same	Different
Same	**TP**	**FN**
Different	**FP**	**TN**

We can derive these also from the all the pairwise overlaps between clusters [which is used by information theoretic measures]

	B	G	R	Y
B	12	6	0	0
R	0	0	11	0
Y	0	0	0	5

For example we can compute TP as:

$$\mathbf{TP} = \binom{12}{2}+\binom{6}{2}+\binom{11}{2}+\binom{5}{2}$$

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \sum_i\binom{n_{i.}}{2}\sum_j\binom{n_{.j}}{2}/n^2}{1/2[\sum_i\binom{n_{i.}}{2}+\sum_j\binom{n_{.j}}{2}] - \sum_i\binom{n_{i.}}{2}\sum_j\binom{n_{.j}}{2}/n^2}$$

Read more here

Generalization: Linking the two families

Both pair counting and information theoretic measures are quantifying dispersion in the confusion/contingency table

	B	G	R	Y	Σ
B	12	6	0	0	18
R	0	0	11	0	11
Y	0	0	0	5	5
Σ	12	6	11	5	34

$$\left.\begin{array}{l} \phi(18) - \phi(12) - \phi(6) \\ \phi(11) - \phi(11) \\ \phi(5) - \phi(5) \end{array}\right\}\Sigma \qquad A = \frac{\phi(18) - \phi(12) - \phi(6)}{\phi(34)}$$

Subsumes pair counting $\Phi(x) = \binom{x}{2} \Rightarrow A = 1 - \dfrac{TP + TN}{TP + TN + FP + FN}$ **Rand Index**

Subsumes information theoretic $\Phi(x) = x\log(x) \Rightarrow A = \dfrac{1}{\log(34)}\left[H(U,V) - I(U,V)\right]$ **Variation of Information**

Read more here

Generalization: Linking the two families

Both pair counting and information theoretic measures are quantifying dispersion in the confusion/contingency table

	B	G	R	Y	Σ
B	12	6	0	0	18
R	0	0	11	0	11
Y	0	0	0	5	5
Σ	12	6	11	5	34

$$\left.\begin{array}{l}\phi(18)-\phi(12)-\phi(6)\\ \phi(11)-\phi(11)\\ \phi(5)-\phi(5)\end{array}\right\}\sum$$

$$\hat{A}=\frac{\phi(18)-\phi(12)-\phi(6)}{\phi(\frac{18}{34}\times\frac{12}{34})+\phi(\frac{18}{34}\times\frac{6}{34})+\phi(\frac{18}{34}\times\frac{11}{34})+\dots}$$

Subsumes pair counting $\Phi(x)=\binom{x}{2}\Rightarrow\hat{A}=ARI$ Adjusted Rand Index

Subsumes information theoretic $\Phi(x)=x\log(x)\Rightarrow\hat{A}=NMI$ Normalized Mutual Information

ARI is more robust to changes in number of clusters NMI tends to increase with number of clusters

Read more here & here

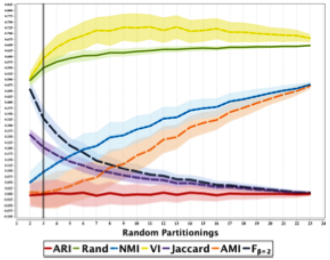
External Evaluation

Which agreement measure to choose?

either ARI and NMI, both are quantifying dispersion in the contingency table

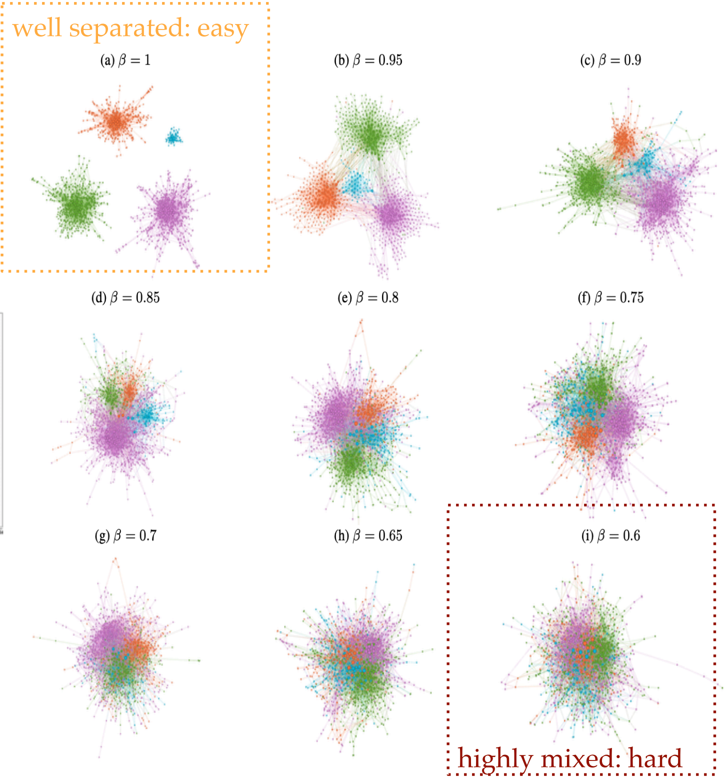
- ARI is more robust to changes in number of clusters and is a better choice when number of cluster varies too much
- NMI tends to increase with number of clusters even when clusters are random

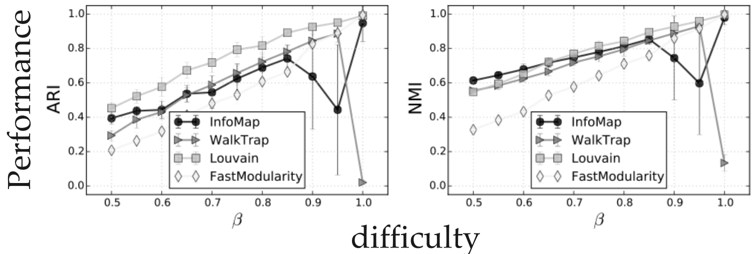
Read more here



Which benchmarks to use?

There are few small **real world benchmarks** with known clustering, as well as large ones with attributes closely related to known clustering that we use as a proxy for ground-truth (e.g. venues papers are published in for citation graph). But we often mostly evaluate algorithms in a controlled setting with **synthetic graph generators that have builtin ground-truth**, e.g. SBM, LFR or FARZ where we can control how well separated the clusters are [difficulty of the task]





Matrix Formulation of Clusters for Overlapping Clusters

- Vector or a function:

  $C \in [1 \ldots k]^n$

  $C_i \in [1..k]$ gives cluster index of node i

- Set of disjoint sets:

  $C = \{C_1, C_2 \ldots C_k\}$

  C_i gives set of nodes belonging to cluster i

  $C_i \cap C_j = \varnothing \forall i \neq j$

  $\cup_1^k C_i = V$: set of all nodes

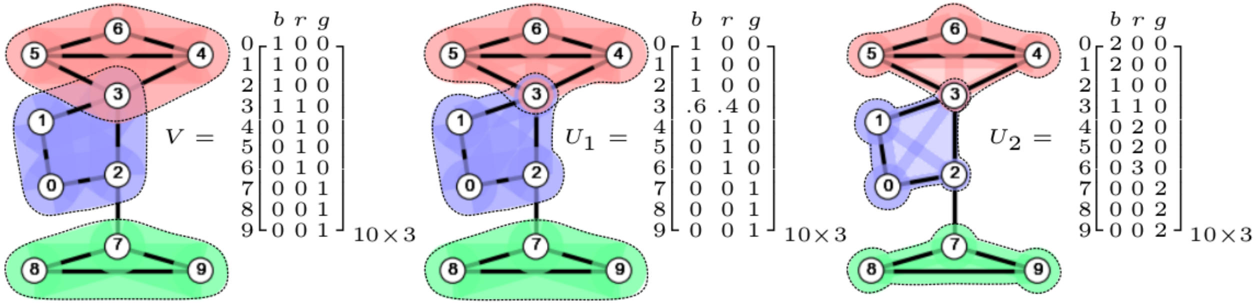
- **Membership Matrix:**

  $C \in \mathbb{R}^{n \times k}$

  C_{ik} gives the degree to which node i belonging to cluster k

$(UU^\top)_{ij}$: how many clusters node i and j appeared together

$(U^\top U)_{ij}$: how many nodes clusters i and j have in common



$$V = \begin{array}{c|ccc} & b & r & g \\ \hline 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 3 & 1 & 1 & 0 \\ 4 & 0 & 1 & 0 \\ 5 & 0 & 1 & 0 \\ 6 & 0 & 1 & 0 \\ 7 & 0 & 0 & 1 \\ 8 & 0 & 0 & 1 \\ 9 & 0 & 0 & 1 \end{array}_{10\times 3}$$

$$U_1 = \begin{array}{c|ccc} & b & r & g \\ \hline 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 3 & .6 & .4 & 0 \\ 4 & 0 & 1 & 0 \\ 5 & 0 & 1 & 0 \\ 6 & 0 & 1 & 0 \\ 7 & 0 & 0 & 1 \\ 8 & 0 & 0 & 1 \\ 9 & 0 & 0 & 1 \end{array}_{10\times 3}$$

$$U_2 = \begin{array}{c|ccc} & b & r & g \\ \hline 0 & 2 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 3 & 1 & 1 & 0 \\ 4 & 0 & 2 & 0 \\ 5 & 0 & 2 & 0 \\ 6 & 0 & 3 & 0 \\ 7 & 0 & 0 & 2 \\ 8 & 0 & 0 & 2 \\ 9 & 0 & 0 & 2 \end{array}_{10\times 3}$$

Read more here & here