MCCXXII

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Network Science

A.Y. 23/24

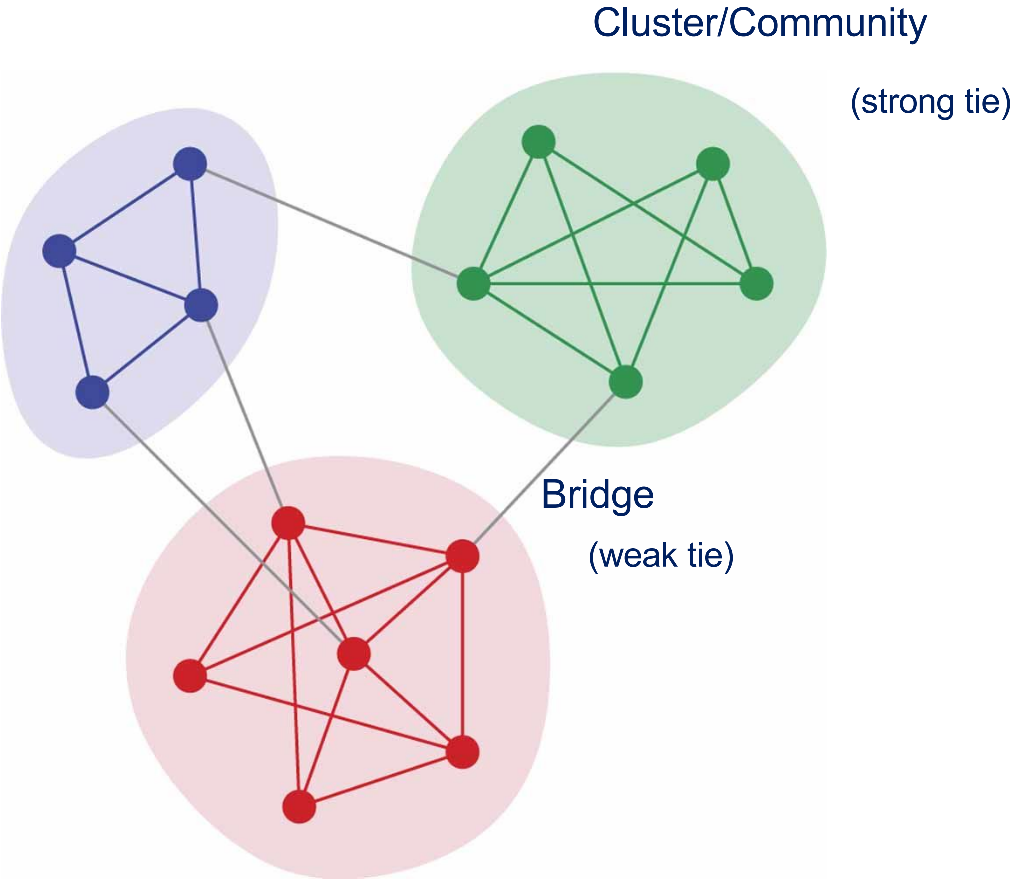
ICT for Internet & multimedia, Data science, Physics of data

Community detection

Identify communities in a network

Conceptual picture of a network

explaining the role of community detection



- ☐ We often think of networks looking like this
- ☐ But, where does this idea come from?

Granovetter’s explanation

Granovetter, The strength of weak ties [1973]
https://www.jstor.org/stable/pdf/2776392.pdf

Q: How do people discovered their new jobs?

A: Through personal contacts, and mainly through acquaintances rather than through close friends

Local cluster/community
Strong ties

Remark: Good jobs are a scarce resource

Conclusion:

- Structurally embedded edges are also socially strong, but are heavily redundant in terms of information access
- Long-range edges spanning different parts of the network are socially weak, but allow you to gather information from different parts of the network (and get a job)

Bridges
Weak ties

Community detection

the general approach

- Granovetter's theory suggests that networks are composed of tightly connected sets of nodes (i.e., communities), loosely connected between them
- We want to be able to automatically find such densely connected group of nodes
- We look for unsupervised methods, as most of the times no ground truth is available
- We look for a measure of the goodness of a community assignment, to be able to compare the performance of different algorithms
- Applications in:
    - social networks
    - functional brain networks in neuroscience
    - scientific interactions

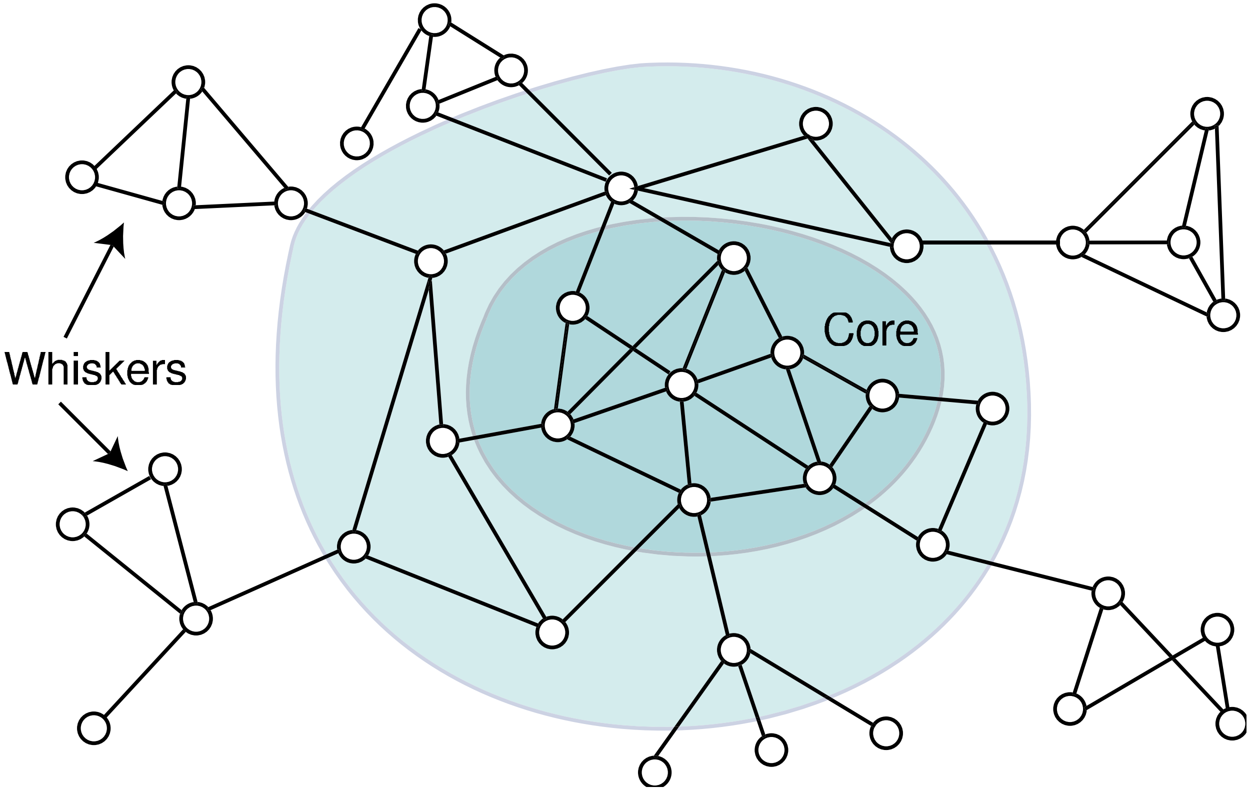
Figure 2 | A network of collaborations among scientists at a research

The core periphery model

Lescovec, Lang, Dasgupta, Mahoney, Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters (2008)
https://arxiv.org/abs/0810.1355

Can we find a justification for this?

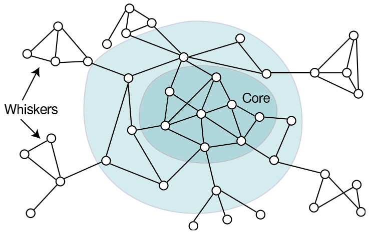
Small, peripheral clusters



Caricature of network structure

Overlapping communities

to explain the core periphery model

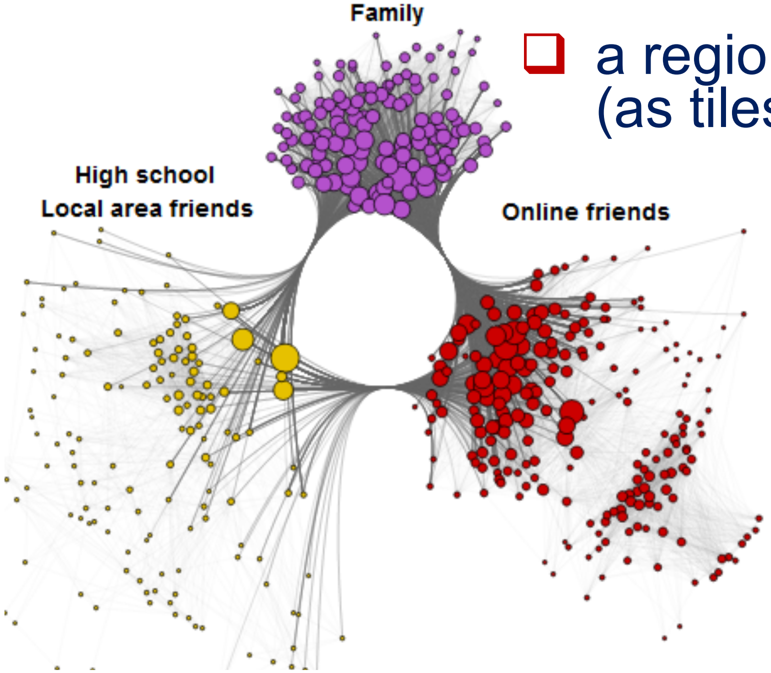


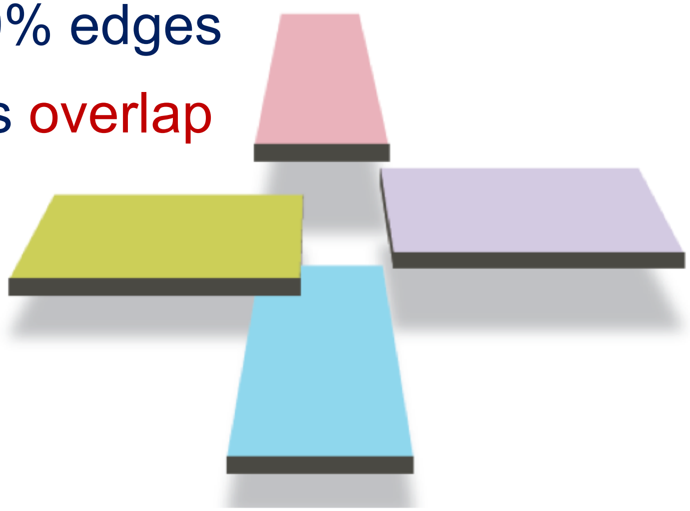
Wiskers

- are typically of size 100
- are responsible of good communities

Core

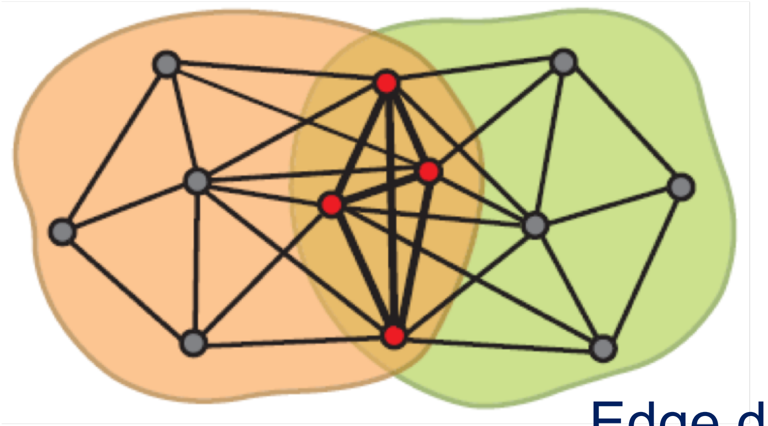
- denser and denser region
- contains 60% nodes and 80% edges
- a region where communities overlap (as tiles)



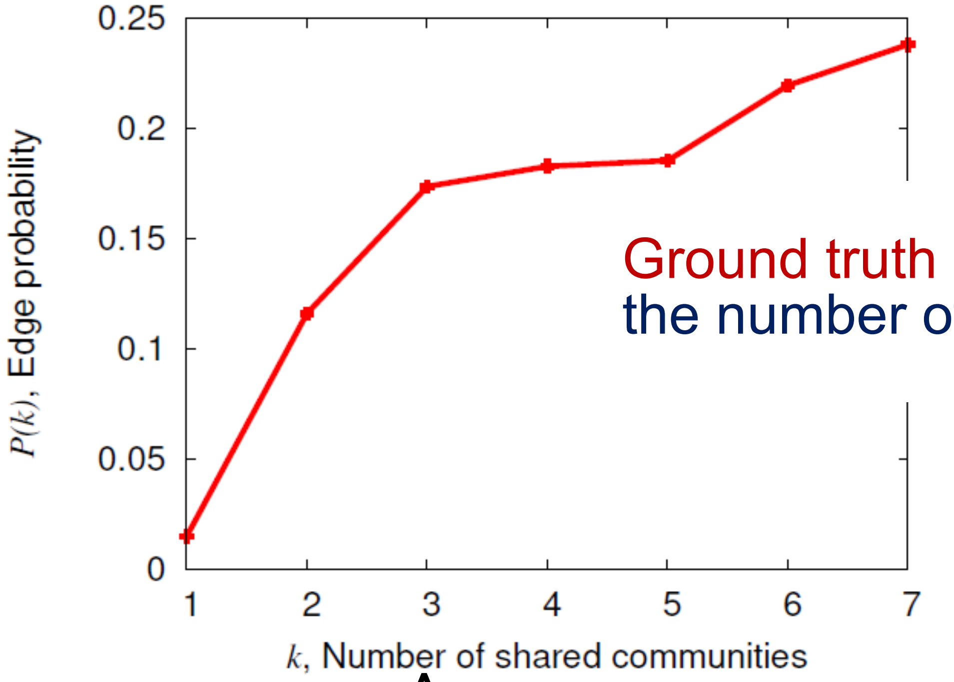


Measuring overlapping

in social networks



Edge density is bigger in the overlap



Amazon

Ground truth - Edge probability increases with the number of shared communities

Feld, The focused organization of social ties, [1981]
The more different communities that two individuals share, the more likely is that they will be tied

Modularity

Measuring the goodness of a community assignment

Modularity

Newman, Modularity and community structure in networks (2006)
https://www.pnas.org/content/pnas/103/23/8577.full.pdf

Want to:

- ☐ measure of how well a network is partitioned into communities (i.e., sets of tightly connected nodes)

Idea:

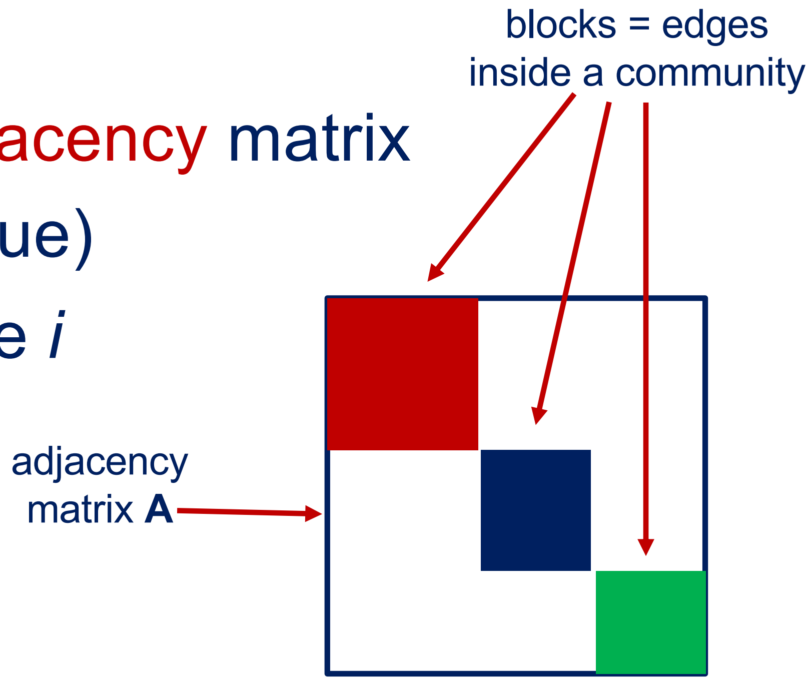
- ☐ “If the number of edges between two groups is only what one would expect on the basis of random chance, then few thoughtful observers would claim this constitutes evidence of meaningful community structure”
- ☐ Modularity is “the number of edges falling within groups minus the expected number in an equivalent network with edges placed at random”

Number of edges falling within groups

an adjacency matrix overview

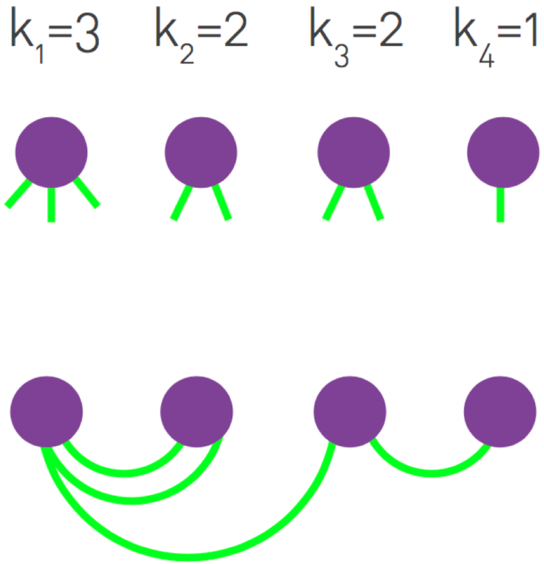
$$Q_1 = \sum_{ij} a_{ij} \cdot \eta(c_i = c_j)$$

- ☐ a_{ij} entries of the (binary) adjacency matrix
- ☐ η indicating function (=1 if true)
- ☐ c_i community (value) of node i



Network with edges places at random

Molloy-Reed model (1995)



1. unwire nodes by breaking edges
   but keep stubs (2L in number)
   so that nodes keep their degree

2. rewire stubs at random

The resulting graph may contain cycles and multiple links (but are a few)

Rewiring probability is $p_{ij} = k_i \ k_j / 2L$

number of trials
from node i

probability of
linking to node j

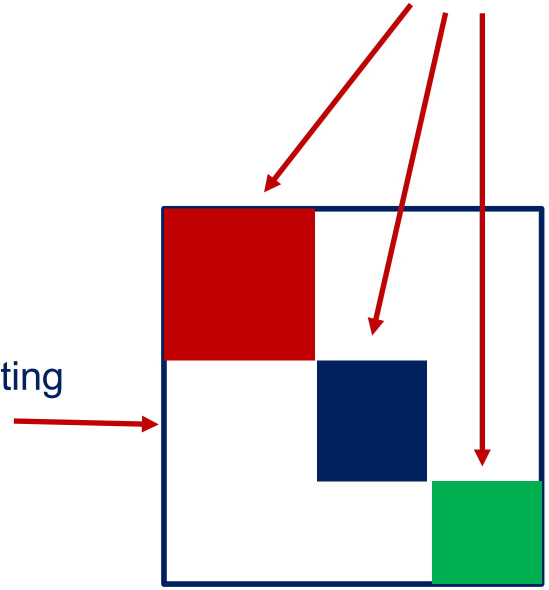
Minus expected number of edges

an adjacency matrix overview

$$Q_2 = \sum_{ij} p_{ij} \cdot \eta(c_i = c_j)$$

The null model !

- wiring probability $p_{ij} = k_i \cdot k_j / 2L$
- $k_i = \sum_j a_{ij}$ = node degree
- $2L = \sum_i k_i$ = # of stubs



blocks = edges
inside a community

matrix collecting
values p_{ij}

Modularity

a definition

Modularity (normalized $-1 \leq Q \leq 1$)

$$Q = (Q_1 - Q_2)/2L$$

$$= \boxed{1/2L \cdot \sum_{ij} (a_{ij} - k_i \cdot k_j /2L) \cdot \eta(c_i = c_j)}$$

- $Q > 0$ if the edges within groups exceed the (expected) random number
- $Q \in [0.3, 0.7]$ for a significant community structure
- Q grows with size of the graph/number of (well-separated) clusters (Good et al, 2009) and cannot use Q to compare graphs very different in size

Modularity

matrix formalization for undirected networks

original adjacency matrix (symmetric, can be fractional) $\boldsymbol{A}_0$

sum of the entries of $\mathbf{A}_0$ $D_0 = \boldsymbol{1}^\mathrm{T} \boldsymbol{A}_0 \boldsymbol{1}$

corresponds to 2L

normalised adjacency matrix (entries sum up to 1) $\boldsymbol{A} = \boldsymbol{A}_0 / D_0$ ← corresponds to $a_{ij}/2L$

normalised degree vector (entries sum up to 1) $\boldsymbol{d} = \boldsymbol{A}\,\boldsymbol{1}$ ← corresponds to $k_i/2L$

community assignment matrix (binary, one active entry per column)

$$\boldsymbol{C} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

← community 1

← community 2

← community 3

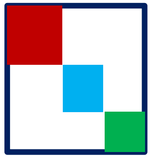
nodes 1 and 2 belong to community 1

nodes 4, 5 and 6 belong to community 3

modularity

$$\boxed{Q = \mathrm{trace}(\boldsymbol{C}\,(\boldsymbol{A} - \boldsymbol{d}\,\boldsymbol{d}^\mathrm{T})\,\boldsymbol{C}^\mathrm{T})}$$

corresponds to selecting blocks pertaining to communities

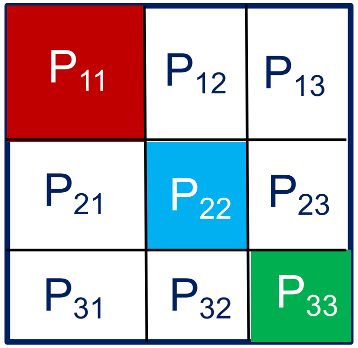


Modularity

another useful matrix formalization for undirected networks

$$\begin{tikzcd}
\boldsymbol{A}_0 \arrow[dr] & & \\
D_0 = \boldsymbol{1}^{\mathrm{T}} \boldsymbol{A}_0 \, \boldsymbol{1} \arrow[r] & \boldsymbol{A} = \boldsymbol{A}_0 / D_0 \arrow[r] & \boldsymbol{P}_{CC} = \boldsymbol{C}\,\boldsymbol{A}\,\boldsymbol{C}^T
\end{tikzcd}$$

can be interpreted as a probability matrix linking communities, its entries are the sum of the links of **A** from community i to community j



P_{11}	P_{12}	P_{13}
P_{21}	P_{22}	P_{23}
P_{31}	P_{32}	P_{33}

can be interpreted as the probability vector of communities

$$\boldsymbol{p}_C = \boldsymbol{P}_{CC}\,\boldsymbol{1} = \boldsymbol{C}\,\boldsymbol{A}\,\boldsymbol{1} = \boldsymbol{C}\,\boldsymbol{d}$$

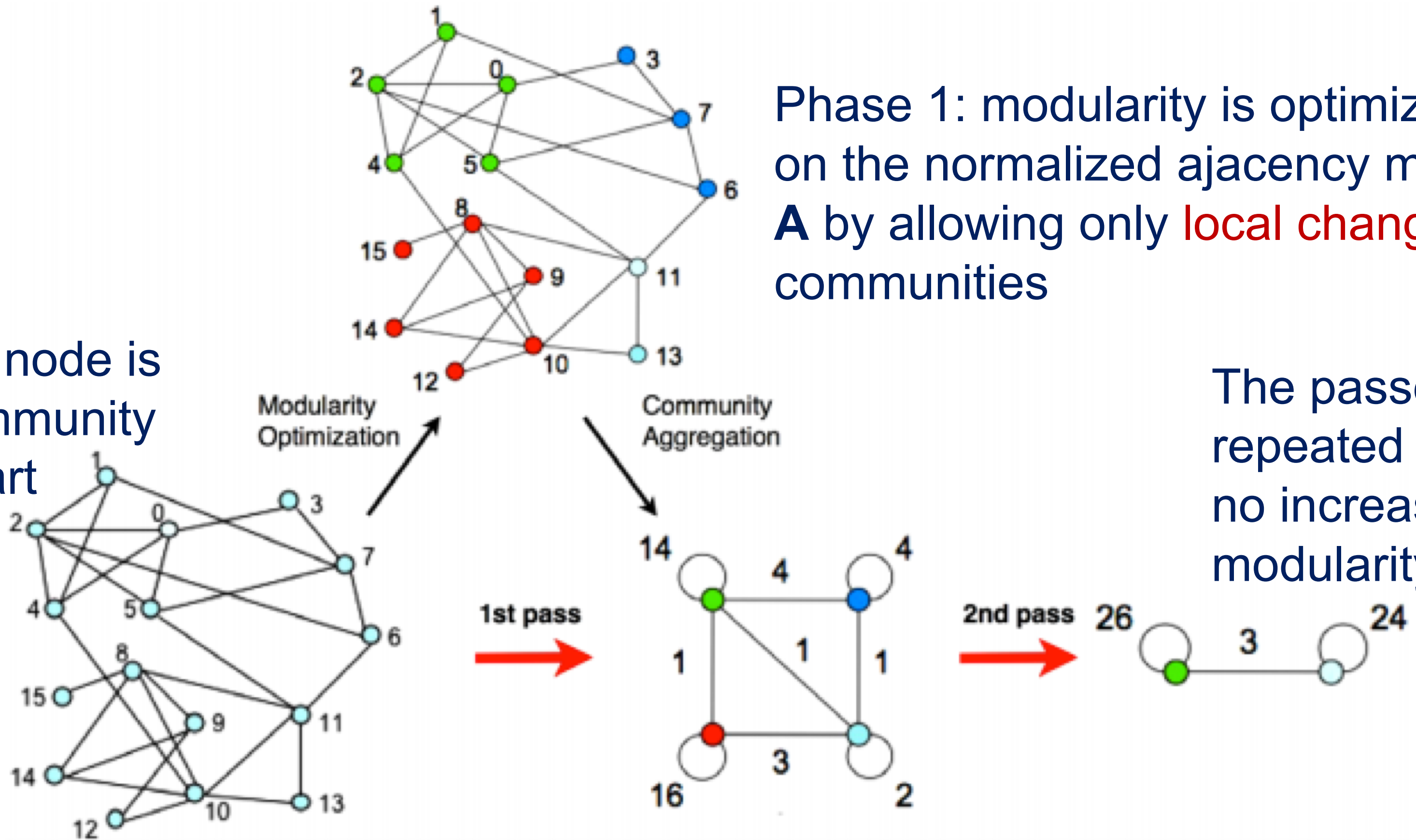
modularity

$$\boxed{Q = \mathrm{trace}(\boldsymbol{P}_{CC} - \boldsymbol{p}_C\,\boldsymbol{p}_C^T)}$$

The Louvain algorithm

Blondel, Guillaume, Lambiotte, Lefebvre, Fast unfolding of communities in large networks (2008)

https://arxiv.org/abs/0803.0476



Each node is a community @ start

Phase 1: modularity is optimized on the normalized ajacency matrix **A** by allowing only local changes of communities

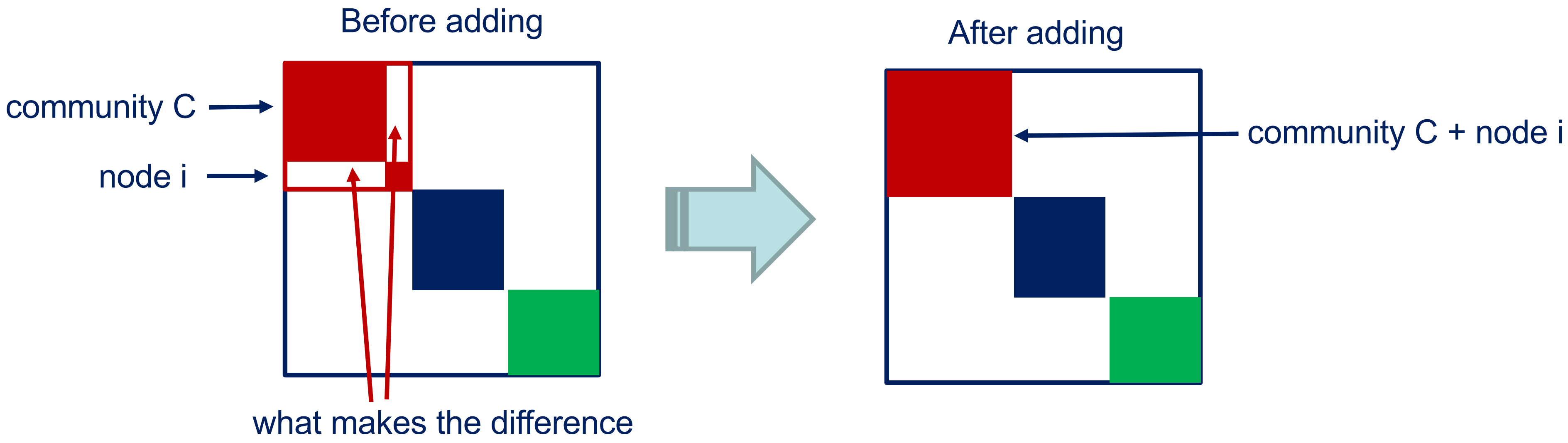
The passes are repeated iteratively until no increase of modularity is possible

Phase 2: the communities found are aggregated (sum of links) in order to build a new network of communities with normalized adjacency matrix $\boldsymbol{P}_{CC}$

Local changes in Louvain

as elementary calculations ensuring scalability

Adding a separate node to a community: increment *ΔQ* in modularity



$$\Delta Q = 2\,\boldsymbol{c}^{T}\,(\boldsymbol{a}_i - d_i\,\boldsymbol{d})$$

binary vector identifying community C

ith column of **A**

ith entry of **d**

- Can be used (with inverse sign) to remove node *i* from a community
- Node *i* is placed in the community ensuring the maximum gain (and positive)
- Easy to calculate, i.e., scalable

Characteristics of Louvain

what makes it interesting

- Implements modularity optimization
- Scalable (low complexity)
- Effective
- Available as the reference implementation in any programming language
- A greedy technique (the order of nodes is selected at random)

can be mitigated by consensus clustering

Consensus clustering

the rationale

Applying Louvain *P* times to a network ***A*** yelds different partitions, but we expect that these are somehow related

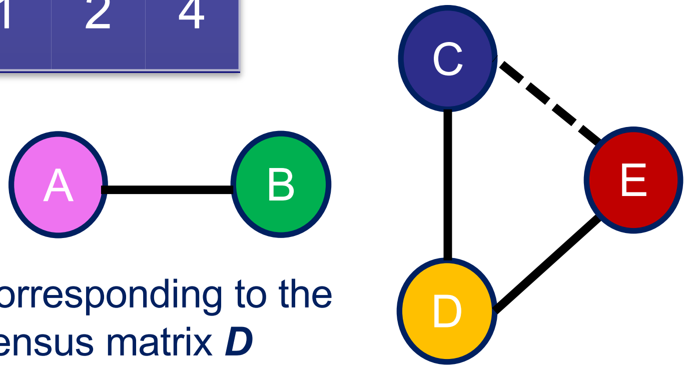
1	1	3	2
1	2	3	2
2	3	2	1
2	4	1	1
3	5	1	1

P=4 community assignments

We capture the recurrent patterns through a consensus matrix ***D***, whose entries correspond to the fraction of times two nodes appear in the same community

4	3	0	0	0
3	4	0	0	0
0	0	4	2	1
0	0	2	4	2
0	0	1	2	4

5x5 consensus matrix (unnormalized)



network corresponding to the consensus matrix ***D***

Consensus clustering

Lancichinetti & Fortunato, Consensus clustering in complex networks (2012)
https://www.nature.com/articles/srep00336

Apply Louvain to ***A*** to yield *P* community detections ***C***$_P$ (partitions)

1. Compute the consensus matrix ***D***
   - D_{ij} is the fraction of partitions in which vertices *i* and *j* are assigned to the same cluster in ***C***$_P$
   - entries below a chosen threshold are set to zero
2. Apply Louvain to ***D*** to yield a new ***C***$_P$
   - if the partitions are all equal, stop
   - otherwise go back to 1.

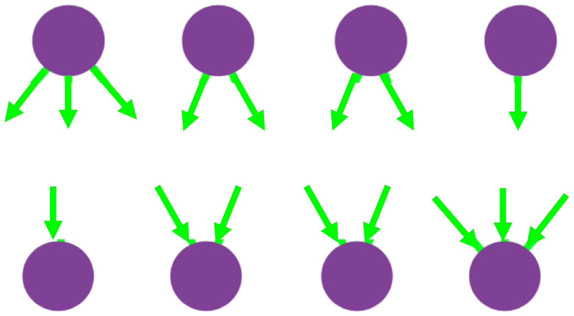
Cycle until convergence

Generalizing modularity

directed and signed networks

The null model for a directed network

the role of in- and out-degree



1. unwire nodes by breaking edges

   but keep stubs and their direction

   so that nodes keep their in/out degree

2. rewire stubs at random, linking output stubs to input stubs

Rewiring probability $j \rightarrow i$ is $p_{ij} = k_j^{out} \ k_i^{in} / L$

number of available links

number of trials from node j

probability of linking to node i

Modularity

matrix formalization for directed networks

original adjacency matrix
(asymmetric, can be fractional) A_0

sum of the
entries of $\mathbf{A}_0$ $D_0 = \mathbf{1}^\mathrm{T} A_0 \mathbf{1}$

corresponds to L

normalised adjacency matrix
(entries sum up to 1) $A = A_0 / D_0$ ← corresponds to a_{ij}/L

normalised in-degree vector
(entries sum up to 1) $d_{in} = A \mathbf{1}$ ← corresponds to k_i^{in}/L

normalised out-degree vector
(entries sum up to 1) $d_{out} = A^T \mathbf{1}$ ← corresponds to k_j^{out}/L

community assignment matrix
(binary, one active entry per column) C

not equivalent to making
A symmetric via ½($\mathbf{A}+\mathbf{A}^\mathrm{T}$)

modularity

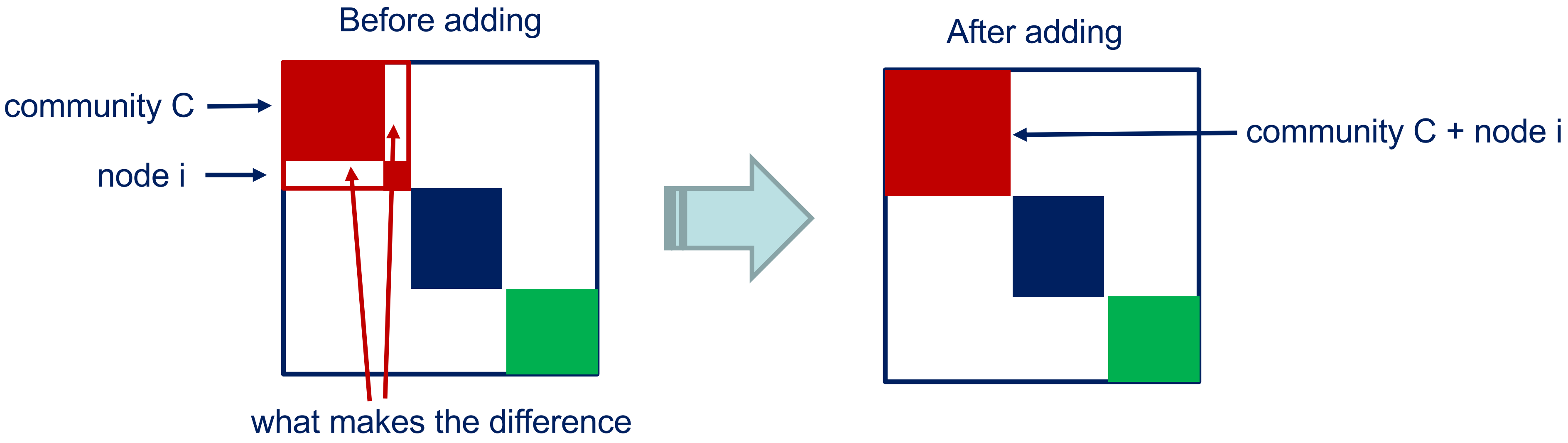
$$Q = \mathrm{trace}(C\,(A - d_{in}\, d_{out}^{\mathrm{T}})\, C^{\mathrm{T}})$$

Leicht and Newman, "Community structure in directed networks." (2008)
https://link.aps.org/pdf/10.1103/PhysRevLett.100.118703

Local changes in Louvain

in the directed case

Adding a separate node to a community: increment ΔQ in modularity



community C

node i

community C + node i

what makes the difference

$$\Delta Q = \mathbf{c}^T (\mathbf{a}_i + \mathbf{r}_i - d_i^{out}\, \mathbf{d}_{in} - d_i^{in}\, \mathbf{d}_{out})$$

binary vector identifying community C

ith column of **A**

ith row of **A**

ith entry of $\mathbf{d}_{in}$ and $\mathbf{d}_{out}$

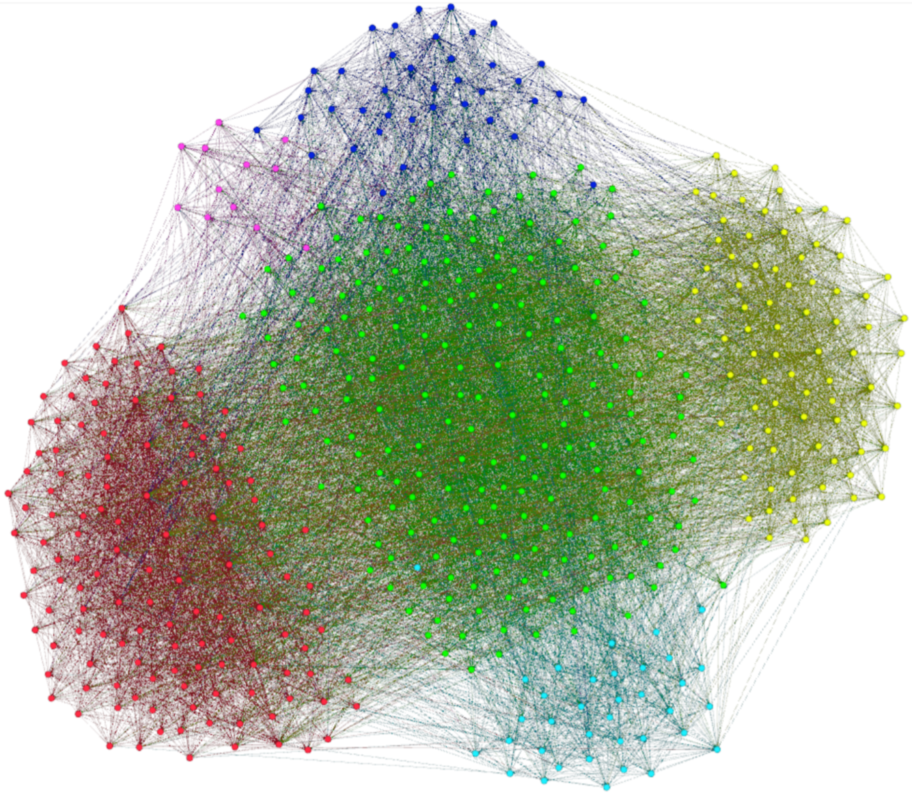
- Keeps its simplicity
- But be aware that it is not always implemented in standard packages

Directed versus undirected Louvain

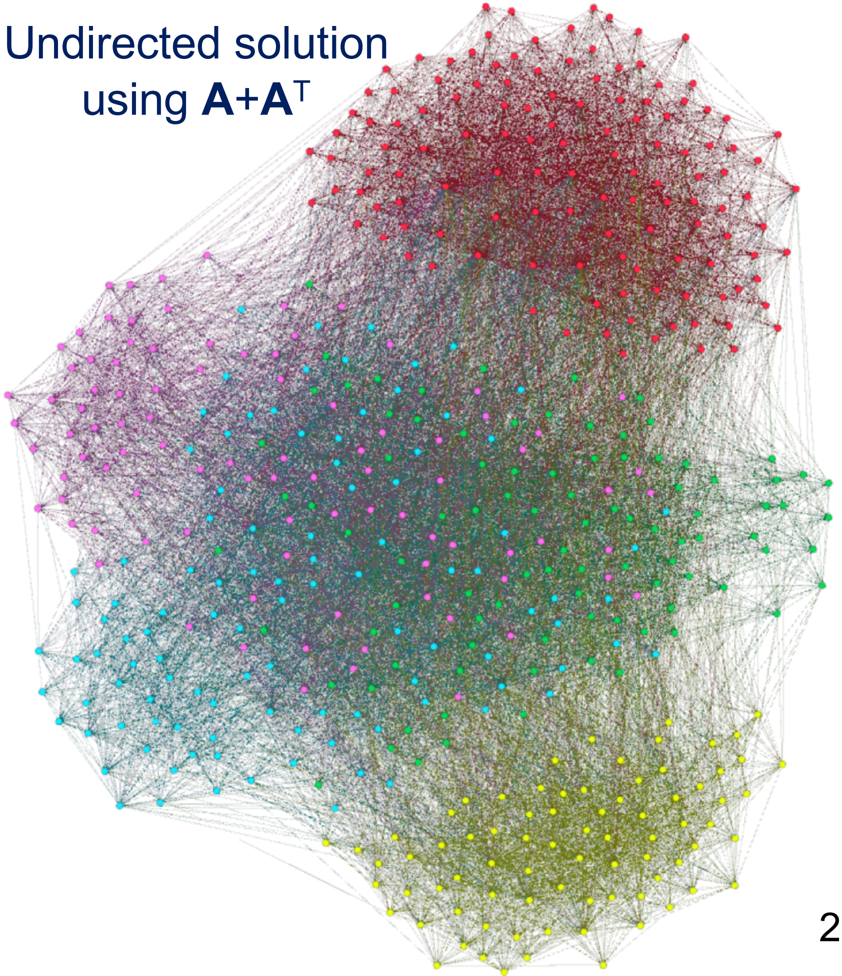
Dugué and Perez, *Directed Louvain: maximizing modularity in directed networks* (2015)

https://hal.science/hal-01231784/document

Directed solution using **A** (colors correspond to ground thruth)

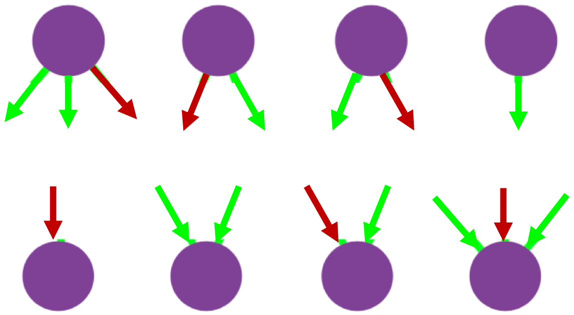


Undirected solution using $\mathbf{A}+\mathbf{A}^T$



The null model for a signed network

the role of positive and negative components



1. unwire nodes by breaking edges

   but keep stubs, their direction, and sign

2. rewire stubs at random, linking output stubs to input stubs, with same sign

Rewiring probability $j \rightarrow i$ is $p_{ij} = k_j^{out+} \; k_i^{in+} / L^+ - k_j^{out-} \; k_i^{in-} / L^-$

positive
contributions with
positive sign

negative
contributions with
negative sign

Modularity

matrix formalization for signed and directed networks

original adjacency matrix (asymmetric, signed)	$\boldsymbol{A}_0 = \boldsymbol{A}_0^+ - \boldsymbol{A}_0^-$ $\qquad D_0^\pm = \boldsymbol{1}^\mathrm{T} \boldsymbol{A}_0^\pm \boldsymbol{1}$
normalised adjacency matrices (entries sum up to 1)	$\boldsymbol{A}^\pm = \boldsymbol{A}_0^\pm / D_0^\pm$
normalised in-degree vectors (entries sum up to 1)	$\boldsymbol{d}_{in}^\pm = \boldsymbol{A}^\pm \boldsymbol{1}$
normalised out-degree vectors (entries sum up to 1)	$\boldsymbol{d}_{out}^\pm = (\boldsymbol{A}^\pm)^T \boldsymbol{1}$
community assignment matrix (binary, one active entry per column)	$\boldsymbol{C}$
mixing constant	$\alpha = D_0^+ / (D_0^+ + D_0^-)$

modularity

$$\boxed{\begin{aligned} Q = {} & \alpha \,\mathrm{trace}(\boldsymbol{C}\, (\boldsymbol{A}^+ - \boldsymbol{d}_{in}^+ \boldsymbol{d}_{out}^{+\mathrm{T}})\, \boldsymbol{C}^\mathrm{T}) \\ & - (1-\alpha)\, \mathrm{trace}(\boldsymbol{C}\, (\boldsymbol{A}^- - \boldsymbol{d}_{in}^- \boldsymbol{d}_{out}^{-\mathrm{T}})\, \boldsymbol{C}^\mathrm{T}) \end{aligned}}$$

Traag, Bruggeman, "Community detection in networks with positive and negative links." (2009)
https://journals.aps.org/pre/pdf/10.1103/PhysRevE.80.036115

Increasing the resolution

boosting or decreasing the role of the null model

The resolution limit:

- prevents the algorithms in detecting small communities
- arises because the null model assumes that each node has an equal probability of connecting to every other node

Can be mitigated by controlling the strength of the null model, i.e.:

it is implemented in standard packages

$$\boxed{Q = \mathrm{trace}(\boldsymbol{P}_{CC} - \gamma\, \boldsymbol{p}_C \boldsymbol{p}_C^{T})}$$

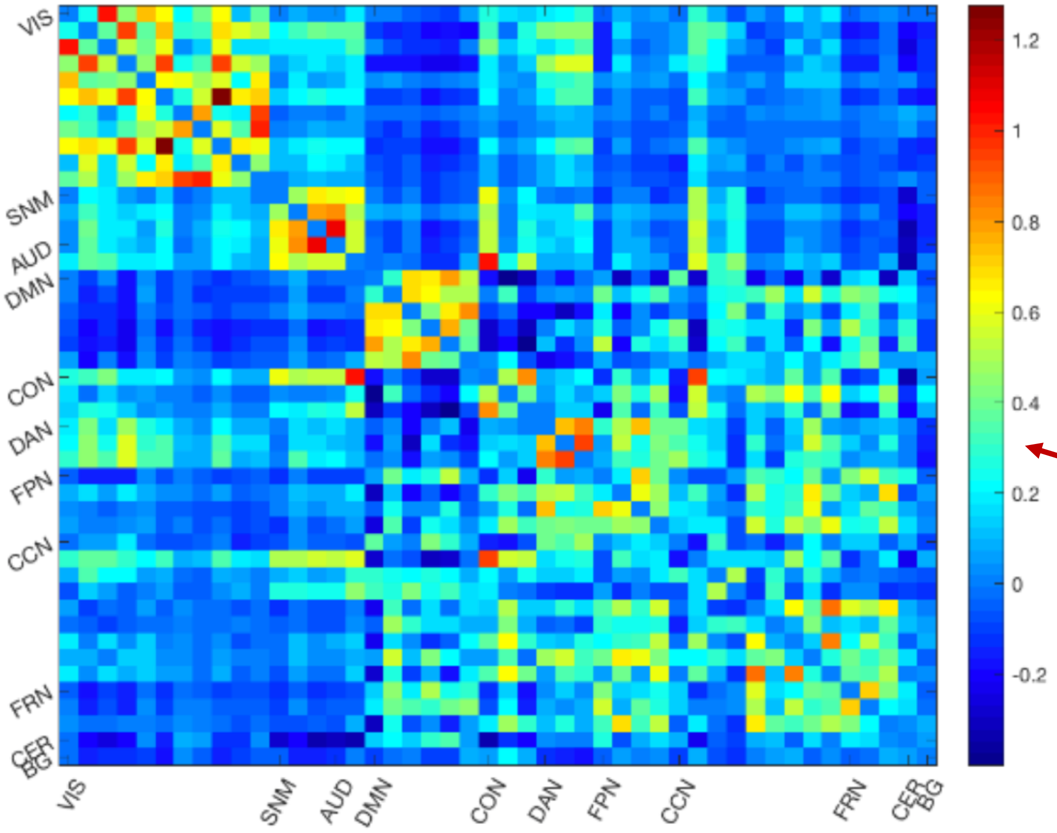
tunable value $\gamma>1$ increases the number of communities

$\gamma<1$ decreases it

An application example

interconnections in brain regions through fMRI data

fMRI correlation matrix



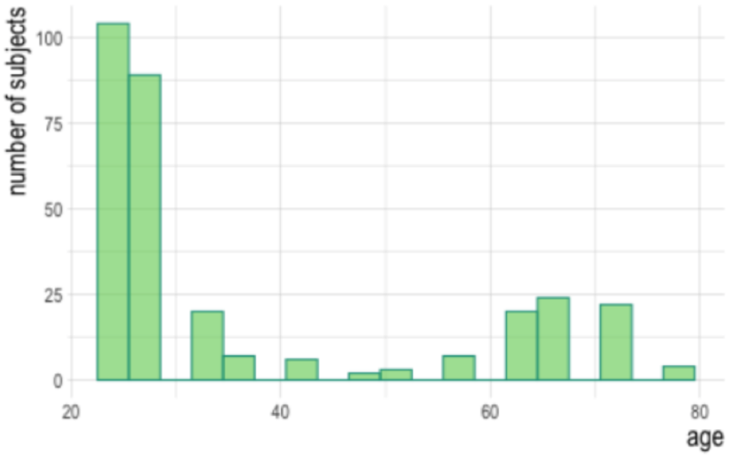
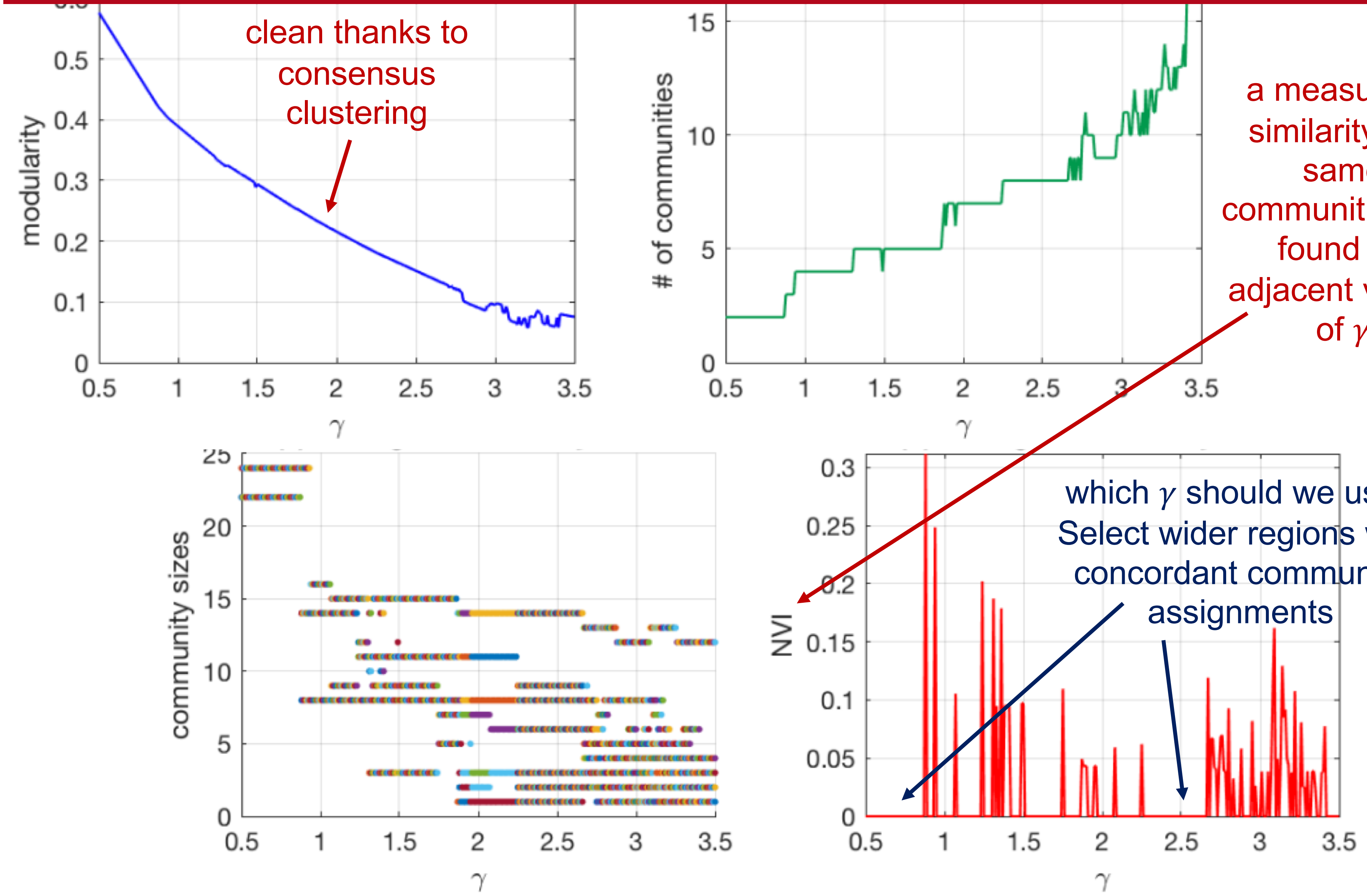


Figure 1: Age histogram of the 308 subjects under study

Can run Louvain on it to
identify meaningful
patterns (communities)
for each subject

On the dependency on γ

Nastaran Amini, community and hub detection in human functional brain networks, *master thesis*, (2020)



clean thanks to consensus clustering

a measure of similarity, 0 if same communities are found for adjacent values of γ

which γ should we use? Select wider regions with concordant community assignments

Modularity in overlapping communities

relaxing the reference model

Target problem:
maximize $Q = \mathrm{trace}(\boldsymbol{C}\,\boldsymbol{B}\,\boldsymbol{C}^T)$, $\boldsymbol{B} = \boldsymbol{A} - \boldsymbol{d}_{in}\boldsymbol{d}_{out}^T$
wrt $\boldsymbol{C}$ subject to $\boldsymbol{C} \geq \boldsymbol{0}$
$\boldsymbol{C}^T\,\mathbf{1} = \mathbf{1}$
~~**C** binary~~

we drop the binary condition to
allow for overlapping communities

- Can be implemented by alternate search on nodes (possibly in a random order) starting from the output of a standard Louvain approach
- It improves modularity
- It is not available in standard packages

Alternate search basics

optimizing the community coefficients of node *i*

Target problem:

maximize $Q = \mathrm{trace}(\boldsymbol{C}\,\boldsymbol{B}\,\boldsymbol{C}^{\mathrm{T}})$

wrt $\boldsymbol{c}_i$ subject to $\boldsymbol{c}_i \geq \boldsymbol{0},\ \boldsymbol{c}_i^{\mathrm{T}}\,\mathbf{1} = 1$

Here $\boldsymbol{c}_i$ is the *i*th column of $\boldsymbol{C}$

Gets a reasonable form

by writing $\boldsymbol{C} = \boldsymbol{C}_{\sim i} + \boldsymbol{c}_i\,\delta_i^{T}$

*i*th column of ***C*** only

*i*th column of ***C*** set to ***0***

binary vector active
only in position i

$$Q = \mathrm{trace}(\boldsymbol{c}_i\,\delta_i^{T}\,\boldsymbol{B}\,\delta_i\,\boldsymbol{c}_i^{\mathrm{T}}) + \mathrm{trace}(\boldsymbol{C}_{\sim i}\,(\boldsymbol{B}+\boldsymbol{B}^{\mathrm{T}})\,\delta_i\,\boldsymbol{c}_i^{\mathrm{T}}) + \mathrm{const}$$

$$= B_{ii}\,|\boldsymbol{c}_i|^2 + \boldsymbol{c}_i^{\mathrm{T}}\,\boldsymbol{C}_{\sim i}\,(\boldsymbol{b}_i+\boldsymbol{r}_i^{\mathrm{T}}) + \mathrm{const}$$

*i*th column of ***B***   *i*th row of ***B***

Alternate search algorithm

Part 1

Target problem:

maximize wrt $\boldsymbol{c}_i$

$$\tfrac{1}{2}\, a\, |\boldsymbol{c}_i|^2 + \boldsymbol{c}_i^{\mathsf{T}} \boldsymbol{v}$$

subject to $\boldsymbol{c}_i \geq \boldsymbol{0},\ \boldsymbol{c}_i^{\mathsf{T}} \boldsymbol{1} = 1$

with $a = B_{ii}$, $\boldsymbol{v} = \tfrac{1}{2}\, \boldsymbol{C}_{\sim i} (\boldsymbol{b}_i + \boldsymbol{r}_i^{\mathsf{T}})$

concave

the value lies in the range of the values of $\boldsymbol{v}$

Case 1: $a \geq 0$

$$\operatorname{argmax} \tfrac{1}{2}\, a\, |\boldsymbol{c}_i|^2 + \boldsymbol{c}_i^{\mathsf{T}} \boldsymbol{v}$$

subject to $\boldsymbol{c}_i \geq \boldsymbol{0},\ \boldsymbol{c}_i^{\mathsf{T}} \boldsymbol{1} = 1$

Solution: $\boldsymbol{c}_i = \delta_j, \quad j = \operatorname{argmax}_i \boldsymbol{v}_i$

we force it to the maximum value of $\boldsymbol{v}$

Alternate search algorithm

Part 2

convex

$\boldsymbol{u} = -\boldsymbol{v}/a$

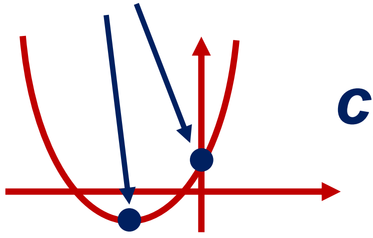
Case 2: a<0

$$\text{argmin } \tfrac{1}{2}\,|\boldsymbol{c}_i|^2 - \boldsymbol{c}_i^T \boldsymbol{u}$$
$$\text{subject to } \boldsymbol{c}_i \geq \boldsymbol{0},\ \boldsymbol{c}_i^T\, \mathbf{1} = 1$$

Solution: we exploit the Lagrangian

$$L = \tfrac{1}{2}|\boldsymbol{c}_i|^2 - \boldsymbol{c}_i^T \boldsymbol{u} + \lambda\,(\boldsymbol{c}_i^T\, \mathbf{1} - 1)$$
$$\nabla L = \boldsymbol{c}_i - \boldsymbol{u} + \lambda\, \mathbf{1} = \mathbf{0}$$

if the global minimum is below 0, then 0 is the best choice



$$\boldsymbol{c}_i = [\boldsymbol{u} - \lambda\, \mathbf{1}]^+ \text{ where } \lambda \text{ is such that } \mathbf{1}^T\, [\boldsymbol{u} - \lambda\, \mathbf{1}]^+ = 1$$

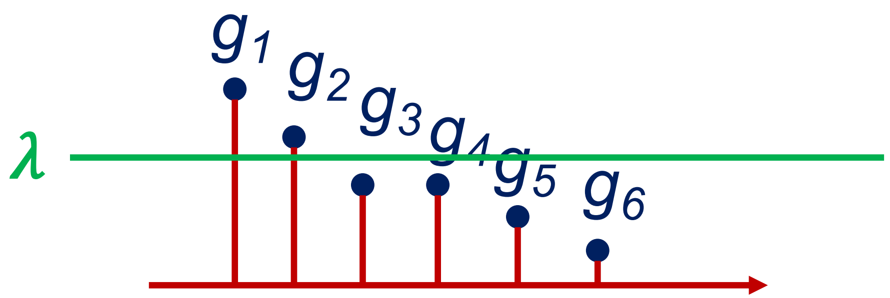
positive part operator: $[x]^+$ is x for x>0 and 0 otherwise

Alternate search algorithm

Identifying the correct λ

Problem $\boxed{\text{find } \lambda \text{ such that } \mathbf{1}^T [\boldsymbol{u} - \lambda\, \mathbf{1}]^+ = 1}$

Solution: sort vector $\boldsymbol{u}$ in decreasing order → $\boldsymbol{g}$



if λ is in between g_2 and g_3,
then it must be that
$g_1 + g_2 - 2g_3 \geq 1$

- $\boldsymbol{z} = [\ \mathrm{cumsum}(\boldsymbol{g}_{1:N-1}) - (1:N-1) \cdot \boldsymbol{g}_{2:N},\ \infty\ ]$
- let z_n be the first entry of $\boldsymbol{z}$ satisfying $z_n \geq 1$
- hence λ lies between $\boldsymbol{g}_n$ and $\boldsymbol{g}_{n+1}$ (use $\boldsymbol{g}_{N+1} = -\infty$)

- therefore $\lambda = (\mathrm{sum}(\boldsymbol{g}_{1:n}) - 1) / n \geq \boldsymbol{g}_{N+1}$

Modularity in overlapping communities

comments

- ☐ It provides a binary outcome only for $B_{ii} \geq 0$ (single community)
- ☐ In all other cases the result is fractional (multiple communities) but not all the communities are necessarily active

- ☐ Would be nice to see it implemented by someone in the class ☺

The spectral approach

for modularity optimization

The two communities case

a compact modularity expression

modularity $\boxed{Q = \mathrm{trace}(\boldsymbol{C}\,\boldsymbol{B}\,\boldsymbol{C}^{T}), \quad \boldsymbol{B} = \boldsymbol{A} - \boldsymbol{d}\,\boldsymbol{d}^{T}}$

what if we have only two communities ?

$$\boldsymbol{C} = \begin{pmatrix} \mathbf{v} \\ \mathbf{1 - v} \end{pmatrix} \begin{array}{l} \leftarrow \text{community 1} \\ \leftarrow \text{community 2} \end{array}$$

idea: signed vector $\boldsymbol{s} = 2\boldsymbol{v} - 1$

$$\boldsymbol{v} = \tfrac{1}{2}\,(1 + \boldsymbol{s})$$

$$1 - \boldsymbol{v} = \tfrac{1}{2}\,(1 - \boldsymbol{s})$$

+1s identify community 1,
and -1s identify community 2

$$\boxed{Q = \tfrac{1}{2}\,\boldsymbol{s}\,\boldsymbol{B}\,\boldsymbol{s}^{T}} \longleftarrow \text{since } \boldsymbol{B}\,\boldsymbol{1} = \boldsymbol{0}$$

The two communities case

an optimization overview

Target problem:

maximize $\boxed{Q = \tfrac{1}{2}\, \mathbf{s}\, \boldsymbol{B}\, \mathbf{s}^{\mathrm{T}}}$ wrt the binary vector ***s***

a non trivial NP problem

We exploit the eigendecomposition of ***B***

- $\boldsymbol{B} = \sum \boldsymbol{b}_i \boldsymbol{b}_i^T \lambda_i$
- $\boldsymbol{b}_i$ normalized eigenvector $|\boldsymbol{b}_i|=1$
- λ_i eigenvalue

What if we only keep the strongest component?

Target problem revisited:

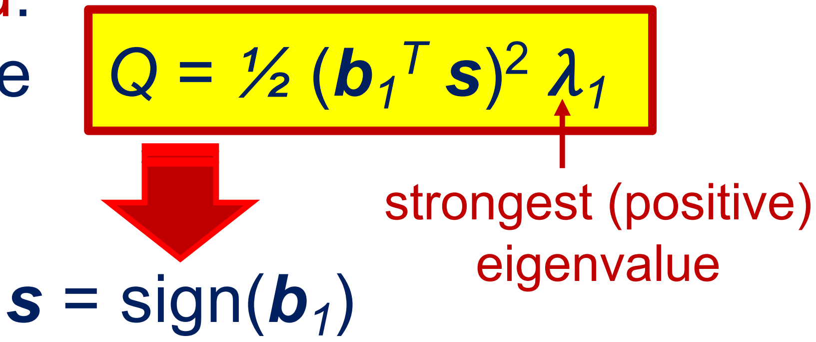
maximize $\boxed{Q = \tfrac{1}{2} \sum_i (\boldsymbol{b}_i^T \mathbf{s})^2 \lambda_i}$

Spectral approach

to modularity optimization

Target problem relaxed:

maximize



$$Q = \tfrac{1}{2} (\boldsymbol{b}_1^T \boldsymbol{s})^2 \lambda_1$$

strongest (positive) eigenvalue

$$\boldsymbol{s} = \mathrm{sign}(\boldsymbol{b}_1)$$

- it is a simple approach (e.g., related to PCA decomposition) that needs to be recursively applied
- Can be refined by switching community of nodes if modularity increases
- Can also be refined by exploiting more than one eigenvalue
- Still, its performance is rather poor, and for this reason it is deprecated

Takeaways

for modularity

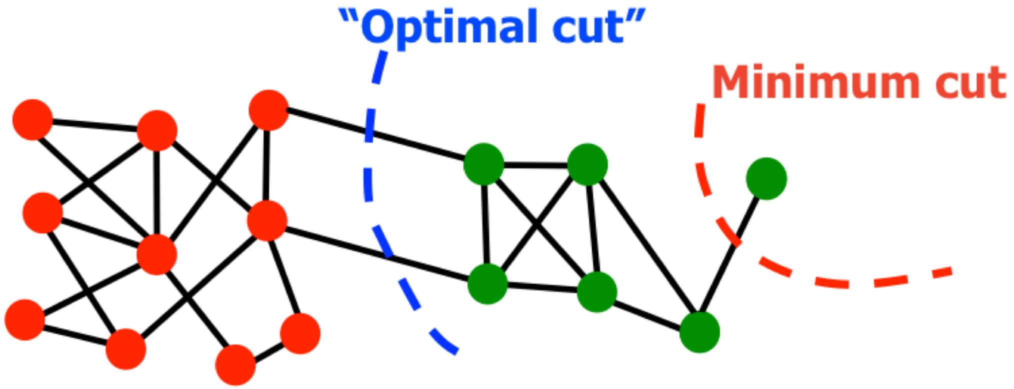
- ❑ Modularity is a key performance metric in community detection
- ❑ Optimizing modularity through the Louvain approach is the bare minimum required in any project
- ❑ Implementation of generalized modularity (directed, signed, overlapping) is highly welcome to get a top grade

The normalized cut criterion

an old (worth citing) alternative to modularity

The minimum cut criterion

towards an alternative measure



- ☐ We want to partition an (undirected) graph in two disjoint groups
- ☐ A good partition is one that

also this criterion is needed

maximizes the # of within-group connections

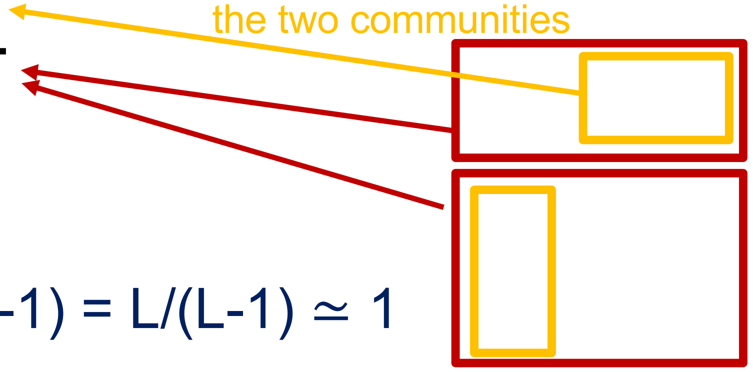
minimizes the # of between-group connections

the minimum cut criterion

The normalized cut criterion

two community case

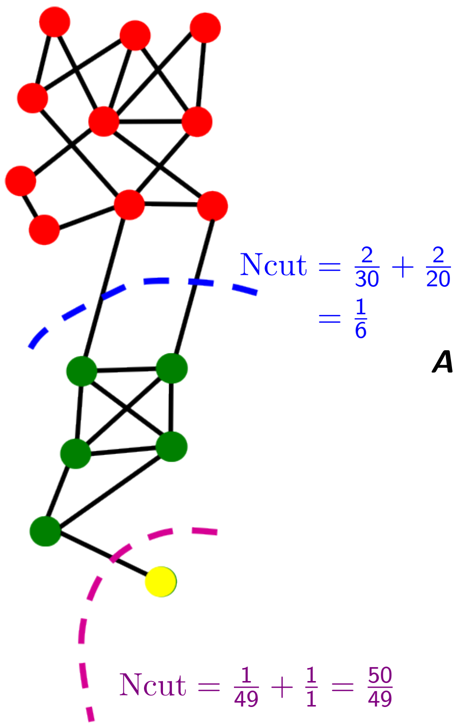
$$\mathrm{Ncut}(A,B) = \frac{\mathrm{cut}(A,B)}{\mathrm{assoc}(A)} + \frac{\mathrm{cut}(B,A)}{\mathrm{assoc}(B)}$$



sum of the links that connect the two communities

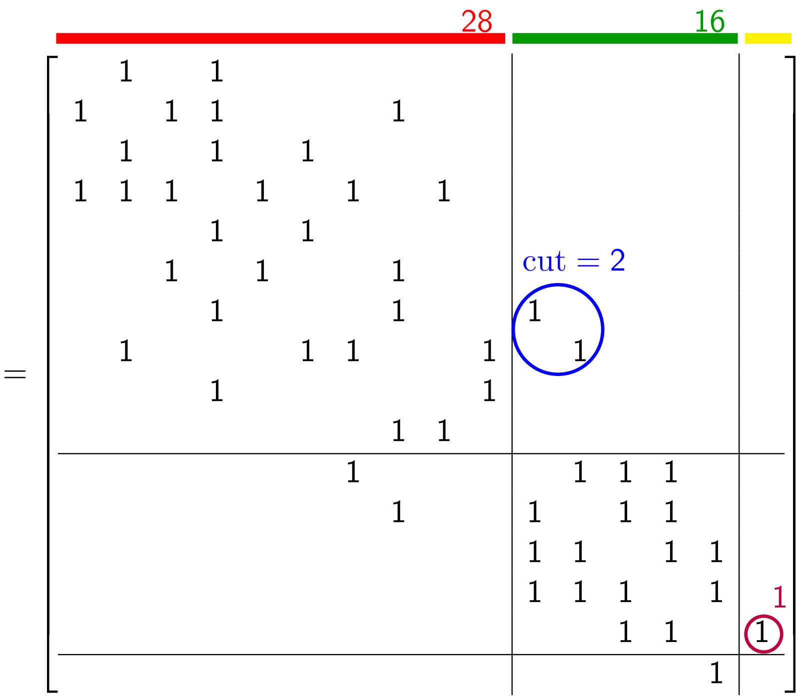
sum of the links departing from each community

- ☐ Produces more balanced partitions
- ☐ Avoids single nodes, Ncut = 1/1 + 1/(L-1) = L/(L-1) ≃ 1



$\mathrm{Ncut} = \frac{2}{30} + \frac{2}{20} = \frac{1}{6}$

$\mathrm{Ncut} = \frac{1}{49} + \frac{1}{1} = \frac{50}{49}$



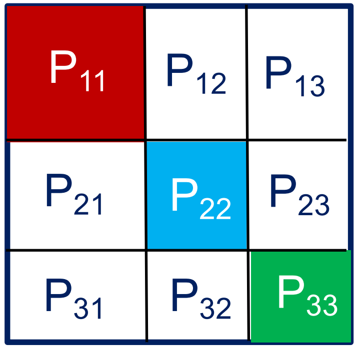
$$\boldsymbol{A} = \left[\begin{array}{cccccccccc|ccccc|c}
 & 1 & & 1 & & & & & & & & & & & & \\
1 & & 1 & 1 & & & & 1 & & & & & & & & \\
 & 1 & & 1 & & 1 & & & & & & & & & & \\
1 & 1 & 1 & & 1 & & 1 & & 1 & & & & & & & \\
 & & & 1 & & 1 & & & & & & & & & & \\
 & & 1 & & 1 & & & 1 & & & & & & & & \\
 & & & 1 & & & & 1 & & & 1 & & & & & \\
 & 1 & & & & 1 & 1 & & & 1 & & 1 & & & & \\
 & & & 1 & & & & & & 1 & & & & & & \\
 & & & & & & & 1 & 1 & & & & & & & \\
\hline
 & & & & & & 1 & & & & & 1 & 1 & 1 & & \\
 & & & & & & & 1 & & & 1 & & 1 & 1 & & \\
 & & & & & & & & & & 1 & 1 & & 1 & 1 & \\
 & & & & & & & & & & 1 & 1 & 1 & & 1 & \\
 & & & & & & & & & & & & 1 & 1 & & \bigcirc\!\!\!\!1 \\
\hline
 & & & & & & & & & & & & & & 1 &
\end{array}\right]$$

The normalized cut criterion

general case versus modularity

$$\begin{tikzcd}
A_0 \arrow[r] & A = A_0 / D_0 \arrow[r] & P_{CC} = C\,A\,C^T \\
D_0 = 1^{\mathrm{T}} A_0\, 1 \arrow[ur] & &
\end{tikzcd}$$

can be interpreted as a probability matrix linking communities, its entries are the sum of the links of **A** from community i to community j



🟥 P_{11}	P_{12}	P_{13}
P_{21}	🟦 P_{22}	P_{23}
P_{31}	P_{32}	🟩 P_{33}

can be interpreted as the probability vector of communities

$$p_C = P_{CC}\,1 = C\,A\,1 = C\,d$$

normalized cut

$$\boxed{\mathrm{Ncut} = \sum_i (p_i - P_{ii}) / p_i > 0}$$

to be minimized

modularity

$$\boxed{Q = \sum_i (P_{ii} - p_i^2) < 1}$$

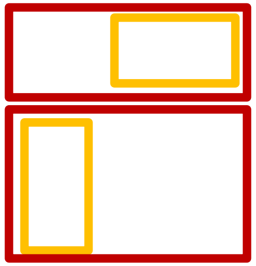
to be maximized

The normalized cut criterion

two communities case

$$C = \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = ½ \begin{pmatrix} 1 + s \\ 1 - s \end{pmatrix}$$

$A =$



$d = A\,1$ →

$\text{cut} = v_1 A v_2^T = ¼\,(1 - s A s^T)$

$\text{assoc}_1 = v_1 d = ½\,(1 + s d)$

$\text{assoc}_2 = 1 - \text{assoc}_1 = ½\,(1 - s d)$

$$\text{Ncut} = \frac{\text{cut}}{\text{assoc}_1} + \frac{\text{cut}}{\text{assoc}_2} = \frac{\text{cut}}{\text{assoc}_1\,\text{assoc}_2} = \frac{1 - s A sT}{1 - (s d)^2}$$

$$\text{Ncut} = \frac{s L sT}{1 - (s d)^2}, \qquad L = \text{diag}(d) - A$$

Laplacian matrix

$L\,1 = 0$

Solving the binary case

an NP complex problem

$$\text{minimize } \frac{\mathbf{s\,L\,s}\boldsymbol{T}}{1-(\mathbf{s\,d})^2}$$

$$\text{s. to } \mathbf{s} \in \{\pm 1\}^N$$

an NP complex problem

work on ***s***

reference norm, weighted by ***d***

$$\mathbf{s} = a\mathbf{1} + b\,\mathbf{u}, \quad \text{with } \langle \mathbf{u},\mathbf{1}\rangle_{\mathbf{d}} = \mathbf{u\,d} = 0$$

$$|\mathbf{u}|_{\mathbf{d}}^2 = 1$$

$$\mathbf{s\,L\,s}^{\mathrm{T}} = b^2\,\mathbf{u\,L\,u}^{\mathrm{T}}$$

$$\mathbf{s\,d} = \langle \mathbf{s},\mathbf{1}\rangle_{\mathbf{d}} = a\,\langle \mathbf{1},\mathbf{1}\rangle_{\mathbf{d}} + b\,\langle \mathbf{u},\mathbf{1}\rangle_{\mathbf{d}} = a$$

$$1 = |\mathbf{s}|_{\mathbf{d}}^2 = a^2\,|\mathbf{1}|_{\mathbf{d}}^2 + b^2|\mathbf{u}|_{\mathbf{d}}^2 - 2ab\,\langle \mathbf{u},\mathbf{1}\rangle_{\mathbf{d}} = a^2+b^2$$

$$\text{minimize } \mathbf{u\,L\,u}^T$$

$$\text{s. to } |\mathbf{u}|_{\mathbf{d}}^2 = 1$$

$$\mathbf{u\,d} = 0$$

$$\mathbf{s} = a\mathbf{1} + \sqrt{1-a^2}\,\mathbf{u} \in \cancel{\{\pm 1\}^N}$$

by construction
sign(***s***) = sign(***u***)
since |a|<1

still an NP complex problem
(but we can relax the binary constraint)

Spectral clustering

a suboptimum solution to the Ncut criterion in the binary case

minimize $\boldsymbol{v}\, L_1\, \boldsymbol{v}^T$
s. to $|\boldsymbol{v}|^2 = 1$
$\boldsymbol{v}\sqrt{\boldsymbol{d}} = 0$

sign($\boldsymbol{s}$) = sign($\boldsymbol{v}$)

$$\xleftarrow[\text{sign}(\boldsymbol{v}) = \text{sign}(\boldsymbol{u})]{\boldsymbol{v} = \text{diag}(\boldsymbol{d})^{½}\, \boldsymbol{u}}$$

minimize $\boldsymbol{u}\, L\, \boldsymbol{u}^T$
s. to $|\boldsymbol{u}|_{\boldsymbol{d}}^2 = 1$
$\boldsymbol{u}\, \boldsymbol{d} = 0$

sign($\boldsymbol{s}$) = sign($\boldsymbol{u}$)

normalized Laplacian

$$\boldsymbol{L_1} = \boldsymbol{I} - \text{diag}(\boldsymbol{d})^{-½}\, \boldsymbol{A}\, \text{diag}(\boldsymbol{d})^{-½}$$

positive semidefinite matrix

$$2 \geq \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{N-1} \geq \lambda_N = 0$$

$\boldsymbol{e}_{N-1}$ is Fiedler’s eigenvector

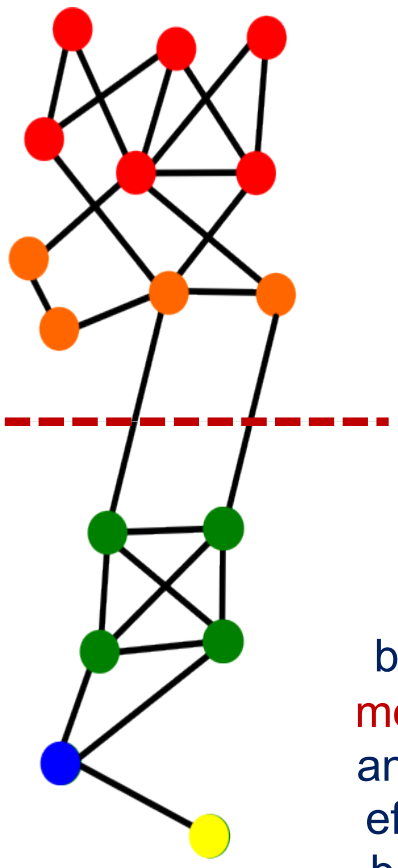
λ_{N-1} is the algebraic connectivity

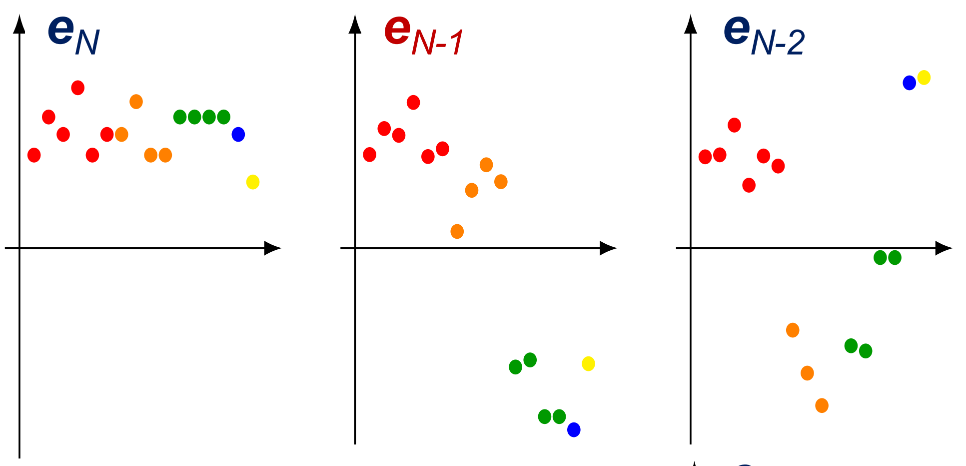
$\boldsymbol{e}_N = \sqrt{\boldsymbol{d}}$

Shi and Malik, “Normalized cuts and image segmentation," 2000
Ng, Jordan, Weiss, ”On spectral clustering: analysis and an algorithm," 2002

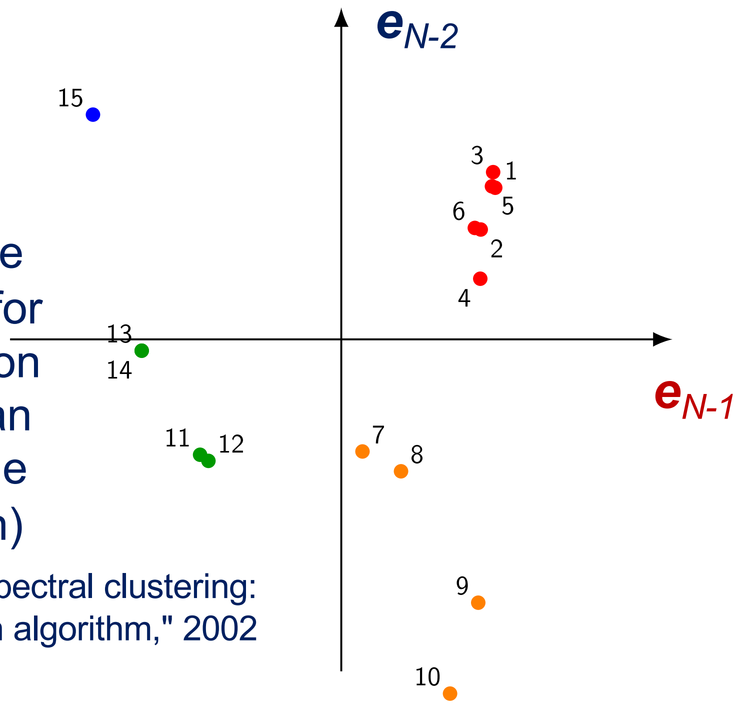
$\boldsymbol{v} = \boldsymbol{e}_{N-1}$
sign($\boldsymbol{s}$) = sign($\boldsymbol{e}_{N-1}$)

An example
of spectral clustering





Fiedler's eigenvector provides the best partition



but we can also use more eigenvectors for an enhanced partition effect (clustering can be performed on the new representation)

Ng, Jordan, Weiss, "On spectral clustering: analysis and an algorithm," 2002

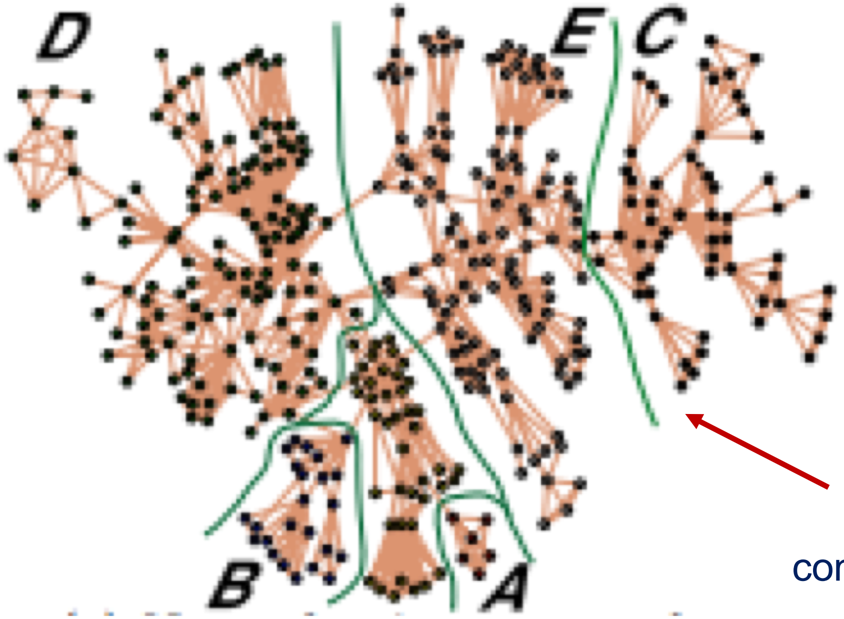
The network community profile

and the role of conductance in the binary community case

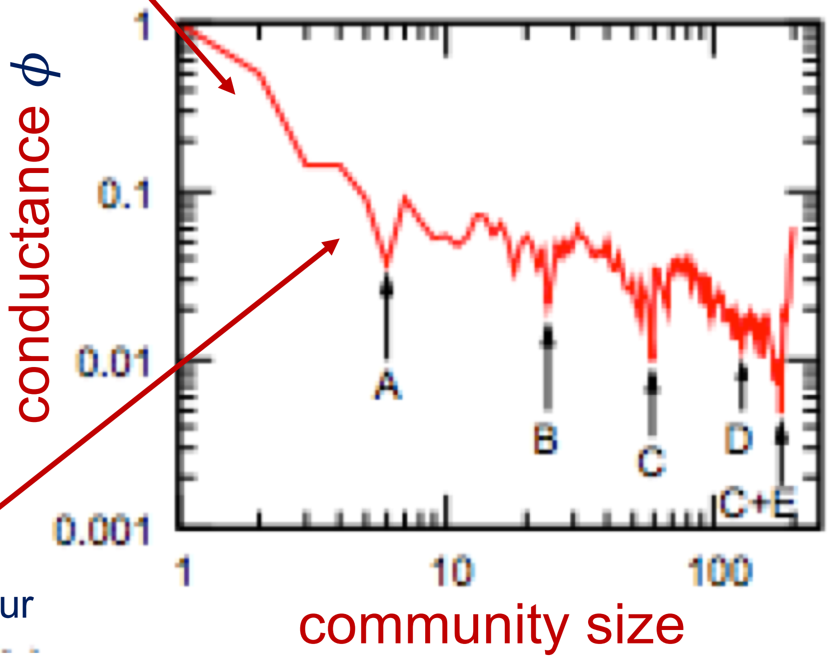
conductance ϕ = cut / min(assoc, *1*-assoc)

run your community detection algorithm(s) many times, for communities of the same size keep the one giving the lowest conductance value

network



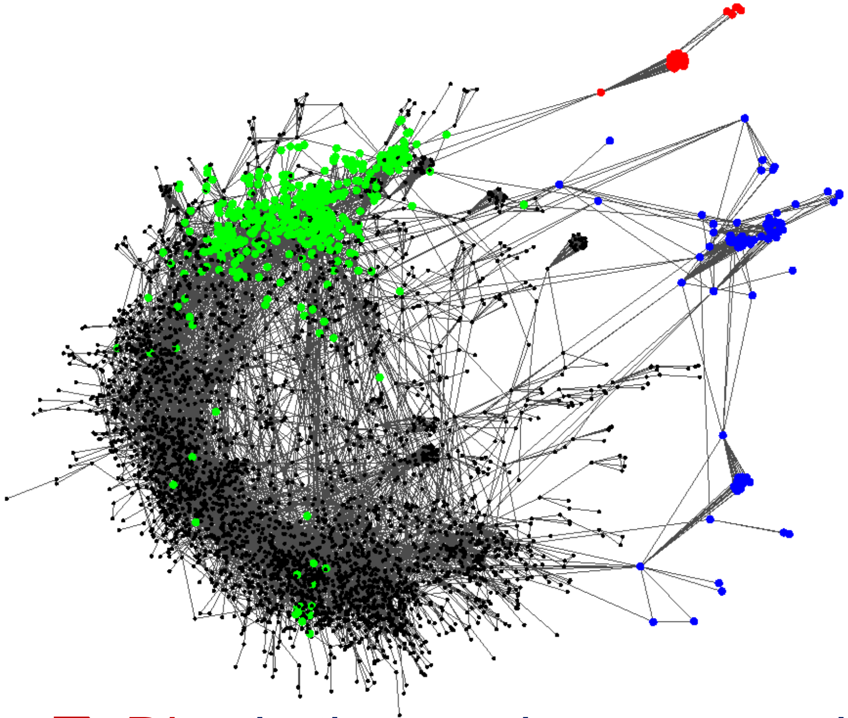
NCP = network community profile

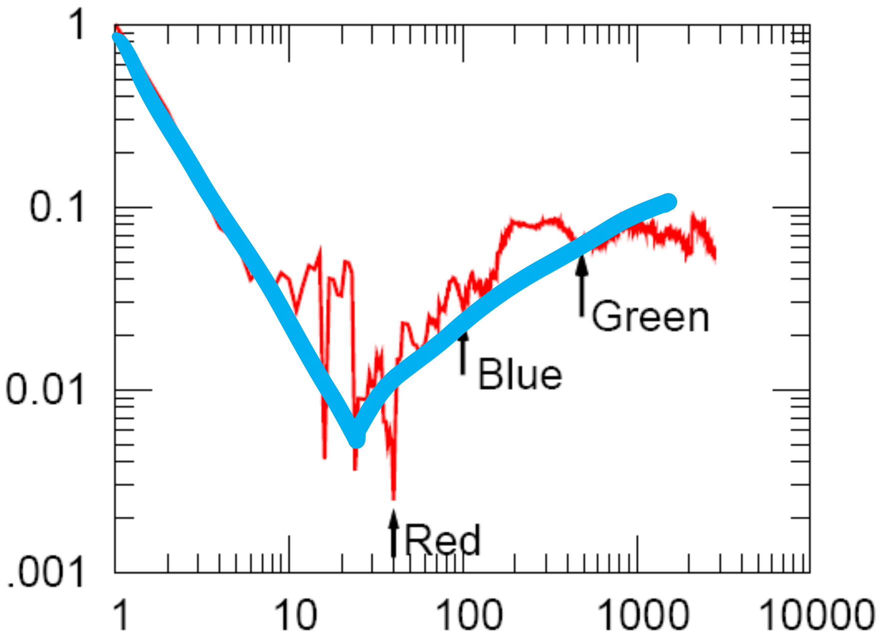


local minima identify good communities in our network

The V shape of the NCP

explaining the core-periphery structure





- ☐ Dips in the graph correspond to the good communities
- ☐ Slope corresponds to the dimensionality of the network
- ☐ The V shape is common in large (social) networks
- ☐ Best communities have about 100 nodes → wiskers
- ☐ Large communities get worse performance → core

Takeaways

for conductance and the normalized cut criterion

- Normalized cut is an old quality measures that set the beginning of image segmentation (clustering) algorithms
- It only works for unsigned undirected graphs
- Its outcomes correlates in general with modularity, although the literature suggests it is a weaker measure
- It is an alternative to modularity, better suited as a quality measure rather than as an optimization approach
- Do not spend any time in implementing any normalised cut optimization
- The performance of the spectral approach is weak, and for this reason it is **deprecated** (but will turn out useful later on)
- The network community profile provides an interesting view on the network structure, would like to see it implemented in jour projects

Infomap

an approach based on PageRank and information theory

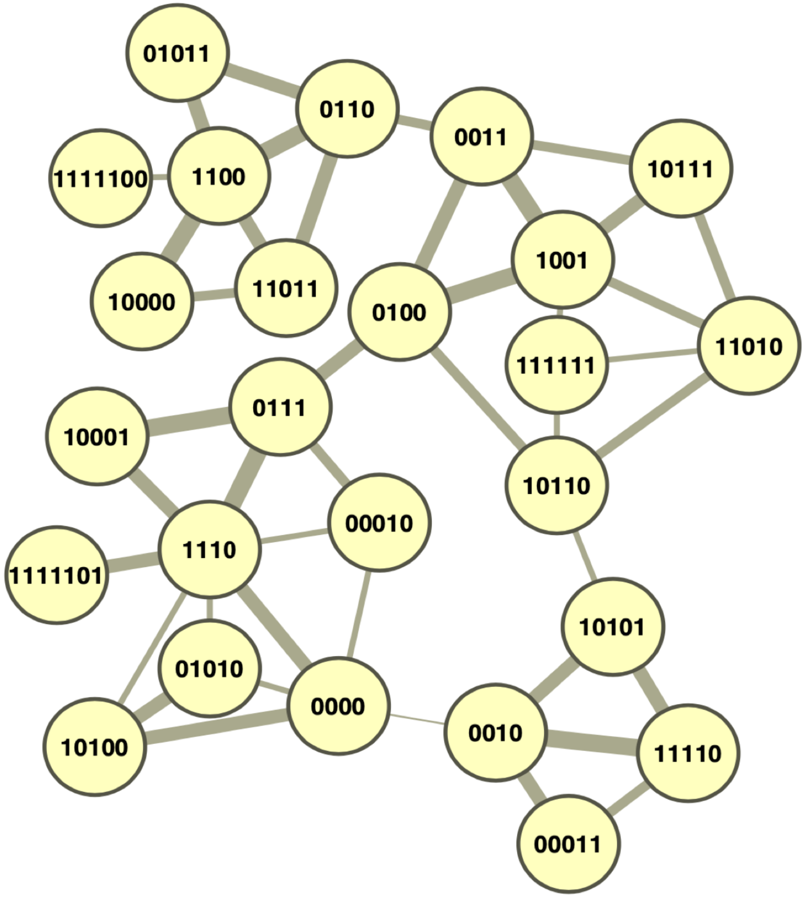
The InfoMap principle

Rosvall, Bergstrom. "Maps of random walks on complex networks reveal community structure." (2008)
https://www.pnas.org/doi/pdf/10.1073/pnas.0706851105?download=true

The most compact way of describing a random walk through a network is by encoding node entries according to their probability ***p*** (e.g., PageRank) using the Huffman procedure that guarantees an average encoding length

$$L \gtrsim H(\boldsymbol{p}) = \sum p_i \log(1/p_i)$$

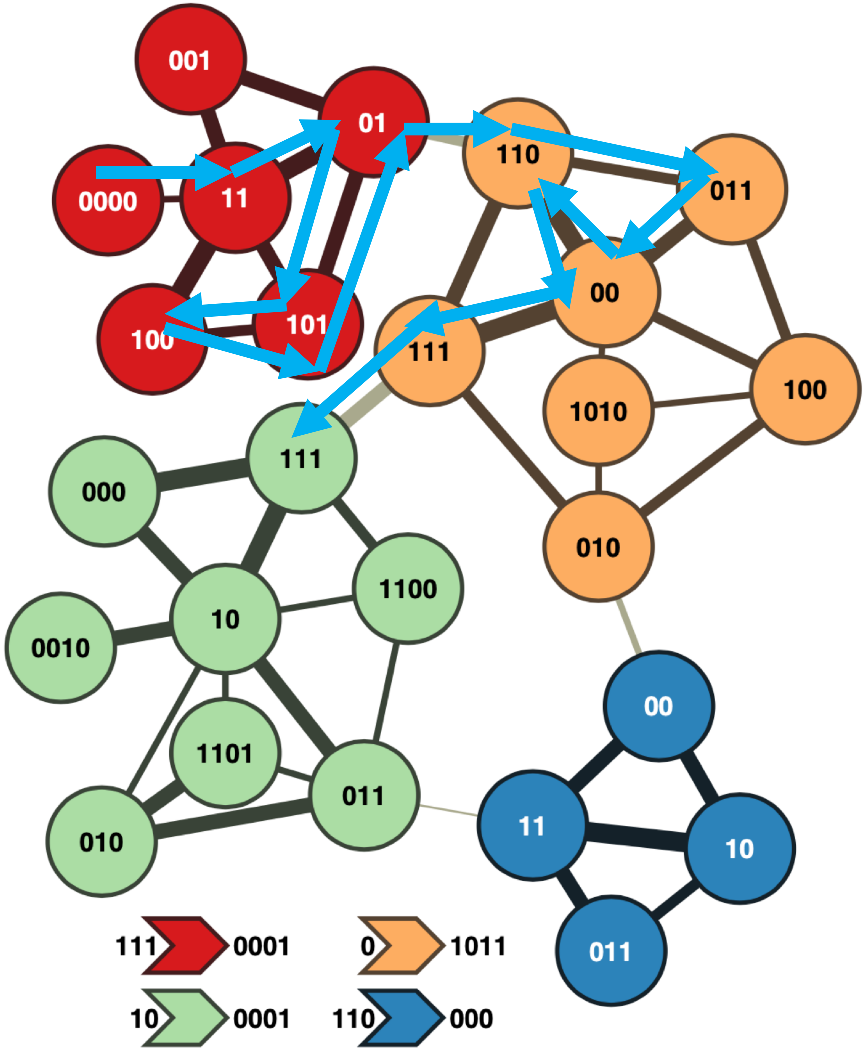
entropy based on the probability vector ***p***



1111100 1100 0110 11011 10000 11011 0110 0011 10111 1001 0011 1001 0100 0111 10001 1110 0111 10001 0111 1110 0000 1110 10001 0111 1110 0111 1110 1111101 1110 0000 10100 0000 1110 10001 0111 0100 10110 11010 10111 1001 0100 1001 10111 1001 0100 1001 0100 0011 0100 0011 0110 11011 0110 0011 0100 1001 10111 0011 0100 0111 10001 1110 10001 0111 0100 10110 111111 10110 10101 11110 00011

The InfoMap principle

the community view



111 0000 11 01 101 100 101 01 0001 0 110 011 00 110 00 111 1011 10
111 000 10 111 000 111 10 011 10 000 111 10 111 10 0010 10 011 010
011 10 000 111 0001 0 111 010 100 011 00 111 00 011 00 111 00 111
110 111 110 1011 111 01 101 01 0001 0 110 111 00 011 110 111 1011
10 111 000 10 000 111 0001 0 111 010 1010 010 1011 110 00 10 011

Under a community assignment we can code the community we are in (each time we switch community) and, separately, the nodes visited inside each community (+ the exiting state)

switching probability

probability of being in community *i*

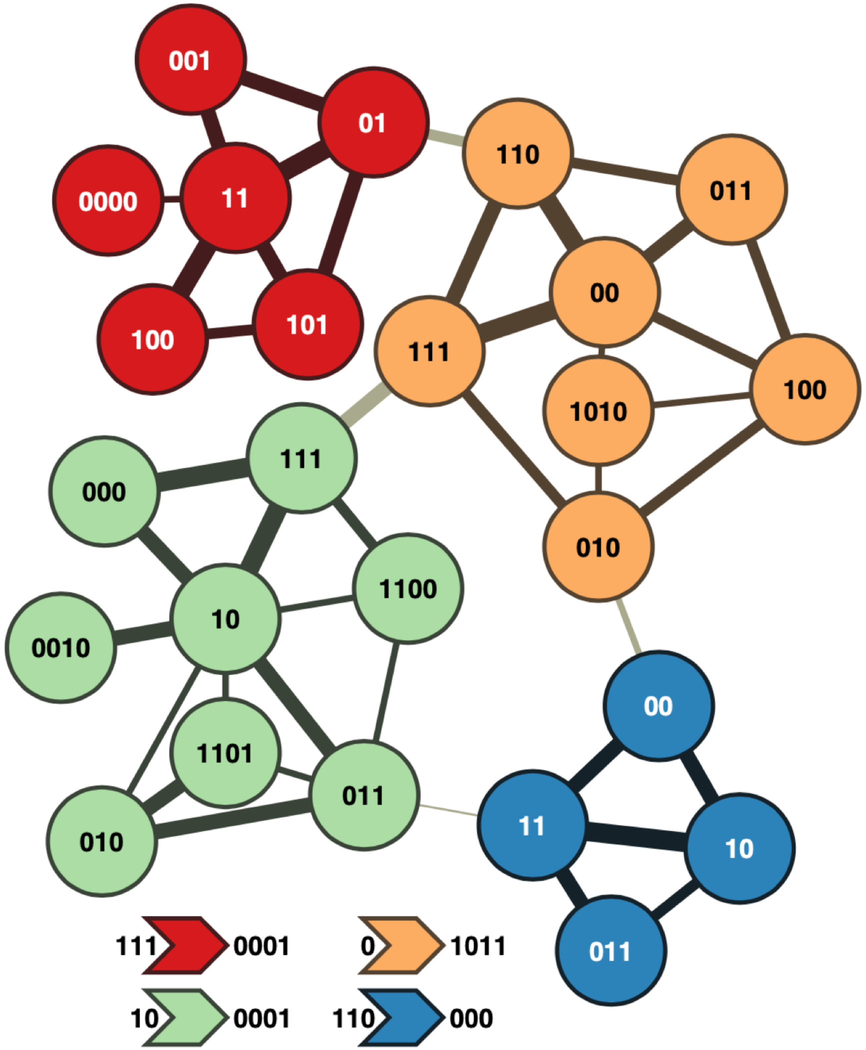
$$L \gtrsim P_S H(\boldsymbol{s}_c) + \sum P_i H(\boldsymbol{u}_i)$$

community ranking (probability vector) under a switch

nodes ranking (probability vector) inside community *i* *(nodes include exit to another community)*

The InfoMap principle

rationale



111 0000 11 01 101 100 101 01 0001 0 110 011 00 110 00 111 1011 10
111 000 10 111 000 111 10 011 10 000 111 10 111 10 0010 10 011 010
011 10 000 111 0001 0 111 010 100 011 00 111 00 011 00 111 00 111
110 111 110 1011 111 01 101 01 0001 0 110 111 00 011 110 111 1011
10 111 000 10 000 111 0001 0 111 010 1010 010 1011 110 00 10 011

- ❑ We want to optimize *L* wrt the community assignment
- ❑ More compact encoding = better community assignment
- ❑ We expect that this corresponds to keeping the random walk inside the communities
- ❑ A flow-based optimization
- ❑ Different (but related to) from modularity (strength-based)

PageRank for nodes

node probability in a random walk with restart

adjacency matrix (can be fractional)

transition probability matrix: $\boldsymbol{M} = \boldsymbol{A}\,\mathrm{diag}^{-1}(\boldsymbol{d}), \qquad \boldsymbol{d} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{1}$

random walk with restart formalization: $\boldsymbol{T} = c\,\boldsymbol{M} + (1\text{-}c)\,\boldsymbol{1}\,\boldsymbol{1}^{\mathrm{T}} / N$ ← equally likely teleport vector

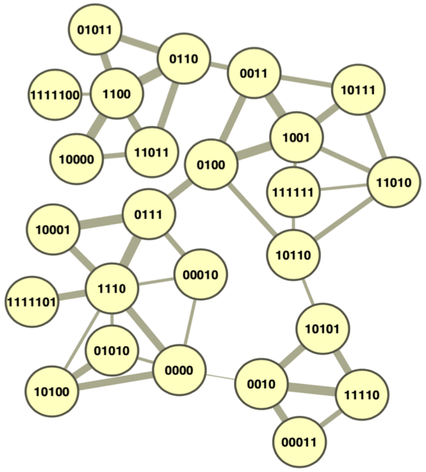
transition probability matrix $\boldsymbol{1}^T\boldsymbol{T} = \boldsymbol{1}^T$

PageRank centrality vector

Markov chain: $\boldsymbol{p}_{t+1} = \boldsymbol{T}\,\boldsymbol{p}_t\,, \qquad \boldsymbol{p} = \boldsymbol{p}_\infty$

PageRank equation:

$$\boxed{\begin{aligned}\boldsymbol{p} &= \boldsymbol{T}\,\boldsymbol{p}\\ &= c\,\boldsymbol{M}\,\boldsymbol{p} + (1\text{-}c)\,\boldsymbol{1}/N\end{aligned}}$$



$\boldsymbol{p}$ is a stochastic vector whose entry p_i is the probability of ending in node *i* → node view

PageRank for communities

extending the idea under a community assignment

node view

$$\boldsymbol{p}_n = \boldsymbol{T}_{n|n}\, \boldsymbol{p}_n$$

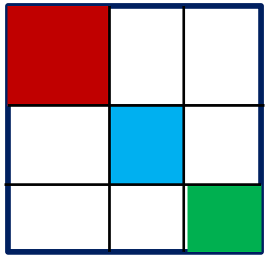
← probability (PageRank) vector at steady state

↑ transition probability matrix $\boldsymbol{1}^T \boldsymbol{T}_{n|n} = \boldsymbol{1}^T$

$$\boldsymbol{P}_{nn} = \boldsymbol{T}_{n|n}\, \mathrm{diag}(\boldsymbol{p}_n)$$

← joint probability matrix at steady state $\boldsymbol{P}_{\mathrm{nn}} \boldsymbol{1} = \boldsymbol{p}_{\mathrm{n}}$, $\boldsymbol{1}^{\mathrm{T}} \boldsymbol{P}_{nn} = \boldsymbol{p}_n^{\mathrm{T}}$

community view



$$\boldsymbol{P}_{cc} = \boldsymbol{C}\, \boldsymbol{P}_{nn}\, \boldsymbol{C}^{\mathrm{T}}$$

← joint probability matrix at steady state $\boldsymbol{P}_{\mathrm{cc}} \boldsymbol{1} = \boldsymbol{p}_{\mathrm{c}}$, $\boldsymbol{1}^{\mathrm{T}} \boldsymbol{P}_{cc} = \boldsymbol{p}_c^{\mathrm{T}}$

$$\boldsymbol{p}_c = \boldsymbol{P}_{cc}\, \boldsymbol{1} = \boldsymbol{C}\, \boldsymbol{p}_n$$

← probability vector at steady state $\boldsymbol{p}_{\mathrm{c}} = \boldsymbol{T}_{\mathrm{c|c}}\, \boldsymbol{p}_{\mathrm{c}}$

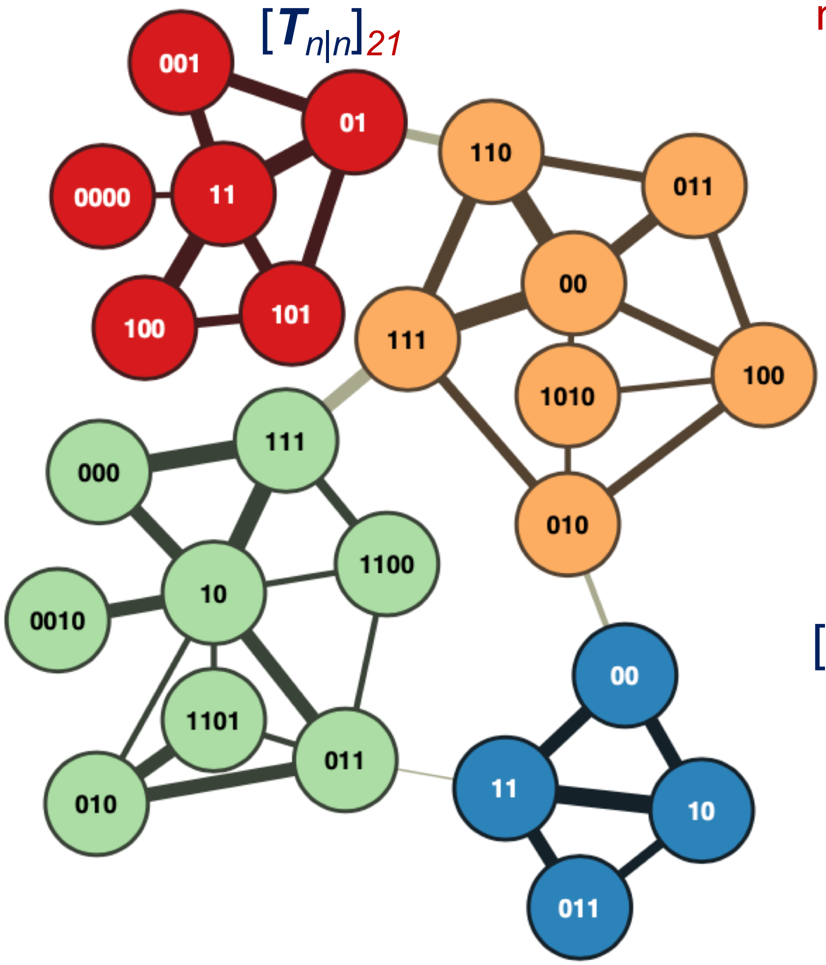
$$\boldsymbol{T}_{c|c} = \boldsymbol{P}_{cc}\, \mathrm{diag}(\boldsymbol{p}_c)^{-1}$$

← transition probability matrix $\boldsymbol{1}^T \boldsymbol{T}_{c|c} = \boldsymbol{1}^T$

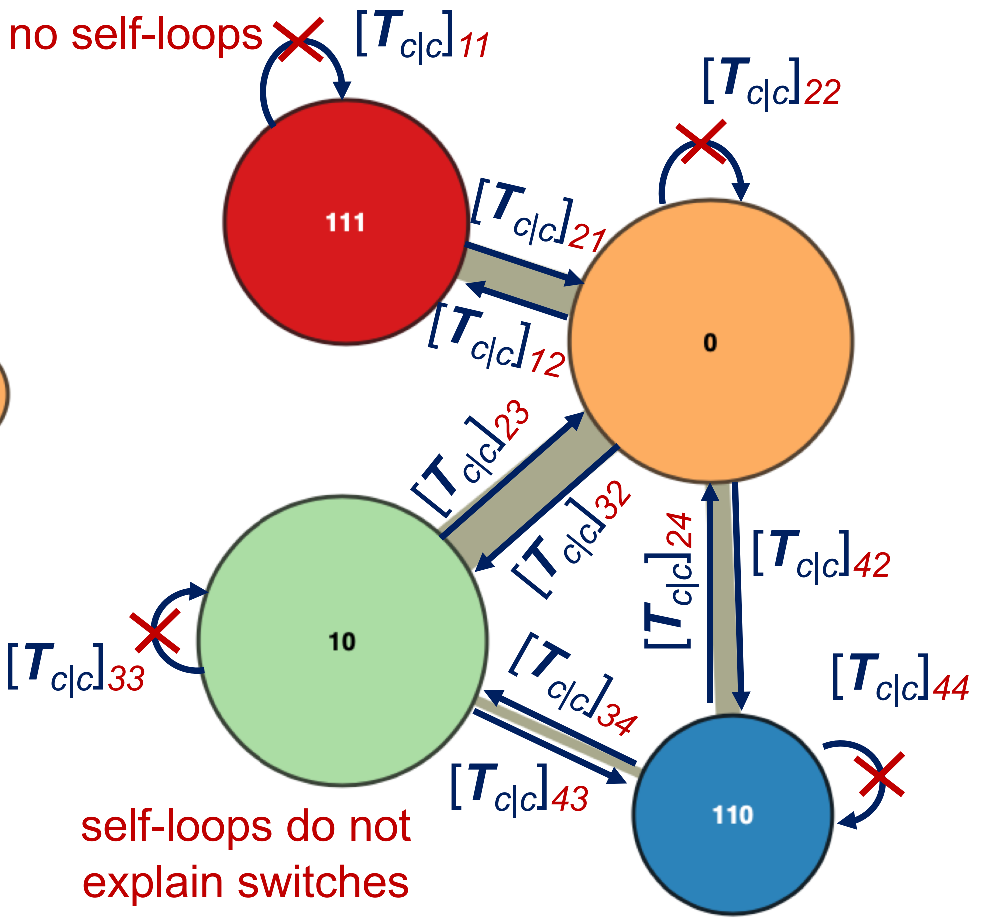
The community view

generating the entry codes

node view



community view



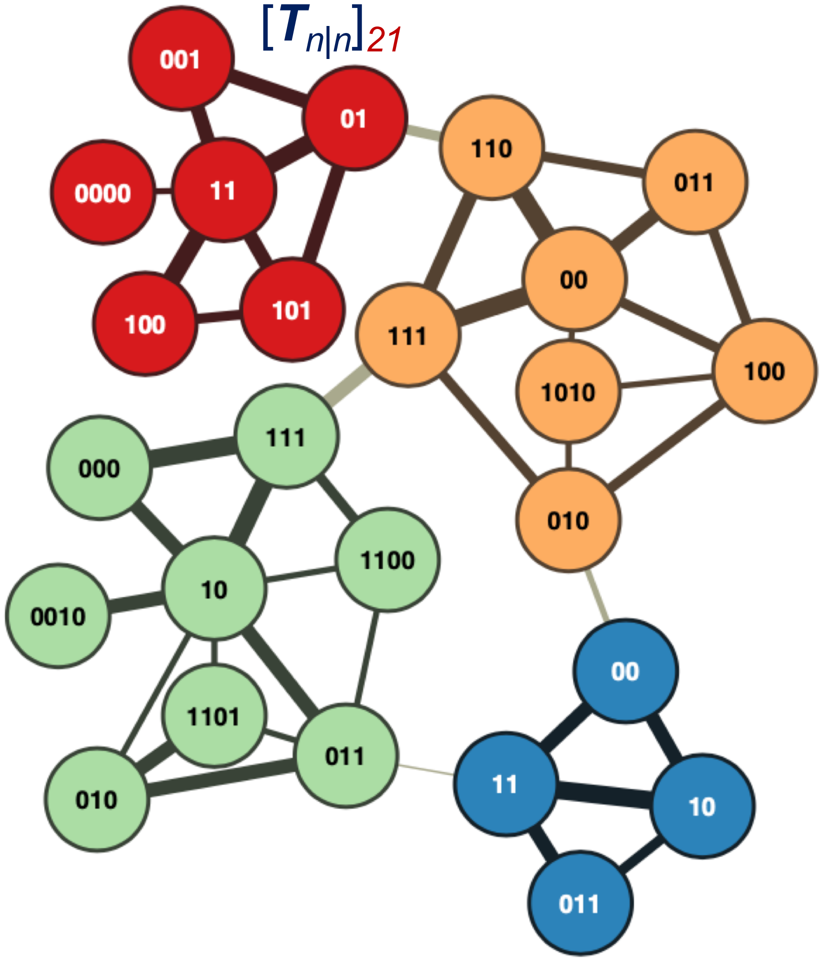
no self-loops

self-loops do not explain switches between communities

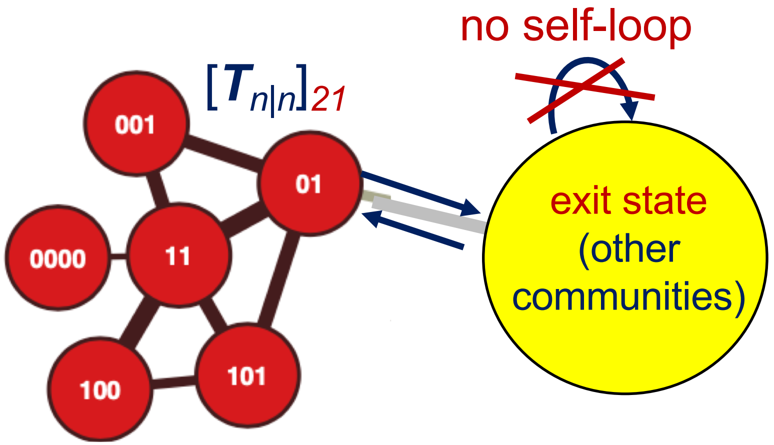
View inside a community

generating nodes codes and exit codes

node view



inside-the-community view



all nodes internal to the community are kept separate

$$C_i = \begin{pmatrix} I & 0 \\ 0^T & 1^T \end{pmatrix}$$

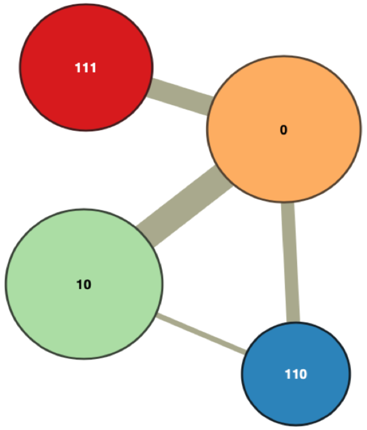
community i

all nodes external to the community are put in a single entity (exit state)

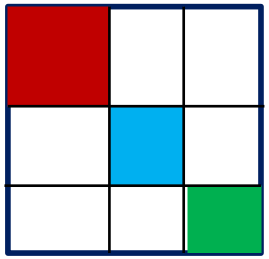
UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Steady state probabilities

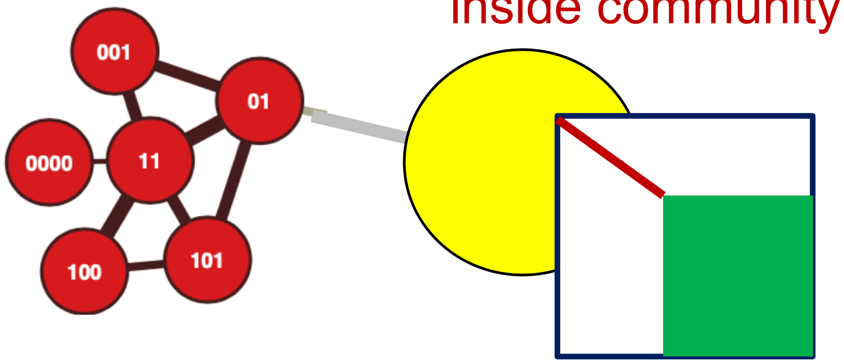
for community switch and inside communities



community switch



inside community *i*



$$\boldsymbol{P} = \boldsymbol{C}\,\boldsymbol{P}_{nn}\,\boldsymbol{C}^{\mathrm{T}} \qquad \boldsymbol{p} = \boldsymbol{P}\,\boldsymbol{1}$$

$$\boldsymbol{T} = \boldsymbol{P}\,\mathrm{diag}(\boldsymbol{p})^{-1}$$

switching probability matrix $\boldsymbol{1}^T\boldsymbol{S} = \boldsymbol{1}^T$

$$\boldsymbol{S} = [\boldsymbol{T} - \mathrm{diag}(\boldsymbol{T})]\,[\boldsymbol{I} - \mathrm{diag}(\boldsymbol{T})]^{-1}$$

eigenvector

$$\boldsymbol{q} = [\boldsymbol{I} - \mathrm{diag}(\boldsymbol{T})]\,\boldsymbol{p} = \boldsymbol{p} - \mathrm{vdiag}(\boldsymbol{P})$$

normalized

$$\boxed{\begin{aligned} \boldsymbol{s}_c &= \boldsymbol{q} / P_s \\ P_s &= \boldsymbol{1}^{\mathrm{T}}\boldsymbol{q} \end{aligned}}$$

$$\boldsymbol{P}_i = \boldsymbol{C}_i\,\boldsymbol{P}_{nn}\,\boldsymbol{C}_i^{\mathrm{T}} \qquad \boldsymbol{p}_i = \boldsymbol{P}_i\,\boldsymbol{1}$$

$$\boldsymbol{T}_i = \boldsymbol{P}_i\,\mathrm{diag}(\boldsymbol{p}_i)^{-1}$$

keeps only the bottom right element

$$\boldsymbol{S}_i = [\boldsymbol{T}_i - \mathrm{lowel}(\boldsymbol{T}_i)]\,[\boldsymbol{I} - \mathrm{lowel}(\boldsymbol{T}_i)]^{-1}$$

$$\boldsymbol{z}_i = [\boldsymbol{I} - \mathrm{lowel}(\boldsymbol{T}_i)]\,\boldsymbol{p}_i = \boldsymbol{p}_i - \mathrm{vlowel}(\boldsymbol{P}_i)$$

last element is q_i

$$\boxed{\begin{aligned} \boldsymbol{u}_i &= \boldsymbol{z}_i / P_i \\ P_i &= \boldsymbol{1}^{\mathrm{T}}\boldsymbol{z}_i \end{aligned}}$$

Wrap-up on InfoMap

works also with overlapping communities

adjacency matrix (can be fractional)

transition probability matrix

$$\boldsymbol{M} = \boldsymbol{A}\,\mathrm{diag}^{-1}(\boldsymbol{d}), \qquad \boldsymbol{d} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{1}$$

PageRank vector

$$\boldsymbol{r} = c\,\boldsymbol{M}\,\boldsymbol{r} + (1-c)\,\boldsymbol{1}/N$$

node domain

---

communities domain

$$\boldsymbol{z}_i = \boldsymbol{c}_i\,\mathrm{diag}(\boldsymbol{r})$$

Here $\boldsymbol{c}_i$ is the ith row of $\boldsymbol{C}$

q vector entries

$$q_i = \left(1 - (1-c)\frac{\boldsymbol{c}_i\boldsymbol{1}}{N}\right)\boldsymbol{z}_i\boldsymbol{1} - c\,\boldsymbol{c}_i\,\boldsymbol{M}\,\boldsymbol{z}_i^T$$

InfoMap

$$\boxed{f(\boldsymbol{q}) + \sum_{\boldsymbol{i}} f([q_i, \boldsymbol{z}_i])}$$

entropy function

$$f(\boldsymbol{x}) = -\sum\nolimits_j x_j \log\left(\frac{x_j}{\sum_j x_j}\right)$$

The InfoMap algorithm

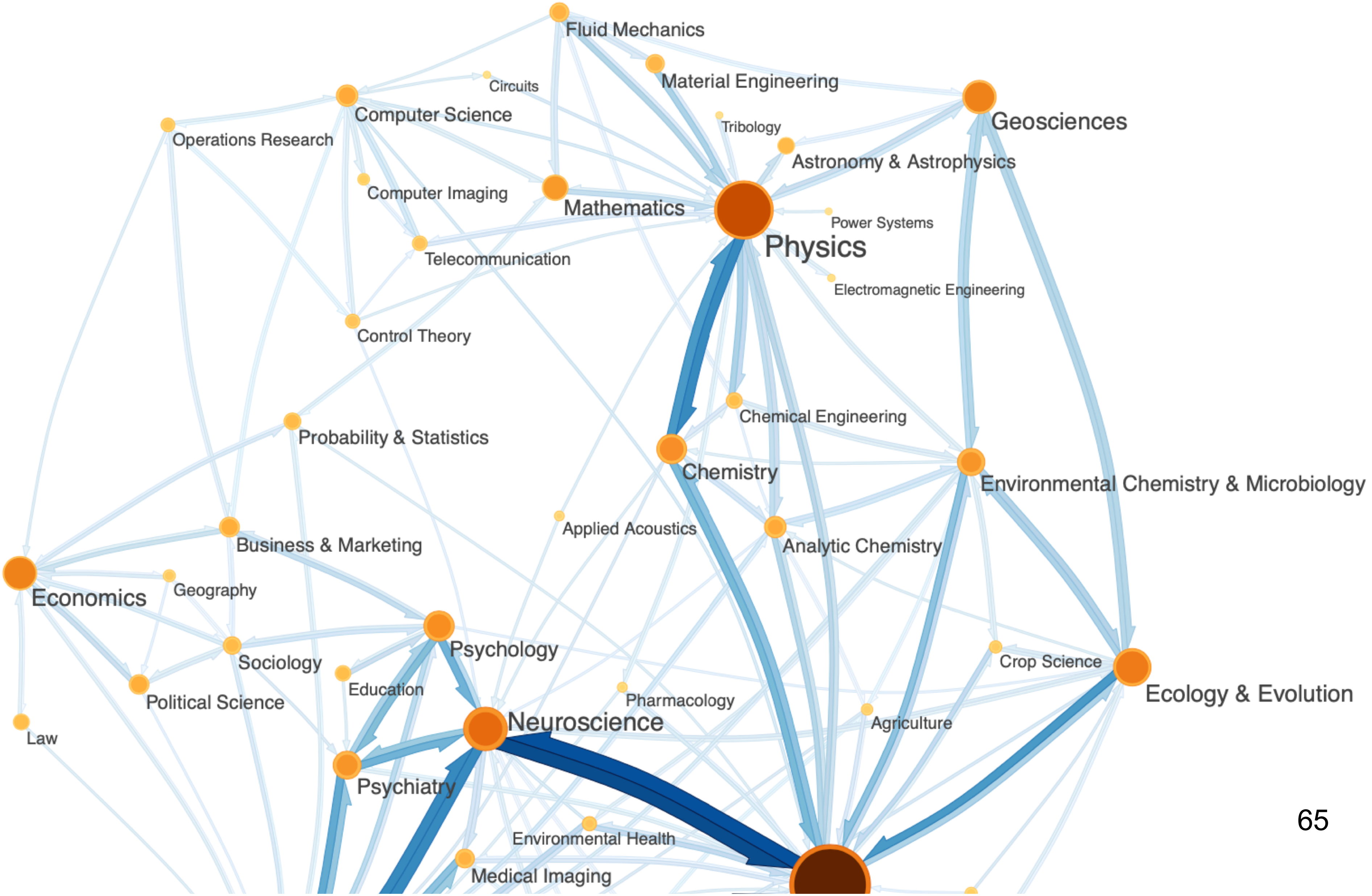
an iterative procedure

- ☐ Assign every node to a community
- ☐ Merge the two communities that provide the best improvement in the InfoMap measure, until convergence
- ☐ Refine of the result by simulated annealing, moving one node per time

Not strikingly different from Louvain

Application example

map of science based on citation patterns as in 2008



Takeaways

for InfoMap

- InfoMap is an alternative quality metric to modularity
- It is especially useful when the information available explains flows in the network
- Its fairly easy to calculate, which makes it a scalable approach
- Code for the standard approach is available on the web but only for non-overlapping communities

Normalized mutual information

a measure based on statistics

Venn diagram

on the statistical dependence among two discrete random variables

joint probability matrix $\boldsymbol{P}_{XY}$

$\boldsymbol{p}_X = \boldsymbol{P}_{XY}\,\boldsymbol{1}$ projection on X

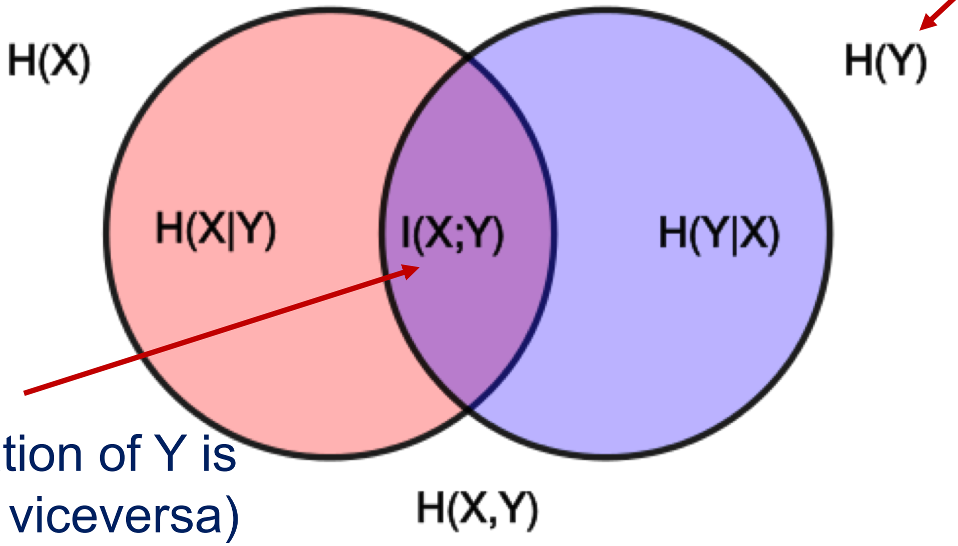
$\boldsymbol{p}_Y = \boldsymbol{P}_{XY}^{\mathrm{T}}\,\boldsymbol{1}$ projection on Y

$$\mathrm{H}(X) = \sum_x p_x \log(1/p_x)$$

$$\mathrm{H}(Y) = \sum_y p_y \log(1/p_y)$$

entropy of X
information
carried by X

entropy of Y
information
carried by Y



mutual information
how much information of Y is
explained by X (or viceversa)

$$\mathrm{I}(X;Y) = \sum_{x,y} P_{xy} \log(P_{xy}/pxpy)$$

joint entropy $\mathrm{H}(X,Y) = \sum_{x,y} P_{xy} \log(1/Pxy)$

Normalized mutual information

in unsupervised community detection

We assume a true joint probability description $\boldsymbol{P}_{nn}$ is available, e.g., a normalized adjacency $\boldsymbol{A}$

$\boldsymbol{C}$ community assignment to be assessed for quality

statistical dependencies about being in a community and ending in another

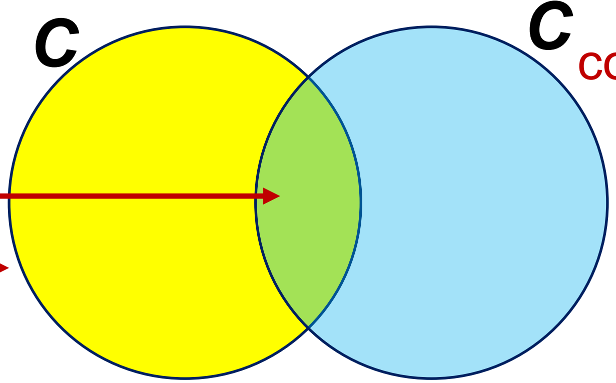
$$\boldsymbol{P}_{\mathrm{C,C}} = \boldsymbol{C}\,\boldsymbol{A}\,\boldsymbol{C}^{\mathrm{T}} \longrightarrow \boldsymbol{p}_{\mathrm{C}} = \boldsymbol{P}_{\mathrm{C,C}}\,\boldsymbol{1}$$

probability of ending in a community

fraction of knowledge related to the community we will end up in (between 0 and 1, equal to 1 for statistically independent communities)

$$\mathrm{NMI}(\mathrm{C}) = \frac{\mathrm{I}(\mathrm{C};\mathrm{C})}{\mathrm{H}(\mathrm{C})}$$

can also use H(C,C), but its interpretation is weaker



Wrap-up

on metrics for community detection

Takeaways

on quality measures – unsupervised community detection

quality measure	approach	undirected	directed	overlapping	signed
Modularity	number of links inside communities, compares to a random model	YES	YES	YES	YES
Conductance, Ncut	number of links outside communities divided by total links of the community	YES	NO	YES	NO
Normalized mutual information	fraction based on entropies and mutual information	YES	YES	YES	NO
InfoMap	average encoding length under a PageRank information flow	YES	YES	YES	NO

would be nice to see these in your projects

BigCLAM

a simple statistical inference model for community detection

The statistical inference approach

statistically modeling a graph

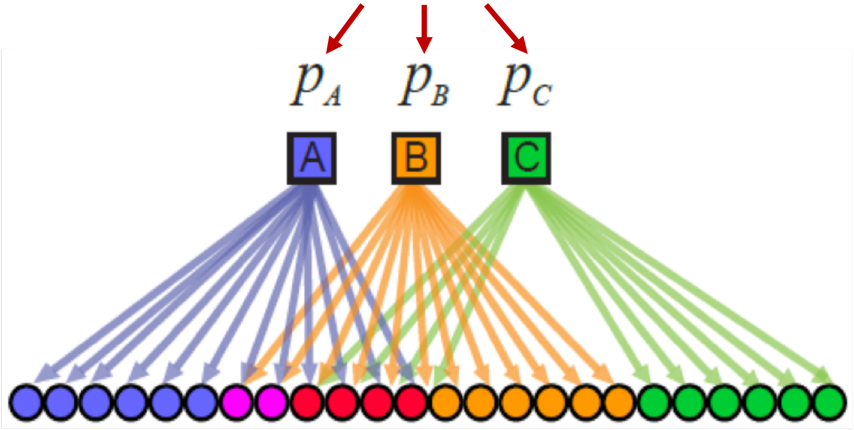
- ☐ Let $p(\boldsymbol{A}|\gamma)$ be a probabilistic model describing a network (i.e., its adjacency matrix $\boldsymbol{A}$) through some parameters γ
- ☐ The parameters γ are assumed to capture relevant information about the network, e.g., its community structure
- ☐ An a priori statistical description $p(\gamma)$ of the parameters can also be available, in case it is not simply set $p(\gamma)=1$ (i.e., consider equally likely parameters)
- ☐ Since, for a given network, $\boldsymbol{A}$ is known, the optimal parameters fit is found by the maximum a posteriori (MAP) principle

$$\boxed{\hat{\gamma} = \operatorname{argmax}_{\gamma}\, p(\boldsymbol{A}|\gamma)\, p(\gamma)}$$

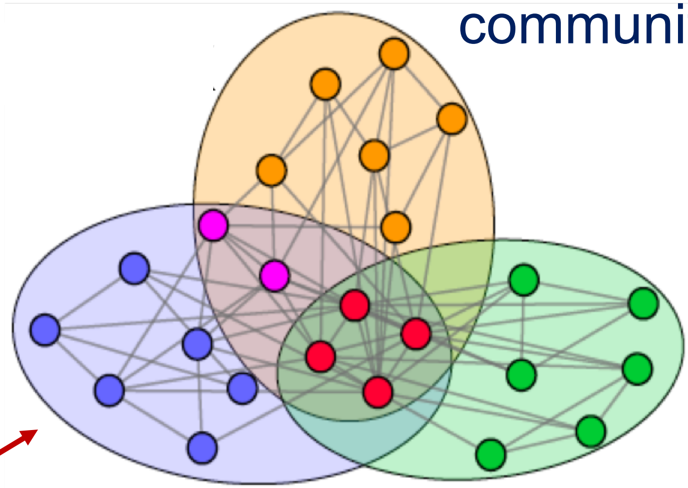
The BigCLAM statistical model

for binary adjacency matrices ***A***

communities are described through probabilities p_c that express the probability that a link between two nodes (inside the community) is active, these are collected in vector ***p***



we assume overlapping communities

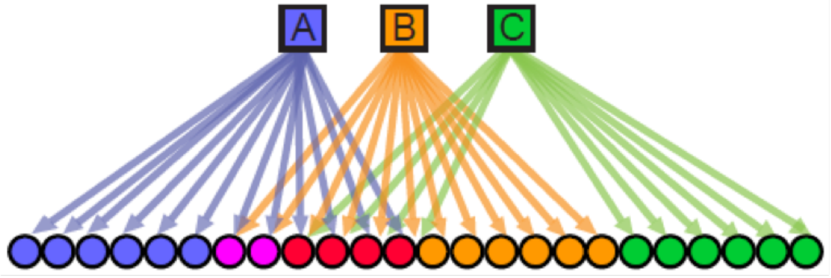


the map from nodes to communities is collected in a *C* x *N* membership matrix ***C*** whose *i*th column $\boldsymbol{c}_i$ is a binary vector identifying the communities to which node *i* belongs

$\boldsymbol{c}_i$ = [0 1 1 0 1] tells that node *i* belongs to communities 2, 3, and 5

The BigCLAM statistical model

probability of not activating an edge



Probability Q_{ij} that edge (i,j) is not active is the probability that it is not active in any of the communities linking i and j, that is

communities in common between i and j

$$Q_{ij} = \prod_{c \in M_i \cap M_j} (1 - p_c)$$

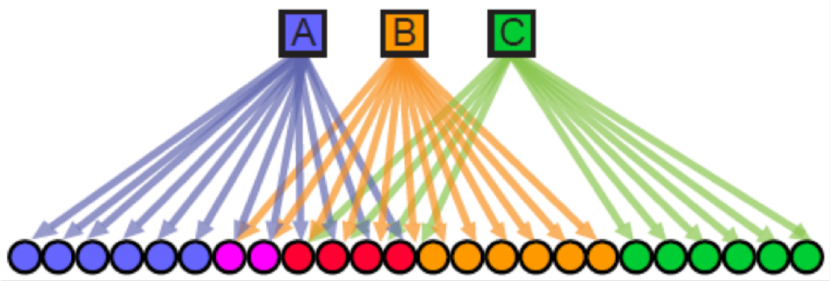
$$\log(Q_{ij}) = \sum_{c \in M_i \cap M_j} \log(1 - p_c) = \boldsymbol{c}_i^T \,\mathrm{diag}\, \log(\boldsymbol{1} - \boldsymbol{p})\, \boldsymbol{c}_j$$

Here $\boldsymbol{c}_i$ is the ith column of $\boldsymbol{C}$

$$\boxed{\boldsymbol{Q} = \exp(-\boldsymbol{C}^T \,\mathrm{diag}(\boldsymbol{q})\, \boldsymbol{C})}\;, \quad \boldsymbol{q} = -\log(\boldsymbol{1} - \boldsymbol{p}) > \boldsymbol{0}$$

The BigCLAM statistical model

Yang & Leskovec, Overlapping community detection at scale:
a nonnegative matrix factorization approach, (2013)



The graph probability description $p(\boldsymbol{A}|\boldsymbol{C},\boldsymbol{q})$ therefore is

$$p(\boldsymbol{A}|\boldsymbol{Q}) = \prod_{(i,j)\in\mathcal{E}}(1-Q_{ij}) \prod_{(i,j)\notin\mathcal{E}} Q_{ij}$$

edge set $\mathcal{E} = \{(i,j) \mid a_{ij} = 1\}$

maximize $p(\boldsymbol{A}|\boldsymbol{Q})$

wrt $\boldsymbol{C}$, $\boldsymbol{q}$

s.to $\boldsymbol{Q} = \exp(-\boldsymbol{C}^T \operatorname{diag}(\boldsymbol{q})\, \boldsymbol{C})$

$\boldsymbol{C}$ binary, $\boldsymbol{q} > \boldsymbol{0}$

NP complex
reference
optimization problem

need to set the number C of communities, $\boldsymbol{A}$ is binary

Model relaxation

a relaxed counterpart to the optimization problem

maximize $p(\boldsymbol{A}|\boldsymbol{C},\boldsymbol{q})$
wrt $\boldsymbol{C}$, $\boldsymbol{q}$
s.to $\boldsymbol{Q} = \exp(-\boldsymbol{C}^T \operatorname{diag}(\boldsymbol{q})\, \boldsymbol{C})$
$\boldsymbol{C}$ ~~binary~~, $\boldsymbol{q} > 0$

we relax the binary constraint
(and include $\boldsymbol{q}$ into $\boldsymbol{C}$)

⇒

maximize $\log p(\boldsymbol{A}|\boldsymbol{M})$
wrt $\boldsymbol{M}$
s.to $\boldsymbol{Q} = \exp(-\boldsymbol{M}^T \boldsymbol{M})$
$\boldsymbol{M} > 0$

we obtain an overlapping community
assignment by normalizing
$\boldsymbol{M} = \operatorname{sqrt}(\operatorname{diag}(\boldsymbol{q}))\, \boldsymbol{C}$ by column

$$\log p(\boldsymbol{A}|\boldsymbol{M}) = \sum_{(i,j)\in\mathcal{E}} \log(1-Q_{ij}) + \sum_{(i,j)\notin\mathcal{E}} \log(Q_{ij})$$

$\log(Q_{ij}) = -\boldsymbol{m}_i^T \boldsymbol{m}_j$

$$= \sum_{(i,j)\in\mathcal{E}} \log\left(\frac{1-Q_{ij}}{Q_{ij}}\right) + \sum_{i,j} \log(Q_{ij})$$

we add some
$\log(Q_{ij})$ here

$$= -\sum_{(i,j)\in\mathcal{E}} g(\boldsymbol{m}_i^T \boldsymbol{m}_j) - \sum_{i,j} \boldsymbol{m}_i^T \boldsymbol{m}_j$$

$g(x) = -\log(e^{x}-1)$

Here $\boldsymbol{m}_i$
is the ith
column
of $\boldsymbol{M}$

The BigCLAM algorithm

a gradient descent search for the optimum

$$\min \sum_{i,j\in N_i} g(\boldsymbol{m}_i^T\boldsymbol{m}_j) + \sum_{i,j} \boldsymbol{m}_i^T\boldsymbol{m}_j$$

wrt $\boldsymbol{m}_i > \boldsymbol{0}$

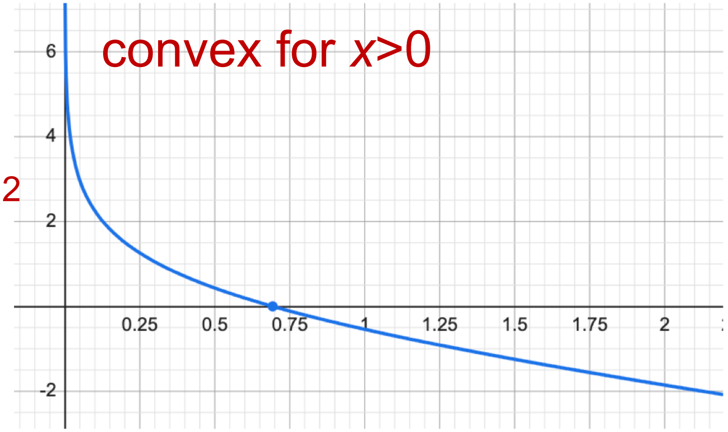
convex problem with
linear constraints

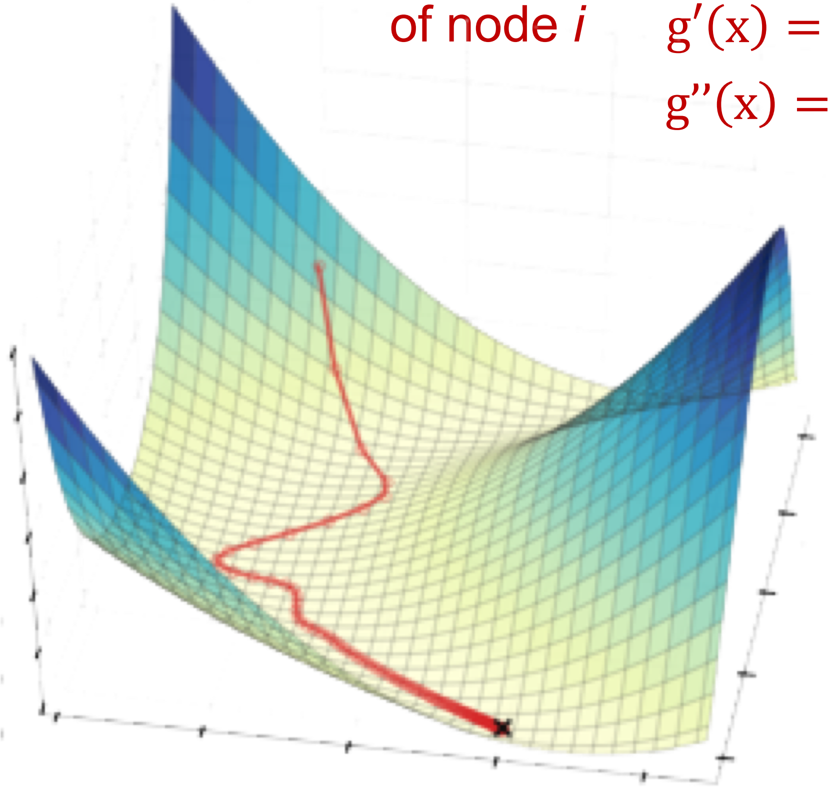
neighbours
of node i

$g(x) = -\log(e^{x\,-}\,1)$

$g'(x) = -1/(1 - e^{-x})$

$g''(x) = e^{-x}/(1 - e^{-x})^2$


convex for x>0



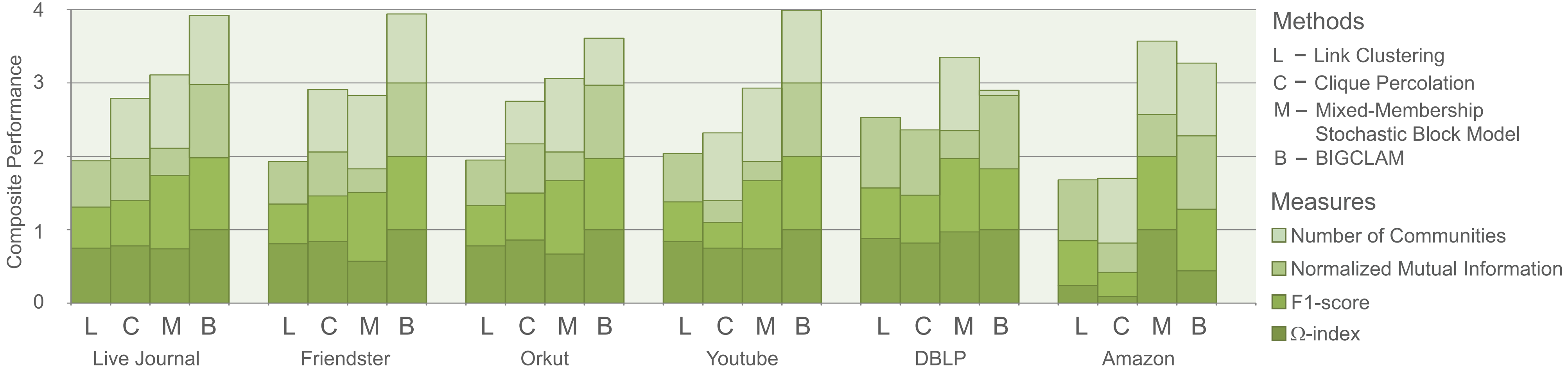
can be solved using
gradient descent methods
and standard algorithms

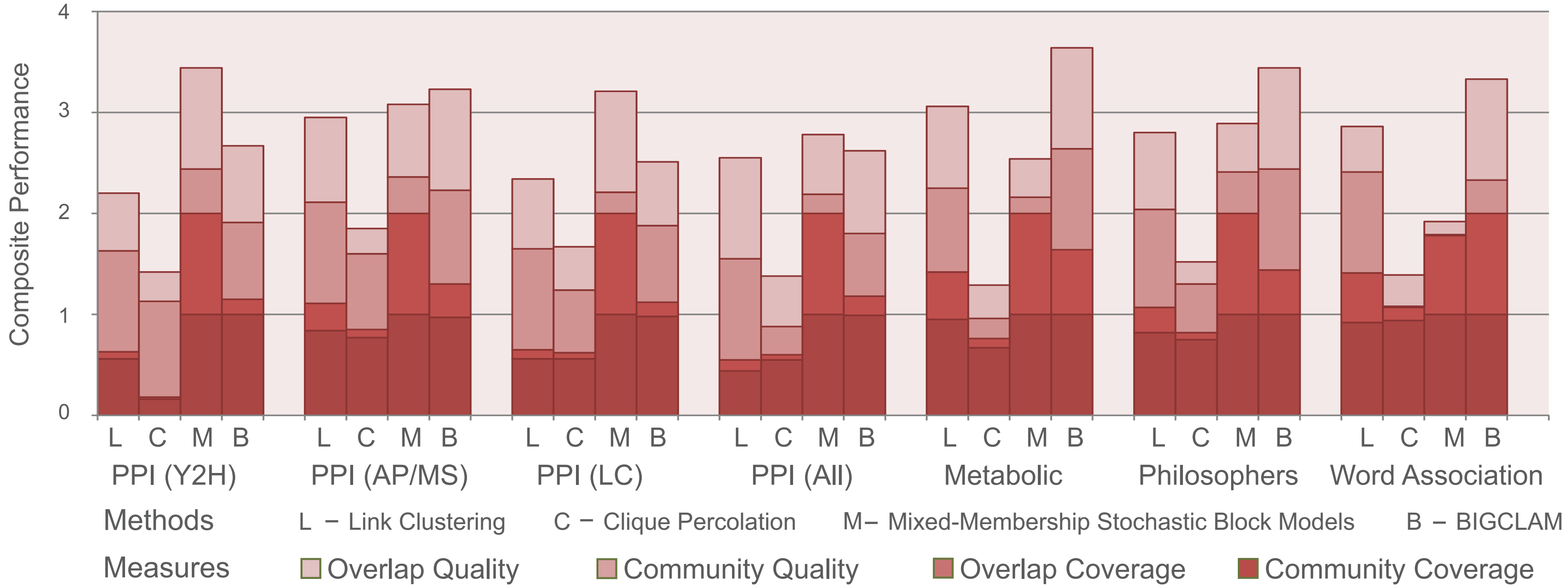
$$\nabla_{\mathbf{m}i} = \sum_{j\in N_i} 2\, g'(\boldsymbol{m}_i^T \boldsymbol{m}_j)\, \boldsymbol{m}_j + \sum_j 2\; \boldsymbol{m}_j$$

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

BigCLAM quality performance

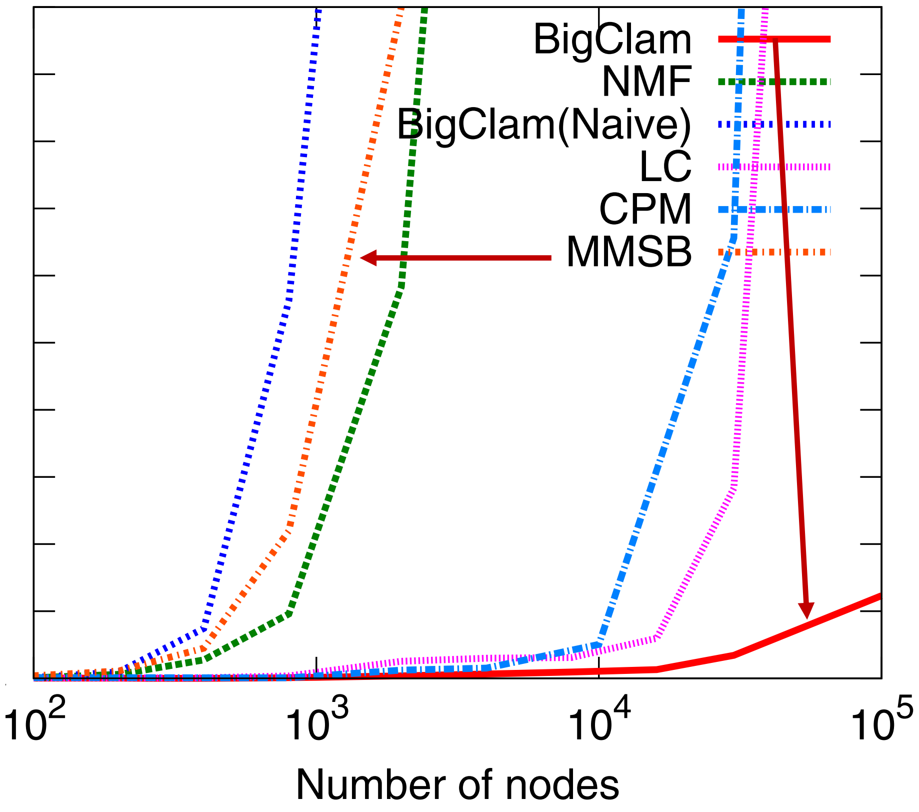
compared to state-of-the-art algorithms at the time

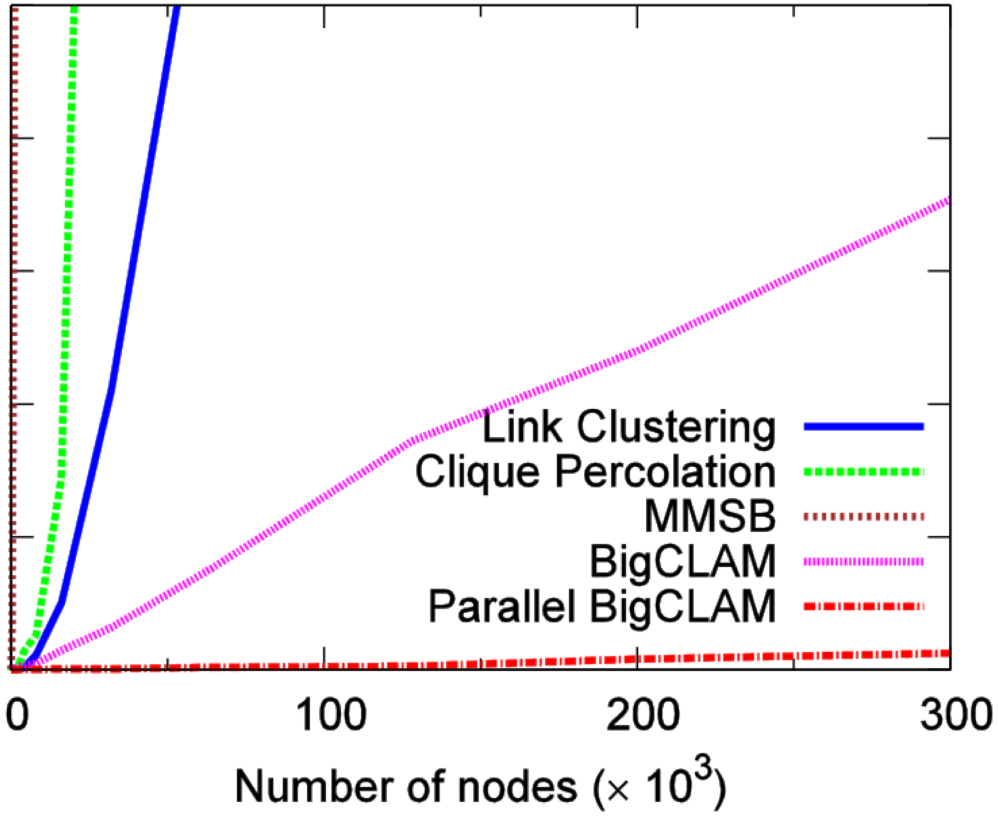




BigCLAM complexity

compared to state-of-the-art algorithms at the time





Takeaways

for BigCLAM

- A simple statistical inference model to explain the concept
- A proof of concept
- Highly scalable
- Applicable to binary symmetric adjacency matrices only
- Literature shows its performance may be not striking with synthetic networks
- Would be interesting to see it implemented in your projects ☺

Stochastic Block Models

SBM for community detection

Stochastic block model - SBM

for a binary adjacency matrix

- probability block matrix ***B***
- ***B*** is not stochastic, simply $0 \leq \mathbf{B} \leq 1$
- B_{ab} expresses the probability that a node in community *a* links to a node in community *b*
- community indicator vector ***c***
- c_i expresses the community of node *i*
- edges are Bernoulli distributed (conditioned on their group memberships) with probability $B_{c_i c_j}$

i < *j* for undirected networks

Stochastic model: $p(\mathbf{A}|\mathbf{B},\mathbf{c}) = \prod_{i,j} \left(B_{c_i c_j}\right)^{a_{ij}} \left(1 - B_{c_i c_j}\right)^{1-a_{ij}}$

binary adjacency matrix

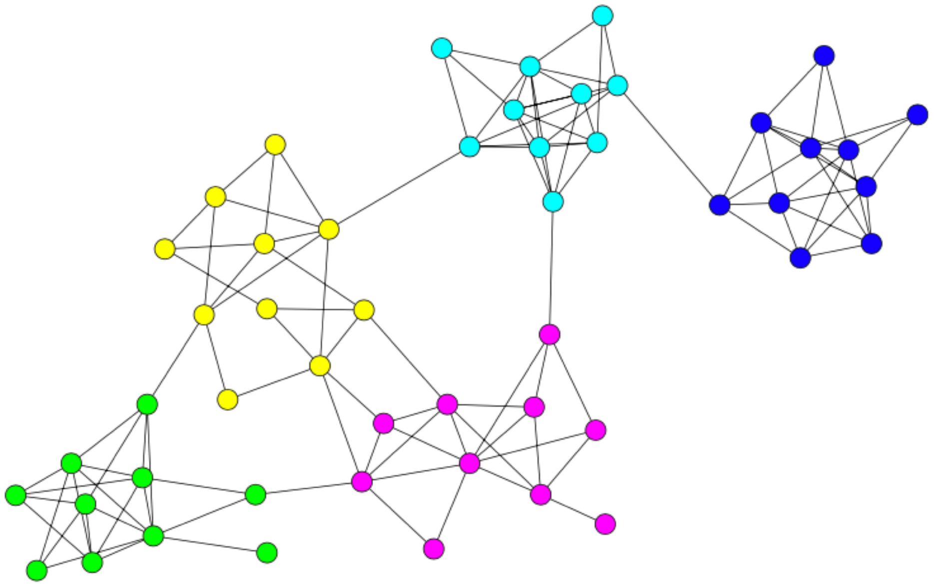
can also be expressed in terms of the community assignment matrix ***C***

$B_{c_i c_j} = [\mathbf{C}^{\mathsf{T}} \mathbf{B} \mathbf{C}]_{ij} = \mathbf{c}_i^{\mathsf{T}} \mathbf{B} \mathbf{c}_j$

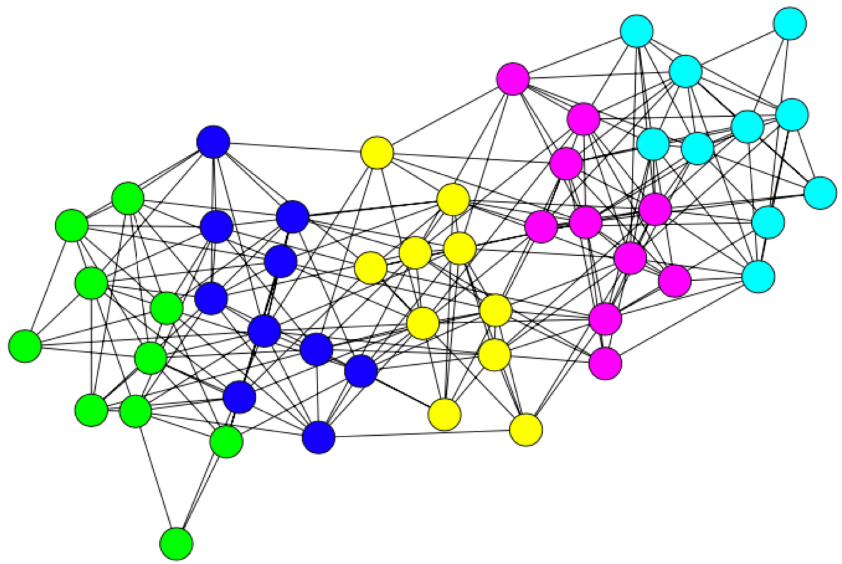
Here $\mathbf{c}_i$ is the *i*th column of ***C***

SBM examples

assortative and ordered communities case



assortative communities



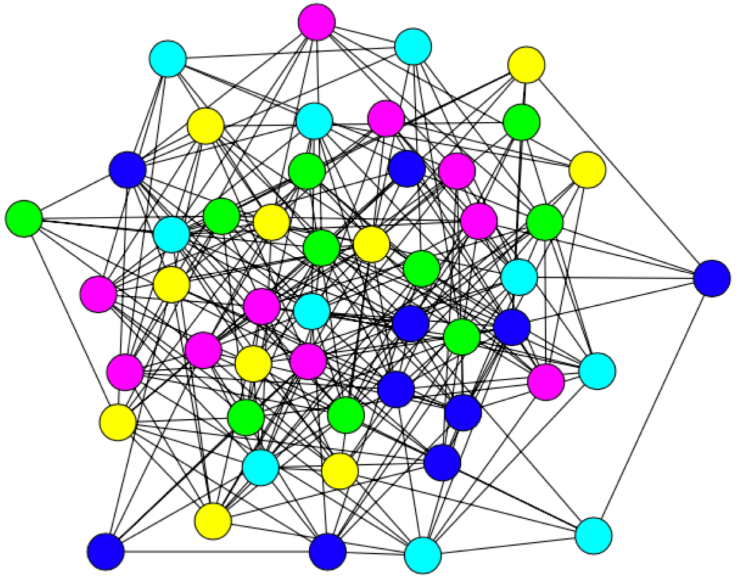
ordered communities

$$\boldsymbol{B} = \begin{array}{ccccc} .5 & .1 & .1 & .1 & .1 \\ .1 & .5 & .1 & .1 & .1 \\ .1 & .1 & .5 & .1 & .1 \\ .1 & .1 & .1 & .5 & .1 \\ .1 & .1 & .1 & .1 & .5 \end{array}$$

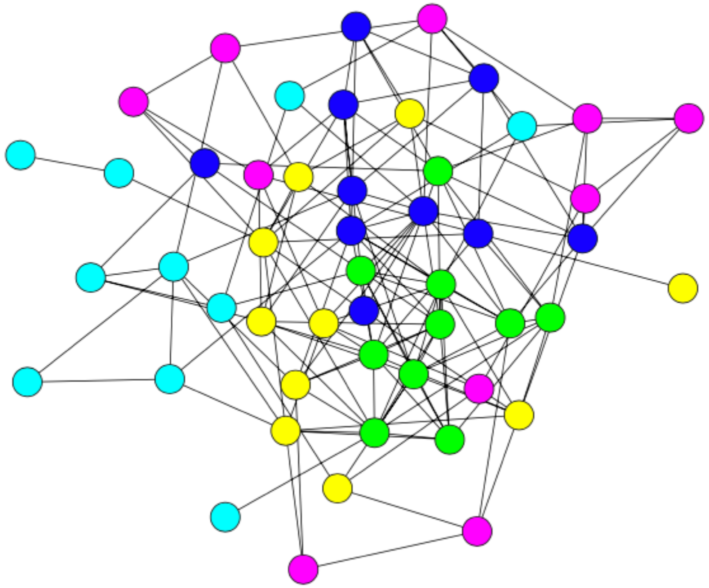
$$\begin{array}{ccccc} .5 & .3 & & & \\ .3 & .5 & .3 & & \\ & .3 & .5 & .3 & \\ & & .3 & .5 & .3 \\ & & & .3 & .5 \end{array}$$

SBM examples

random and core-periphery communities case



random graph



core-periphery structure

$$B = \begin{array}{ccccc} .2 & .2 & .2 & .2 & .2 \\ .2 & .2 & .2 & .2 & .2 \\ .2 & .2 & .2 & .2 & .2 \\ .2 & .2 & .2 & .2 & .2 \\ .2 & .2 & .2 & .2 & .2 \end{array}$$

$$\begin{array}{ccccc} \mathbf{.7} & .24 & .14 & .09 & .05 \\ .24 & \mathbf{.42} & .14 & .09 & .05 \\ .14 & .14 & \mathbf{.25} & .09 & .05 \\ .09 & .09 & .09 & \mathbf{.15} & .05 \\ .05 & .05 & .05 & .05 & \mathbf{.09} \end{array}$$

Some remarks
on the SBM model

- SBM can naturally handle directed and undirected networks
- for undirected networks we force ***B*** to be symmetric
- we relax this assumption for directed networks: in this way the probability of a link running in one direction is different of the probability that a link runs in the opposite direction
- SBM can also naturally handle overlapping communities, as $\boldsymbol{C}^{\mathrm{T}}\,\boldsymbol{B}\,\boldsymbol{C}$ makes sense also in this case
- closely related to BigCLAM

$$\log p(\boldsymbol{A}|\boldsymbol{B},\boldsymbol{c}) = \sum_{i,j} a_{ij} \log(1 - Qij) + (1 - a_{ij}) \log(Q_{ij})$$

$$Q_{ij} = 1 - \boldsymbol{c}_i^{\mathrm{T}}\boldsymbol{B}\,\boldsymbol{c}_j \cong \exp(-\boldsymbol{c}_i^{\mathrm{T}}\boldsymbol{B}\,\boldsymbol{cj})$$

SBM assumption

BigCLAM assumption, with diagonal ***B***

Compact form

for binary ***A*** and non-overlapping communities

log $p(\boldsymbol{A}|\boldsymbol{B},\boldsymbol{C})$ = $\sum_{i,j} a_{ij} \log\left(B_{c_i c_j}\right) + (1 - a_{ij}) \log\left(1 - B_{c_i c_j}\right)$

$$\max \sum_{u,v} m_{uv} \log(B_{uv}) + (n_{uv} - m_{uv}) \log(1 - B_{uv})$$

s. to $\boldsymbol{B} \geq 0$, $\boldsymbol{M} = \boldsymbol{C}\,\boldsymbol{A}\,\boldsymbol{C}^{\mathsf{T}}$, $\boldsymbol{N} = \boldsymbol{C}\,\boldsymbol{1}\,\boldsymbol{1}^{\mathsf{T}}\,\boldsymbol{C}^{\mathsf{T}}$

number of active links between communities

max number of links between communities

$\hat{B}_{uv} = m_{uv}/n_{uv}$

$$\max \sum_{u,v} m_{uv} \log(m_{uv}) + (n_{uv} - muv) \log(n_{uv} - muv)$$

$$-nuv \log(n_{uv})$$

s. to $\boldsymbol{M} = \boldsymbol{C}\,\boldsymbol{A}\,\boldsymbol{C}^{\mathsf{T}}$, $\boldsymbol{N} = \boldsymbol{C}\,\boldsymbol{1}\,\boldsymbol{1}^{\mathsf{T}}\,\boldsymbol{C}^{\mathsf{T}}$

Degree-corrected SBMs

Karrer, Newman. "Stochastic blockmodels and community structure in networks." (2011)

https://www.asc.ohio-state.edu/statistics/dmsl/Karrer_Newman_2010.pdf

node-dependent parameters added

$$\log p(\boldsymbol{A}|\boldsymbol{B},\boldsymbol{C}) = \sum_{i,j} a_{ij} \log\left(\theta_i \vartheta_j B_{c_i c_j}\right) - \theta_i \vartheta_j B_{c_i c_j}$$

binary ***A***,
non-overlapping
communities

Poisson
approximation

$$f(k;\lambda) = \begin{cases} e^{-\lambda}, & k = 0 \\ \lambda\, e^{-\lambda}, & k = 1 \\ \dfrac{\lambda^k e^{-\lambda}}{k!}, & \cancel{\text{otherwise}} \end{cases}$$

$$\max \sum_{u,v} m_{uv} \log(m_{uv}/n_{uv})$$

$$\text{s.to } \boldsymbol{M} = \boldsymbol{C}\,\boldsymbol{A}\,\boldsymbol{C}^{\mathsf{T}},\ \boldsymbol{N} = \boldsymbol{M}\,\boldsymbol{1}\,\boldsymbol{1}^{\mathsf{T}}\,\boldsymbol{M}^{\mathsf{T}}$$

this is mutual
information *I(C;C)*

this is $\boldsymbol{P}_{CC}$

this is $\boldsymbol{p}_C\, \boldsymbol{p}_C^T$

Further remarks

on the SBM model

- approximations make the problem simple
- can naturally handle undirected and weighted networks, and overlapping communities, but we are forcing its interpretation
- the degree-corrected model indentifies mutual information $I(C;C)$ as the cost measure: strongly related to NMI: strenghtens that result
- can be optimized by

  Gibbs sampling/Simulated annealing, Gradient descent, Expectation Maximization, Variational inference

Mixed membership SBM

Airoldi, et al. "Mixed membership stochastic blockmodels." (2008)
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3119541/pdf/nihms54993.pdf

$$\log p(\boldsymbol{A}|\boldsymbol{C},\boldsymbol{B}) = \sum_{i,j} a_{ij}\log(\boldsymbol{c}_i^T\,\boldsymbol{B}\,\boldsymbol{c}_j) + (1-aij)\log(1-\boldsymbol{c}_i^T\,\boldsymbol{B}\,\boldsymbol{c}_j)$$

Here $\boldsymbol{c}_i$ is the *i*th coumn of $\boldsymbol{C}$

we use a variational approach

$$log(\boldsymbol{c}_i^T\,\boldsymbol{B}\,\boldsymbol{c}_j) = log\left(\sum_{m,n}\rho_{ijmn}\,\frac{c_{im}\,Bmn\,cjn}{\rho_{ijmn}}\right) \geq \sum_{m,n}\rho_{ijmn}\,\log\left(\frac{c_{im}\,Bmn\,cjn}{\rho_{ijmn}}\right)$$

distribution function, sums up to 1

equality for $\rho_{ijmn} = \frac{c_{im}\,Bmn\,cjn}{\boldsymbol{c}_i^T\,\boldsymbol{B}\,\boldsymbol{c}_j}$

$$f(\boldsymbol{B},\boldsymbol{C},\boldsymbol{\rho},\boldsymbol{\mu}) = \sum_{i,j,m,n} a_{ij}\,\rho_{ijmn}\log\left(\frac{c_{im}\,Bmn\,cjn}{\rho_{ijmn}}\right) + (1-aij)\mu_{ijmn}\log\left(\frac{c_{im}\,(1-B_{mn})\,c_{jn}}{\mu_{ijmn}}\right)$$

$$max_{\boldsymbol{B},\boldsymbol{C}}\log p(\boldsymbol{A}|\boldsymbol{B},\boldsymbol{C}) = max_{\boldsymbol{B},\boldsymbol{C},\boldsymbol{\rho},\boldsymbol{\mu}}\, f(\boldsymbol{B},\boldsymbol{C},\boldsymbol{\rho},\boldsymbol{\mu})$$

can use an alternating minimization approach

Equations update

alternating search for the maximum

dummy distributions update

$$\rho_{ijmn} = \frac{c_{im}\, Bmn\; cjn}{\boldsymbol{c}_i^T\, \boldsymbol{B}\; \boldsymbol{c}_j}$$

$$\mu_{ijmn} = \frac{c_{im}c_{jn} - cim\, B_{mn}\, c_{jn}}{1 - \boldsymbol{c}_i^T\, \boldsymbol{B}\; \boldsymbol{c}_j}$$

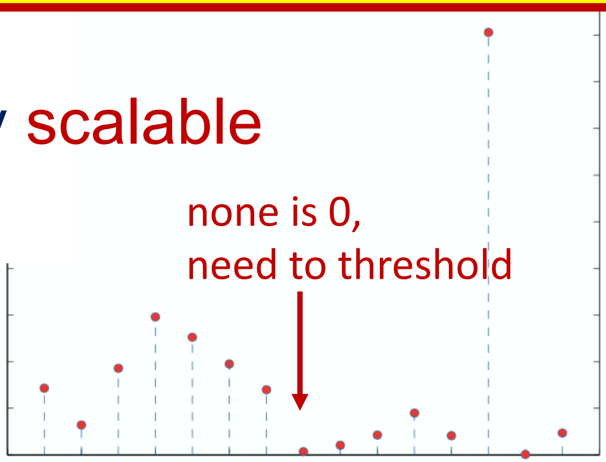
mixing matrix update

$$B_{mn} = \frac{\sum_{i,j} a_{ij}\, \rho_{ijmn}}{\sum_{i,j} a_{ij}\, \rho_{ijmn} + (1 - a_{ij})\, \mu_{ijmn}} < 1$$

community assignment update (normalized)

$$c_{im} = \frac{\sum_{j,n} a_{ij}\, \rho_{ijmn} + (1 - a_{ij})\mu_{ijmn} + a_{ji}\, \rho_{jinm} + (1 - a_{ji})\mu_{jinm}}{\sum_j 2}$$

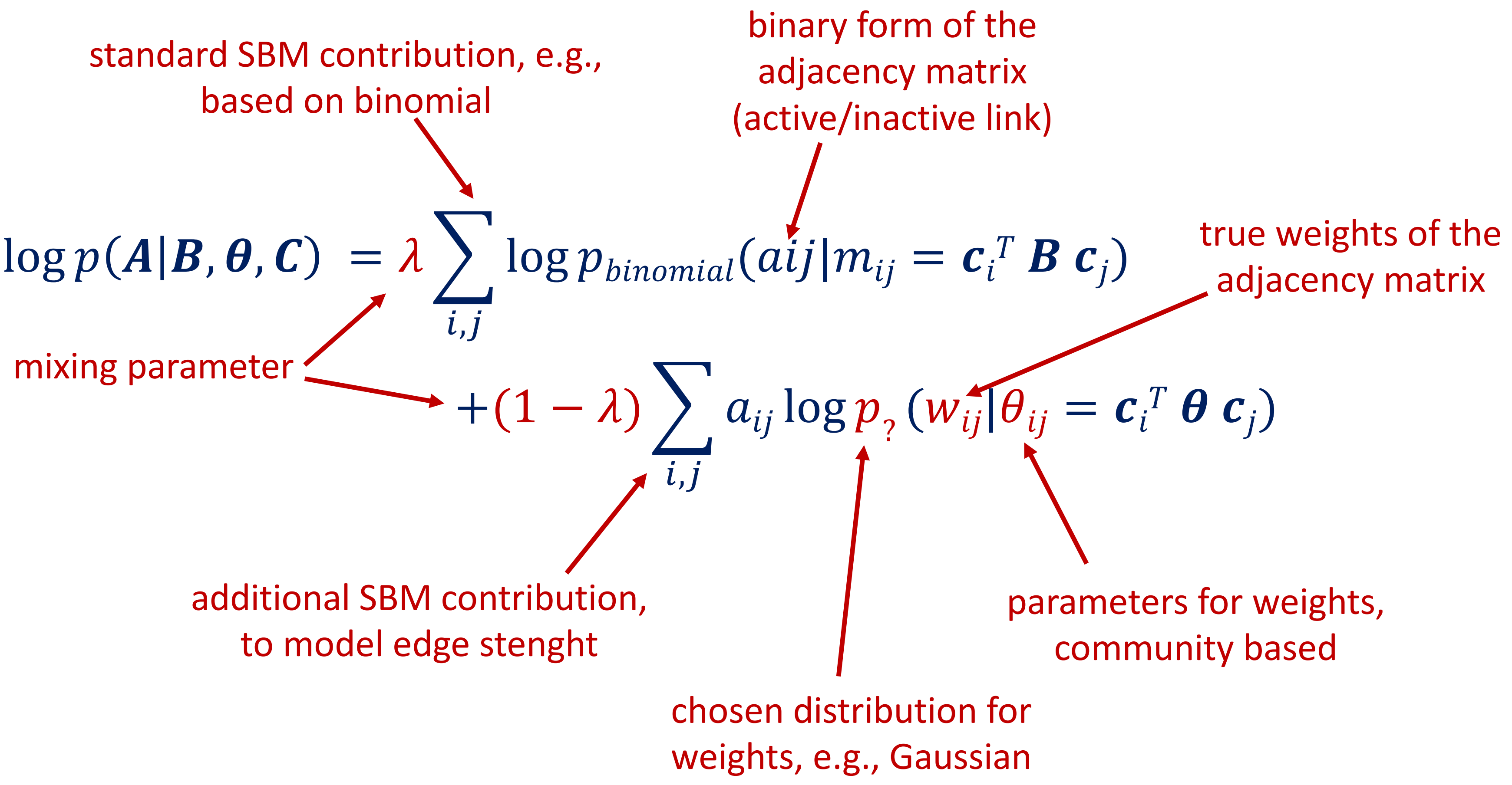
- ❑ simple algorithm, but not really scalable
- ❑ soft community assignments
- ❑ binary matrix ***A***



none is 0,
need to threshold

Weighted SBM

Christopher, Jacobs, Clauset. "Learning latent block structure in weighted networks." (2015)



standard SBM contribution, e.g., based on binomial

binary form of the adjacency matrix (active/inactive link)

$$\log p(\boldsymbol{A}|\boldsymbol{B},\boldsymbol{\theta},\boldsymbol{C}) = \lambda \sum_{i,j} \log p_{binomial}(aij|m_{ij} = \boldsymbol{c}_i{}^T\ \boldsymbol{B}\ \boldsymbol{c}_j)$$

$$+(1-\lambda) \sum_{i,j} a_{ij} \log p_{?}\,(w_{ij}|\theta_{ij} = \boldsymbol{c}_i{}^T\ \boldsymbol{\theta}\ \boldsymbol{c}_j)$$

true weights of the adjacency matrix

mixing parameter

additional SBM contribution, to model edge stenght

parameters for weights, community based

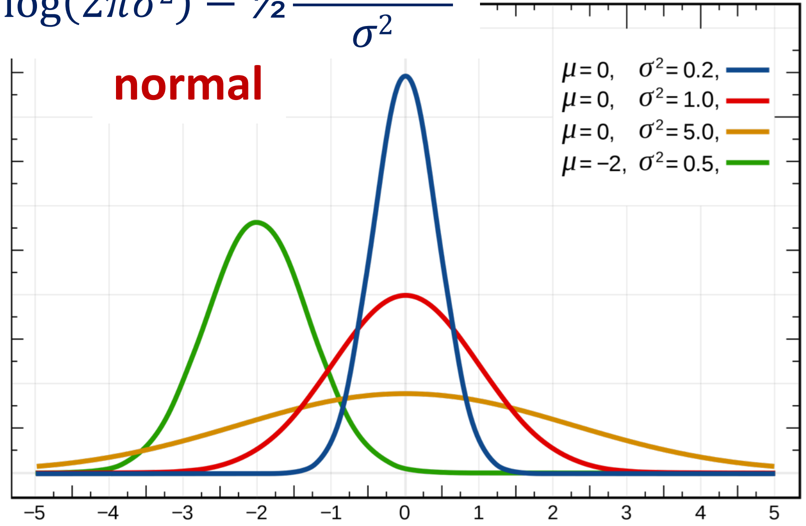
chosen distribution for weights, e.g., Gaussian

Weighted SBM

a few plausible distributions

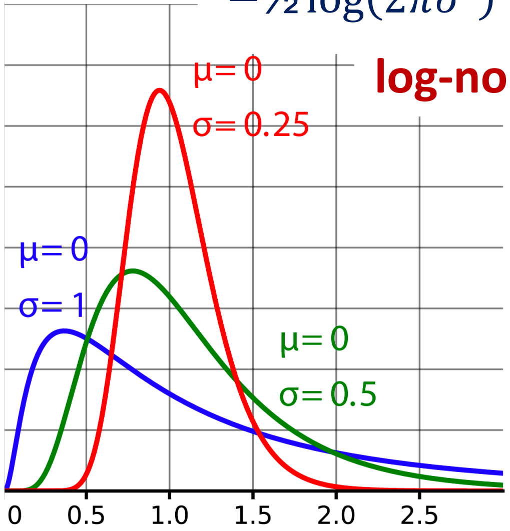
$$-½\log(2\pi\sigma^2) - ½\frac{(a-\mu)^2}{\sigma^2}$$

normal



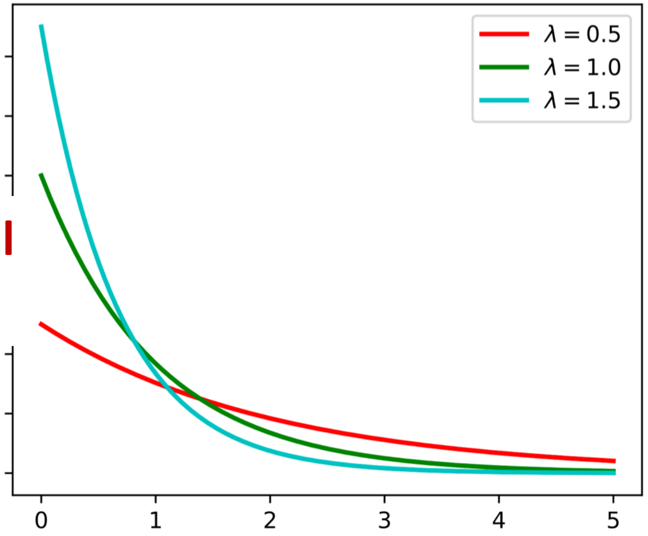
$$-½\log(2\pi\sigma^2) - \log(a) - ½\frac{(\log(a)-\mu)^2}{\sigma^2}$$

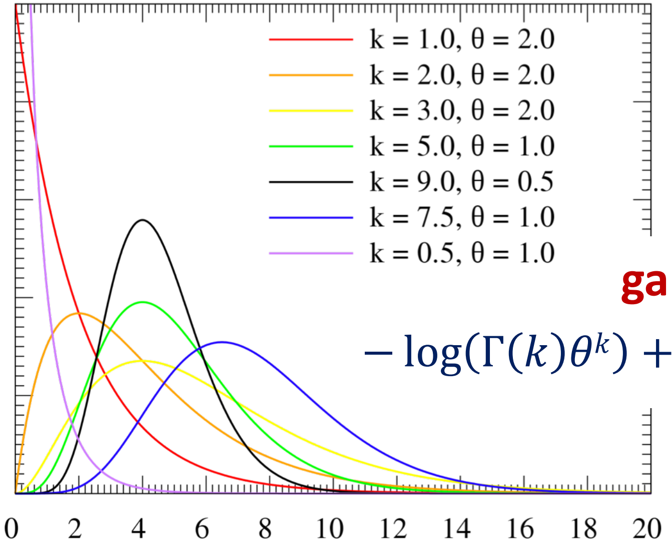
log-normal



exponential

$$\log(\lambda) - \lambda a$$





gamma

$$-\log(\Gamma(k)\theta^k) + (k-1)\log(a) - \frac{a}{\theta}$$

Weighted SBM

the exponential and Gaussian case

$$\log p(\boldsymbol{A}|\boldsymbol{L},\boldsymbol{C}) = \sum_{i,j} a_{ij} \log(\boldsymbol{c}_i^T\, \boldsymbol{L}\, \boldsymbol{c}_j) - w_{ij}\, \boldsymbol{c}_i^T\, \boldsymbol{L}\, \boldsymbol{c}_j$$

can be converted in useful form by variational inequality (it exploits the concavity of log(x))

same here, thanks to concavity of both functions –x^2 and -1/x

$$\log p(\boldsymbol{A}|\boldsymbol{M},\boldsymbol{\Sigma},\boldsymbol{C}) = ½ \sum_{i,j} -a_{ij} \log(\boldsymbol{c}_i^T\, \boldsymbol{\Sigma}\, \boldsymbol{c}_j) - a_{ij} \frac{(w_{ij} - \boldsymbol{c}_i^T\, \boldsymbol{M}\, \boldsymbol{c}_j)^2}{\boldsymbol{c}_i^T\, \boldsymbol{\Sigma}\, \boldsymbol{c}_j}$$

-log(x) is convex, so in this case the approach does not work

variational Bayes solutions are needed for overlapping communities!

Takeaways

for stochastic block models

- ☐ an advanced generative model to capture the underlying network structure
- ☐ easily extendable to a many different scenarios
- ☐ optimization problem is difficult to solve (but efficient methods exist)
- ☐ not fully scalable
- ☐ some models (e.g., degree-corrected SBM) provide results related to NMI, and modularity
- ☐ performance not always striking with synthetic networks
- ☐ would be interesting to see it implemented in your projects ☺

Dendrograms

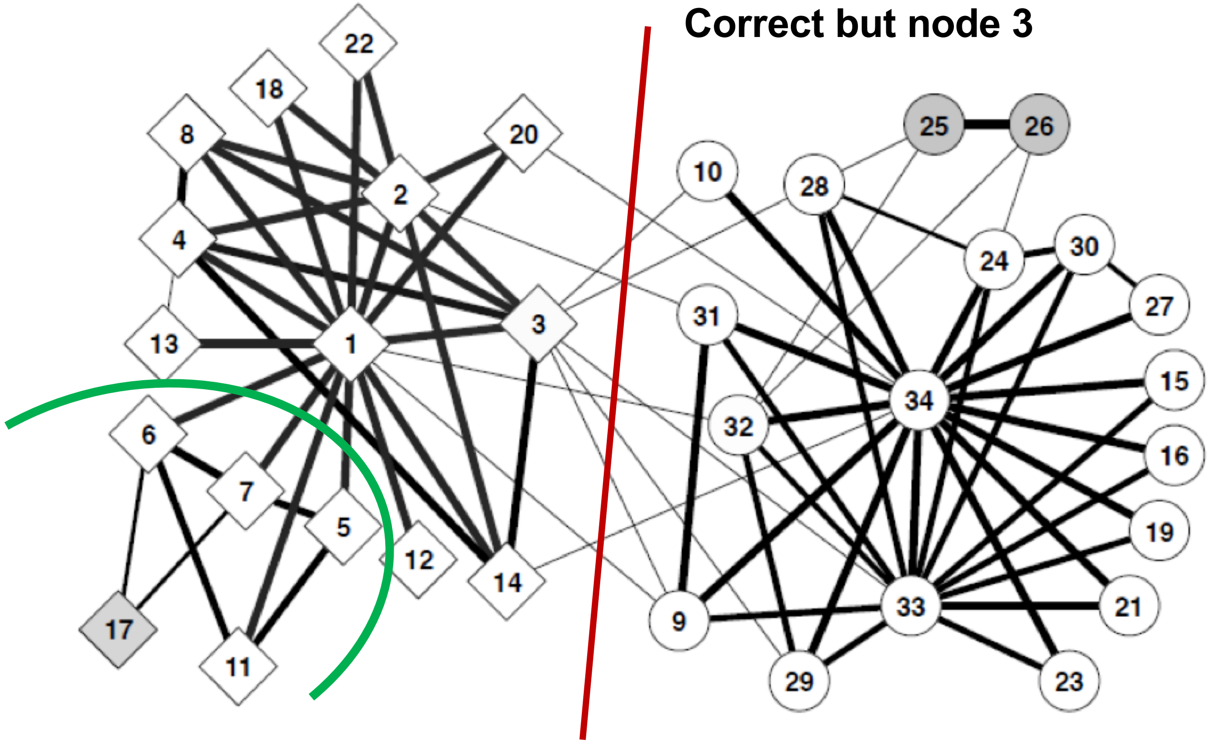
an older (but still in use) approach to community detection

Dendrograms

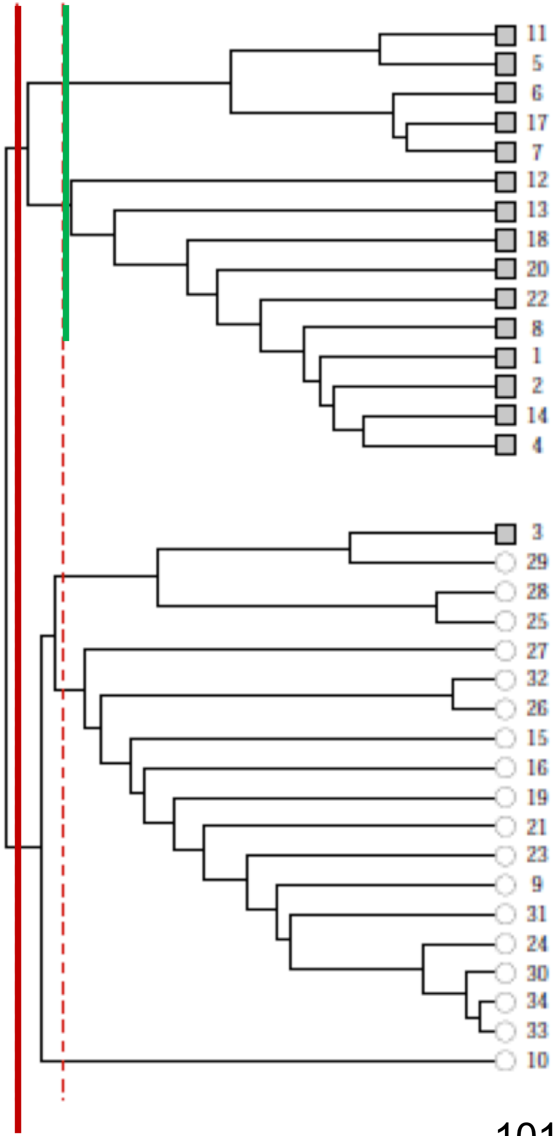
overall idea for community detection

Zachary's karate club network

1 - instructor
34 - president
Correct but node 3

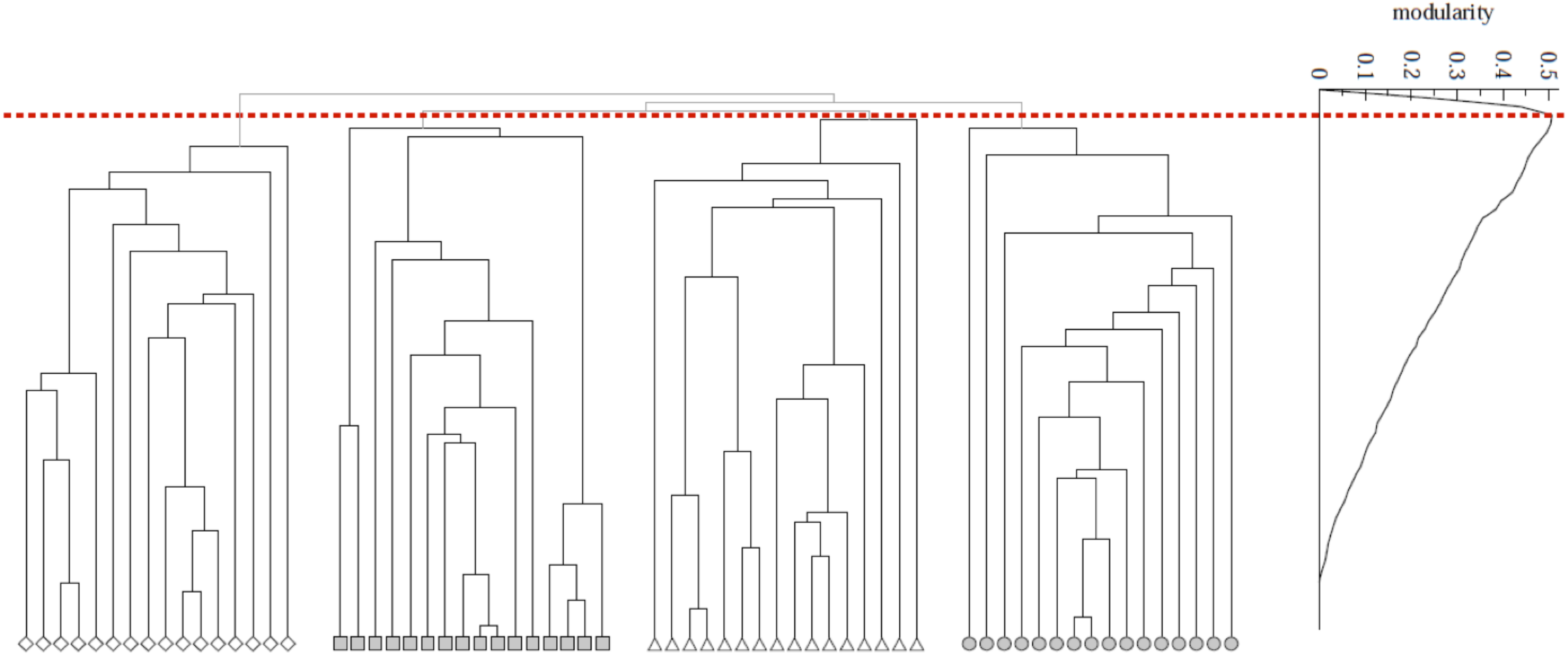


social ties and rivalries in a university club; during observation conflict led the group to split



Modularity in dendrograms

for selecting the number of clusters



… but NMI, normalized cut or InfoMap measures would also work

Two approaches

to dendrograms

Dendrograms is an hierarchical clustering algorithm that can be approached in two ways:

- ☐ Agglomerative: progressively **add** edges, from the strongest and ending with the weakest ones; new connected components that arise identify a new (upper) dendrogram level
- ☐ Divisive: progressively **delete** edges, from the strongest and ending with the weakest ones; new disconnected components that arise identify a new (lower) dendrogram level

Performance strongly depends on the chosen weight (local weight definitions typically provide weak solutions)

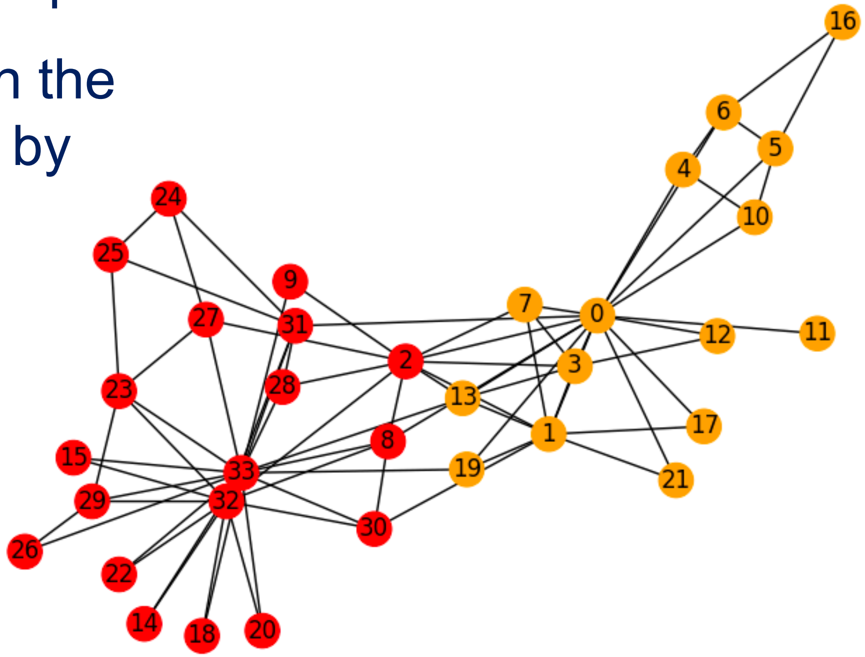
Girvan-Newman method

a divisive approach

Girvan, Newman. "Community structure in social and biological networks." (2002)
https://www.pnas.org/doi/full/10.1073/pnas.122653799

Repeat until no edges are left in the graph:

- ☐ (re)calculate edge betweenness in the current graph – complexity O(*LN*) by using a smart algorithm
- ☐ remove edges with highest betweenness



Complexity $O(L^2N)$ → pretty scalable

Recalculation step is essential to detect meaningful communities

May provide poor results: useful method, far from perfect

Edge beteweeness

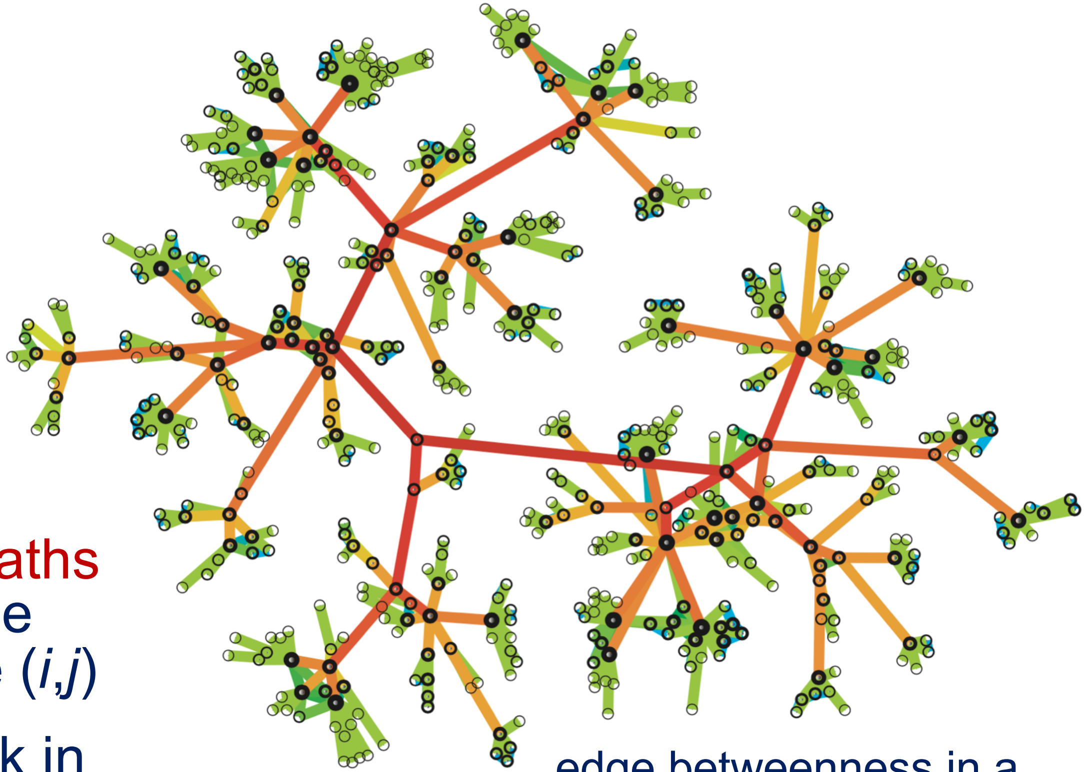
a generalization of node betweenness

$$b_{ij} = \sum_{(k,\ell)\in\mathcal{N}^2} \frac{\sigma_{k,\ell}(i,j)}{\sigma_{k,\ell}}$$

where σ_{kl} is the # of shortest paths connecting *k* to *l*, and $\sigma_{kl}(i,j)$ the subset of these including edge (*i*,*j*)

- ☐ expresses centrality of a link in the network
- ☐ can be normalized to range [0,1]

$$(b_{ij} - b_{\min}) / (b_{\max} - b_{\min})$$



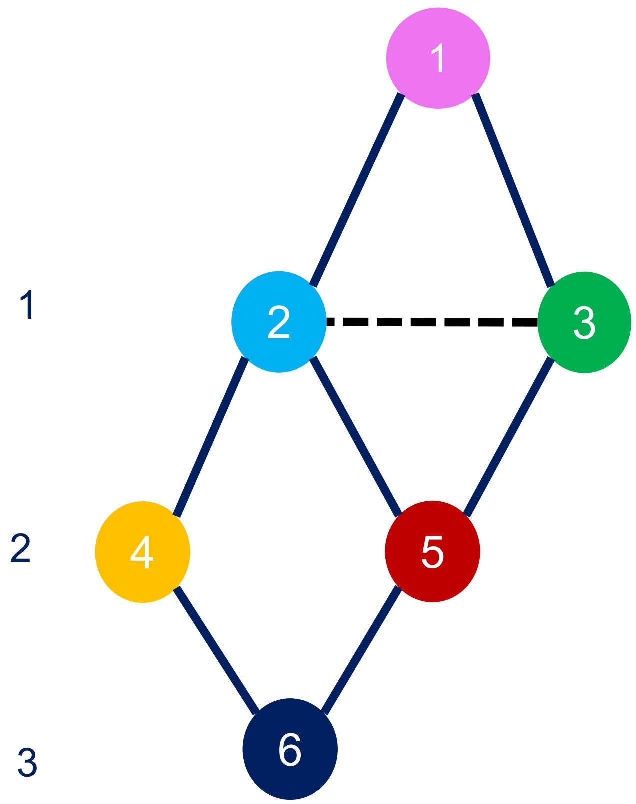
edge betweenness in a cellular call network

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

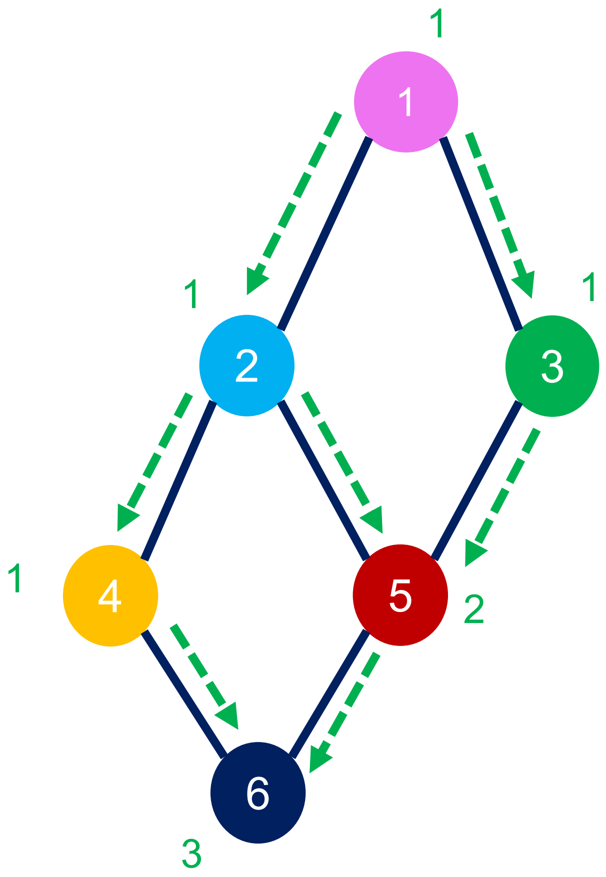
Calculating betweeness

Part 1

Breadth first search
(from node 1)



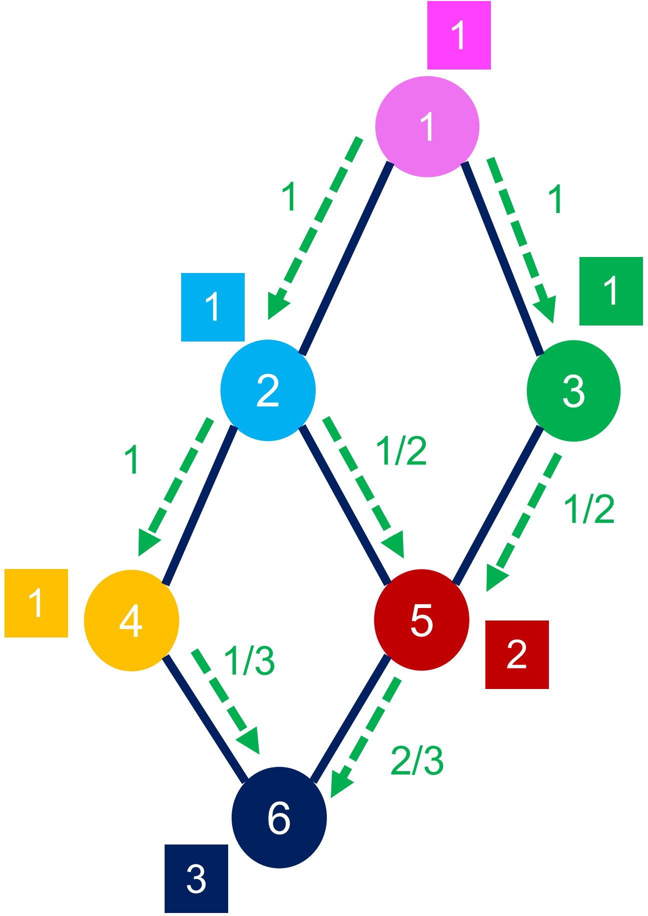
Count # of shortest paths
(from node 1)



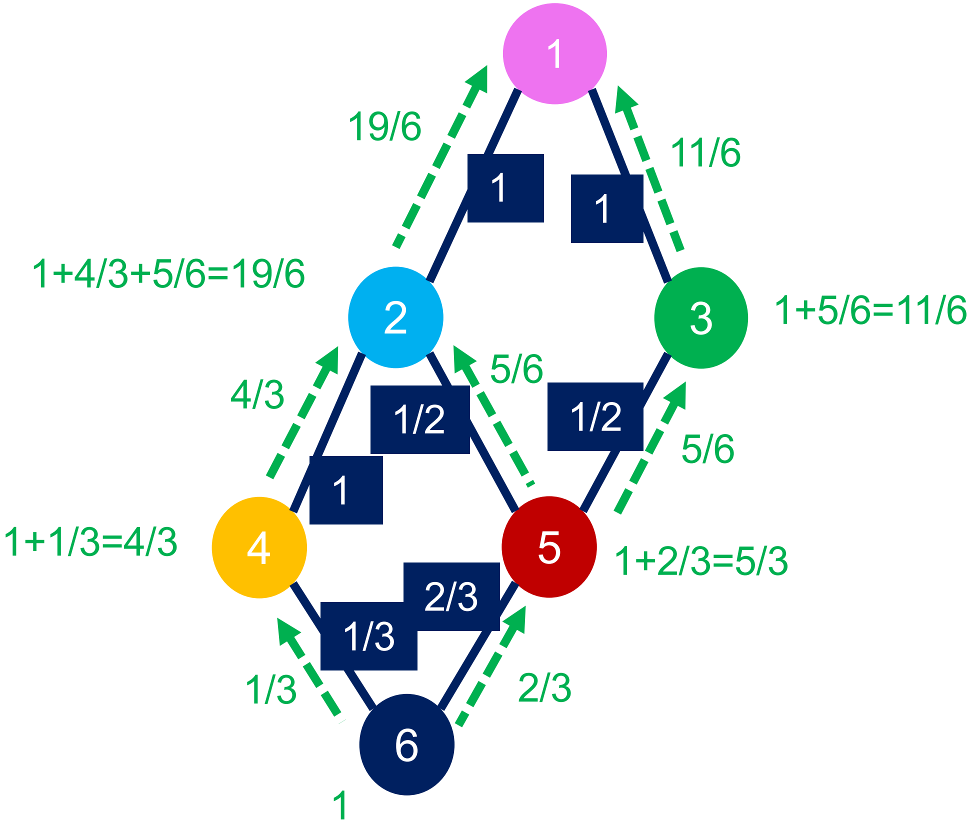
Calculating betweeness

Part 2

Measure edge flow (fractions)



Measure edge betweenness (from node 1)

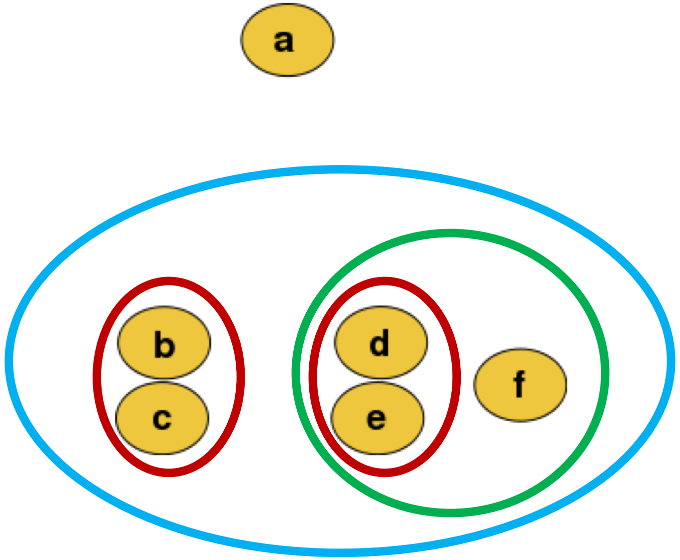


… then repeat for all other nodes!!! O(LN)

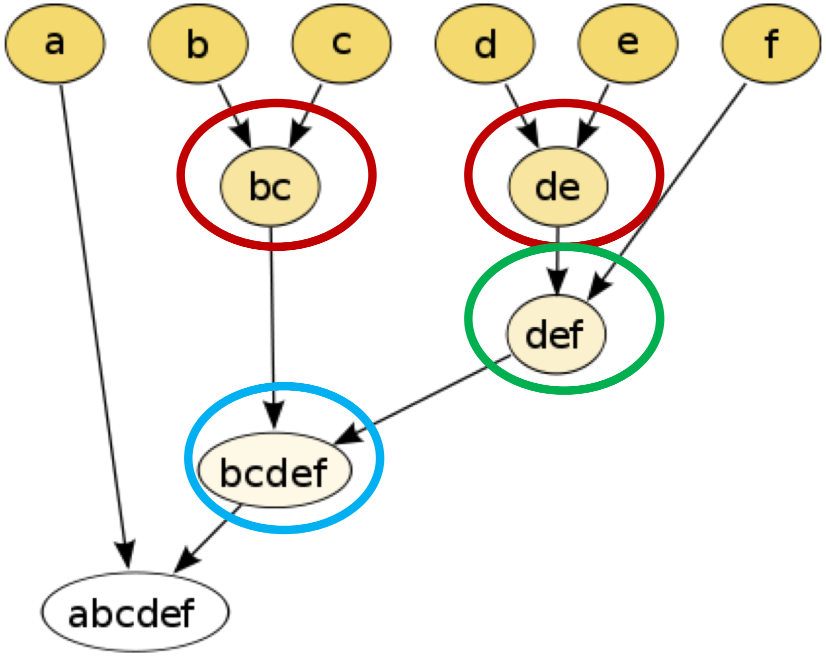
Agglomerative clustering

a toy example based on Euclidean distance

Network



Dendrogram



Algorithm

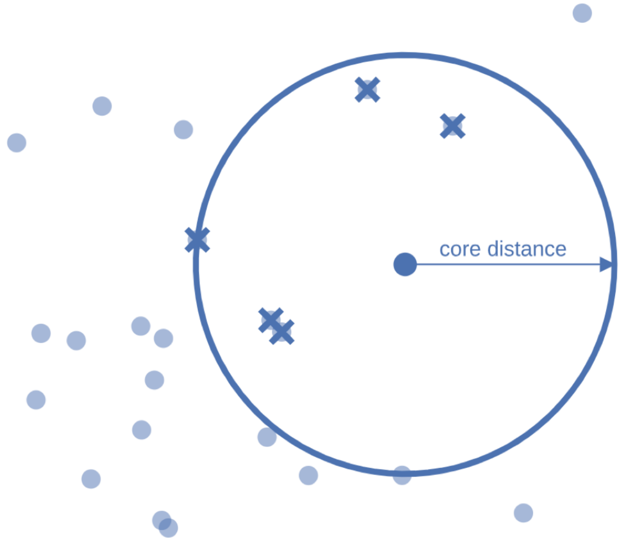
- ☐ Start with each node being a separate community
- ☐ Progressively add a community to the one that is closer

HDBSCAN

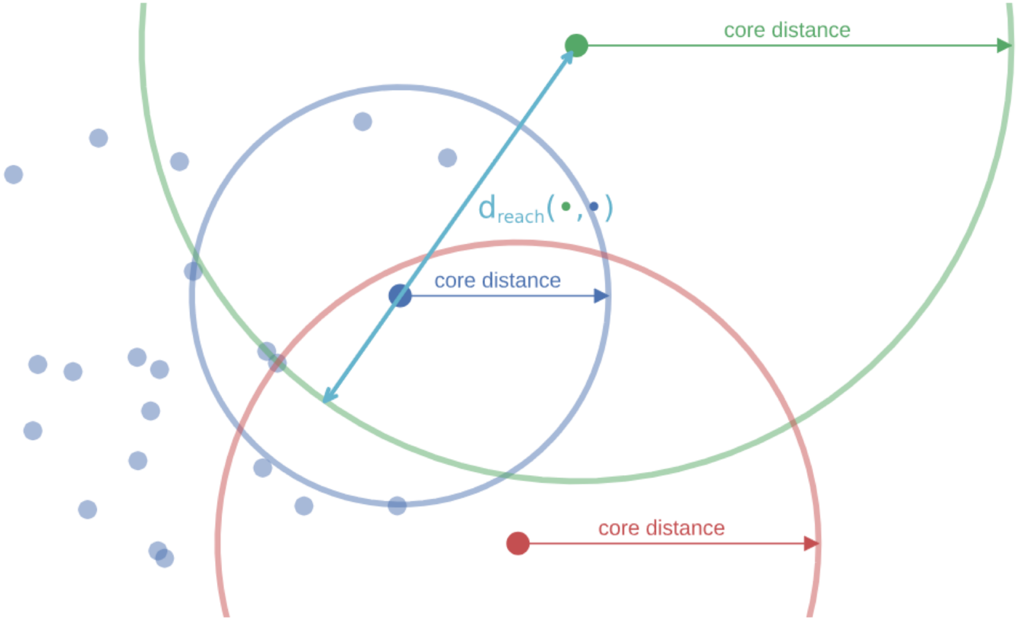
an agglomerative approach

K = number of nearest neighbours to be considered

this sets the core distance of a node



the mutual reachability distance between two nodes is the maximum between their effective distance and their core distances



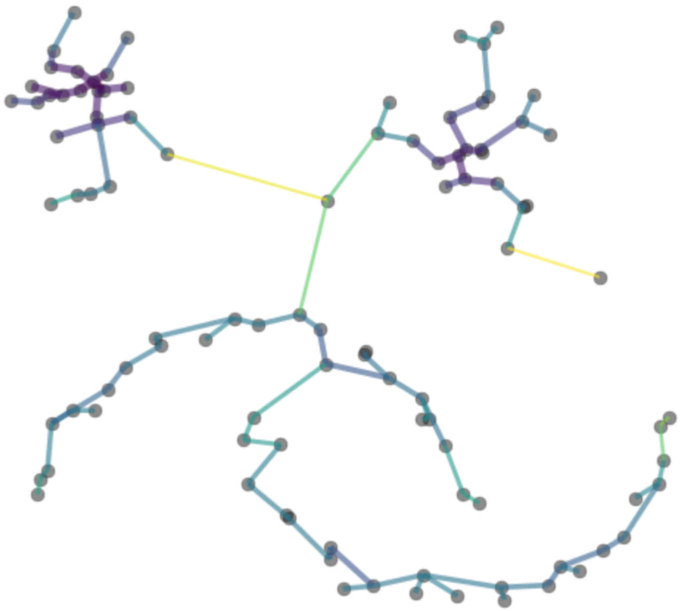
under this metric dense points (with low core distance) remain the same distance from each other but sparser points are pushed away to be at least their core distance away from any other point

HDBSCAN

the clusters hierarchy

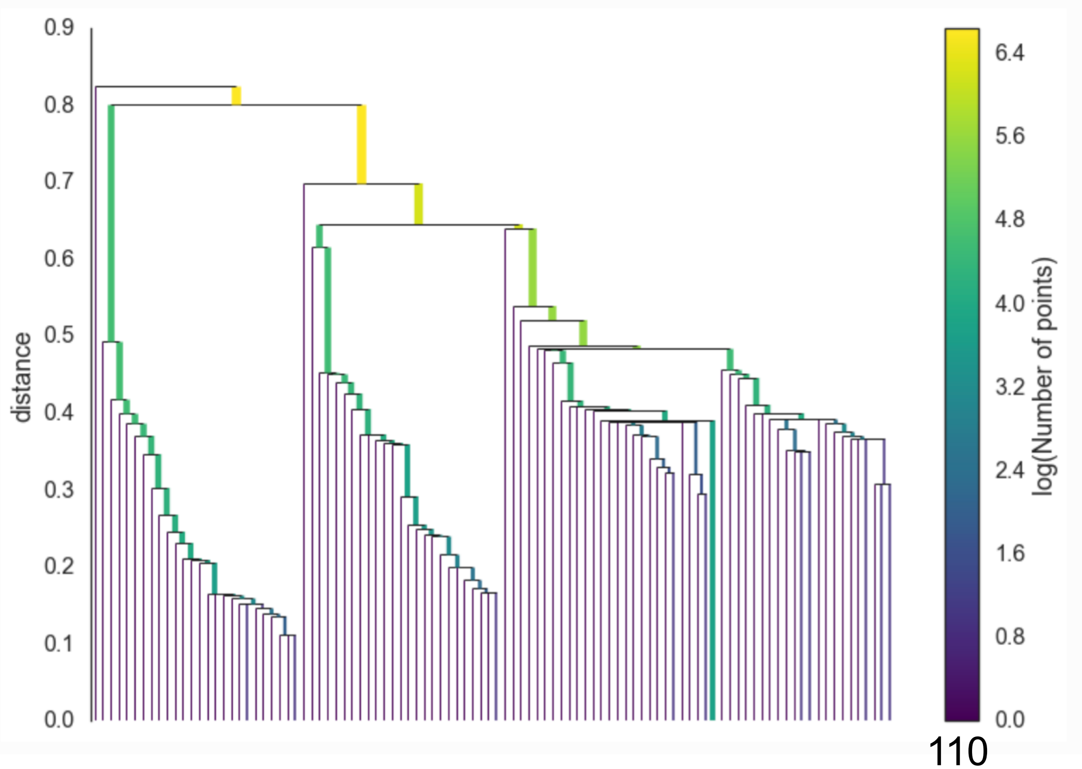
Step 1

by using the mutual reachability distance, build a minimum spanning tree (a spanning three whose sum of the edge weights is as small as possible)



Step 2

build a cluster hierarchy by adding links in the spanning tree in order of distance, starting from the links with smaller distance (agglomerative approach)

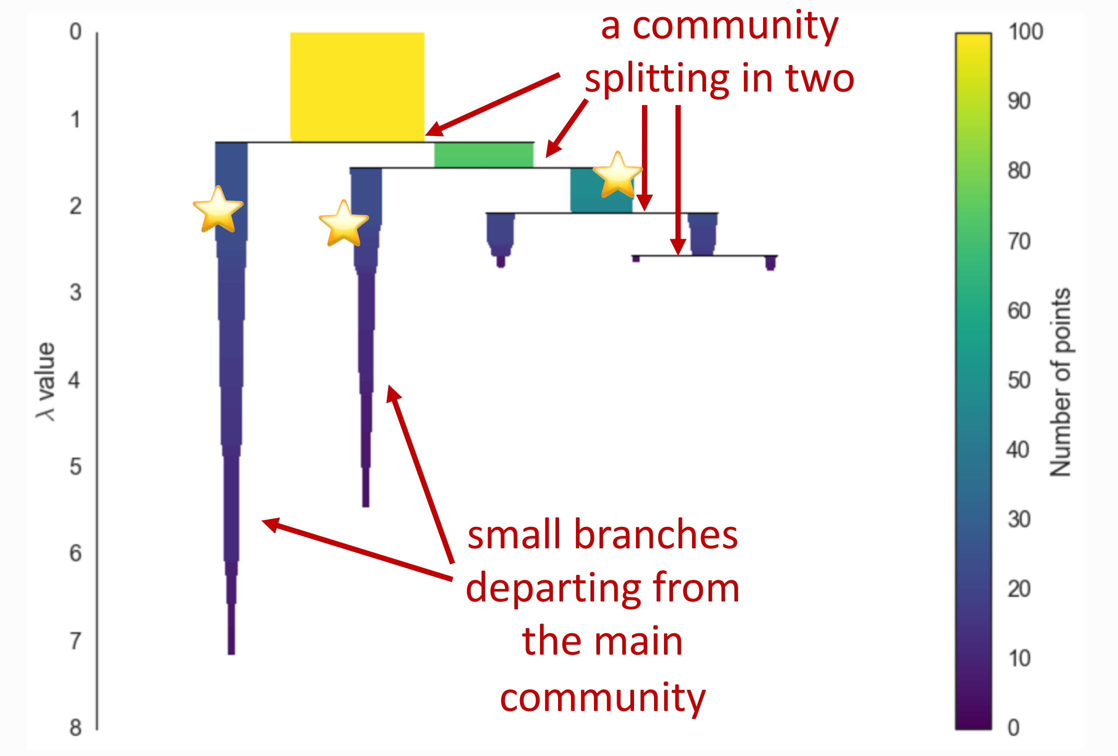


HDBSCAN

identifying good communities

Step 3

simplify the hierarchy by removing (from top to bottom) those branches that have size less than the minimum cluster size parameter, to avoid outliers



a community splitting in two

small branches departing from the main community

Step 4

identify a stability value for each cluster as

$$\sum_n \frac{1}{d_n} - \frac{1}{d_{birth}}$$

distance at which node n fell out of the cluster

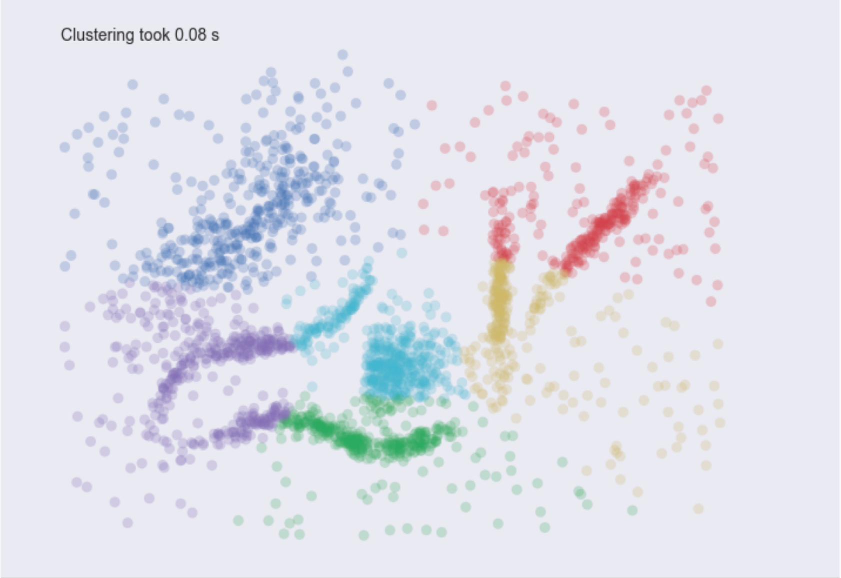
distance at which the cluster is born

keep the parent cluster (⭐) if its stability is bigger than the sum of the stabilities of its two child clusters, otherwise iterate (keep the communities that last longer)

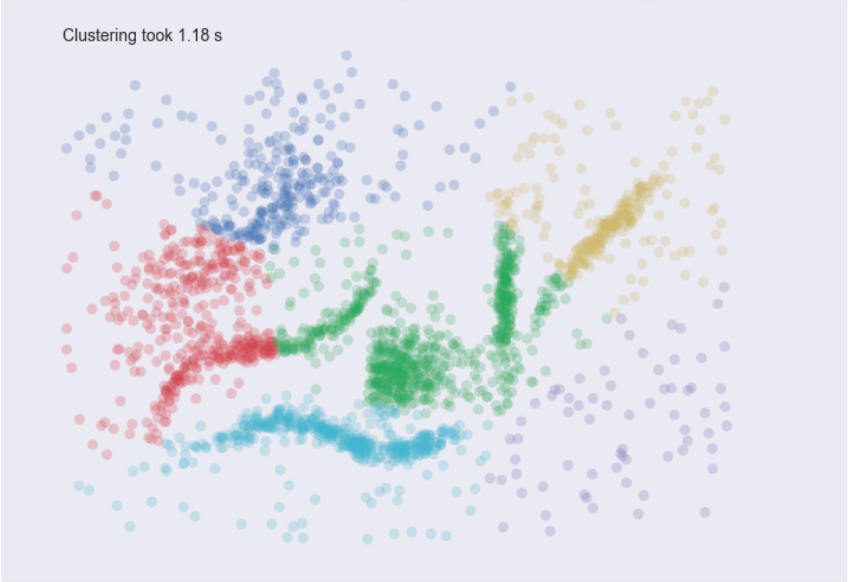
A comparison example

https://hdbscan.readthedocs.io/en/latest/index.html

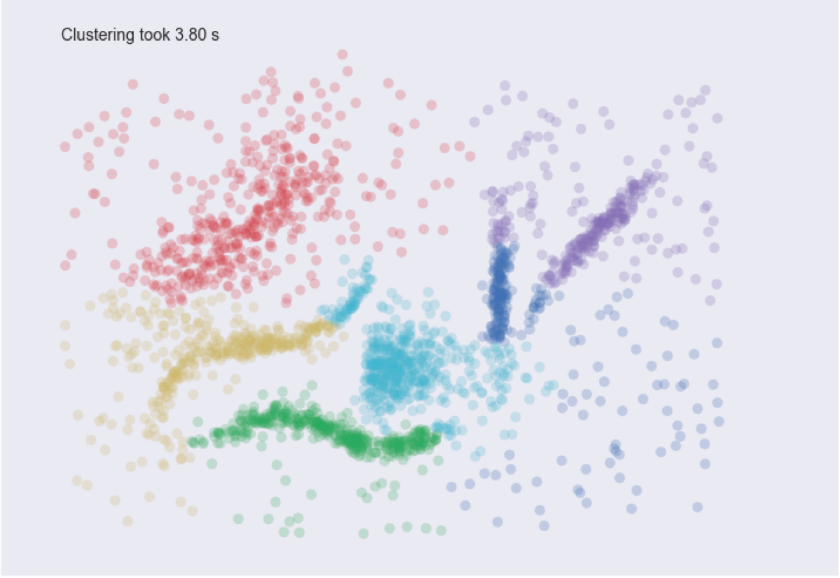
Clusters found by KMeans



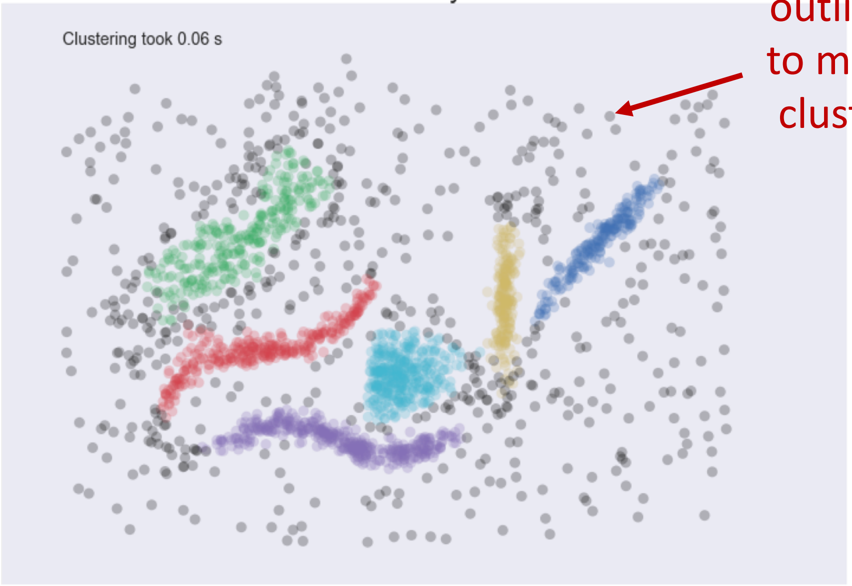
Clusters found by SpectralClustering



Clusters found by AgglomerativeClustering



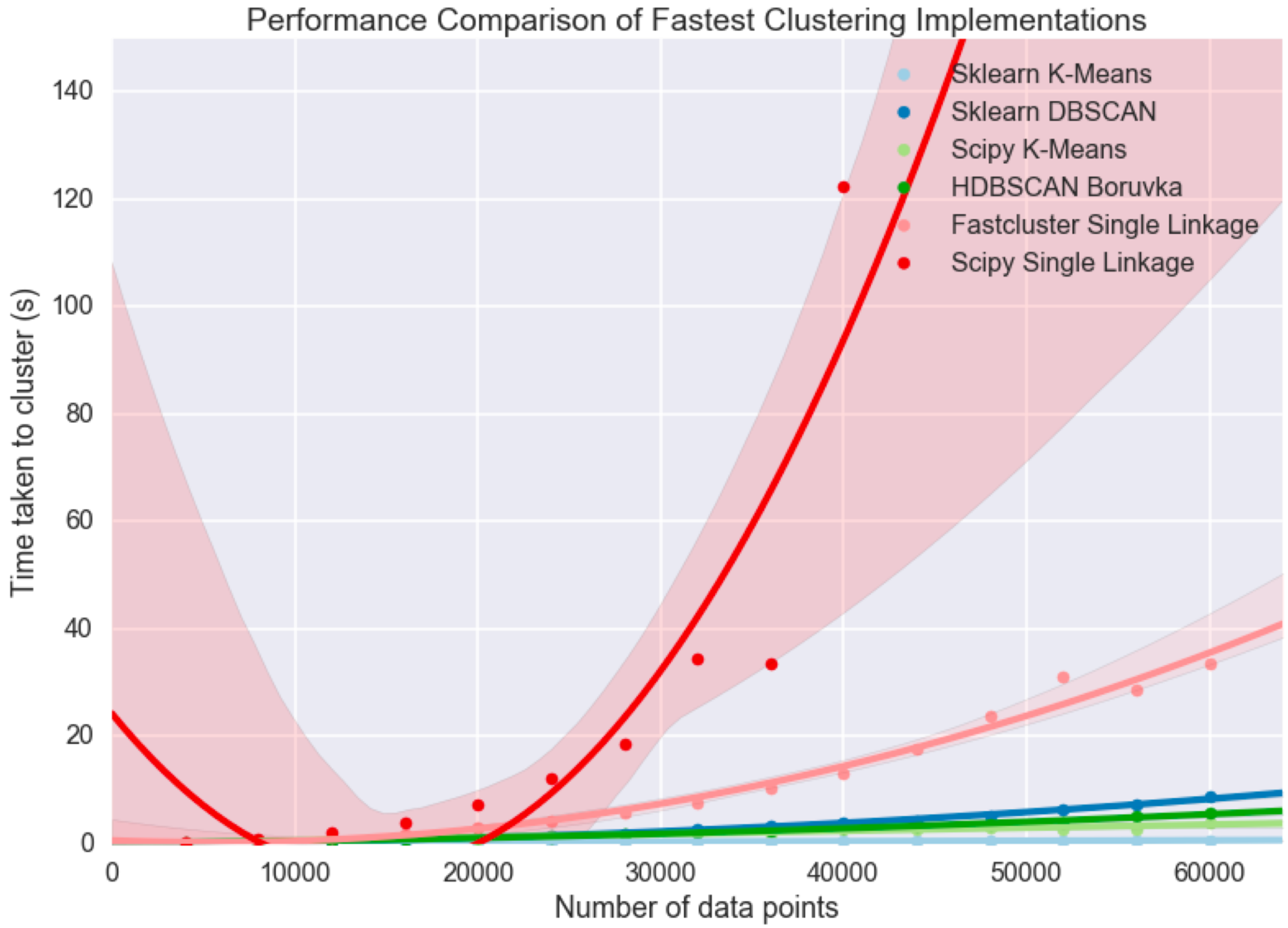
Clusters found by HDBSCAN



outliers due
to minimum
cluster size

Complexity comparison

https://hdbscan.readthedocs.io/en/latest/index.html



HDBSCAN parameters

main parameters

```
class hdbscan.hdbscan_.HDBSCAN(
min_cluster_size = 5,
min_samples = None,
```

parameter K identifying the core distance, set by default to min_cluster_size (small K → true distances and few outliers, larger K → many outliers)

```
metric = 'euclidean',
```

how to calculate distances from data vectors, e.g., 'cosine', 'dice', 'euclidean' – can also be 'precomputed' from a similarity matrix ***A*** in which case $d_{ij}=1/a_{ij}$ or if correlation values ***A*** are available $d_{ij}=1-a_{ij}$

```
algorithm = 'best',
approx_min_span_tree = True,
```

options for the spanning tree algorithm

```
cluster_selection_method = 'eom',
```

the more elaborate excess of mass approach ('eom'), or simply select leaves ('leaf') for a finer partition

```
allow_single_cluster = False)
```

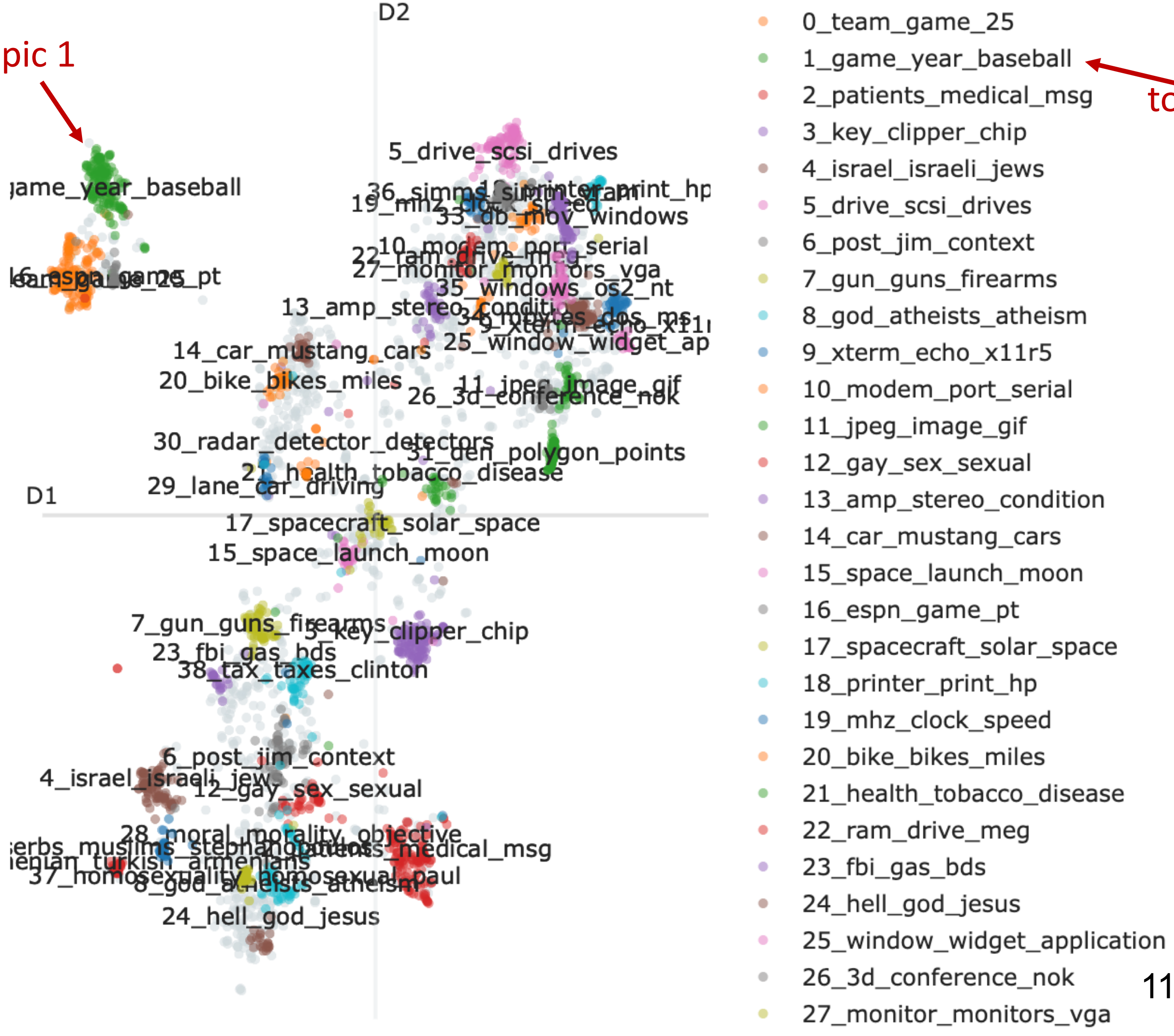
HDBSCAN in BERTopic

clustering documents into different topics

1. each document is mapped into an embedding (vector) by BERT

2. cosine metric is used to identify distances among documents

3. HDBSCAN is run to identify topics



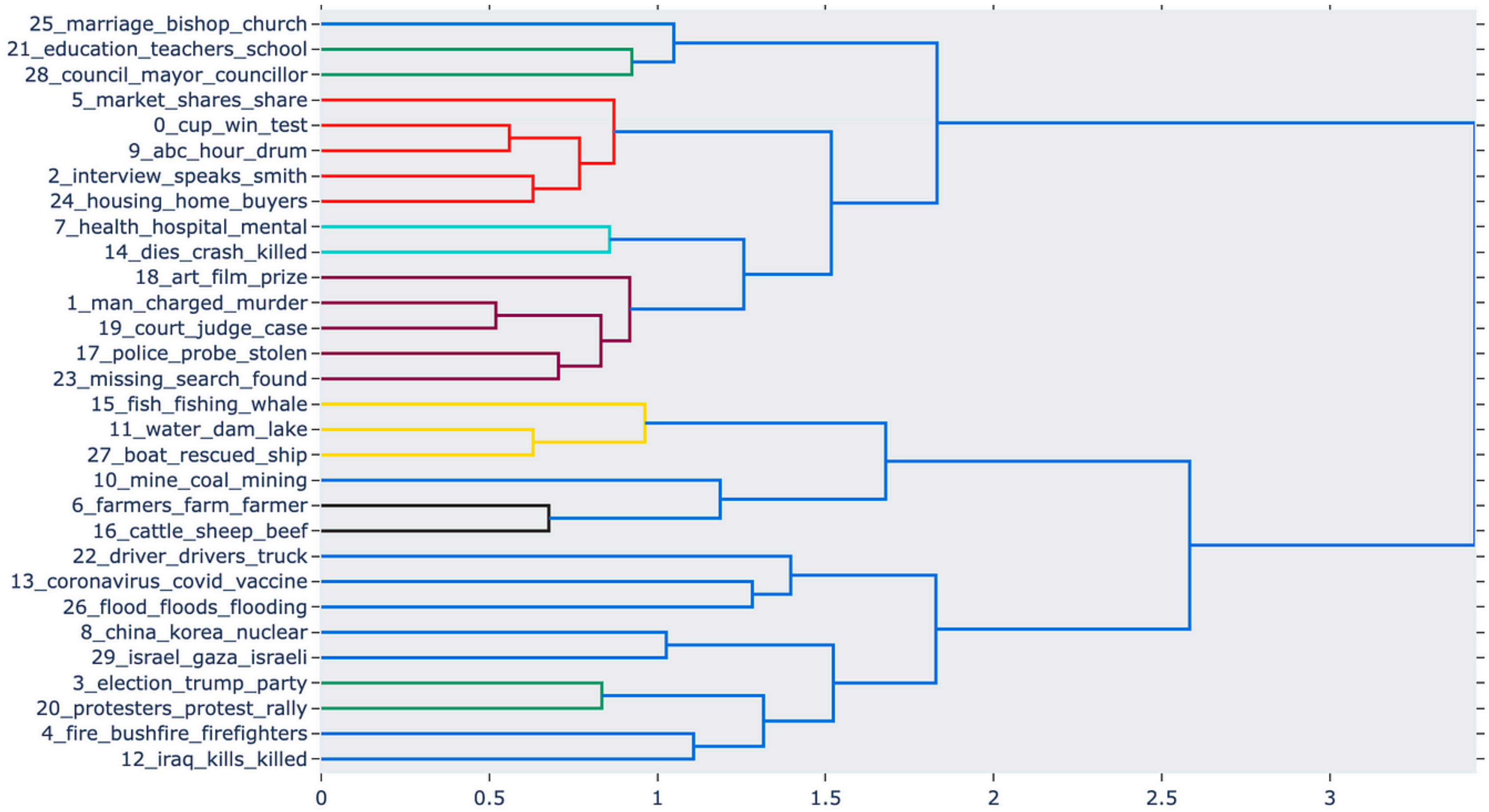
topic 1

topic 1

HDBSCAN in BERTopic

hierarchical clustering of topics

HDBSCAN hierarchy of topics, with those selected



Takeaways

for HDBSCAN

- an advanced agglomerative method to identify communities (clusters)
- works on distance (or similarity) data
- fully scalable
- it implements overlapping communities (soft clustering)
- striking performance with communities that are not exaggeratedly overlapping in space
- it naturally generates outliers, since small clusters are dropped
- mostly dependent on the min_cluster_size parameter

Clique percolation

what should never be used for overlapping community detection

Clique percolation

general idea

Idea

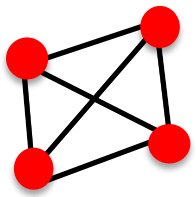
- Two nodes belong to the same community if they can be connected through adjacent *k* cliques

k clique

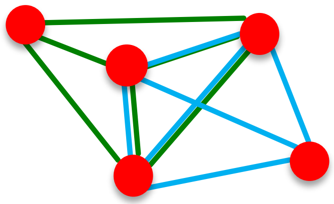
- Fully connected graph of *k* nodes

Adjacent *k* cliques

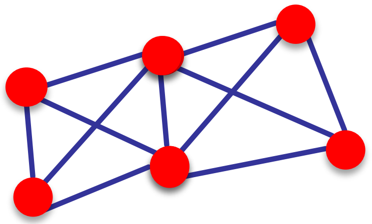
- Overlap in *k-1* nodes



4-clique



Adjacent 4-cliques

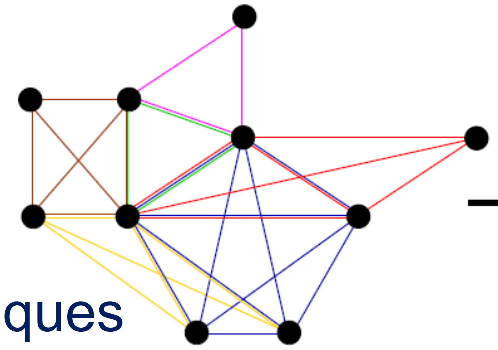


Non-adjacent 4-cliques

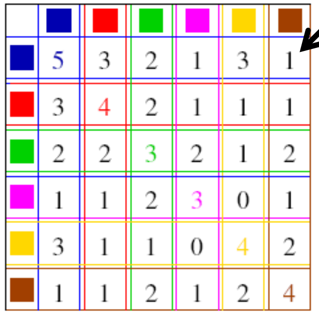
Clique percolation

the algorithm

(1) identify cliques



Graph



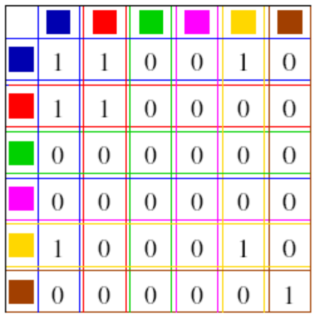
	🟦	🟥	🟩	🟪	🟨	🟫
🟦	5	3	2	1	3	1
🟥	3	4	2	1	1	1
🟩	2	2	3	2	1	2
🟪	1	1	2	3	0	1
🟨	3	1	1	0	4	2
🟫	1	1	2	1	2	4

(2) Build a clique overlap matrix

Clique overlap matrix

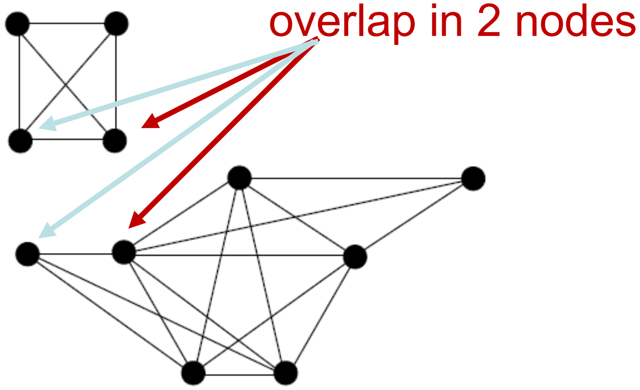
k=4

(3) Set a threshold (the one providing the richest community structure = most widely distributed cluster sizes)



	🟦	🟥	🟩	🟪	🟨	🟫
🟦	1	1	0	0	1	0
🟥	1	1	0	0	0	0
🟩	0	0	0	0	0	0
🟪	0	0	0	0	0	0
🟨	1	0	0	0	1	0
🟫	0	0	0	0	0	1

Thresholded matrix at 3

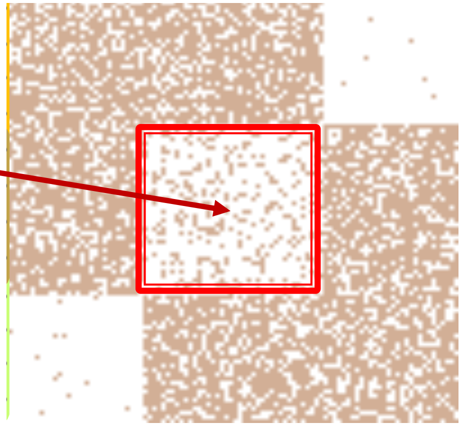


Communities (connected components)

Takeaways

for clique percolation

- simple approach (too simple?)
- reasonably scalable
- it implements overlapping communities
- very poor performance
- it is based on a wrong overlapping model



fewer connections in the overlap

- do not use it!

Wrap-up

on community detection

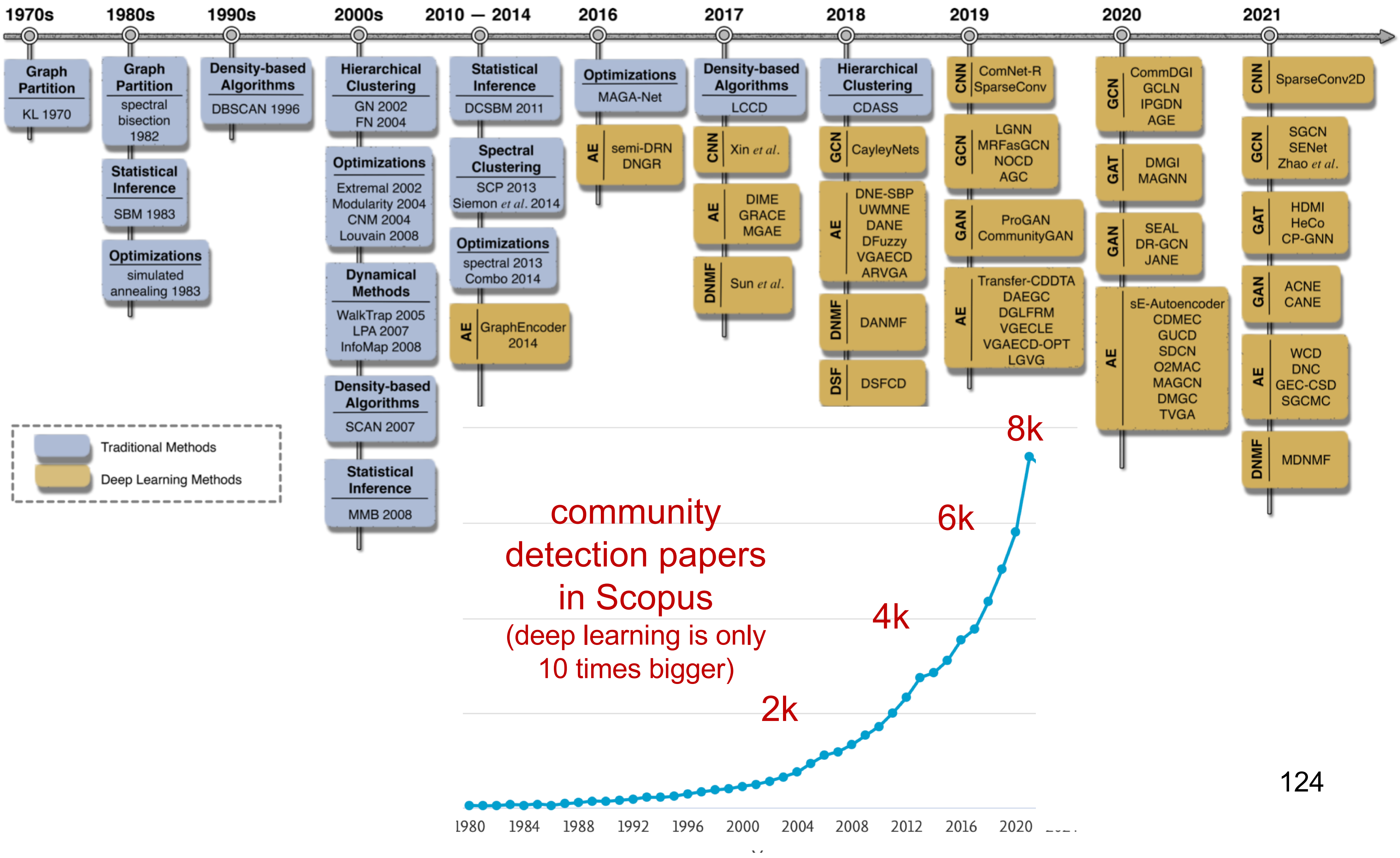
Algorithms

for unsupervised community detection

algorithm	rationale	weighted	directed	overlapping	signed	scalable
Louvain	optimizes modularity	YES	YES	YES	YES	YES
Spectral clustering ✗	optimizes Ncut based on the normalized Laplacian	YES	YES	YES	NO	YES
Infomap	optimizes the InfoMap measure	YES	YES	YES	NO	YES
BigCLAM	model based approach	NO	NO	YES	NO	YES
SBM ✗	model based approach	YES	YES	NO	YES	NO
MM-SBM	model based approach	YES	YES	YES	YES	NO
Girvan-Newman ✗	divisive dendrogram based on betweenness	YES	YES	NO	NO	NO
HDBSCAN	agglomerative approach based on distances	YES	NO	YES	YES	YES
Clique percolation ✗	approach based on cliques overlapping	YES	NO	YES	NO	NO

Timeline

on community detection development



1970s
- **Graph Partition**: KL 1970

1980s
- **Graph Partition**: spectral bisection 1982
- **Statistical Inference**: SBM 1983
- **Optimizations**: simulated annealing 1983

1990s
- **Density-based Algorithms**: DBSCAN 1996

2000s
- **Hierarchical Clustering**: GN 2002, FN 2004
- **Optimizations**: Extremal 2002, Modularity 2004, CNM 2004, Louvain 2008
- **Dynamical Methods**: WalkTrap 2005, LPA 2007, InfoMap 2008
- **Density-based Algorithms**: SCAN 2007
- **Statistical Inference**: MMB 2008

2010 — 2014
- **Statistical Inference**: DCSBM 2011
- **Spectral Clustering**: SCP 2013, Siemon *et al.* 2014
- **Optimizations**: spectral 2013, Combo 2014
- AE: GraphEncoder 2014

2016
- **Optimizations**: MAGA-Net
- AE: semi-DRN, DNGR

2017
- **Density-based Algorithms**: LCCD
- CNN: Xin *et al.*
- AE: DIME, GRACE, MGAE
- DNMF: Sun *et al.*

2018
- **Hierarchical Clustering**: CDASS
- GCN: CayleyNets
- AE: DNE-SBP, UWMNE, DANE, DFuzzy, VGAECD, ARVGA
- DNMF: DANMF
- DSF: DSFCD

2019
- CNN: ComNet-R, SparseConv
- GCN: LGNN, MRFasGCN, NOCD, AGC
- GAN: ProGAN, CommunityGAN
- AE: Transfer-CDDTA, DAEGC, DGLFRM, VGECLE, VGAECD-OPT, LGVG

2020
- GCN: CommDGI, GCLN, IPGDN, AGE
- GAT: DMGI, MAGNN
- GAN: SEAL, DR-GCN, JANE
- AE: sE-Autoencoder, CDMEC, GUCD, SDCN, O2MAC, MAGCN, DMGC, TVGA

2021
- CNN: SparseConv2D
- GCN: SGCN, SENet, Zhao *et al.*
- GAT: HDMI, HeCo, CP-GNN
- GAN: ACNE, CANE
- AE: WCD, DNC, GEC-CSD, SGCMC
- DNMF: MDNMF

Traditional Methods

Deep Learning Methods

community
detection papers
in Scopus
(deep learning is only
10 times bigger)

Readings

on community detection approaches and measures

Fortunato, "Community detection in graphs." (2010)
https://doi.org/10.1016/j.physrep.2009.11.002

Fortunato, Newman. "20 years of network community detection." (2022)
https://www.nature.com/articles/s41567-022-01716-7

Clement, Wilkinson. "A review of stochastic block models and extensions for graph clustering." (2019)
https://appliednetsci.springeropen.com/articles/10.1007/s41109-019-0232-2

Di, et al. "A survey of community detection approaches: From statistical modeling to deep learning." (2021)
https://ieeexplore.ieee.org/abstract/document/9511798

Xing, et al. "A comprehensive survey on community detection with deep learning." (2022).
https://doi.org/10.1109/TNNLS.2021.3137396

SBMs in multi-layer networks

some readings

- ☐ De Bacco, Power, Larremore , Moore, "Community detection, link prediction, and layer interdependence in multilayer networks." (2017)

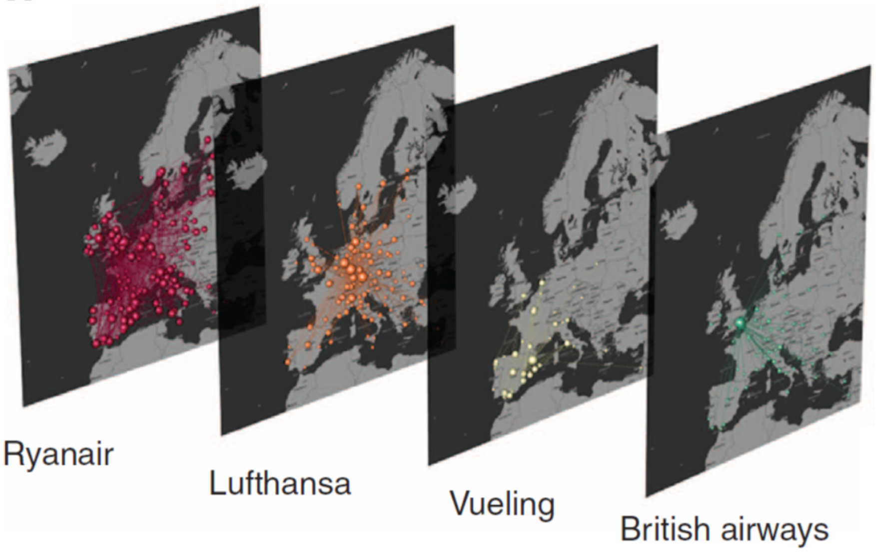
  core.ac.uk/download/pdf/146486854.pdf

- ☐ Contisciani, Power, De Bacco. "Community detection with node attributes in multilayer networks." (2020)

  www.nature.com/articles/s41598-020-72626-y

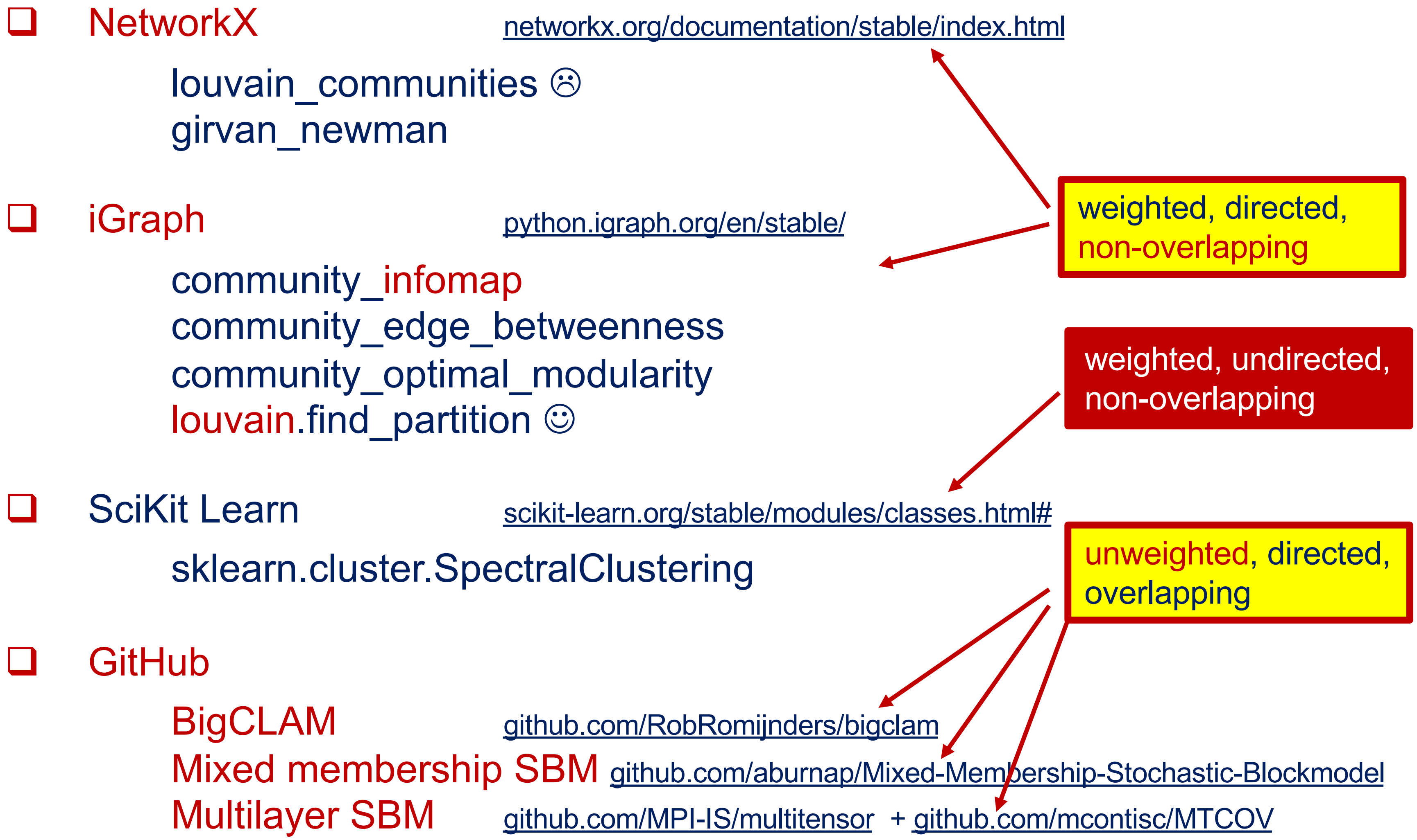
- ☐ Contisciani, Battiston, De Bacco. "Inference of hyperedges and overlapping communities in hypergraphs." (2022)

  www.nature.com/articles/s41467-022-34714-7



Python software tools

a few of the many available



☐ NetworkX networkx.org/documentation/stable/index.html

louvain_communities ☹
girvan_newman

☐ iGraph python.igraph.org/en/stable/

community_infomap
community_edge_betweenness
community_optimal_modularity
louvain.find_partition ☺

☐ SciKit Learn scikit-learn.org/stable/modules/classes.html#

sklearn.cluster.SpectralClustering

☐ GitHub

BigCLAM github.com/RobRomijnders/bigclam
Mixed membership SBM github.com/aburnap/Mixed-Membership-Stochastic-Blockmodel
Multilayer SBM github.com/MPI-IS/multitensor + github.com/mcontisc/MTCOV

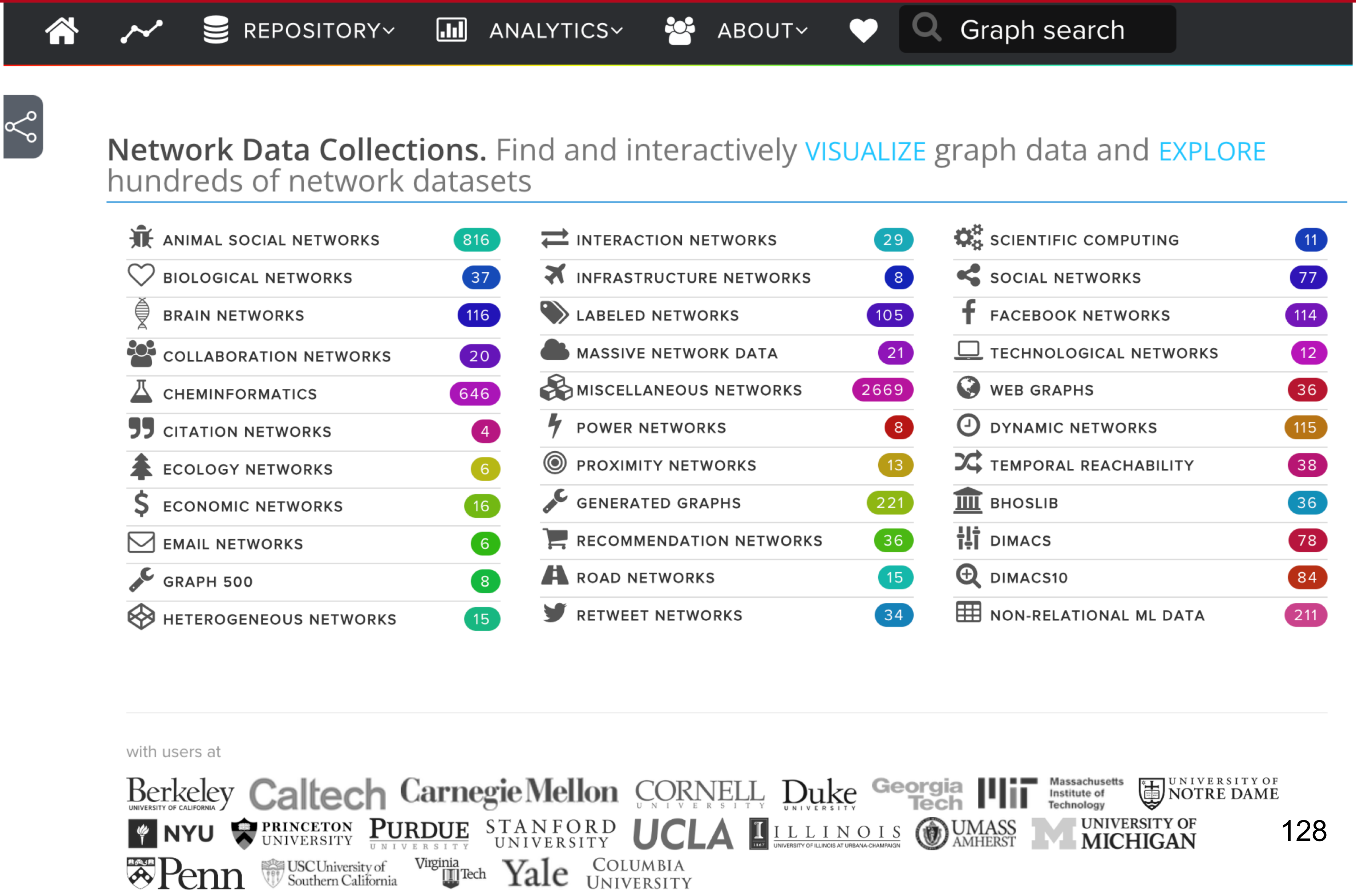
weighted, directed,
non-overlapping

weighted, undirected,
non-overlapping

unweighted, directed,
overlapping

Network repository

https://networkrepository.com/index.php



Network Data Collections. Find and interactively VISUALIZE graph data and EXPLORE hundreds of network datasets

ANIMAL SOCIAL NETWORKS	816
BIOLOGICAL NETWORKS	37
BRAIN NETWORKS	116
COLLABORATION NETWORKS	20
CHEMINFORMATICS	646
CITATION NETWORKS	4
ECOLOGY NETWORKS	6
ECONOMIC NETWORKS	16
EMAIL NETWORKS	6
GRAPH 500	8
HETEROGENEOUS NETWORKS	15
INTERACTION NETWORKS	29
INFRASTRUCTURE NETWORKS	8
LABELED NETWORKS	105
MASSIVE NETWORK DATA	21
MISCELLANEOUS NETWORKS	2669
POWER NETWORKS	8
PROXIMITY NETWORKS	13
GENERATED GRAPHS	221
RECOMMENDATION NETWORKS	36
ROAD NETWORKS	15
RETWEET NETWORKS	34
SCIENTIFIC COMPUTING	11
SOCIAL NETWORKS	77
FACEBOOK NETWORKS	114
TECHNOLOGICAL NETWORKS	12
WEB GRAPHS	36
DYNAMIC NETWORKS	115
TEMPORAL REACHABILITY	38
BHOSLIB	36
DIMACS	78
DIMACS10	84
NON-RELATIONAL ML DATA	211

with users at

Takeaways

for your projects

- ☐ Louvain community detection is the bare minimum for any project
- ☐ want to see different metrics on it (modularity, Ncut, NMI, InfoMap) though
- ☐ comparing the performance of Louvain with algorithms available in the literature is a plus
- ☐ a very good project would implement an algorithm, e.g., overlapping Louvain/InfoMap/NMI or BigCLAM/MM-SBM

Correlation networks

a few insights

How can you correlate data?

an overview

Cosine similarity

positive for positive valued data ***x*** and ***y***

$$\cos(\boldsymbol{x},\boldsymbol{y}) = \frac{\boldsymbol{x}^T\boldsymbol{y}}{|\boldsymbol{x}|_2\ |\boldsymbol{y}|_2}$$

norm 2

Pearson correlation coefficient

always with a sign

$$\mathrm{r}(\boldsymbol{x},\boldsymbol{y}) = \frac{(\boldsymbol{x}-m_x)^T\,(\boldsymbol{y}-m_y)}{\sigma_x\,\sigma_y}$$

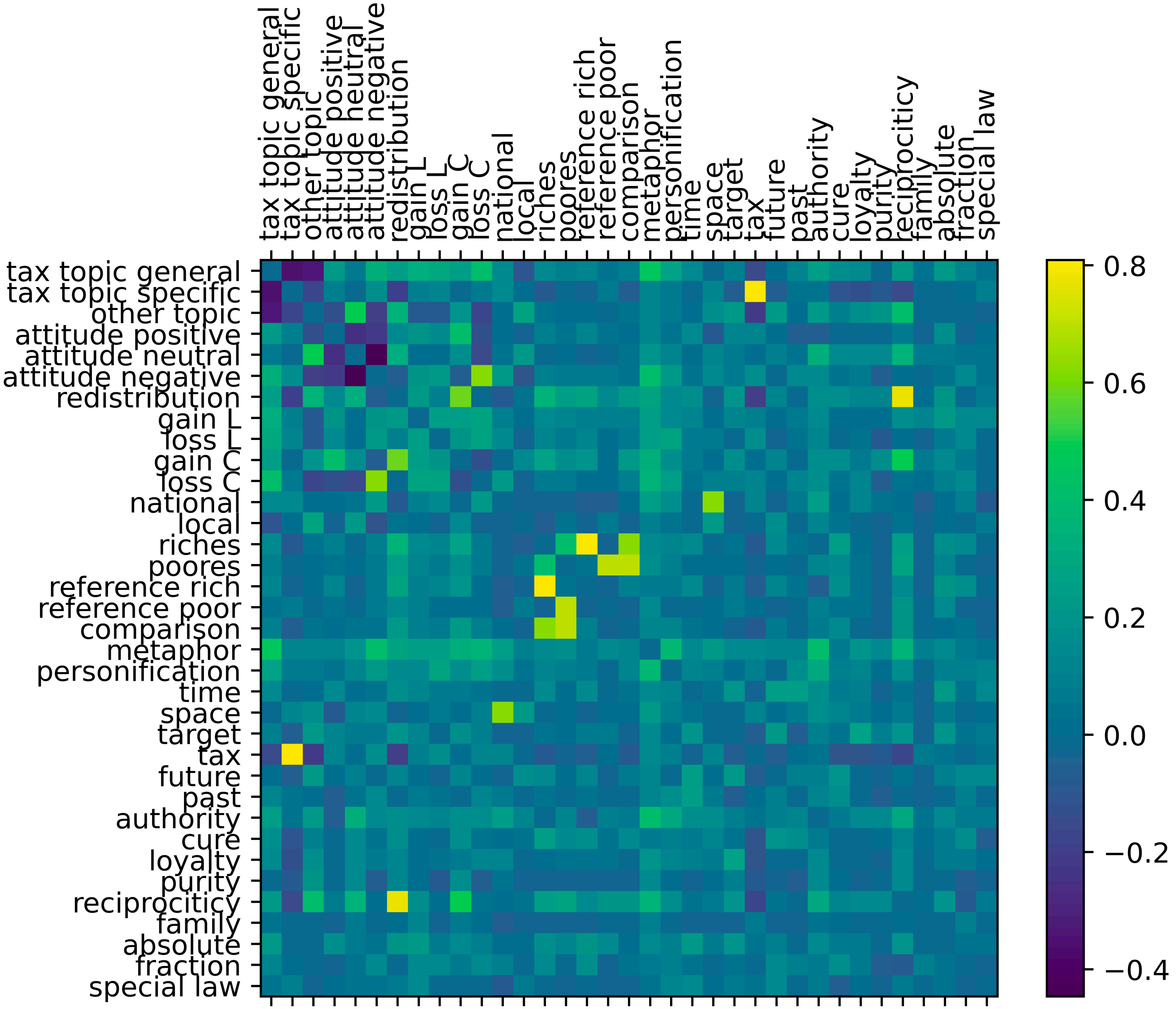
Sørensen-Dice coefficient

$$\mathrm{dice}(\boldsymbol{x},\boldsymbol{y}) = \frac{\boldsymbol{x}^T\boldsymbol{y}}{½|\boldsymbol{x}|_1 + ½|\boldsymbol{y}|_1}$$

for binary data
(it is an F1 score)
always positive

Tax questionnaire example

Pearson correlation used



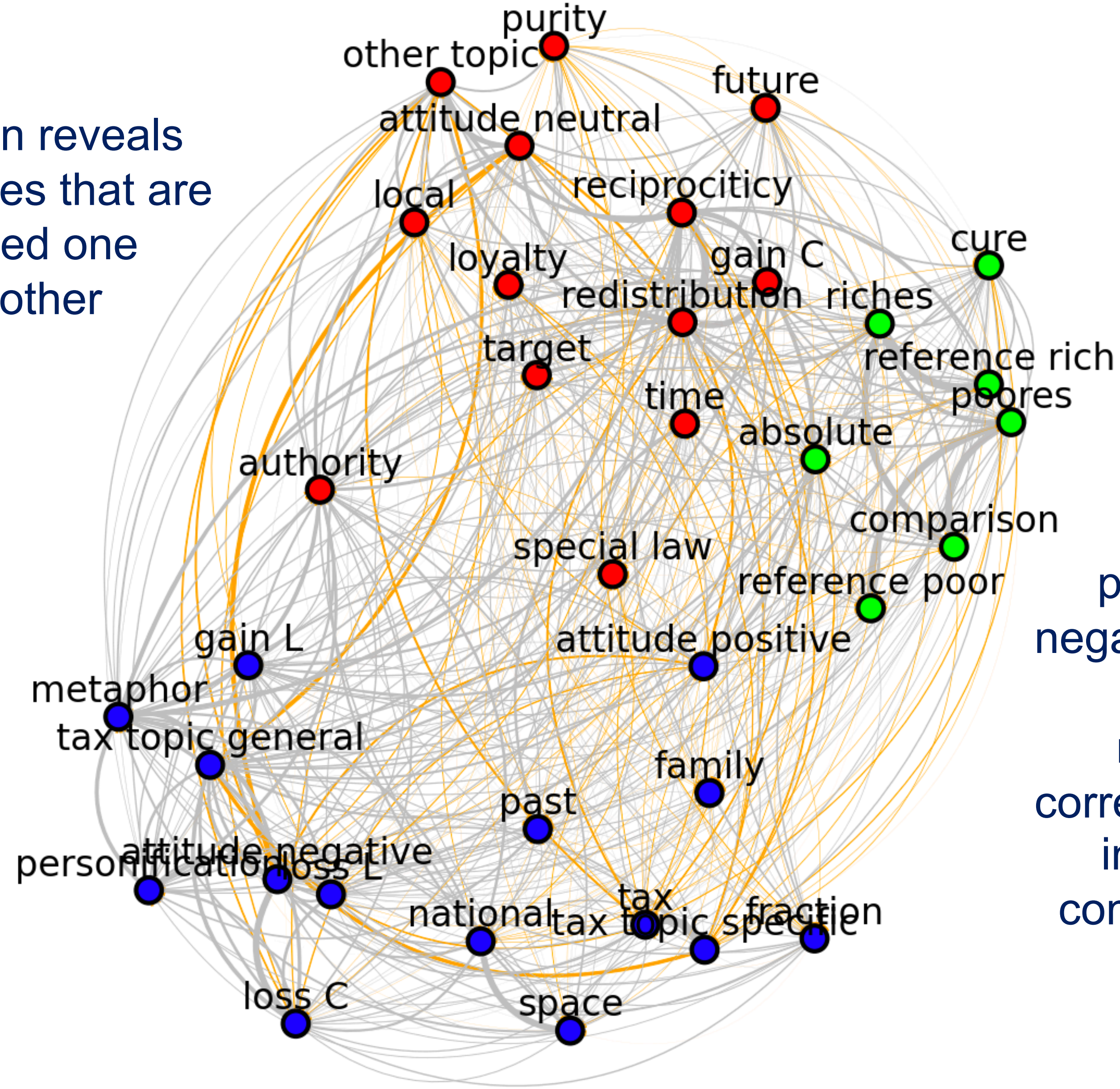
x and **y** data are values (binary or not) for different categories among a wide set of people interviewed

(can do the same with posts and sentiment analysis over the posts)

Tax questionnaire example

signed (and soft) Louvain community detection

Louvain reveals categories that are related one another

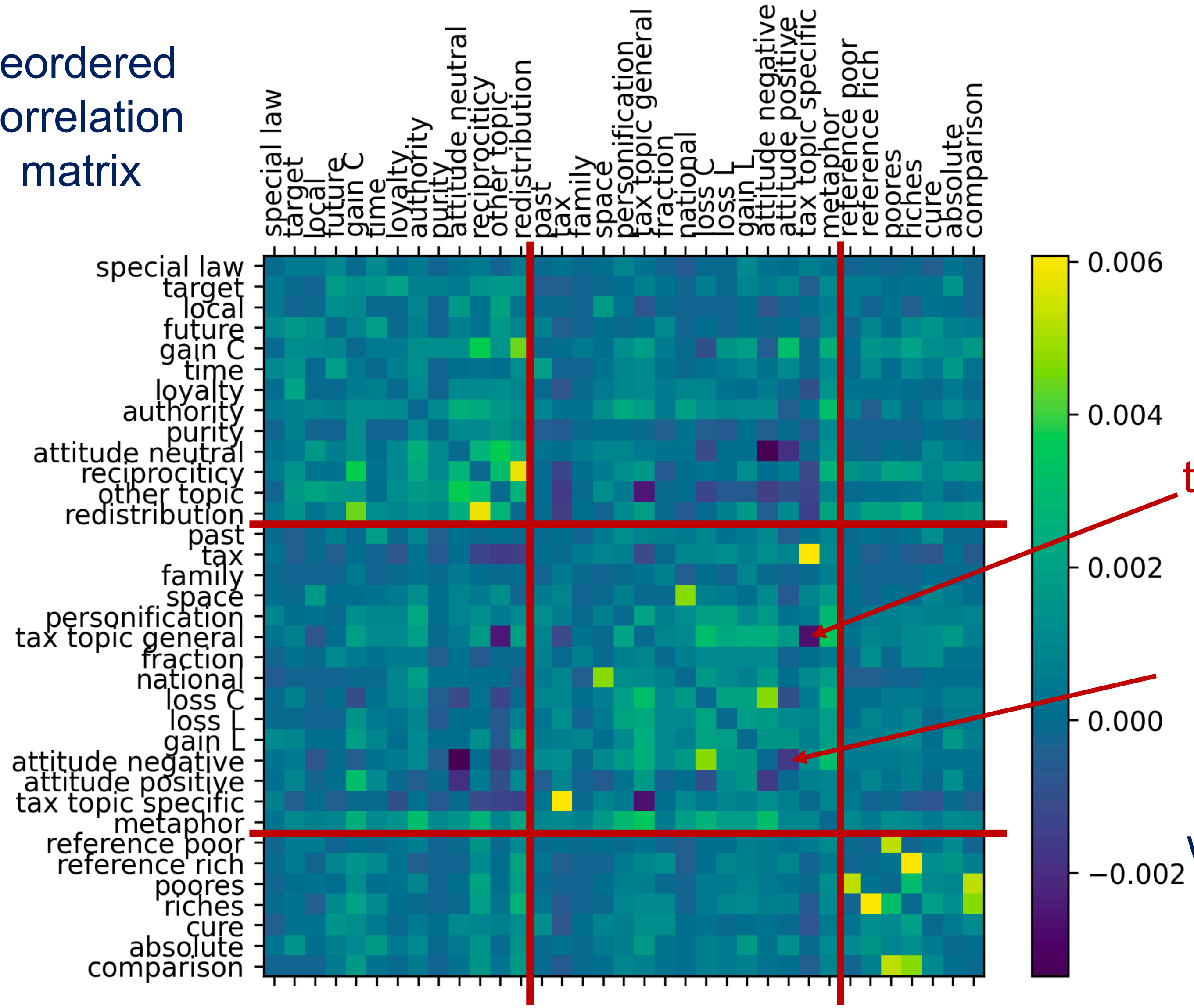


positive and negative attitudes, although negatively correlated, appear in the same community !?!?!

Tax questionnaire example

how Louvain solves correlation inconsistencies

reordered correlation matrix



the two tax topics do not correlate!!!

positive and negative attitudes do not correlate!!!

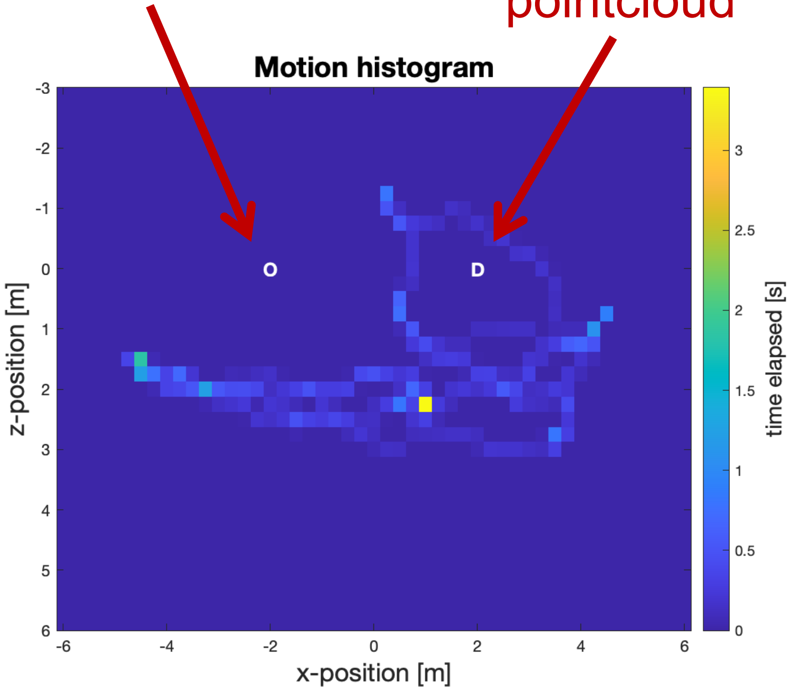
but both correlate with the rest of the community

Motion patterns in VR example

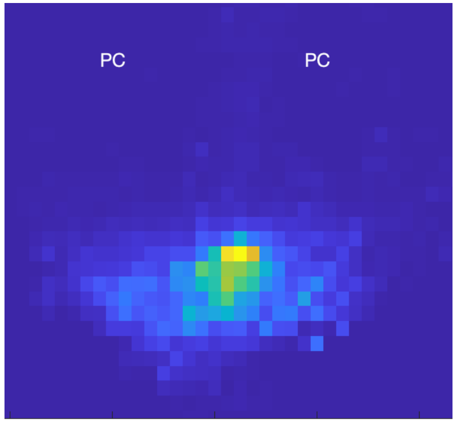
studying immersive environments

original pointcloud

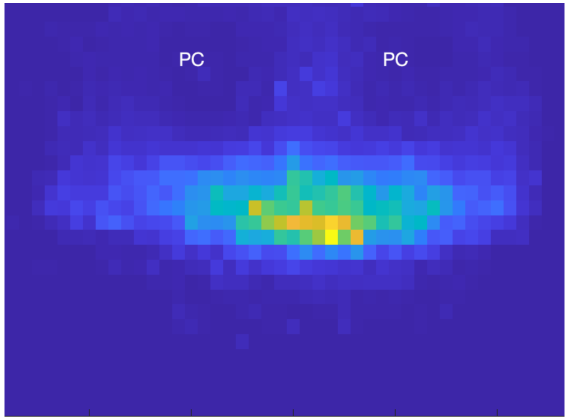
distorted pointcloud

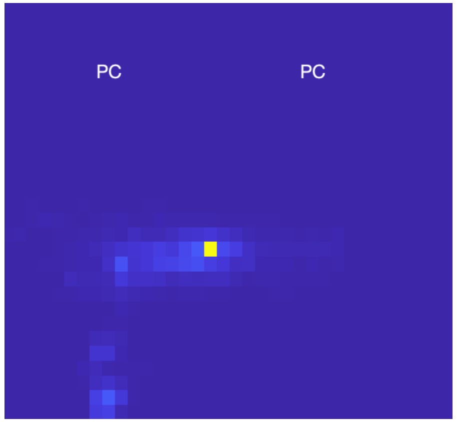


Cluster 1: walking from a distance



Cluster 2: walking closely



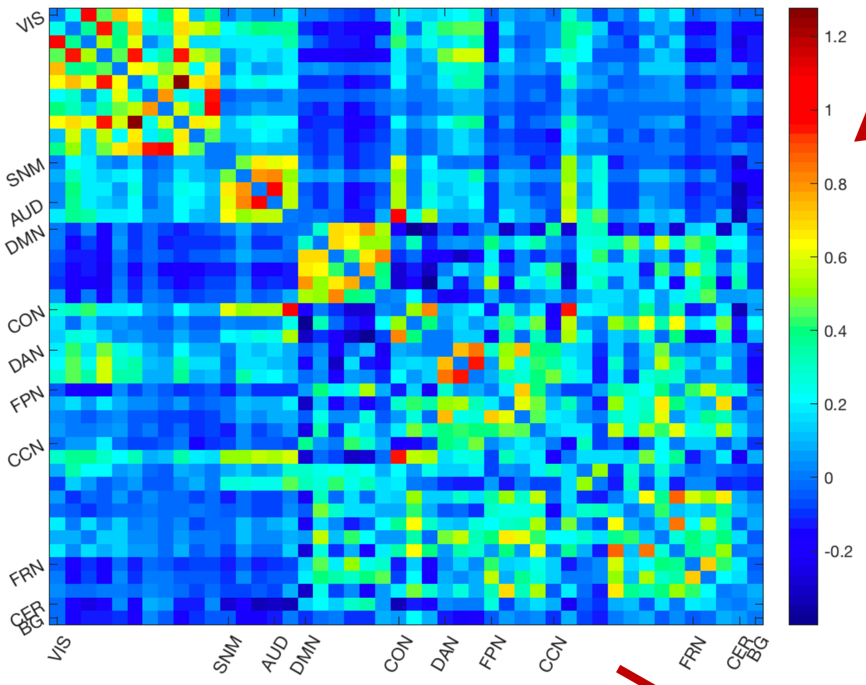


Cluster 3: standing still

motion behaviours detected by Louvain on Pearson correlations over (filtered) motion patterns

fMRI data example

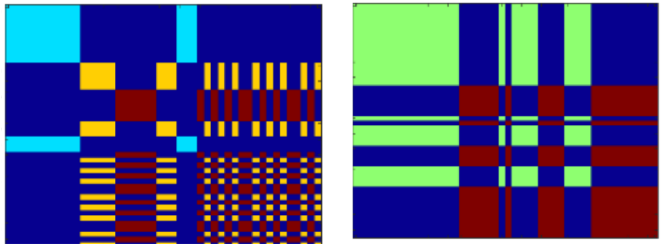
fMRI = functional magnetic resonance imaging



Pearson's correlation coefficient

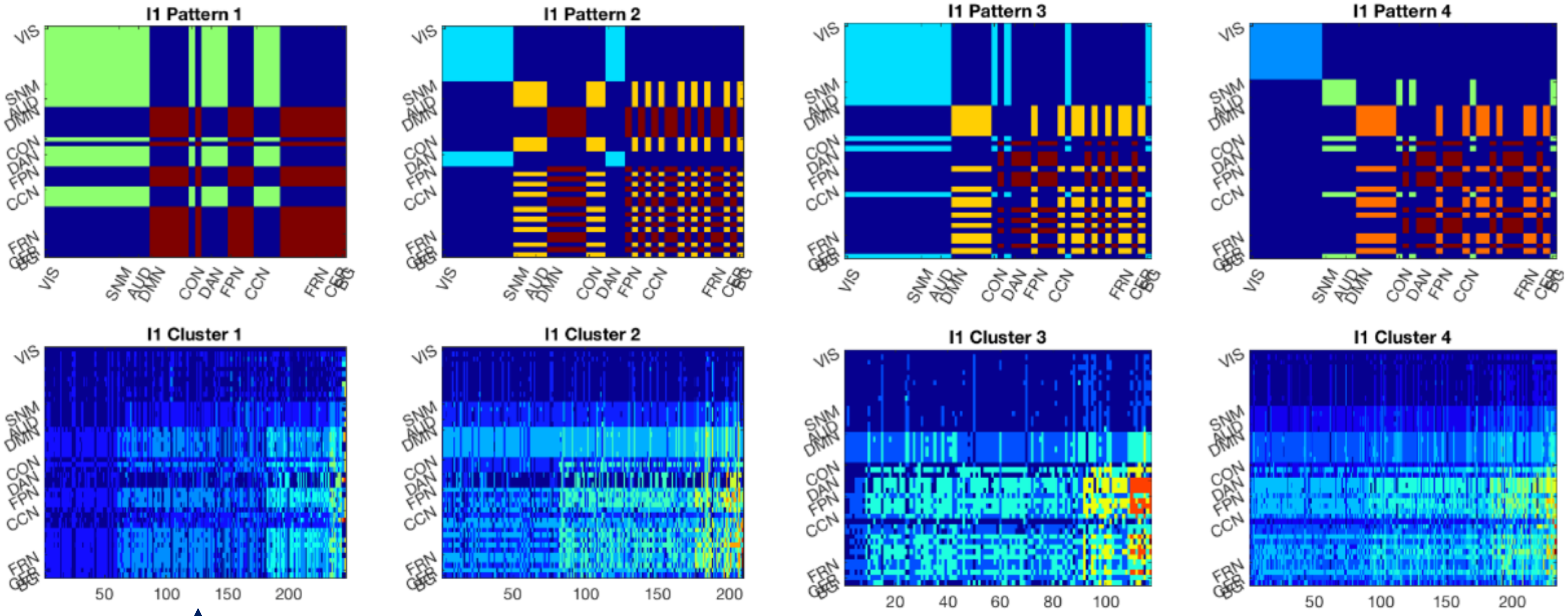
but be aware that the data waveforms, prior to correlation, are highly polished (e.g., from motion-related artifacts and physiological noise fluctuations, multiple-echoes, etc.)

with Louvain we can identify community patterns $\boldsymbol{P} = \boldsymbol{C}^T \boldsymbol{C}$ *whose similarity can be captured by the Dice coefficient*



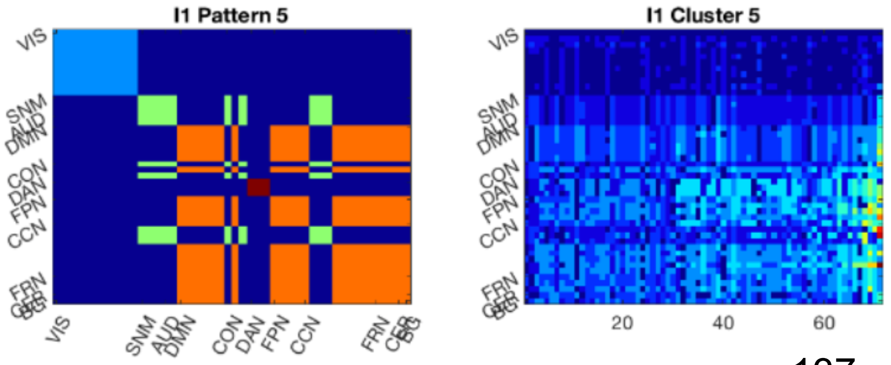
fMRI data example

Louvain communities for community patterns = behaviours



community assignments in cluster 1 (one per column)

average representative of cluster 5 (by consensus clustering)



Partial correlation

to remove counfounding contributions

partial correlation measures the degree of association between two random variables, with the effect of a set of controlling random variables removed

when determining the numerical relationship between two variables of interest, using their correlation coefficient will give misleading results if there is another confounding variable that is numerically related to both variables of interest

$$\text{partial}(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{e}_x{}^T \boldsymbol{e}_y}{|\boldsymbol{e}_x|_2 \, |\boldsymbol{e}_y|_2}$$

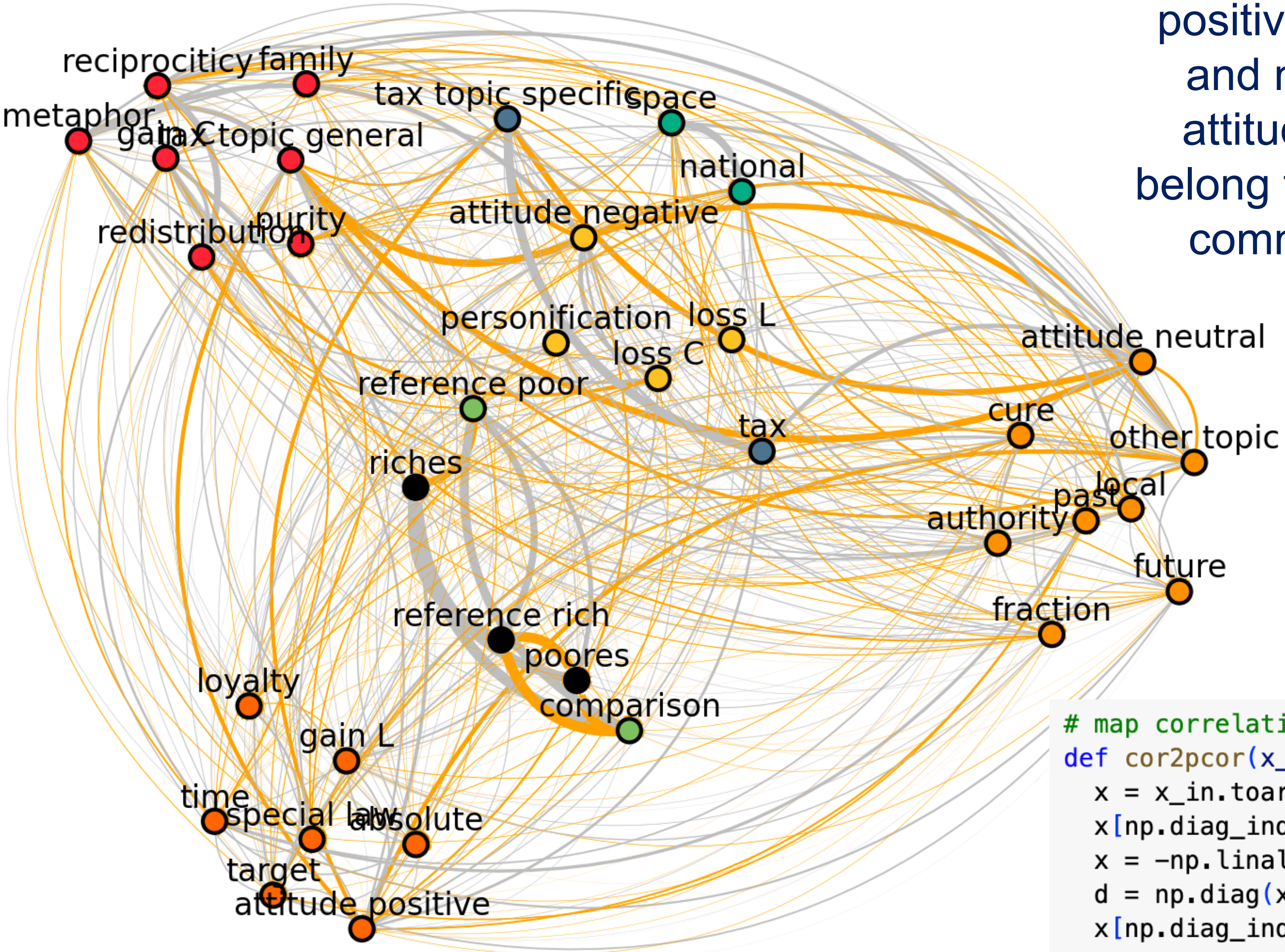
collection of data vectors, other than ***x*** and ***y***, plus the constant vector ***1***

$$\boldsymbol{e}_x = (\underbrace{\boldsymbol{I} - \boldsymbol{Z}(\boldsymbol{Z}^T\boldsymbol{Z})^{-1}\boldsymbol{Z}^T}_{\text{projection on the space orthogonal to span}(\boldsymbol{Z})})\boldsymbol{x}$$

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Tax questionnaire example

with partial correlation

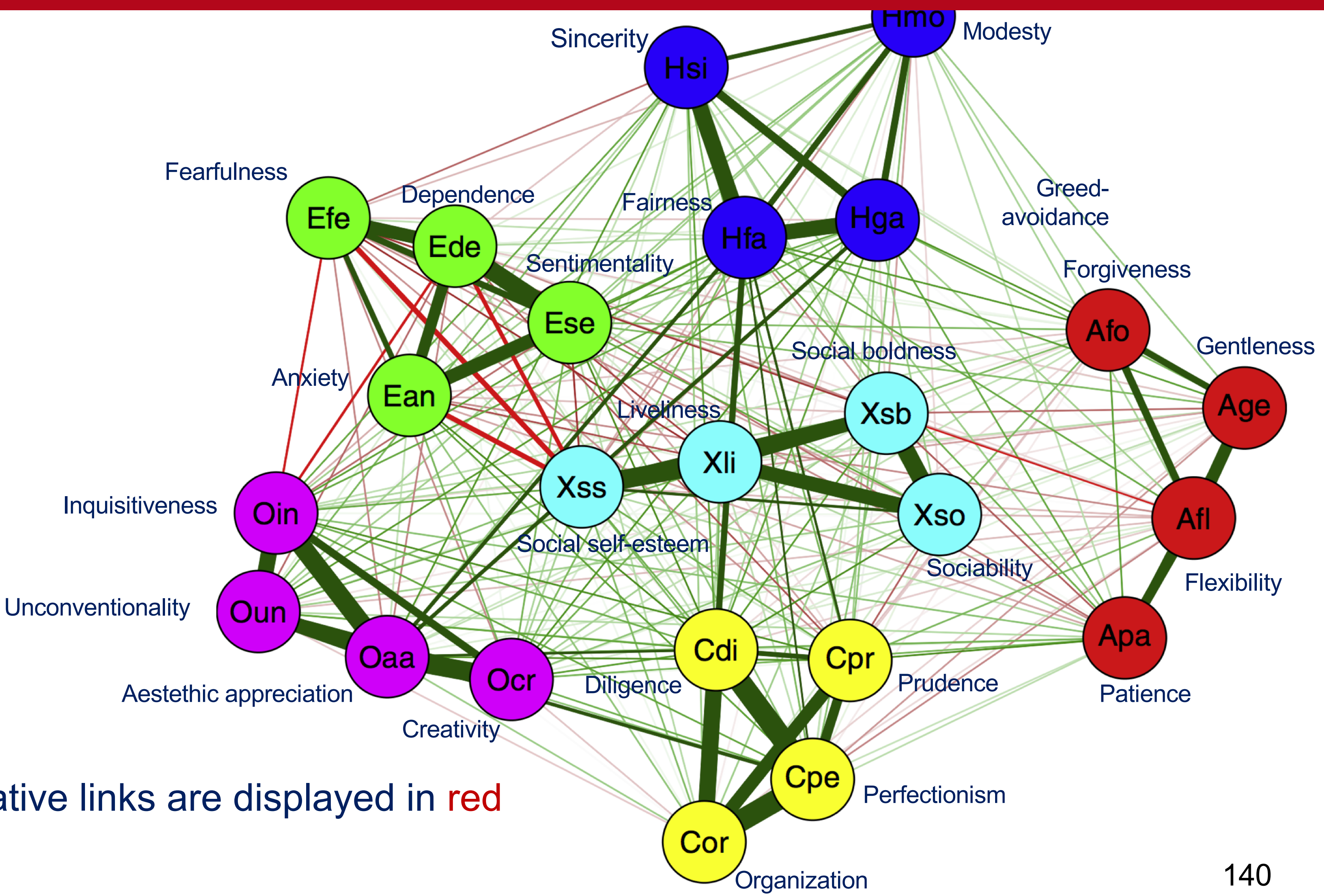


positive, neutral and negative attitudes now belong to different communities

```
# map correlation into partial correlation
def cor2pcor(x_in):
  x = x_in.toarray()
  x[np.diag_indices_from(x)] = 1
  x = -np.linalg.pinv(x)
  d = np.diag(x)
  x[np.diag_indices_from(x)] = -d
  x = x/np.sqrt(d)
  x = x.T/np.sqrt(d)
  return x
```

A personality network example

Costantini et al. (2015)



Negative links are displayed in red