Linear Clustering Process on Networks

MSc Thesis Defense

Presenter: Beichen Wang

Thesis advisor: Prof. Piet Van Mieghem

Daily supervisor: Dr. Ivan Jokić

29/08/2023

TU Delft

NAS

Content

Linear Clustering Process

- LCP with known number of clusters
- Non-back tracking LCP

Other involved clustering algorithms

- Modularity-based
- Spectral method
- Local Dominance

Benchmarks & Performance metrics

Simulation results

- Stochastic Block Model (SBM)
- LFR benchmark
- Random network models (ER, BA, WS)
- Real-world networks

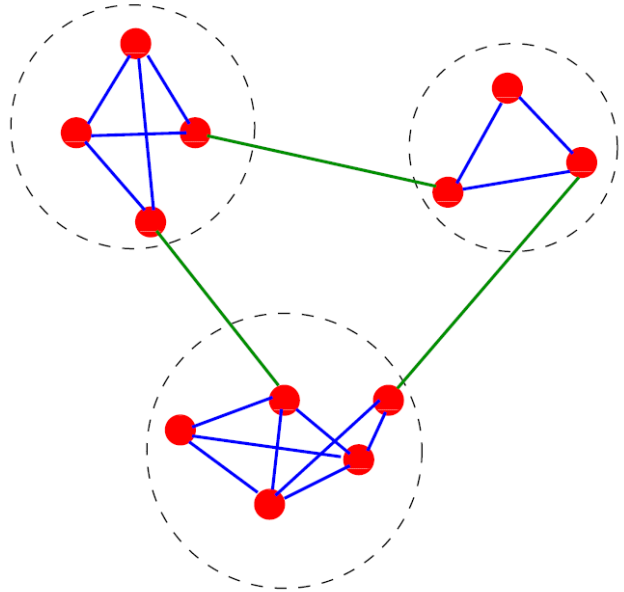
TU Delft

Network Clustering

- Network clustering represents “organization of nodes in clusters, with many links joining nodes of the same cluster and comparatively few links joining nodes of different clusters” [1].

- Widely accepted quality function for a given network partition is modularity index m.

$$m = \frac{1}{2L} \cdot \sum_{i=1}^{N} \sum_{j=1}^{N} \left(a_{ij} - \frac{d_i \cdot d_j}{2L} \right) \cdot \mathbf{1}_{\{i \text{ and } j \in \text{ same cluster}\}}$$



A simple graph with three communities, from [1]

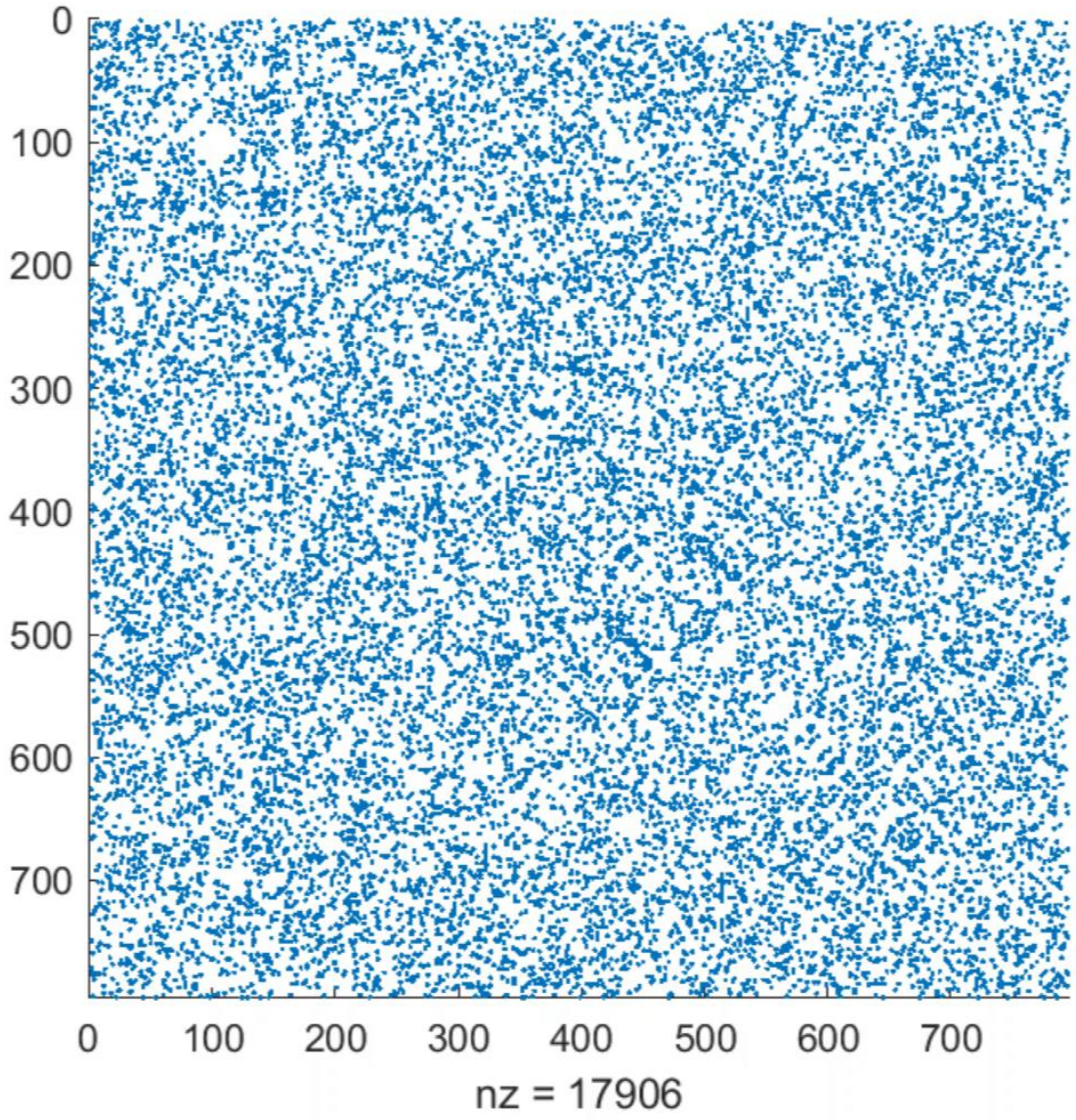
A Adjacency matrix
L Number of links
d_i Node i degree
m Modularity index

[1] Fortunato, S., 2010. Community detection in graphs. Physics reports, 486(3-5), pp.75-174.

Linear Clustering Process (LCP)

Node-level governing equation

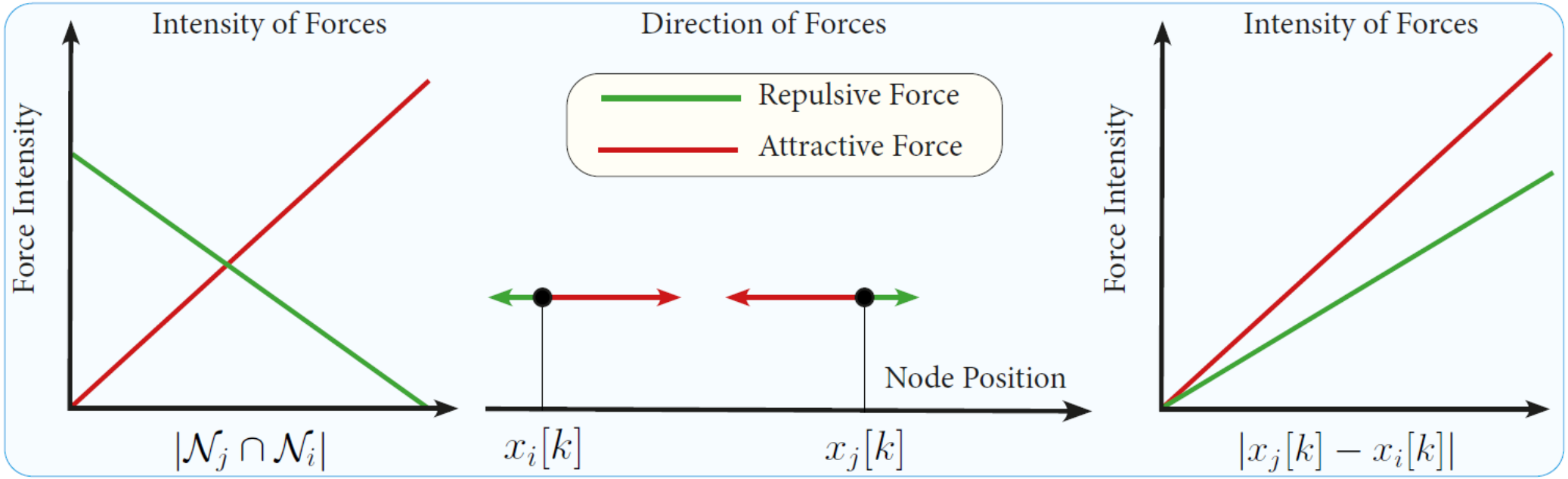
- Each node i in the graph G is assigned a position $x_i[k]$ on a line at discrete time k.
- LCP consists of two opposite and simultaneous forces that change nodal position in time
  - **Attraction**: Adjacent nodes that share many common neighbours (i.e. triangles!) are mutually attracted.
  - **Repulsion**: Two adjacent nodes are repulsed with a force proportional to the number of neighbours they do not share.



TU Delft

Linear Clustering Process (LCP)

Node-level governing equation



$$d_i = |\mathcal{N}_i \setminus \mathcal{N}_j| + |\mathcal{N}_i \cap \mathcal{N}_j|$$

$$x_i[k+1] = x_i[k] + \sum_{j \in \mathcal{N}_i} \left(\frac{\alpha \cdot (|\mathcal{N}_j \cap \mathcal{N}_i| + 1)}{d_j d_i} - \frac{\frac{1}{2} \cdot \delta \cdot (|\mathcal{N}_j \setminus \mathcal{N}_i| + |\mathcal{N}_i \setminus \mathcal{N}_j| - 2)}{d_j d_i} \right) \cdot \left(x_j[k] - x_i[k] \right)$$

k	Time
$x_i[k]$	Node i position in time k
α	Attraction strength
δ	Repulsive strength
$\mathcal{N}_i$	Node i neighbours

The attractive force between two adjacent nodes is always of higher strength than the repulsive force, preserving the system's stability but negatively influencing the steady-state. Although the repulsive force is a simplified version of the Newtonian repulsive force, the LCP is proportional to the graph topology, which helps revealing the clusters.

TU Delft

Linear Clustering Process (LCP)

Network-level governing equation

$$x_i[k+1] = x_i[k] + \sum_{j \in \mathcal{N}_i} \left(\frac{\alpha \cdot (|\mathcal{N}_j \cap \mathcal{N}_i| + 1)}{d_j d_i} - \frac{\frac{1}{2} \cdot \delta \cdot (|\mathcal{N}_j \setminus \mathcal{N}_i| + |\mathcal{N}_i \setminus \mathcal{N}_j| - 2)}{d_j d_i} \right) \cdot \left(x_j[k] - x_i[k] \right)$$

Node-level

Network-level

$$x[k+1] = (I + W - diag(W \cdot u)) \cdot x[k]$$

$$W = (\alpha + \delta)\, \Delta^{-1} \cdot \left(A \circ A^2 + A \right) \cdot \Delta^{-1} - \frac{1}{2} \cdot \delta \left(\Delta^{-1} \cdot A + A \cdot \Delta^{-1} \right)$$

$x[k]$ node position vector

A adjacency matrix

I identity matrix

Δ Degree matrix

$\circ$ Hadamard product

TU Delft

Time-dependence of the LCP

If all nodes share the same position, the LCP is in a trivial steady state. Thus, the all-one vector u is a trivial solution for the steady-state of LCP.

$$x[k] = (I + W - \mathrm{diag}\,(W \cdot u))^k\, x[0]$$

$$W \cdot u - \mathrm{diag}\,(W \cdot u) \cdot u = 0$$

Instead, the hierarchical structure of the network emerge in the time evolution of LCP.

$$W - \mathrm{diag}\,(W \cdot u) = Y \mathrm{diag}(\beta) Y^T$$

A scaled and shifted nodal position vector tends to the eigenvector y_2 with an exponentially decreasing error in time k. Thus, we estimate the clusters based on y_2.

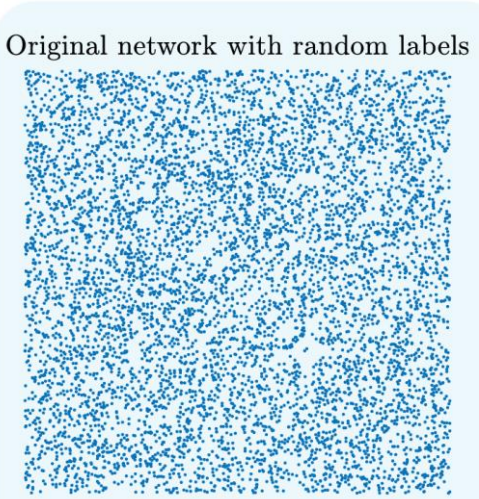
$$x[k] - \frac{u^T x[0]}{\sqrt{N}} u = \sum_{j=2}^{N} (1+\beta_j)^k (y_j^T x[0])\; y_j$$

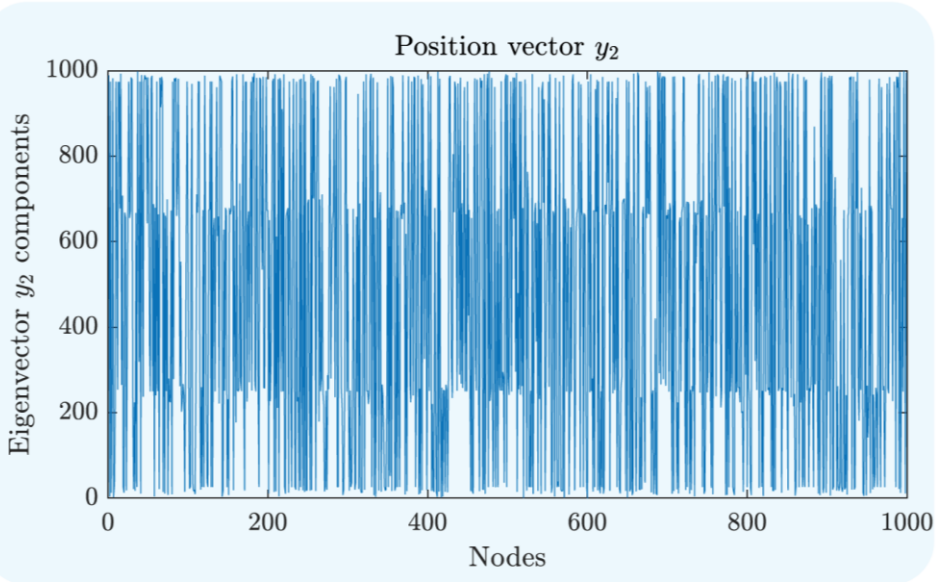
$$\frac{x[k] - \frac{u^T x[0]}{\sqrt{N}} u}{(1+\beta_2)^k\; (y_2^T x[0])} = y_2 + O\left(\frac{1+\beta_3}{1+\beta_2}\right)^k$$

TU Delft

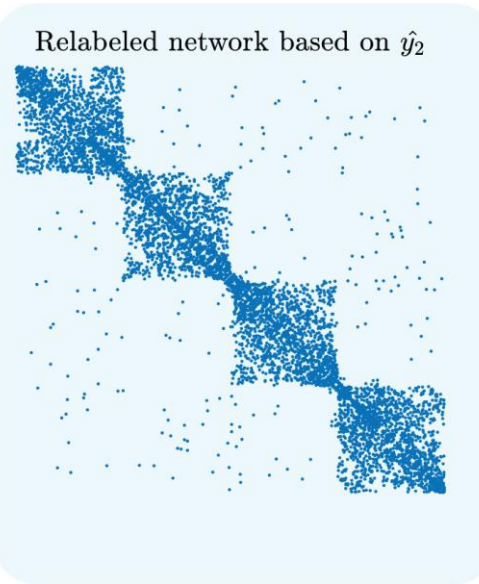
Cluster estimation based on the eigenvector y_2

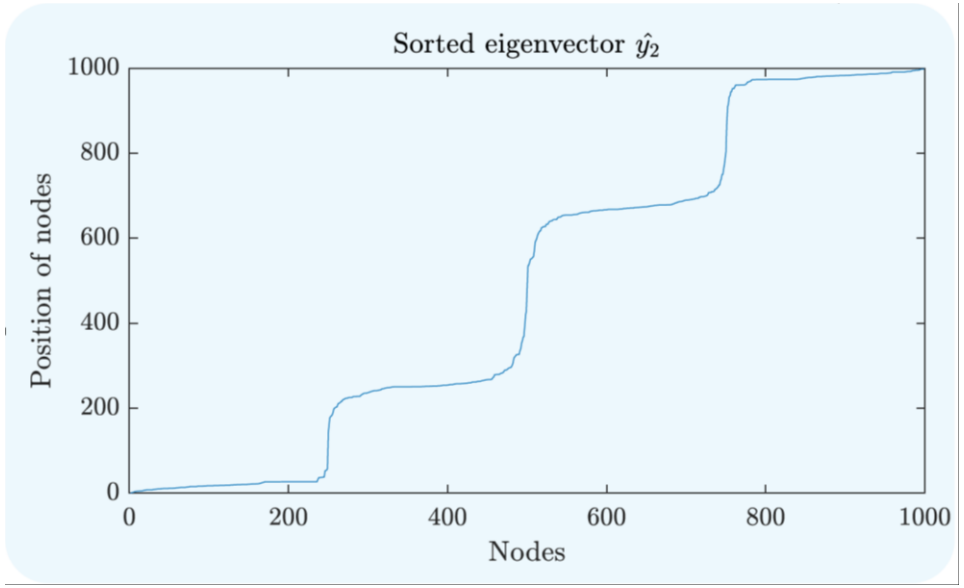
To estimate the clusters based on the eigenvector y_2, we need to detect the borders of the clusters in the sorted vector y_2.





We determine the borders of clusters by optimizing recursively the modularity index m.

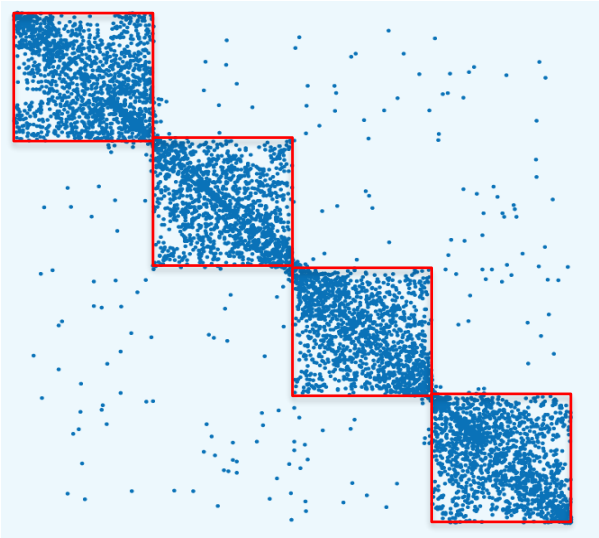




TU Delft

Modularity index m optimization

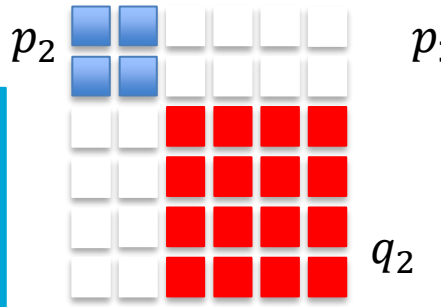
Based on the sorted eigenvector y_2, we define the permutation matrix R and transform the adjacency matrix A and the degree vector d.

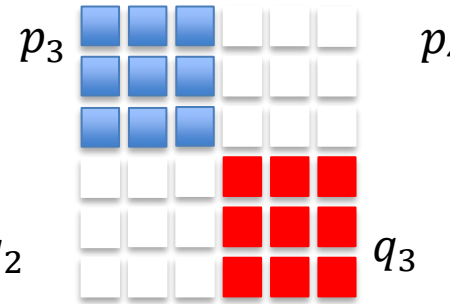


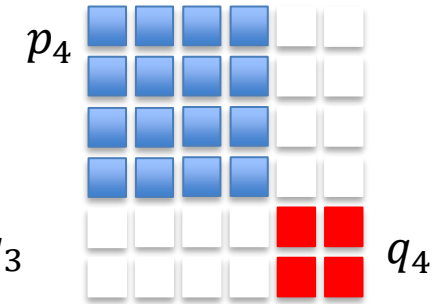
$$\begin{cases} \hat{A} &= R^T \cdot A \cdot R \\ \hat{d} &= R \cdot d \\ \hat{y_2} &= R \cdot y_2 \end{cases}$$

$$m = \frac{1}{2L} \cdot \sum_{i=1}^{c} \hat{e_i}^T \cdot \left(\hat{A} - \frac{1}{2L} \cdot \left(\hat{d} \cdot \hat{d}^T \right) \right) \cdot \hat{e_i}$$

Transformed relation defines the modularity index m as a sum of the elements of the matrix within brackets, corresponding to clusters. Such definition allows for a simple recursive optimization.





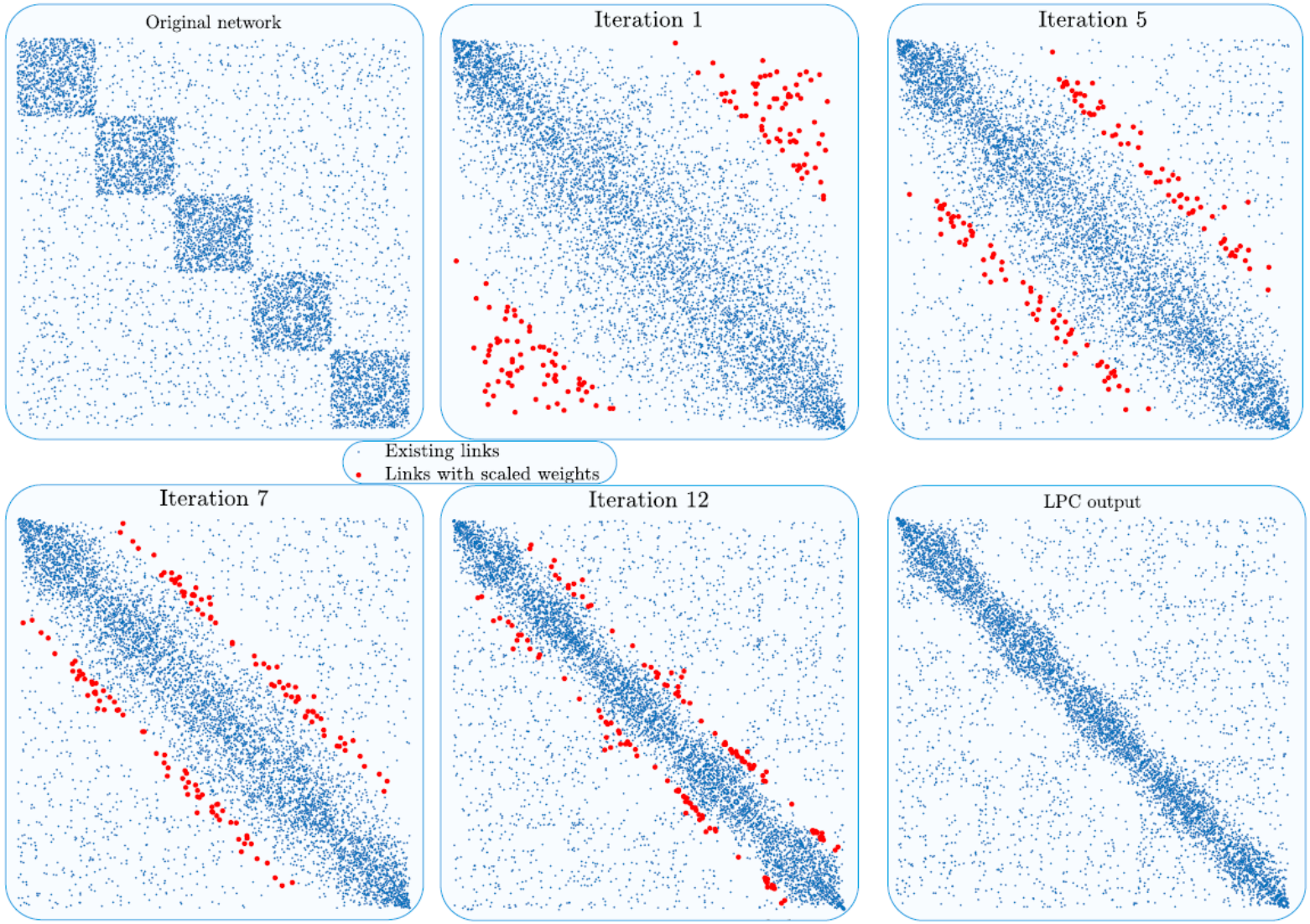


TU Delft

Scaling the weights of inter-community links

we reduce the importance of links between nodes from different clusters, based on the partition from previous iteration.

Links that connect nodes with the highest ranking difference are most likely inter-community links.



TU Delft

Two derived variants of the proposed LCP

LCP for a known number of communities: incorporating the real number of clusters as an input parameter. Merging adjacent clusters and minimizing the negative impact on modularity to achieve the target number of clusters. This method aims to enhance the precision and quality of community partition. This variant is denoted as LCP_c.

Non-backtracking variant of LCP: modifying the non-backtracking algorithm by substituting its original matrix with W matrix in the LCP algorithm. This approach is designed to parallel the performance of the original non-backtracking method, providing similar efficiency and results. This variant is denoted as LCP_n.

Existing clustering algorithms

- **Newman** method (2006 PNAS, complexity $O\big((L+N)N\big)$)
- **Louvain** method (2008 JSM, complexity $O(L)$ for sparse networks)
- **Eigengap** method (2009 PRE, complexity $O(N^3)$)
- **Non-backtracking** method (2013 PNAS, complexity $O(N^3)$)
- **Leiden** method (2019 Nature Scientific Reports, $O(L)$ for sparse networks)
- **Local dominance** (2022 arXiv, complexity $O(L)$)
- **Linear Clustering Process** (2023 IEEE TNSE, complexity $O(N \cdot L)$)

TU Delft

Existing clustering algorithms

- **Newman** method (2006 PNAS, complexity $O((L+N)N)$)
- **Louvain** method (2008 JSM, complexity $O(L)$ for sparse networks)
- **Leiden** method (2019 Nature Scientific Reports, $O(L)$ for sparse networks)

Modularity-based

- **Eigengap** method (2009 PRE, complexity $O(N^3)$)
- **Non-backtracking** method (2013 PNAS, complexity $O(N^3)$)

Spectral method

- **Local dominance** (2022 arXiv, complexity $O(L)$)

Hierarchical method

- **Linear Clustering Process** (2023 IEEE TNSE, complexity $O(N \cdot L)$)

Linear process with modularity

Simulations

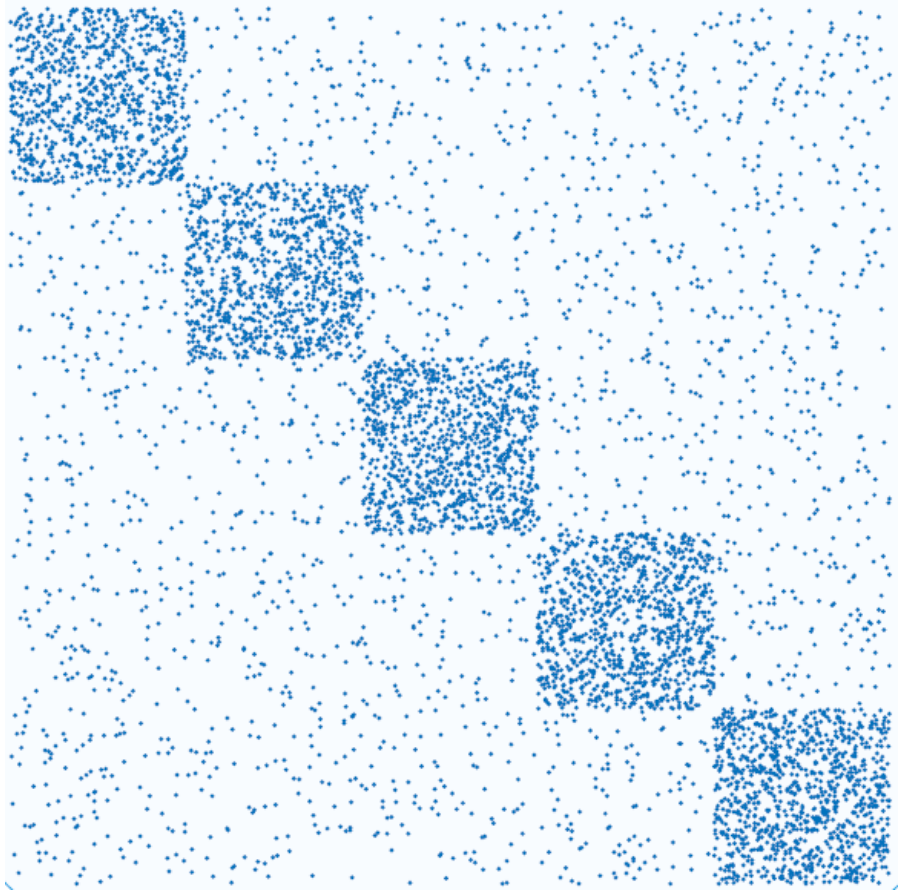
- Benchmarks
  - Stochastic Block Model (SBM)
  - LFR benchmark
  - Random graph models (ER, BA, WS)
  - Real-world networks
- Performance metrics
  - Estimated number of clusters c
  - Modularity m
  - Normalized Mutual Information (NMI)
  - Element-centric Similarity (ECS)

Clustering comparison

TU Delft

Stochastic Block Model (SBM)

- Two nodes are connected via a link with probability p_{in} if they belong to the same cluster, otherwise the direct link exists with probability p_{out}.
- $b_{in} = N \times p_{in}, b_{out} = N \times p_{out}$
- In our simulations, we use symmetric SBM (SSBM), where all clusters in the network have the same size.

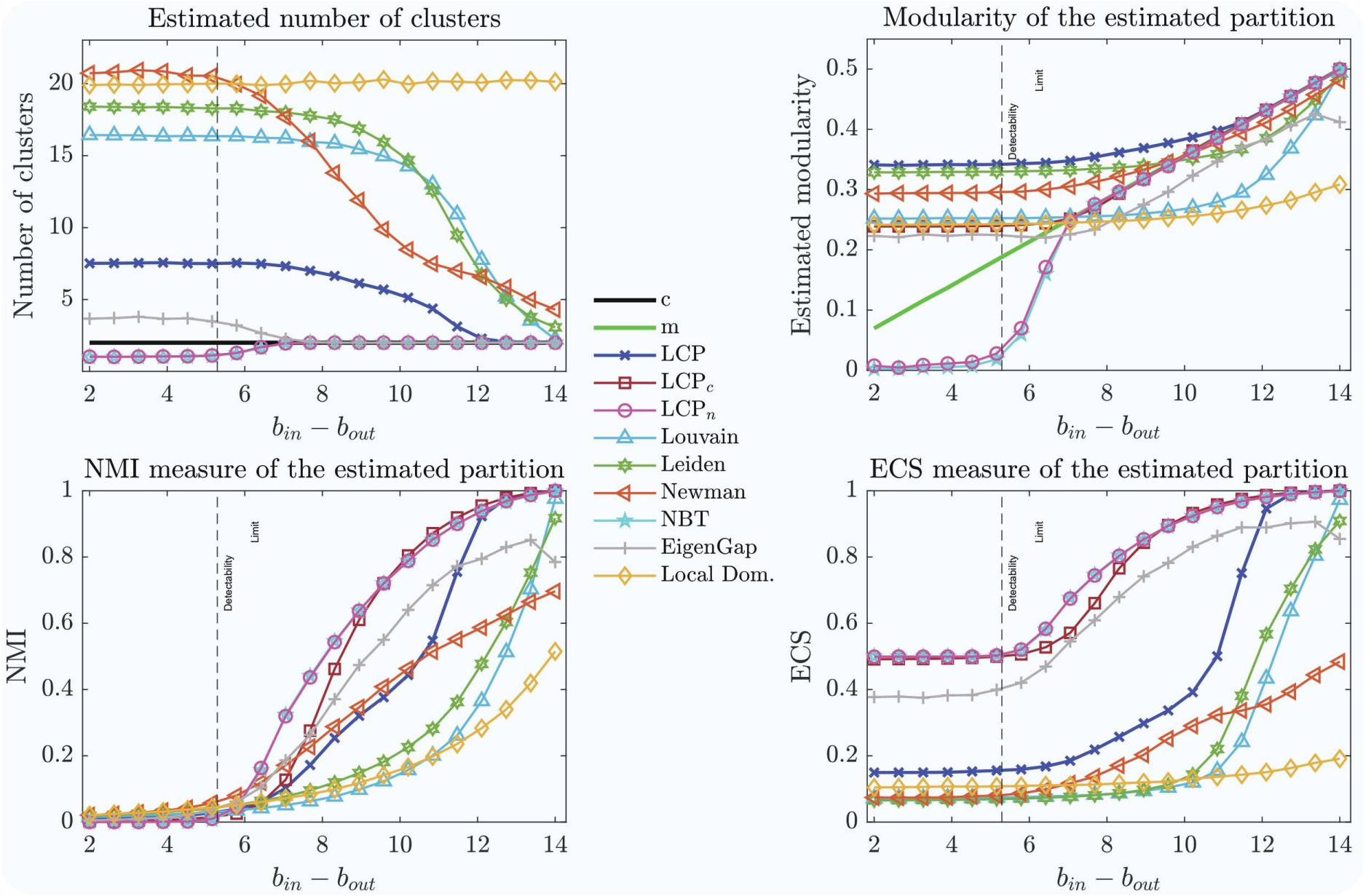


An example of SSBM network with $N = 1000$ nodes and $c = 5$ clusters.

TU Delft

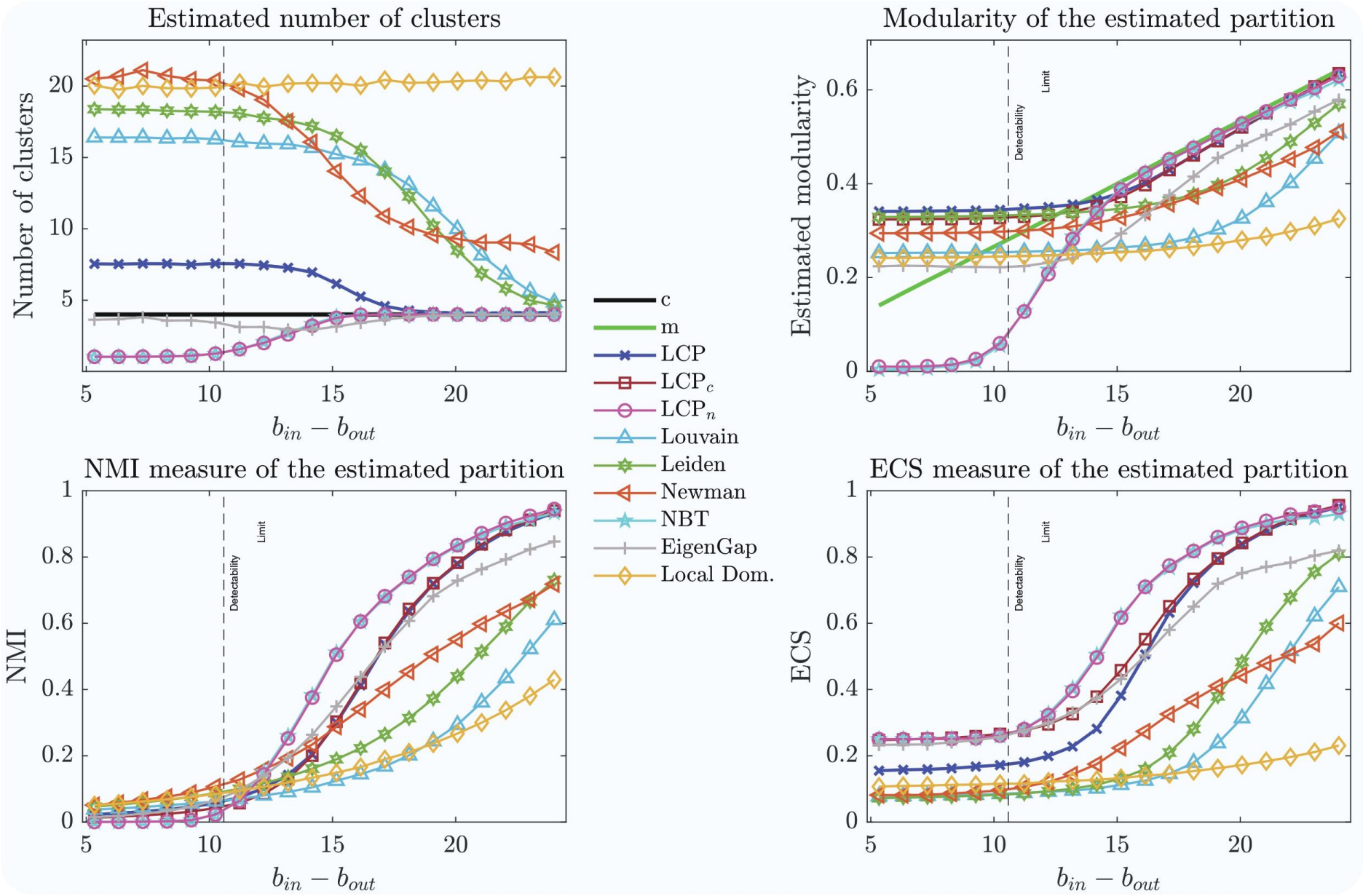
SBM benchmark results

$N = 500, d_{av} = 7, c = 2$



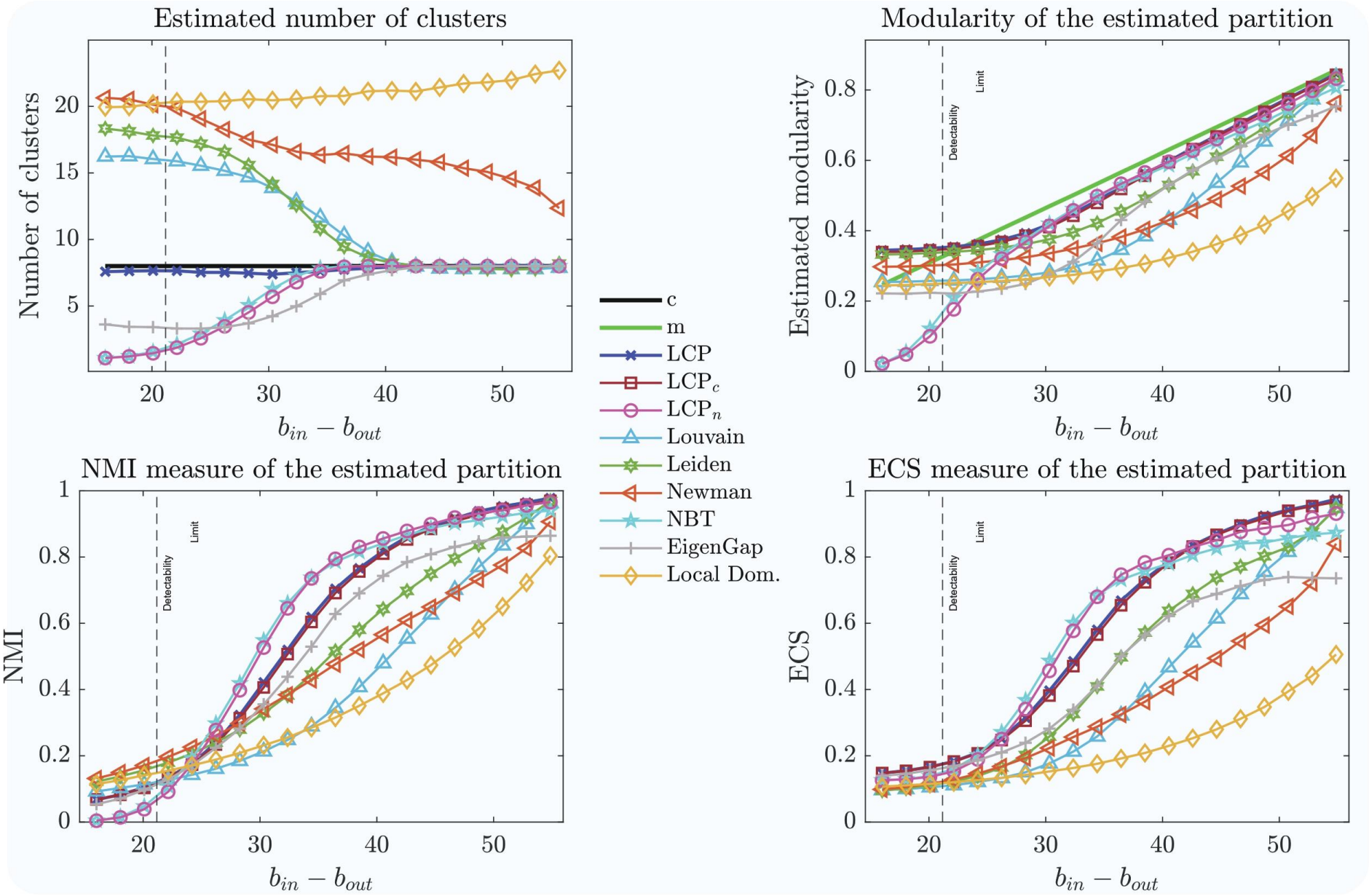
SBM benchmark results

$N = 500, d_{av} = 7, c = 4$



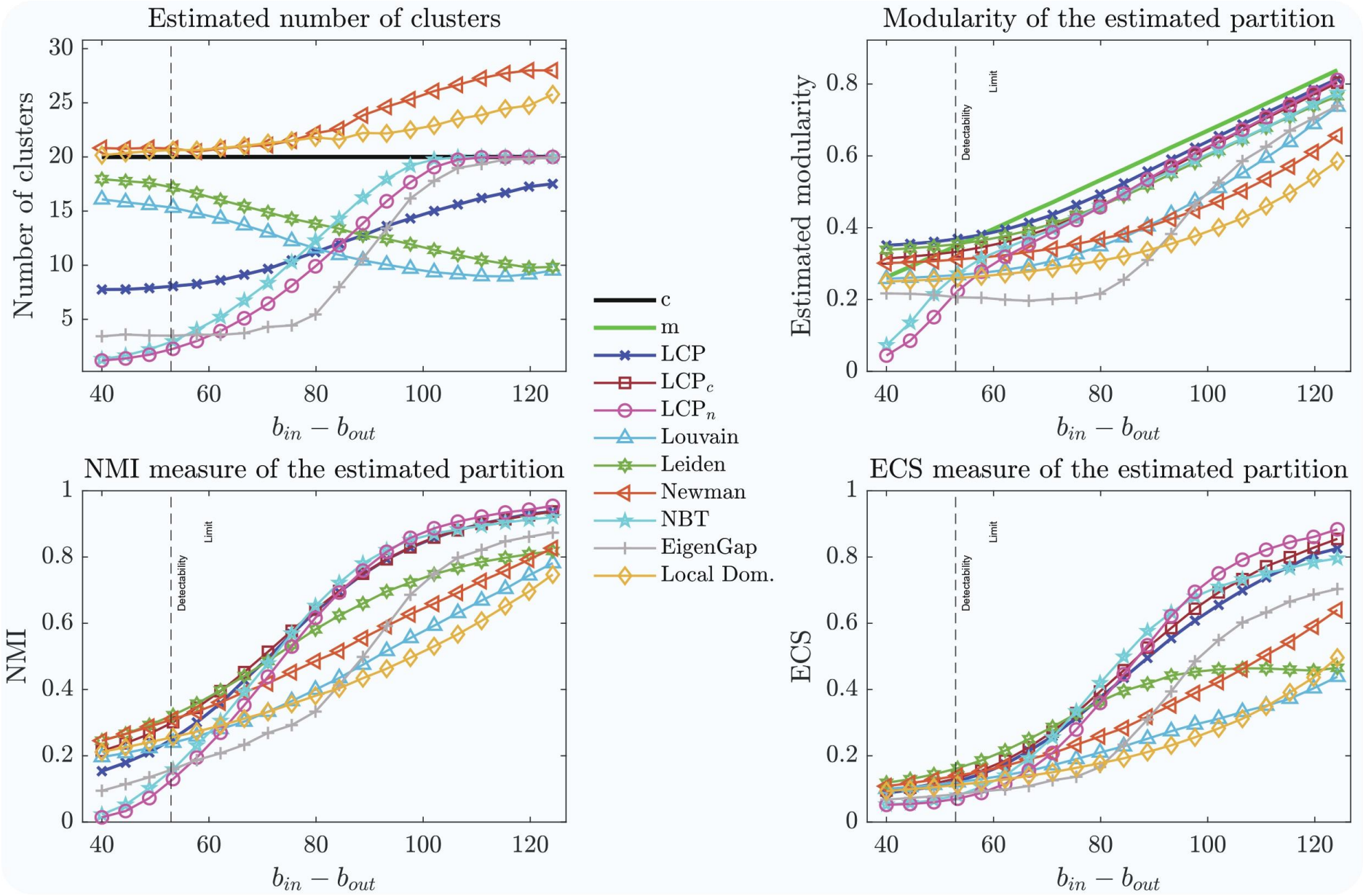
SBM benchmark results

$N = 500, d_{av} = 7, c = 8$



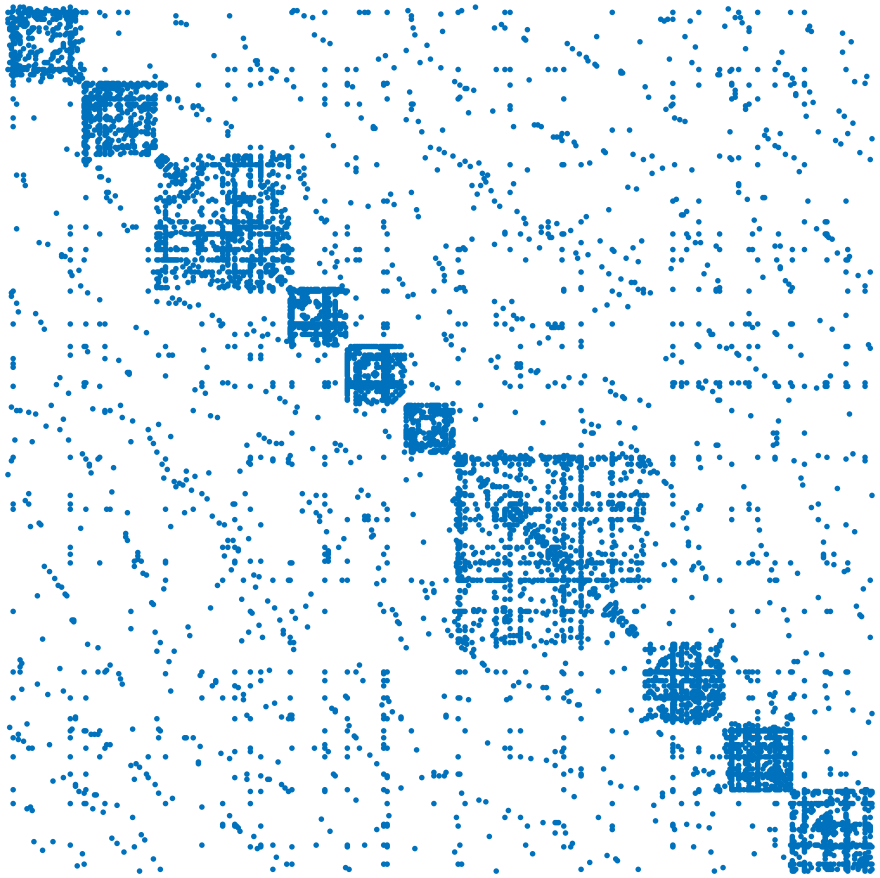
SBM benchmark results

$N = 500, d_{av} = 7, c = 20$



LFR benchmark

- The degree vector d is sampled from a power-law distribution with exponent γ.
- The community size vector n is sampled from a power-law distribution with the exponent β.
- Each node shares $\mu < 1$ ratio of links with nodes from other communities.



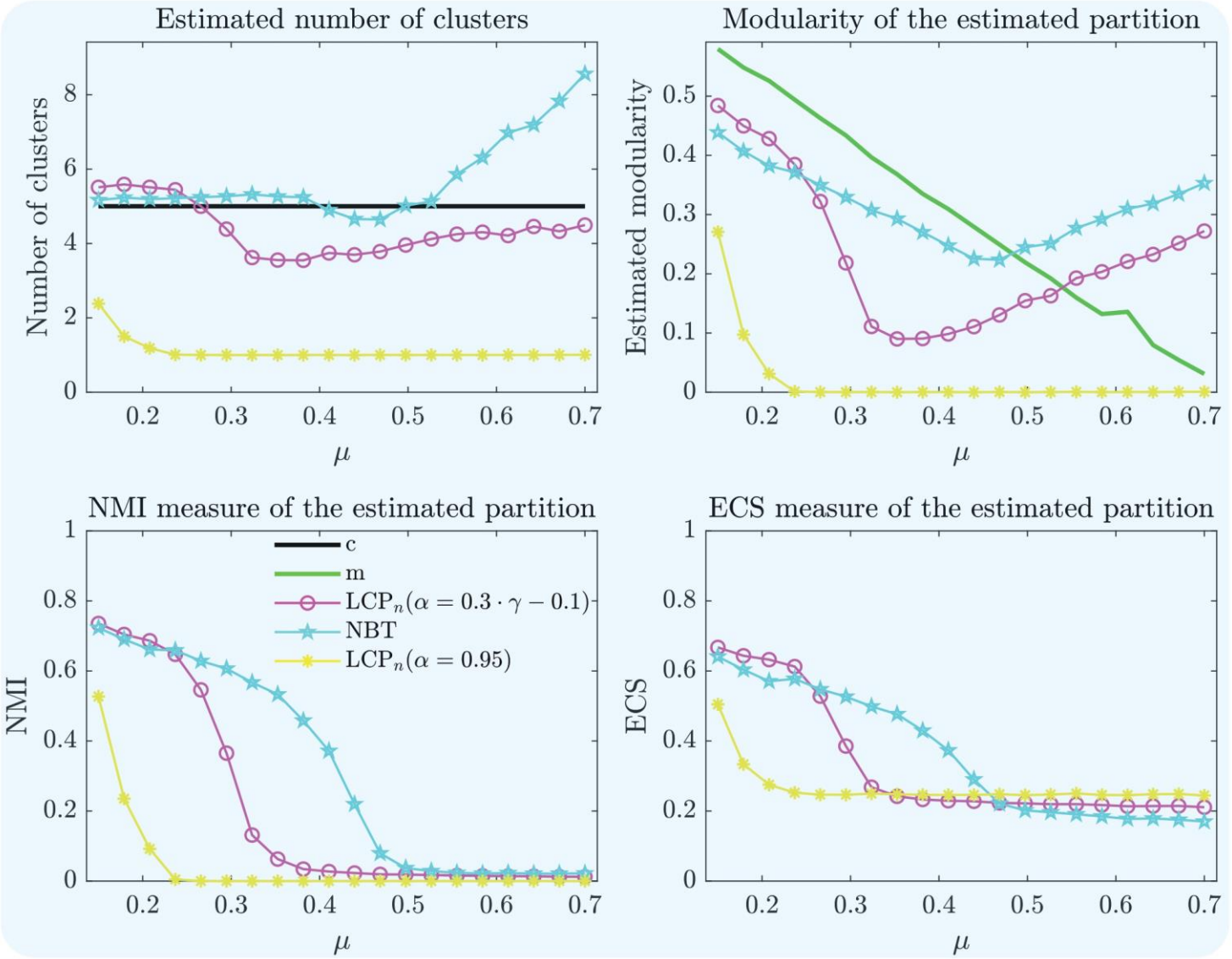
An example of the LFR network with $N = 500$ nodes, $c = 10$ clusters and parameters $\gamma = 2$, $\beta = 3$ and $\mu = 0.25$.

LFR benchmark

$N = 500, d_{av} = 12, c = 5, \gamma = 2, \beta = 3$

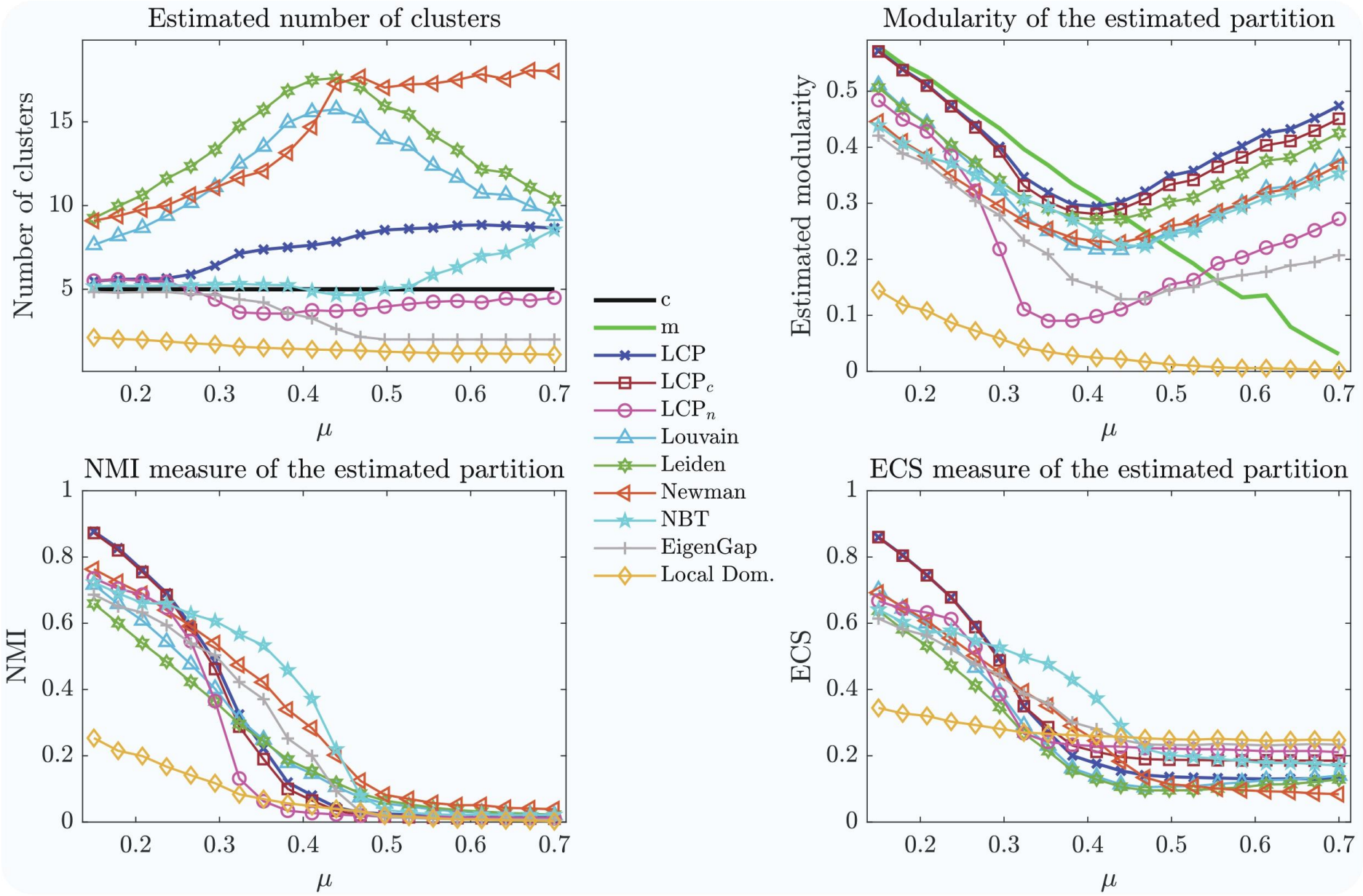
We provide an empirical strategy for tuning α of LCP_n according to the power-law exponent γ of degree distribution of the network.

$$\alpha = \begin{cases} 0.3 \times \gamma - 0.1 & \text{if } 1 < \gamma \leq 3.5 \\ 0.95 & \text{otherwise} \end{cases}$$



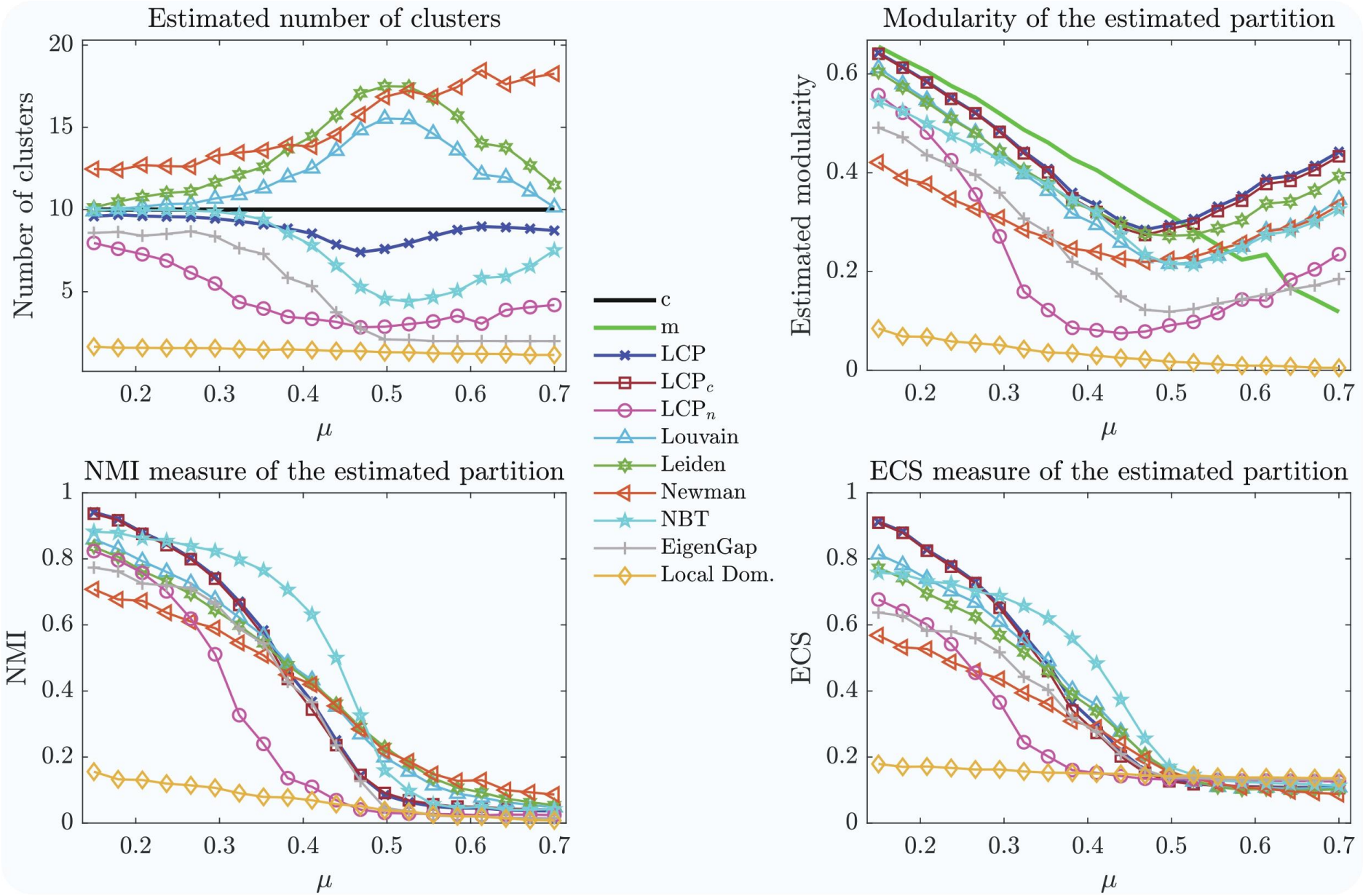
LFR benchmark results

$N = 500, d_{av} = 12, c = 5, \gamma = 2, \beta = 3$



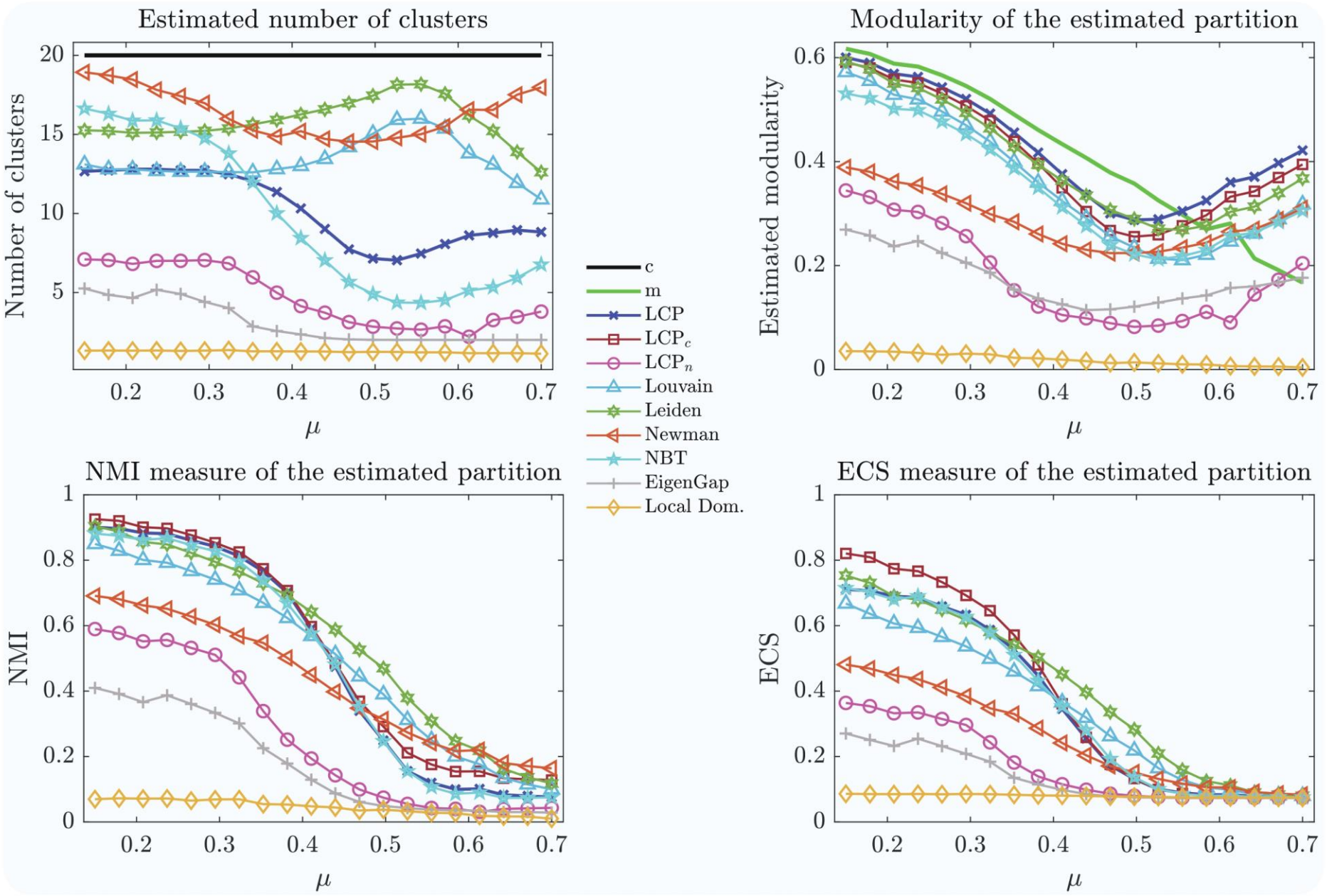
LFR benchmark results

$N = 500, d_{av} = 12, c = 10, \gamma = 2, \beta = 3$



LFR benchmark results

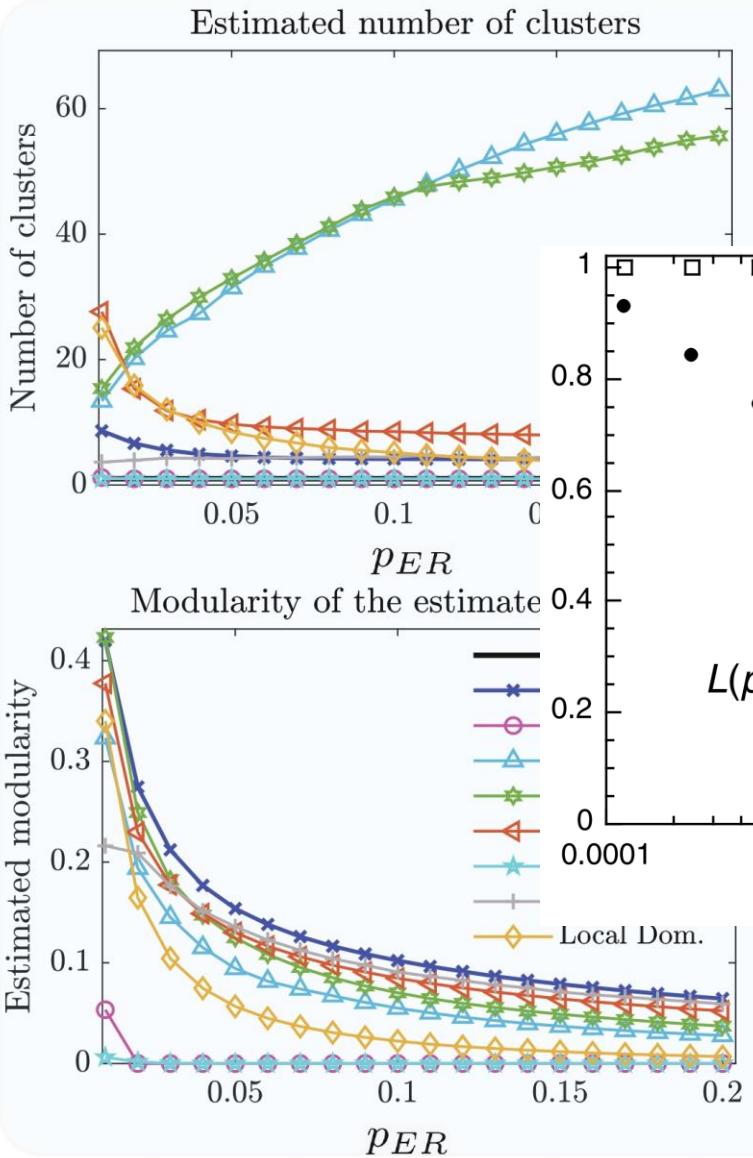
$N = 500, d_{av} = 12, c = 20, \gamma = 2, \beta = 3$



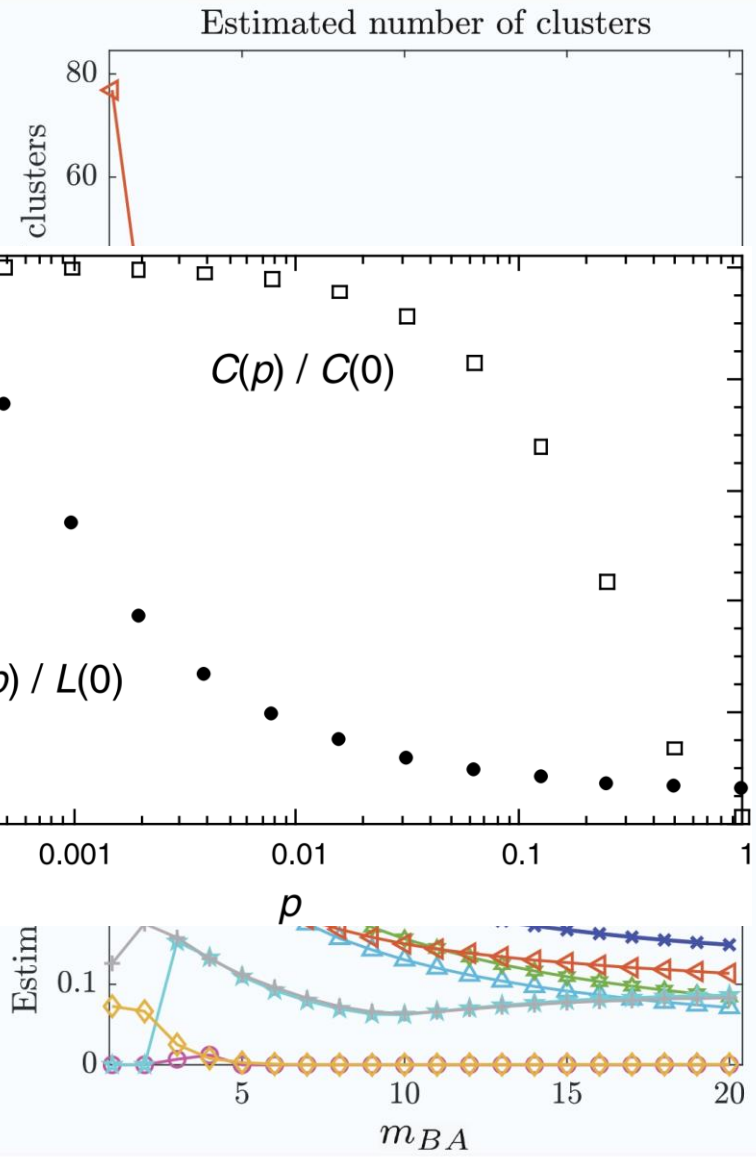
Random network results

$N = 500$

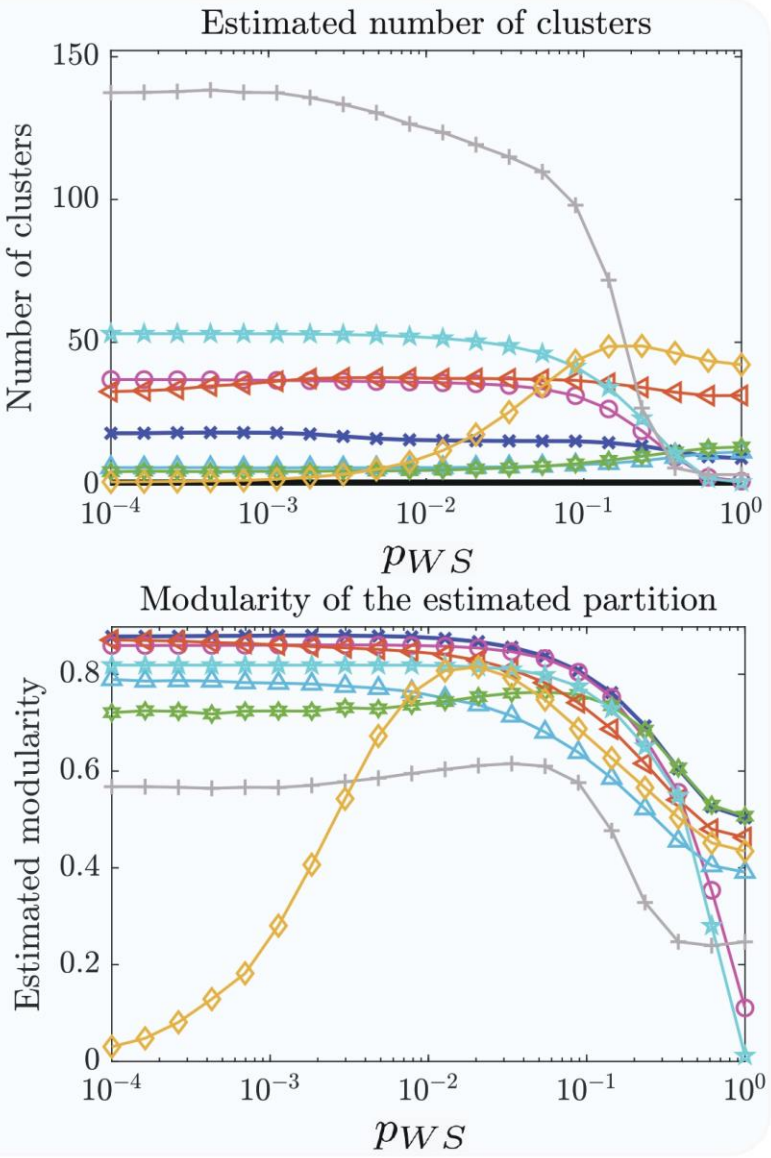
Erdős-Rényi



Barabási-Albert

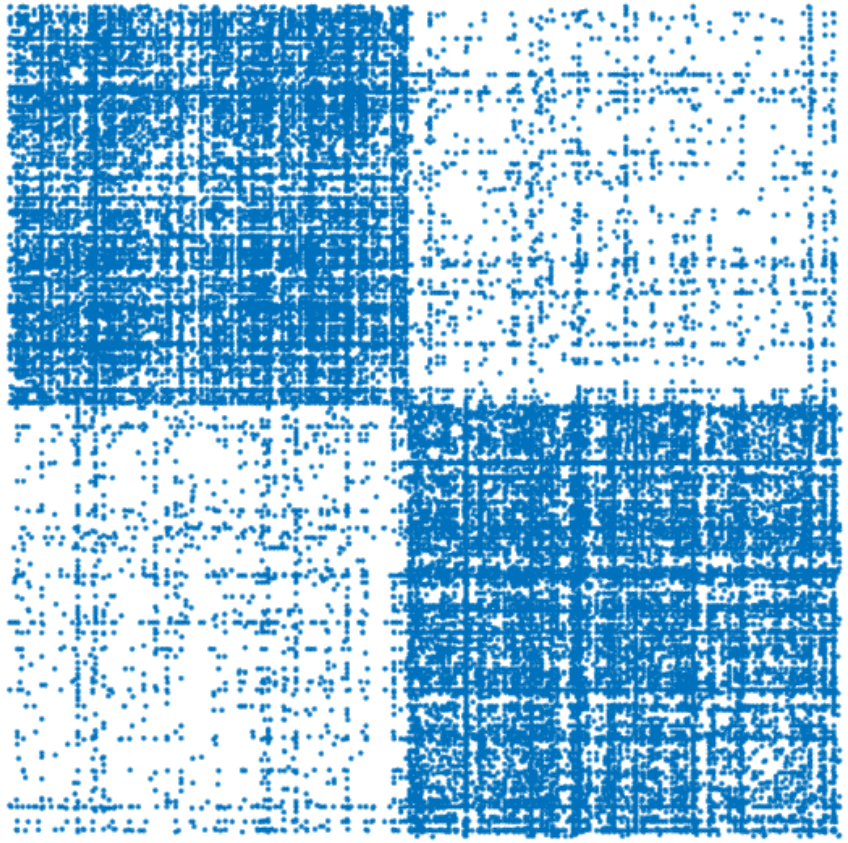


Watts-Strogatz



Real-world networks

Network	N	L
Karate Club	34	78
Dolphins	62	159
Polbooks	105	441
Football	115	613
Facebook	347	2519
Polblogs	1490	19090
Co-authorship	1589	2742



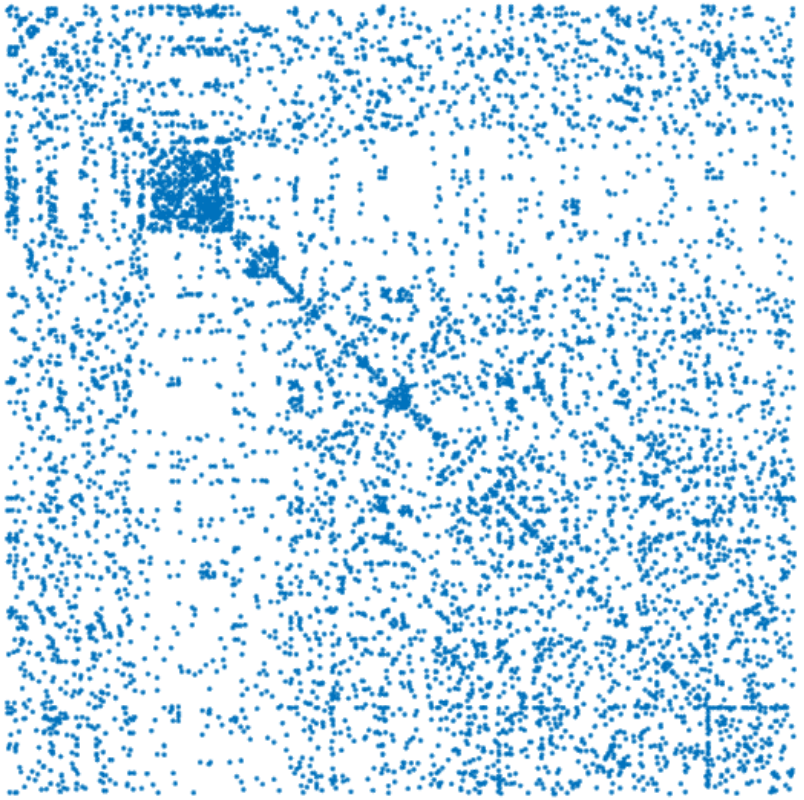
Political Blogs

Results for the real-world networks

Network	LCP		Louvain		Leiden		Newman	
	c	m	c	m	c	m	c	m
Karate	3	**0.3922**	4	0.3565	4	0.3729	5	0.3776
Dolphins	4	0.5057	4	0.4536	5	**0.5105**	6	0.4894
Polbooks	3	**0.5160**	4	0.4897	4	0.5026	8	0.4160
Football	7	0.5894	7	0.5442	7	0.5635	11	0.4623
Facebook	8	**0.4089**	16	0.3726	18	0.3792	23	0.3770
Polblogs	19	**0.4224**	7	0.3385	11	0.3117	4	0.3459
Co-authorship	40	0.9296	272	0.9423	270	0.9410	28	0.7393
Network	**Local Dominance**		**Non-backtracking**		**LCP_n**		**Eigengap**	
	c	m	c	m	c	m	c	m
Karate	2	0.3123	2	0.3715	1	0.0000	2	0.2780
Dolphins	3	0.3620	2	0.3698	2	0.3698	2	0.3115
Polbooks	2	0.4451	3	0.5085	2	0.4546	2	0.4167
Football	6	0.3205	10	**0.5939**	5	0.5522	11	0.5927
Facebook	8	0.2067	8	0.3638	7	0.3544	2	0.2836
Polblogs	3	0.2799	8	0.2149	5	0.3480	2	0.2679
Co-authorship	277	**0.9431**	23	0.5005	17	0.5806	2	0.1288

Real-world networks with ground-truth communities

Network	N	L	C	Modularity
Email-EU	1005	25571	42	0.2880
Cora	2708	5429	7	0.6401
Citeseer	3264	9072	6	0.5042



Citeseer

TU Delft

Results for the real-world networks with ground-truth communities

Network	C	M	LCP				LCP_c				Louvain			
			c	m	nmi	ecs	c	m	nmi	ecs	c	m	nmi	ecs
Email-EU	42	0.2880	9	**0.3860**	0.5466	0.2513	-	0.3005	0.5855	0.3066	12	0.3795	0.5530	0.2747
Cora	7	0.6401	25	0.7296	0.3138	0.2121	-	0.6553	0.2102	0.2370	86	0.6775	0.3691	**0.2808**
Citeseer	6	0.5042	65	0.8027	0.1399	0.0737	-	0.7050	0.0710	0.1919	394	0.7722	0.3095	0.1878
Network	**C**	**M**	**Leiden**				**Newman**				**Local Dominance**			
			c	m	nmi	ecs	c	m	nmi	ecs	c	m	nmi	ecs
Email-EU	42	0.2880	17	0.3745	**0.6352**	**0.3711**	14	0.3492	0.5668	0.3003	1	0.0000	0.0000	0.0669
Cora	7	0.6401	85	**0.7403**	0.4201	0.2722	68	0.7166	0.4146	0.1896	181	0.6728	**0.4271**	0.1648
Citeseer	6	0.5042	395	0.7367	0.3438	**0.2064**	311	**0.8276**	0.3307	0.0560	505	0.7691	**0.3889**	0.0619
Network	**C**	**M**	**Non-back tracking**				**LCP_n**				**Eigengap**			
			c	m	nmi	ecs	c	m	nmi	ecs	c	m	nmi	ecs
Email-EU	42	0.2880	17	0.2792	0.5061	0.2599	2	0.0932	0.2310	0.1211	5	0.2931	0.3538	0.1673
Cora	7	0.6401	40	0.5000	0.3016	0.1558	39	0.5674	0.3320	0.1943	2	0.1072	0.1599	0.1961
Citeseer	6	0.5042	17	0.3595	0.1618	0.1594	16	0.4389	0.1573	0.1581	3	0.1583	0.0960	0.1760

Conclusions

- Among considered clustering algorithms, LCP is superior in modularity consistently;
- In case when clusters are well-defined in a graph, LCP excels at precisely recovering partitions that closely mirror the real clusters, i.e., LCP identifies partitions with high NMI and ECS measures
  - In other cases, LCP provides lower NMI and ECS values, because it identifies alternative partitions from the original one, but with higher modularity;
- ECS reveals how real number of clusters as input benefit LCP partitions;
- Non-backtracking LCP variant achieves as good results as the non-backtracking method except power-law networks, while providing a clear underlying process and reasoning.

TU Delft

Appendix

Newman method

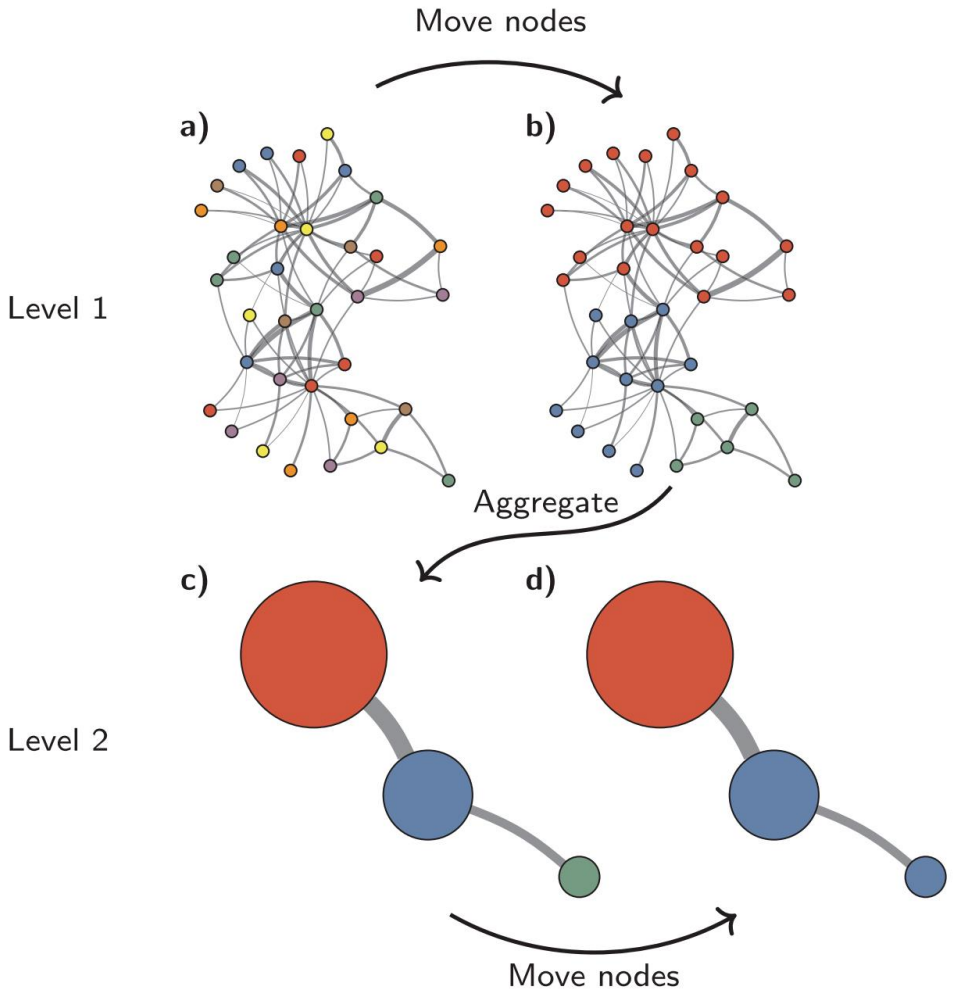
To maximize the modularity *m*, Newman proposes to make recursive splits according to the leading eigenvector of the modularity matrix *M*. The modularity *m* can be rewritten as:

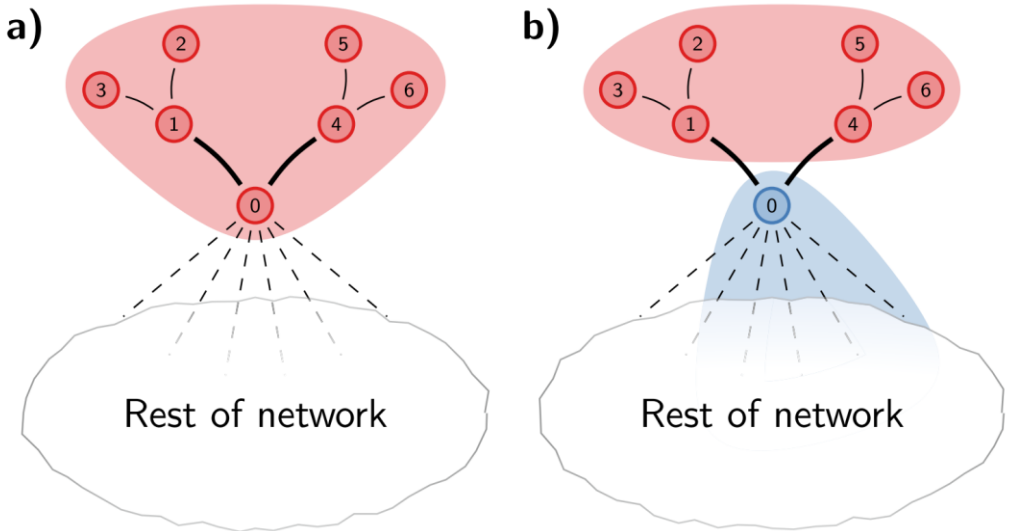
$$m = \frac{1}{4L} y^T M y = \frac{1}{4L} \sum_{j=1}^{N} \beta_j^2 \lambda_j(M)$$

which *y* is composed of values 1 and −1, denoting cluster membership of each node, the modularity matrix $M = A - \frac{1}{2L} \cdot d \cdot d^T$

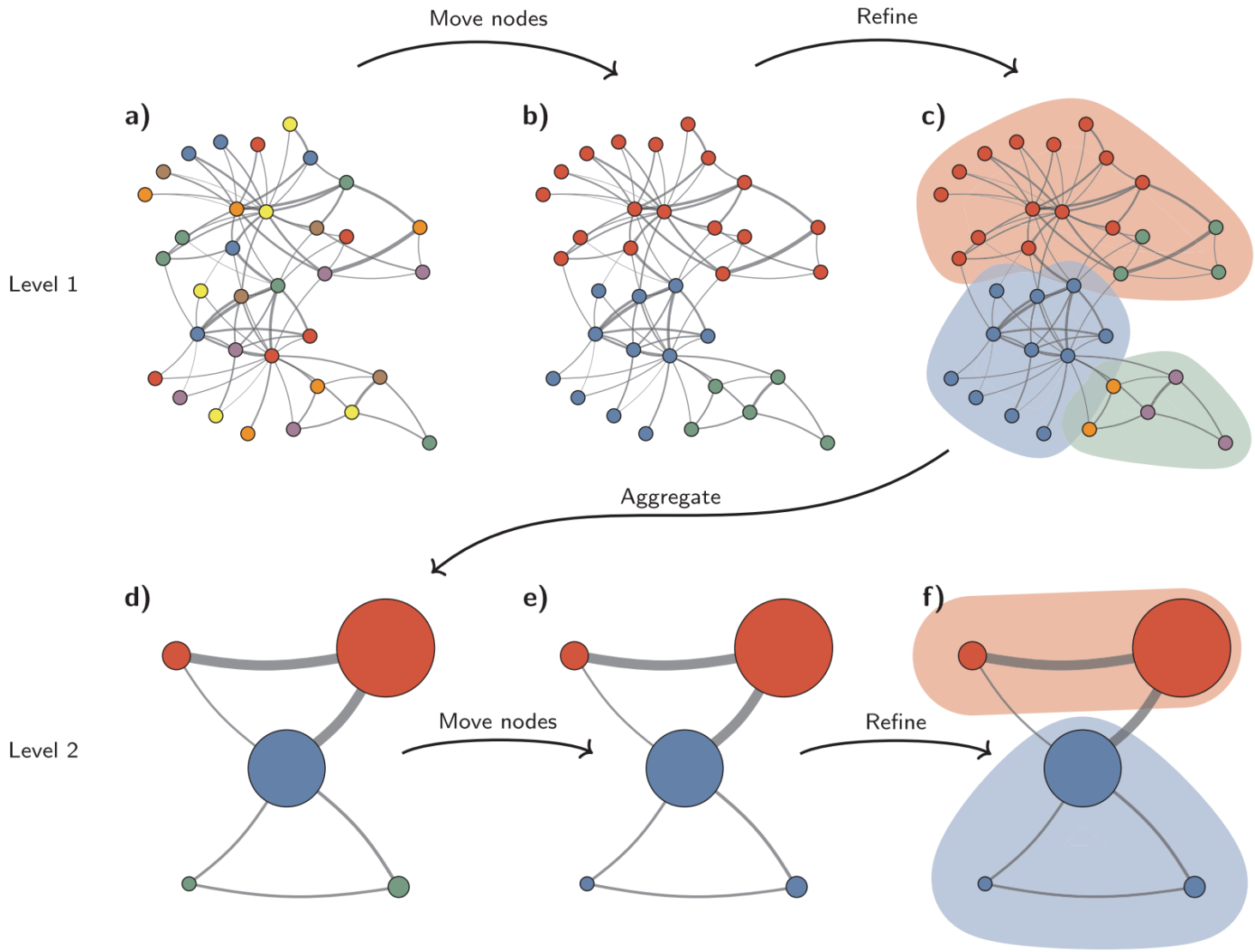
The vector *y* can be written as a linear combination of the orthogonal eigenvectors $w_1, w_2, \ldots, w_N$ of the modularity matrix *M*, $y = \sum_{j=1}^{N} \beta_j w_j$, with coefficients $\beta_j = y^T w_j$. Therefore, maximizing the modularity *m* is equivalent to choosing the vector *y* with cluster memberships proportional to the eigenvectors corresponding to a few of the largest eigenvalues.

Louvain method

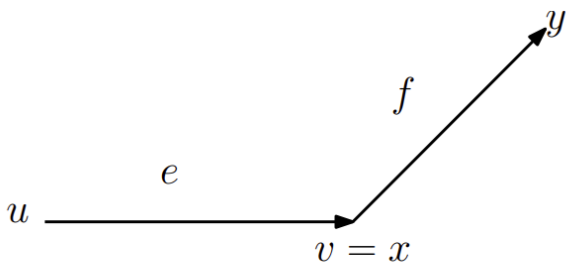


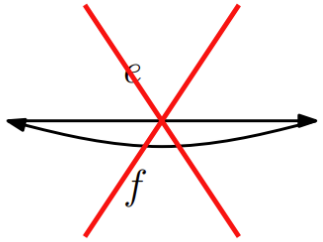


Leiden method

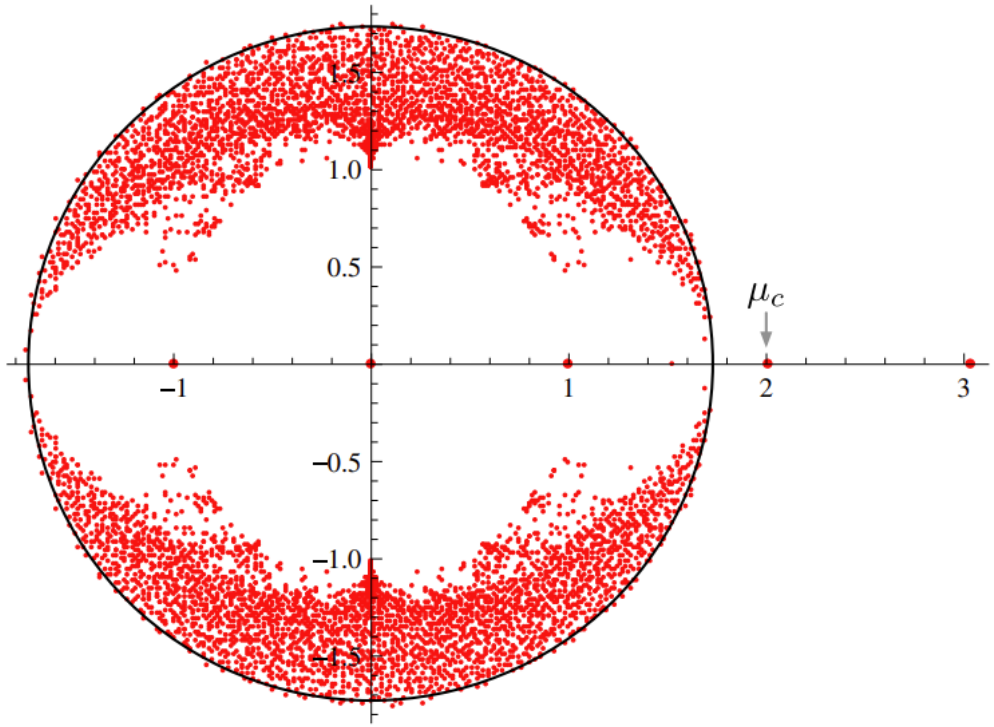


Non-backing tracking method





$$B_{(u\to v),(x\to y)} = \begin{cases} 1, if\ v{=}x\ and\ u{\neq}y \\ 0, otherwise \end{cases}$$



Modularity eigengap

Modularity matrix: $$M_{ij} = A_{ij} - \frac{d_i d_j}{2L}$$

The eigenvalues of the modularity matrix M be sorted in descending order:

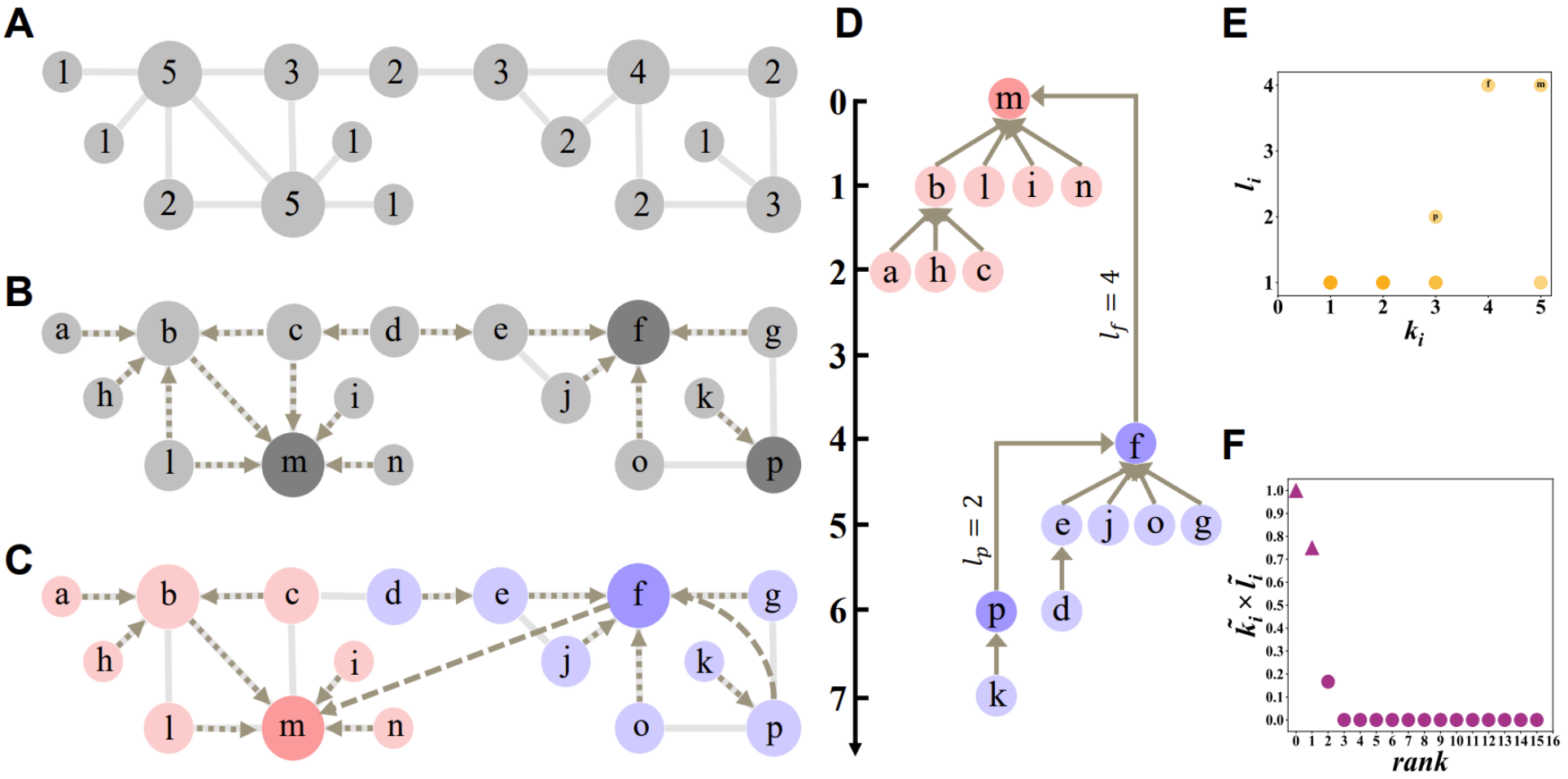
$$\lambda_1(M) \geq \lambda_2(M) \geq \cdots \geq \lambda_N(M)$$

The maximum eigengap property then maximizes the difference $\lambda_{i-1}(M) - \lambda_i(M)$ in the sequence of N eigenvalues as:

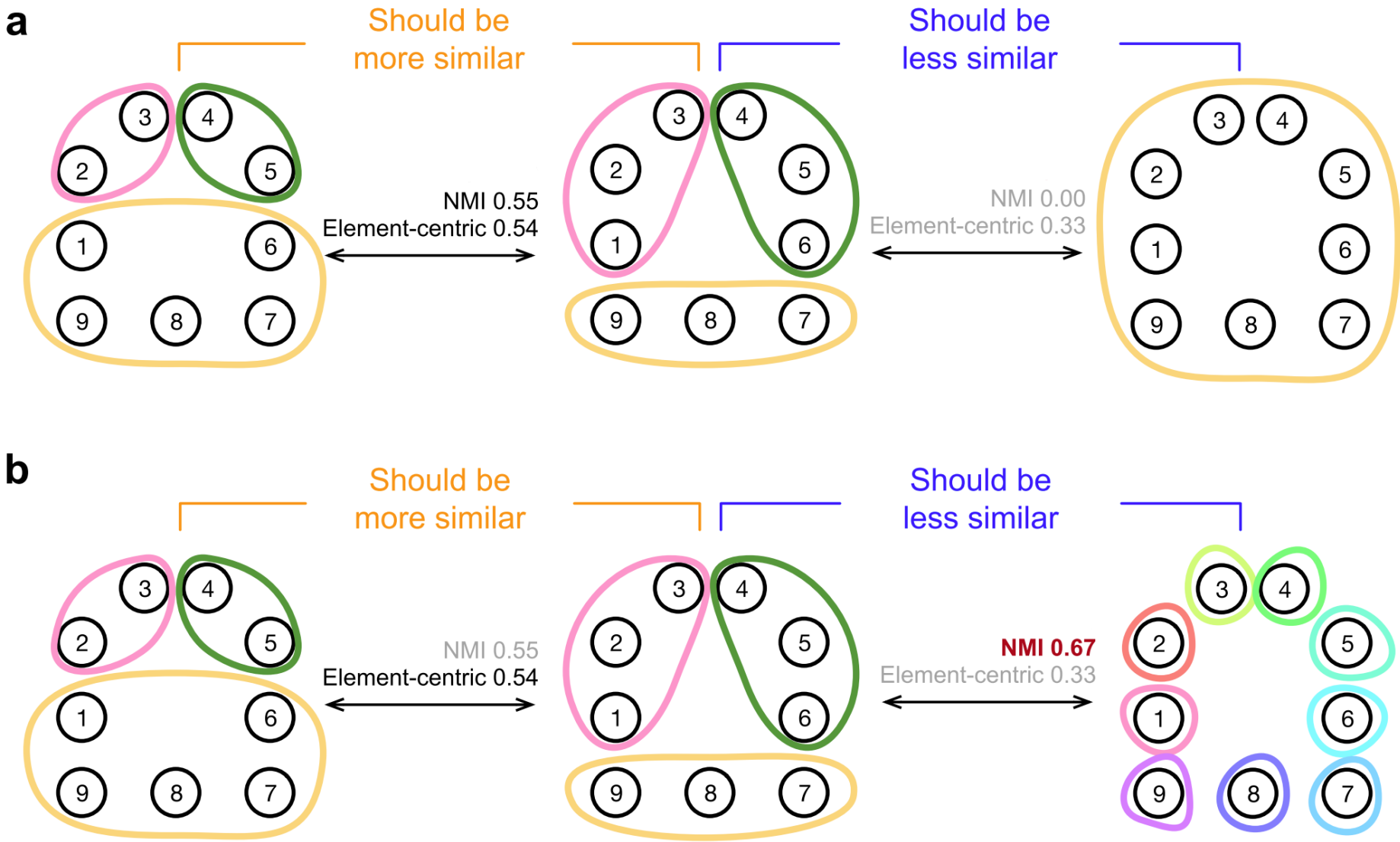
$$c^* = \arg\max_i \left(\lambda_{i-1}(M) - \lambda_i(M)\right), \qquad i = 2, \ldots, N,$$

TU Delft

Local Dominance

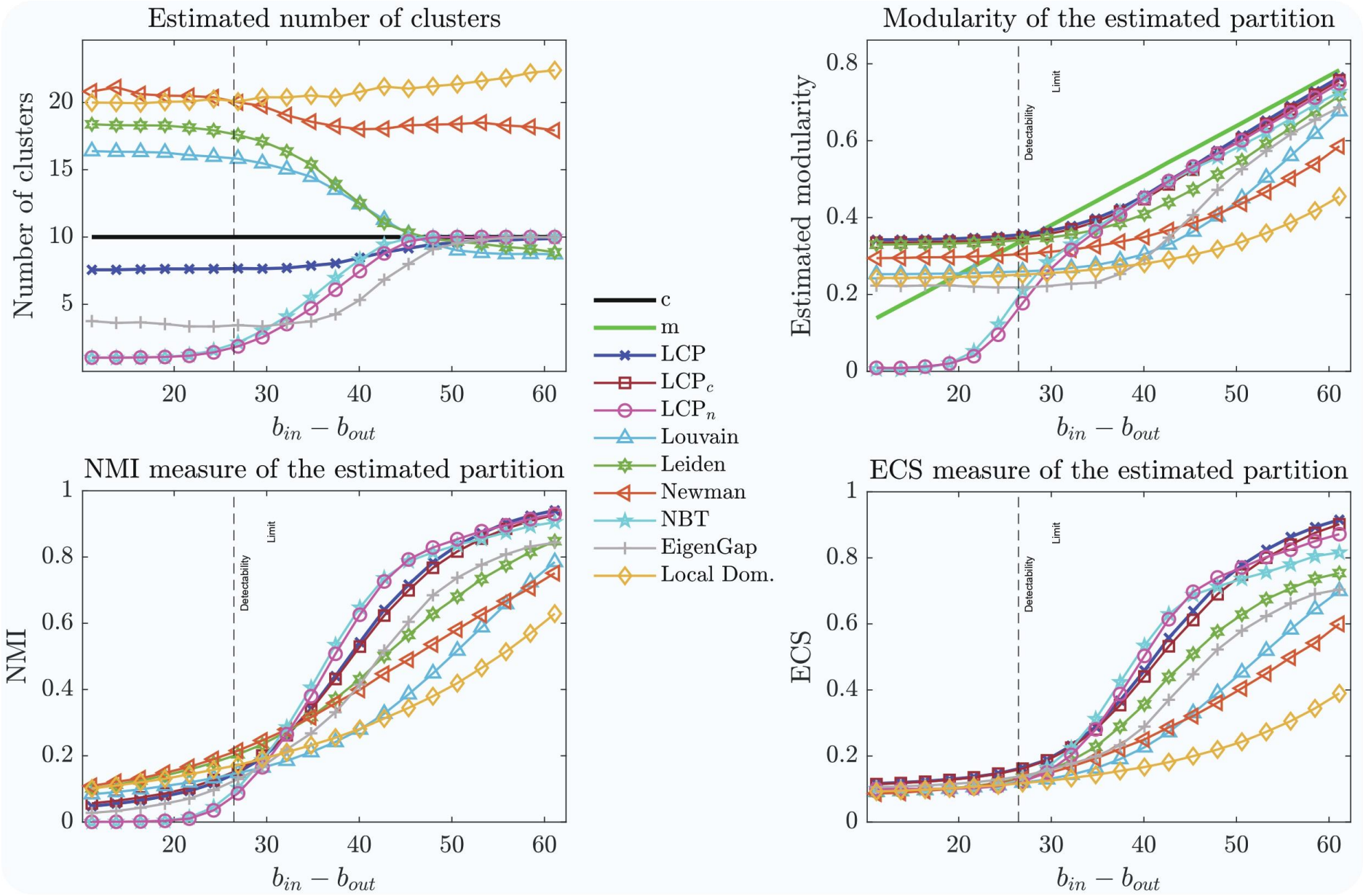


Performance metrics



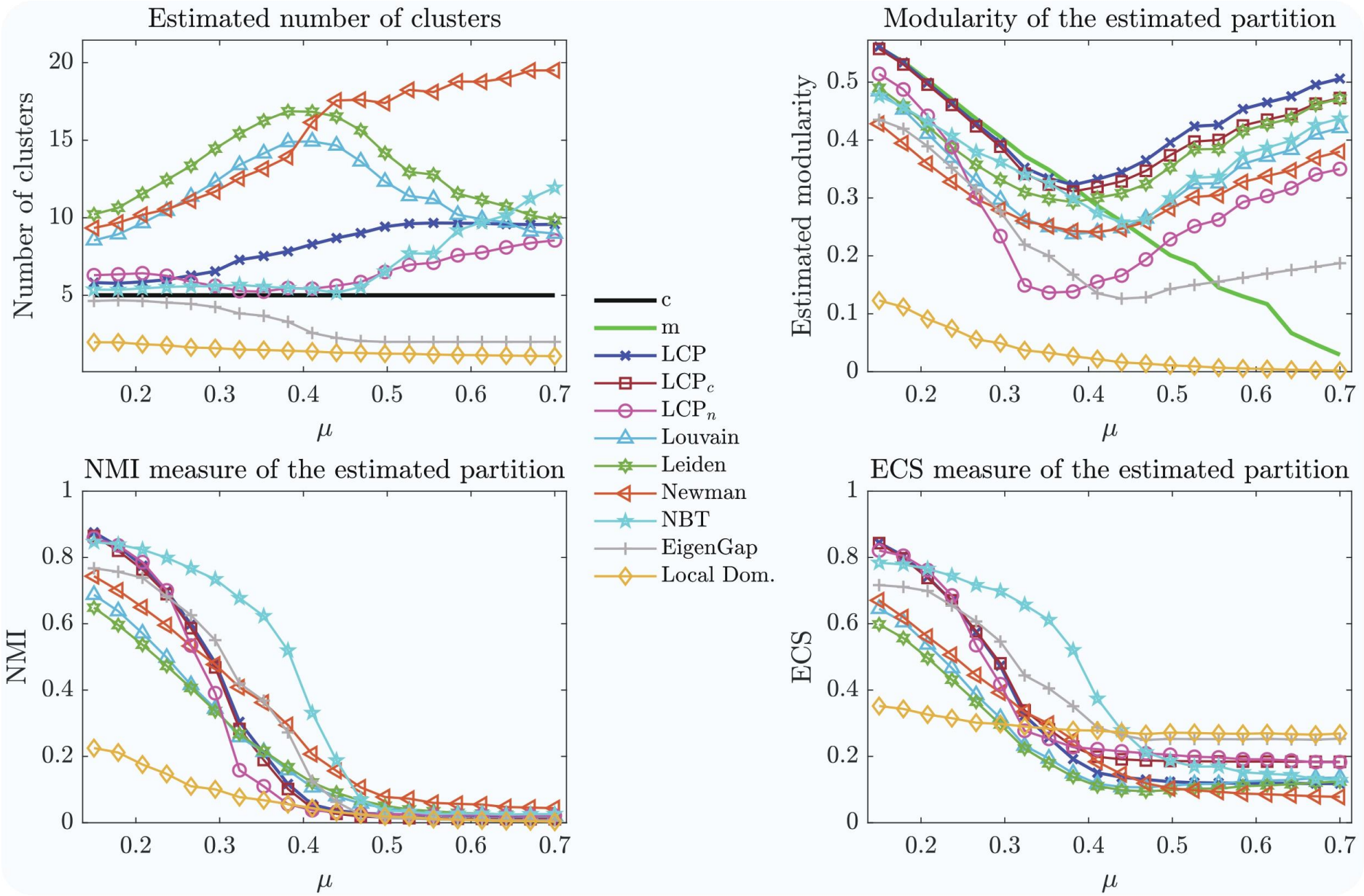
SBM benchmark results

$N = 500, d_{av} = 7, c = 10$



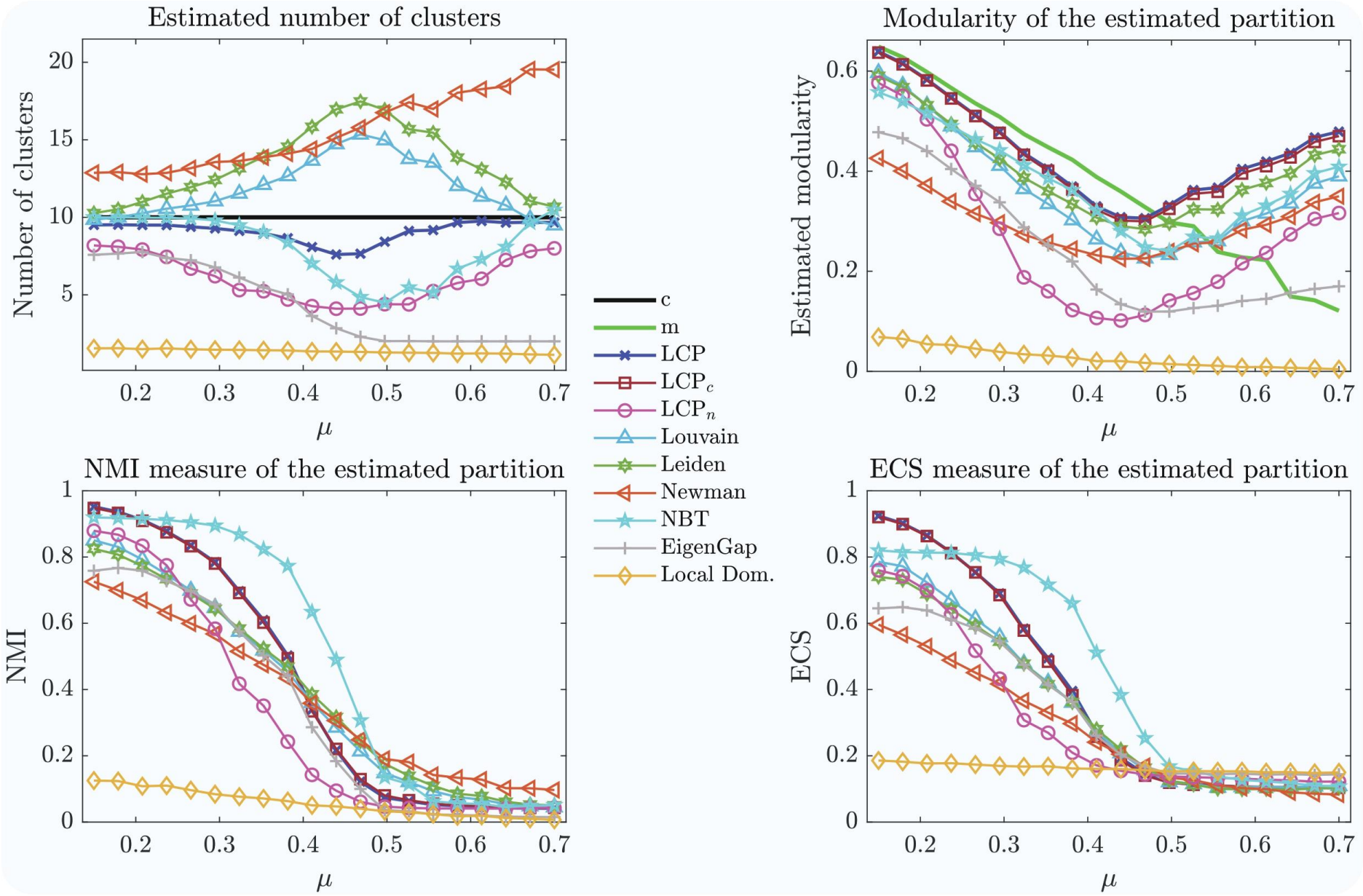
LFR benchmark results

$N = 500, d_{av} = 12, c = 5, \gamma = 2.5, \beta = 2.5$



LFR benchmark results

$N = 500, d_{av} = 12, c = 10, \gamma = 2.5, \beta = 2.5$



LFR benchmark results

$N = 500, d_{av} = 12, c = 20, \gamma = 2.5, \beta = 2.5$