IMA Journal of Complex Networks (2024) Page 1 of 31
doi:10.1093/comnet/xxx000

Linear Clustering Process on Networks: A Comparative Study

IVAN JOKIĆ
Faculty of Electrical Engineering, Mathematics and Computer Science, P.O Box 5031, 2600 GA Delft, The Netherlands

BEICHEN WANG*
Faculty of Electrical Engineering, Mathematics and Computer Science, P.O Box 5031, 2600 GA Delft, The Netherlands
*Corresponding author: kaka19980625@hotmail.com

AND

PIET VAN MIEGHEM
Faculty of Electrical Engineering, Mathematics and Computer Science, P.O Box 5031, 2600 GA Delft, The Netherlands

[Received on 24 April 2024]

By a thorough performance comparison, we compare the recently proposed, *operator-based* Linear Clustering Process on a network with classical, existing clustering algorithms. The Linear Clustering Process produces clusters or partitions based on the eigenstructure of a linear operator on a graph that replaces nodes to ”more natural” positions by attractive and repulsive forces. Synthetic benchmarks, along with real-world networks possessing or lacking a known community structure, are considered. Our comparative analysis demonstrates that our Linear Clustering Process generates superior partitions compared to the algorithms assessed in most instances, while of comparable computational complexity with the simplest existing clustering algorithms.

Keywords: linear clustering process, graph partitioning, spectral clustering

2000 Math Subject Classification: 34K30, 35K57, 35Q80, 92D25

1. Introduction

Networks[2] [21] pervade many disciplines, encompassing a broad range from social and information networks to biological and transportation systems. In these complex networks, the interaction and connection among constituting systems (nodes) is crucial for a comprehensive understanding of the system's overall behavior. One of the most prominent and challenging tasks in network analysis is the identification of communities or clusters, representing groups of nodes with the majority of links connecting nodes within the same cluster, while only a few links join nodes from different clusters[12]. This task, known as community detection or network clustering or graph partitioning, has been the focus of a substantial body of research, primarily due to its profound implications in various fields.

A reliable community detection algorithm should be able to identify “good” partitions[12]. Consequently, the implementation of a dependable quantitative criterion can be instrumental, and at times even indispensable, in distinguishing between “good” and “bad” clusterings and enhancing the algorithm's capacity to produce high-quality partitions. Newman and Girvan pioneered a quality function known as modularity[23], which subsequently inspired a series of optimization algorithms[22]. Building on this concept, Blondel *et al.* introduced the Louvain method[4], a heuristic approach based on modularity optimization. This method was lauded for its computational efficiency and was considered one of the best-performing algorithms of its time[18]. More recently, the Leiden method[27] has emerged as an enhancement to the Louvain method, designed to address some of its known limitations[27] and thereby yield higher-quality community partitions. Shang *et al.* further contributed to this evolving field by proposing the Local Dominance algorithm[26], which aims to unearth the hidden hierarchy within networks by harnessing local information.

Jokić and Van Mieghem [16] proposed the Linear Clustering Process (LCP) on networks for community detection, a novel approach based on properties of a linear operator, specified by a real symmetric matrix, that acts upon a graph. The key idea is similar to Hermitian operators in quantum mechanics, whose eigenstructure relates to the steady-state energy states of a set of particles under the action of the operator. The LCP algorithm involves clustering by moving nodes in a one-dimensional space via attractive and repulsive forces. LCP was shown in [16] to outperform existing algorithms, while maintaining similar computational complexity in a limited number of cases. This paper thoroughly extends the evaluation of LCP's performance. We utilize the Stochastic Block Model (SBM)[14] and the Lancichinetti-Fortunato-Radicchi (LFR) benchmark[19], which includes both random networks and power-law networks. When assessing community partitions, we employ the Element-Centric Similarity (ECS)[13] measure to provide a fairer evaluation, given the inherent bias of Normalised Mutual Information (NMI)[8]. Furthermore, we also equip all spectral methods with the capability to generate community partitions using K-means clustering, thus ensuring a more comprehensive comparison. Throughout rigorous testing conditions, the LCP consistently demonstrates its superior performance. LCP excels in optimizing modularity and its ability to partition communities is particularly remarkable when dealing with well-defined clusters.

In Section 2, we present the performance metrics employed in this study, including modularity, NMI and ECS, as well as various synthetic benchmarks. Section 3 overviews our LCP. Section 4 showcases a comparative analysis of the LCP's performance against that of the Louvain, Leiden, Newman, Non-backtracking, Eigengap, and Local Dominance algorithms. We conclude the paper with a summary of our findings and their implications in the final section.

2. Network or graph clustering

2.1 *Modularity*

Newman and Girvan [23] introduced the concept of modularity to facilitate network partitioning. The modularity measure is denoted as m and defined as follows

$$m = \frac{1}{2L} \cdot \sum_{i=1}^{N} \sum_{j=1}^{N} \left(a_{ij} - \frac{d_i \cdot d_j}{2L} \right) \cdot \mathbf{1}_{\{i \,\text{and}\, j \in \text{ same cluster}\}}, \tag{2.1}$$

where $\mathbf{1}_x$ is an indicator function that equals 1 if statement x is true, and 0 otherwise. Modularity m in equation (2.1), quantifies the difference between the actual number of links between nodes belonging to the same community and the expected number of such links in a randomly connected network. The modularity value, m, approaching 0 indicates that the estimated partition is as good as a random one. Conversely, a modularity value close to 1 suggests a clear partitioning of the network into distinct clusters. Optimising modularity is NP-complete [5], and approximations have been proposed [29]. By defining the $N \times N$ cluster matrix C as:

$$C_{ij} = \begin{cases} 1 & \text{if nodes } i \text{ and } j \text{ belong to the same cluster} \\ 0 & \text{otherwise,} \end{cases} \tag{2.2}$$

we can express the modularity (2.1) as a quadratic form:

$$m = \frac{1}{2L} \cdot u^T \cdot \left(A \circ C - \frac{1}{2L} \cdot \left(d \cdot d^T \right) \circ C \right) \cdot u, \tag{2.3}$$

where the symbol $\circ$ represents the Hadamard product [15]. The number of clusters in the network is denoted as c and the $c \times 1$ vector $n = \begin{bmatrix} n_1 & n_2 & \dots & n_c \end{bmatrix}$, with n_i representing the number of nodes in cluster i, specifies the size of each cluster.

2.2 *Normalised Mutual Information (NMI)*

Danon *et al.* [8] introduced the normalised mutual information (NMI) metric as a means of comparing partitions in network analysis. The metric relies on a confusion matrix, denoted as F, which represents the correspondence between the original communities and the estimated clusters. The ij-th element F_{ij} in the confusion matrix indicates the number of nodes belonging to both the true community i and the estimated community j. The normalised mutual information metric, denoted as $I_n(P_0, P_e)$, between the known partition P_0 and the estimated partition P_e is defined in [8] as:

$$I_n(P_0, P_e) = \frac{-2 \sum\limits_{i=1}^{c_0} \sum\limits_{j=1}^{c_e} F_{ij} \log \left(\frac{F_{ij} N}{F_{i.} F_{.j}} \right)}{\sum\limits_{i=1}^{c_0} F_{i.} \log \left(\frac{F_{i.}}{N} \right) + \sum\limits_{j=1}^{c_e} F_{.j} \log \left(\frac{F_{.j}}{N} \right)}, \tag{2.4}$$

where c_0 represents the number of known clusters, c_e represents the number of estimated clusters, $F_{i.}$ denotes the sum of the i-th row in F, $F_{.j}$ denotes the sum of the j-th column and N represents the total number of nodes in the network. The NMI metric takes a value of 1 when two partitions are identical, and tends towards 0 when the partitions are independent. The NMI measure has been widely employed in evaluating the performance of various clustering algorithms [12] and continues to be a valuable tool in network analysis.

2.3 *Element-centric Similarity (ECS)*

Gates *et al.* [13] introduced the Element-centric Similarity (ECS) metric to tackle the biases inherent in existing clustering comparison measures. The authors highlight that such biases are pervasive in popular metrics. For instance, when comparing the similarity between two clusterings, one is typically fixed as the ground truth while the number of clusters in the other is increased. As a consequence, the NMI value rises, whereas the ECS value declines. This indicates that NMI exhibits a bias towards a greater number of clusters, whereas ECS remains unbiased in this regard.

The process of ECS commences by computing the "personalized PageRank" or "random walk with restart" p_{ij}, incorporating all possible paths between elements to derive the equilibrium distribution for a personalized diffusion process on the graph:

$$p_{ij} = (\alpha_{ecs} / \left| c_\beta \right|) + (1 - \delta_{ij})(1 - \alpha_{ecs}))\delta_{\beta\gamma}, \tag{2.5}$$

where δ is the Kronecker delta function, element v_i is in cluster c_γ, and element v_j is in cluster c_β. And $|c_\beta|$ denotes the cluster size of c_β. $1.0 - \alpha_{ecs}$ represents the restart probability, where $\alpha_{ecs} = 0.9$.

With the personalized PageRank p_{ij}, we can further calculate the element-wise similarity of an element v_i in two clusterings $\mathscr{A}$ and $\mathscr{B}$,

$$S_i(\mathscr{A},\mathscr{B}) = 1.0 - \frac{1}{2\alpha_{ecs}} \sum_{j=1}^{N} \left| p_{ij}^{\mathscr{A}} - p_{ij}^{\mathscr{B}} \right|. \tag{2.6}$$

Finally, the ECS score $S(\mathscr{A},\mathscr{B})$ of two clusterings $\mathscr{A}$ and $\mathscr{B}$ is the average of $S_i(\mathscr{A},\mathscr{B})$,

$$S(\mathscr{A},\mathscr{B}) = \frac{1}{N} \sum_{i=1}^{N} S_i(\mathscr{A},\mathscr{B}). \tag{2.7}$$

The closer the value of ECS is to 1, the more similar the two clusterings are, while the closer the value of ECS is to 0, the more dissimilar the two clusterings are.

Furthermore, it is noteworthy that the complete version of ECS possesses the capability to assess both overlapping clustering and hierarchical clustering. However, within the scope of this paper, all utilized algorithms, benchmarks, and real-world networks exclusively pertain to classic community problem: where each node is singularly allocated to a lone community. As such, our reimplementation of ECS focuses solely on this facet of its functionality.

2.4 *Benchmarks*

2.4.1 *Stochastic Block Model (SBM)* The clustering methods employed in this study were evaluated using random graphs generated by the Stochastic Block Model (SBM), which was proposed by Holland [14]. The SBM generates a random graph with a community structure, where the presence of a link between two nodes depends on whether they belong to the same cluster or not and the probability of the link varies accordingly. This study specifically focuses on the symmetric stochastic block model (SSBM), which involves defining only two distinct probabilities for links. If two nodes belong to the same cluster, they are connected by a link with a probability of p_{in}. Otherwise, a direct link exists between them with a probability of p_{out}. Communities are formed when the link density within clusters is greater than the inter-community link probability, i.e., when $p_{in} > p_{out}$. Additionally, the clusters are constrained to have the same size, denoted by $n_i = \frac{N}{c}$, where $i \in 1,2,\ldots,c$. This constraint ensures that the expected degree is equal for all nodes, regardless of their cluster membership. The expected degree $E[D]$, where D is the random variable of the degree:

$$E[D] = \frac{b_{in} + (c-1)\cdot b_{out}}{c}. \tag{2.8}$$

In this study, we consider a sparse and assortative variant of the SSBM. The terms sparse and assortative imply that the link probabilities, $p_{in} = \frac{b_{in}}{N}$ and $p_{out} = \frac{b_{out}}{N}$, are defined based on positive constants b_{in} and b_{out}, which remain fixed as the network size N approaches infinity. Decelle *et al.* [9, 10] discovered that when the difference between b_{in} and b_{out} surpasses a detectability threshold, represented by the equation:

$$b_{in} - b_{out} > c \cdot \sqrt{E[D]}, \tag{2.9}$$

it becomes theoretically possible to accurately identify the cluster membership of nodes. Conversely, if the difference between b_{in} and b_{out} violates the inequaliy in (2.9), the network's community structure cannot be distinguished from randomness. This threshold inequality $c\sqrt{E[D]}$ in (2.9) represents a critical transition point between the undetectable and theoretically detectable regimes of the SSBM.

2.4.2 *LFR benchmark* Lancichinetti *et al.* [19] proposed the LFR benchmark as an alternative to SSBM graphs, aiming to generate more realistic random graphs that incorporate inherent community structures. Unlike SSBM graphs, where all nodes have the same expected degree, the authors argue that real-world networks often exhibit heterogeneous degree distributions. Moreover, the tails of these distributions are frequently characterized by power laws [20]. In contrast, the LFR benchmark takes into account the observed properties of real-world networks, where community size distributions often follow heavy-tailed distributions [24]. This benchmark produces graphs with the following characteristics:

- Each node's degree is sampled from a power law distribution with an exponent determined by the input parameter γ;
- The size of each community is sampled from a power law distribution with an exponent determined by the input parameter β_{lfr};
- A fraction $1-\mu$ of the links of each node are assigned as intra-community links.

In addition to the parameters mentioned above, the LFR benchmark requires inputs for the network size N, the average degree d_{av}, and the number of communities c.

3. Linear Clustering Process

Jokić and Van Mieghem[16] proposed a linear process of attraction and repulsion between adjacent nodes of a graph, which can be utilised to identify partition.

In the graph G, every node i is allocated a position $x_i[k]$ on a linear axis, which corresponds to a one-dimensional space at a specific, discrete point in time, denoted as k. LCP consists of two opposite and simultaneous forces that change nodal position in time:

- Attraction: Adjacent nodes that have a lot of common neighbors are drawn towards each other with a force that is proportional to the quantity of shared neighbors. In particular, the attractive force between node i and its neighboring node j is proportional to $\alpha \cdot \left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right| + 1\right)$, where α is the attraction strength and $\mathcal{N}_j \cap \mathcal{N}_i$ denotes the set of common neighbors of node i and node j;
- Repulsion: Adjacent nodes are repulsed with a force proportional to the number of neighbours they do not share. In particular, the repulsive force between node i and its neighboring node j is proportional to $\delta \cdot \left(\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| - 1\right)$, where δ is the repulsive strength and $\mathcal{N}_j \setminus \mathcal{N}_i$ denotes the set of neighbors of node j that do not belong to node i, accounted for the direct link between node i and node j, that is contained in $\mathcal{N}_j \setminus \mathcal{N}_i$. To obtain a symmetric repulsion force between node i and node j, when these nodes are interchanged, we define the repulsion force to be proportional to $\delta \cdot \left(\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right| - 2\right)$.

The governing equation at position $x_i[k]$ of node i at discrete time k is

$$x_i[k+1] = x_i[k] + \sum_{j \in \mathcal{N}_i} \left(\frac{\alpha \cdot \left(\left|\mathcal{N}_j \cap \mathcal{N}_i\right| + 1\right)}{d_j d_i} - \frac{\frac{1}{2} \cdot \delta \cdot \left(\left|\mathcal{N}_j \setminus \mathcal{N}_i\right| + \left|\mathcal{N}_i \setminus \mathcal{N}_j\right| - 2\right)}{d_j d_i} \right) \cdot \left(x_j[k] - x_i[k]\right) \tag{3.1}$$

where d_i and d_j denote the degree of node i and node j, respectively. 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. In [16, p. 5], we provide bounds on the attraction α and repulsion δ strength, respectively, to preserve the stability of the LCP process. The node-level governing equation of LCP can be further transformed to the network level and rewritten to a matrix form. The discrete time process (3.1) satisfies the following linear matrix difference equation, as derived in [16, Theorem 1]

$$x[k+1] = (I + W - \text{diag}(W \cdot u)) \cdot x[k], \tag{3.2}$$

where the $N \times 1$ all-one vector is notated as u, the $N \times N$ identity matrix is denoted by I, while the $N \times N$ topology-based matrix W is defined as

$$W = (\alpha + \delta) \Delta^{-1} \cdot \left(A \circ A^2 + A \right) \cdot \Delta^{-1} - \frac{1}{2} \cdot \delta \left(\Delta^{-1} \cdot A + A \cdot \Delta^{-1} \right) \tag{3.3}$$

where $\circ$ denotes the Hadamard product. The eigenvalue decomposition of the $N \times N$ governing matrix $W - \text{diag}(W \cdot u)$ can be written as

$$W - \text{diag}(W \cdot u) = Y \text{diag}(\beta) Y^T \tag{3.4}$$

where the $N \times 1$ eigenvalue vector $\beta = (\beta_1, \beta_2, \cdots, \beta_N)$ with $\beta_1 \geqslant \beta_2 \geqslant \cdots \geqslant \beta_N$ and Y is the $N \times N$ orthogonal eigenvector matrix with the eigenvectors $y_1, y_2, \cdots, y_N$ in the columns obeying $Y^T Y = Y Y^T = I$. The dominance of attraction forces over repulsive ones governs the LCP dynamics eventually to a trivial steady state, where each node occupies the same position, characterised by $\beta_1 = 0$ and $y_1 = u$, as derived in [16, p. 4]. Therefore, LCP's steady state yields no useful information about communities. However, before converging to a single position, nodes from the same community tend to be relatively closer on a line, while being more distant from remaining nodes. Since $\beta_1 = 0$ and $-1 < \beta_j < 0$ for $j > 1$, $|1 + \beta_2| > |1 + \beta_3|$ holds. Therefore, after importing (3.4) into (3.2) we obtain

$$\frac{x[k] - \frac{u^T x[0]}{\sqrt{N}} u}{(1+\beta_2)^k \left(y_2^T x[0] \right)} = y_2 + O\left(\frac{1+\beta_3}{1+\beta_2} \right)^k, \tag{3.5}$$

The left-hand side of (3.5), representing a shifted or normalized position vector, gravitates toward the second eigenvector y_2, accompanied by an exponentially diminishing error as k increases. However, this only holds true for sufficiently large, but not excessively large, values of k. Thus, the information used for clustering the graph is provided by this scaled and shifted position vector.

A block diagonal structure of the $N \times N$ adjacency matrix A can be found by sorting the $N \times 1$ eigenvector y_2, in ascending or descending order. This sorting process leads to a graph relabeling, where each node is assigned a new label based on its position in the sorted eigenvector, denoted as $\hat{y_2}$. Consequently, the original two-dimensional clustering problem[1] is transformed into a one-dimensional problem. In this new formulation, we either group nodes with similar values in $\hat{y_2}$ to form communities, or we determine the community boundaries by optimizing a quality function, such as the modularity.

Clusters are estimated for a node ranking derived from the sorted eigenvector y_2 through recursive optimisation of the modularity. In the first iteration, all possible partitions of the network into two clusters are examined and their modularity is computed. The partition that yields the highest modularity is selected. In the second iteration, the same procedure is repeated for each sub-graph and the best partitions into two clusters are found. Once the best partitions for both sub-graphs are determined, they

[1]Since a graph structure is defined by a matrix, capturing node-pair relations, clustering problem is a two-dimensional problem, as it involves grouping nodes in communities. In LCP, we focus on the sorted $\hat{y_2}$ eigenvector and identify borders of each cluster. Therefore, LCP reduces the clustering problem to one dimension.

are adopted if the modularity of the generated partition surpasses the modularity of a parent cluster from the previous iteration. The recursive procedure is halted when there can be no further improvement in the modularity. The pseudo-code for this recursive algorithm is in the Appendix F of [16]. Our LCP's performance could be further enhanced by integrating the dynamic programming-based algorithm proposed by Patania et al. [25]. This algorithm offers the advantage of guaranteeing an optimal partition for a given one-dimensional node embedding.

Finally, to further improve the quality of community partition, non-linearity in the forces can be introduced iteratively through the scaling of inter-community link weights. A link between nodes i and j will be scaled down to a positive value smaller than 1 when the difference of their rankings in the sorted eigenvector $\hat{y_2}$ is greater than the threshold value. This scaling artificially reduces the strength of forces between nodes from different clusters. By iteratively decreasing inter-community link weights, LCP enhances node grouping precision. This reduction weakens the attractive forces between adjacent node pairs likely belonging to different communities. Without this adjustment, the dominant attraction force (which scales linearly with distance) would intensify over time. This could erroneously pull neighboring nodes from distinct communities closer together. Instead, the importance of links between nodes from different clusters is diminished, guided by the partition derived from the preceding iteration. This variant is denoted as LCP in Section 4.

3.1 *LCP for a known number of communities*

The aforementioned algorithm for recursively optimizing modularity can also be applied to graph partitioning with a known number of communities c. In this scenario, the recursive procedure outlined does not stop when the modularity can no longer be improved, but at iteration $(\log_2 c + 1)$. In every iteration, the partition with the highest modularity is accepted, regardless of whether it is negative. This variant is denoted as LCP_c in Section 4.

3.2 *Non-backtracking variant of LCP*

The non-backtracking matrix B is based on the concept of non-backtracking walks on a network G, which are walks where one does not immediately "reverse" along a link just traversed. In particular, a non-backtracking matrix B starts by creating a list of directed links for an undirected network, which means replacing every link with two directed links in opposite directions. Therefore, for an undirected network $G(\mathscr{N}, \mathscr{L})$, the corresponding directed network is constructed with $2L$ links. Then, the $2L \times 2L$ non-backtracking matrix B is defined by:

$$B_{(u\to v),(w\to z)} = \begin{cases} 1 & \text{if } v = w \text{ and } u \neq z \\ 0 & \text{otherwise,} \end{cases} \tag{3.6}$$

where $v, w, z \in \mathscr{N}$.

Krzakala *et al.*[17] proposed a spectral method for clustering networks based on non-backtracking matrix. The asymmetric non-backtracking matrix B generally has complex eigenvalues. Most of these eigenvalues are located within a bulk whose center is the origin of the complex plane and whose radius is the square root of the largest real eigenvalue. Therefore, they estimated that the number of clusters in graph G corresponds to the number of real eigenvalues that lie outside this bulk. The computational complexity of non-backtracking method is $O(L^3)$, mainly due to the eigenvalue decomposition., but it can be [6] reduced to $O(N^3)$.

Angel *et al.*[1] shows another approach of computing the eigenvalues of a non-backtracking matrix. Instead of directly decomposing the $2L \times 2L$ non-backtracking matrix B, these $2N$ non-trivial eigenvalues are also contained in the decomposed $2N \times 2N$ matrix B^*:

$$B^* = \begin{bmatrix} A & I-\Delta \\ I & O \end{bmatrix}, \tag{3.7}$$

where O denotes the $N \times N$ all-zero matrix. This matrix B^* can be rewritten as:

$$B^* = \begin{bmatrix} I+(A-\Delta)+(\Delta-I) & -(\Delta-I) \\ I & O \end{bmatrix} \tag{3.8}$$

Hence, the matrix B^* can be viewed as a state-space matrix of a process on a network, consisting of attractive and repulsive force between adjacent nodes, similar to the LCP process defined in equation (3.2). In the process outlined by the matrix B^*, the final N states (corresponding to the second block row in B^*) store nodal position vector form a previous iteration. The matrix B^* establishes a set of N second-order difference equations. The governing equation that dictates the position of node i is defined by

$$x_i[k+1] = x_i[k] + \sum_{j \in \mathcal{N}_i} (x_j[k] - x_i[k]) + (d_i - 1) \cdot (x_i[k] - x_i[k-1]), \tag{3.9}$$

where the second term can be considered as a uniform attractive force between neighboring nodes, whereas in LCP, the intensity of the attractive force is dependent on the number of common neighbors between two adjacent nodes. In addition, we observe from (3.9) that node i is pushed away from its former position $x_i[k]$ in the direction of its most recent position change $x_i[k] - x_i[k-1]$. We define the $2N \times 2N$ matrix W^*, corresponding to B^*, as follows

$$W^* = \begin{bmatrix} I+\alpha \cdot (A \circ A^2 + A - \text{diag}((A \circ A^2 + A) \cdot u) + (\Delta - I) & -(\Delta - I) \\ I & O \end{bmatrix} \tag{3.10}$$

where the attractive force is retained according to (3.1), while the repulsive force is adopted as in the non-backtracking clustering method. The number of clusters c is determined by counting the number of real eigenvalues in W^*, which exceed the square root of its largest real eigenvalue.

We named the approach LCP_n and we test it on several benchmarks, random network models and real-world networks. Specific results and analysis are presented in Section 4. The positive aspect is that LCP_n performs almost identically to non-backtracking on the Stochastic Block Model and random network models. However, LCP_n performs poorly on power-law networks, such as the LFR benchmark and some of real-world networks as shown in Figure 1. After analysis and testing, we found that for power-law networks, the attraction strength α has a significant impact on the performance of the algorithm. Previously, we have fixed $\alpha = 0.95$, mainly because an α close to 1 helps to maximize the difference between the second largest eigenvalue β_2 and the third largest eigenvalue β_3 of the $N \times N$ matrix $W - \text{diag}(W \cdot u)$. Therefore, for the power-law network, we adopt the strategy of dynamically adjusting the value of α, which is described below.

The probability density function $f_D(d)$ of the power-law distribution [7] is as follows

$$f_D(d) = c_p \cdot d^{-\gamma}, \tag{3.11}$$

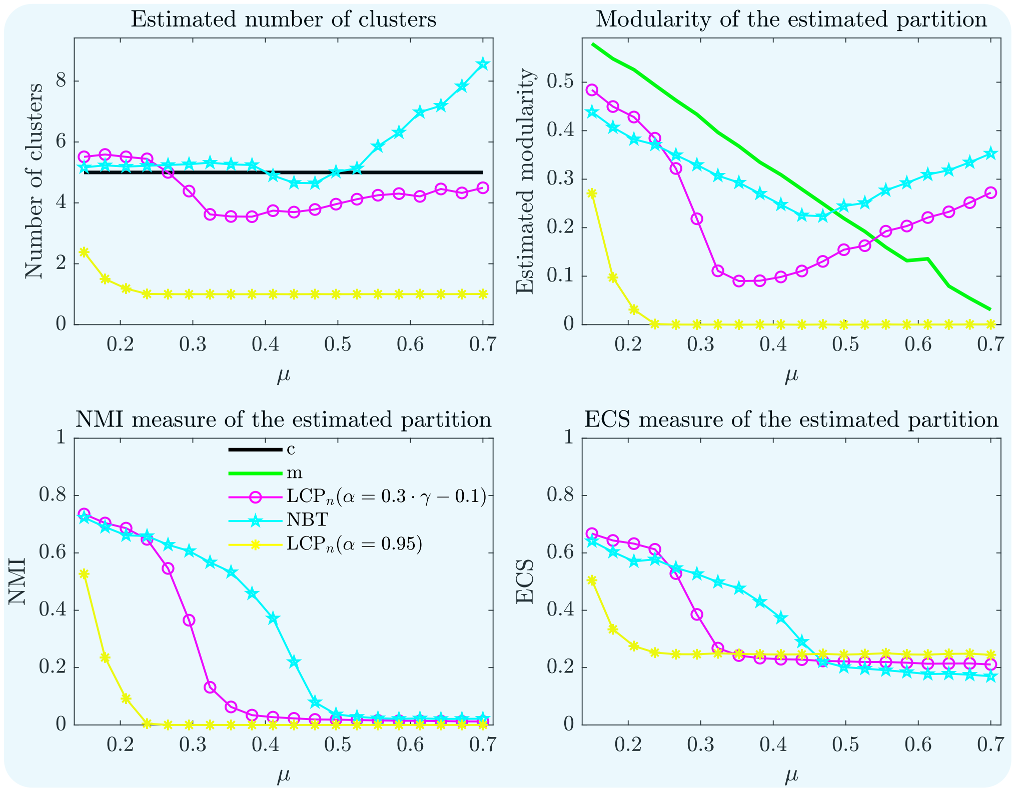

FIG. 1. The estimated number of clusters (upper left) and estimated modularity (upper right) in LFR benchmark graphs with $N = 500$ nodes, an average degree of $d_{av} = 12$, comprising $c = 5$ clusters. The graphs are generated using parameters $\gamma = 2$ and $\beta_{lfr} = 3$ and varying the parameter μ. The lower left and lower right figures display the Normalized Mutual Information (NMI) and Element-centric similarity (ECS) measures for each clustering algorithm.

where $c_p = \frac{1-\gamma}{d_{max}^{1-\gamma} - d_{min}^{1-\gamma}}$ denotes the normalisation constant, while γ denotes the power-law exponent or scaling parameter. For a given network, we utilise the $N \times 1$ degree vector d to estimate the corresponding power-law exponent γ, which helps determining whether a network is a power-law network and how "skew" the degree distribution is. In particular, the power-law exponent γ determines the shape of the distribution[7][3]:

- A smaller power-law exponent indicates a heavy-tailed distribution, meaning that there are more highly connected nodes (or "hubs") in the network. These networks are sometimes described as "scale-free" because their degree distributions are the same at all scales. In other words, the proportion of nodes with a certain number of links remains constant, regardless of the total number of links;

- A larger power-law exponent indicates a lighter-tailed distribution. This means that the network has fewer highly connected nodes. As the power-law exponent increases, the distribution becomes more similar to an exponential distribution, which has fewer hubs and is less skewed.

Based on this power-law characteristic, we consider that if power-law exponent γ increases, the value of the attraction strength α should also increase, because its network properties will be closer to a random network rather than a power-law network. We use the LFR benchmark to generate networks with different γ to find the relationship between power-law exponent γ and attraction strength α, by changing attraction strength α and testing on networks with fixed γ to find the γ value that can make the algorithm perform optimally. Finally, we summarize our findings in the following function:

$$\alpha = \begin{cases} 0.3 \times \gamma - 0.1 & \text{if } 1 < \gamma \leqslant 3.5 \\ 0.95 & \text{otherwise,} \end{cases} \tag{3.12}$$

In (3.12), we set an upper bound $\alpha = 0.95$ and the relationship between γ and γ for $1 < \gamma \leqslant 3.5$ is derived from a linear fit to the results of the above experiments. Figure 1, with the strategy of dynamically adjusting the value of γ shows that the performance of LCP_n improves significantly from that with fixed α, but is clearly still inferior (in the modularity m, NMI and ECS measure) to that of the Non-backtracking matrix.

4. Comparing clustering algorithms

4.1 *Computational complexity*

Table 1. Computational Complexity of considered clustering algorithms

Clustering Algorithm	**Computational Complexity**
Leiden	$\mathcal{O}(L)$
Louvain[2]	$\mathcal{O}(L)$
Local Dominance	$\mathcal{O}(L)$
LCP	$\mathcal{O}(N \cdot L)$
LCP_c	$\mathcal{O}(N \cdot L)$
LCP_n	$\mathcal{O}(N^3)$
Newman	$\mathcal{O}(N^3)$
Non-backtracking	$\mathcal{O}(N^3)$
Eigengap	$\mathcal{O}(N^3)$

The Louvain method [4] is known for its efficient clustering performance, requiring minimal computational complexity. Studies have demonstrated that its time complexity is linear with the number of links L in sparse graphs. An improved version of Louvain, called the Leiden method, achieves comparable complexity when performing community partitioning. On the other hand, the Local Dominance approach utilizes local information to estimate communities in a graph, resulting also in a computational complexity that scales linearly with the number of links. In contrast, our LCP demands greater computational effort for estimating partitions. The computational complexity of LCP, as described in [16], scales linearly with the product of the number of nodes N and the number of links L. The variant LCP_c of our LCP, where the number of communities c is provided as input, exhibits a comparable computational complexity. Furthermore, our LCP_n variant employs eigenvectors of the corresponding governing matrix W^* to estimate communities. This approach requires computational effort that scales with the cube of the network size, denoted as $\mathcal{O}(N^3)$. A similar computational complexity also applies to any spectral clustering method, which is necessary for performing spectral decomposition.

4.2 *SBM benchmark*

4.2.1 *SSBM network with $c = 2$ clusters* Figure 2 illustrates the clustering performance of the LCP algorithm and other methods considered in this study. Regarding the estimation of the number of communities c (top-left figure), the non-backtracking (NBT) method and the equivalent version of LCP (i.e. LCP_n, see Section 3.2) demonstrate superior performance, converging to two clusters for values of $b_{in} - b_{out}$ above the detectability threshold. The EigenGap method follows closely, achieving convergence to the exact number of clusters for higher values of $b_{in} - b_{out}$, while estimating more than $c = 2$ communities in the undetectable regime. In terms of precision in estimating the true number of clusters, LCP ranks fourth and converges to $c = 2$ only when the clusters are clearly distinct (i.e., when there is a large difference between b_{in} and b_{out}), because it optimises modularity m, as explained in the following paragraph. Although the Newman method performs the worst when the clusters are indistinguishable, it outperforms Louvain, Leiden and Local Dominance methods once the communities start to emerge from the randomness in the SSBM network structure. The Louvain and Leiden methods perform similarly on SSBM networks with $c = 2$ clusters and converge to the correct number of clusters only in cases when there are almost no inter-community links. These two algorithms' tendency to identify a large number of communities stems from their design, which initializes every node as a separate community. Finally, the Local Dominance performs the worst overall, irrespective of the number of inter-community links.

LCP identifies partitions with the highest modularity m overall, as depicted in the upper right-hand side of Figure 2. Even when original clusters are indistinguishable from randomness, LCP identifies alternative communities by optimising the modularity m. The variant LCP_c of LCP, which receives the true number of communities as input, achieves the original modularity m in the region where communities are visible. Furthermore, just above the detectability threshold, the modularity achieved by LCP_c exceeds the original modularity m, indicating that alternative partitions with the same number of communities but higher modularity coexist around the threshold! The same argument explains the incorrect number of clusters c identified by LCP, as the achieved modularity is consistently higher than that of the original partition. Non-backtracking method and our LCP_n achieve modularity m of the planted partition once they estimate the correct number of clusters c. In contrast, when original communities are indistinguishable from randomness, these methods fail to reveal any clusters, and thus the resulting modularity is zero. For lower values of $b_{in} - b_{out}$, the Leiden method performs closely to LCP. At the same time, the performance declines as the number of intra-community links increases compared to other algorithms. Leiden consistently outperforms Louvain, which aligns with the idea of the Leiden method being an improvement of the Louvain algorithm. Since the Newman method converges to the correct number of clusters c, its modularity increases and tends towards the original modularity when clusters are visible while identifying alternative partitions with superior modularity when original clusters are not distinguishable. Finally, Louvain and Local Dominance perform similarly in modularity when original clusters are not clearly visible, while Louvain estimates better partitions as the number of inter-community links decreases.

We utilise the NMI measure explained in Section 2.2, together with the ECS measure described in Section 2.3, to compare the estimated partitions of each clustering algorithm considered with corresponding planted partitions. From the NMI measure presented in the bottom-left part of Figure 2, we observe that the non-backtracking method (NBT) and our LCP_n reveal partitions most similar to the planted partition overall, when they estimate a correct number of clusters. The LCP_c variant[3] per-

[3]which receives the number of clusters c as input

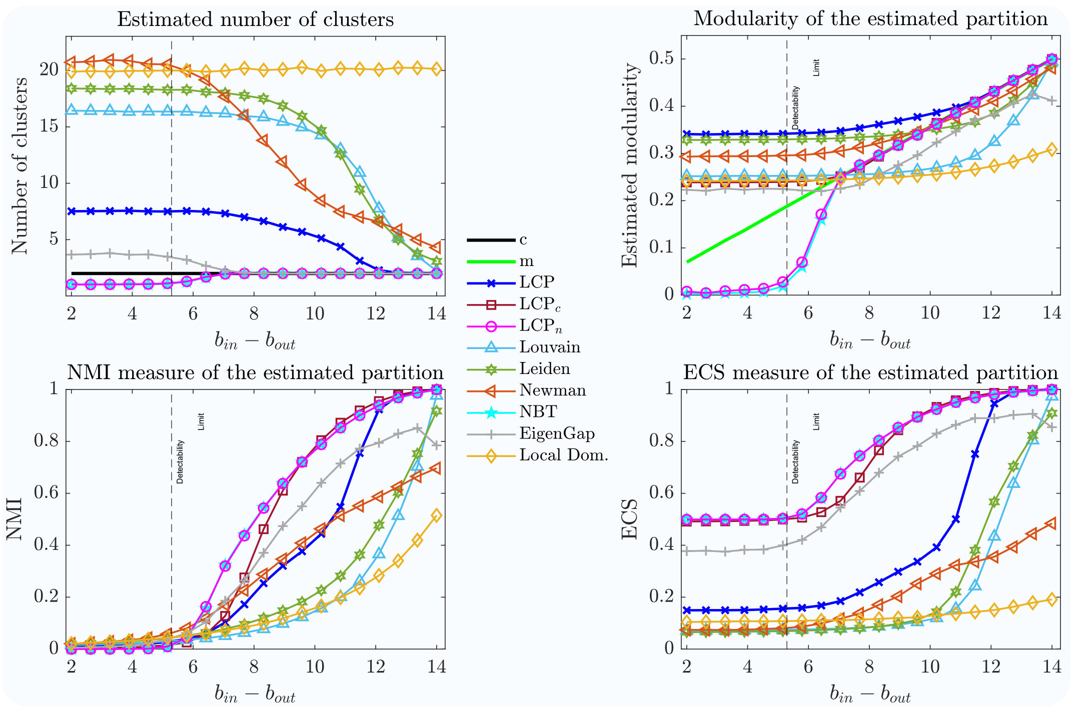

FIG. 2. The estimated number of clusters (upper left-hand figure) and estimated modularity (upper right-hand figure) in case of SSBM graph with $N = 500$ nodes, average degree $d_{av} = 7$ and $c = 2$ clusters, for different values of parameters b_{in} and b_{out}. The NMI and ECS measures per each clustering algorithm are provided in the lower left-hand side and lower right-hand side figures, respectively. The vertical dashed line indicates the clustering detectability threshold.

forms the best when clusters are visible (i.e. for higher values of $b_{in} - b_{out}$). However, just above the detectability threshold, LCP$_c$ identifies alternative communities with superior modularity m and, thus, less similar to the planted partition. The Newman method and LCP perform similarly in the region above the detectability threshold but worse than the EigenGap method. At the same time, LCP outperforms remaining considered algorithms, especially as the number of inter-community links decreases. Leiden and Louvain perform similarly, with Leiden reproducing original partitions with slightly higher accuracy, while Local Dominance performs the worst.

The Element-centric similarity (ECS) measure, proposed in [13], accommodates overlapping and hierarchical clusterings, thereby resolving the bias in the NMI measure towards many clusters. The ECS measure reveals that NBT, our LCP$_n$ and LCP$_c$, dominantly outperform other clustering algorithms when identifying partitions in the SSBM network with $c = 2$ communities, regardless of the number of inter-community links. The ECS measure achieved by LCP$_c$ drops locally above the detectability threshold due to observed alternative partitions, as explained in the previous paragraph. EigenGap follows the aforementioned algorithms in ECS performance while being outperformed by our LCP in the case of a relatively low number of inter-community links. Overall, our LCP predominantly outperforms all non-spectral clustering methods considered. It is worth noting that LCP performs clustering based on a linear physical process while estimating the borders of clusters by maximizing modularity. Therefore, the ECS measure clearly indicates that LCP$_c$ accurately discovers clusters through its underlying process. Other algorithms perform similarly, whereas Newman outperforms the remaining algorithms for the $b_{in} - b_{out}$ values above the detectability threshold. In contrast, the Leiden and Louvain algorithms dominate once the clusters become more visible. The modularity of the community partition found by LCP is always quite higher than the original, which means that LCP found a better alternative partition and therefore failed to restore the original partition. This is an inherent shortcoming of LCP when dealing with networks with dense inter-community links.

4.2.2 *SSBM network with $c = 4$ clusters*

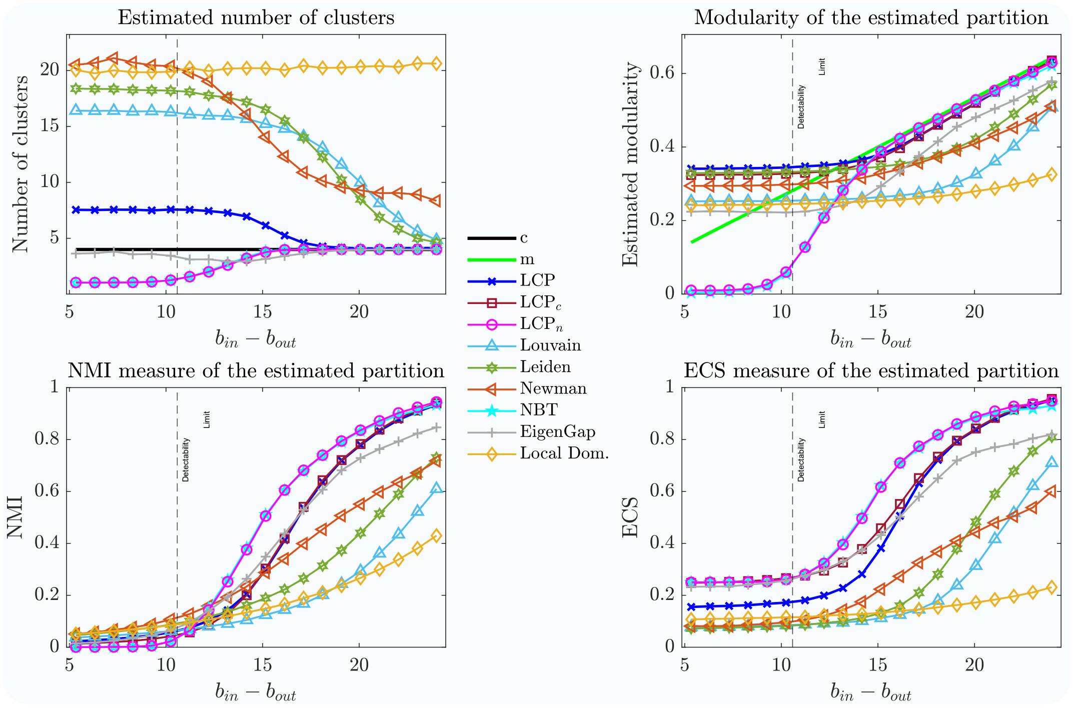

FIG. 3. The estimated number of clusters (upper left-hand figure) and estimated modularity (upper right-hand figure) in SSBM graphs with $N = 500$ nodes, average degree $d_{av} = 7$ and $c = 4$ clusters, for different values of parameters b_{in} and b_{out}. The NMI and ECS measures per each clustering algorithm are provided in the lower left-hand side and lower right-hand side figures, respectively. The vertical dashed line indicates the clustering detectability threshold.

Figure 3 illustrates the clustering performance of our LCP and other considered clustering algorithms for the SSBM network with $c = 4$ communities. The rankings of clustering algorithms in accurately estimating the number of clusters remain consistent with those in the case of SSBM graphs having $c = 2$ communities. We observe that each considered clustering algorithm, in the case of SSBM networks with different $b_{in} - b_{out}$ values below the detectability threshold, tends to reveal the same number of clusters, regardless of the number of planted communities c. The algorithms, namely Eigengap, non-backtracking method, and our LCP$_n$, exhibit the highest precision in estimating the number of communities, followed by LCP. Newman, Louvain, and Leiden methods demonstrate comparable performances, eventually converging to the correct value of c, particularly when the number of inter-community links is relatively low. However, Local Dominance erroneously estimates the community structure irrespective of the number of links connecting nodes from different communities.

Our LCP produces partitions with the highest modularity overall, as depicted in the right-upper part of Figure 3. In this case, knowing the exact number of communities does not benefit LCP from higher modularity, as Figure 3 illustrates from the modularity performance of LCPc. Just above the detectability threshold, our LCP$_\mathrm{c}$ reveals partitions different from the planted one, with the exact number of clusters but of superior modularity. NBT and our LCP$_\mathrm{n}$ achieve modularity of the planted partition as their estimated number of clusters converges to the correct number of communities, outperforming EigenGap consistently when communities are distinguishable from randomness. Leiden and Newman methods exhibit similar performances, followed by Louvain, while the Local Dominance algorithm performs the worst.

The lower part of Figure 3 illustrates the similarity between the estimated community structure of each clustering algorithm and the planted partition in the generated SSBM graph, using the NMI (left lower part) and ECS (right lower part) measures. Based on the NMI and ECS measures, NBT and our LCP$_\mathrm{n}$ exhibit the highest similarity to the original community structure overall for $b_{in} - b_{out}$ values above the detectability threshold. Their performance is followed by our LCP, which produces partitions significantly more similar to the original one than the remaining clustering algorithms considered. However, around and below the detectability threshold, the NMI measure of other clustering algorithms surpasses that of our LCP. This trend can be attributed to the bias of the NMI measure towards a larger number of clusters. Louvain, Leiden, Newman and the Local dominance method estimate more clusters than our LCP, leading to higher NMI measures for lower values of $b_{in} - b_{out}$. As explained in subsection 2.3, the ECS measure effectively mitigates the inherent bias towards a greater number of communities in the NMI measure. Consequently, the lower-right section of Figure 3 consistently demonstrates the superiority of our LCP across the entire range of $b_{in} - b_{out}$ values. Finally, Figure 3 illustrates that our LCP$_\mathrm{c}$ leverages the input of the number of clusters to achieve enhanced performance in the ECS measure, particularly around and below the detectability threshold.

Our LCP demonstrates strong performance in identifying communities within SSBM networks featuring equal expected node degrees, as long as these communities remain distinguishable. However, as the proportion of inter-community links increases, LCP's ability to identify meaningful communities diminishes (despite achieving the highest modularity among considered algorithms).

4.3 *LFR benchmark*

The Lancichinetti-Fortunato-Radicchi (LFR) benchmark, proposed in [19], aims to overcome limitations in the SBM benchmark, including the uniform degree and community size distribution. The LFR benchmark incorporates a power-law degree distribution and a community size distribution inspired by observations in real-world networks. In this subsection, we evaluate the performance of our LCP method and other clustering algorithms on LFR graphs.

Figure 4 illustrates the clustering performance of our LCP and other clustering algorithms considered in a network with $N = 500$ nodes and $c = 5$ communities. These LFR communities were generated with parameters $\beta = 3$ and $\gamma = 2$. In the top left section of the figure, we observe that, overall, our LCP$_\mathrm{n}$ outperforms all other methods in estimating the number of clusters. When the number of inter-community links is high, the NBT method incorrectly estimates the number of communities. Nevertheless, once the clusters become clearly visible in the network, NBT provides the most precise estimate among all

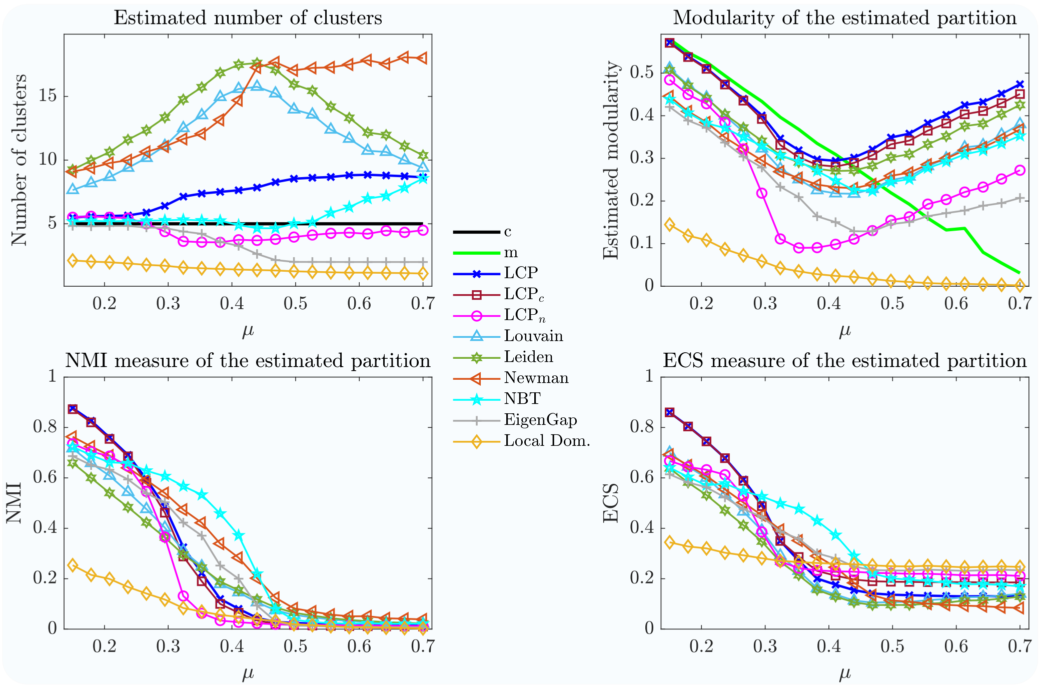

FIG. 4. The estimated number of clusters (upper left) and estimated modularity (upper right) in LFR benchmark graphs with $N = 500$ nodes, an average degree of $d_{av} = 12$, comprising $c = 5$ clusters. The graphs are generated using parameters $\gamma = 2$ and $\beta_{lfr} = 3$ and varying the parameter μ. The lower left and lower right figures display the Normalized Mutual Information (NMI) and Element-centric similarity (ECS) measures for each clustering algorithm.

the methods considered. Our LCP converges to the correct value of c for relatively low μ values, while identifying alternative community clusters elsewhere, which have a higher number of communities. In contrast, the EigenGap method exhibits the opposite trend. It successfully reveals the true number of communities when there are relatively few connections between nodes from different clusters. However, as the ratio of inter-community links increases, EigenGap method converges to a single community. The Louvain and Leiden methods perform similarly, with Leiden consistently providing a more accurate estimate of the true number of communities. Interestingly, both approaches tend to estimate an increasing number of communities as μ increases, until a certain value is reached. After this point, there is a decreasing trend in estimating c, indicating the presence of alternative community structures for large values of μ. We will further discuss this trend when analysing the achieved modularity m. Newman's method estimates an increasing number of communities as μ increases and reaches a saturation point around $c = 16$ for a large ratio of inter-community links. Finally, the Local Dominance method fails to reveal any meaningful community structure across all the considered values of μ.

The top-right portion of Figure 4, evidences that our LCP provides the highest overall modularity of the estimated partition. The initial decreasing trend in the modularity m indicates the augmenting difficulty of accurately estimating the community structure as the ratio of inter-community links. However, for large values of μ, both our LCP and nearly all other clustering algorithms considered manage to recover community structures with higher modularity than that of the original community structure. The deviation from the original partition clearly demonstrates the emergence of an alternative community structures, distinct from the original one, as μ values increase. When the number of communities is known, our LCP$_\mathrm{c}$ performs slightly worse than LCP in terms of modularity but still surpasses the performance of other considered algorithms, thus confirming the superiority of our LCP, where the proposed linear process of relocating nodes on a one-dimensional line successfully reveals communities with excellent accuracy. Leiden, NBT, Louvain, and Newman follow with comparable modularity performance, while our LCP$_\mathrm{n}$ and EigenGap exhibit significantly poorer results. Lastly, Local Dominance exhibits very low modularity m, converging to 0, which aligns with its failure to unveil any meaningful community structure for the majority of μ values.

From both the recorded NMI (bottom-left) and ECS (bottom-right) similarity measures presented in Figure 4, consistent patterns are observed for all the clustering algorithms considered. Whenever our LCP accurately estimates the true number of clusters, LCP exhibits the highest accuracy in recovering the original partition. However, for higher μ values, LCP starts revealing alternative communities, resulting in relatively lower values in both similarity measures. On the other hand, NBT demonstrates the slowest decrease in similarity measures as the ratio of inter-community links increases. Local Dominance performs the worst, while the remaining approaches achieve comparable results in terms of NMI and ECS values.

In case of LFR networks with $c = 5$ communities, the NBT variant of Linear Clustering Process (LCP$_\mathrm{n}$) performs significantly worse in estimating the number of communities compared to other spectral methods, namely NBT and Spectral Gap. This is illustrated in the upper-left section of Figure 5. While LCP generally provides the most precise estimation of the number of clusters c, it does not necessarily generate community partitions that closely resemble the original planted partition in this particular case. This observation is supported by corresponding NMI and ECS similarity measures. As discussed in Section 2.4.1, for Stochastic Block Model (SSBM) networks, when fails to identify the original partition, LCP tends to discover 8 communities, even though the true number of clusters is different. Such a behaviour persists and becomes more pronounced for larger values of the parameter μ, where LCP identifies communities that differ from the planted partition but eventually converges to 8 communities, regardless of the actual number of clusters c. Louvain outperforms both Leiden and Newman method consistently, while Local Dominance estimates communities with worst precision, and failing to discover any community for larger values of μ.

Our LCP and LCP$_\mathrm{c}$ yield partitions with the highest modularity overall, as shown in the upper-right portion of Figure 5. The performance of the other clustering methods considered is comparable to those obtained for LFR networks with $c = 5$ communities. The lower left (right) part of Figure 5 illustrates the similarity measures NMI (ECS) and demonstrates the dominance of our LCP within the range of μ values where it accurately estimates the number of communities. However, for μ values greater than 2.5, the NBT method outperforms our LCP in identifying the correct number of clusters and, consequently, produces partitions more similar to the planted community structures.

Despite the inherent variability of node degrees and community sizes within the LFR benchmark, our LCP demonstrates exceptional performance in identifying communities with the highest modularity among considered algorithms. Even when communities become indistinguishable within the LFR benchmark (i.e. for a relatively large μ value), LCP and other algorithms still discover alternative community structures that exhibit higher modularity than the original graph.

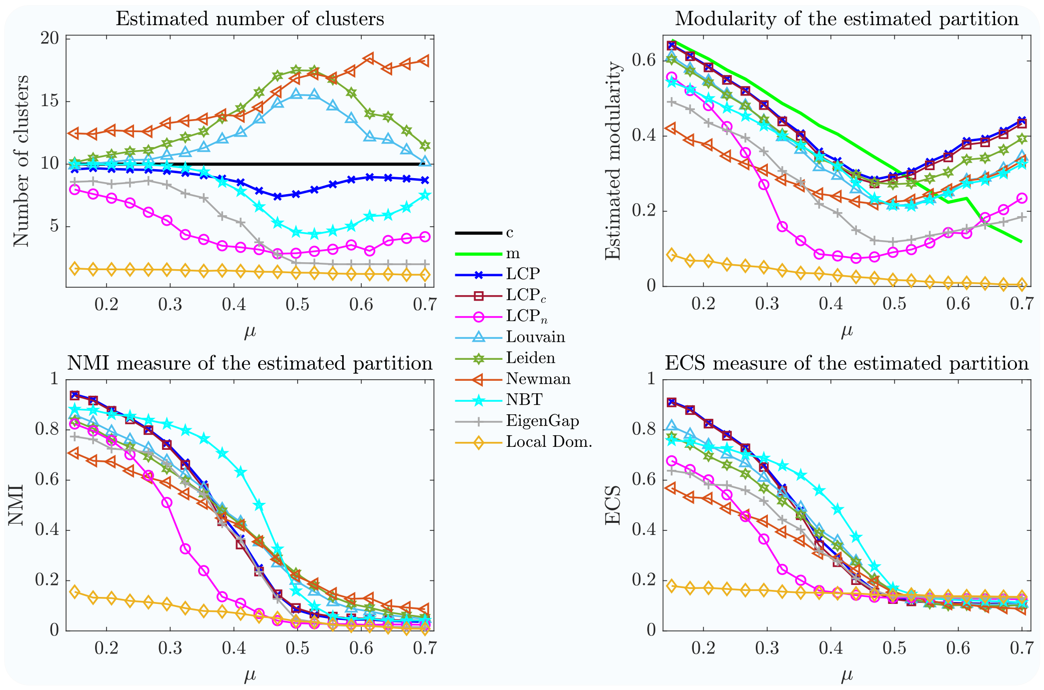

FIG. 5. The estimated number of clusters (upper left) and estimated modularity (upper right) in LFR benchmark graphs with $N = 500$ nodes, an average degree of $d_{av} = 12$, comprising $c = 10$ clusters. The graphs are generated using parameters $\gamma = 2$ and $\beta_{lfr} = 3$ and varying the parameter μ. The lower left and lower right figures display the Normalized Mutual Information (NMI) and Element-centric similarity (ECS) measures for each clustering algorithm.

4.4 *Random Graph models*

Erdős–Rényi random graph [11], denoted as $G(N, p_{ER})$, is a fundamental model in network science, characterised by their random and independent assignment of links between nodes based on a given link probability p_{ER}. Erdős–Rényi random graphs generally lack a clear and well-defined community structure, as demonstrated in [28, p. 630] by the average clustering coefficient, a measure of the ratio of the number of links between a node's neighbours and the maximum possible number of links. Due to the random nature of connections between nodes, the links tend to be evenly distributed across the nodes without observable community structure. However, community detection algorithms can still be applied to uncover potential structures or patterns within these graphs.

The upper-left part of Figure 6 illustrates the estimated number of clusters c by LCP and other considered clustering algorithms on ER graphs with $N = 500$ nodes and varying link density p_{ER}. Apart from the non-backtracking (NBT) method and our LCP$_\text{n}$, which accurately recover the exact number of communities, the remaining algorithms provide incorrect estimates. LCP and Eigengap exhibit similar performance, while Local Dominance and Newman methods perform comparatively poorer in estimating the number of communities but with increased precision as the link density p_{ER} rises. Notably, Louvain and Leiden algorithms estimate a higher number of communities, which increases with the link density p_{ER}.

Regarding the achieved modularity m of the estimated partition, our LCP algorithm surpasses other analysed clustering algorithms for all considered values of link density p_{ER}. Newman and Leiden algorithms rank as the second and third best in terms of modularity performance, while Local Dominance performs the worst. Overall, clustering algorithms reveal a community structure with higher modularity in sparse ER graphs compared to graphs with higher link density. Although ER graphs, on average, do not exhibit a community structure, specific instances of the graph may contain more visible clusters. The probability of finding communities in ER graphs is greater when the graph is sparser, as an increased number of links makes the graph more regular in nature, causing communities to dissipate.

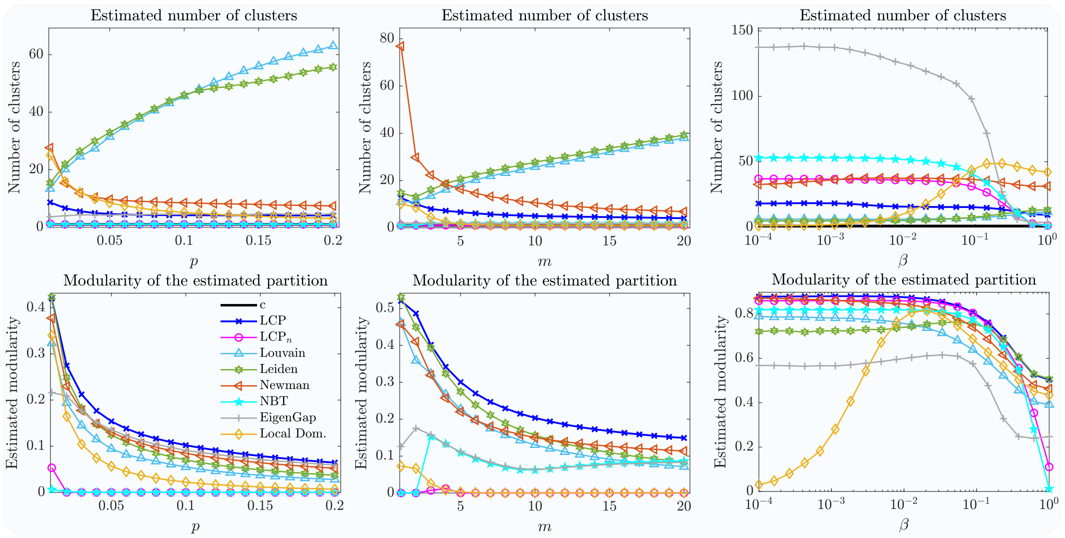

FIG. 6. The estimated number of clusters (upper figures) and estimated modularity (bottom figures) in Erdős-Rényi random graph (left-hand side) with varying link density p_{ER}, Barabási-Albert graph (middle part) with varying parameter m_{BA}, and Watts-Strogatz graph (right-hand side) with varying rewiring probability p_{WS}. The graphs consist of $N = 500$ nodes.

Interestingly, when applied to the Barabási-Albert graphs, the Local Dominance algorithm provides cluster estimates with a precision closely resembling that of the non-backtracking method (NBT) and our LCP$_n$, as shown in the upper middle section of Figure 6. On the other hand, our LCP algorithm estimates a significantly lower number of clusters than the other three algorithms, gradually decreasing the number of discovered clusters as the parameter m_{BA} increases. The Newman method ranks fifth in the estimated number of communities c, also exhibiting a decrease in the number of communities as the parameter m_{BA} increases. In contrast, the Louvain and Leiden algorithms, while optimizing modularity m, estimate an increasing number of clusters c that does not result in the highest modularity m, as evidenced by the LCP performance in the lower middle section of Figure 6. We observe that Local Dominance and our LCP$_n$ achieve the lowest modularity m, followed by NBT and EigenGap. As with ER graphs, our LCP algorithm achieves the highest modularity overall. Newman, Louvain, and Leiden methods perform similarly in modularity m, which decreases as the parameter m_{BA} of the BA model increases.

The Watts-Strogatz graph model, proposed in [30], is derived from a regular ring lattice by randomly rewiring connections while maintaining a fixed number of nodes N and links L. By increasing the rewiring probability p_{WS}, a small-world phenomenon emerges abruptly, as illustrated by the average shortest path hop count on a logarithmic scale in [30, Figure 2]. However, this phenomenon is not observed locally, as demonstrated by the clustering coefficient (defined in [28, page 630]). The right-hand part of Figure 6 the number of clusters c (upper figure) and achieved modularity m (lower figure) of considered clustering algorithms, when applied to the Watts-Strogatz model with $N = 500$ nodes, using different rewiring probabilities p_{WS}.

For low values of p_{WS}, the Local Dominance method fails to detect any community structure, resulting in a low modularity value. As the rewiring probability p_{WS} increases, the Local Dominance method gradually reveals an increasing number of communities with modularity levels comparable to other algorithms. Surprisingly, the EigenGap method underperforms by estimating the highest number of communities c, thus achieving the lowest modularity m values. Overall, the Louvain and Leiden methods estimate the lowest number of communities, leading to lower modularity m values than the remaining methods. Our LCP records fewer communities than the remaining clustering methods while achieving the highest modularity overall. Finally, we highlight that our LCP_n consistently outperforms the non-backtracking (NBT) method.

4.5 *Real-world networks*

Network	N	L	C	M	LCP				LCP_c				Louvain			
					c	m	nmi	ecs	c	m	nmi	ecs	c	m	nmi	ecs
Email-EU	1005	25571	42	0.2880	9	**0.3860**	0.5466	0.2513	-	0.3005	0.5855	0.3066	12	0.3795	0.5530	0.2747
Cora	2708	5429	7	0.6401	25	0.7296	0.3138	0.2121	-	0.6553	0.2102	0.2370	86	0.6775	0.3691	**0.2808**
Citeseer	3264	9072	6	0.5042	65	0.8027	0.1399	0.0737	-	0.7050	0.0710	0.1919	394	0.7722	0.3095	0.1878
Network	N	L	C	M	Leiden				Newman				Local Dominance			
					c	m	nmi	ecs	c	m	nmi	ecs	c	m	nmi	ecs
Email-EU	1005	25571	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	2708	5429	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	3264	9072	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	N	L	C	M	Non-backtracking				LCP_n				Eigengap			
					c	m	nmi	ecs	c	m	nmi	ecs	c	m	nmi	ecs
Email-EU	1005	25571	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	2708	5429	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	3264	9072	6	0.5042	17	0.3595	0.1618	0.1594	16	0.4389	0.1573	0.1581	3	0.1583	0.0960	0.1760

Table 2. Clustering performance of our LCP and considered existing clustering algorithms on real-world networks with ground-truth communities

4.5.1 *Real-world networks with known communities*

In this section, we evaluate the performance of our Linear Clustering Process (LCP) and other clustering algorithms on real-world networks with known[4] ground-truth community structures. The Email_eu network comprises $N = 1005$ nodes and has the highest link density, with a total of $L = 25571$ links. This network forms $c = 42$ communities. Our LCP identifies a partition with the highest modularity, while the Leiden algorithm estimates more accurately the number of clusters, resulting in a partition that closely aligns with the ground truth. The Cora network consists of $N = 2708$ nodes interconnected by $L = 5429$ links, forming $c = 7$ communities. The Leiden algorithm produces a community structure with the highest modularity, closely followed by our LCP. However, our LCP excels in estimating the number of communities compared to Leiden. Finally, the Citeseer network exhibits the smallest number of clusters ($c = 6$). While the Newman algorithm identifies a partition with the highest modularity, its estimated number of communities deviates significantly from the correct value.

[4]We acknowledge that the identified communities in considered real-world networks represent one possible grouping. Ground truth data for communities is often subjective and depends on the specific attributes used.

Based on the results presented in Table 2, we note a significant sensitivity of the NMI and ECS similarity measures to the estimated number of clusters. The NMI measure exhibits a bias towards partitions with a larger number of communities. Conversely, for the Citeseer network, we observe that the ECS measure of both the Louvain and Eigengap algorithms is similar, despite their substantially different estimates of the number of clusters.

4.5.2 *Real-world networks without known communities* Table 3 provides a summary of the clustering performance of our LCP algorithm and other clustering algorithms from existing literature on real-world networks where the ground-truth community structure is unknown. The network sizes range from $N = 34$ (Karate networks) to $N = 1589$ (Co-authorship network). Among the seven networks considered, our LCP algorithm achieves the highest modularity m in four cases, while it ranks as the second best in the Dolphins and Football networks. In one case, specifically the Co-authorship network, it ranks as the fourth best algorithm in terms of modularity.

Network	N	L	LCP		Louvain		Leiden		Newman	
			c	m	c	m	c	m	c	m
Karate club	34	78	3	**0.3922**	4	0.3565	4	0.3729	5	0.3776
Dolphins	62	159	4	0.5057	4	0.4536	5	**0.5105**	6	0.4894
Polbooks	105	441	3	**0.5160**	4	0.4897	4	0.5026	8	0.4160
Football	115	613	7	0.5894	7	0.5442	7	0.5635	11	0.4623
Facebook	347	2519	8	**0.4089**	16	0.3726	18	0.3792	23	0.3770
Polblogs	1490	19090	19	**0.4224**	7	0.3385	11	0.3117	4	0.3459
Co-authorship	1589	2742	40	0.9296	272	0.9423	270	0.9410	28	0.7393
Network	N	L	Local Dominance		NBT		LCP$_n$		Eigengap	
			c	m	c	m	c	m	c	m
Karate club	34	78	2	0.3123	2	0.3715	1	0.0000	2	0.2780
Dolphins	62	159	3	0.3620	2	0.3698	2	0.3698	2	0.3115
Polbooks	105	441	2	0.4451	3	0.5085	2	0.4546	2	0.4167
Football	115	613	6	0.3205	10	**0.5939**	5	0.5522	11	0.5927
Facebook	347	2519	8	0.2067	8	0.3638	7	0.3544	2	0.2836
Polblogs	1490	19090	3	0.2799	8	0.2149	5	0.3480	2	0.2679
Co-authorship	1589	2742	277	**0.9431**	23	0.5005	17	0.5806	2	0.1288

Table 3. Clustering performance of our LCP and considered existing clustering algorithms on real-world networks without ground-truth communities

5. Conclusion

In this study, we conducted a comprehensive analysis of the clustering performance of the recently introduced Linear Clustering Process (LCP) on networks. We compared LCP with several widely used clustering algorithms that employ different techniques for estimating partitions, including modularity optimization and spectral methods. The results of our simulations demonstrate that in the majority of cases, both for synthetic and real-world networks, LCP generates communities with significantly higher modularity. Specifically, when the underlying clusters are clearly discernible, LCP successfully recovers partitions that closely resemble the true clusters. Moreover, even in situations where clusters are not easily distinguishable from random patterns, LCP is able to identify alternative clusters with superior modularity. Overall, our findings highlight the effectiveness of LCP in achieving highly modular and meaningful community structures in diverse network settings.

Acknowledgment

I. Jokić was funded by NExTWORKx, a collaboration between TU Delft and KPN. P. Van Mieghem has been funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 101019718).

References

1. Angel, O., Friedman, J. & Hoory, S. (2015) The non-backtracking spectrum of the universal cover of a graph. *Transactions of the American Mathematical Society*, **367**(6), 4287–4318.

2. Barabási, A.-L. (2013) Network science. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, **371**(1987), 20120375.

3. Barabási, A.-L. & Albert, R. (1999) Emergence of scaling in random networks. *science*, **286**(5439), 509–512.

4. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. (2008) Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, **2008**(10), P10008.

5. Brandes, U., Delling, D., Gaertler, M., Gorke, R., Hoefer, M., Nikoloski, Z. & Wagner, D. (2007) On modularity clustering. *IEEE transactions on knowledge and data engineering*, **20**(2), 172–188.

6. Budel, G. & Van Mieghem, P. (2020) Detecting the number of clusters in a network. *Journal of Complex Networks*, **8**(6), cnaa047.

7. Clauset, A., Shalizi, C. R. & Newman, M. E. (2009) Power-law distributions in empirical data. *SIAM review*, **51**(4), 661–703.

8. Danon, L., Diaz-Guilera, A., Duch, J. & Arenas, A. (2005) Comparing community structure identification. *Journal of statistical mechanics: Theory and experiment*, **2005**(09), P09008.

9. Decelle, A., Krzakala, F., Moore, C. & Zdeborová, L. (2011a) Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, **84**(6), 066106.

10. Decelle, A., Krzakala, F., Moore, C. & Zdeborová, L. (2011b) Inference and phase transitions in the detection of modules in sparse networks. *Physical Review Letters*, **107**(6), 065701.

11. Erdős, P., Rényi, A. et al. (1960) On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, **5**(1), 17–60.

12. Fortunato, S. (2010) Community detection in graphs. *Physics reports*, **486**(3-5), 75–174.

13. Gates, A. J., Wood, I. B., Hetrick, W. P. & Ahn, Y.-Y. (2019) Element-centric clustering comparison unifies overlaps and hierarchy. *Scientific reports*, **9**(1), 8574.

14. Holland, P. W., Laskey, K. B. & Leinhardt, S. (1983) Stochastic blockmodels: First steps. *Social networks*, **5**(2), 109–137.

15. Horn, R. A. & Johnson, C. R. (2012) *Matrix analysis*. Cambridge university press.

16. Jokić, I. & Van Mieghem, P. (2023) Linear Clustering Process on Networks. *IEEE Transactions on Network Science and Engineering*.

17. Krzakala, F., Moore, C., Mossel, E., Neeman, J., Sly, A., Zdeborová, L. & Zhang, P. (2013) Spectral redemption in clustering sparse networks. *Proceedings of the National Academy of Sciences*, **110**(52), 20935–20940.

18. Lancichinetti, A. & Fortunato, S. (2009) Community detection algorithms: a comparative analysis. *Physical review E*, **80**(5), 056117.

19. Lancichinetti, A., Fortunato, S. & Radicchi, F. (2008) Benchmark graphs for testing community detection algorithms. *Physical review E*, **78**(4), 046110.

20. Muchnik, L., Pei, S., Parra, L. C., Reis, S. D., Andrade Jr, J. S., Havlin, S. & Makse, H. A. (2013) Origins of power-law degree distribution in the heterogeneity of human activity in social networks. *Scientific reports*, **3**(1), 1783.

21. Newman, M. (2018) *Networks*. Oxford university press.

22. Newman, M. E. (2006) Modularity and community structure in networks. *Proceedings of the national academy of sciences*, **103**(23), 8577–8582.

23. Newman, M. E. & Girvan, M. (2004) Finding and evaluating community structure in networks. *Physical review E*, **69**(2), 026113.

24. Palla, G., Derényi, I., Farkas, I. & Vicsek, T. (2005) Uncovering the overlapping community structure of complex networks in nature and society. *nature*, **435**(7043), 814–818.

25. Patania, A., Allard, A. & Young, J.-G. (2023) Exact and rapid linear clustering of networks with dynamic programming. *arXiv preprint arXiv:2301.10403*.

26. Shang, F., Chen, B., Expert, P., Lü, L., Yang, A., Stanley, H. E., Lambiotte, R., Evans, T. S. & Li, R. (2022) Local dominance unveils clusters in networks. *arXiv preprint arXiv:2209.15497*.

27. Traag, V. A., Waltman, L. & Van Eck, N. J. (2019) From Louvain to Leiden: guaranteeing well-connected communities. *Scientific reports*, **9**(1), 5233.

28. Van Mieghem, P. (2014) *Performance analysis of complex networks and systems*. Cambridge University Press.

29. Van Mieghem, P., Ge, X., Schumm, P., Trajanovski, S. & Wang, H. (2010) Spectral graph analysis of modularity and assortativity. *Physical Review E*, **82**(5), 056113.

30. Watts, D. J. & Strogatz, S. H. (1998) Collective dynamics of ‘small-world’networks. *nature*, **393**(6684), 440–442.

A. List of used notations

Notation	Explanation
G	Graph
$\mathscr{N}$	Set of N nodes of graph G
$\mathscr{N}_i$	Set of neighboring nodes of node i
$\mathscr{L}$	Set of L links of graph G
N	Number of nodes in graph G
L	Number of links in graph G
A	Adjacency matrix of graph G
a_{ij}	ij-th element of adjacency matrix A
d	Degree vector
d_i	Degree of node i
Δ	Degree diagonal matrix
u	$N \times 1$ all-one vector
m	Modularity
C	Cluster matrix
c	Number of clusters
n_i	Number of nodes in cluster i
F	Confusion matrix
$I_n(P_0, P_e)$	Normalized mutual information metric
P_0	Known partition
P_e	Estimated partition
c_0	Number of known clusters
c_e	Number of estimated clusters
p_{ij}	Personalized PageRank
δ	Kronecker delta function
c_γ, c_β	Cluster
v_i, v_j	Element of corresponding cluster
α_{ecs}	Restart probability constant
$\mathscr{A}, \mathscr{B}$	Clustering

Notation	Explanation
$S(\mathscr{A},\mathscr{B})$	Element-centric similarity score
p_{in}, b_{in}	Intra-community link probability
p_{out}, b_{out}	Inter-community link probability
$E[D]$	Expected degree
γ	Power-law exponent of degree distribution
β_{lfr}	Power-law exponent of community size distribution
μ	Inter-community link probability
d_{av}	Average degree
α	Attraction strength
δ	Repulsion strength
$x_i[k]$	Position of node i at time k
k	Discrete time
I	$N \times N$ identity matrix
W	$N \times N$ topology-based matrix
β	$N \times 1$ eigenvalue vector
β_i	Eigenvalue i
Y	$N \times N$ orthogonal eigenvector matrix
y_i	Eigenvector i
$\hat{y}_i$	Sorted eigenvector i
B	$2L \times 2L$ non-backtracking matrix
B^*	$2N \times 2N$ matrix
O	$N \times N$ all-zero matrix
$f_D(d)$	Probability density function of degree d
c_p	Normalization constant
p_{ER}	Link density
m_{BA}	Number of nodes connected by the new node
p_{WS}	Rewiring probability
y	Cluster membership
M	$N \times N$ modularity matrix
z_i	i-th eigenvector
ζ_i	i-th eigenvalue
$\tilde{A}$	$N \times N$ weighted adjacency matrix
Δm	Modularity gain
$\sum_{\text{in}}$	Sum of the weights of intra-community links in community h
$\sum_{\text{tot}}$	Sum of the weights of all links in G incident to any node in community h
$\lambda(A)$	Eigenvalues of the adjacency matrix A
$\lambda(M)$	Eigenvalues of the modularity matrix M

Table A.4: Notations used in the paper

B. Considered clustering algorithms

B.1 *Louvain method*

The Louvain method, proposed by Blondel *et al.* [4], is a powerful yet straightforward heuristic clustering algorithm. This method employs an iterative and unsupervised two-step procedure to optimize modularity, denoted as m. Initially, a directed graph G with an $N \times N$ weighted adjacency matrix M is partitioned into N clusters, where each node constitutes its own cluster or community.

In the first stage, the algorithm evaluates the change in graph modularity m if node i were to be assigned to the community of its neighboring node $j \in \mathscr{N}i$. The modularity gain Δm resulting from assigning node i to community h of adjacent node j is defined in [4] as follows:

$$\begin{aligned}\Delta m = &\left(\frac{\sum_{\text{in}} + 2\sum_{l:C_{lj}=1} M_{il}}{2L} - \left(\frac{\sum_{\text{tot}} + d_i}{2L}\right)^2\right) - \\ &\left(\frac{\sum_{\text{in}}}{2L} - \left(\frac{\sum_{\text{tot}}}{2L}\right)^2 - \left(\frac{d_i}{2L}\right)^2\right),\end{aligned} \tag{A.1}$$

where $\sum_{\text{in}}$ represents the sum of the weights of intra-community links in community h, and $\sum_{\text{tot}}$ denotes the sum of the weights of all links in G incident to any node in community h. Node i is assigned to the community that yields the largest positive gain in modularity m. If all computed gains Δm are either negative or smaller than a predefined small positive threshold value, node i remains in its original community. The first stage concludes when modularity m can no longer be increased by re-assigning nodes to neighboring communities.

In the second stage of each iteration, the weighted graph obtained from the first stage is transformed into a new weighted graph, where each node represents a community. The link weight between two nodes h and g is equal to the sum of weights of all links between communities h and g in the graph obtained from the first stage. Moreover, the weight of a self-loop of node g in the new graph equals the sum of weights of all intra-community links in cluster g of the graph from the previous stage. This new graph is then fed into the first stage for the next iteration. The algorithm terminates when modularity m can no longer be increased. The time complexity of the Louvain method is linear with the number of links $\mathscr{O}(L)$ for sparse graphs [4].

B.2 *Leiden method*

Although the Louvain method is widely used in clustering algorithms, it has a drawback of identifying poorly connected or disconnected communities. This limitation was first identified by Traag *et al.*, who introduced the Leiden algorithm as an improvement to the Louvain method in their work [27]. The Leiden algorithm aims to estimate graph partitions while ensuring the creation of connected communities. The Leiden algorithm involves three iterative steps:

1 Local moving of nodes: This step is an enhanced version of the first step in the Louvain algorithm, described in Equation (A.1). While the Louvain algorithm randomly visits each node until modularity can no longer be improved by moving a node to a different community, the Leiden algorithm only visits nodes whose adjacent nodes have been relocated. This is achieved by placing nodes in a queue and iteratively checking if the cluster membership of a node can be updated to enhance the quality function. When a node is moved to another community, its neighbors from other communities are placed in the queue.

2 Refinement of the partition: In this step, each node is initially assigned its own community. Nodes are merged locally, meaning only within communities estimated in the previous stage. Two nodes within the same community are merged only if both nodes are well connected to the community from the previous stage. At the end of the refinement stage, partitions from the first stage are often split into multiple communities.

3 Aggregation of the network: This step involves aggregating the network based on the refined partition obtained from the previous stage, similar to the second stage of the Louvain algorithm.

The Leiden algorithm achieves faster clustering compared to the Louvain algorithm while generally providing improved partitions [27].

B.3 *Newman method*

Newman introduced a clustering algorithm based on modularity optimization [22]. The algorithm begins by estimating the bisection of a graph, aiming to generate the highest modularity value m using Equation (2.1), which can be expressed as:

$$m = \frac{1}{4L} y^T \cdot M \cdot y, \tag{A.2}$$

where, the $N \times 1$ vector y represents the cluster membership of each node, with values of either 1 or -1. The $N \times N$ modularity matrix $M = A - \frac{1}{2L} \cdot d \cdot d^T$ can be decomposed into its eigenvalues and eigenvectors as:

$$M = \sum_{i=1}^{N} \zeta_i \cdot z_i \cdot z_i^T, \tag{A.3}$$

where, the $N \times 1$ eigenvector z_i corresponds to the i-th eigenvalue ζ_i. The vector $y = \sum_{j=1}^{N} (z_j^T \cdot y) \cdot z_j$ can be expressed as a linear combination of the eigenvectors $z_{i 1 \leqslant i \leqslant N}$, transforming Equation (A.2) into:

$$m = \frac{1}{4L} \sum_{i=1}^{N} \zeta_i \cdot (z_j^T \cdot y)^2. \tag{A.4}$$

To maximize the modularity m, Newman proposed setting $y_i = 1$ if $(z_1)i > 0$, otherwise $y_i = -1$. This procedure is repeated in subsequent iterations by dividing the graph into two partitions based on spectral properties. However, considering only the block sub-matrix of M corresponding to the cluster g in the next iteration would ignore inter-community links. Instead, for the estimated cluster g, the modularity matrix M_g is updated using:

$$M_g = m_{ij} - \left(\sum_{k \in g} m_{ik} \right) \cdot \delta_{ij}, \tag{A.5}$$

Here, the Kronecker delta δ_{ij} is 1 if $i = j$, and 0 otherwise. The algorithm terminates when further improvement in modularity m is no longer possible.

B.4 *Eigengap method*

Besides modularity as a quality function, Newman[22] also defined a $N \times N$ real symmetric matrix M with elements

$$M_{ij} = A_{ij} - \frac{d_i d_j}{2L} \tag{A.6}$$

which called the modularity matrix. The estimation of the number of clusters c is determined by the maximum eigengap of the modularity matrix M, following the same process described for the adjacency matrix A. The eigenvalues of the modularity matrix M can firstly be sorted in descending order $\lambda_1(M) \geqslant \lambda_2(M) \geqslant \cdots \geqslant \lambda_N(M)$. Hence, it can be observed that the eigenvalues of the modularity matrix M and the adjacency matrix A are interlaced[6]:

$$\lambda_1(A) \geqslant \lambda_1(M) \geqslant \lambda_1(A) \geqslant \lambda_2(M) \geqslant \cdots \geqslant \lambda_1(A) \geqslant \lambda_N(M) \tag{A.7}$$

Finally, the number of clusters c could be estimated by

$$c = \arg\max_i (\lambda_{i-1}(M) - \lambda_i(M)), \qquad i = 2, \ldots, N \tag{A.8}$$

where $\lambda_{i-1}(M) - \lambda_i(M)$ denotes the difference of two consecutive eigenvalues of modularity matrix M and estimated number of clusters c is equal to the index number of smaller eigenvalue in the maximum eigengap.

B.5 *Local Dominance method*

Shang *et al.* proposed the Local Dominance[26] algorithm, designed to reveal the hidden hierarchy in the network by utilizing local information. Local Dominance starts with calculating the degree of each node. For a given node, a link pointing to an adjacent node is connected if the following criteria are met: the degree of the adjacent node is greater than or equal to the original node, and the degree of the adjacent node is the largest among all neighbors. Loops are not permitted in this process, meaning that if a link already exists between two nodes, a second link in the opposite direction cannot be established.

Upon traversing all nodes, if any nodes possess multiple outgoing links, one is randomly preserved. Consequently, the algorithm transforms the initial network into a collection of tree structures, with each tree's root node referred to as the local leader and the leaf nodes as followers. In the context of Local Dominance, each tree represents a community. The total number of communities corresponds to the quantity of local leaders, and a local leader, along with their followers, constitutes the members of a community.

The time complexity of Local Dominance is linear in the number of links $O(L)$[26], which could be one of the fastest community detection algorithms.

C. Pseudo-code of the Linear Clustering Process on Networks

D. SSBM benchmark

In contrast to the SSBM networks with $c = 2$ and $c = 4$ clusters, the modularity of the estimated partition by each considered algorithm for the range of $b_{in} - b_{out}$ values above the detectability limit are lower than the modularity m of the planted partition for $c = 8$, as depicted in the upper right part of Figure A.7. This trend indicates that, as the number of clusters c increases, the planted partition possesses the highest modularity among all possible partitions on a given graph.

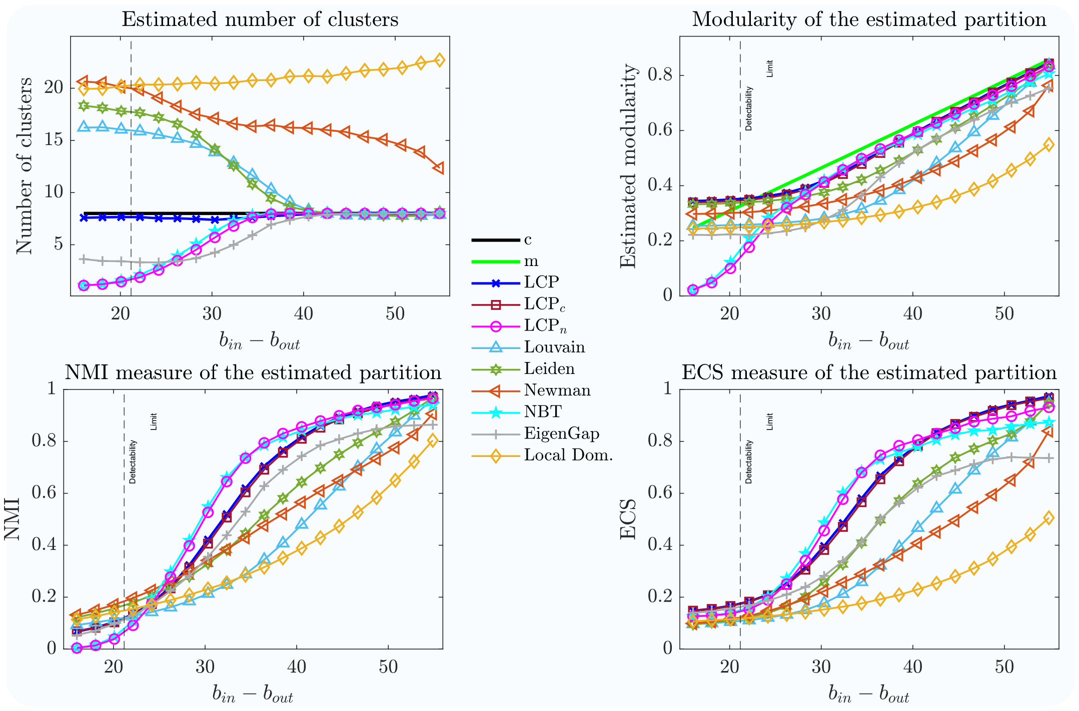

FIG. A.7. The estimated number of clusters (upper left-hand figure) and estimated modularity (upper right-hand figure) in SSBM graphs with $N = 500$ nodes, average degree $d_{av} = 7$ and $c = 8$ clusters, for different values of parameters b_{in} and b_{out}. The NMI and ECS measures per each clustering algorithm are provided in the lower left-hand side and lower right-hand side figures, respectively. The vertical dashed line indicates the clustering detectability threshold.

E. LFR benchmark

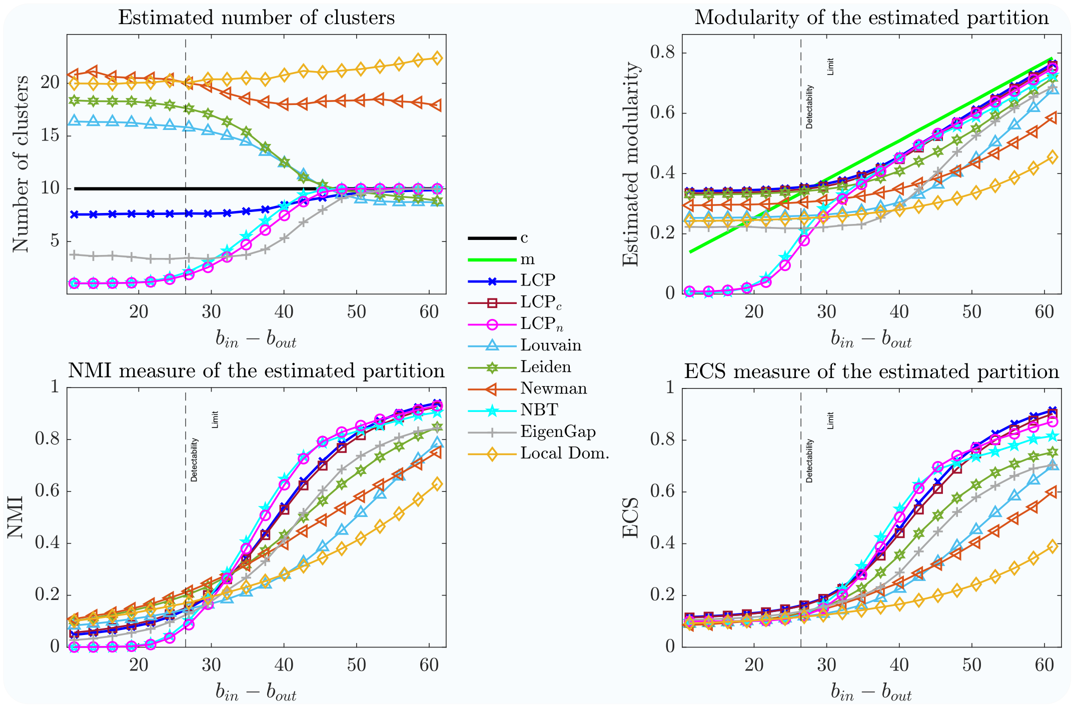

FIG. A.8. The estimated number of clusters (upper left-hand figure) and estimated modularity (upper right-hand figure) in SSBM graphs with $N = 500$ nodes, average degree $d_{av} = 7$ and $c = 10$ clusters, for different values of parameters b_{in} and b_{out}. The NMI and ECS measures per each clustering algorithm are provided in the lower left-hand side and lower right-hand side figures, respectively. The vertical dashed line indicates the clustering detectability threshold.

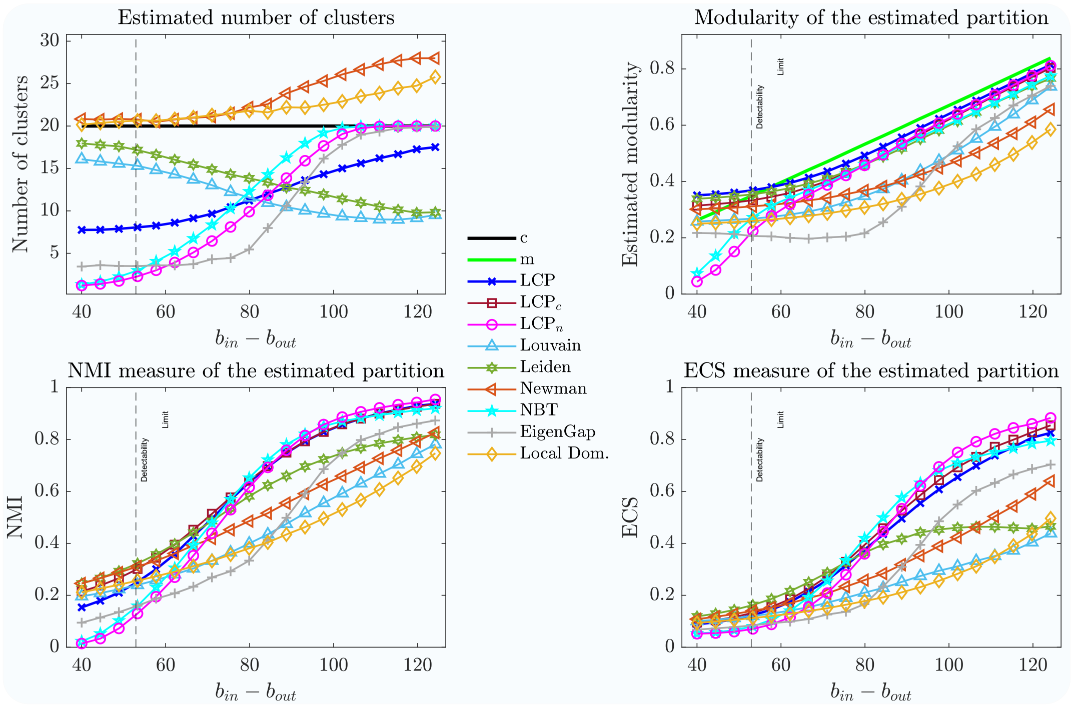

FIG. A.9. The estimated number of clusters (upper left-hand figure) and estimated modularity (upper right-hand figure) in SSBM graphs with $N = 500$ nodes, average degree $d_{av} = 7$ and $c = 20$ clusters, for different values of parameters b_{in} and b_{out}. The NMI and ECS measures per each clustering algorithm are provided in the lower left-hand side and lower right-hand side figures, respectively. The vertical dashed line indicates the clustering detectability threshold.

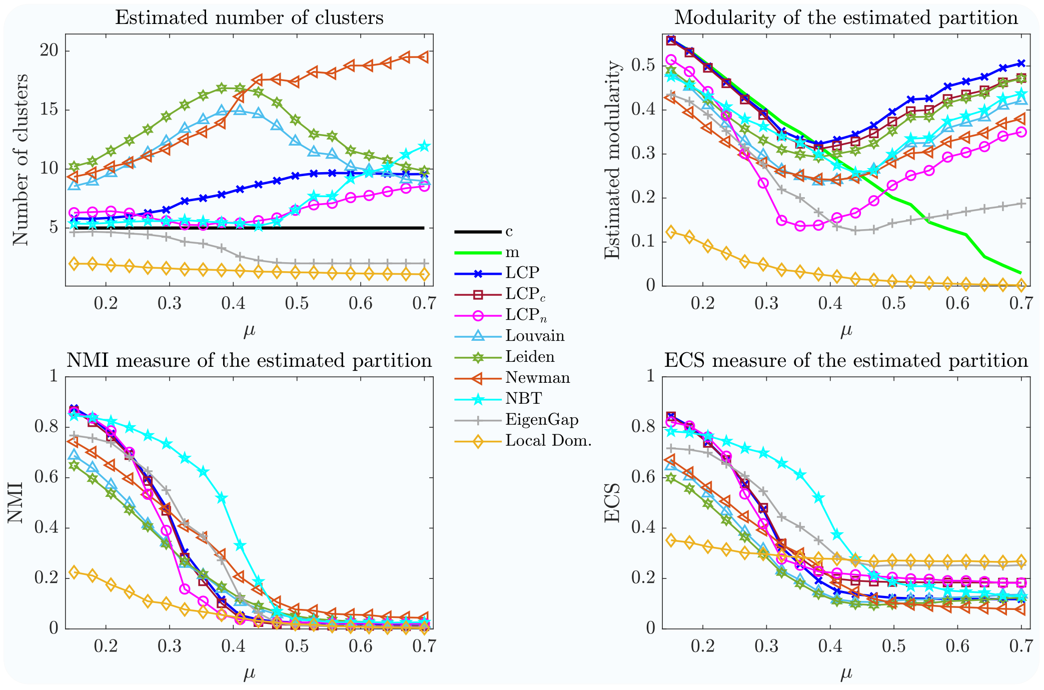

FIG. A.10. The estimated number of clusters (upper left) and estimated modularity (upper right) in LFR benchmark graphs with $N = 500$ nodes, an average degree of $d_{av} = 12$, comprising $c = 5$ clusters. The graphs are generated using parameters $\gamma = 2.5$ and $\beta_{lfr} = 2.5$ and varying the parameter μ. The lower left and lower right figures display the Normalized Mutual Information (NMI) and Element-centric similarity (ECS) measures for each clustering algorithm.

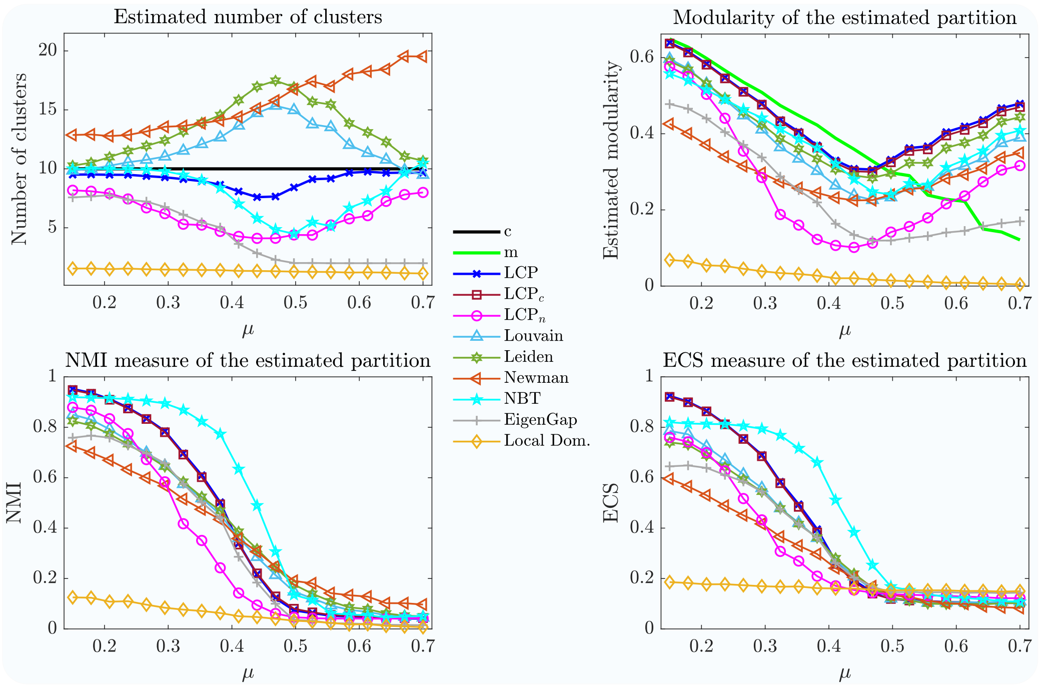

FIG. A.11. The estimated number of clusters (upper left) and estimated modularity (upper right) in LFR benchmark graphs with $N = 500$ nodes, an average degree of $d_{av} = 12$, comprising $c = 10$ clusters. The graphs are generated using parameters $\gamma = 2.5$ and $\beta_{lfr} = 2.5$ and varying the parameter μ. The lower left and lower right figures display the Normalized Mutual Information (NMI) and Element-centric similarity (ECS) measures for each clustering algorithm.