Supplementary Materials for

Local dominance unveils clusters in networks

Dingyi Shi, Fan Shang, Bingsheng Chen, Paul Expert, Linyuan Lü,
H. Eugene Stanley, Renaud Lambiotte, Tim S. Evans, Ruiqi Li

Supplementary Notes

Supplementary Note 1: The Local Search (LS) algorithm

Supplementary Note 1.1: Identification of local leaders

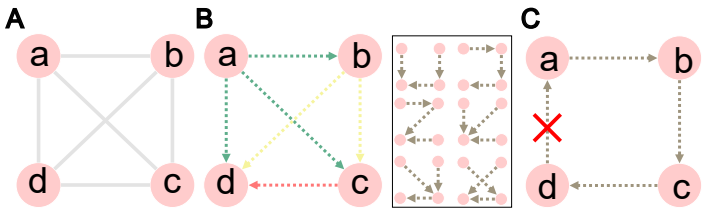
As mentioned in the main text, when there are multiple nearest nodes of a larger or equal degree than the ego node i, then i will first point to all of them. Then, after traversing all nodes, for each node we keep one out-going edge, chosen at random, which destroys any loops (see Lemma 1 and Supplementary Fig. 1C) and gives us a tree. Such a setting gives good partitions even for some extreme cases (e.g., a lattice or clique).

Supplementary Note 1.1.1: Strictly homogeneous networks

When there is a complete subgraph in the neighborhood of a node (see Supplementary Fig. 1A), the mapping of local dominance can vary (see Supplementary Fig. 1B) due to randomness, but the different mappings all have the same eventual partition. In a lattice network, following the same process of the LS algorithm, the whole network will be classified into just one community.

Lemma 1 ***No formation of loops by the LS algorithm.*** *The mapping in Supplementary Fig. 1C is impossible.*

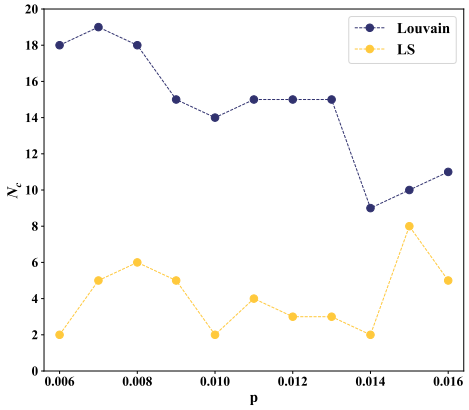
Proof*: If $a \to b$, $b \to c$, and $c \to d$, then it means that $k_d \geq k_c \geq k_b \geq k_a$. Now, if $d \to a$, it indicates $k_a \geq k_d$, then we have $k_a = k_b = k_c = k_d$ and $A_{ad} = 1$. But if so, given $A_{ad} = 1$ and $k_a = k_d$, then as a is first traversed, there will be $a \to d$ instead of $d \to a$, and eventually, a will only keep one out-going link, thus no such loop shown in Supplementary Fig. 1C will be formed.* ■



Supplementary Figure 1. The mapping of local dominance in a complete network. (**A**) The structure of a complete graph, and here we assume that nodes will be traversed from a to d in order (any other orders for the nodes lead to the same result). (**B**) Node a will first point to all three other nodes as $k_j \geq k_a$, then node b will point to c and d but not a (as a is already a possible follower of b), then c will point to d (but not a and b, as they are also possible followers of c), and d points to no one as there is no remaining neighbor with $k_j \geq k_d$. Then we randomly keep only one out-going link for each node. For example, b can either point to c or d with an equal probability. The eventual mapping of local dominance can be either a chain or a star in two extremes. For such a 4-clique, there are 6 possible mappings (see the Inset on the right), and all of them yield just one community. (**C**) No such a loop will be formed, see proof of the Lemma 1.

Supplementary Note 1.1.2: Relatively homogeneous ER random networks

In relatively homogeneous networks (e.g., ER random networks), degree differences exist, and communities may emerge. In such networks, large degree nodes often have a relatively higher probability connecting to each other, which will lead to fewer local leaders and eventually fewer communities. In the thermodynamic limit, an ER network is regarded as having no community structure (or the whole network should be classified into one community) due to a strong homogeneity. When implementing the LS and Louvain algorithm on ER networks, the LS method obtains fewer communities, which is closer to the common assumption (see Supplementary Fig. 2).



Supplementary Figure 2. The number of communities detected by the Louvain and LS methods in ER random networks with $N = 1000$. The LS method detects fewer communities than the Louvain algorithm. For ER random networks with 10,000 nodes and p equals 0.001, the LS method ends up with 15 communities and the Louvain with 22 communities.

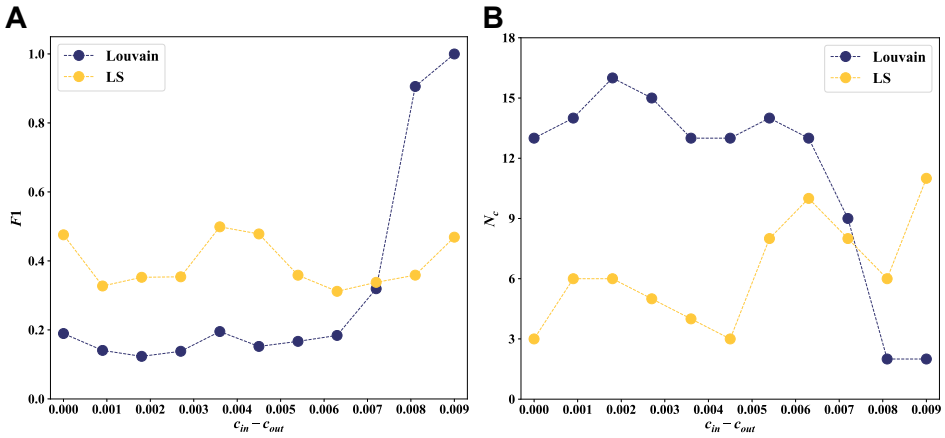
For random networks generated by stochastic block model (SBM) [1, 2] that comprise two communities, each of which are of 1,000 nodes (i.e., $N = 2000$) with intra-connection probability $c_{in} = 0.009$, when the inter-connection probability $c_{out} = 0$, the LS algorithm detects 9 communities and the Louvain algorithm detects two communities. The result obtained by the Louvain algorithm aligns with ground truth, but also reflects the resolution limit [3], and in contrast, the LS algorithm still detects as many as community centers as when looking at each individual ER network. When we fix c_{in} and gradually increase c_{out}, the F_1-score of the LS algorithm is roughly around 0.4, while the Louvain algorithm has a fast decrease, which is mainly influenced by internal connections and the number of communities detected.

Supplementary Note 1.1.3: Heterogeneous Ravasz-Barabási networks

In a classical Ravasz-Barabási networks (see Supplementary Fig. 4A) that exhibits strong heterogeneity, we can easily identify local leaders and followers. Nodes b, c, d, and e (with $k_i = 20$, $l_i = 2$) are identified as local leaders by the LS algorithm (see Supplementary Fig. 4C), as they are of a larger degree than all their neighbors (other neighbors are of a degree $k_i \in [3, 5]$) and are relatively far away from a larger node (here, only the central node a is of a larger degree ($k_a = 84$) and is two steps away from them). For example, those four nodes surrounding c are not connected to node a, thus they point to c, but other nodes in the four 2-level peripheral clusters (e.g., those four clusters surrounding the cluster centered at c) are all also directly connected to the central node a, as $k_a > k_c$ and $d_{ia} = d_{ic} = 1$, thus those nodes all point to a and are eventually partitioned to the community centered at a. As for the Louvain algorithm, the partition results strongly depend on the traversal sequence, and it maximizes modularity in a greedy way if the modularity gain $\Delta M_{c_i c_j}$ is positive.

$$\Delta M_{c_i c_j} = \frac{1}{E}\left(E_{c_i c_j} - \frac{k_{c_i} k_{c_j}}{2E}\right), \tag{1}$$

where E is the number of edges in the network ($E = 344$ in the Ravasz-Barabási network in Supplementary Fig. 4A), $E_{c_i c_j}$ is the number of edges that connect nodes in community c_i and community



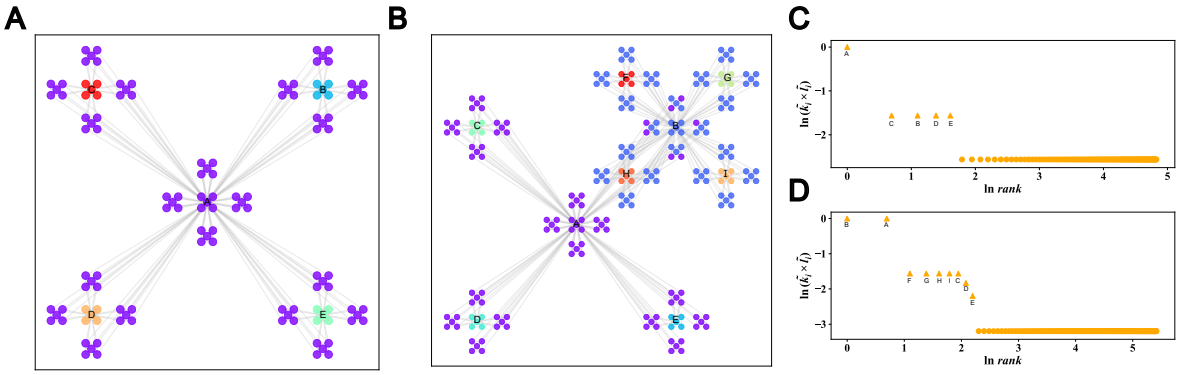
Supplementary Figure 3. The comparison between the Louvain algorithm and our LS method on synthesized networks by stochastic block model (SBM). (**A**) The F_1-score and (**B**) the number of communities N_c detected by the LS and Louvain algorithms. We generate networks with two communities by SBM, each of which has 1,000 nodes. c_{in} = 0.009 is the density of edges between nodes the same group, and c_{out} is the density of edges between nodes from different groups. We gradually increase c_{out} until it equals c_{in}.

c_j, and k_{c_i} denotes the sum of degree of all nodes in community i. For example, as shown in Fig. 2F in the main text, the small cluster (denote as c_1) will not be merged to the purple one (denote as c_2), as $E_{c_1c_2} = 4$, and $k_{c_1} = 4 \times 5 = 20$, $k_{c_2} = 84 + 3 \times 4 + 4 \times 5 \times 3 = 156$, in this case, $\Delta M_{c_1c_2} < 0$, and these two communities will not be merged together.

When we further modify the network in Supplementary Fig. 4A to add a 3-level branching to the top-right 2-level cluster centered at b (see Supplementary Fig. 4B), then the degree of node b become the same as the original central node a (i.e., $k_a = k_b = 84$). As all those four 2-level peripheral clusters surrounding node b are both connected to both a and b, thus, they will eventually point to one of them at random (in this realization, there are seven nodes point to a and eventually colored as purple, and the remaining nodes point to b and colored as blue). Again, due to the same reason as in Supplementary Fig. 4A, four smaller new communities emerge, which are centered at nodes f, g, h, and i, respectively (see Supplementary Fig. 4D). As we remove 4 links from node d, and 8 links from node e, thus $k_d = 16$ and $k_e = 12$, and this is why in the decision graph in Supplementary Fig. 4D, nodes d and e drops a little bit each. In addition, the decision graph can also make meaningful ranking for community centers (see Supplementary Fig. 4D). The network can be partitioned into nine communities at the finest resolution (see Supplementary Fig. 4B), or into two big communities (small communities centered at c, d, and e will follow a, and other four small communities follow b as a is more further away from them), which reswembles a multiscale community structure.

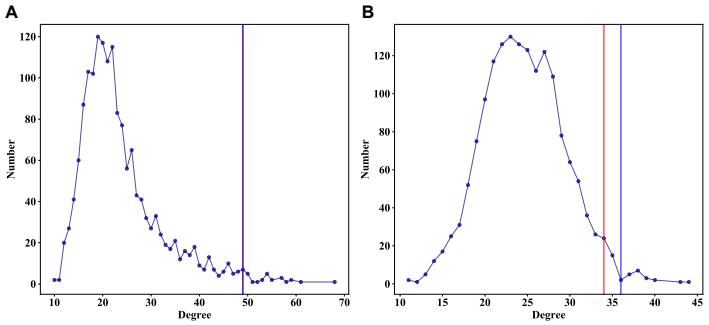
Supplementary Note 1.1.4: Multiscale community structure

To further verify the impacts of homogeneity on the LS method for detecting multiscale community structure, we obtain the degree frequency distribution for the scale-free setting (see Fig. 3A in the main text and Supplementary Fig. 5A for its degree distribution) and the random setting (see Fig. 3B in the main text and Supplementary Fig. 5B). Though the maximal degree in the random setting is smaller than in the scale-free setting, but there are more nodes beyond the reference value in the random setting (red line in the figure, i.e., the smallest degree value among the largest nodes in each of sixteen communities). In the random setting (see Supplementary Fig. 5B), for a largest node in a ground-truth community that is of a degree near the reference value, there are more nodes with a larger degree out there in the network, and it will have a relatively higher probability to connect to one of them. If this happens, this largest degree node in the ground-truth community will be diminished as a follower, and the corresponding community cannot be detected. In contrast, in the scale-free setting in Supplementary Fig. 5A, there are fewer larger nodes and thus a smaller chance of direct connections between them. In addition, in the random setting in Supplementary Fig. 5B, a small decrease of the reference value will lead to a large increase on the number of nodes beyond it. While in the scale-free



Supplementary Figure 4. The community partitions by the LS algorithm on Ravasz-Barabási networks. (**A**) A 2-level Ravasz-Barabási network [4], where four small communities identified by the LS method. The color correspond to community partitions. (**B**) A modified Ravasz-Barabási with the top-right 2-level cluster with 3-level branching. For the bottom-left 2-level cluster, we remove four links; and for the bottom-right 2-level cluster, we remove eight links. (**C**) The decision graph for the network in **A** by the LS algorithm. (**D**) The decision graph for the network in **B** by the LS algorithm.

setting (see Supplementary Supplementary Fig. 5A), small fluctuations to the reference value would have a much milder influence. This explains that why the LS algorithm works better on the scale-free setting on detecting multiscale community structure than in the random setting (see Fig. 3 in the main text).

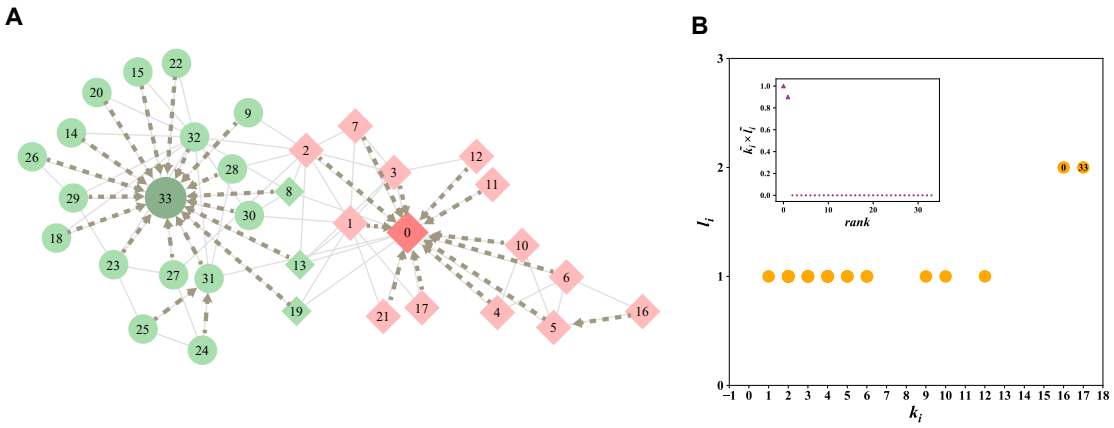


Supplementary Figure 5. The degree frequency distribution of multiscale networks. (**A**) The scale-free setting (i.e., each second level community is a scale-free network). The whole network is still of a heterogeneous degree distribution. (**B**) The random setting (i.e., each second level community is a ER random network). The degree distribution is relatively homogeneous. Networks in **A** and **B** are of the same average degree. The red line indicates the smallest value of all of the largest node in each of all sixteen ground-truth communities, and this value is what we called "reference value" in the main text. And the blue line indicates the smallest value of centers of all identified communities. In **A**, these two lines overlap; while in **B**, the blue line is larger than the reference value, which also indicates that some clusters are.

Supplementary Note 1.2: Identification of community centers

For better clarity and later use, we define **C** as the set of all local leaders (i.e., potential community centers), and **M** as a subset of local leaders that are of the maximal degree in the network (i.e., $\mathbf{M} \subseteq \mathbf{C}$). It is worth noting that not all nodes with the maximal degree in the network will be in the set of **M**. For example, in Fig. 1C in the main text, nodes m, f, and p are in the set **C**, and node m in the set **M** as it is of the maximal degree in the whole network and is a local leader. In contrast, though the node b is also of the maximal degree in the whole network, it is not in **M** or **C** as it already follows node m and is not identified as a local leader. In the Football network [5, 6, 7], there are several nodes

with the maximal degree are not in **M** (see those nodes with a degree equal 12 at the right-bottom of Fig. 7A).



Supplementary Figure 6. The community detection result by the LS algorithm on the Zachary Karate Club network. (**A**) The shape of the node represents the ground-truth community label, where the circle represents one group and the square represents the other. The color of the node represents the predicted labels by the LS algorithm. Zachary [8] identifies two social groups, one led by the instructor, node 0, and the other led by the club president, node 33 (the node index here is one less than that used in [8]). These nodes are correctly identified as community centers by the LS method. Nodes 8, 13, and 19 are connected to both centers and eventually classified to the community centered at the club president (node 33) due to a slight degree difference between two centers ($k_{33} = 17 > k_0 = 16$). Out-going links $i \to j$ indicates that node j dominates node i. (**B**) The scatter plot of degree k_i and path length l_i for all nodes. Community centers are identified as the ones with both a larger k_i and l_i. (Inset) The decision graph for quantitatively determining community centers (indicated by triangles) based on the product of rescaled degree $\tilde{k}_i$ and rescaled distance $\tilde{l}_i$. Community centers can be determined by a visual inspection for a big gap or by more sophisticated automatic detection methods.

For each node i, finding a nearest node j with a largest degree and $k_j \geq k_i$ can give rise to a field of hidden directionality $(i \to j)$ [1] on networks as shown in Fig. 1C in the main text and Supplementary Fig. 6A. We denote the length of the shortest-path from i to j as l_i, and there will be at most one out-going link for each node. The degree k_i and path length l_i can provide effective information for determining community centers, which are nodes with both a larger k_i and l_i (see Fig. 1E in the main text, Supplementary Fig. 6B, and Supplementary Figs. 7-8).

As most real networks are scale-free with a power-law distribution on degree, to reduce the impacts of extreme degree values caused by hub nodes, we calculate a ranked degree

$$k_i^* = \text{rank}(k_i), \tag{2}$$

where the rank() function returns the index of a value in the ascendingly sorted set. The node(s) with the smallest degree will (all) have $k_i^* = 1$ regardless of its specific degree value k_i, and the node(s) with the largest degree will have $k_i^* = len(set(k_i))$, where $set(k_i)$ returns all unique degree values of all nodes in the network, and $len()$ is a function calculating the length of the set. For example, for a network with degree values as $(1, 1, 1, 3, 3, 3, 6)$, the corresponding k_i^* will be $[1, 1, 1, 2, 2, 2, 3]$. So, in this case, when $k_i = 6$, $k_i^* = 3$. Meanwhile, in order to ensure that the shortest-path length differences are more distinguishable, we calculate a rescaled path length

$$l_i^* = l_i^2. \tag{3}$$

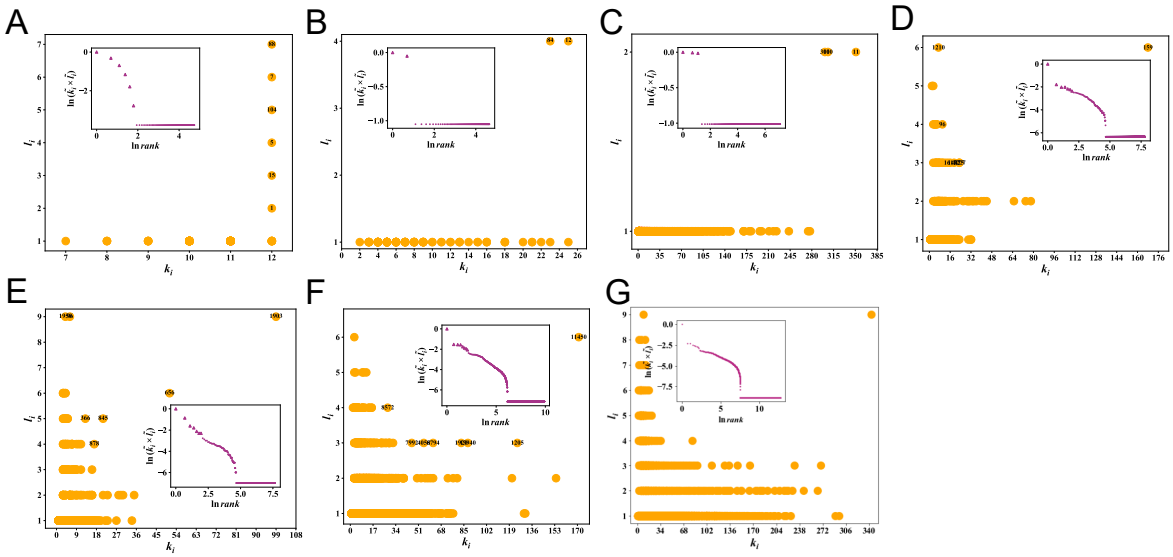
[1] When referring to specific out-going links that signifies the local dominance, we also call it hidden directionality. While directionality is an explicit characteristic of edges in a directed network, asymmetric relationship between nodes can be seen as an intrinsic higher-order directionality in undirected networks. Asymmetric relationships are profound in networks, for example, two connected nodes with unequal degrees (e.g., $k_i > k_j$) might have different influence over each other [9, 10, 11], node i may receive only $1/k_i$ influence from j which is smaller than the $1/k_j$ influence received by j from i.

Note that most networks generated from 2D benchmark vector data (see Supplementary Fig. 12) are of a relatively narrow degree distribution, in such cases, k_i^* is quite comparable with k_i, and l_i is also relatively more distinguishable (see Supplementary Fig. 12). Thus, in those cases, k_i and l_i can already have a quite comparable performance as of k_i^* and l_i^*. One practical reason is that the density is quite comparable across different clusters in most benchmark 2D vector data sets [12, 13, 14, 15]. When the density is more heterogeneous across clusters, the degree distribution might be closer to a power-law and then the operation of calculating k_i^* will be necessary.

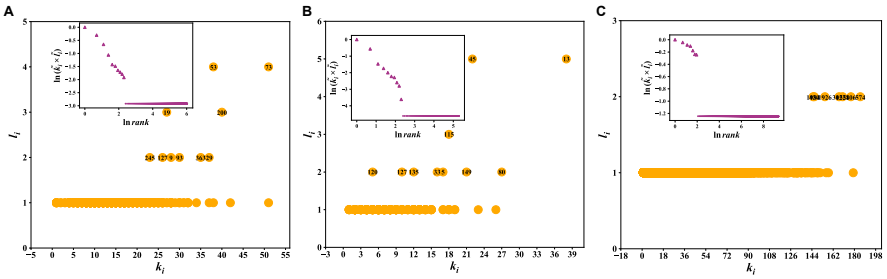
To make k_i^* and l_i^* more comparable on magnitudes, we do a min-max normalization for them.

$$\tilde{k}_i = \frac{k_i^* - \min(k_i^*)}{\max(k_i^*) - \min(k_i^*)}, \quad \tilde{l}_i = \frac{l_i^* - \min(l_i^*)}{\max(l_i^*) - \min(l_i^*)}. \tag{4}$$

The number of community centers can be easily identified according to the rank distribution of $\tilde{k}_i \times \tilde{l}_i$, which is a composite indicator on the centerness of a node. Generally, there will be a gap between community centers and other nodes (see Supplementary Fig. 6B and insets of Supplementary Fig. 7), which can be better visually inspected in a log-log plot (see insets of Supplementary Fig. 7D-F).



Supplementary Figure 7. Identification of community centers of networks. Scatter plots of k_i and l_i for (**A**) Football [5, 6, 7], (**B**) Polbooks [16], (**C**) Polblogs [17], (**D**) Cora [18], (**E**) Citeseers [19], (**F**) Pubmed [20], and (**G**) DBLP [21]. (Insets) Decision graph for quantitatively determining community centers based on the product of $\tilde{k}_i$ and $\tilde{l}_i$. Identified community centers are highlighted by larger triangles.



Supplementary Figure 8. Identification of community centers of human mobility flow networks in three diversified cities. (**A**) Dakar in Senegal, Africa. (**B**) Abidjan in Côte d'Ivoire, Africa. (**C**) Beijing in China, Asia. Identified cluster centers are highlighted by larger triangles.

Supplementary Note 1.3: Pseudo Code of our LS algorithm

Algorithm 1: The Local Search (LS) algorithm for community detection

Input: $G = (V, E)$: undirected and unweighted graph, where V is the set of vertices, and E is the set of edges.

Output: Community partitions.

for *node* i **in** V **do**
  j = node with the maximum degree in neighborhood of i;
  if $k_j \geq k_i$ **then**
    $i \rightarrow j$;
    $l_i = 1$;
  end
end
Build DAGs by *directionality*$(i \rightarrow j)$, and obtain **C** and **M**;
  #**C** is the set of nodes with no outgoing edges in DAGs;
  #**M** is the set of node(s) with the maximum degree in **C**;
for *node* i **in C\M do**
  $m = 2$;
  Perform a *local-BFS*$(G,\ i)$ to find m-order neighborhood of i;
  j = node with the maximum degree in m-order neighborhood;
  if $k_j \geq k_i$ **then**
    $i \rightarrow j$;
    $l_i = d_{ij}$;
    continue;
  else
    m += 1;
    go to line **13**;
  end
end
for *node* i **in** M **do**
  $l_i = max(l_k),\ \forall k \in V$;
end
for *node* i **in** C **do**
  $\delta_i = \tilde{l}_i \times \tilde{k}_i$; # $\tilde{l}_i$ and $\tilde{k}_i$ are calculated according to Eq.(3).
end
Determine the *community centers* according to descendingly sorted δ_i;
Assign community labels from *community centers* to their neighbors along the reverse direction $i \leftarrow j$ of hidden directionality $(i \rightarrow j)$;
return partitioned sets of nodes according to community labels

Supplementary Note 1.4: Time Complexity

In the LS algorithm, for each local leader, we perform a local-BFS that will stop further searching when encountering a nearest local leader that is of a larger or equal degree. Based on empirical analysis on real networks, the average number of searching layers $\langle l \rangle_{\mathbf{C}\backslash\mathbf{M}}$ is only slightly larger than 2 in most cases (see Table 1). For other follower nodes, they just need to traverse all their first-order neighbors. In addition, such a local-BFS, which is theoretically approximated by $\langle k \rangle^{\langle l \rangle_{\mathbf{C}\backslash\mathbf{M}}}$, is bounded by the number of edges E in the network, thus the maximal value of $(|\mathbf{C}| - |\mathbf{M}|)\langle k \rangle^{\langle l \rangle_{\mathbf{C}\backslash\mathbf{M}}}$ is $(|\mathbf{C}| - |\mathbf{M}|)E$, which is usually acceptable, and is usually smaller than E (see the last column of Table 1). It is worth noting that not all nodes with the maximal degree in the network will be in the set of **M**, only those identified as local leaders and are of the maximal degree are. For example, if there are several nodes with the maximal degree and they are connected to each other (see Supplementary Fig. 1A), then there will be only one node identified as a local leader (see Supplementary Fig. 1B).

For clustering vector data, before performing our LS algorithm, the network construction process takes $O(N \log_b N)$ if accelerated by R-tree [22, 23], where b is the branching factor of the R-tree. If this process is not accelerated by R-tree, the time complexity of this part would be $O(N^2)$, which

	N	E	$\lvert\mathbf{C}\rvert$	$\langle k\rangle$	$\langle l\rangle$	$\langle l\rangle_{\mathbf{C}\setminus\mathbf{M}}$	$(\lvert\mathbf{C}\rvert-\lvert\mathbf{M}\rvert)\langle k\rangle^{\langle l\rangle_{\mathbf{C}\setminus\mathbf{M}}}$
Karate [8]	34	78	2	4.59	1.058	2.00	21.05
Football [5, 6, 7]	115	613	6	10.66	1.069	–	–
Polbooks [16]	105	441	2	8.40	1.057	4.00	4978.71 (441)
Polblogs [17]	1,222	16,717	3	27.36	1.002	2.00	1,497.15
Cora [18]	2,485	5,069	104	4.08	1.047	2.16	2,132.77
Citeseers [19]	2,110	3,668	106	3.48	1.080	2.76	3279.96
PubMed [20]	19,717	44,327	472	4.50	1.025	2.06	10,445.59

Supplementary Table 1. Basic statistics on real networks with ground-truth community labels and intermediate results of the LS algorithm. N is the number of nodes in the network, E is the number of edges, $|\mathbf{C}|$ is the size of the set of potential centers (i.e., local leaders) identified by our LS algorithm, $\langle k\rangle$ is the average degree of the network, $\langle l\rangle = \sum_i l_i$ is the average number of layers searched when a node finds a nearest node with a larger or equal degree than itself, $\langle l\rangle_{\mathbf{C}\setminus\mathbf{M}}$ is the average number of layers searched for all potential centers except the local leader(s) in $\mathbf{M}$ with the maximal degree in the whole network (i.e., $i \in \mathbf{C}\setminus\mathbf{M}$), thus $(|\mathbf{C}|-|\mathbf{M}|)\langle k\rangle^{\langle l\rangle_{\mathbf{C}\setminus\mathbf{M}}}$ is the theoretical average number of links need to be traversed by the LS algorithm, which is bounded by $(|\mathbf{C}|-|\mathbf{M}|)E$ and is usually smaller than E. In most cases, there will be only one or very a few nodes in $\mathbf{M}$. For example, in Polbooks, there is one node in $\mathbf{M}$, thus $\langle l\rangle_{\mathbf{c}\setminus\mathbf{M}}$ reflects the length of shortest-path between the secondary potential center and the first potential center. The shortest path length is 4 and the calculation based on $(|\mathbf{C}|-|\mathbf{M}|)\langle k\rangle^{\langle l\rangle_{\mathbf{C}\setminus\mathbf{M}}}$ is 4978.71, however, it is at most 441 in practice to traverse all edges (which is indicated in parentheses). In the Football network, we identify 6 potential centers, and they are all of the same maximal degree in the network, thus there is no need to perform local-BFS for any of them, and they are identified as centers. Thus we did not report the values in the last two columns of the Football network.

calculates a full distance matrix. After the network is constructed, the remaining process is the same, detected communities correspond to cluster partitions.

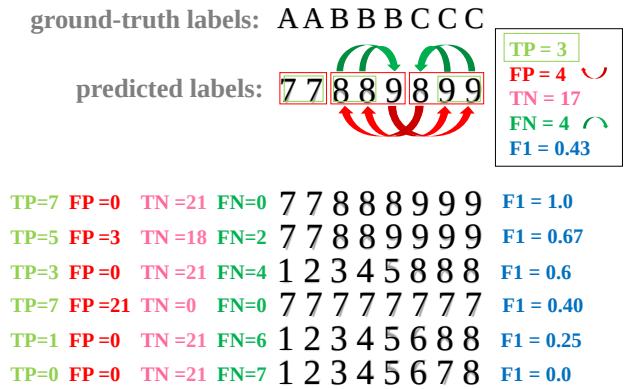
Supplementary Note 2: Evaluation indicators on the performance

Supplementary Note 2.1: F1 score for both community detection and cluster analysis

The performance of both community detection and cluster analysis can be evaluated by F_1-score that is defined as follows:

$$\begin{aligned} \text{Precision} &= \frac{\text{TP}}{\text{TP}+\text{FP}}, \\ \text{Recall} &= \frac{\text{TP}}{\text{TP}+\text{FN}}, \\ F_1 &= \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}. \end{aligned} \tag{5}$$

TP is the number of combinations of nodes that are of the same predicted labels and the same ground-truth labels (see the combinations in green rectangles in Supplementary Fig. 9). FP is the number of combinations of nodes that are of the same predicted labels but different ground-truth labels (see red curved arrows in Supplementary Fig. 9). FN is the number of combinations of nodes that are of different predicted labels but the same ground-truth labels (see green curved arrows in Supplementary Fig. 9). TN is the number of combinations of nodes that are of the different predicted labels and different ground-truth labels (see the calculation process in Supplementary Fig. 9). Precision is the ratio of TP to the sum of TP and FP, which measures the fraction of correct (or positive) results among the predictions of the algorithm. The higher the precision, a larger fraction of combinations of nodes predicted in the same community are really in the same community according to the ground-truth labels. Recall is the ratio of TP to the sum of TP and FN, which measures the coverage of the positive



Supplementary Figure 9. The definition of TP, FP, TN and FN in evaluating grouping results. The objects within red boxes indicate that their true labels belong to the same communities. The objects in green boxes have the same ground-truth labels and predicted labels. The number of combinations of objects within each green box is the value of TP. The number of green arrows is FN, which corresponds to combinations of objects that are of the different predicted labels but the same ground-truth label. Likewise, the objects connected by red arrows have different ground-truth labels but the same predicted label, the number of which is FP (indicated by red arrows). TN is the number of objects that their true labels and predicted labels both belong to different communities. In the case of $(7,7,8,8,9,8,9,9)$, $TN = 2\times3+2\times3+2\times2+1\times1$. Note that $TP+FP+TN+FN = C(n,2)$, where n is the number of objects and $C(n,2)$ is the combination of choosing 2 out of n objects. Thus when three of them are calculated, the other one can be calculated in this way, e.g., in the above example, $TN = C(8,2)-3-4-4 = 17$. The formula of F_1 is shown in Supplementary Equation (6). F_1 is in the range of $[0,1]$, when the difference between the predicted labels and the ground-truth labels is smaller, the F_1 score will be larger.

predictions against the ground-truth positive set. The higher the Recall, a larger ratio of positive (or correct) combinations of nodes in the predicted community according to the ground-truth labels to the combinations of nodes in the same ground-truth community. However, Precision is strongly affected by the size of the predicted positive set, which means that precision can easily reach a high score if you only predict a tiny set of positive items that are highly probably positive, but then the coverage of positive items can be quite low. While, Recall can be one if you predict all items are positive, as it only measures the coverage of positive items. To be more comprehensive on both sides, F_1 takes the weighted harmonic average of Precision and Recall as its value, where the weights of them are usually set to be 1. The weight can be tuned by the different focus on either aspect. F_1 is in the range of $[0,1]$, a higher F_1 score indicates a better partition. In comparison, the normalized mutual information (NMI) is biased to smaller groups [24], i.e., when small communities are detected correctly, the NMI will increase more. NMI ignores the information content of the contingency table, and, in addition, its symmetric normalization introduces spurious dependence [24]. And modularity only seeks for maximizing the difference between the partition and a global random null model [3], it can be blind to other generating process of networks in reality [25]. In networks constructed from vector data, some clusters may not corresponds to a high modularity value (e.g., networks constructed from some manifolds as shown in Fig. 4C,G,H in the main text). In flow networks, evidence indicates that the optimizing modularity might not lead to an optimal partition in some cases due to neglecting the structural regularity of interdependence characterized by flows [26]. Sometimes, a partition with the largest modularity is highly probably not the actual division of communities in real networks [3].

It is worth noting that although the evaluation of the classification performance of an algorithm with a ground truth is standard practice, the choice of the ground truth(s) are crucial. The ground truth of community partitions usually require detailed survey, which can be quite difficult for very large networks [2], and usually cannot be regarded as the same as metadata [2, 25]. For example, in the well known Zachary Karate Club network [8], the metadata of nodes can also be their gender, age, major, ethnicity, however, most of which are irrelevant to the community structure when interested in understanding the split of the club[2, 25]. In the DBLP collaboration network [21], where two

authors are connected if they publish at least one paper together, the meta data of papers is the publication venue (i.e., journal or conference) which forms the basis of identifying communities in ref. [21] (i.e., authors who published to a certain journal or conference form a community), but the meta data is different from ground truth, if there is any. In some cases, meta data is related to community structure, but they two are not equivalent, otherwise, there is no need to do community detection based on topology but just look at meta data [2, 25]. For the Football network [5], we also used the ground truth labeling[2] given by Evans [6, 7]. We calculate the F_1-score for all methods based on both sets (see Table 1), and we find that results obtained by most methods better align with the ones in refs. [6, 7], which is indicated by higher F_1-scores. As for the Polblogs network, there might be indeed three groups in it (i.e., apart from liberal and conservative, there is a neutral community) [17], which partially explains why the LS does not work that well on this example when compared with other examples, as LS detects three communities (see Table 1 in the main text). In addition, we also compare the LS algorithm with the GDG (geodesic density gradient) algorithm [27] in Supplementary Table 2 via F-1 score.

	N	E	N_c	GDG			LS			Δt (ms)
				F1	N_c	t(ms)	F1	N_c	t(ms)	
Karate	34	78	2	**0.91**	3	40	0.83	**2**	**6**	34
Football	115	613	10	**0.60**	12	276	0.35	**6**	**20**	256
Polbooks	105	441	3	0.75	**3**	118	**0.80**	2	**8**	110
Polblogs	1,490	19,090	2	0.66	3	72,939	**0.69**	3	**212**	72,727
Cora	2,708	5,429	7	0.29	5	800,015	**0.33**	**7**	**139**	799,876
Citeseers	3,264	9,072	6	**0.63**	5	907,446	0.45	7	**131**	907,315
PubMed	19,717	44,327	3	0.19	529	87,238,940	**0.46**	**8**	**2,298**	87,236,642

Supplementary Table 2. Comparisons between the GDG algorithm [27] and LS on networks with ground truth community labels. As GDG requires calculating the shortest path between all node pairs to embed the network in a vector space, whose dimension equals the number of nodes, for large scale networks, a PCA is applied in GDG to reduce the dimension [27], and we select the first one thousand components as the embedding. GDG achieves the best performance in two out of seven networks when compared with a broad range of other popular algorithms (see Table 2 in the main text). LS is much more efficient in implementation, and has a better performance in four out of seven networks when compared with GDG.

It is worth pointing out the analogies and differences between GDG and LS algorithms: Most community detection algorithms directly work on network data, while the GDG algorithm eventually works in a vector space, which is continuous and comply with metric properties (e.g., mutual distance, triangle inequality), and implicitly relies on the density of data points [15] in the embedded space; LS algorithm works on network data, which is discretised and may easily violates metric properties (i.e., asymmetric interaction between nodes, and $l_{ik} + l_{kj}$ might be smaller than l_{ij}), with a focus on the hidden leader-follower relation revealed by local dominance. When applying the LS algorithm to clustering vector data, we establish a connection between the concept of density of data points in the vector space and degree of nodes in the network. This connection enhances our understanding of both clustering analysis and community detection, which have been long regarded as similar tasks, and better captures the underlying structure within the data.

Supplementary Note 2.2: Conductance for community detection

Conductance is commonly used in algorithms for graph partitioning, community detection, and spectral clustering, among other applications in network analysis. In the context of measuring the performance of community detection algorithms, apart from maximizing modularity, it also can be evaluated by

[2]The original Football network data [5] (`http://www-personal.umich.edu/~mejn/netdata`) provided the games between American college football teams from one season but the conference assignments (community labels) were from a different season. The data in [6, 7] which is used here has the same games as the original dataset but the conference assignments are for the same season as the games. The community labels change for about 10% of the teams.

	N	E	Nc	Inferential		LS	
				F1	Nc	F1	Nc
Karate	34	78	2	0.653	1	**0.83**	**2**
Football	115	613	10	**0.865**	**11**	0.35	6
Polbooks	105	441	3	0.771	**3**	**0.80**	2
Polblogs	1,490	19,090	2	0.324	22	**0.69**	**3**
Cora	2,708	5,429	7	0.309	20	**0.33**	**7**
Citeseers	3,264	9,072	6	0.399	10	**0.45**	**7**
PubMed	19,717	44,327	3	0.074	53	**0.46**	**8**

Supplementary Table 3. Comparisons between an inferential algorithm [28, 29] and LS on networks with ground truth community labels.

the extend of minimizing conductance [30, 21] that is defined as follows:

$$\text{Conductance} = \sum_i \frac{\text{Cut}(\text{S}_\text{i}, \hat{\text{S}}_\text{i})}{\min\left(\text{vol}(\text{S}_\text{i}), \text{vol}(\hat{\text{S}}_\text{i})\right)} / |Communities|, \tag{6}$$

where S_i is the set of nodes in the community i and $\hat{S}_i$ is the set of all other nodes not in the community i, $Cut(S_i, \hat{S}_i)$ is the set of edges where one endpoint of edge is in S_i and the other end is in $\hat{S}_i$, $vol(S_i)$ is the sum of degree of all nodes in S_i, $|Communities|$ is the number of detected communities. The conductance ranges from 0 to 1, and a smaller conductance indicates a better community partition. Different from the modularity that focuses on connection patterns within communities, conductance considers both connections between communities and within communities via their averaged ratio.

Here, we calculate the value of conductance for Louvain, GDG, LS algorithm based on datasets mentioned above (see Supplementary Table 4). And we find that our LS algorithm is of the smallest conductance in four out of seven real-world networks. And it is interesting to observe that our LS algorithm is only of a slightly larger conductance value than the Louvain algorithm, though LS ranks the last when measured by F-1 score based on the ground-truth labels (see Table 2 in the main text). In some sense, the LS algorithm still obtain some good partition, which is just different from the real labeling.

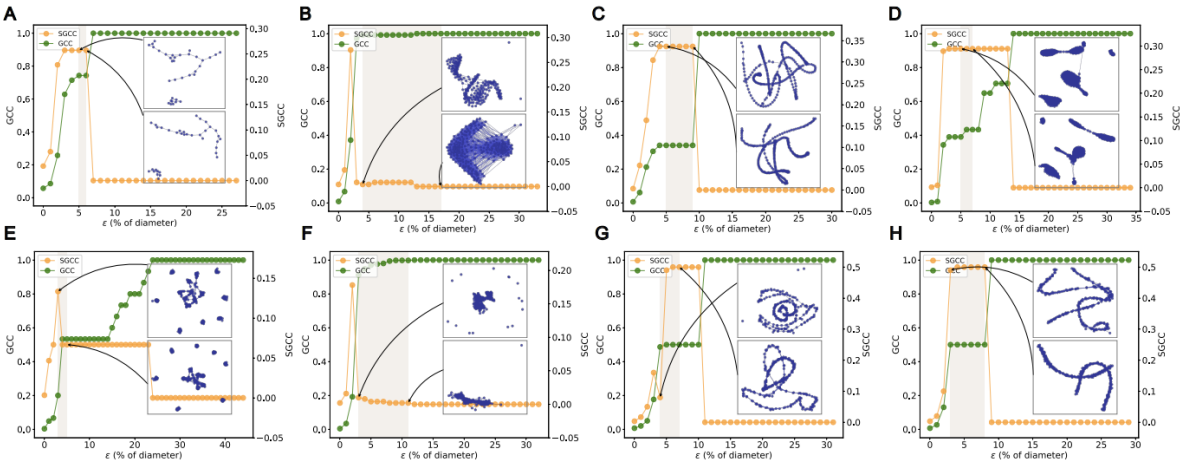
	Louvain	GDG	LS
Karate	0.29	0.43	**0.21**
Football	**0.28**	0.60	0.29
Polbooks	0.24	0.21	**0.06**
Polblogs	**0.33**	0.48	0.41
Cora	0.14	0.42	**0.11**
Citeseers	0.08	0.24	**0.03**
PubMed	**0.14**	0.95	0.22

Supplementary Table 4. Comparisons between the partition performance of GDG (geodesic density gradient) [27], Louvain, and LS algorithms on networks when evaluated by conductance. The LS algorithm is of the smallest conductance in four out of seven networks.

Supplementary Note 3: Network representation of vector data

There are a variety of ways to build a network from vector data, including ε-ball, kNN (k-nearest-neighbors) and its variants, mutual kNN, continuous kNN, relaxed maximum spanning tree [31], percolation or threshold related methods [32, 33], and more sophisticated ones [34]. The main challenge is determining a value for the parameters of the chosen methods, e.g., a distance threshold ε for ε-ball method or k for kNN types methods. When using kNN, each object is connected to its k closest neighbours. However, it is difficult to determine the most appropriate value of k. When k is too small, the whole network may be too disconnected, and if k is too large, the network will be too densely connected, both of which are likely not to reflect meaningful local structures.

Supplementary Note 3.1: Finding ε in the ε-ball method



Supplementary Figure 10. The phase diagram for constructing networks from 2D benchmark vector data. (**A**) The test case in Ref.[35], for which the DDB algorithm [36] fails. (**B**) Flame [12]. (**C**) Spiral [13]. (**D**) Aggregation [14].(**E**) R15 [15]. (**F**) Blobs with 500 points. (**G**) Circles with 300 points. (**H**) Moons with 300 points. (**F**)-(**H**) are generated by the standard scikit-learn 0.19.2 in Python (`https://pypi.org/project/scikit-learn/0.19.2/`). The original spatial layout of vector data is shown in Supplementary Fig. 11. The largest giant connected component (GCC) represents the ratio of the size of the maximal connected sub-graph to the size of the network, and SGCC represents the second-largest maximal connected graph. The shading area indicates the optimal range of ε, during which the LS method can get the correct partition that align with the common consensus. (Insets) Corresponding constructed networks.

In this paper, we use ε-ball method to construct networks from vector data. Its main advantage in our context is its ability to separate local and global information with an appropriate value of ε. Classical clustering methods (e.g., k-means, which are effective on spherical data) are typically unable to group vector data with arbitrary shape (see Supplementary Fig. 11B-D,G,H), one of the reason is their inability to capture continuity in local information. For example, in Supplementary Fig. 11H, based on the global metric, every point is accessible to others, and the Euclidean distance between the two end nodes in the red group are farther than to some points in the blue group; when based on the local metric, only points within ε-ball of the point are accessible, and the two end nodes in the red group are reachable to each other but is not reachable to any point in the blue cluster when ε is not very large.

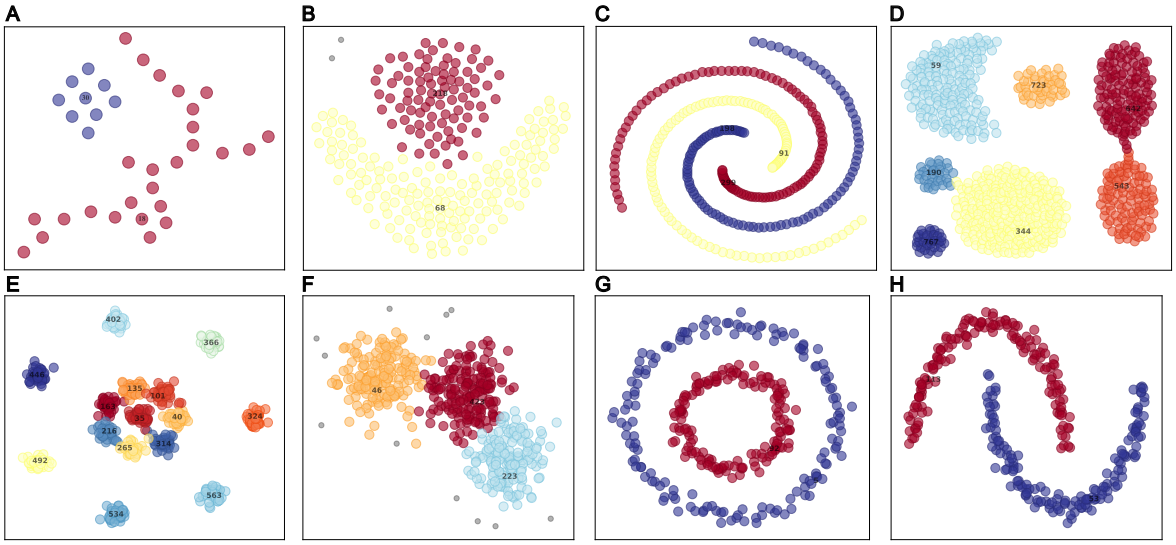
Building networks via the ε-ball method, which can be accelerated by R-tree data structure [22, 23], with a proper distance threshold ε can naturally help on separate local metrics from global ones.

However, determining ε used to be based on empirical experience and usually subject to arbitrariness [35], with similar problems existing in many clustering methods (e.g., DBSCAN [37] or DDB [36]). When building networks using the ε-ball method, the network will undergo a transition from isolated nodes or connected components to a fully connected complete graph as the value of ε increases. A critical value for ε can be derived from percolation phase transition as the value for which the size of the second largest connected component is maximal [38, 39, 40].

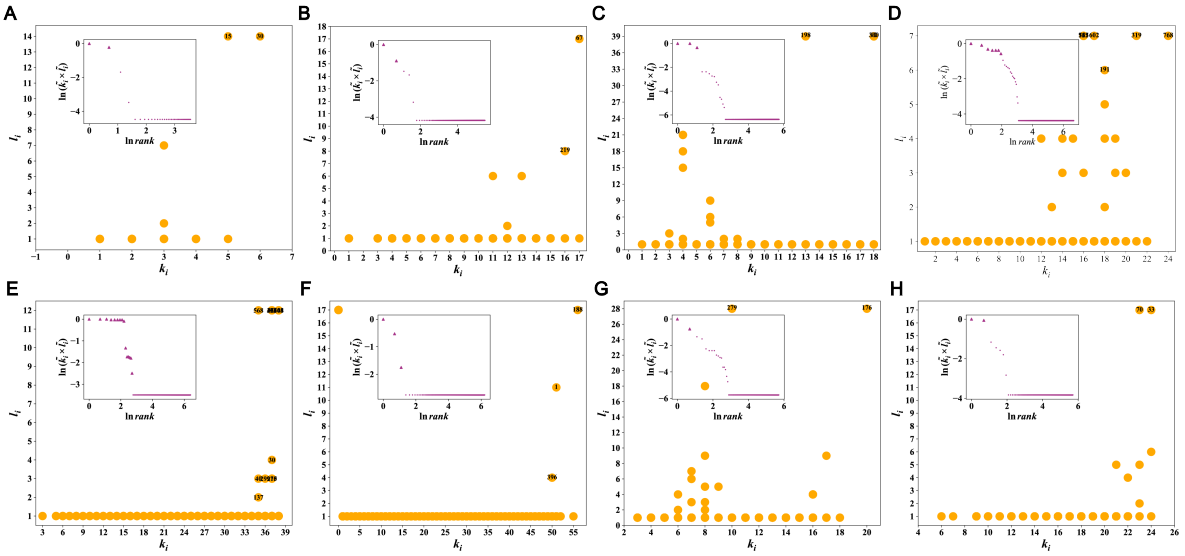
Experimental results indicate that setting ε near the critical value when building networks from vector data is usually a good choice to balance local and global structures (see Supplementary Fig. 10 and corresponding clustering results by the LS algorithm in Supplementary Fig. 11). In addition, the range of ε is not narrow in most cases (see Supplementary Fig. 10), giving a notion of stability to the resulting networks. With a similar idea, some works chose the value of ε such the network is fully connected [32], but it may not be always the optimal setting. After the critical point of the percolation phase transition, isolated nodes or small-sized groups of nodes can be considered as noisy data that do not belong to any cluster (see grey nodes in Supplementary Fig. 11B,F). We find that with the presence of non-spherical data (see examples in Supplementary Fig. 10A,C,G,H), the LS algorithm can have a better performance if ε is chosen before the critical point; in other cases, a value higher than the

critical point (see Supplementary Fig. 10B,E,F). However, this is just a rough summary from cases in this work, and the reason behind is not the focus of this paper and worth future closer investigations.

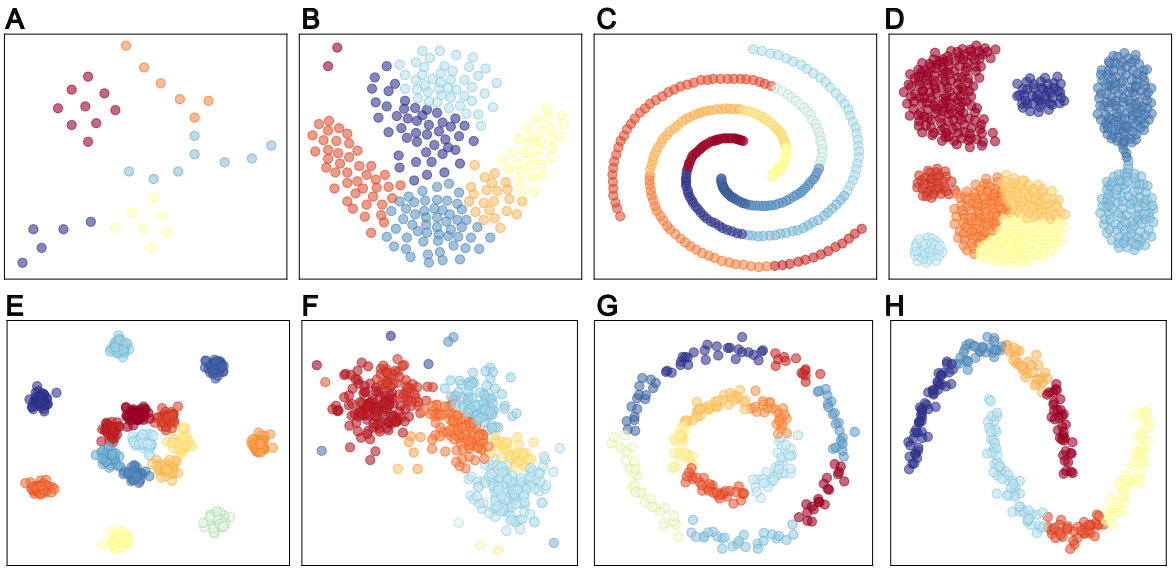
Applying the LS algorithm to the network built around the critical threshold value, we can obtain the partition results as shown in Fig. 6 in the main text and Supplementary Fig. 11 according to the decision graph (see Supplementary Fig. 12). In contrast, on the same constructed networks from the vector data, the Louvain algorithm cannot obtain partitions that align with the common consensus (see Supplementary Fig. 13), which tend to have more smaller clusters in a more segmented way.



Supplementary Figure 11. The original spatial layout of 2D benchmark vector data and clustering results obtained by the LS algorithm on networks constructed from them. (**A**)-(**H**) corresponds to clustering results obtained from networks in the optimal range in Supplementary Fig. 10. Node color indicates clustering partitions. Grey nodes corresponds to identified noisy data. The index of identified cluster centers are labeled in the figure.



Supplementary Figure 12. Identification of cluster centers for 2D benchmark vector data by the LS algorithm. (**A**)-(**H**) corresponds to networks in Supplementary Fig. 10, whose original spatial layout is shown in Supplementary Fig. 11. (Insets) Decision graphs. Identified cluster centers are highlighted by larger triangles.



Supplementary Figure 13. Clustering results obtained by the Louvain algorithm on networks constructed from vector data. The Louvain algorithm works on the same network constructed from vector data as of the LS method. Node color indicates clustering partitions.

Supplementary Note 3.1.1: The advantage of building networks for clustering high dimensional vector data

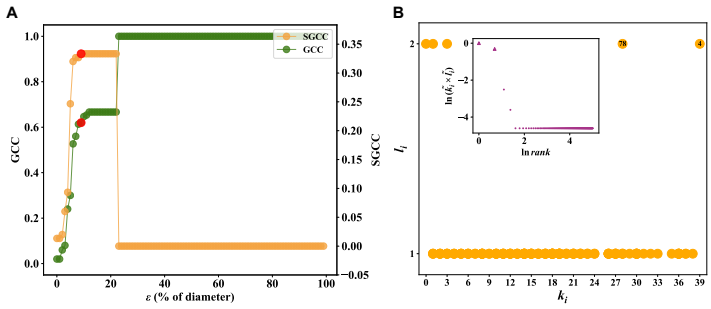
Apart from 2-dimensional benchmark vector data, datasets with very high dimensions are very common in practice. We further apply our LS algorithm to several high dimensional datasets, including Iris [41] (see Supplementary Fig. 14), Wine [42] (see Supplementary Fig. 15), MNIST [43] (see Supplementary Fig. 16), Olivetti [44] (see Supplementary Fig. 17 and Fig. 18).

- The Iris flower data set is 4-dimensional and contains 50 samples of 3 species (Iris setosa, Iris virginica, and Iris versicolor). Each sample has four features: the length and width of both sepals and petals.
- The Wine data set is of 13-dimension that contains 178 samples of 3 cultivars. The 13 features are the quantities of 13 constituents by the chemical analysis of the 3 cultivars, including alcohol, malic acid, ash, alcalinity of ash, magnesium, total phenols, flavanoids, nonflavanoid phenols, proanthocyanins, color intensity, hue, OD280/OD315 of diluted wines and proline.
- The MNIST handwritten digit dataset is obtained from the National Institute of Standards and Technology, which is widely used in machine learning. It is of 784 (28×28)-dimension and contains 60,000 samples of 10 classes, each of which corresponds to 0-9, respectively. We randomly sample 1,000 figures of 10 digits, and each digit has 100 samples.
- The Olivetti database of human face is widely used in machine learning. It is of 10,304 (92×112)-dimension and contains 400 samples of 10 classes, each of which is consisted of the 40 photos of a person, respectively. The positions of the faces in the photos are the same.

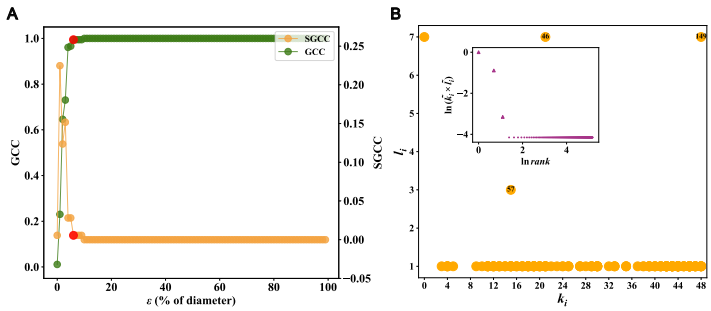
For these high dimensional vector data, the phase transition diagrams and corresponding decision graphs are shown in Figs. 14-18. The networks we choose to perform the LS algorithm are highlighted by the red dot in Figs. 14-18, and they are near the critical point of the percolation phase transition. Though degree can be regarded as an analogy to density, and path length as an analogy to Euclidean distance from an object to a nearest larger object, the LS algorithm can have a better performance than the DDB algorithm [36] on datasets with relatively high dimensions. Possible reasons behind the good performance of the LS algorithm might reside in the network construction, which can be regarded as a coarse-graining process and noise filtering, and this also helps on better identifying asymmetric relation between objects.

For the Olivetti dataset, which is a typical example of a challenging high dimensional dataset with small sample size (each person only has ten images, each of which is of 10,304 pixels), we still calculate the Euclidean distance between objects (instead of a more complicated "complex wavelet structural similarity (CW-SSIM)" image similarity measure [45, 36]), and we do not perform a stricter association criterion as in ref.[36]. The LS algorithm clearly identifies 9 and 33 clusters for the Olivetti dataset with 100 images and all 400 images, respectively. In contrast, the DDB algorithm does not identify the correct number of clusters (note that in Table 2 in the main text, we report the result when we manually set the number of clusters as 10 and 40 for DDB, respectively). Besides, we can recognise more images of each individual than the DDB method (see Fig. 4D in ref. [36] and Supplementary Fig. 19B for comparisons), as reflected by a higher F_1 score. And each cluster only comprise images of the same object.

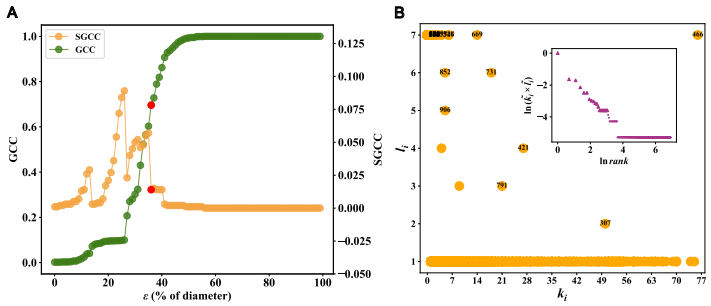
In more complex situations with more objects and clusters, the phase diagram also can be more complex with stronger fluctuations. Compared to the Olivetti dataset with the first 100 images, whose phase transition diagram is quite clear (see Supplementary Fig. 17), when we analyze the Olivetti dataset with all 400 images for 40 persons, its phase diagram become more complex with stronger fluctuations (see Supplementary Fig. 18), which is also the cases for MNIST dataset (see Supplementary Fig. 16).



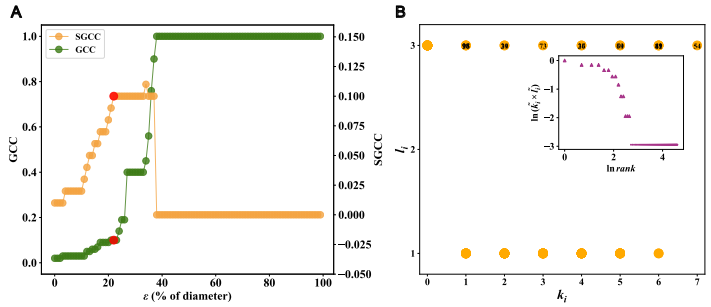
Supplementary Figure 14. The phase transition diagram and decision graph for Iris data set. (**A**) The distribution of GCC and SGCC on Iris data set. (**B**) The decision graph for Iris data set. ε=9% diameter of the original vector data, where the diameter equals the longest distance between any two objects. Three cluster centers are highlighted by larger triangles.



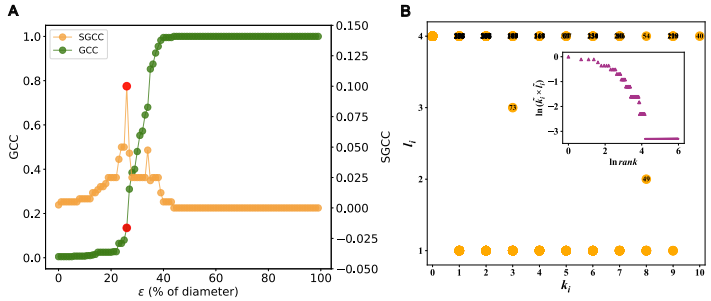
Supplementary Figure 15. The phase transition diagram and decision graph for Wine data set. (**A**) The distribution of GCC and SGCC on Wine data set. (**B**) The decision graph for Wine data set. ε=6% diameter of the original vector data, where the diameter equals the longest distance between any two objects. Two cluster centers are highlighted by larger triangles.



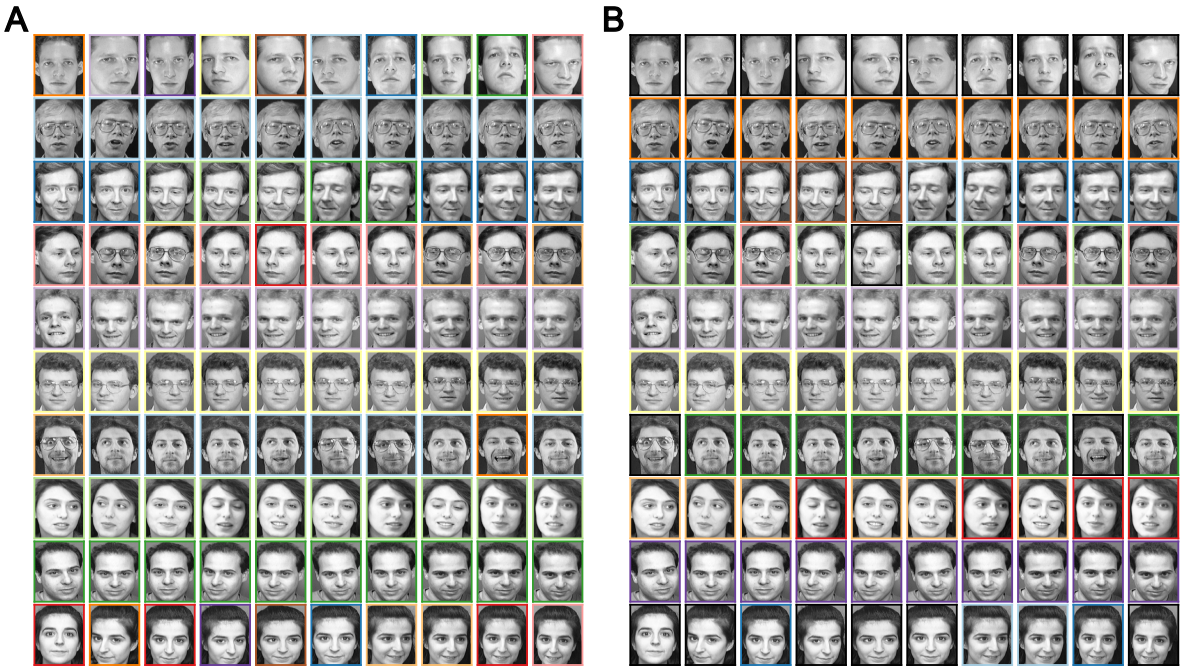
Supplementary Figure 16. The phase transition diagram and decision graph for MNIST data set.(**A**) The phase transition diagram and (**B**) decision graph for 1000 sampling data from MNIST handwritten digits database. There are 10 classes in total and every class has 100 grayscale image with a size of 28×28 pixels. ε=36% diameter of the original vector data, where the diameter equals the longest distance between any two objects. Nine cluster centers are highlighted by larger triangles.



Supplementary Figure 17. The phase transition diagram and decision graph for Olivetti sample database. (**A**) The phase transition diagram and (**B**) decision graph for the first 100 images of Olivetti database, the number of clusters is 10, with 10 image for each cluster in 92×112 dimensions. ε=22% diameter of the original vector data, where the diameter equals the longest distance between any two objects. Nine cluster centers are highlighted by larger triangles.



Supplementary Figure 18. The phase transition diagram and decision graph for Olivetti database. (**A**) The phase transition diagram and (**B**) decision graph for the whole Olivetti database, the number of clusters is 40, with 10 images for each clusters in 92×112 dimensions. ε=26% diameter of the original vector data, where the diameter equals the longest distance between any two objects. Thirty-three cluster centers are highlighted by larger triangles.

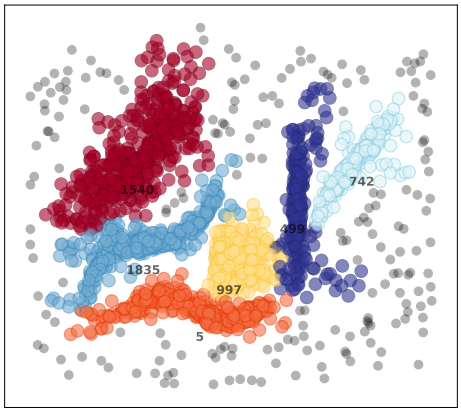


Supplementary Figure 19. The clustering result by Louvain and our LS algorithm on the Olivetti data set [19] with its first 100 images.(**A**) Louvain and (**B**) LS algorithm, Each identified category is indicated by the color of rectangle border of the figure, and the black color corresponds to noisy objects identified by our method. See the phase transition diagram and decision graph in Supplementary Fig. 17. This example further demonstrates the strength of our framework on high-dimensional vector data (see comparisons on F_1 scores between our LS method and the DDB algorithm [36] in Table 2 in the main text. The dataset and permission of usage is obtained from AT&T Laboratories Cambridge (`https://cam-orl.co.uk/facedatabase.html`).

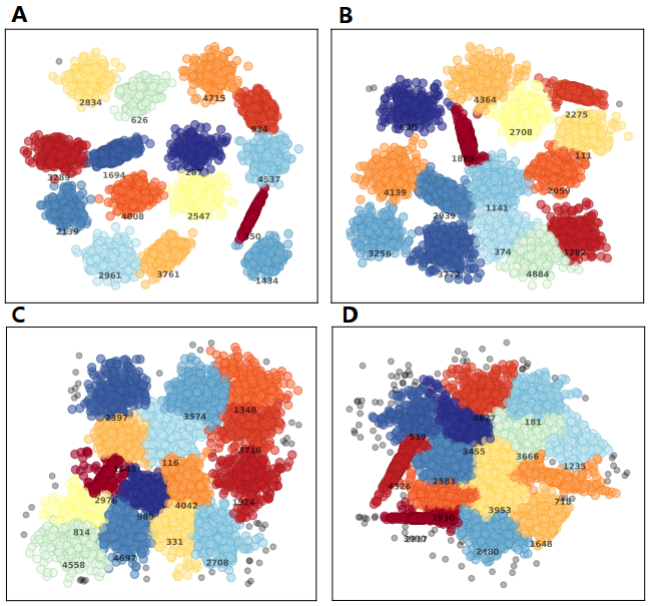
Supplementary Note 4: Robustness tests of LS algorithm

Supplementary Note 4.1: Robustness tests on vector data

We first test our LS algorithm on the Clusterable-data dataset (https://www.kaggle.com/lvk110395/clusterable-datanpy) that contains many noisy points (see Supplementary Fig. 20). The result obtained by the LS algorithm (see Supplementary Fig. 20) is quite comparable to results by the HDBSCAN method [46]. For the S-sets dataset that comprises 15 Gaussian clusters, with the increase of degree of cluster overlapping (see Supplementary Fig. 21A-D), the clustering results obtained by our LS algorithm remain relatively robust with the F_1-score equals 0.99, 0.94, 0.74, 0.62, respectively.



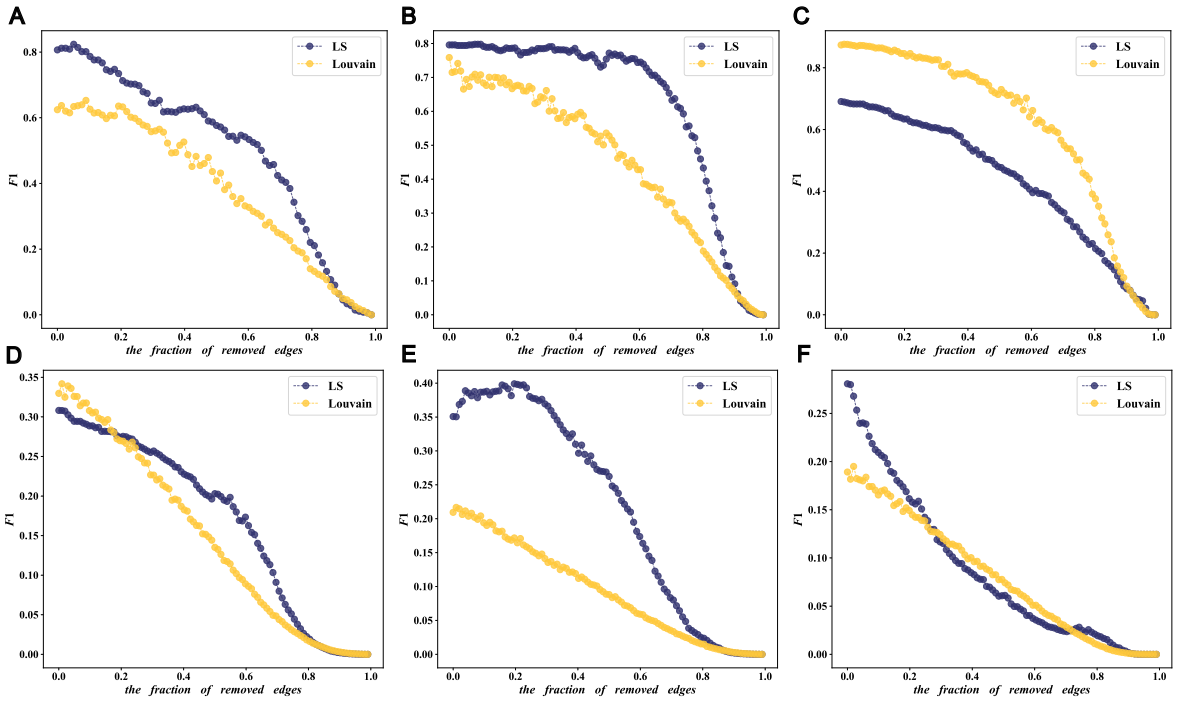
Supplementary Figure 20. Clustering result obtained by the LS algorithm on the Clusterable-data dataset. Grey points are identified as noise or can be regarded as cluster halos. Different clusters are indicated by different colors, and the index of cluster centers are labeled.



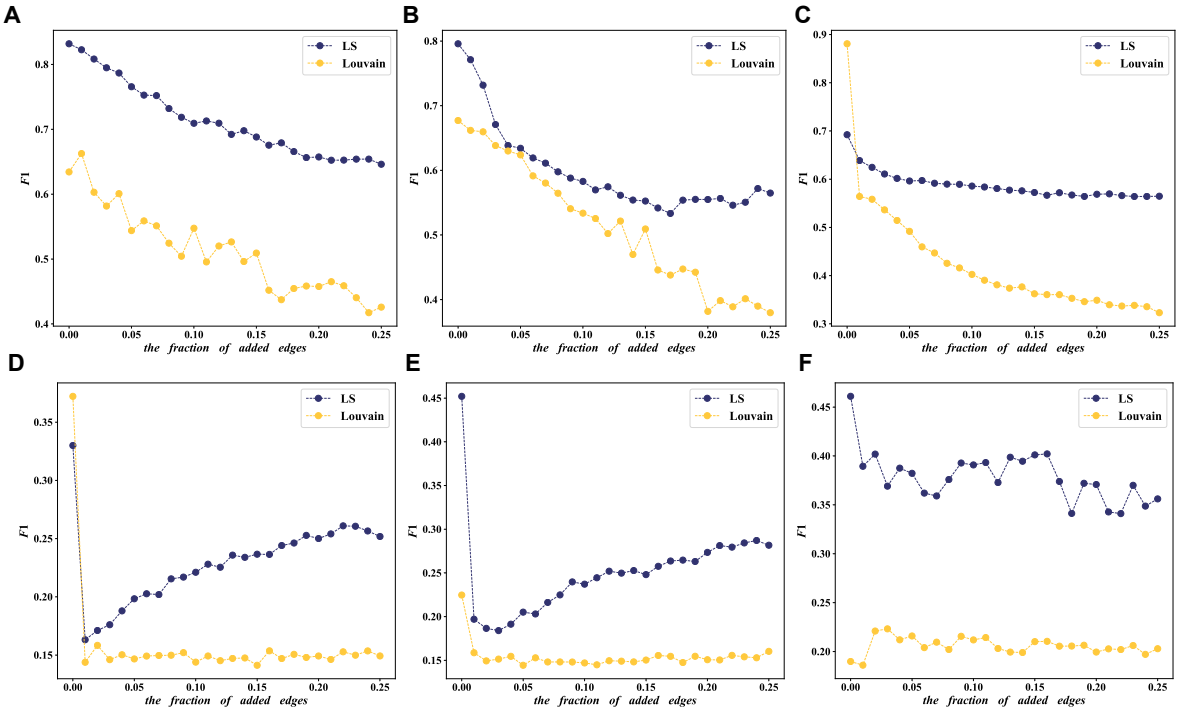
Supplementary Figure 21. Clustering results on S-sets dataset by the LS algorithm. S-sets dataset comprises 15 Gaussian clusters with different degree of cluster overlap and standard deviations. Along with the increase of the degree of cluster overlapping from (**A**)-(**D**). F_1 scores are 0.99, 0.94, 0.74, 0.62 for (**A**)-(**D**), respectively. Grey points are identified noisy data.

Supplementary Note 4.2: Robustness tests on networks

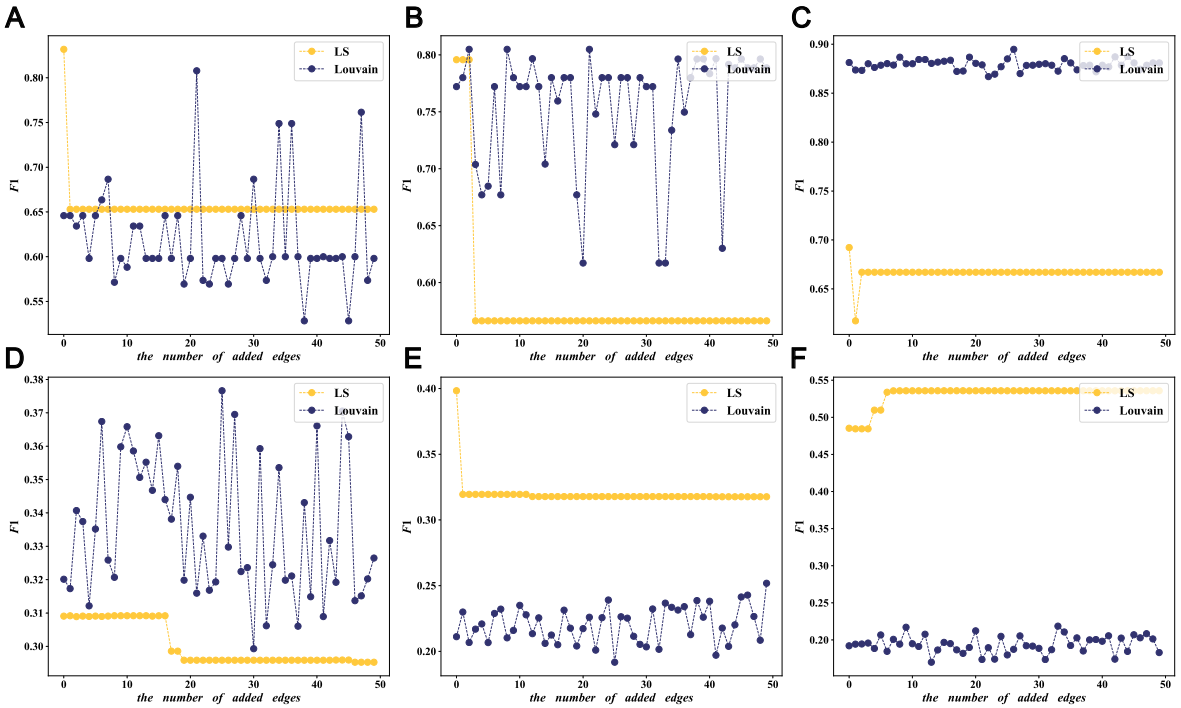
In networks, we first test on two common scenarios: randomly removing from or adding edges to the original networks. Our LS algorithm is generally more robust in most networks against random noises in both cases than the Louvain algorithm (see Supplementary Fig. 22 and Supplementary Fig. 23). Due to the nature of our method, the LS algorithm is quite vulnerable to targeted failures that connects largest nodes (i.e., we sequentially add edges between the largest node to the remaining forty-nine largest nodes if they were disconnected, see results in Supplementary Fig. 24). For example, when two local leaders are connected, one will be diminished as a follower, and its community will not be correctly detected. Very occasionally, when the whole network is dominated by a few large communities, adding edges between large nodes make the LS algorithm detect fewer communities and have a higher F_1-score (see Supplementary Fig. 24F). Results shown in Figs. 22-23 are average of ten realizations, and results in Supplementary Fig. 24 is just one realization, as the network structure is definite.



Supplementary Figure 22. The comparison between our LS algorithm and Louvian method on F_1 scores along with randomly removing edges in six networks with ground-truth community labels. (**A**) Zachary Karate Club [8], (**B**) Polbooks [16], (**C**) Polblogs [17], (**D**) Cora [18], (**E**) Citeseers [19], and (**F**) Pubmed [20]. Our LS algorithm is generally more robust against missing links in most networks, which is indicated by higher F_1-scores and a flatter decreasing.



Supplementary Figure 23. The comparison between our LS algorithm and Louvian method on F_1 scores along with randomly adding edges in six networks with ground-truth community labels. (**A**)-(**F**) correspond to the same networks in Supplementary Fig. 22. Our LS algorithm is generally more robust against added noisy links in most networks, which is indicated by higher F_1-scores.



Supplementary Figure 24. The targeted failure that adding edges between largest nodes. (**A**)-(**F**) correspond to the same networks in Supplementary Fig. 22. For such targeted failures, it usually diminish one previous community center as a follower, and in this case, the number of communities detected by the LS algorithm will decrease.

Supplementary References

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

[2] Newman, M. E. & Clauset, A. Structure and inference in annotated networks. *Nature Communications* **7**, 1–11 (2016).

[3] Fortunato, S. & Barthelemy, M. Resolution limit in community detection. *Proceedings of the National Academy of Sciences* **104**, 36–41 (2007).

[4] Ravasz, E. & Barabási, A.-L. Hierarchical organization in complex networks. *Physical review E* **67**, 026112 (2003).

[5] Girvan, M. & Newman, M. E. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences* **99**, 7821–7826 (2002).

[6] Evans, T. S. Clique graphs and overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment* P12037 (2010).

[7] Evans, T. S. American college football network files. doi: 10.6084/m9.figshare.93179.v2.

[8] Zachary, W. W. An information flow model for conflict and fission in small groups. *Journal of anthropological research* **33**, 452–473 (1977).

[9] Serrano, M. Á., Boguná, M. & Vespignani, A. Extracting the multiscale backbone of complex weighted networks. *Proceedings of the National Academy of Sciences* **106**, 6483–6488 (2009).

[10] Lee, M. J. *et al.* Uncovering hidden dependency in weighted networks via information entropy. *Physical Review Research* **3**, 043136 (2021).

[11] Blondel, V. D., Guillaume, J.-L., Hendrickx, J. M., de Kerchove, C. & Lambiotte, R. Local leaders in random networks. *Physical Review E* **77**, 036114 (2008).

[12] Fu, L. & Medico, E. Flame, a novel fuzzy clustering method for the analysis of dna microarray data. *BMC bioinformatics* **8**, 1–15 (2007).

[13] Chang, H. & Yeung, D.-Y. Robust path-based spectral clustering. *Pattern Recognition* **41**, 191–203 (2008).

[14] Gionis, A., Mannila, H. & Tsaparas, P. Clustering aggregation. *Acm transactions on knowledge discovery from data (tkdd)* **1**, 4–es (2007).

[15] Veenman, C. J., Reinders, M. J. T. & Backer, E. A maximum variance cluster algorithm. *IEEE Transactions on pattern analysis and machine intelligence* **24**, 1273–1280 (2002).

[16] Krebs, V. Social network analysis software & services for organizations, communities, and their consultants (2013).

[17] Adamic, L. A. & Glance, N. The political blogosphere and the 2004 us election: divided they blog. In *Proceedings of the 3rd international workshop on Link discovery*, 36–43 (2005).

[18] Rossi, R. & Ahmed, N. The network data repository with interactive graph analytics and visualization. In *Twenty-Ninth AAAI Conference on Artificial Intelligence* (2015).

[19] Bollacker, K. D., Lawrence, S. & Giles, C. L. Citeseer: An autonomous web agent for automatic retrieval and identification of interesting publications. In *Proceedings of the second international conference on Autonomous agents*, 116–123 (1998).

[20] Sen, P. *et al.* Collective classification in network data. *AI magazine* **29**, 93–93 (2008).

[21] Yang, J. & Leskovec, J. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems* **42**, 181–213 (2015).

[22] Guttman, A. R-trees: A dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, 47–57 (1984).

[23] Li, R. *et al.* Simple spatial scaling rules behind complex cities. *Nature Communications* **8**, 1–7 (2017).

[24] Jerdee, M., Kirkley, A. & Newman, M. Normalized mutual information is a biased measure for classification and community detection. *arXiv preprint arXiv:2307.01282* (2023).

[25] Peel, L., Larremore, D. B. & Clauset, A. The ground truth about metadata and community detection in networks. *Science Advances* **3**, e1602548 (2017).

[26] Rosvall, M. & Bergstrom, C. T. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences* **105**, 1118–1123 (2008).

[27] Mahmood, A., Small, M., Al-Maadeed, S. A. & Rajpoot, N. Using geodesic space density gradients for network community detection. *IEEE Transactions on Knowledge and Data Engineering* **29**, 921–935 (2016).

[28] Peixoto, T. P. Bayesian stochastic blockmodeling. In *Advances in network clustering and blockmodeling*, vol. 11, 289–332 (Wiley Online Library, 2019).

[29] Peixoto, T. P. Hierarchical block structures and high-resolution model selection in large networks. *Physical Review X* **4**, 011047 (2014).

[30] Shi, J. & Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence* **22**, 888–905 (2000).

[31] Qian, Y., Expert, P., Panzarasa, P. & Barahona, M. Geometric graphs from data to aid classification tasks with graph convolutional networks. *Patterns* **2**, 100237 (2021).

[32] Bullmore, E. T. & Bassett, D. S. Brain graphs: graphical models of the human brain connectome. *Annual review of clinical psychology* **7**, 113–140 (2011).

[33] Hoffmann, T., Peel, L., Lambiotte, R. & Jones, N. S. Community detection in networks without observing edges. *Science Advances* **6**, eaav1478 (2020).

[34] Berenhaut, K. S., Moore, K. E. & Melvin, R. L. A social perspective on perceived distances reveals deep community structure. *Proceedings of the National Academy of Sciences* **119** (2022).

[35] Pizzagalli, D. U., Gonzalez, S. F. & Krause, R. A trainable clustering algorithm based on shortest paths from density peaks. *Science Advances* **5**, eaax3770 (2019).

[36] Rodriguez, A. & Laio, A. Clustering by fast search and find of density peaks. *Science* **344**, 1492–1496 (2014).

[37] Ester, M., Kriegel, H.-P., Sander, J., Xu, X. *et al.* A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, vol. 96, 226–231 (1996).

[38] Bollobás, B. & Béla, B. *Random graphs*. 73 (Cambridge university press, 2001).

[39] Bunde, A. & Havlin, S. *Fractals and disordered systems* (Springer Science & Business Media, 2012).

[40] Li, D. *et al.* Percolation transition in dynamical traffic network with evolving critical bottlenecks. *Proceedings of the National Academy of Sciences* **112**, 669–672 (2015).

[41] Fisher, R. The use of multiple measures in taxonomic problems. *Ann. Eugenics* **7**, 179–188 (1936).

[42] Aeberhard, S., Coomans, D. & De Vel, O. Comparative analysis of statistical pattern recognition methods in high dimensional settings. *Pattern Recognition* **27**, 1065–1077 (1994).

[43] Chatfield, K., Lempitsky, V. S., Vedaldi, A. & Zisserman, A. The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC*, vol. 2, 8 (2011).

[44] Samaria, F. S. & Harter, A. C. Parameterisation of a stochastic model for human face identification. In *Proceedings of 1994 IEEE workshop on applications of computer vision*, 138–142 (IEEE, 1994).

[45] Sampat, M. P., Wang, Z., Gupta, S., Bovik, A. C. & Markey, M. K. Complex wavelet structural similarity: A new image similarity index. *IEEE transactions on image processing* **18**, 2385–2401 (2009).

[46] Campello, R. J., Moulavi, D. & Sander, J. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, 160–172 (Springer, 2013).