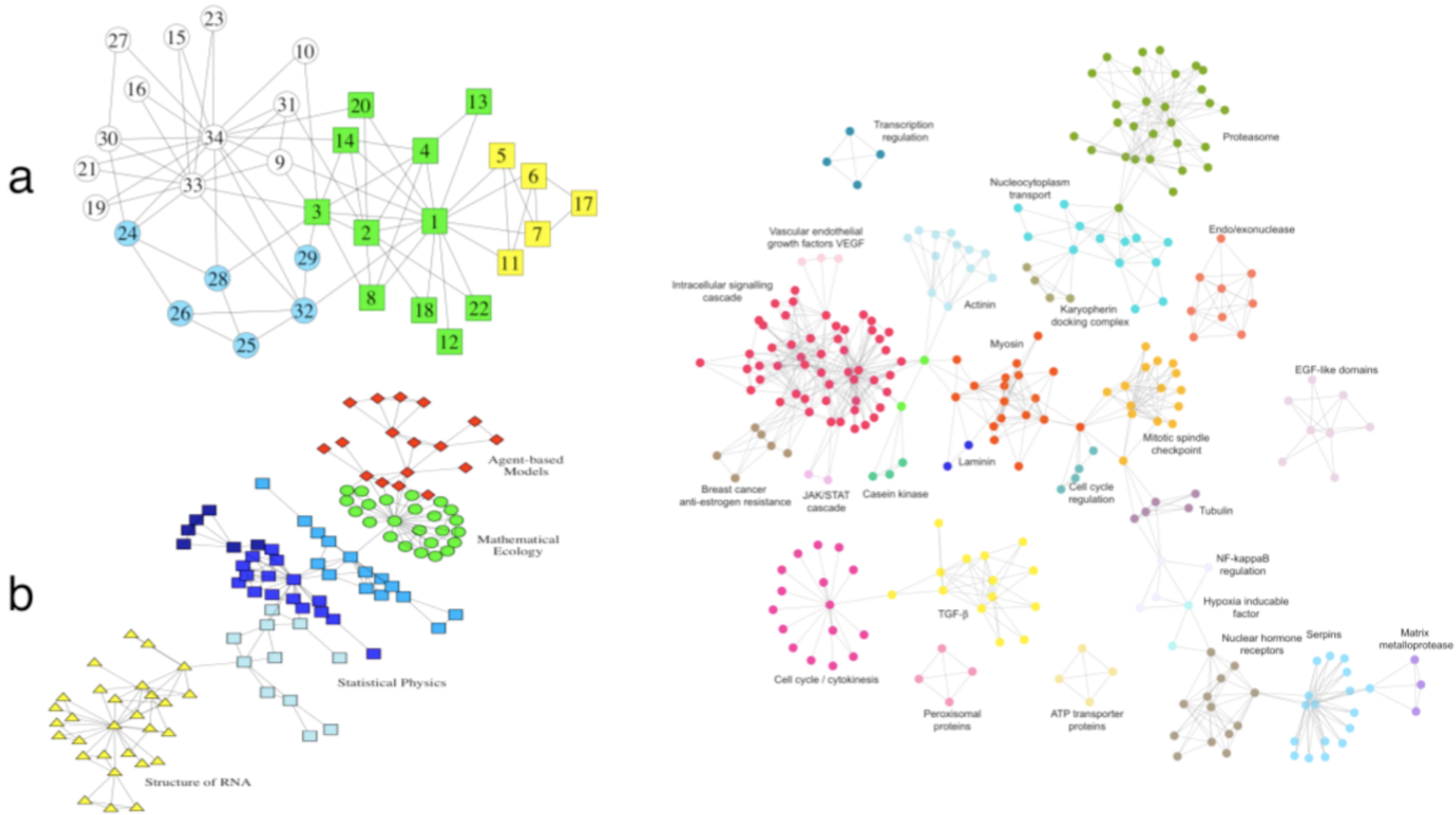
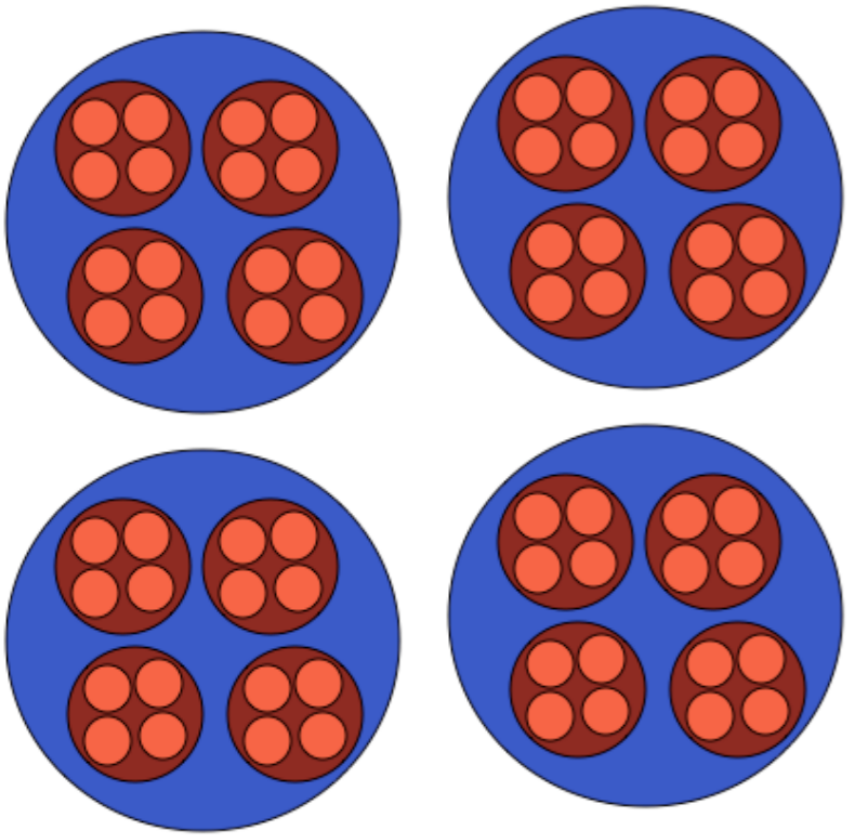
Properties: Modularity

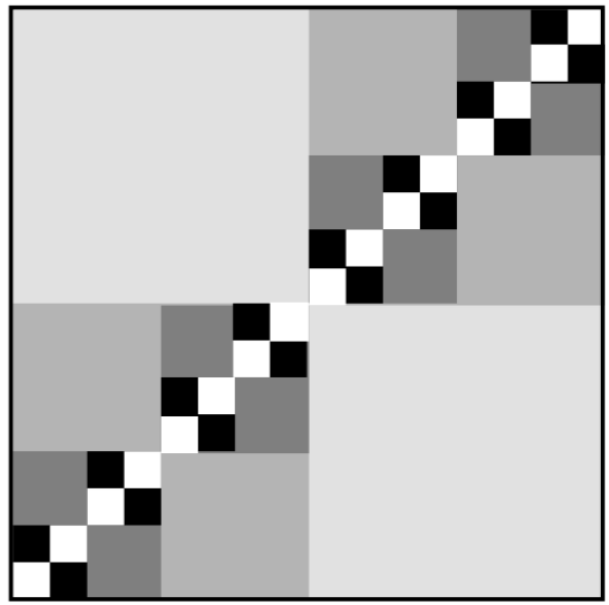
Observed in social, biological and information networks



Properties: Multi-level modularity (hierarchy)

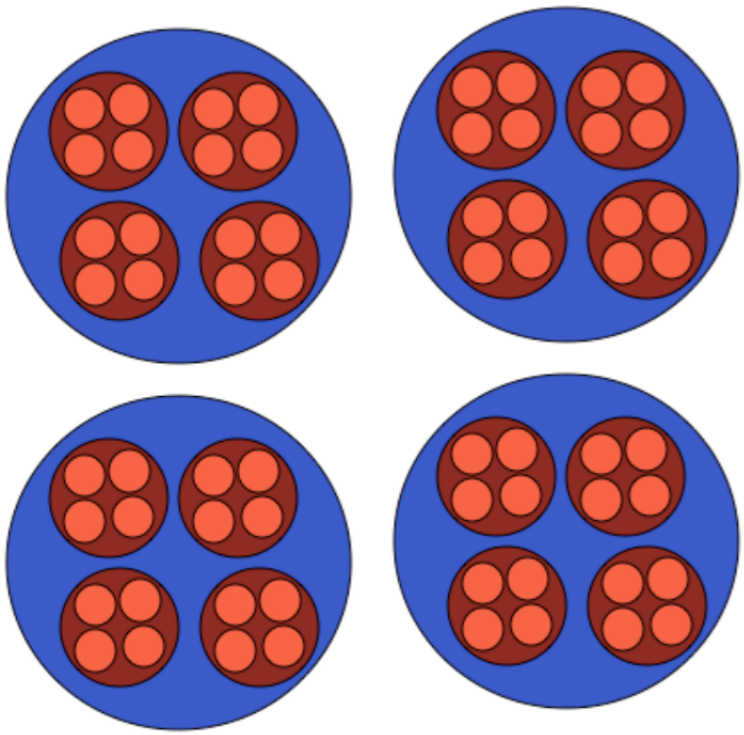
Networks have a hierarchical/multi-scale structure: modules within modules
Nested organization



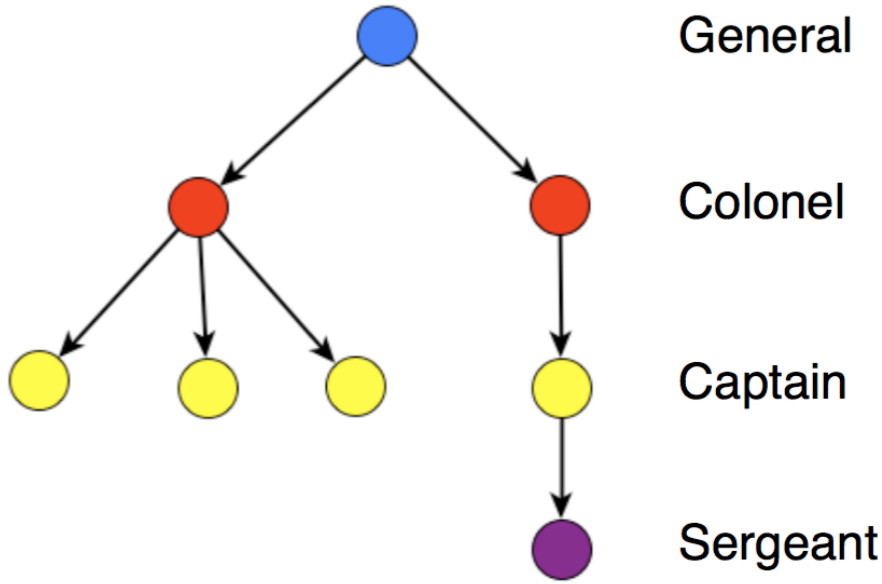


nb. Different notions of hierarchy

Hierarchy = multi-scale structure: modules within modules

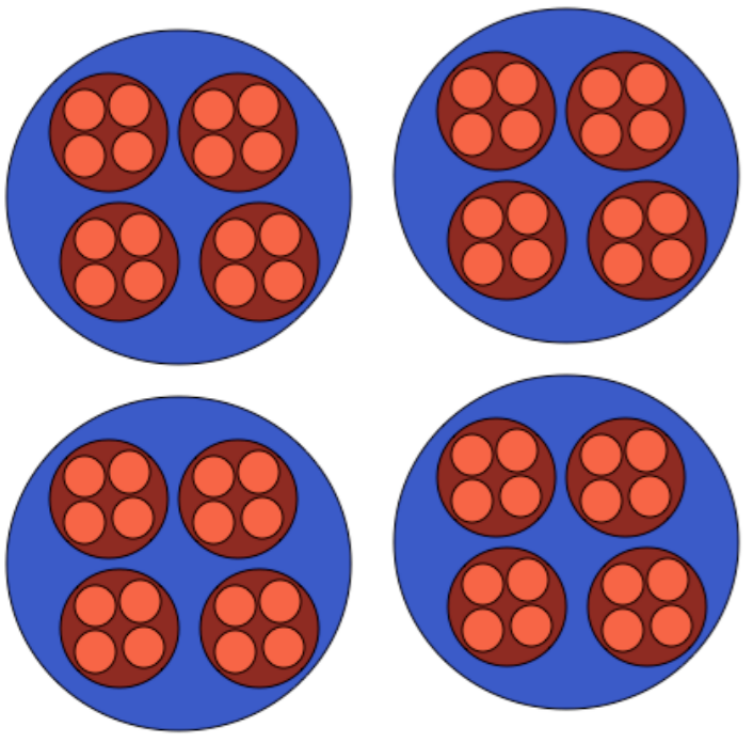


Hierarchy = subordination

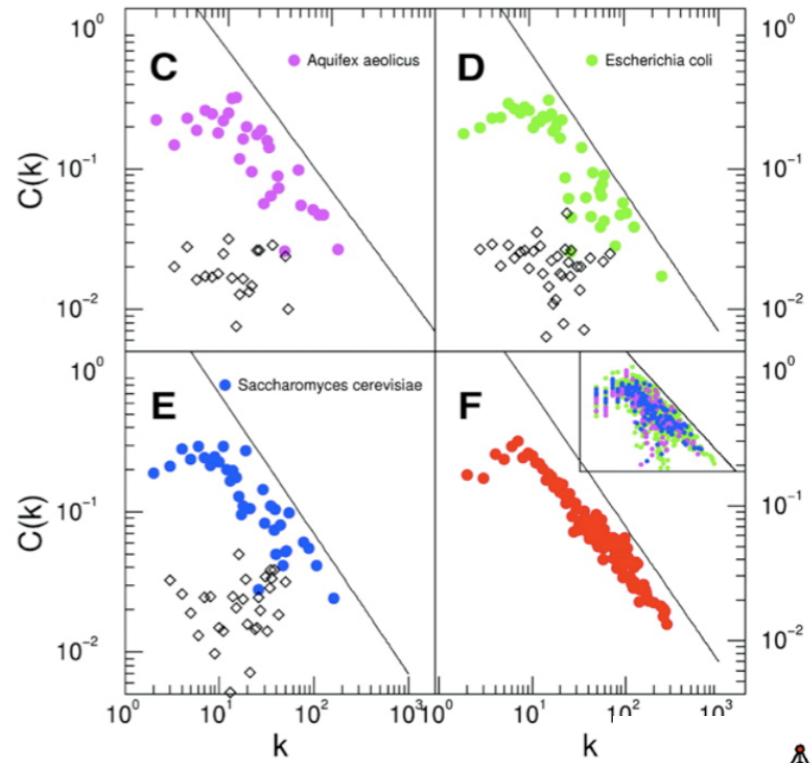


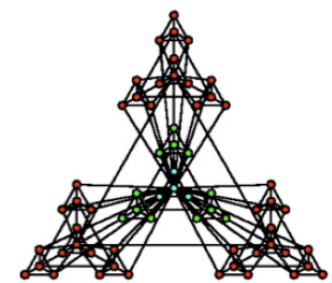
nb. Different notions of hierarchy

Hierarchy = multi-scale structure: modules within modules



Hierarchy of nodes with different degrees of “modularity” (clustering)

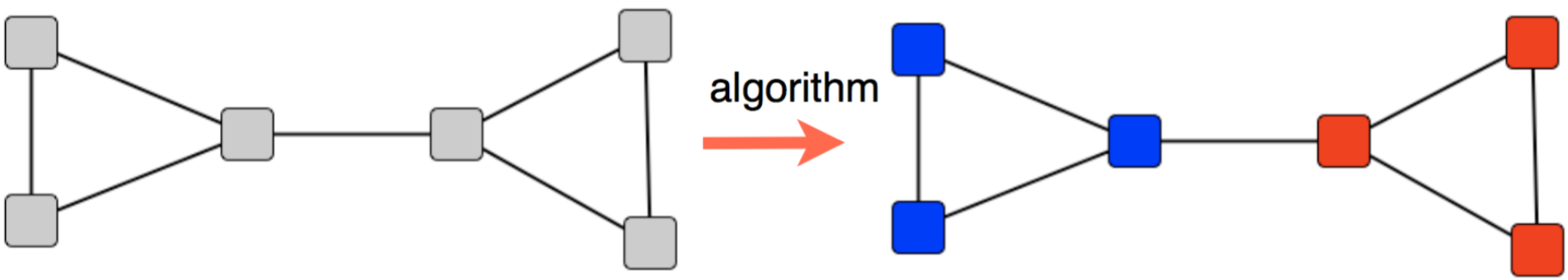




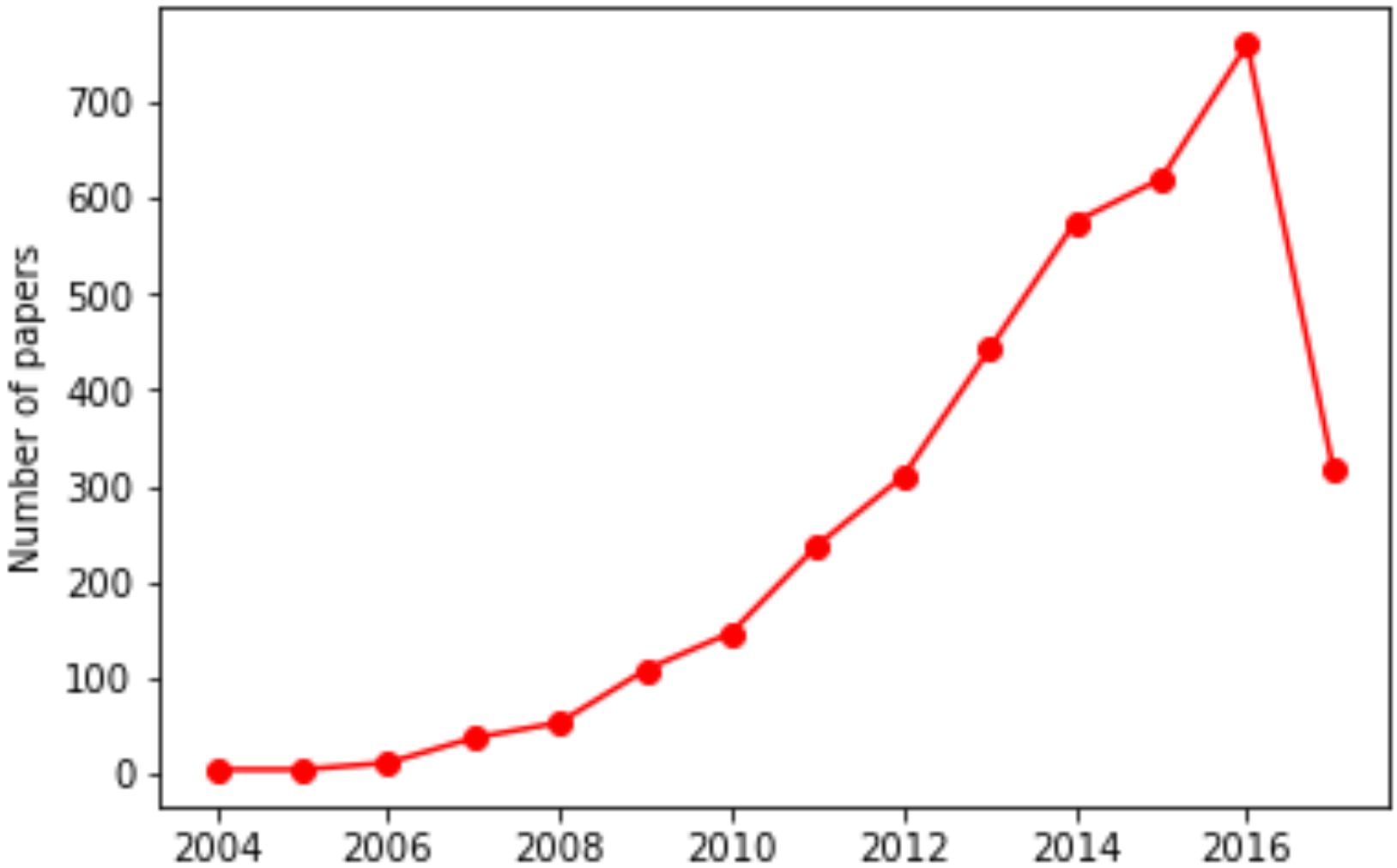
Hierarchical Organization of Modularity in Metabolic Networks, E. Ravasz et al.
http://barabasi.com/f/108.pdf

What is Community Detection?

Is it possible to uncover the (multi-scale) modular organisation of networks in an automated fashion? And please avoid false positives.
Given a graph, we look for an algorithm able to uncover its modules without specifying their number nor their size
The method should be scalable to accomodate very large networks, as often observed in the real-world.



“community detection”, SCOPUS, June 2017

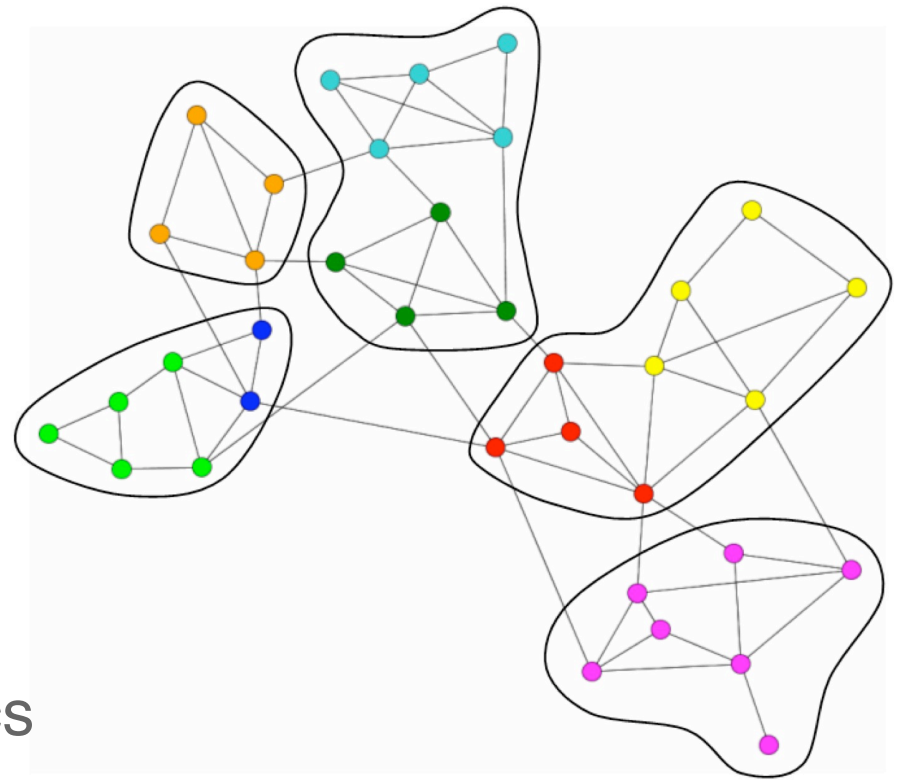


Identify the different motivations that underpin community detection
Help finding the appropriate algorithm for a given purpose

Algorithmic

Optimization

Statistics



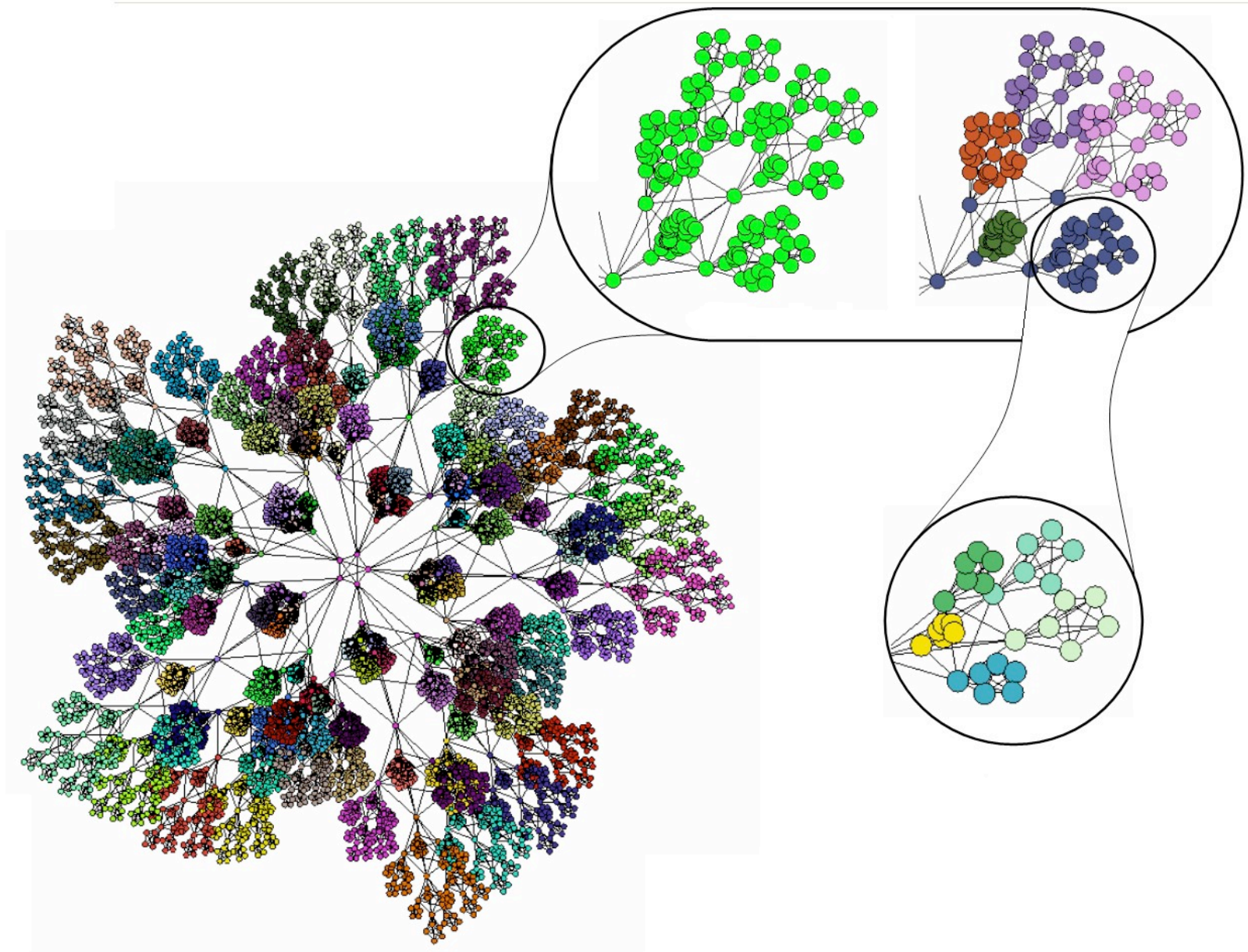
Graph theory

Statistical physics

Linear algebra

Dynamical systems

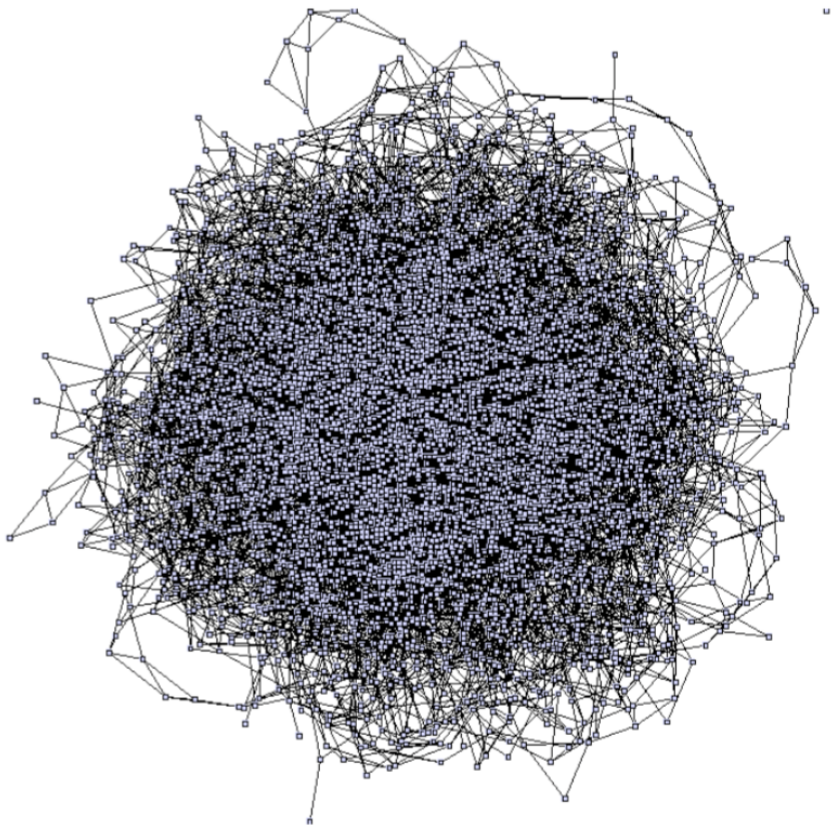
Identify and caracterize the multi-scale structure of networks



Why community detection?

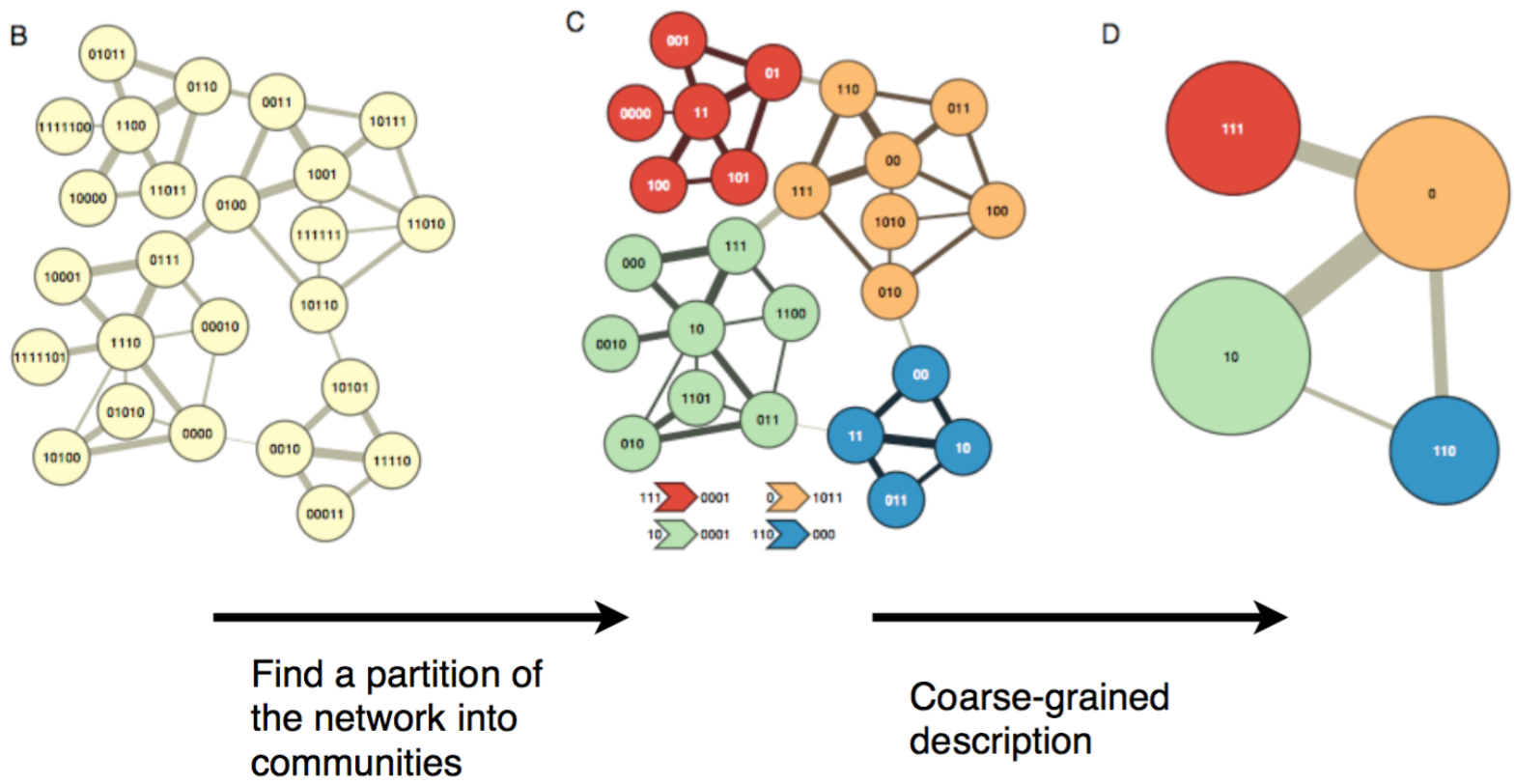
Graphs help us to comprehend in a visual way the global organisation of the system. This works extremely well when the graph is small but, as soon as the system is made of hundreds or thousands of nodes, a brute force representation typically leads to a meaningless cloud of nodes.

FIG. 2.—The direct relationship structure at Jefferson High



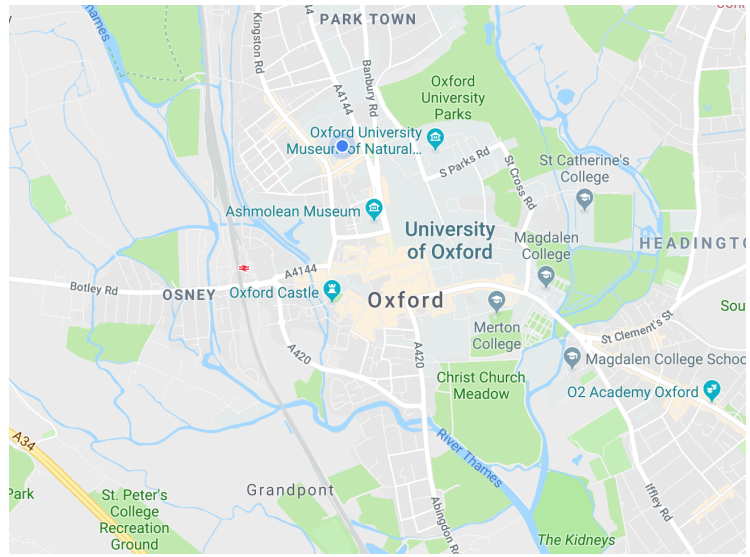
Why community detection?

Uncovering communities/modules helps to change the resolution of the representation and to draw a readable map of the network



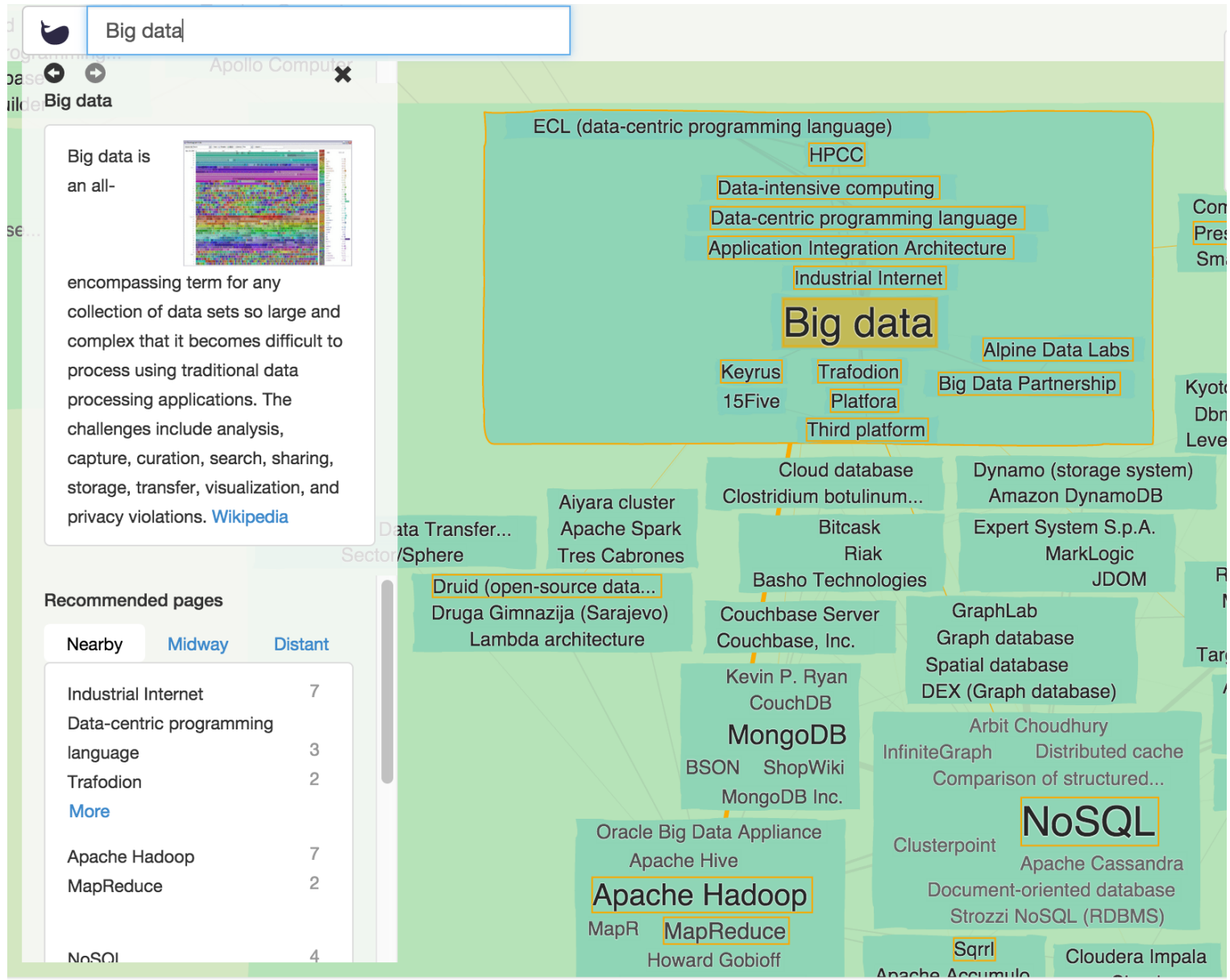
Martin Rosvall and Carl T. Bergstrom, PNAS 105, 1118 –1123 (2008)

Uncovering communities/modules helps to change the resolution of the representation and to draw a readable map of the network



Why community detection?

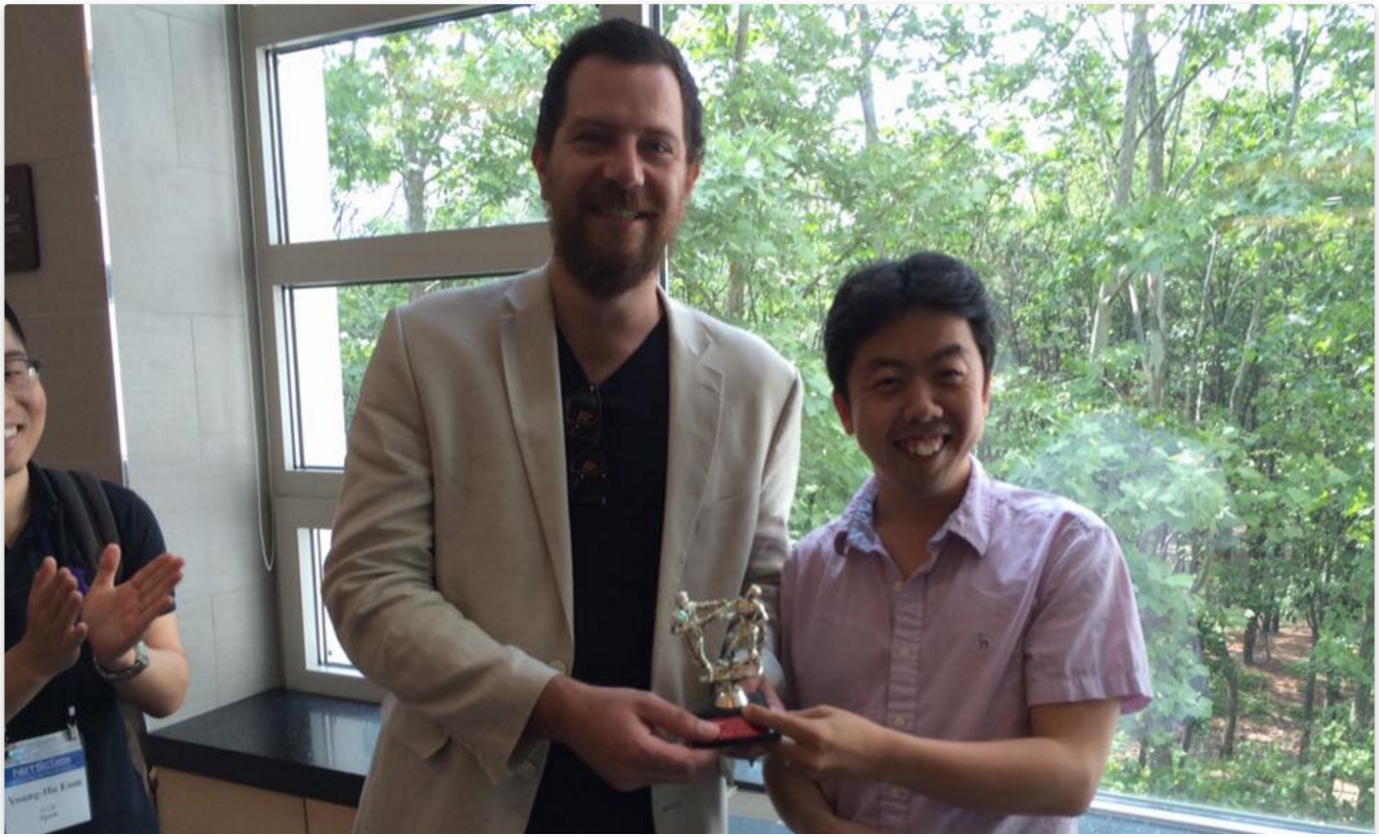
Uncovering communities/modules helps to change the resolution of the representation and to draw a readable map of the network



InfoBaleen

Why community detection?

Network Scientists with Karate Trophies



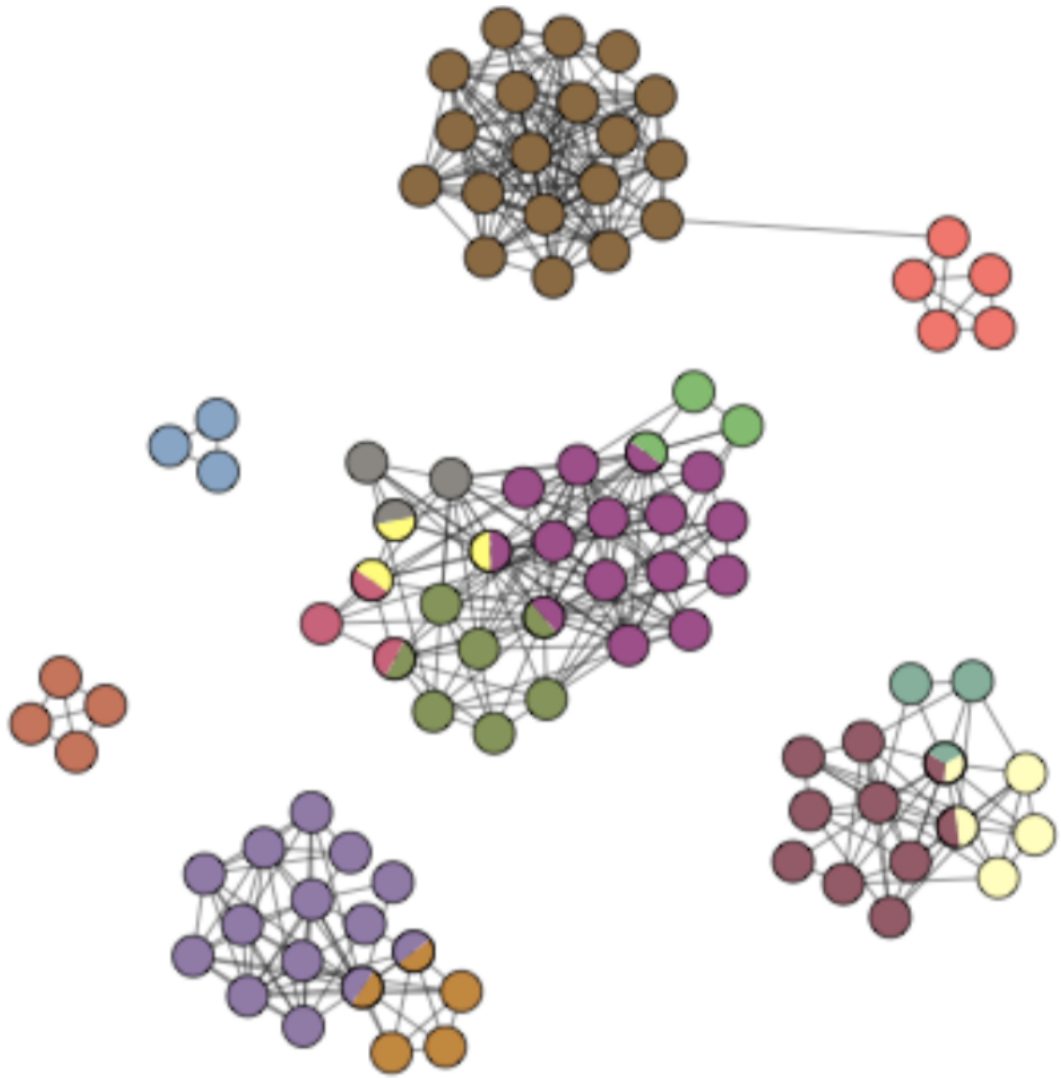
The first scientist at any conference on networks who uses Zachary's karate club as an example is inducted into the Zachary Karate Club Club, and awarded a prize. This tumblr records those moments.

RSS

ARCHIVE

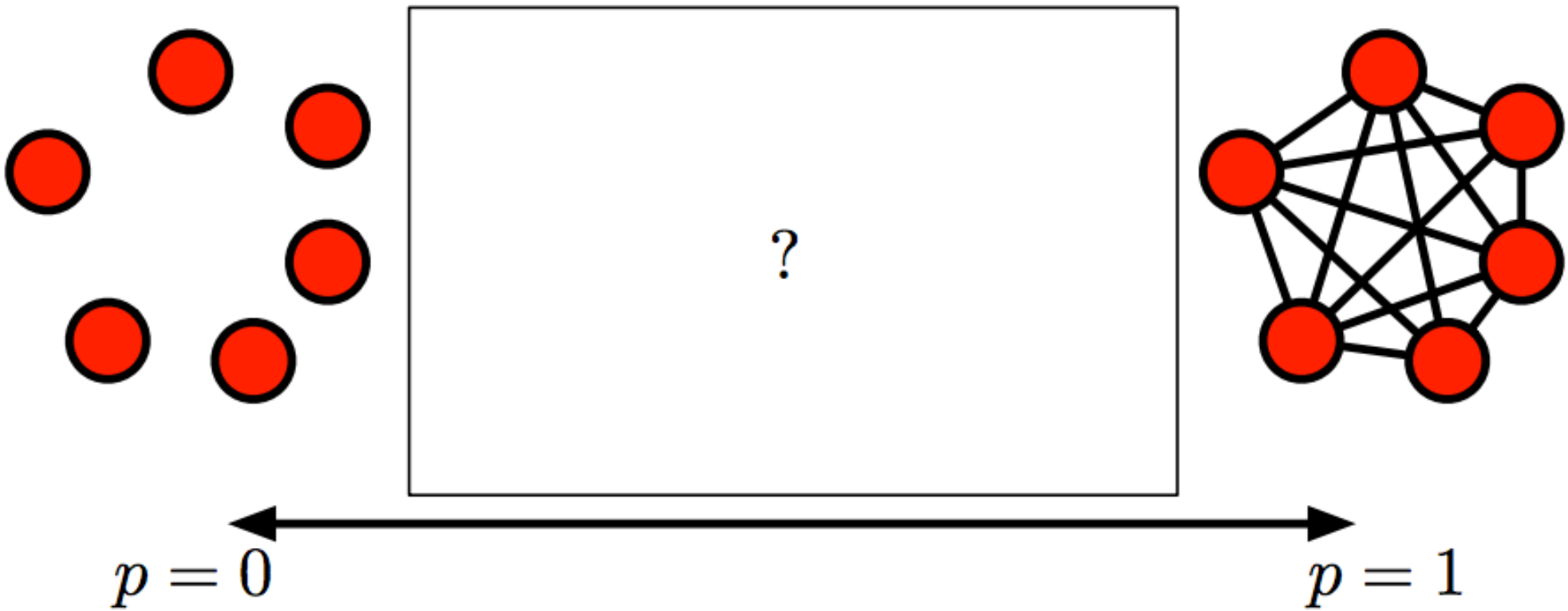
What is a “good” community?

A connected component is certainly a good community, in case of several components

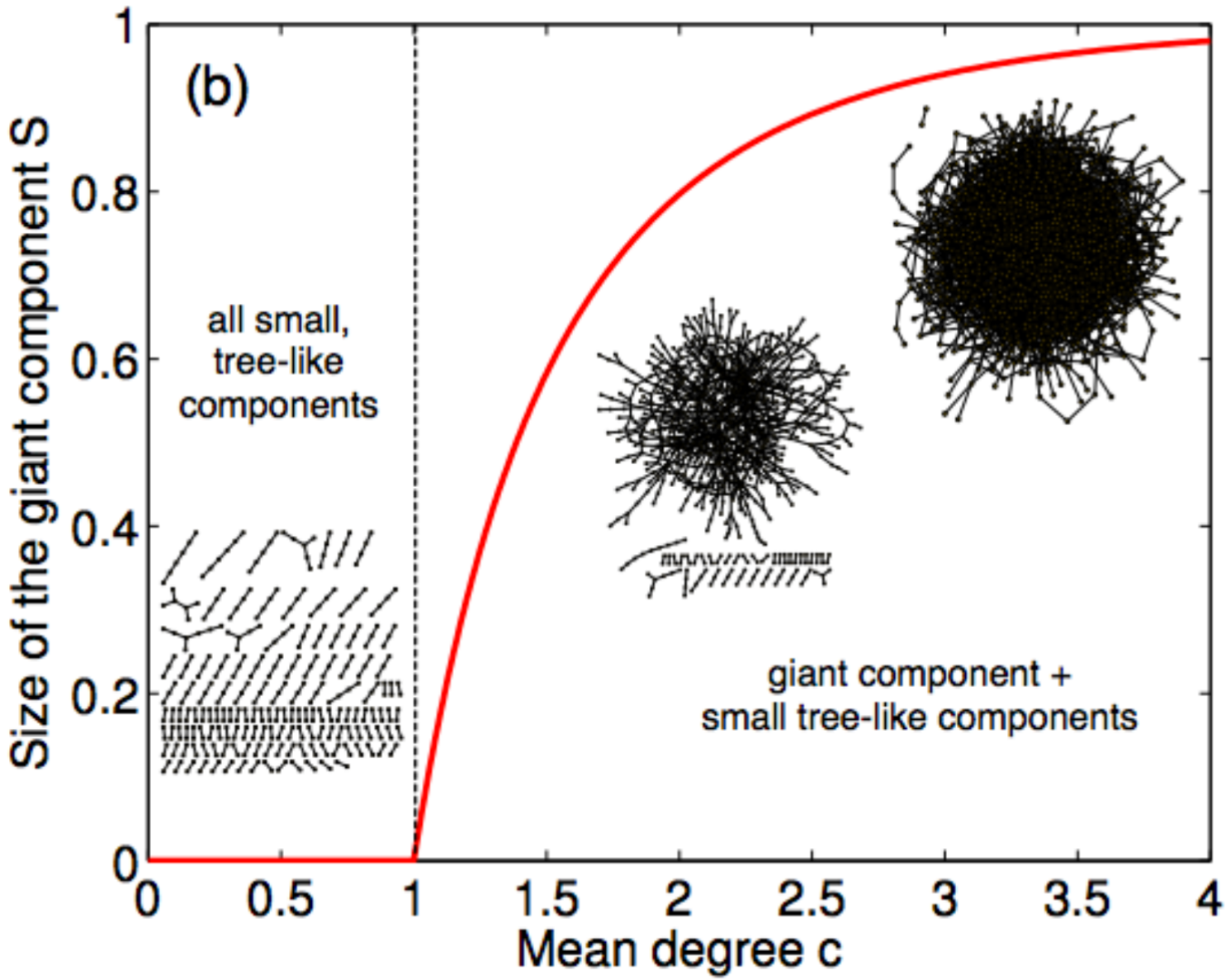


Percolation as a phase transition

Take the Erdos-Renyi network: how many disconnected components shall we expect depending on p?



Percolation as a phase transition



Community detection versus network partitioning

Both terms refer to the division of a network into dense groups

Graph partitioning: the number and size of the groups is **fixed** by the user. For instance, in a typical bisection problem: what is the best division of a network into 2 groups of equal size such that the number of links between the groups is minimised.
Application parallel computing: minimise inter-processor communication. Divides the system into the required number of groups, whatever the organisation of the system.

Community detection: the number and size of the groups are unspecified, but determined by the organisation of the network. Ideally, the method should be able to uncover a mixture of groups of different size in the same system. It should divide a network only when a good subdivision exists and leave it undivided otherwise.

Community detection versus network partitioning

Both terms refer to the division of a network into dense groups

Graph partitioning: the number and size of the groups is **fixed** by the user. For instance, in a typical bisection problem: what is the best division of a network into 2 groups of equal size such that the number of links between the groups is minimised.
Application parallel computing: minimise inter-processor communication. Divides the system into the required number of groups, whatever the organisation of the system.

Community detection: the number and size of the groups are unspecified, but determined by the organisation of the network. Ideally, the method should be able to uncover a mixture of groups of different size in the same system. It should divide a network only when a good subdivision exists and leave it undivided otherwise.

Different but similar methods
In both cases:
1) How to formalise the problem? **Definition of a good community**
2) How to solve it in practice? **Optimisation techniques to find it**

Graph bipartition

Definition of the problem:
Find the best division of a network into 2 groups of size n1 and n2 such that the cut size is minimal, where the cut size is the total number of links between different groups.

Solving the problem:
Looking through all bi-partitions and choose the one with the smallest cut size?
Impossible in practice, as the exhaustive search is extremely costly in terms of computer time

E.g. The number of ways to divide a network of 2n nodes into two groups of n and n nodes is:

$$\frac{(2n)!}{n!n!} \quad \xrightarrow{\text{Stirling}} \quad \frac{2^{n+1}}{\sqrt{n}}$$

No method solving exactly the problem in polynomial time for all networks…
But several existing heuristics allow to find approximate solutions in non-prohibitive times.

Spectral methods

$$R = \frac{1}{2} \sum_{ij \text{ in different groups}} A_{ij}$$

Let us denote by $s_i = \pm 1$ the assignment of node i

$$R = \frac{1}{2} \sum_{ij} A_{ij}(1 - s_i s_j) = \frac{1}{4} \sum_{ij} L_{ij} s_i s_j$$

By performing a spectral decomposition of the Laplacian matrix, one finds:

$$L_{ij} = \sum_{\alpha=2}^{N} \lambda_\alpha v_{\alpha,i} v_{\alpha,j}$$

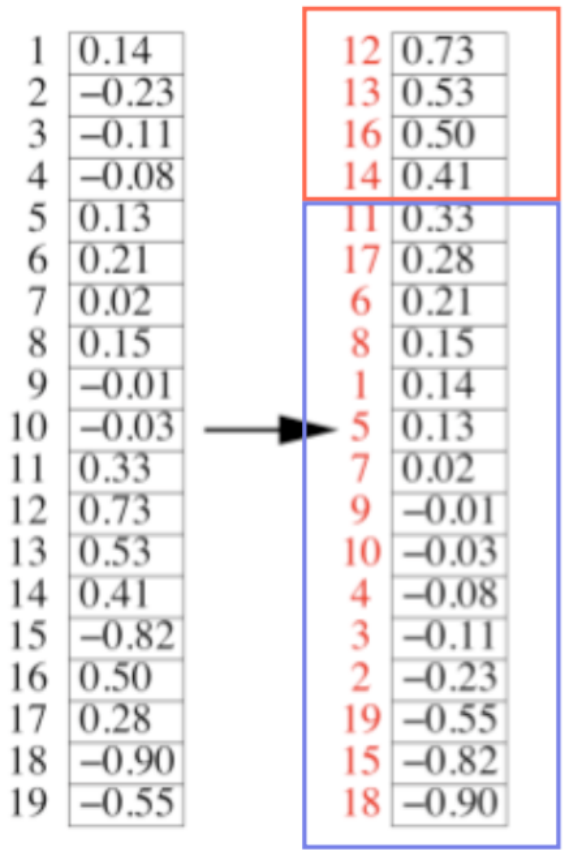
If there is no condition on si, the optimal solution would be $s_i = v_{2,i}$

$$R = \frac{1}{4} \lambda_2$$

But this solution is not a partition (except in extremely trivial situations) and it probably does not satisfy the required size of the groups.

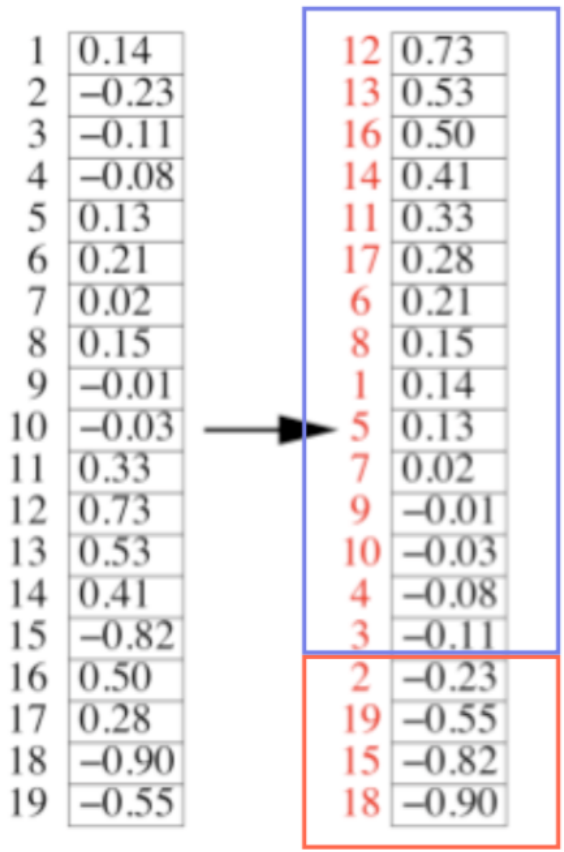
Spectral methods

Approximation: If one wants a split into n1 and n2=n-n1 vertices, one orders the components of the Fiedler vector from the largest positive to the smallest negative and picks the n1 largest (smallest) components of the Fiedler vector



1	0.14
2	−0.23
3	−0.11
4	−0.08
5	0.13
6	0.21
7	0.02
8	0.15
9	−0.01
10	−0.03
11	0.33
12	0.73
13	0.53
14	0.41
15	−0.82
16	0.50
17	0.28
18	−0.90
19	−0.55

12	0.73
13	0.53
16	0.50
14	0.41
11	0.33
17	0.28
6	0.21
8	0.15
1	0.14
5	0.13
7	0.02
9	−0.01
10	−0.03
4	−0.08
3	−0.11
2	−0.23
19	−0.55
15	−0.82
18	−0.90



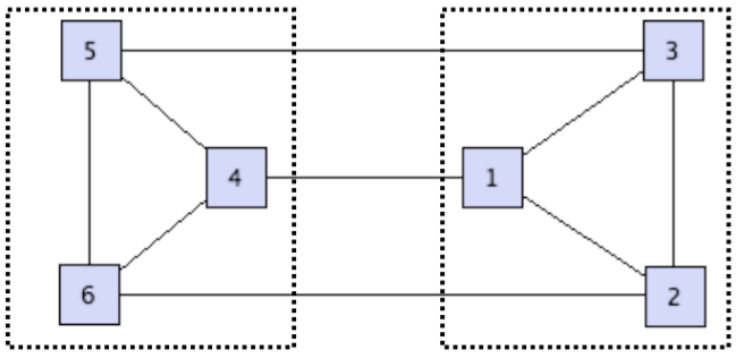
1	0.14
2	−0.23
3	−0.11
4	−0.08
5	0.13
6	0.21
7	0.02
8	0.15
9	−0.01
10	−0.03
11	0.33
12	0.73
13	0.53
14	0.41
15	−0.82
16	0.50
17	0.28
18	−0.90
19	−0.55

12	0.73
13	0.53
16	0.50
14	0.41
11	0.33
17	0.28
6	0.21
8	0.15
1	0.14
5	0.13
7	0.02
9	−0.01
10	−0.03
4	−0.08
3	−0.11
2	−0.23
19	−0.55
15	−0.82
18	−0.90

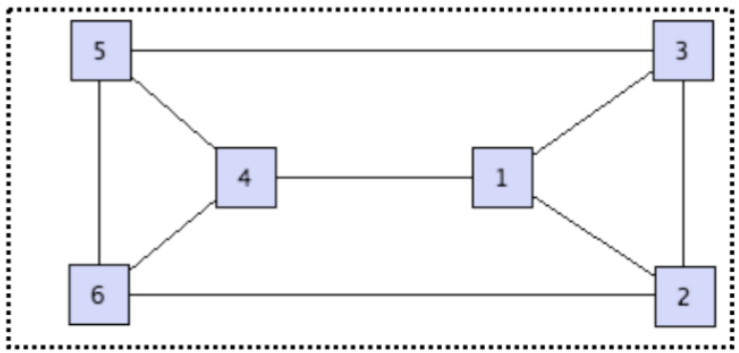
Complexity: O(N^2) on sparse networks

Community detection

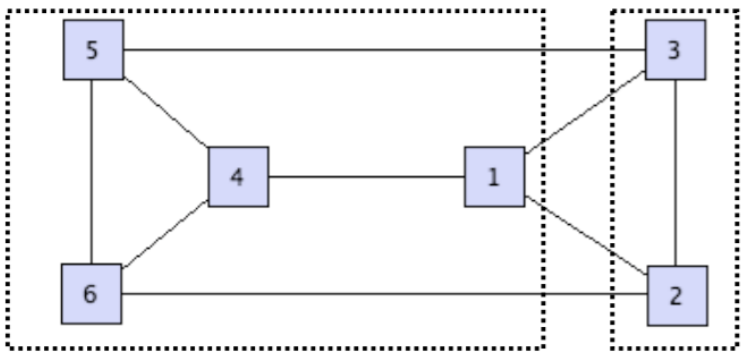
What is the best partition of a network into modules?
How do we rank the quality of partitions of different sizes?



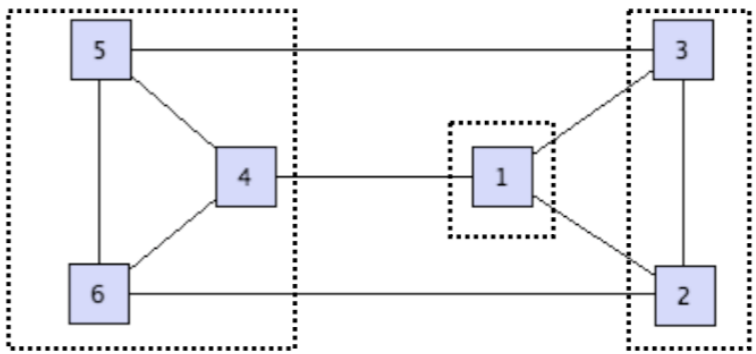
Q1



Q2



Q3



Q4

.....

Newman-Girvan Modularity

Q = fraction of edges within communities - expected fraction of such edges

Let us attribute each node i to a community ci

$$Q = \frac{1}{2m} \sum_{i,j} \Big[A_{ij} - P_{ij} \Big] \delta(c_i, c_j)$$

$P_{ij} = \frac{k_i k_j}{2m}$ expected number of links between i and j

$$Q_C = \frac{1}{2m} \sum_{i,j} \Big[A_{ij} - k_i k_j / 2m \Big] \delta(c_i, c_j) \qquad Q_C \in [-1/2, 1]$$

Allows to compare partitions made of different numbers of modules

M.E.J. Newman and M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E, 69, 026113, 2004.

Note on the null model

Random network with constrained degrees $P_{ij} = \frac{k_i k_j}{2m}$

What if one has extra information about the nodes?

Directed networks -> $P_{ij} = k_i^{\text{in}} k_j^{\text{out}} / m$

Spatially-embedded networks -> $P_{ij}^{\text{Spa}} = N_i N_j f(d_{ij})$

Or if the information on the degrees is expected to be irrelevant:

$$P_{ij} = \langle k \rangle^2 / 2m = \langle k \rangle / N$$

$$\sum_{ij} P_{ij} = \sum_{ij} A_{ij} = 2m$$

Modularity

Property 1 *A partition where all the vertices are grouped into the same community has a modularity equal to zero. This proves to be simply shown from the definition of the null model* $\frac{k_i k_j}{2m}$ *for which* $\sum_{ij} \frac{k_i k_j}{2m} = 2m$ *and from the expression of modularity in this particular case*

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] = 0 \ . \tag{3}$$

This property implies that any partition with a positive modularity is better than this trivial one, but also that it is always possible to find a partition such that $Q \geq 0$.

Modularity

Property 2 *If a partition contains a disconnected community, it is always preferable (in terms of modularity) to split this community into connected communities. Let us consider, for the sake of simplicty, the case of a disconnected community C_1 formed by two connected subgraphs C_{11}, C_{12}. In this case, modularity is given by*

$$
\begin{aligned}
Q = & \frac{1}{2m}\left[\sum_{C\neq C_1}\sum_{i,j\in C}(A_{ij}-\frac{k_ik_j}{2m})+\sum_{i,j\in C_1}(A_{ij}-\frac{k_ik_j}{2m})\right] \\
= & \frac{1}{2m}\left[\sum_{C\neq C_1}\sum_{i,j\in C}(A_{ij}-\frac{k_ik_j}{2m})+\sum_{i,j\in C_{11}}(A_{ij}-\frac{k_ik_j}{2m})\right. \\
& \left.+\sum_{i,j\in C_{12}}(A_{ij}-\frac{k_ik_j}{2m})+2\sum_{i\in C_{11},j\in C_{12}}(A_{ij}-\frac{k_ik_j}{2m})\right. .
\end{aligned}
\tag{4}
$$

Given that $A_{ij} = 0$ if $i \in C_{11}, j \in C_{12}$, the sum $\sum_{i\in C_{11},j\in C_{12}}$ is composed uniquely of negative terms and it is thus preferable to split the community into two subcommunities.

This property implies that any partition made of disconnected communities is sub-optimal and that the optimal partition of a graph is only made of connected communities.

Modularity Optimization

Optimization of modularity is an NP-complete problem

Need for efficient heuristics

Optimization: Spectral methods

Similar method to one for minimizing the cut, based on the spectral properties of the modularity matrix Q

$$Q_{ij} = A_{ij} - \frac{k_i k_j}{2m}$$

Let us first focus on the best division of the network into 2 communities.

Let us denote by $s_i = \pm 1$ the assignment of node i $\delta(c_i, c_j) = \frac{1}{2}(s_i s_j + 1)$

$$Q = \frac{1}{2m} \sum_{ij} Q_{ij} \delta(c_i, c_j) = \frac{1}{4m} \sum_{ij} Q_{ij} s_i s_j$$

By performing a spectral decomposition of the modularity matrix, one finds:

$$Q_{ij} = \sum_{\alpha=1}^{N} \lambda_\alpha v_{\alpha,i} v_{\alpha,j}$$

si is chosen to be as similar to the dominant eigenvector

$$s_i = 1 \text{ if } v_{N,i} > 0$$
$$s_i = -1 \text{ if } v_{N,i} < 0$$

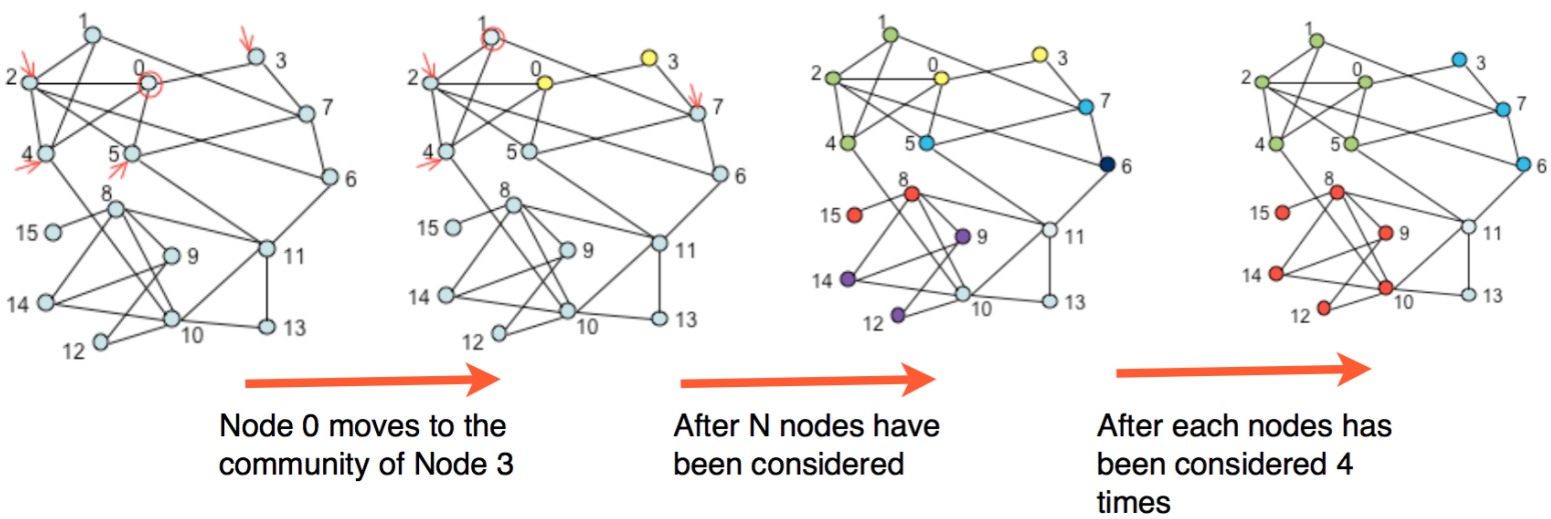
M.E.J. Newman, Finding community structure in networks using the eigenvectors of matrices, Phys. Rev. E, vol. 74, 036104, 2006.

Optmization: Greedy optimization

Louvain: multi-scale, agglomerative and greedy

The algorithm is based on two steps that are repeated iteratively. **First phase**: Find a local maximum

1) Give an order to the nodes (0,1,2,3,...., N-1)
2) Initially, each node belongs to its own community (N nodes and N communities)
3) One looks through all the nodes (from 0 to N-1) in an ordered way. The selected node looks among its neighbours and adopt the community of the neighbour for which the increase of modularity is maximum (and positive).
4)This step is performed iteratively until a local maximum of modularity is reached (each node may be considered several times).

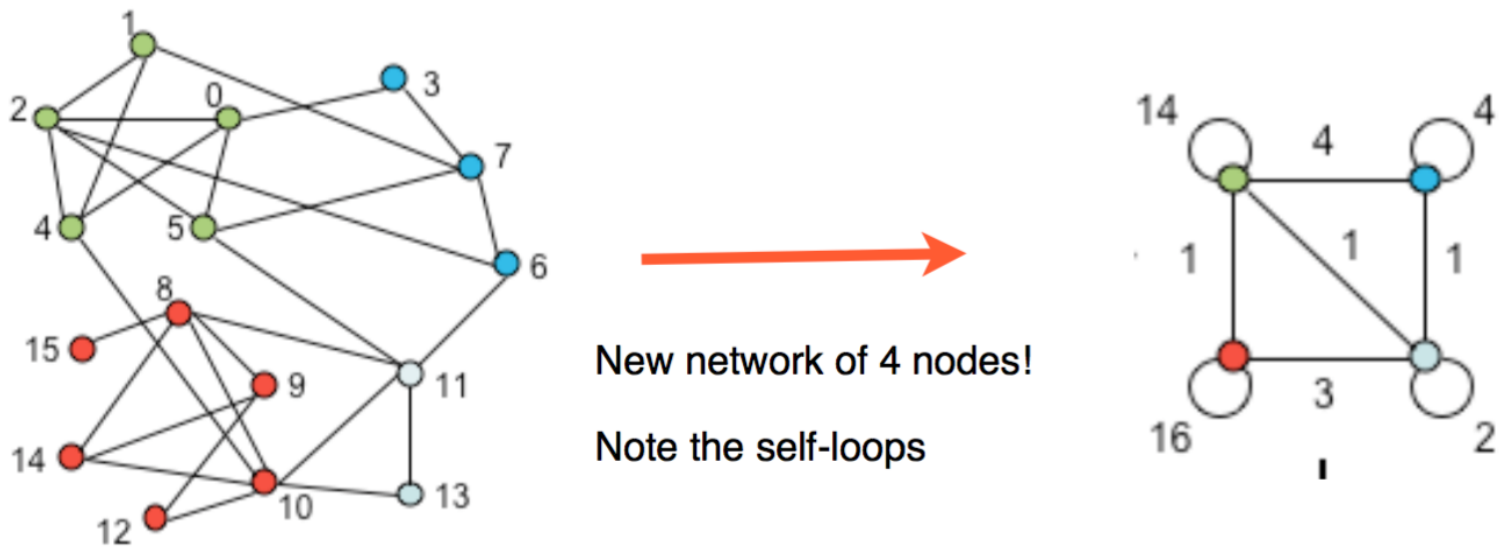


V.D. Blondel, J.-L. Guillaume, R. Lambiotte and E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech., P10008, 2008.

Optmization: Greedy optimization

Louvain: multi-scale, agglomerative and greedy

Once a local maximum has been attained, **second phase**:
We build a new network whose nodes are the communities. The weight of the links between communities is the total weight of the links between the nodes of these communities.



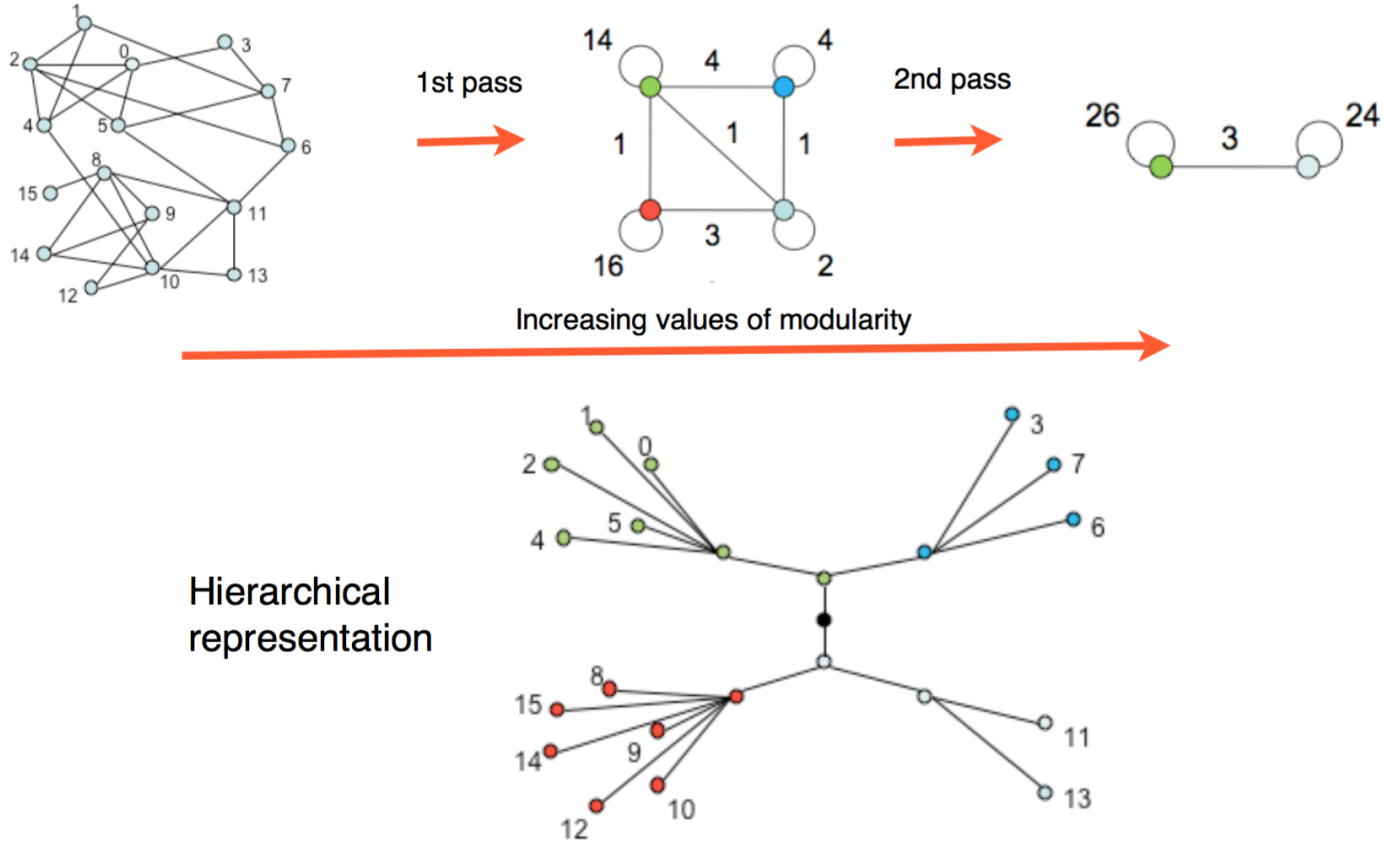
In typical realizations, the number of nodes diminishes drastically at this step.

V.D. Blondel, J.-L. Guillaume, R. Lambiotte and E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech., P10008, 2008.

Optmization: Greedy optimization

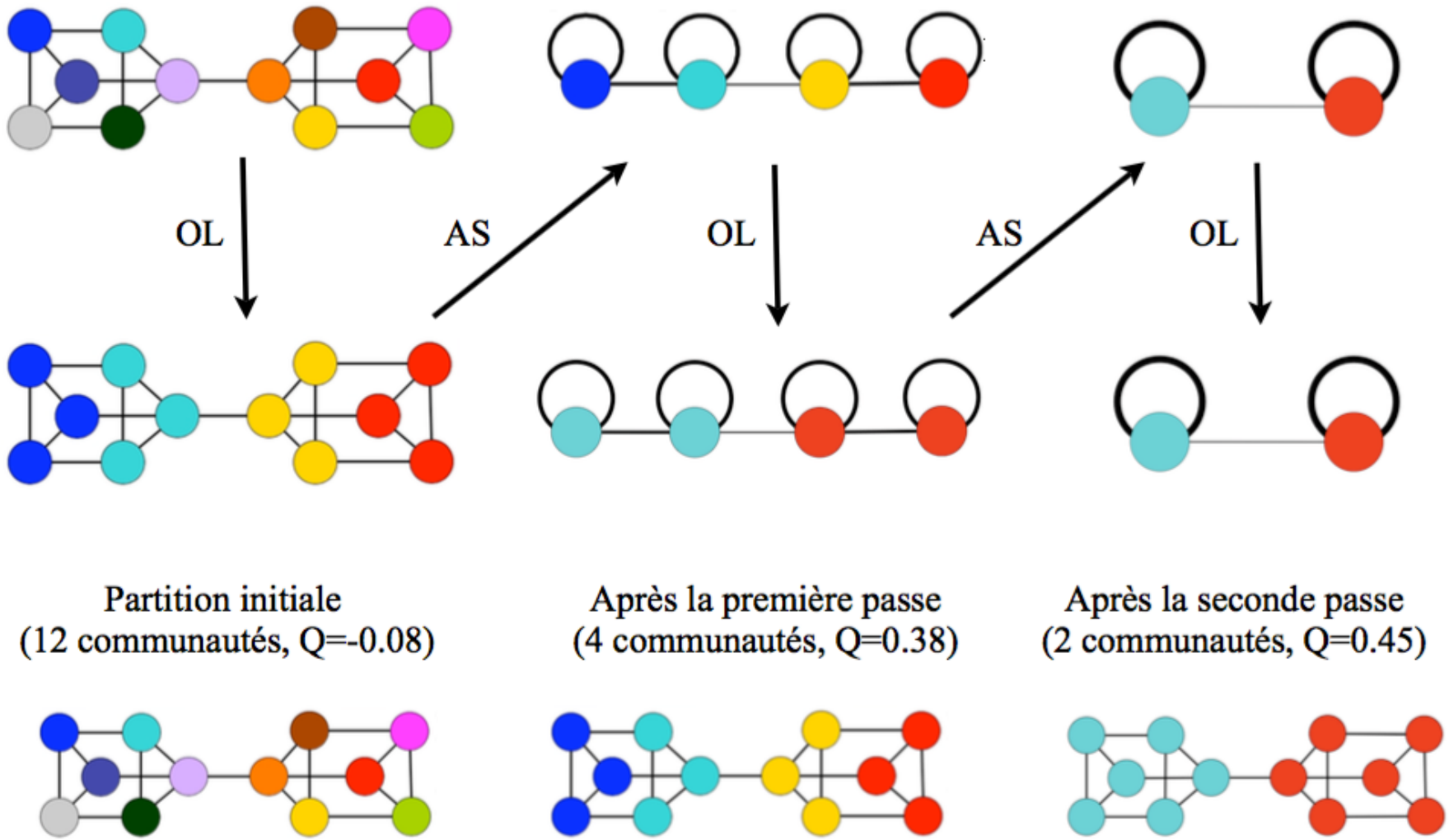
Louvain: multi-scale, agglomerative and greedy

The two steps are repeated iteratively, thereby leading to a hierarchical decomposition of the network.
Multi-scale optimisation: local search first among neighbours, then among neighbouring communities, etc.



V.D. Blondel, J.-L. Guillaume, R. Lambiotte and E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech., P10008, 2008.

Optimization: Greedy optimization



V.D. Blondel, J.-L. Guillaume, R. Lambiotte and E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech., P10008, 2008.

Louvain

Algorithm 1 Pseudo-code of the community detection algorithm.

```
1: Community detectionG initial graph
2: repeat
3:    Place each vertex of G into a single community
4:    Save the modularity of this decomposition
5:    while there are moved vertices do
6:       for all vertex n of G do
7:          c ← neighboring community maximizing the modularity increase
8:          if c results in a strictly positive increase then
9:             move n from its community to c
10:         end if
11:      end for
12:   end while
13:   if the modularity reached is higher than the initial modularity, then
14:      end ← false
15:      Display the partition found
16:      Transform G into the graph between communities
17:   else
18:      end ← true
19:   end if
20: until end
```

Louvain

The efficiency of the algorithm partly resides in the fact that the variation of modularity Δ_{ij} obtained by moving a vertex i from its community to the community of one of its neighbors j can be calculated with only local information. In practice, the variation of modularity is calculated by removing i from its community $\Delta_{remove;i}$ (this is only done once) then inserting it into the community of j $\Delta_{insert;ij}$ for each neighbor j of i. The variation is therefore: $\Delta_{ij} = \Delta_{remove;i} + \Delta_{insert;ij}$.

Optimization: Greedy optimization

Louvain: multi-scale, agglomerative and greedy

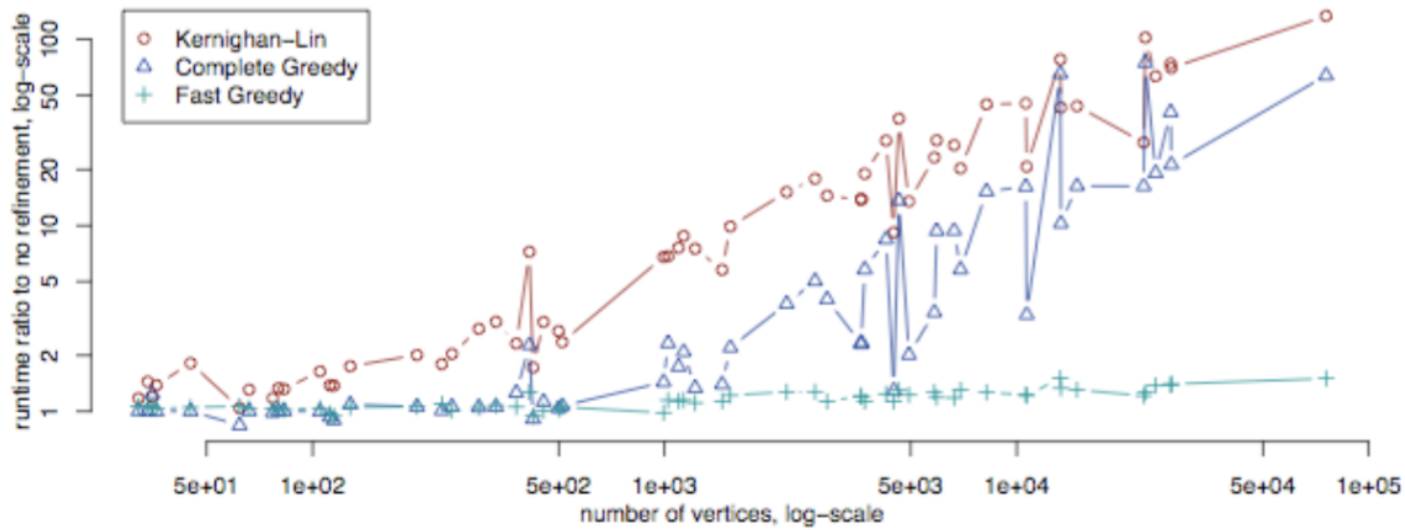
Very fast: O(N) in practice. The only limitation being the storage of the network in main memory
Good accuracy (among greedy methods)

	Karate	Arxiv	Internet	Web nd.edu	Phone	Web uk-2005	Web WebBase 2001
Nodes/links	34/77	9k/24k	70k/351k	325k/1M	2.04M/5.4M	39M/783M	118M/1B
CNM	.38/0s	.772/3.6s	.692/799s	.927/5034s	-/-	-/-	-/-
PL	.42/0s	.757/3.3s	.729/575s	.895/6666s	-/-	-/-	-/-
WT	.42/0s	.761/0.7s	.667/62s	.898/248s	.553/367s	-/-	-/-
Our algorithm	.42/0s	.813/0s	.781/1s	.935/3s	.76/44s	.979/738s	.984/152mn

V.D. Blondel, J.-L. Guillaume, R. Lambiotte and E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech., P10008, 2008.

How to test the methods?

Test the heuristics: what is the value of Q obtained for different algorithms? Time complexity?



graph	size	subdivision		coarsening		local search		math prog		SS+ML
karate [42]	34	[29]	.419	[41]	.4198	[12]	.4188	[1]	.4197	.4197
dolphins [22]	62	[29]	*.4893*	[31]	*.5171*	[33]	*.5285*	[40]	.5285	.5276
polBooks [21]	105	[29]	*.3992*	[37]	*.5269*	[4]	*.5204*	[1]	.5272	.5269
afootball [14]	115	[39]	.602	[41]	.605	[4]	*.6045*	[1]	.6046	.6002
jazz [15]	198	[29]	.442	[9]	.4409	[12]	.4452	[1]	.445	.4446
celeg_metab [12]	453	[29]	.435	[36]	.450	[12]	.4342	[1]	.450	.4452
email [17]	1133	[29]	.572	[9]	.5569	[12]	.5738	[1]	.579	.5774
Erdos02 [16]	6927	[29]	*.5969*	[32]	.6817	[33]	*.7094*			.7162
PGP_main [5]	11k	[29]	.855	[9]	.7462	[12]	.8459			.8841
cmat03_main [25]	28k	[29]	.723	[41]	.761	[12]	.6790			.8146
ND_edu [2]	325k			[7]	.927	[4]	.935			.9509

How to test the methods?

Comparison with real-world data: do modules reveal nodes having similar meta-data?

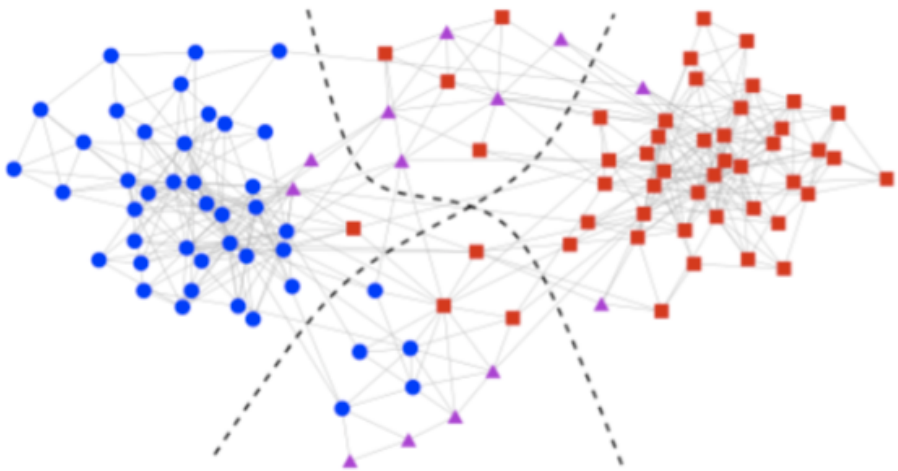
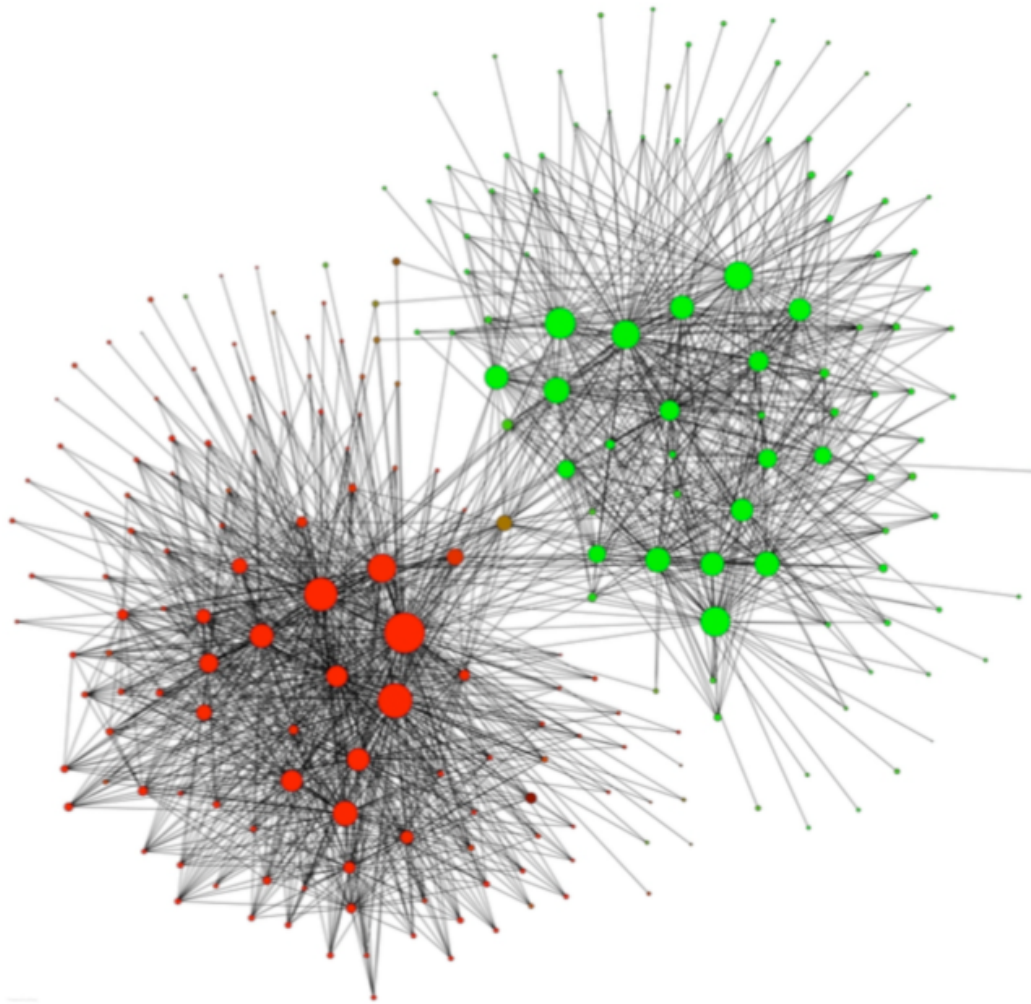


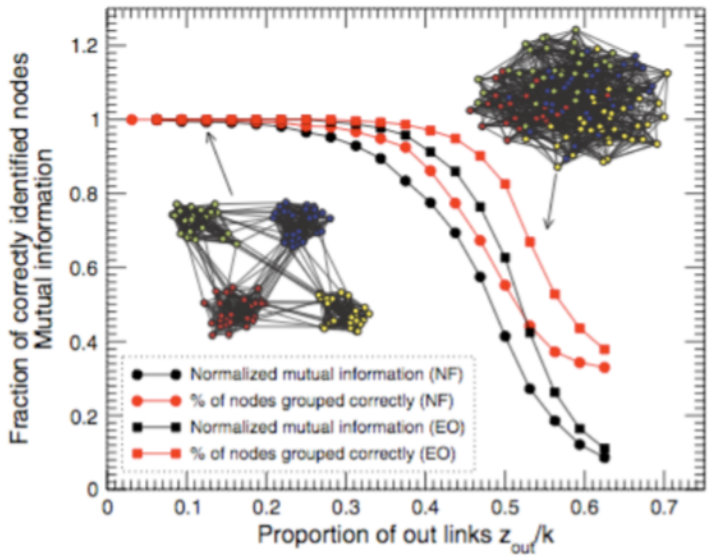
FIG. 3: Krebs' network of books on American politics. Vertices represent books and edges join books frequently purchased by the same readers. Dotted lines divide the four communities found by our algorithm and shapes represent the political alignment of the books: circles (blue) are liberal, squares (red) are conservative, triangles (purple) are centrist or unaligned.

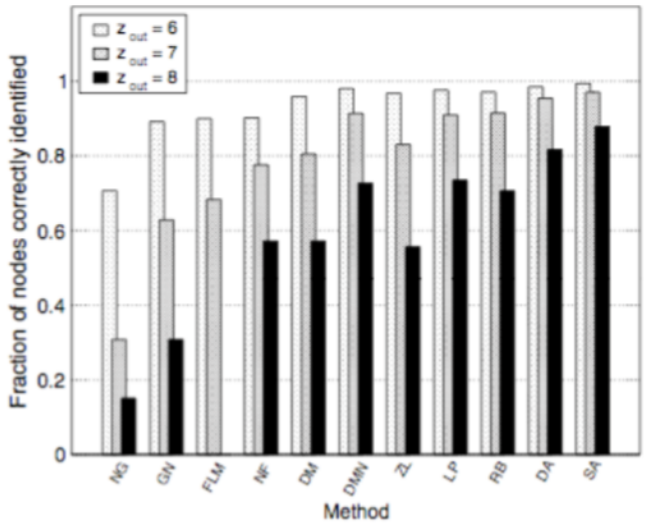


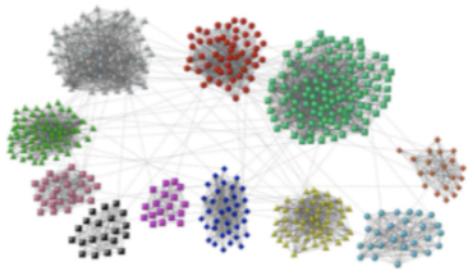
But: meta-data are often unknown. No insurance that modular organization coincides with semantic/cultural organisation

How to test the methods?

Benchmarks: artificial networks with known community structure.





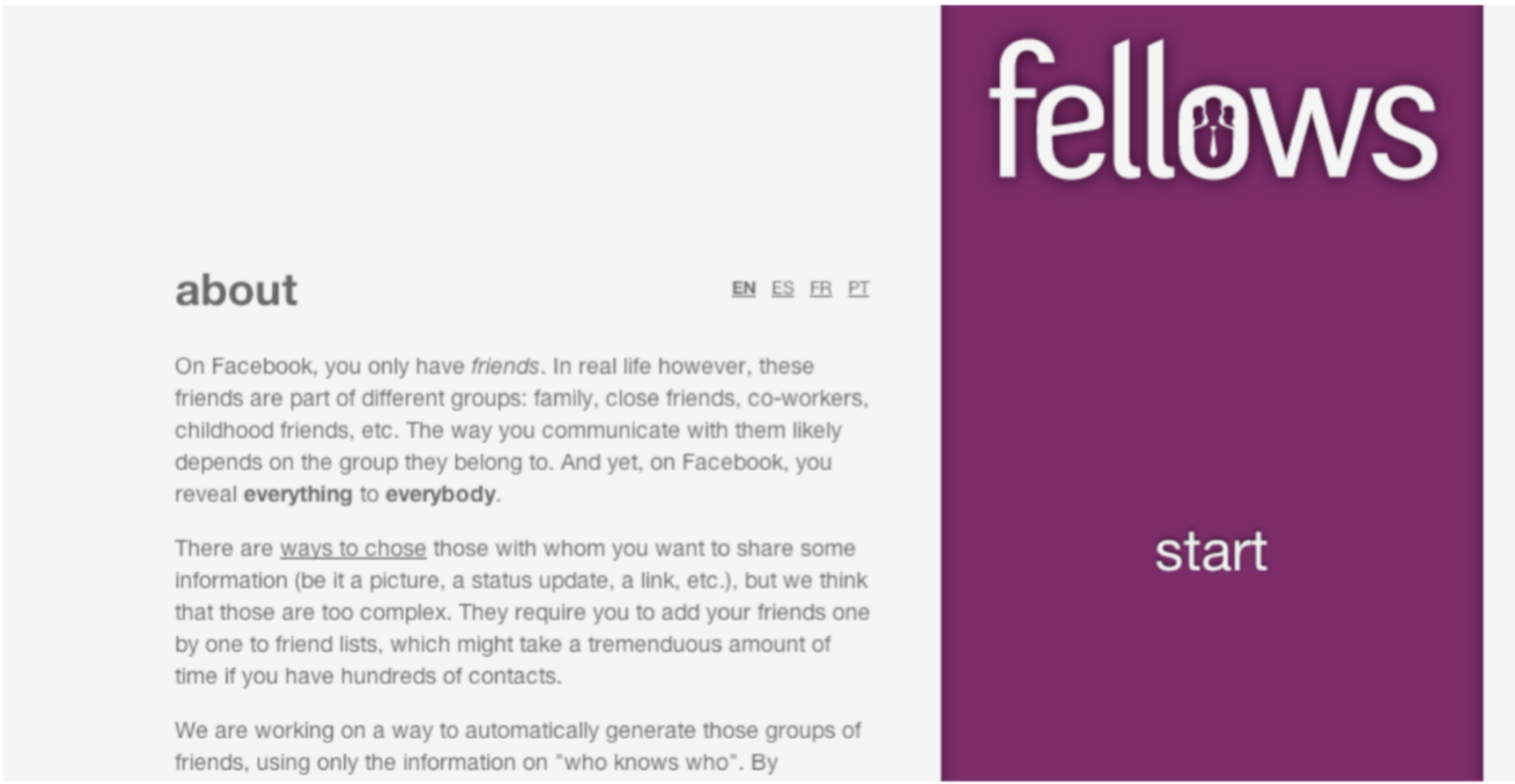


But: random networks (their structure is quite different from real-world networks).
In the way the benchmark is built, there is a (hidden) choice for what good partitions should be

Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi, Phys. Rev. E 78, 046110 (2008)

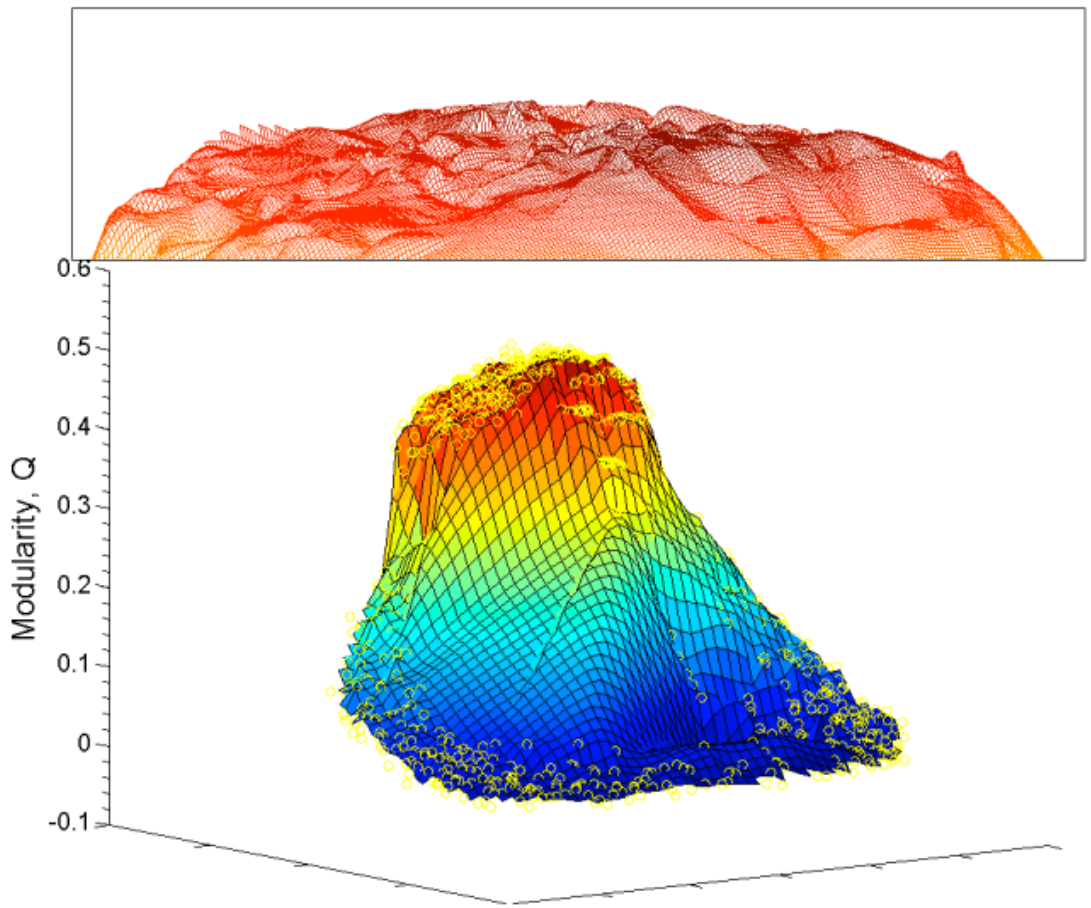
How to test the methods?

Ajk the people!



Limitations of modularity (1)

The modularity landscape tends to be very rugged, with many partitions, possibly very different, having similar value of modularity.



The performance of modularity maximization in practical contexts, Benjamin H. Good et al.
http://arxiv.org/pdf/0910.0165.pdf

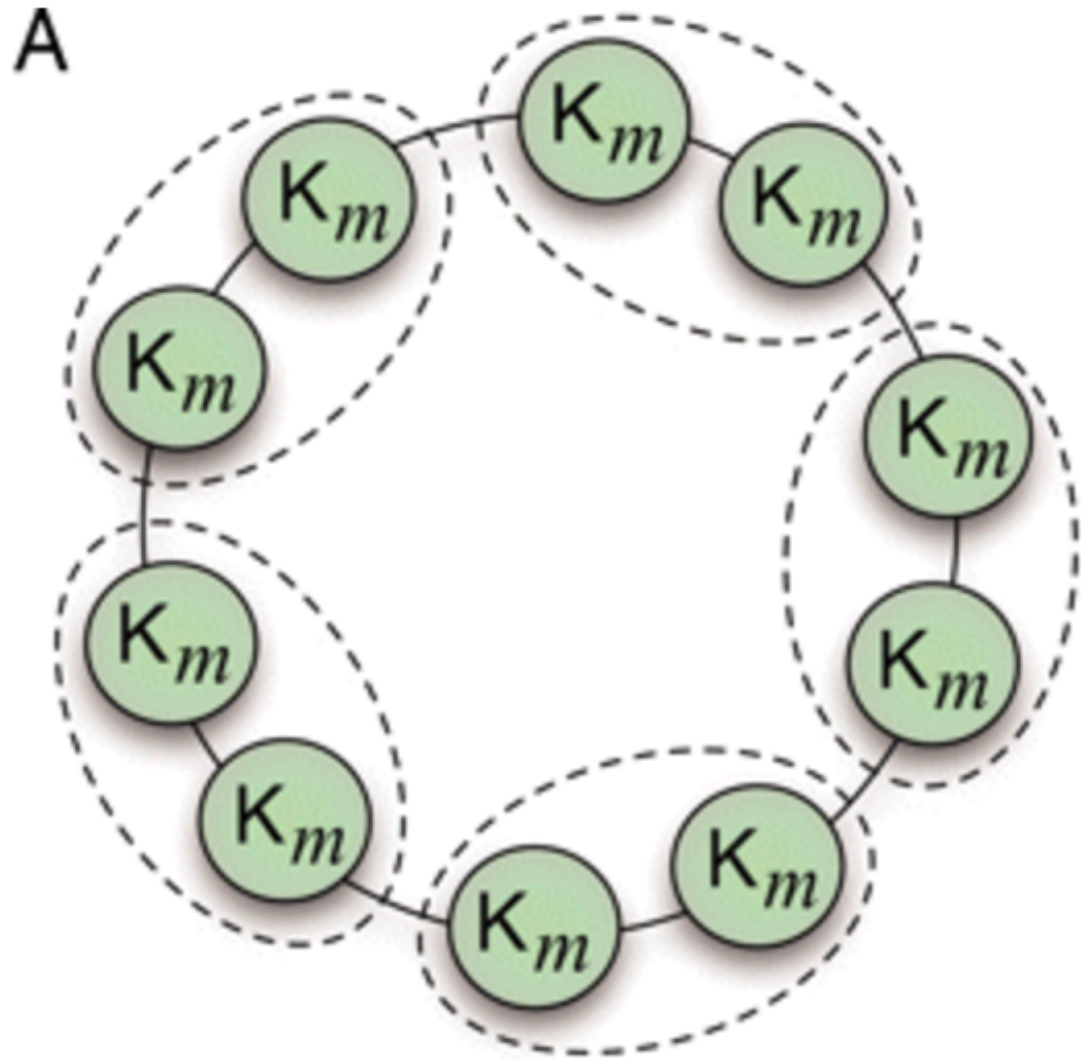
Limitations of modularity (2)

Second, Q exhibits a resolution limit, because using Q it is impossible to detect dense clusters of nodes that are smaller than a certain scale [Fortunato and Barthélemy (2007)]. The resolution limit originates from the dependency of the null model on $2M$. The dependency decreases when the number of links, M, is increased. Then, modularity maximisation tends to favour larger communities. In the limit $M \to \infty$, the null model is neglected and modularity optimisation simply uncovers the connected components. Modularity-based methods implicitly favour communities having a certain size, depending on the size of the entire network, not only on its internal structure.

$$Q_{ij} = A_{ij} - \frac{k_i k_j}{2m}$$

Resolution limit in community detection Santo Fortunato and Marc Barthélemy
http://www.pnas.org/content/104/1/36.full

Limitations of modularity (2)



Resolution limit in community detection Santo Fortunato and Marc Barthélemy
http://www.pnas.org/content/104/1/36.full

Limitations of modularity (3)

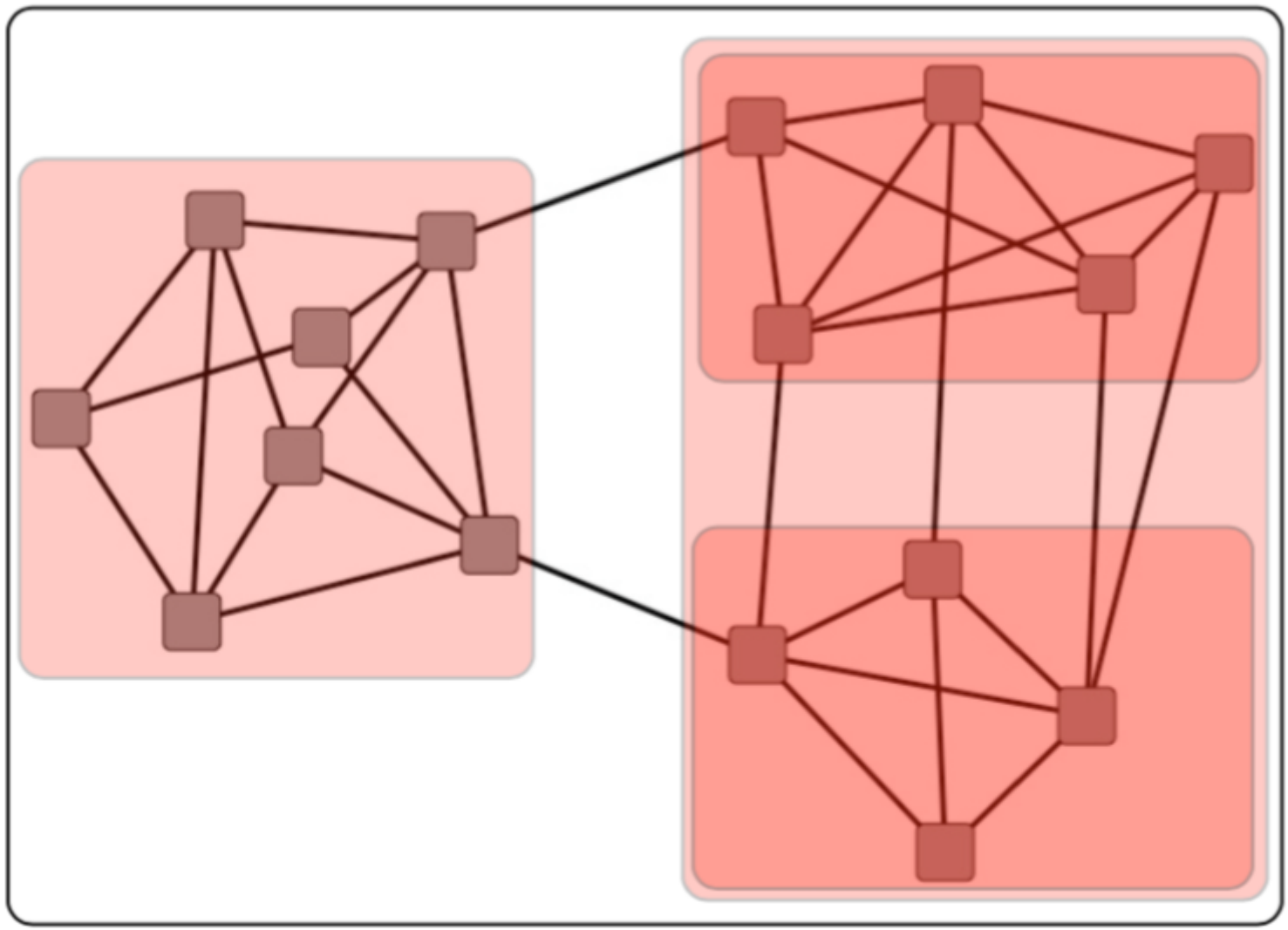
Finally, although modularity allows us to compare partitions of the same network, it is by no means intended to compare modularity values of different networks. Therefore, Q should not be used as a measure of the modularity of a network. For instance, the modularity of the best partition of a random network tends to $Q = 1$ when the network is sufficiently large, whereas this network is by no means modular [Guimerà *et al.* (2004)].

Resolution limit in community detection Santo Fortunato and Marc Barthélemy
http://www.pnas.org/content/104/1/36.full

Multi-level modularity

Resolution limit

What about sub (or hyper)-communities in a hierarchical network?



Multi-level modularity

Add a resolution parameter!

Reichardt & Bornholdt

$$Q_\gamma = \frac{1}{2m} \sum_{i,j} \Big[A_{ij} - \gamma P_{ij} \Big] \delta(c_i, c_j)$$

Arenas et al.

$$Q(A_{ij} + r I_{ij})$$

Tuning parameters allow to uncover communities of different sizes
Reichardt & Bornholdt different of Arenas, except in the case of a regular graph where

$$\gamma = 1 + r / \langle k \rangle$$

J. Reichardt and S. Bornholdt, Phys. Rev. E 74, 016110 (2006). Statistical mechanics of community detection
A Arenas, A Fernandez, S Gomez, New J. Phys. 10, 053039 (2008). Analysis of the structure of complex networks at different resolution levels

Multi-level modularity

Add a resolution parameter!

Reichardt & Bornholdt

$$Q_\gamma = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \gamma P_{ij} \right] \delta(c_i, c_j)$$

Corrected Arenas

$$Q(A_{ij} + r \frac{k_i}{\langle k \rangle} \delta_{ij})$$

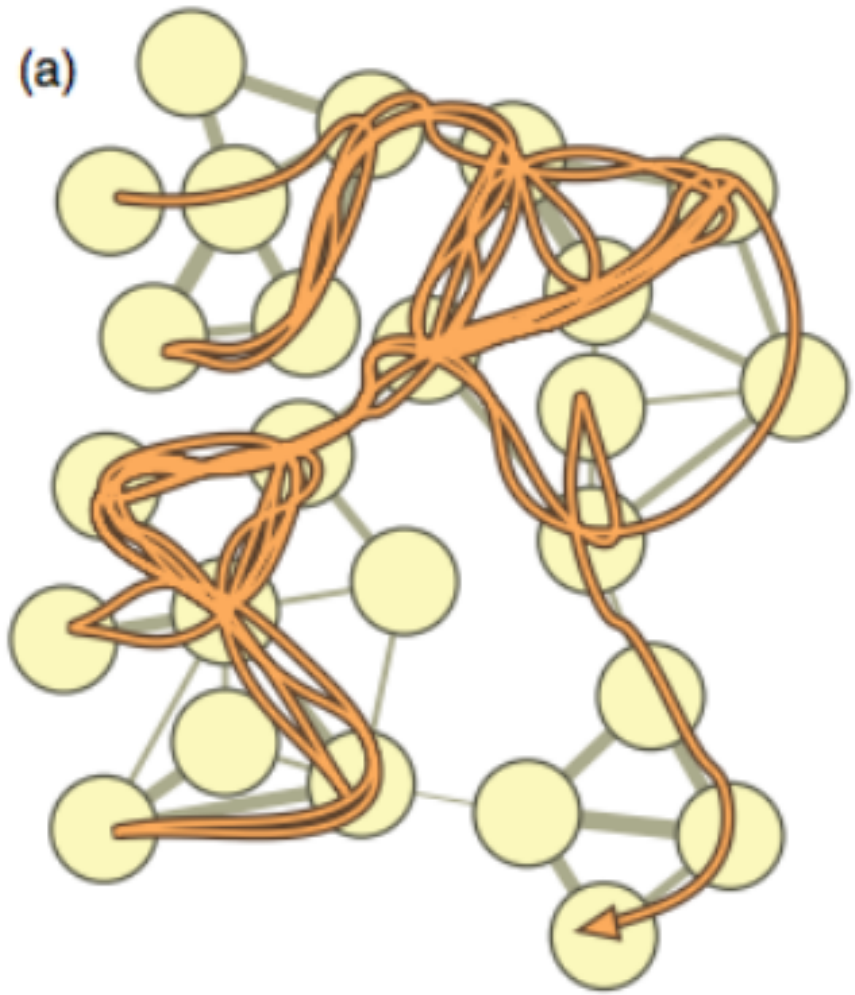
Preserves the eigenvectors of Laplacian (no A) and has a nice dynamical interpretation

Reichardt & Bornholdt = corrected Arenas for any graph

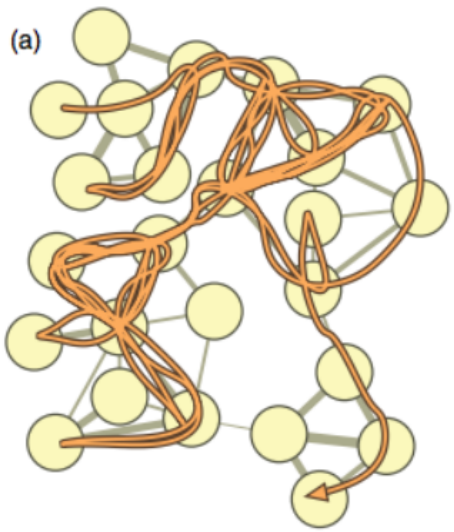
$$\gamma = 1 + r / \langle k \rangle$$

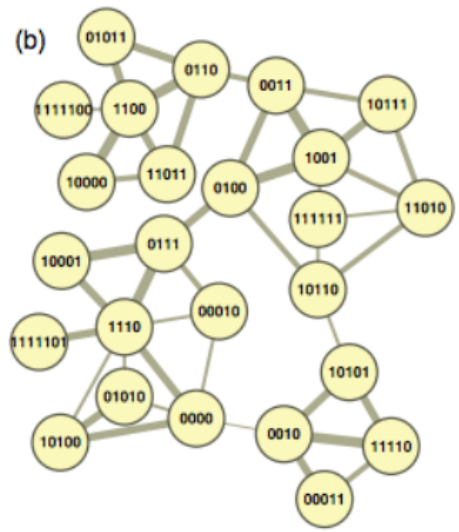
R. Lambiotte, Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on, 546-553 (2010)

Dynamics as way to uncover communities

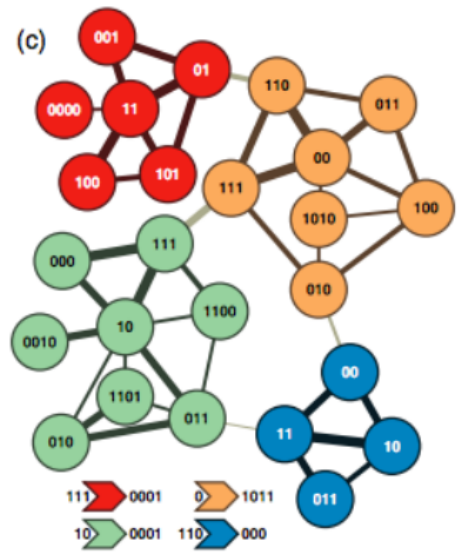


Dynamics as way to uncover communities

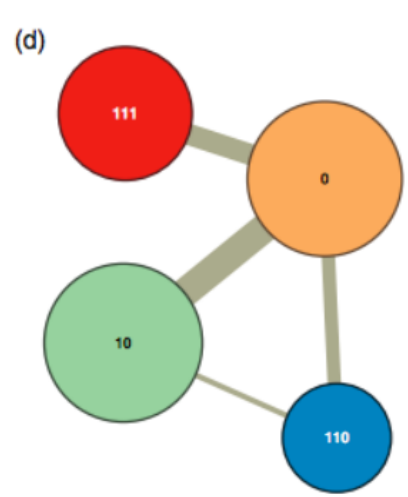




1111100 1100 0110 11011 10000 11011 0110 0011 10111 1001
0011 1001 0100 0111 10001 1110 0111 10001 0111 1110 0000
1110 10001 0111 1110 0111 1110 1111101 1110 0000 10100 0000
1110 10001 0111 0100 10110 11010 10111 1001 0100 1001 10111
1001 0100 1001 0100 0011 0100 0011 0110 11011 0110 0011 0100
1001 10111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



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



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

The Map Equation: coding trajectories

Imagine a random walk on a given network. If the network has community structure, the random walker would wander within a community for a long time before crossing a bridge to a different community. A straightforward way to describe the trajectory of the random walk is to write down the visited nodes in an ordered list, e.g., v_1, v_4, v_1, v_7, v_3, ... The amount of information required to express the trajectory is estimated as follows. We code each node into a finite binary sequence, i.e., a code word, and concatenate the code words. For example, if v_1, v_3, v_4, and v_7 are coded into 000, 010, 011 and 110, the aforementioned trajectory is coded into 000011000110010$\cdots$. For unique decoding, the code has to be prefix-free. In other words, a code word must not be a prefix (i.e., initial segment) of another code word. For example, if v_1 and v_2 are coded into 000 and 0001, respectively, the code is not prefix-free because 000 is an initial segment of 0001.

The Map Equation

The Huffman code is a prefix-free code that encodes symbols separately and generally yields short binary sequences to represent trajectories of the random walk. It assigns a short code word to a frequently visited node and vice versa. The mean code word length per step of the random walk is given by $\sum_{i=1}^{N} p_i^* L(i)$, where p_i^* is the stationary density of the random walk at node v_i and $L(i)$ is the length of the code word for node v_i.

When the symbols (v_i in our case) appear independently, the Huffman code often yields a code length that is close to the theoretical lower bound obtained by the Shannon entropy, which is

$$H = -\sum_{i=1}^{N} p_i^* \log p_i^* \tag{3.85}$$

per step. However, the sequence of nodes is correlated in time because it is produced by the random walk. Then, an alternative coding scheme may lessen the mean code length. In particular, we can design a two-layered variant of the Huffman code to exploit the community structure of the network. Because there are less nodes in a community CM_i as compared to the entire network, we can express a trajectory within CM_i with a shorter, different Huffman code, which is local to CM_i. Based on this observation, we rebuild the Huffman code as follows.

The Map Equation

(1) When the walker enters community CM_i, a code word to represent this entry event is issued.

(2) The walker wanders within CM_i. The trajectory of the walker during this period is encoded by concatenating the code words corresponding to the sequence of the visited nodes. The sequence of these code words is simply placed after the code word produced in the previous step (i.e., entry to CM_i). It should be noted that the intra-community code words make sense only within CM_i. A different community $CM_{i'}$ $(i' \neq i)$ may use the same code word as the one used within CM_i to represent a different node in $CM_{i'}$.

(3) The walker exits CM_i. This event is represented by a special code word, which is concatenated after the sequence of code words produced so far.

(4) The exit from CM_i implies that the walker immediately enters a different community, CM_j. Therefore, a code word to notify that the walker has entered CM_j is issued. Then, the code words local to CM_j are used until the walker exits CM_j. We repeat this procedure.

The Map Equation

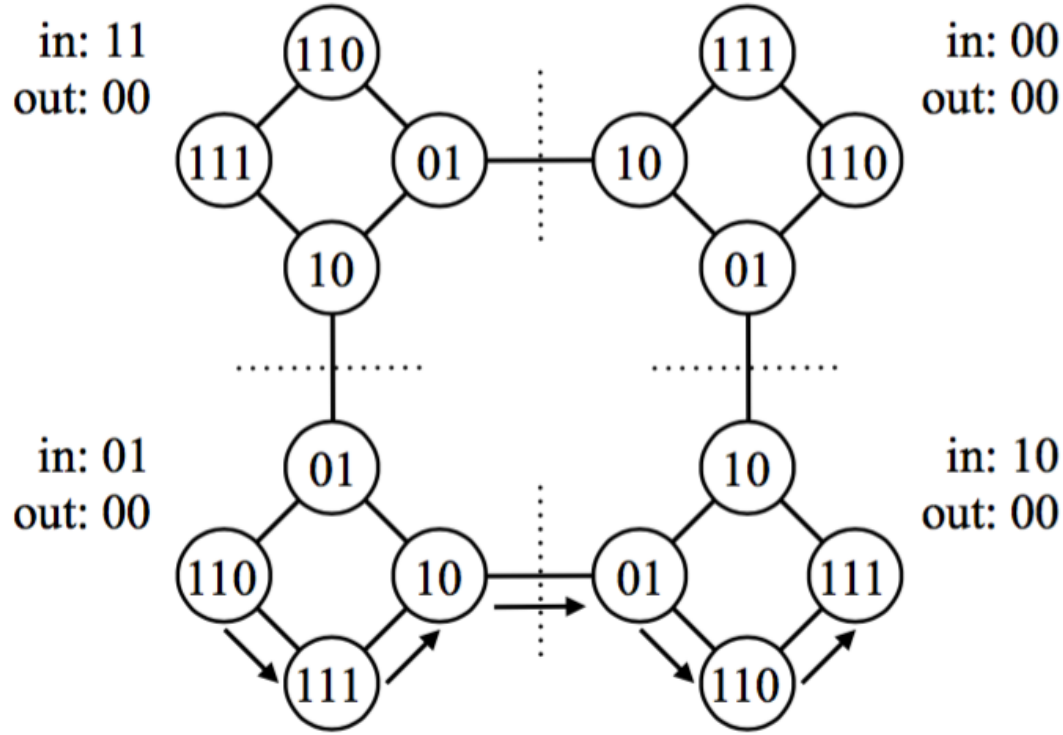


Fig. 3.6 Optimal partitioning according to Infomap and the resulting code words. This example is based on a demo applet available at Martin Rosvall's website http://www.mapequation.org/apps/MapDemo.html.

the trajectory shown by the arrows in the figure is encoded into 011101111000100111011 1. The first 01 indicates that the walk starts in the left-bottom community, and the 110 that follows indicates that the walk starts at the 110 node in this community. 0010 in the middle indicates that the walk exits this community (by the code word 00) and immediately enters the community to the right (by the code word 10).

The Map Equation

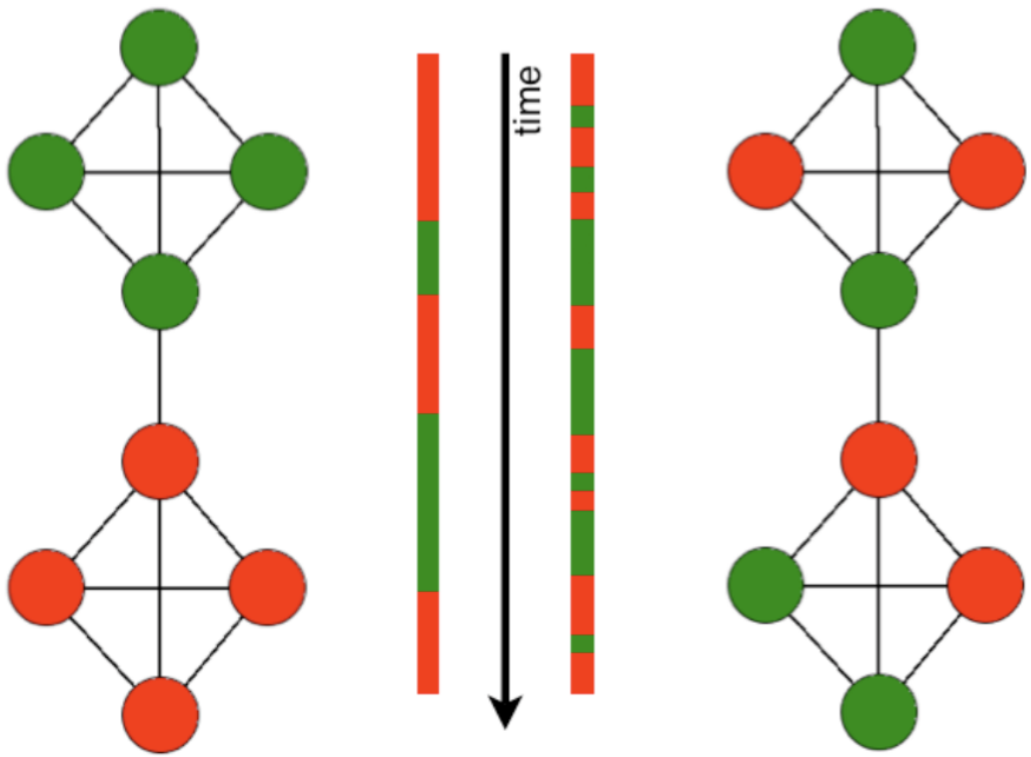
In contrast to the original Huffman code, we have to invest $2N_{\mathrm{CM}}$ code words to mark the entry to and exit from a community. However, we can save the code length when the walker wanders in a community, which occupies a majority of steps. Overall, the mean code length is expected to be smaller with the two-layer code in the presence of community structure. In order to detect communities in practice, there is no need for devising the optimal code of a given partition. Infomap instead proceeds by optimising a quality function, called the map equation, which generalises Eq. (3.85). The resulting quality function provides a theoretical limit of how concisely we can specify a walk in the network using a given partition. The optimisation is then performed by a greedy algorithm similar to the one used for maximising modularity (Section 3.10.1), with additional fine- and coarse-graining steps carried out for improving the partitioning.

$$L(\mathrm{M}) = q_{\curvearrowright} H(\mathscr{Q}) + \sum_{C} p^{C}_{\circlearrowright} H(\mathscr{C}^{C})$$

Minimizing the Map Equation provides the partition giving the best (most efficient) coding scheme

Markov stability

The quality of a partition is determined by the patterns of a flow within the network: a flow should be trapped for long time periods within a community before escaping it.
The stability of a partition is defined by the statistical properties of a random walker moving on the graph



J.-C. Delvenne, S. Yaliraki & M. Barahona, Stability of graph communities across time scales. arXiv:0812.1811.

Markov stability

The quality of a partition is determined by the patterns of a flow within the network: a flow should be trapped for long time periods within a community before escaping it.
The stability of a partition is defined by the statistical properties of a random walker moving on the graph

$$R(t) = \sum_{C \in \mathcal{P}} P(C, t_0, t_0 + t) - P(C, t_0, \infty)$$

$P(C, t_0, t_0 + t)$

probability for a walker to be in the same community at times t0 and t0 +t when the system is at equilibrium

$P(C, t_0, \infty)$

probability for two independent walkers to be in C (ergodicity)

J.-C. Delvenne, S. Yaliraki & M. Barahona, Stability of graph communities across time scales. arXiv:0812.1811.

Markov stability versus Modularity

Let us consider a random walk on an undirected network:

$$p_{i;n+1} = \sum_j \frac{A_{ij}}{k_j} p_{j;n} \quad \xrightarrow{\text{equilibrium}} \quad p_i^* = k_i/2m$$

$$R(1) = \sum_{i,j} \left[\frac{A_{ij}}{k_j} \frac{k_j}{2m} - \frac{k_i k_j}{(2m)^2} \right] \delta(c_i, c_j)$$

Probability that a walker is in the same community initially and at time t=1

Same probability for independent walkers

$$R(1) = Q = \frac{1}{2m} \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

Markov stability versus Modularity

Let us consider a random walk on an **directed** network:

$$p_{i;n+1} = \sum_j \frac{A_{ij}}{k_j^{out}} p_{j;n} \quad \xrightarrow[\text{equilibrium}]{} \quad p_i^* = \pi_i$$

$$R(1) = \sum_{i,j} \left[\frac{A_{ij}}{k_j^{out}} \pi_j - \pi_i \pi_j \right] \delta(c_i, c_j) \neq Q$$

Counting versus flows

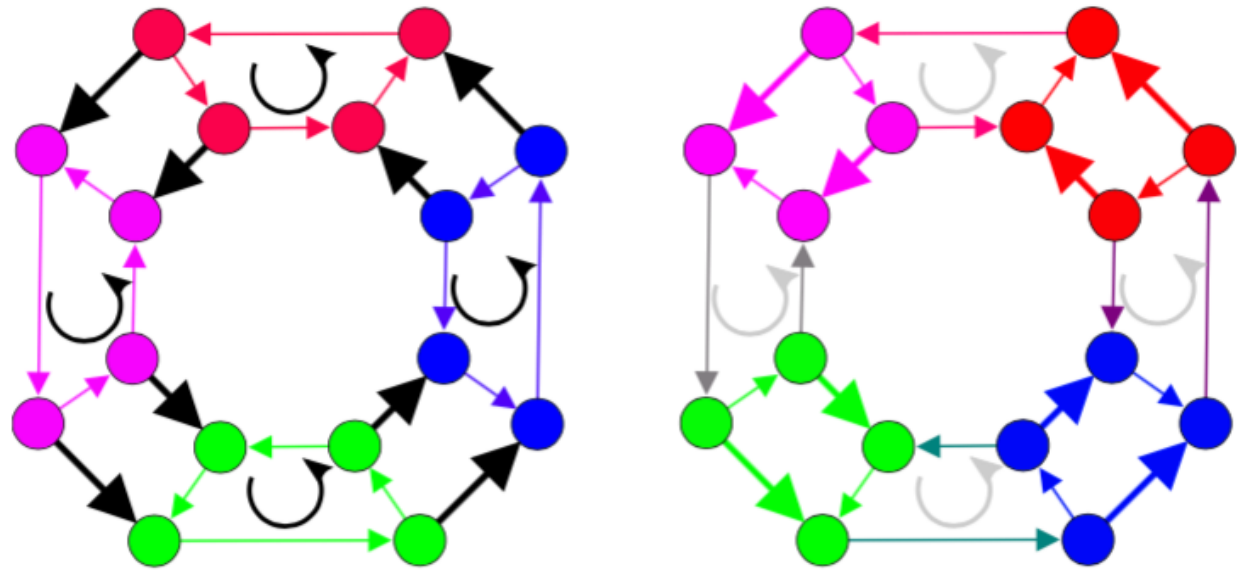
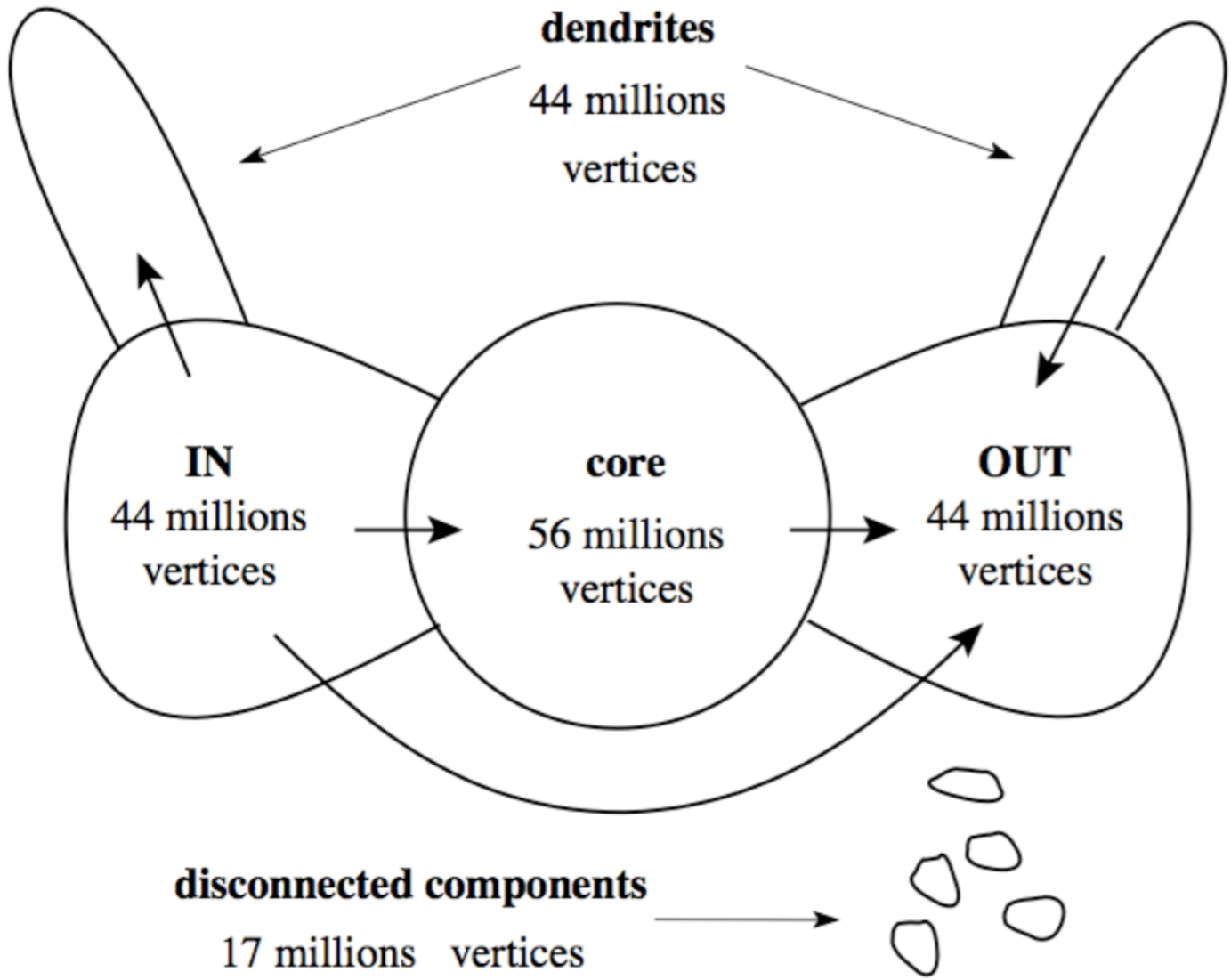


Fig. 4. **Directed Markov Stability versus extensions of modularity.** In this toy network [16], the weight of the bold links is twice the weight of the other links. The partition on the left (indicated by different colors) optimizes directed Markov Stability (34), which intrinsically contains the pagerank as a null model. The partition on the right instead optimizes an extension of modularity based on in- and out-degrees [64], [65]. Hence directed Markov Stability produces flow communities, whereas the extension of modularity ignores the effect of flows.

Counting versus flows



Markov stability versus Modularity

Let us consider a random walk on an **directed** network:

$$p_{i;n+1} = \sum_j \frac{A_{ij}}{k_j^{out}} p_{j;n} \quad \xrightarrow{\text{equilibrium}} \quad p_i^* = \pi_i$$

$$R(1) = \sum_{i,j} \left[\frac{A_{ij}}{k_j^{out}} \pi_j - \pi_i \pi_j \right] \delta(c_i, c_j) \neq Q$$

$$R(1) \neq Q(A) \qquad \text{but} \qquad R(1) = Q(Y)$$

$$Y = \frac{X + X^T}{2} \qquad X_{ij} = \frac{A_{ij}}{k_j^{out}} \pi_j$$

Time as a resolution parameter

Let us consider a continuous-time random walk with Poisson waiting times

$$\dot{p}_i = \sum_j \frac{A_{ij}}{k_j} p_j - p_i \quad \longrightarrow \quad p_i^* = k_i/2m$$

equilibrium

$$R(t) = \sum_{i,j} \left[\left(e^{t(B-I)} \right)_{ij} \frac{k_j}{2m} - \frac{k_i k_j}{(2m)^2} \right] \delta(c_i, c_j)$$

$$B_{ij} = A_{ij}/k_j$$

Probability that a walker is in the same community initially and at time t

Same probability for independent walkers

Time as a resolution parameter

Let us consider a continuous-time random walk with Poisson waiting times

$$R(0) = 1 - \sum_{i,j} \frac{k_i k_j}{(2m)^2} \delta(c_i, c_j)$$ Communities = Single nodes

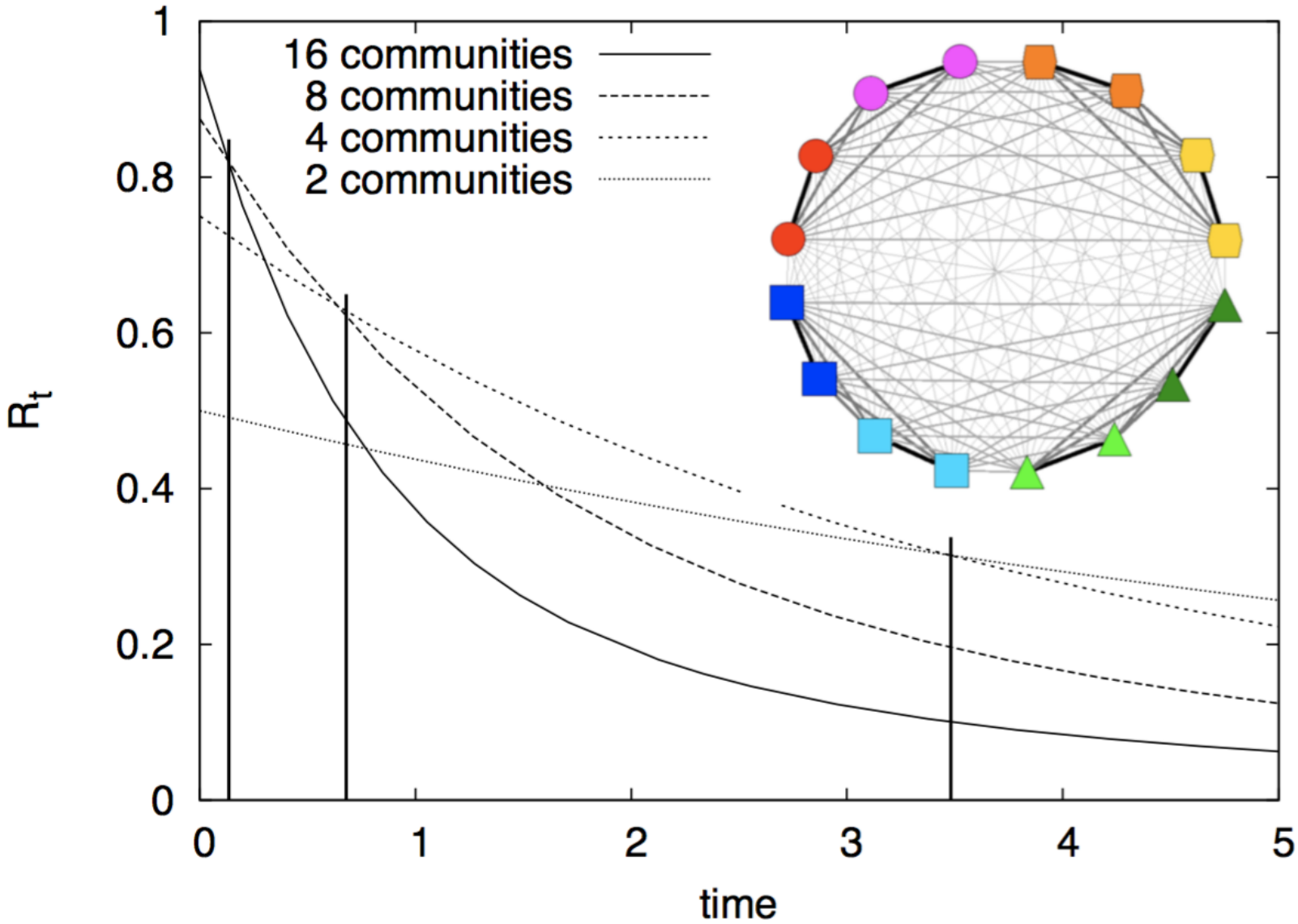
$$R(t) \approx (1-t)R(0) + tQ_C \equiv Q(t)$$ Tuneable modularity of Reichart and Bornholdt

time

Asymptotically, two-way partition given by the Fiedler vector

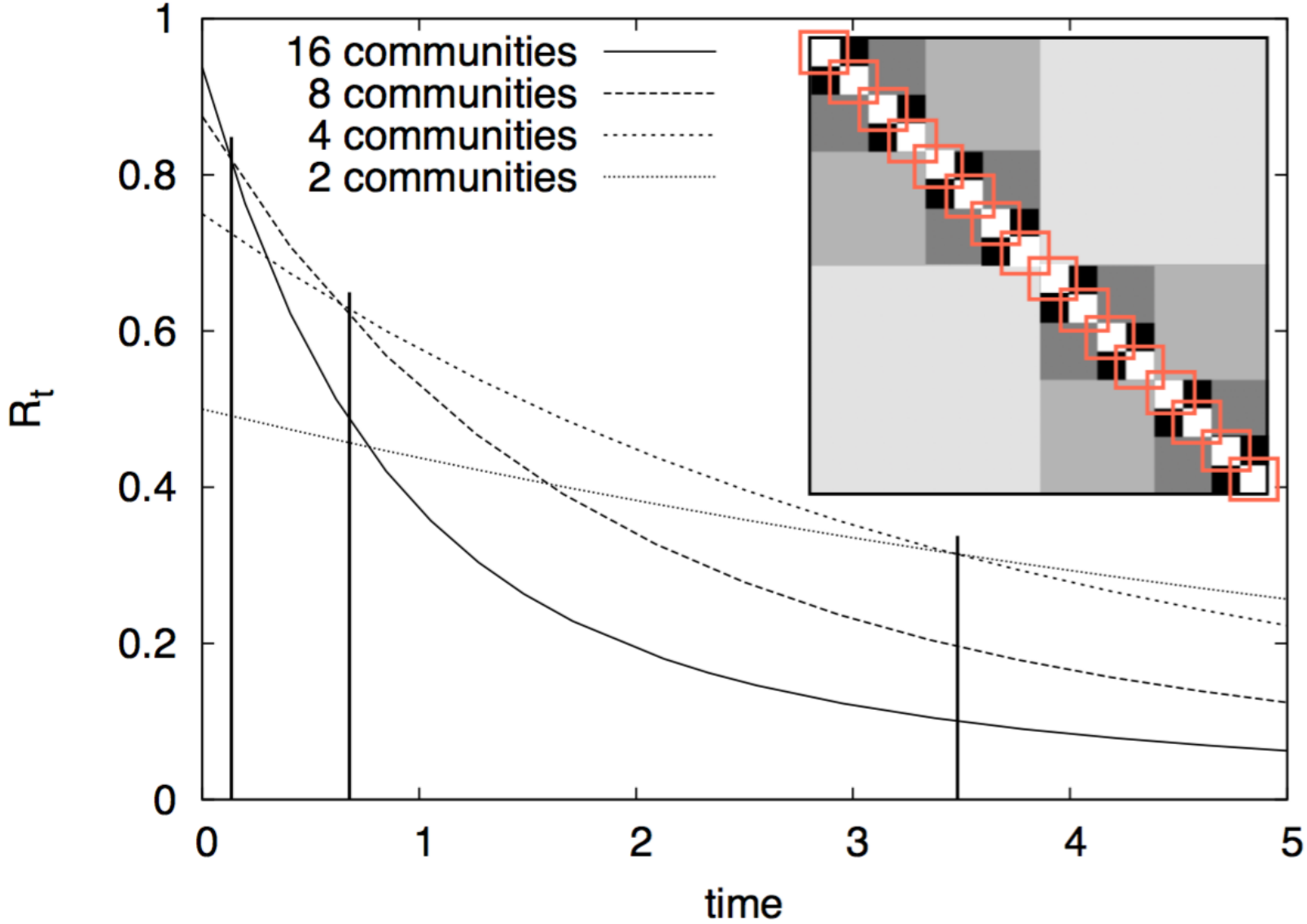
Time as a resolution parameter

Time is a “resolution parameter”: larger and larger communities when time is increased



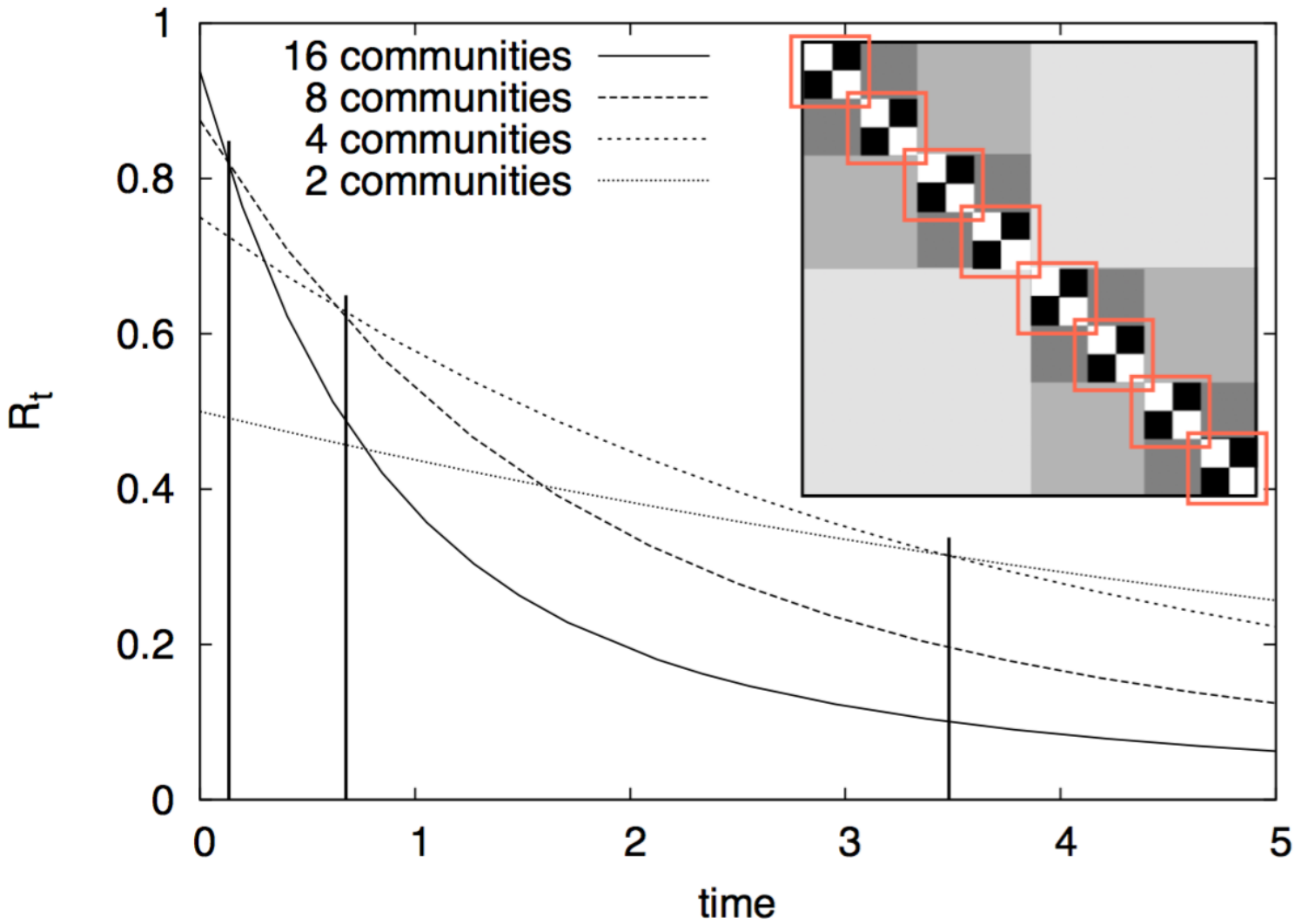
Time as a resolution parameter

Time is a “resolution parameter”: larger and larger communities when time is increased



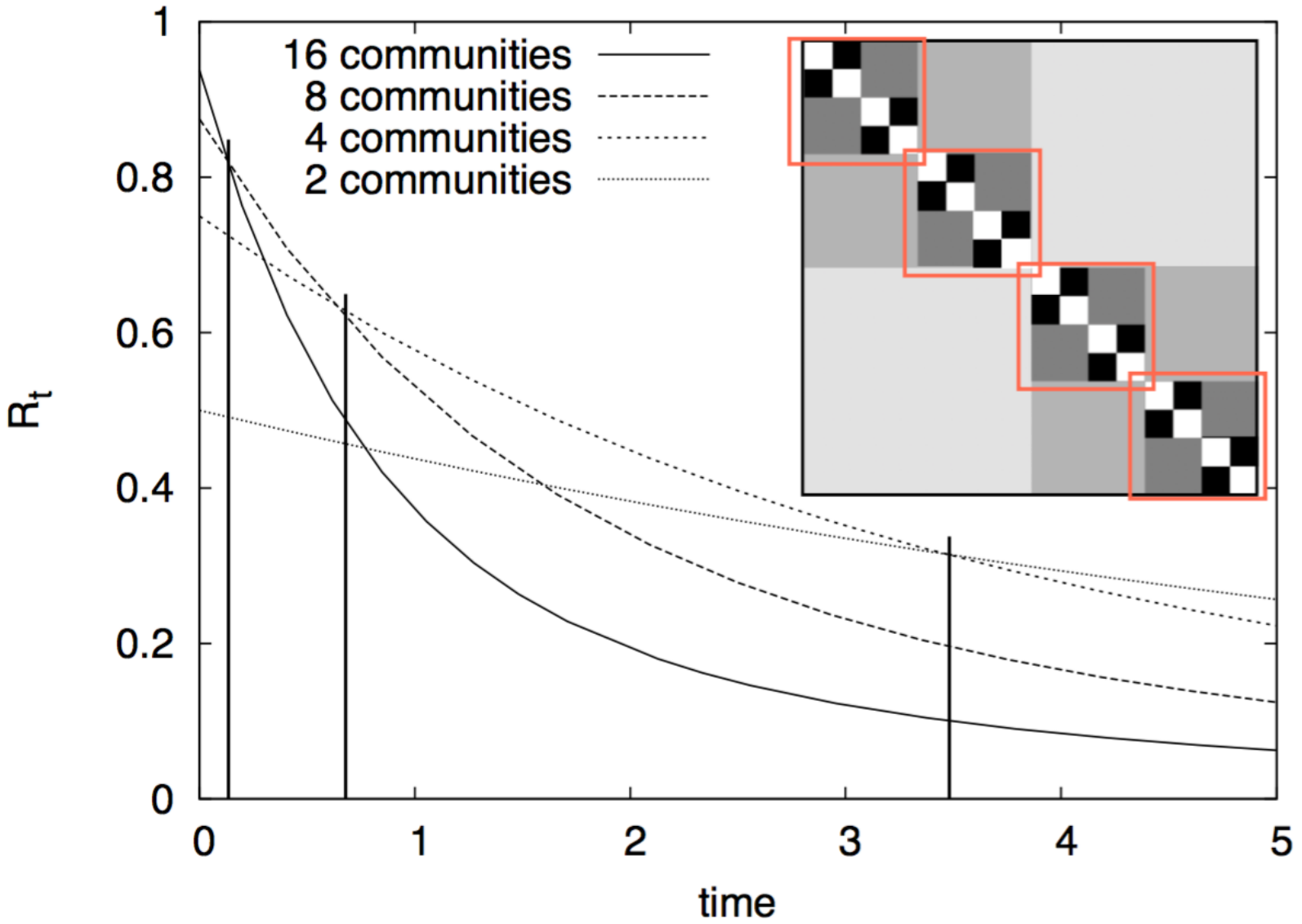
Time as a resolution parameter

Time is a “resolution parameter”: larger and larger communities when time is increased



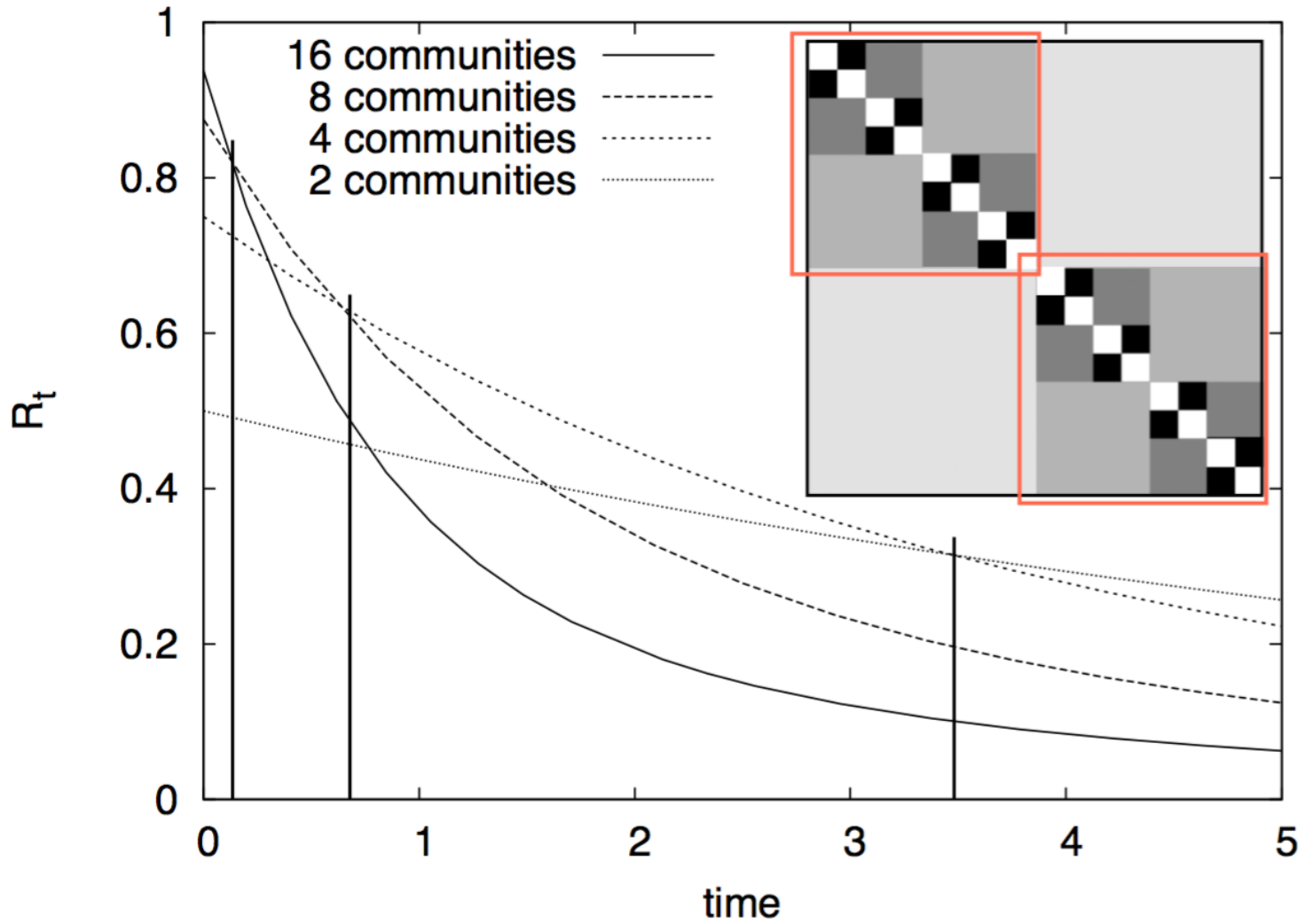
Time as a resolution parameter

Time is a “resolution parameter”: larger and larger communities when time is increased



Time as a resolution parameter

Time is a “resolution parameter”: larger and larger communities when time is increased



In practice: optimization?

The stability R(t) of the partition of a graph with adjacency matrix A is equivalent to the modularity Q of a time-dependent graph with adjacency matrix X(t)

$$X_{ij}(t) = \left(e^{t(B-I)}\right)_{ij} k_j \qquad X_{ij}(t) = X_{ji}(t)$$

which is the flux of probability between 2 nodes at equilibrium and whose generalised degree is

$$\sum_j X_{ij}(t) = k_i$$

$$R(t) = \sum_{i,j} X_{ij}(t)/2m - k_i k_j/(2m)^2 \ \delta(c_i, c_j) = Q(X(t))$$

For very large networks: $R(t) \approx (1-t)R(0) + tQ_C \equiv Q(t)$

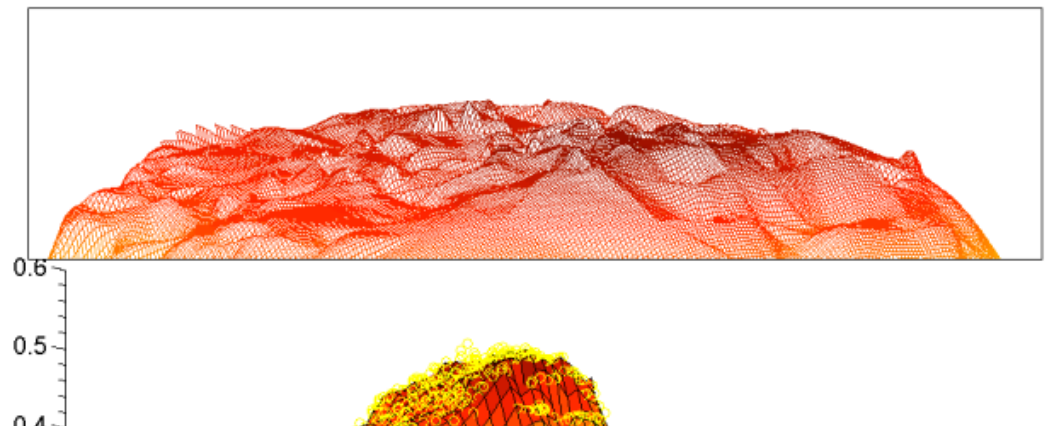
In practice: selection of the significant scales?

The optimization of R(t) over a period of time leads to a sequence of partitions that are optimal at different time scales.

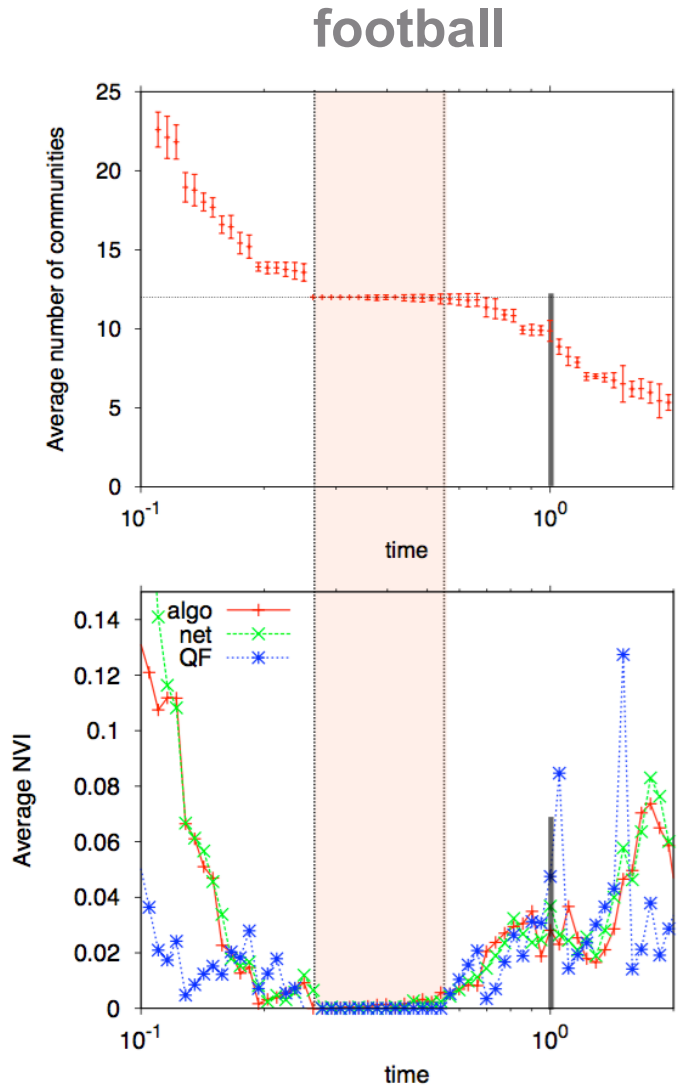
How to select the most relevant scales of description?
The significance of a particular scale is usually associated to a certain notion of robustness of the optimal partition. Here, robustness indicates that a small modification of the optimization algorithm, of the network, or of the quality function does not alter this partition.

We look for regions of time where the optimal partitions are very similar. The similarity between two partition is measured by the normalised variation of information.

Intuition: at a bad scale, several competing maxima make the lanscape more rugged, leading to a sensitivity in the outcome of the algorithm



In practice: selection of the significant scales?



algo: for each t, 100 optimizations of Louvain algorithm while changing the ordering of the nodes

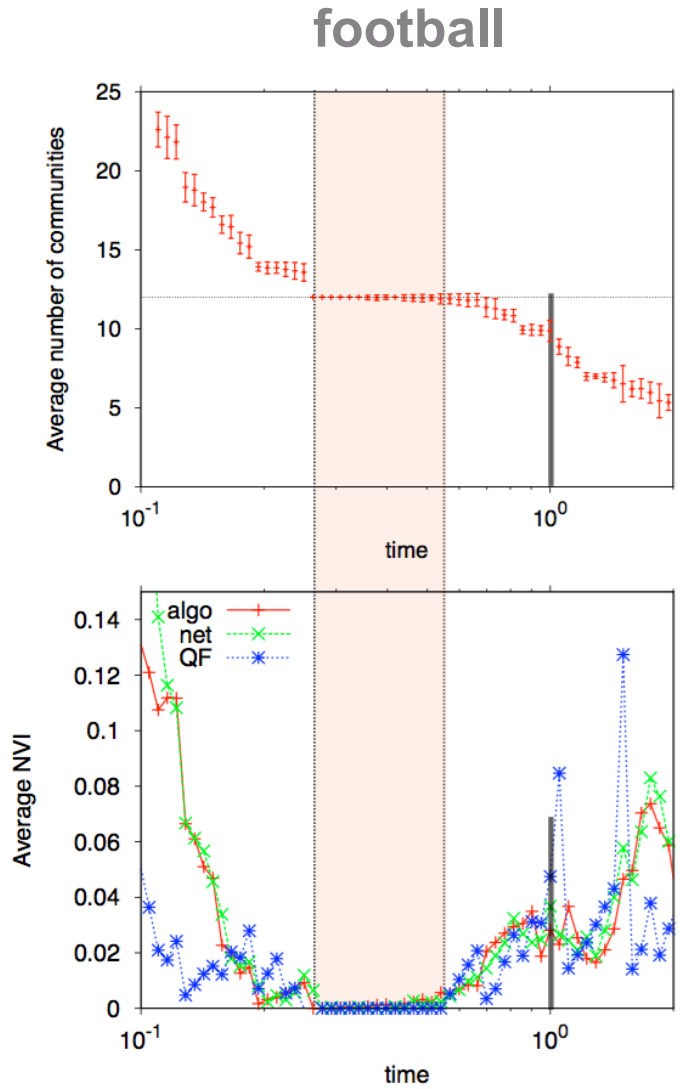
$$\langle V \rangle_{\text{algo}}(t) = \frac{2}{T(T-1)} \sum_{i=1}^{T} \sum_{i'=i+1}^{T} \hat{V}(\mathcal{P}_i(t), \mathcal{P}_{i'}(t))$$

net: for each t, 100 optimizations with a fixed algorithm but randomized modifications of the network

QF: for each t, one optimization. Partitions at 5 successive values of t are compared.

Compatible notions of robustness:
Lack of robustness -> high degeneracy in the landscape: uncovered partitions are not to be trusted; wrong resolution

In practice: selection of the significant scales?



algo: for each t, 100 optimizations of Louvain algorithm while changing the ordering of the nodes

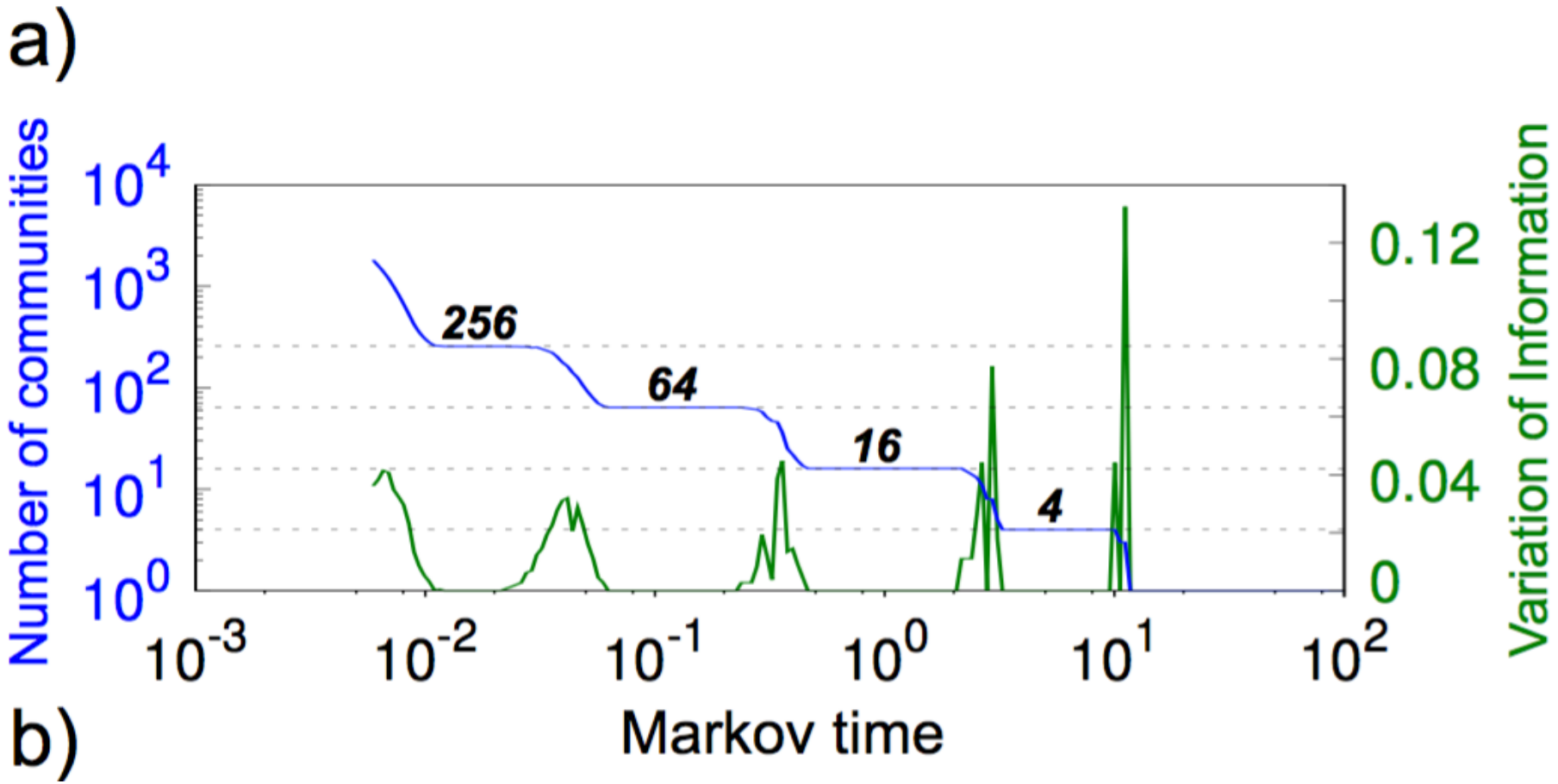
$$\langle V \rangle_{\text{algo}}(t) = \frac{2}{T(T-1)} \sum_{i=1}^{T} \sum_{i'=i+1}^{T} \hat{V}(\mathcal{P}_i(t), \mathcal{P}_{i'}(t))$$

net: for each t, 100 optimizations with a fixed algorithm but randomized modifications of the network

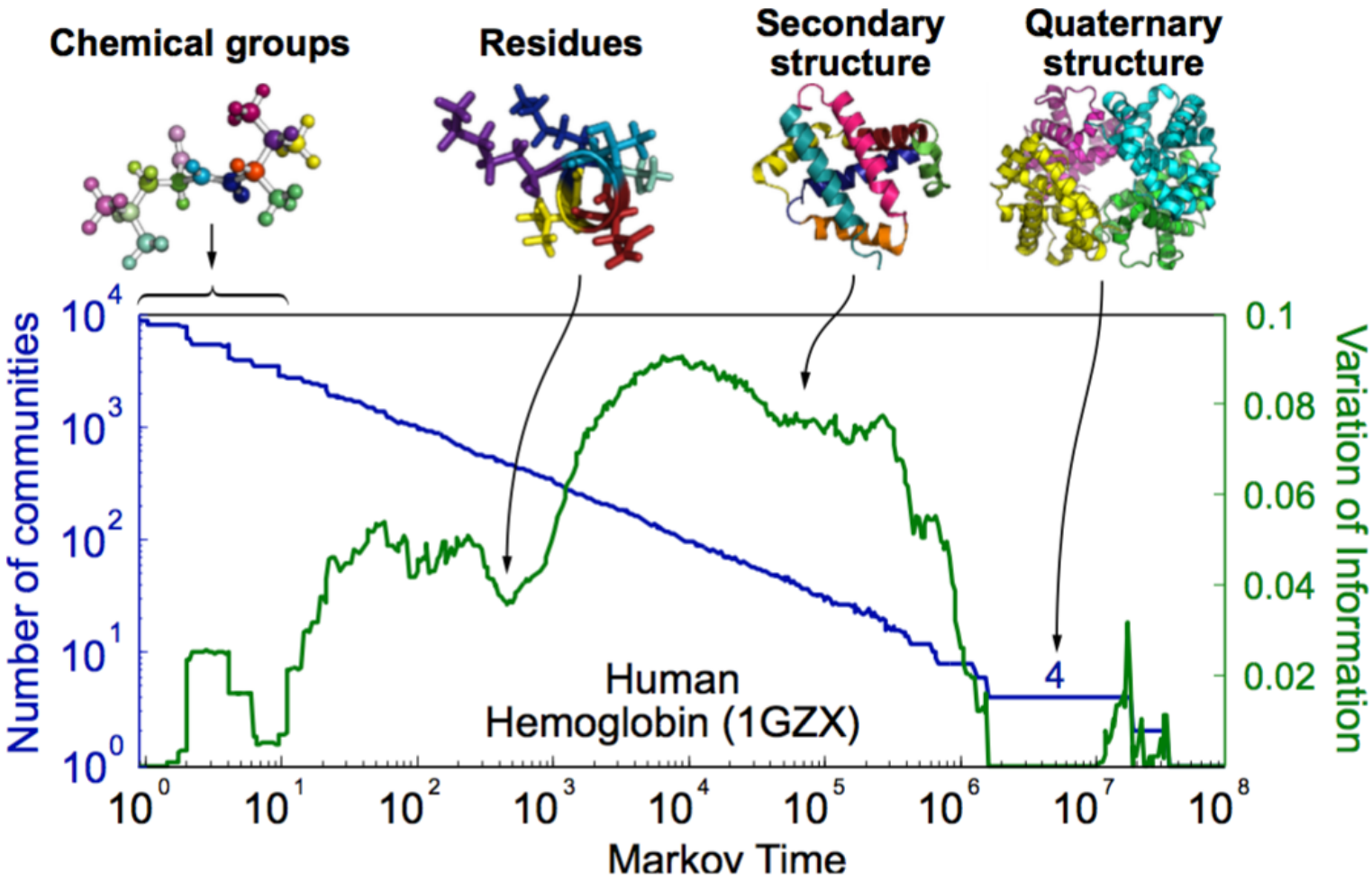
QF: for each t, one optimization. Partitions at 5 successive values of t are compared.

Compatible notions of robustness:
Lack of robustness -> high degeneracy in the landscape: uncovered partitions are not to be trusted; wrong resolution

Time as resolution parameter



Time as resolution parameter



Time as resolution parameter

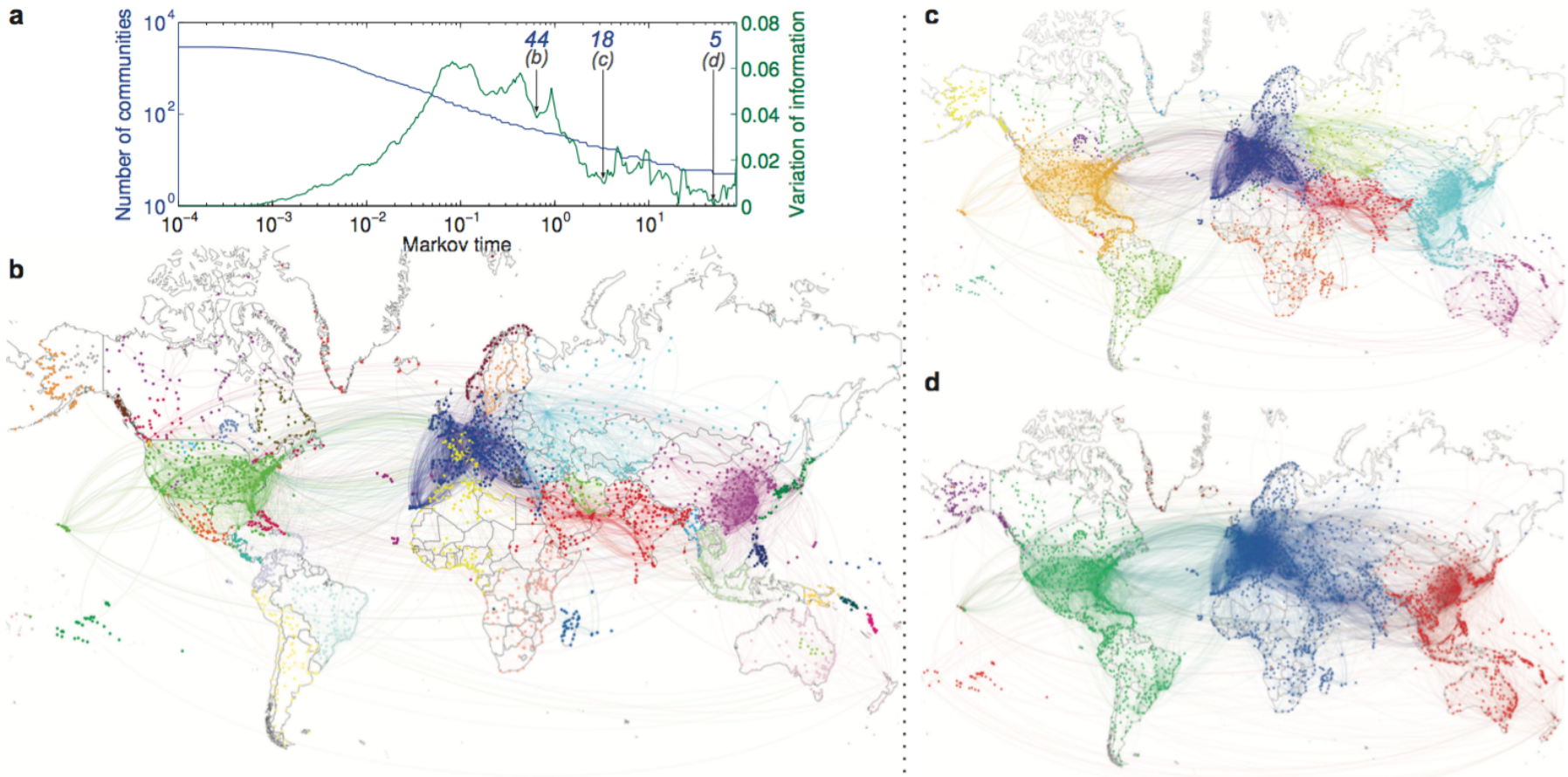


Fig. 8. **Flow communities at multiple scales in an airport network.** The airport network [82] contains $N = 2905$ nodes (airports) and 30442 weighted directed edges. The weights record the number of flights between airports (i.e., the network does not take into account passenger numbers, just the number of connections). Representative partitions at different levels of resolution with (b) 44, (c) 18 and (d) 5 communities are presented. The partitions correspond to dips in the normalized variation of information in (a) and show persistence across time (see Suppl. Info.).

WHAT IS NEXT ?

1. Beyond assortative communities

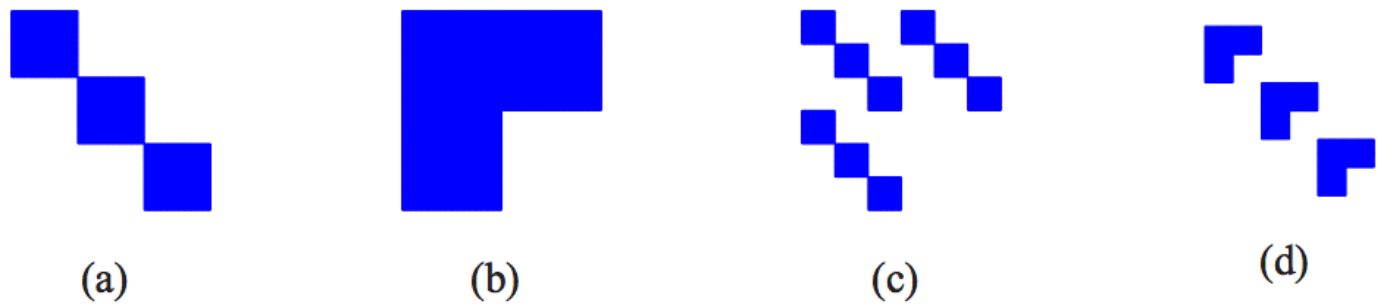


Figure 1: Examples of network block models. (a) Community structure, (b) core-periphery structure, (c) global core-periphery structure with local community structure, and (d) global community structure with local core-periphery structure. Note that (c) and (d) are equivalent.

Two nodes are structurally equivalent if they have exactly the same neighbors.
Two nodes are regularly equivalent if they share similar connections with other equivalence classes, where the notion of similar connection must be specified

Beyond assortative communities - dynamical similarity

- Consider a general linear system of the form:

$$\dot{\mathbf{x}} = \mathcal{A}\mathbf{x} + \mathcal{B}\mathbf{u},$$
$$\mathbf{y} = \mathcal{C}\mathbf{x}.$$

- Idea: group nodes that influence system in similar way
  - Look at the impulse response of each node
  - Build similarity matrix of nodes based on the similarity of impulse responses
  - Use this matrix to cluster nodes, e.g., using a Louvain like algorithm.

Multiscale dynamical embeddings of complex networks, Michael T. Schaub, et al., submitted

DYNAMICAL CLUSTERING PROBLEM — SCHEMATIC

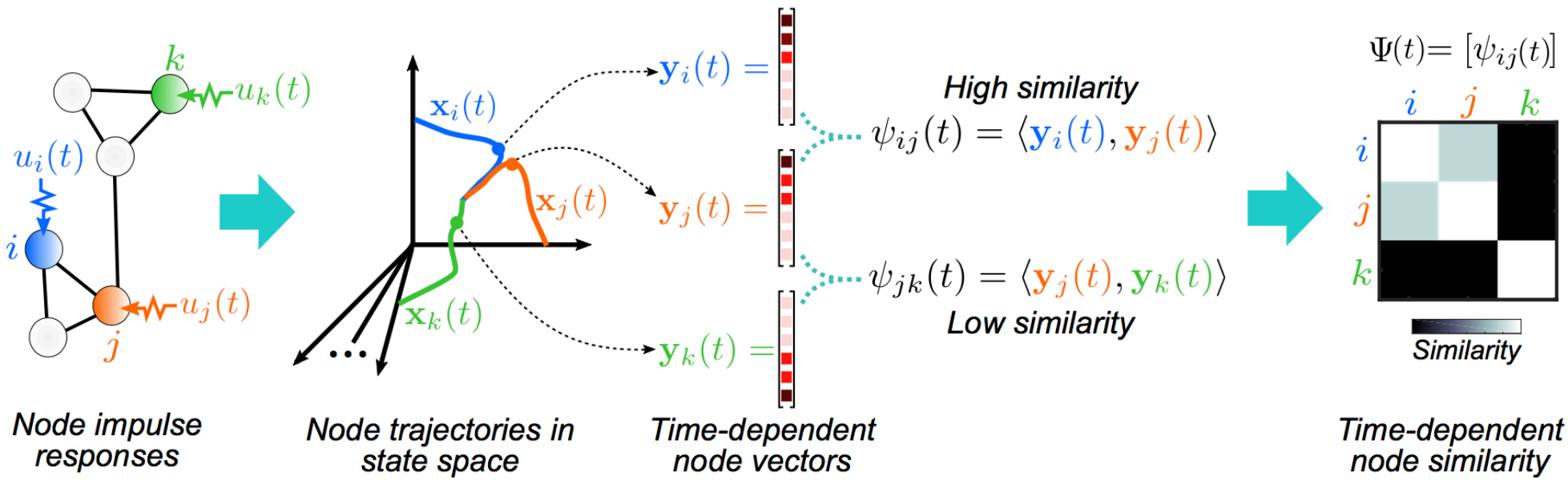
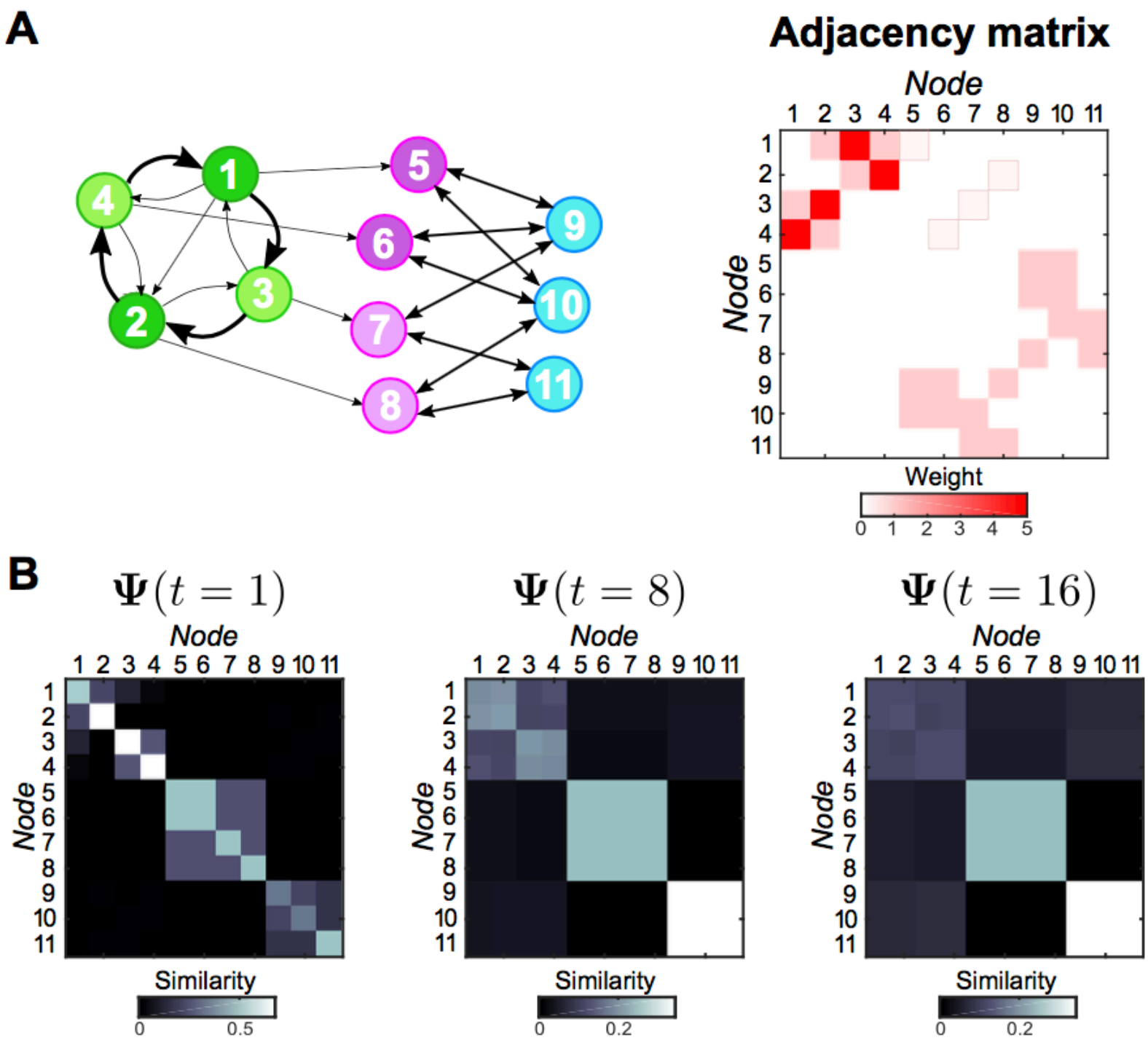
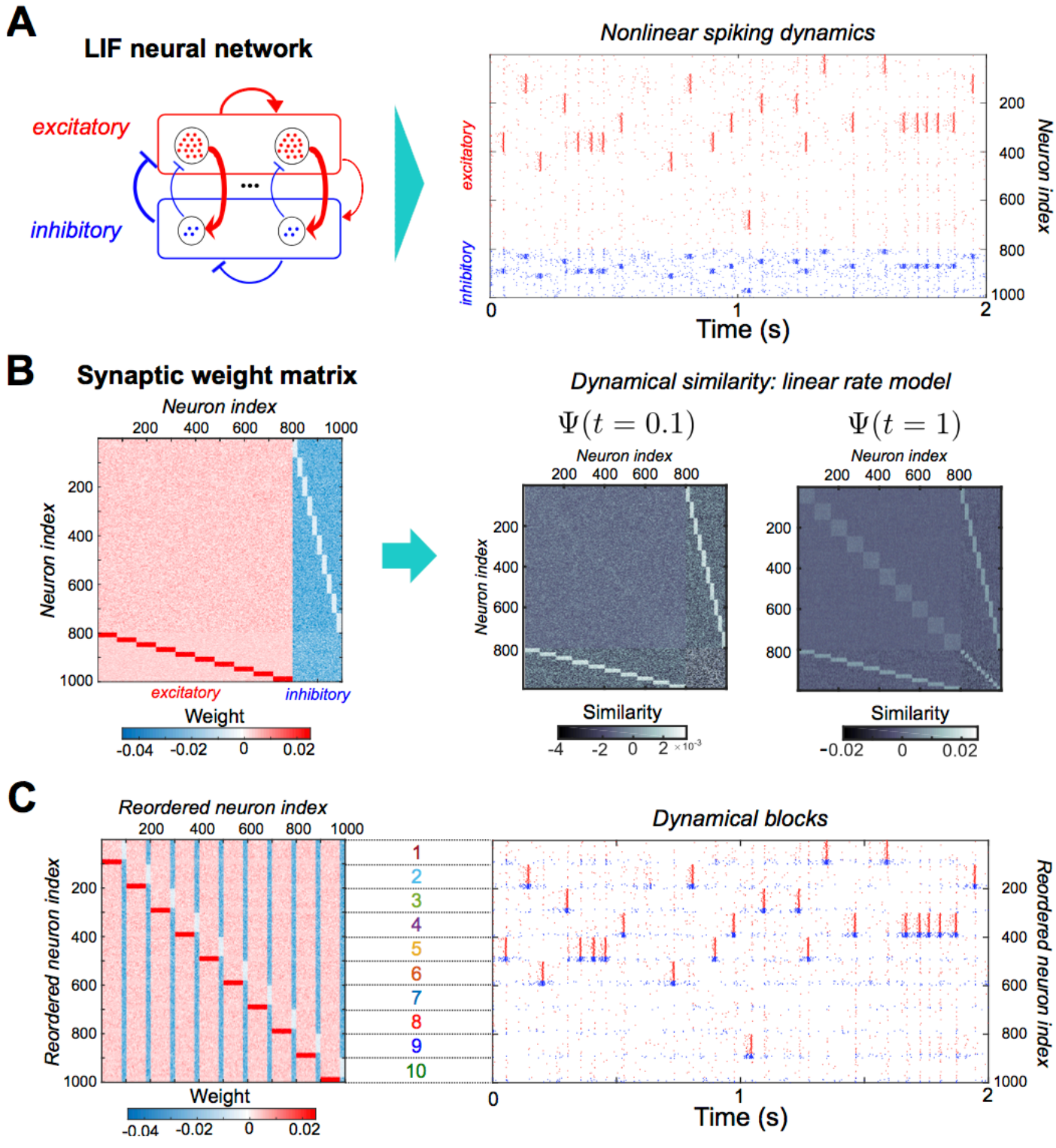


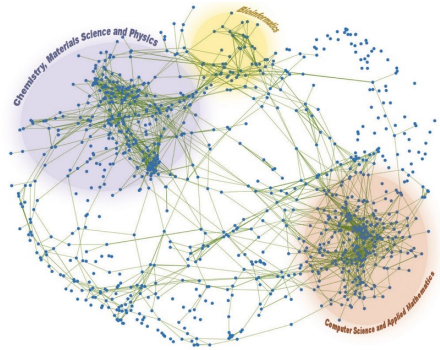
Figure 1. **Construction of the dynamical similarity measure $\Psi(t)$.** Impulses are applied as inputs to different nodes of the network. The responses in time are interpreted as node vectors evolving in state space, and can be compared, e.g., via an inner product from which we construct the similarity matrix $\Psi(t)$. Nodes that drive the system similarly (differently) within the projected subspace are assigned a high (low) similarity score.


A
Adjacency matrix
Node
1 2 3 4 5 6 7 8 9 10 11
Node
1 2 3 4 5 6 7 8 9 10 11
Weight
0 1 2 3 4 5
B
$\Psi(t=1)$
$\Psi(t=8)$
$\Psi(t=16)$
Node
Node
Similarity
0 0.5
Similarity
0 0.2
Similarity
0 0.2

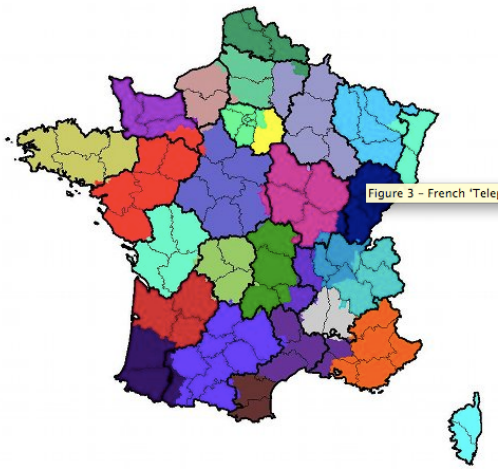

A
LIF neural network
excitatory
inhibitory
Nonlinear spiking dynamics
excitatory
inhibitory
Neuron index
Time (s)
B
Synaptic weight matrix
Neuron index
excitatory
inhibitory
Weight
Dynamical similarity: linear rate model
$\Psi(t = 0.1)$
$\Psi(t = 1)$
Similarity
C
Reordered neuron index
Weight
Dynamical blocks
Time (s)

2. Beyond networks

Multiple nature of Complex Data

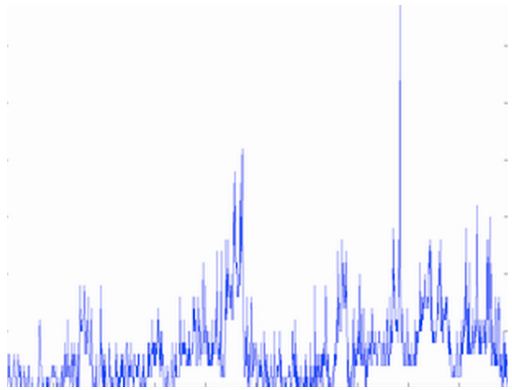


Relational



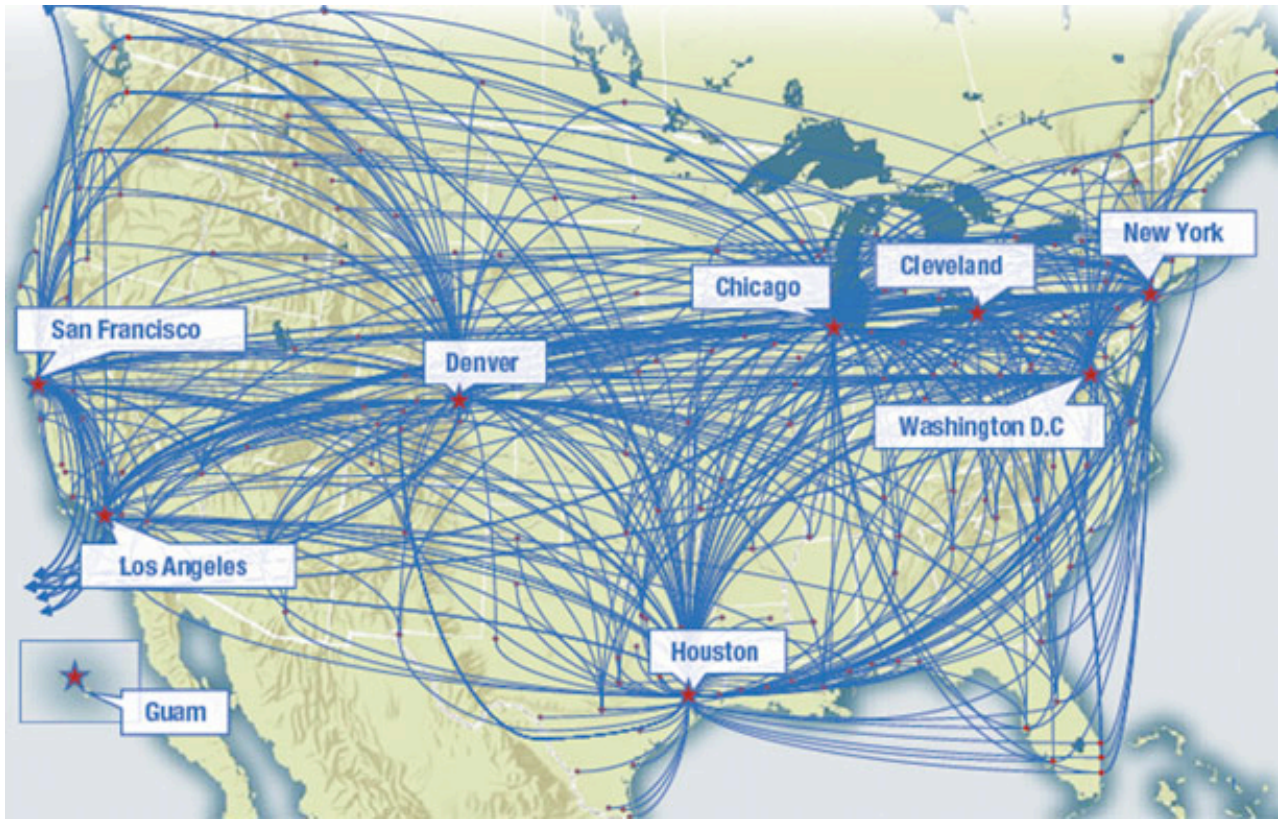
Geometric

Complex
Data



Temporal

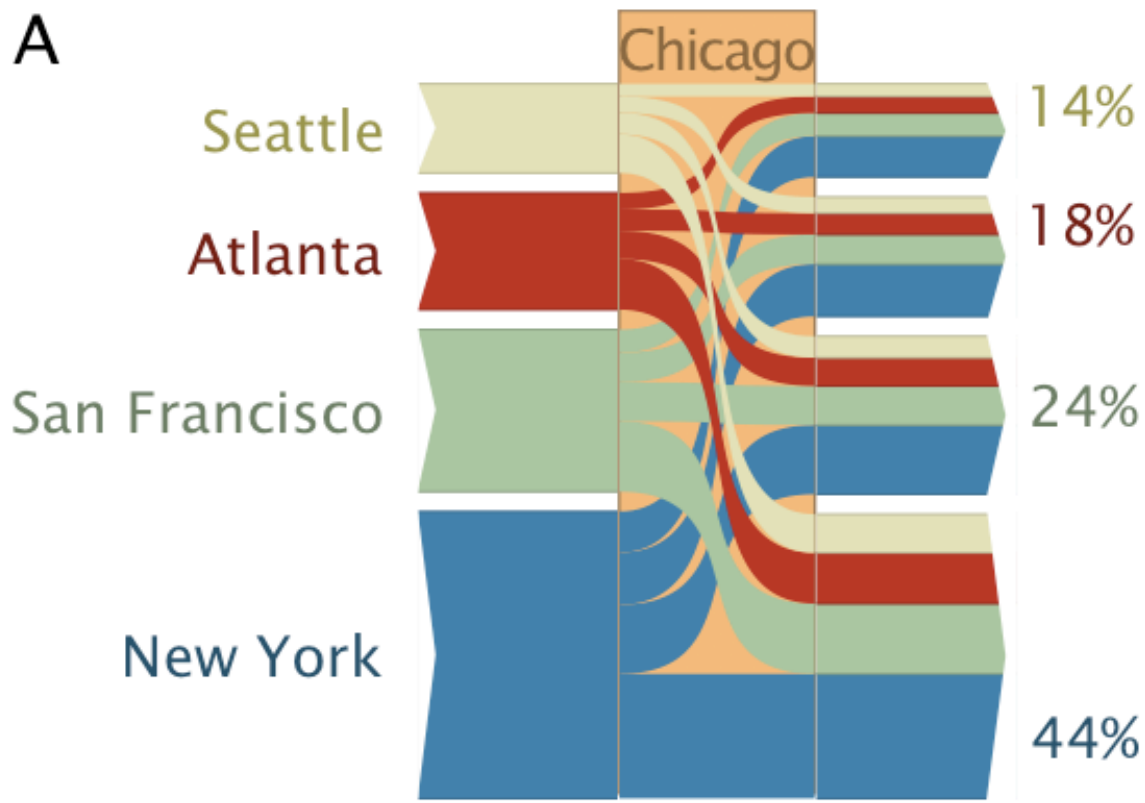
Where you go to depends on where you come from
Mathematics of pathways instead of edges



Rosvall, Martin, et al. "Memory in network flows and its effects on spreading dynamics and community detection." Nature Cmmunications

Importance of Pathways

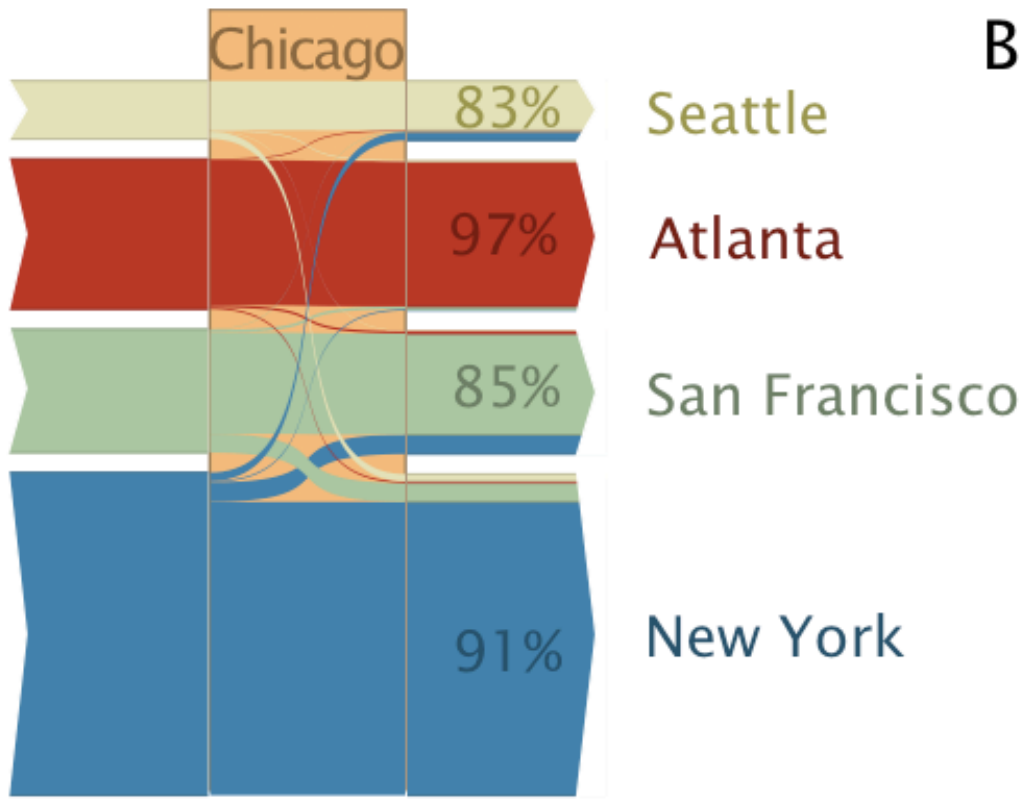
Where you go to depends on where you come from
Mathematics of pathways instead of edges



Rosvall, Martin, et al. "Memory in network flows and its effects on spreading dynamics and community detection." Nature Cmmunications

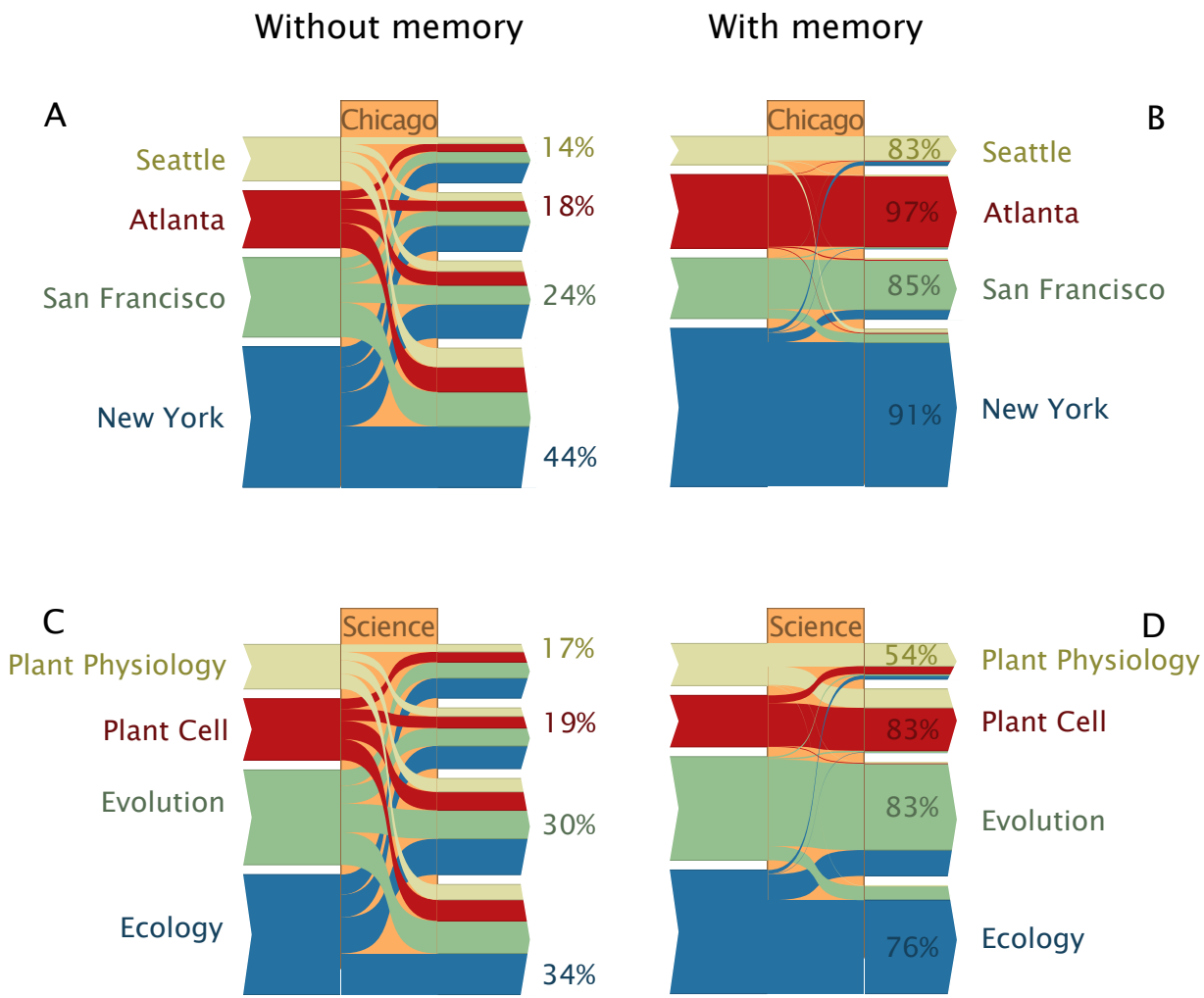
Importance of Pathways

Where you go to depends on where you come from
Mathematics of pathways instead of edges

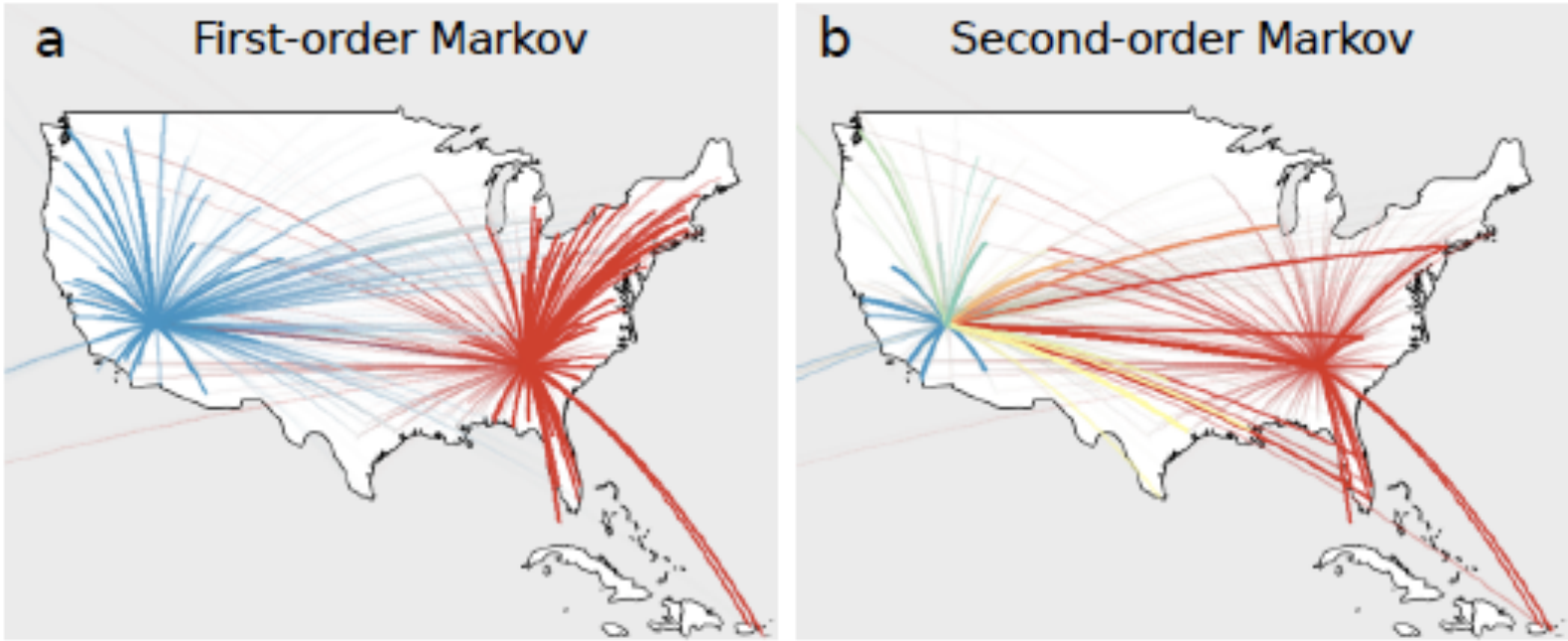


Importance of Pathways

Where you go to depends on where you come from
Mathematics of pathways instead of edges



Algorithms for memory networks: community detection

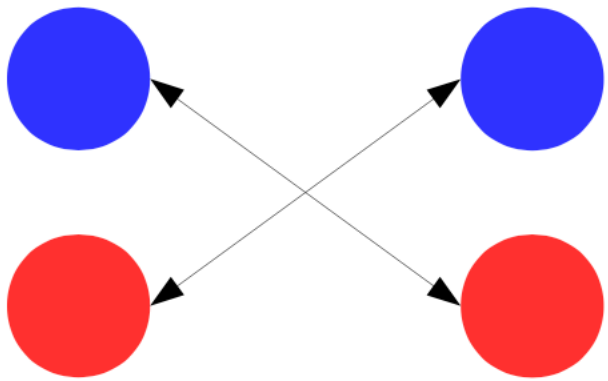


More realistic pathways -> more realistic modules.
Second-order Markov dynamics allow for better compression, because random dynamics on networks obscure essential structural information.
The method reveals smaller, more overlapping networks.
Connections with link partitioning and clique-percolation.

3. Local measures on networks: assortativity

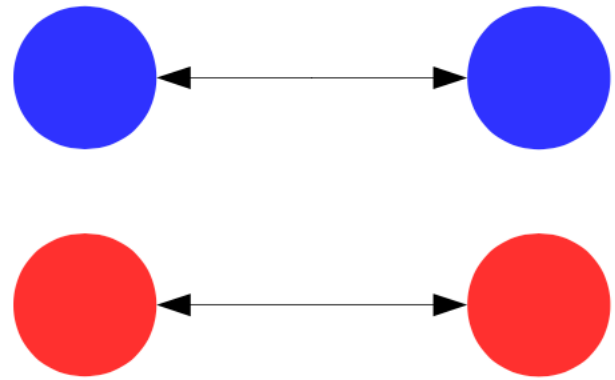
Disassortativity

also called "heterophily"



Assortativity

also called "homophily"



$$r_{\text{global}} = \frac{Q}{Q_{\max}} = \frac{\sum_g e_{gg} - \sum_g a_g^2}{1 - \sum_g a_g^2}$$

$$e_{gh} = \sum_{i:y_i=g} \sum_{j:y_j=h} \pi_i \frac{A_{ij}}{k_i}$$

usually considered globally, but….

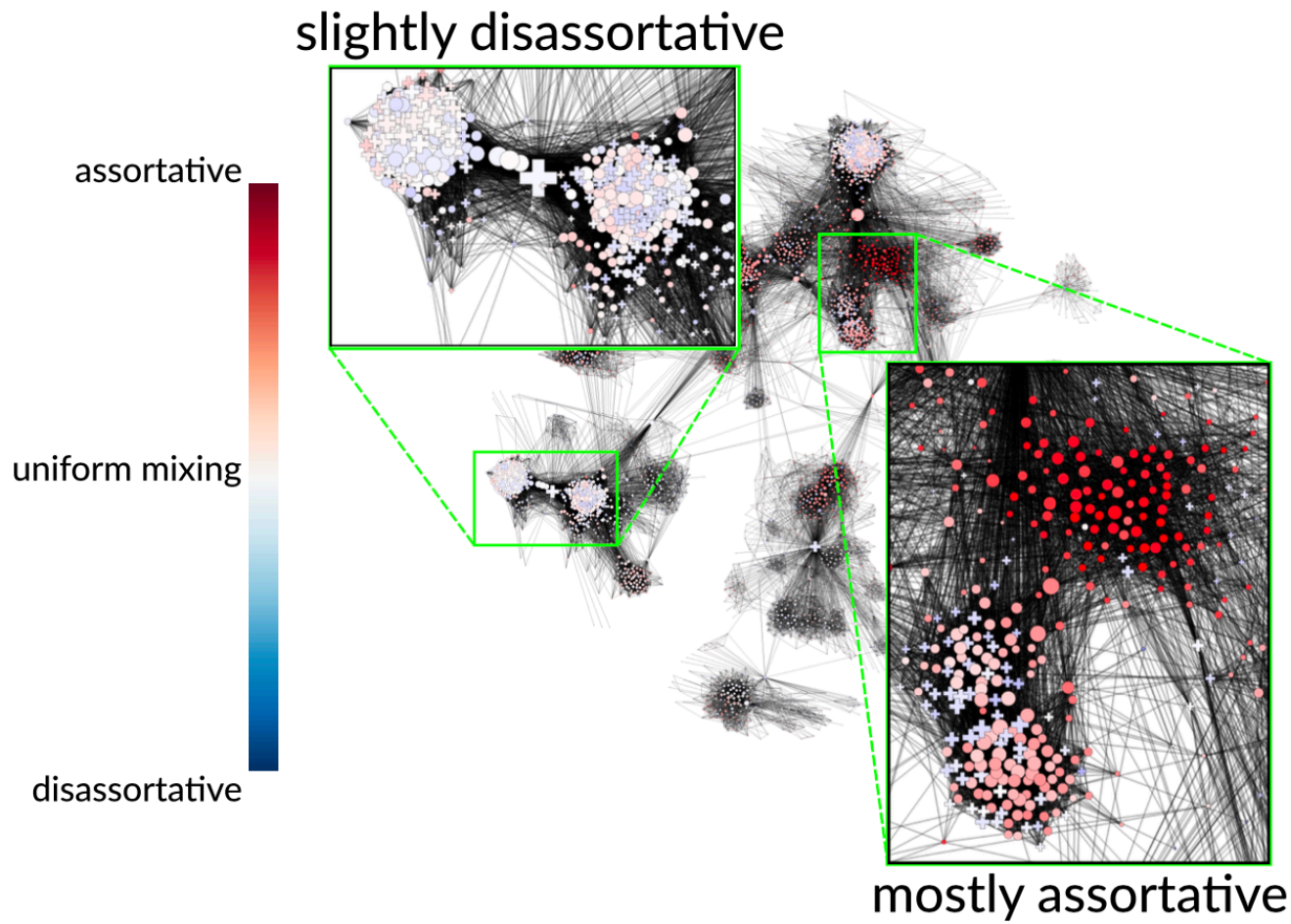


Fig. 1. Local assortativity of gender in a sample of Facebook friendships (14). Different regions of the graph exhibit strikingly different patterns, suggesting that a single variable, e.g. global assortativity, would provide a poor description of the system.

usually considered globally, but….

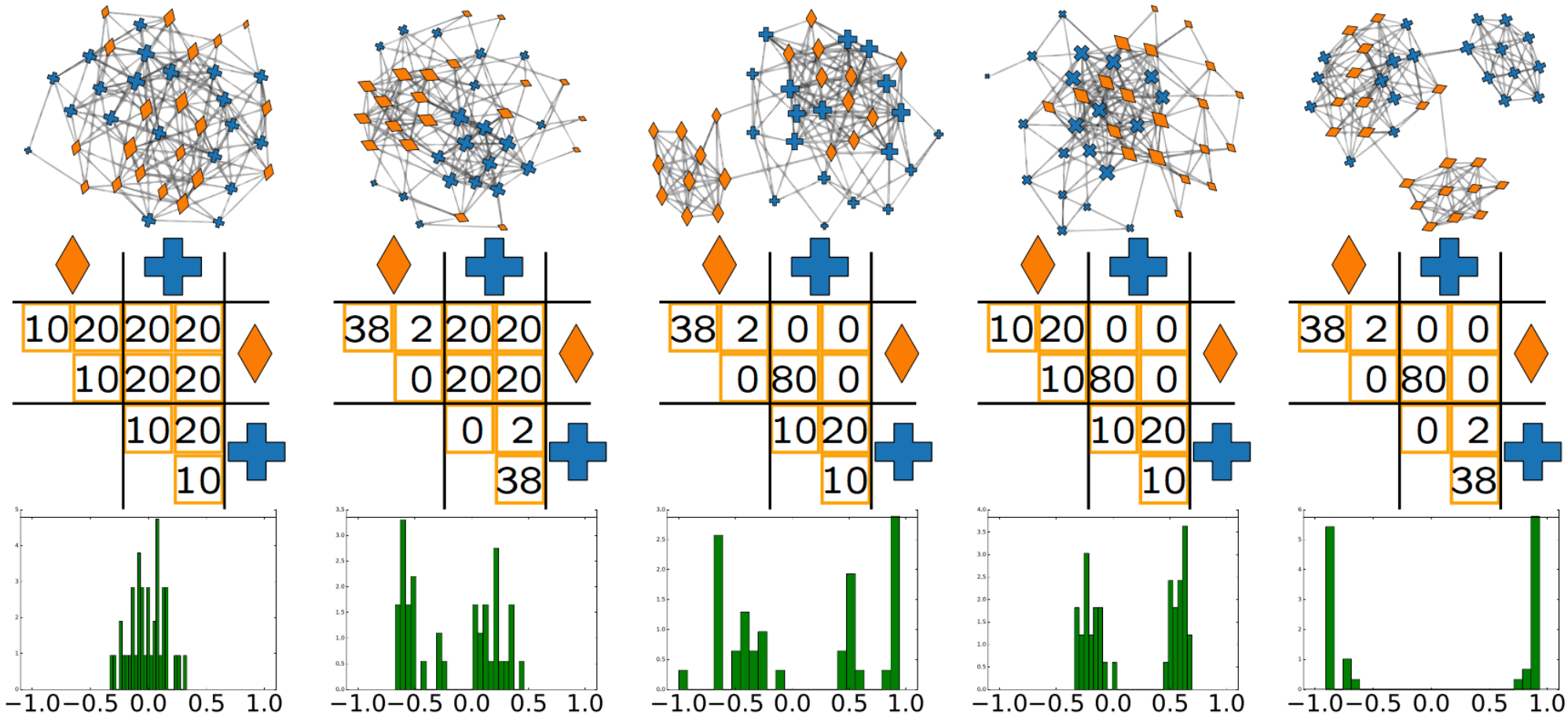
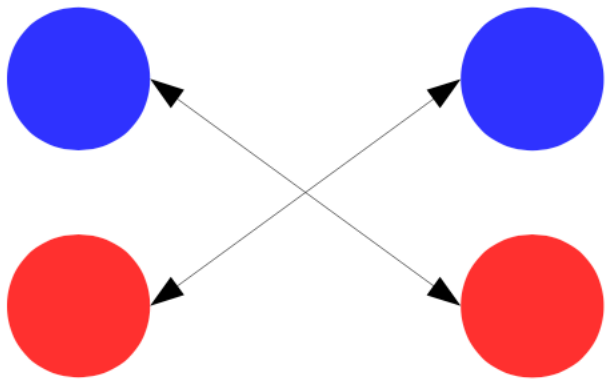


Fig. 2. Five networks (top) of $n = 40$ nodes and $m = 160$ edges with the same global assortativity $r_{\mathrm{global}} = 0$, but with different local mixing patterns as shown by the distributions of r_{multi} (bottom).

Local assortativity on networks

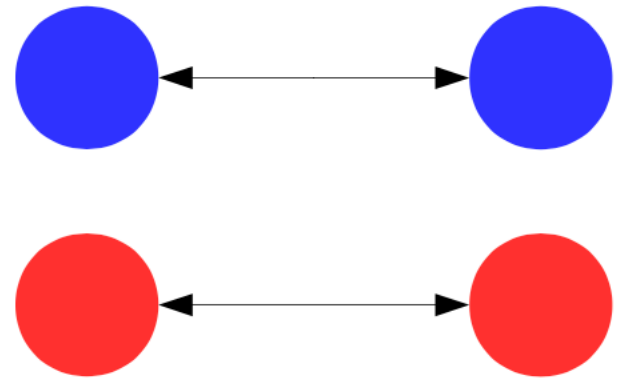
Disassortativity

also called "heterophily"



Assortativity

also called "homophily"

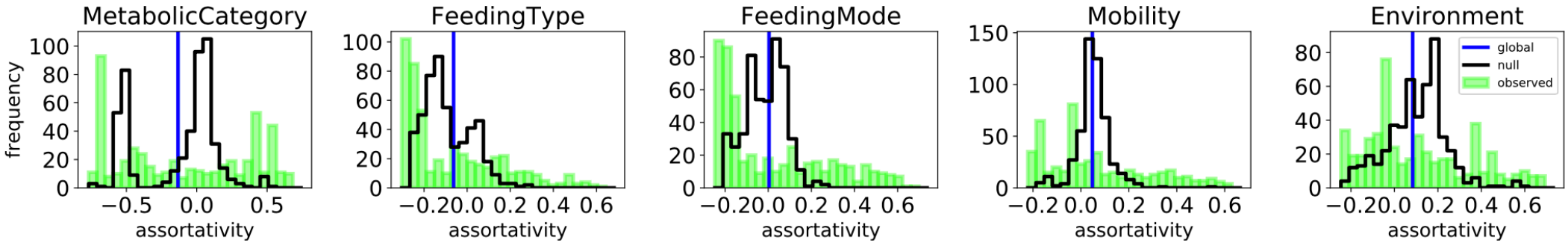


$$r_{\text{global}} = \frac{Q}{Q_{\max}} = \frac{\sum_g e_{gg} - \sum_g a_g^2}{1 - \sum_g a_g^2}$$

$$e_{gh} = \sum_{i:y_i=g} \sum_{j:y_j=h} \pi_i \frac{A_{ij}}{k_i}$$

$$e_{gh}(\ell) = \sum_{i:y_i=g} \sum_{j:y_j=h} w(i;\ell) \frac{A_{ij}}{k_i}$$

3. Local measures on networks: assortativity



null distribution (black) obtained by randomly re-wiring the edges such that the attribute values, degree sequence and global assortativity are all preserved

THANK YOU!

Random walks and diffusion on networks, Naoki Masuda, Mason A.Porter and Renaud Lambiotte
Physics Reports, Volumes 716–717, 22 November 2017, Pages 1-58

The many facets of community detection in complex networks, Michael T. Schaub, Jean-Charles Delvenne, Martin Rosvall, Renaud Lambiotte, Appl Netw Sci (2017) 2: 4

Multiscale mixing patterns in networks, L Peel, JC Delvenne and R Lambiotte, submitted

Multiscale dynamical embeddings of complex networks, Michael T. Schaub, JC Delvenne, R Lambiotte and M Barahona, submitted

From networks to optimal higher-order models of complex systems, Renaud Lambiotte, Martin Rosvall & Ingo Scholtes , Nature Physics 15, pages313–320 (2019)