1 Community structure and mixing patterns

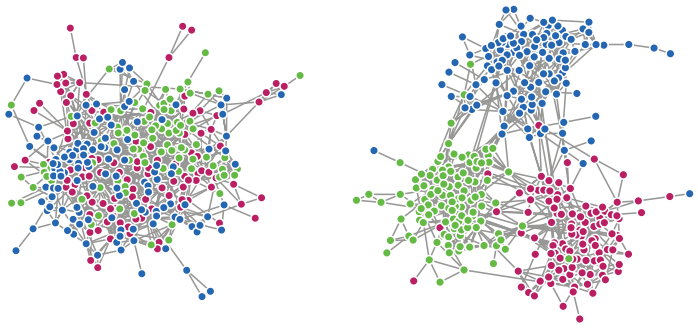
Network structure can vary in ways that local and global measures of a network's structure do not capture well, because local measures focus on individual nodes, and global measures tend to average equally across all nodes. To see this kind of "large-scale" structural variation, we need tools that see structural patterns *between* the local and global levels.

A common approach is to "cluster" nodes and edges into groups, which might range from relatively small (like a small motif) to relatively large (hundreds of nodes, or more), but always above the level of individual nodes and below the level of the whole network. Decomposing a network into such clusters provides what we might call a "coarse graining" of its structure into a simpler view that allows us to talk about how big pieces of a network fit together to construct the whole.

We will define the idea of a network cluster, called a **module** or a **community** or a **compartment**,[1] very generally, and this broad definition will cover a variety more specific notions of large-scale structure that we will learn about below.

a **community** *is a group of nodes that connect to other groups in similar ways.*

This is an inherently statistical definition (the word "similar"): nodes in the same group will follow the same statistical rules for how they connect to other nodes, and, these rules can vary between groups. Some groups might be *assortative*, meaning their nodes connect mainly to each other; some might be *disassortative*, meaning they connect mainly to nodes in other groups; and other groups might exhibit a mixture of these behaviors, connecting preferentially to some groups but not to others. The variation in these group-level connectivity patterns allows us to capture a wide variety of notions of modular structure in a network.



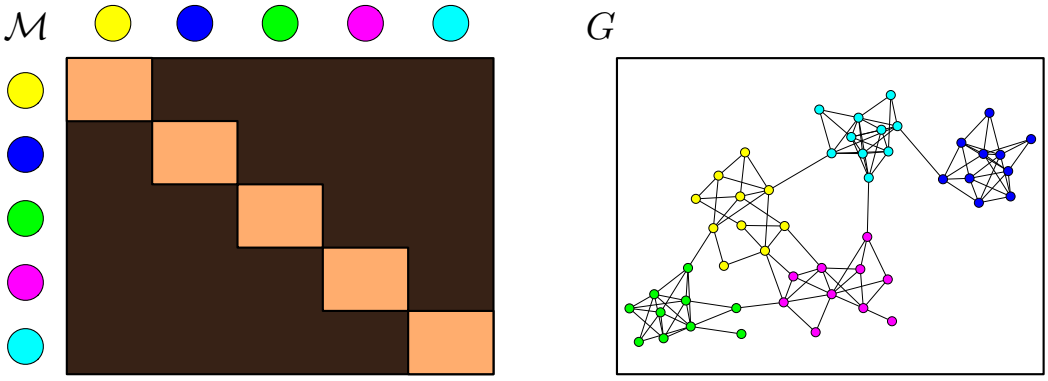
[1]The term "module" is most common in molecular biology, while "community" is more common in the social and physical sciences, and "compartment" is more common in ecology. Each refers to the same underlying idea.

In biological networks, communities (modules) may represent functional clusters of genes or proteins or neurons, while in ecological networks, communities (compartments) may represent groups of species that are functionally similar in an ecosystem, such as a group of predators that consume similar prey. These communities may represent targets of natural selection above the level of individual nodes but below the level of the whole system. In social networks, communities often represent different kinds of human social groups, e.g., with common interests, histories, or behaviors, or combinations thereof. In information networks, communities can indicate clusterings of related or functionally similar knowledge or ideas.

2 The mixing matrix of communities

We can formalize this notion of a community (module, compartment) using a simple tabulation called a **mixing matrix** $\mathcal{M}$. If there are c communities in the network, $\mathcal{M}$ is a $c \times c$ matrix, and the value of the element $\mathcal{M}_{rs}$ is related to the frequency or likelihood that a node in a group r connects to a node in a group s.

Below is a simple mixing matrix and its corresponding network. In the matrix, a lighter color indicates a greater density of edges (higher likelihood of connection), while a darker color indicates a lower density. Because the values along the diagonal $\mathcal{M}_{r=s}$ are higher than the values in the off-diagonal $\mathcal{M}_{r\neq s}$, this mixing matrix defines an *assortative* community structure: two nodes with the same group label are more likely to be connected than are two nodes with different labels. Hence, in the corresponding graph G, we see more edges within each particular group, than between them.



The mixing matrix naturally represents many different group-level connectivity patterns, depending on how the values of $\mathcal{M}_{rs}$ are organized. Once a particular pattern of group-level interactions is specified, we can use $\mathcal{M}$ to *generate* a random graph with that group-level structure, in a manner similar to how we generate other types of random graphs, e.g., via Erdős-Rényi or Chung-Lu. Or,

we could *infer* the $\mathcal{M}$ that describes the structure of some real-world network. Either way, the mixing matrix is a simple summary of how a network is organized at one particular scale because it shows how its parts at this scale connect to each other. If we vary the number of groups, we vary the scale of analysis—fewer groups means larger groups, while more groups means breaking the network into smaller pieces.

If groups all interact with each other in the same way, then it does not matter how many groups we define, because $\forall_{rs}\, \mathcal{M}_{rs} = \text{const}$. In this case, we might as well just say we have $c = 1$ group, and collapse the mixing matrix into a single number that represents the uniform (homogeneous) density of edges in the entire network. This situation should sound familiar: it is exactly the setting of an Erdős-Rényi random graph!

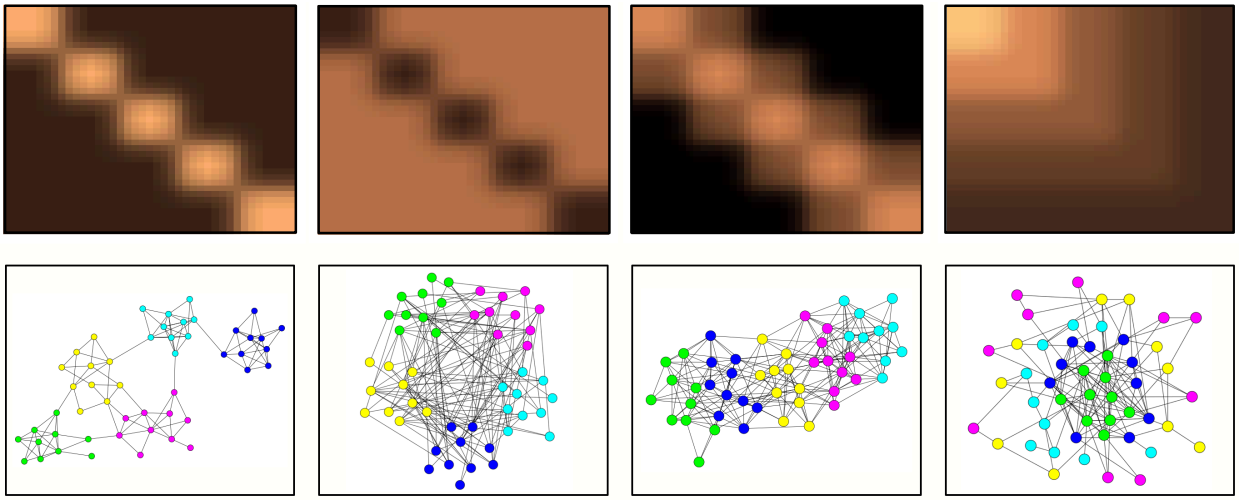
In contrast, if groups interact differently with each other (and themselves), then we say the mixing is heterogeneous—there are many different types of heterogeneous mixing. With $c > 1$ groups, we can arrange the $O(c^2)$ values of $\mathcal{M}$ to represent a wide variety of "stylized" modular patterns, such as:

- **assortative communities**: edges are mainly *within* the groups, and most interactions occur along the mixing matrix's diagonal, so that $\mathcal{M}_{rr} > \mathcal{M}_{r\neq s}$.
- **disassortative communities**: where edges are mainly *between* the groups, and most interactions occur in the matrix's off-diagonal, so that $\mathcal{M}_{rr} < \mathcal{M}_{r\neq s}$.
- **ordered communities**: where edges are mainly within groups or run between "consecutive" groups in a linear ordering; this pattern is similar to assortative communities, except that the values in the first off-diagonal fall between those of the main diagonal and the off-diagonal bulk, i.e., $\mathcal{M}_{rr} > \mathcal{M}_{r,r+1} > \mathcal{M}_{r,r+\ell}$ for $\ell \geq 2$.
- **core-periphery structure**: where edges tend to occur mainly within a relatively dense "core" community, which is surrounded by successively more sparse peripheral layers (kind of like an onion).

In the above examples, we assumed that the mixing matrix is undirected, but these matrices can be naturally generalized to directed interactions, so that $\mathcal{M}_{rs} \neq \mathcal{M}_{sr}$, which would capture the notion that edges tend to run more often in one direction than the other, e.g., in a food web, where predation links point down, toward basal or primary-produce species, or in a social network, where links represent some notion of social status.

2.1 Common forms of community structure

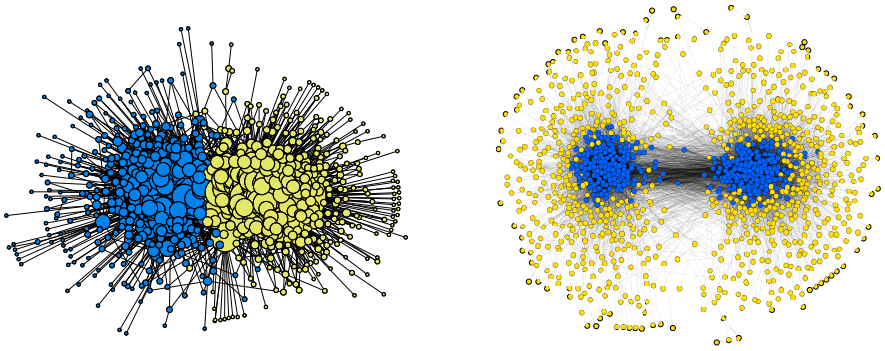
Most real-world networks exhibit a mixture of these four stylized patterns, and which ones we see can vary depending on the scale of analysis, i.e., what value of c we choose. For smaller choices of c, we might observe that communities are fairly assortative, while inside them, they exhibit



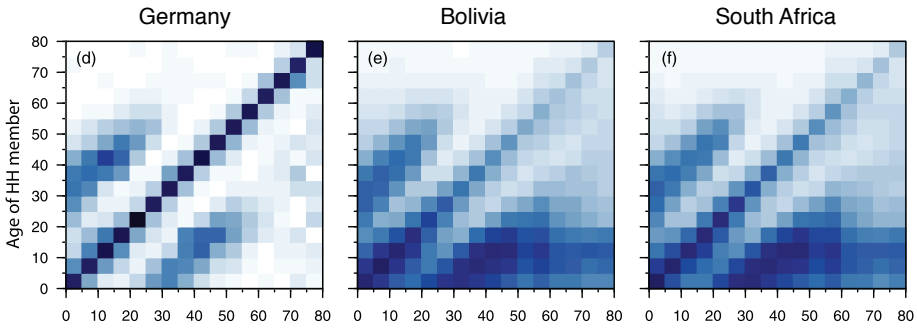
either an ordered or a core-periphery structure. A community might also be assortative (many edges within it), but exhibit different levels of connectivity with other communities, being densely connected to some, and sparsely connected to others. All of these variations are possible, and as c increases, the range of patterns and combinations of patterns also increases. (What happens to the mixing matrix $\mathcal{M}$ in the limit of $c \to n$?) Theoretical and empirical analyses suggest that for most networks $c \approx \sqrt{m}$, meaning that as a network grows in size (more edges m), we need a sublinear additional number of groups to account for the additional connectivity patterns.

Few real-world networks exhibit "clean" versions of the stylized patterns. The political blogs network we've seen previously exhibits a clear *assortative* community structure at the largest scale ($c = 2$; left-hand figure below). However, within each of those communities, there is also a *core-periphery* pattern: a relatively dense core surrounded by a diffuse, weakly interconnected periphery ($c = 4$; right-hand figure).[2] (What would the mixing matrices look like for each of these networks?)

[2]Figures from Karrer & Newman, *Phys. Rev. E* **83**, 016107 (2011) `https://arxiv.org/abs/1008.3926`, and from Zhang, Martin, & Newman, *Phys. Rev. E* **91**, 032803 (2015) `https://arxiv.org/abs/1409.4813`.



Ordered patterns are more prevalent in social networks. For instance, empirical studies show that social interactions tend to be ordered by age: most people interact with others who are within 5 years of their own age, and they interact at slightly lower rates for people within 5–10 years of their age.[3] But, an exception to this pattern also exists: children interact at high rates with individuals about 25–35 years (and about 60 years) older than them, which creates strong but specific off-diagonal patterns in the mixing matrices. This "age-structured" pattern is a key part of modern epidemiological modeling, and helps explain why infectious diseases like COVID-19 tend to spread within specific age groups first.



2.2 Hierarchies, and multiple scales

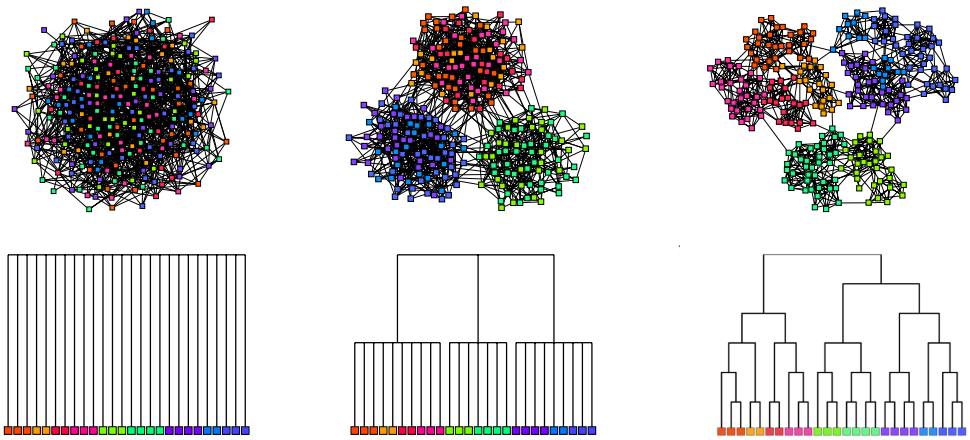
The concept of network modules and communities, and even the "flat" form of the mixing matrix itself, may lead us to conclude that there is unlikely to be additional structure inside a community

[3] Figures from Prem, Cook & Jit, *PLoS Comp. Bio.* **13**, e1005697 (2017) `http://dx.doi.org/10.1371/journal.pcbi.1005697`.

or between pairs of communities. If that's true, then we could conclude that there is only one level of organization between the local and global. But why stop at one?

A natural generalization of community structure is called *hierarchical* community structure, in which vertices divide into groups that further subdivide into groups of groups, and so forth over multiple scales. A hierarchy thus spans all the scales between the local and global levels. This kind of nested structure is often represented using a *dendrogram* or a tree that shows how the smallest groups are nested within larger groups, and they in turn are nested within still larger groups, etc.[4] Taking a "slice" across the dendrogram induces a single decomposition of the network into groups, and sliding that line up or down in the dendrogram provides a natural way to "zoom" in or out across scales.

A common assumption for this kind of large-scale organization is that two nodes that are "closely related", that is, are separated by a short distance on the tree, are more likely to be connected. This is typical, for instance, of human social structures, for instance: our class sits within the larger Computer Science department, which itself sits inside the larger College of Engineering, which sits inside CU, which sits inside Colorado schools, etc.



homogeneous modular hierarchical

3 Homophily and assortative mixing

The idea of *assortative mixing*, which is also called *homophily* in social networks, simply means that the attributes of vertices correlate across edges. That is, suppose vertices i and j have attributes x_i and x_j. If we observe an edge (i, j), then the attributes x_i and x_j will tend to be more similar

[4]Figure below from Clauset, Moore & Newman, *Nature* **453**, 98–101 (2008) `https://arxiv.org/abs/0811.0484`.

than a randomly chosen unconnected pair. Put succinctly, like links with like.[5]

In social networks, nearly all attributes are assortative because people have a very strong tendency to associate with others who are similar to them, e.g., in age, nationality, language, socioeconomic status, educational level, political beliefs, and many others. Crucially, however, the appearance of homophily is only a statement of pattern and does not identify the underlying mechanism. If we observe a pattern of homophily in a social network, e.g., on political beliefs or obesity, we generally cannot distinguish between (i) the edge forming as a result of the attributes being similar, or (ii) the attributes becoming more similar as a result of the edge.

The idea of *disassortative mixing* is the reverse: if the edge (i, j) exists, then x_i and x_j tend to be less similar than a randomly chosen unconnected pair. For example, in the general population, both dating and sexual contact networks are largely disassortative, with the large majority of edges running between men and women, with a smaller number of edges running among men or among women. Similarly, in food webs, predators tend to be connected to their prey, rather than to each other, and in economic networks, producers of widgets tend to connect to consumers of those widgets, rather than to each other.

Before we explore the more general idea of community structure, we will first explore basic ways to quantify the degree to which a network exhibits assortative behavior. Mathematically, there are two ways to do this, depending on whether the attribute x is a label or enumerative value (i.e., an unordered or categorical type like vertex color or shape), or a scalar value. We will cover them both for the sake of completeness, which will also produce a few new network-level summary statistics for describing network structure.

3.1 Enumerative attributes

Enumerative attributes are those that lack any particular ordering, and are sometimes also called categorical variables. They represent things like vertex color, shape, ethnicity, gender, etc. For simplicity of notation, let $x_i \in \mathcal{S}$ denotes an enumerative vertex attribute and let the number of possible values be $|\mathcal{S}| = k$.

The typical measure of this form of assortativity, i.e., the degree to which like things are connected, is called the *modularity* Q. When we are given the attribute x value for each vertex, calculating the modularity answers this question:

How much more often do attributes match across edges than expected at random?

That is, in order to determine whether some attribute mixes assortativity, i.e., occurs at either end of an edge surprisingly often, we need to write down a null model for mixing at random. Given this

[5] Or, birds of a feather link together.

model, we can then compute the total difference Q between the observed and expected fractions of edges for the $\mathcal{S} \times \mathcal{S}$ pairs of types. If $Q > 0$, then we conclude for assortative mixing and begin thinking about why this system should mix in this way.

There are two cases where we should expect $Q = 0$. The first is the trivial case where all vertices of are the same type. Here, there is only one type of edge and thus the observed fraction of edges must equal its expected value. The second is when attributes match no more or less frequently than we would expect at random. (Under what circumstances would this measure yield its maximum value of 1? Note that its minimum value is not zero, but is in fact -1. Under what circumstances would this value be achieved?)

The key question for writing down an expression for modularity Q is choosing a null model. For instance, consider a network in which a fraction q of all vertices are blue while the remaining are red. Under a simple random graph model where edges occur independently with the same probability, i.e., the Erdős-Rényi model, the expected number of edges that connect two blue vertices is $m \times q^2$ and the expected number for two red vertices is $m \times (1-q)^2$. If the empirical counts for these quantities is greater than their expected values, then we could conclude the presence of assortativity.

However, we know that simple random graphs are terrible models of real networks because they produce thin-tailed degree distributions. If some vertices have very high degrees, then their color label participates in more connected pairs, and this will skew our statistics if our null model cannot produce such high-degree vertices. Worse, if red vertices tend to be high-degree vertices, we might incorrectly rule against assortativity. The solution is to use the configuration model as the null model, rather than a simple random graph.

Given a labeling of vertices $\mathbf{x}$, the modularity for an undirected network is given by

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(x_i, x_j) \ , \tag{1}$$

where A is the adjacency matrix, k_i is the degree of vertex i, m is total the number of edges in the network, x_i is the label of vertex i, and $\delta(x_i, x_j)$ is the Kronecker delta function, which equals 1 when its arguments are the same and 0 otherwise.

Let us dissect this equation, starting with the summation. The sum is over all pairs of vertices i, j, regardless of whether there is an edge there or not. The delta function serves as a kind of filter, selecting only those pairs i, j whose labels are the same $x_i = x_j$. Thus, the summation is effectively only over pairs of vertices with the same type label. The inner term is the difference between the observed fraction of all edges between i and j, which is $A_{ij}/2m$, and its expected value under a random graph with the same degree distribution. If the degrees of vertices i and j are k_i and k_j,

then the probability that i and j are connected, if edges are distributed at random conditioned on respecting these degrees, is the number of chances they have to connect $k_i k_j$ divided by the total number of such pairs, which is $(2m)^2$. Because both terms have a common denominator of $1/2m$, we may factor this out, which gives the leading constant.

Notice that the summation is only over edges among vertices of the same type. This allows us to rewrite and simplify the equation by dropping the many zeros in the total summation. To accomplish this, we define two auxiliary quantities

$$e_{uv} = \frac{1}{2m} \sum_{ij} A_{ij}\, \delta(x_i, u)\delta(x_j, v) \ , \tag{2}$$

which is the fraction of edges that join vertices with label u to vertices with label v, and

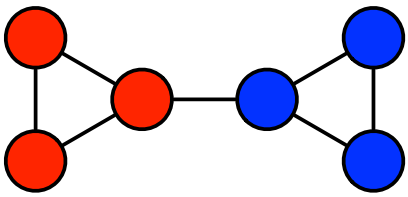
$$a_u = \frac{1}{2m} \sum_{j} k_j\, \delta(x_j, u) \ , \tag{3}$$

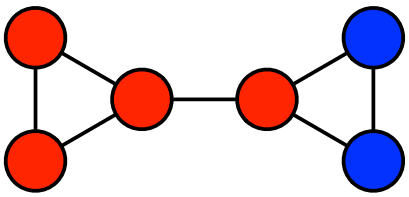
which is the fraction of ends of edges that are attached to vertices with label u. (Exercise: show that $a_u = \sum_v e_{uv}$, i.e., show that a_u is the column (or row) sum of the e_{uv} matrix.) The modularity is then

$$\begin{aligned} Q &= \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \sum_u \delta(x_i, u)\delta(x_j, u) \\ &= \sum_u \left[\frac{1}{2m} \sum_{ij} A_{ij}\, \delta(x_i, u)\delta(x_j, u) - \frac{1}{2m} \sum_i k_i\, \delta(x_i, u) \frac{1}{2m} \sum_j k_j\, \delta(x_j, u) \right] \\ &= \sum_u \left(e_{uu} - a_u^2 \right) \ , \end{aligned} \tag{4}$$

where e_{uu} is the observed density of edges among vertices with label u and a_u^2 is the expected density under the configuration model. This form of Eq. (1) is more compact and may be used even when we do not have the full adjacency matrix. That is, to compute Q using Eq. (4), we only need a matrix containing the number of connections between each pair of vertex types. Note also that both equations are defined only for undirected graphs. Directed or weighted versions are easily defined, although less commonly used.

To illustrate a modularity calculation, consider two distinct labelings on the small modular network with $n = 6$ vertices and $m = 7$ edges shown below. First, because the number of types $k = 2$, we construct a 2×2 matrix based on each labeling, where each matrix element counts the fraction of edges of a given pair type:





labeling 1	red	blue
red	3/7	1/14
blue	1/14	3/7

labeling 2	red	blue
red	4/7	2/14
blue	2/14	1/7

From these matrices, the modularity is straightforward to calculate, yielding $Q_1 = 5/14 = 0.357$ for the first, and $Q = 6/49 = 0.122$ for the second. In this case, labeling 1 yields a higher modularity than labeling 2 because it has a larger fraction of within-type edges, i.e., it displays more assortative mixing.

3.2 Scalar attributes

When vertex attributes are scalar values, like age, weight, or income, we may say not only when two vertices have the same value, as in the previous case, but also when two vertices are close in value. We again let x_i denote the vertex attribute, but now let x vary as an integer or real value.

The typical measure for this form of assortativity is called the *assortativity coefficient*, which is a kind of network-based generalization of the Pearson correlation coefficient. This measure answers the following question:

How much more similar are attributes across edges than expected at random?

That is, we again invoke a null model in order to decide whether vertices tend to link preferentially with those having similar attributes. Now, however, we are interested in attribute similarity, rather than simple matching.

To write down the expression for the assortativity coefficient, we adapt the standard expression of covariance to a network context. The mean value observed at either end of an edge is simply $\mu = (1/2m) \sum_i k_i x_i$, which weights each observed scalar value by the number of edges in which it participates.

The covariance of x across edges is then

$$\begin{aligned}\mathrm{cov}(\vec{x}, A) &= \frac{\sum_{ij} A_{ij}(x_i - \mu)(x_j - \mu)}{\sum_{ij} A_{ij}} \\ &= \frac{1}{2m}\sum_{ij} A_{ij}x_i x_j - \mu^2 \\ &= \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{2m}\right)x_i x_j \ , \end{aligned} \tag{5}$$

where we have reused the definition of μ, and simplified considerably, to get to the last line.

Note that this form of the covariance is remarkably similar to the definition of the modularity given in Eq. (1), except that instead of a delta function selecting only edges for which the attributes match, we now weight every edge by the product $x_i x_j$. When the values of the last term tend to agree, i.e., small values are multiplied by small values and large values multiplied by large values, the covariance will be positive, indicating assortative mixing, while when the opposite is true, the covariance is negative, indicating disassortative mixing.

Finally, just as in the case of the modularity, and in the case of the traditional definition of the Pearson correlation coefficient, it is useful to normalize the covariance so that it ranges from -1 to 1. This version has the form

$$r = \frac{\sum_{ij}\left(A_{ij} - k_i k_j /2m\right)x_i x_j}{\sum_{ij}\left(k_i\,\delta(i,j) - k_i k_j /2m\right)x_i x_j} \ , \tag{6}$$

where $\delta(i,j) = 1$ if $i = j$ and 0 otherwise. We call r the assortativity coefficient, and is exactly the covariance divided by the variance, i.e., it is a correlation measure.

3.3 Vertex degrees

Vertex degree is a scalar attribute that is of particular interest as it reveals important insights about the large-scale structure of the network and can be calculated from the network structure alone. That is, we set $x_i = k_i$, which slightly simplifies Eq. (6).

Assortative mixing by degree produces a network in which the high-degree vertices tend to connect to each other in dense, high-degree core, while the low-degree vertices also connect to each other, producing a sparse, low-degree periphery. In these networks, degree correlates with centrality. By contrast, disassortative mixing by degree produces a network in which the high-degree vertices tend to connect to low-degree vertices, producing star-like structures. In these networks, centrality correlates less strongly with degree. (See Figure 7.12 in *Networks* for a good visualization of this distinction.)

4 Random graphs with modular structure

A more powerful (and modern) way to to define and understand community structure is to define a model of modular random graphs, which is a distribution over graphs $\Pr(G)$ conditioned on having a specified modular or community structure, and optionally a specified degree distribution as in the Chung-Lu model. In this way, the modular random graph model builds on previous random graph models, such as the Erdős-Rényi model $G(n,p)$, which specifies only the edge density p, and the Chung-Lu model, which specifies the expected degree sequence $\vec{k}$.

4.1 The stochastic block model

The simplest random graph with modular structure is a generalization of the Erdős-Rényi model called the **stochastic block model** or SBM.[6] This model was first studied in mathematical sociology in the mid-1980s. Today, it's common in all branches of network analysis, from computational biology, to machine learning and statistical physics.

An SBM graph is typically a simple graph, defined by a tuple of parameters $\theta = (c, \vec{z}, \mathcal{M})$, where

1. c is the number of communities,
2. $\vec{z}$ is a labeling or a **partition** of nodes by community membership so that $z_i \in \{1, 2, \ldots, c\}$, and we say $n_r = \sum_{i=1}^{n} \delta_{z_i,r}$ is the number of nodes in community r, and
3. $\mathcal{M}$ is a $c \times c$ "stochastic block matrix" where $\mathcal{M}_{rs}$ gives the probability that a node in community r connects to a node in community s (the mixing matrix, defined precisely),

where $\delta_{z_i,r} = 1$ if $z_i = r$ and is 0 otherwise.[7]

Given these parameters, the probability of an undirected edge is

$$\forall_{i>j} \qquad A_{ij} = A_{ji} = \begin{cases} 1 & \text{with probability } \mathcal{M}_{z_i,z_j} \\ 0 & \text{otherwise .} \end{cases}$$

That is, given the labels $r = z_i$ and $s = z_j$ for a pair of nodes i, j, we simply look up in the mixing matrix $\mathcal{M}$ the corresponding density parameter, and flip a coin with that bias. The variation in these edge density parameters represents the block or community structure of the graph, and all of the conceptual ideas introduced in Section 2 carry over naturally. For the subgraph defined by a single community, i.e., when $r = s$, the SBM produces a simple Erdős-Rényi random graph with density $\mathcal{M}_{rr}$, while for the subgraph defined by a pair of communities, i.e., when $r \neq s$, the SBM produces a simple random bipartite graph with density $\mathcal{M}_{rs}$.

[6] Sometimes also called the "stochastic blockmodel." Introduced in Holland, Laskey & Leinhardt, *Social Networks* **5**(2), 109–137 (1983) `https://bit.ly/2legJUO`, and then further developed in Wang & Wong, *J. American Statistical Association* **82**, 8–19 (1987) `https://bit.ly/2mKrJKl`.

[7] The term $\delta_{a,b}$ is the Kronecker delta function, which behaves like an indicator variable: `https://en.wikipedia.org/wiki/Kronecker_delta`.

4.1.1 Properties of SBM graphs

At a high level, SBM graphs have the following properties:

- Traditionally a *simple graph*, but also easily generalized to directed edges, with or without self-loops (do you see how?).
- Connectivity is *modular*, and is governed by the sizes of the c groups $\vec{n}$ and their groupwise edge densities $\mathcal{M}$; edges are independent, conditioned on their nodes' labels.
- The degree distribution $\Pr(k)$ is a combination of Poisson distributions, each with mean $\mathcal{M}_{rs}$, which can produce high-variance distributions only if node degree correlates with group label (i.e., high degree nodes are grouped together).
- The diameter and mean geodesic distance are typically $O(\log n)$.
- The clustering coefficient is $C = O(1/n)$, depending on group sizes and densities $\mathcal{M}$.
- The largest connected component (LCC) tends to be $O(n)$ when G is not too sparse.

Because edges are conditionally independent (conditioned on z_i, z_j only), we sometimes say that nodes in the same group are *stochastically equivalent*, because they have equivalent connectivity patterns to other vertices. That is, every vertex in group r has the same set of probability values that govern the connections to other vertices, and this set is given by the rth row (or column) of the matrix $\mathcal{M}$.

In that sense, the SBM is the most literal translation of our the definition of a community we saw in Section 1: a community in the SBM is, by definition, a group of nodes with the same rules for connecting to nodes in other groups. The effective definition of a group r, then, is given by the rth row vector of groupwise densities $\mathcal{M}_{r*}$—if a node follows those rules, then it is "stochastically equivalent" to all the other nodes that follow the same rules, and these nodes define the rth group.

4.1.2 Generating a SBM network

Once the parameters $\theta = (c, \vec{z}, \mathcal{M})$ have been specified, drawing a graph from the ensemble is straightforward:

1. initialize an empty graph G with n nodes
2. for each of the $\binom{n}{2}$ pairs i, j, draw a uniformly random number r_{ij}
3. if $r_{ij} \leq \mathcal{M}_{z_i,z_j}$, then add the undirected edge (i, j) to G.

This procedure is nearly the same as the one we used for generating both Erdős-Rényi graphs and Chung-Lu graphs, which also flip $\binom{n}{2}$ coins, one for each pair $i, j \in V \times V$. All three of these models are "independent" random graph models, in which edge existences are independent and occur with probability $\Pr(i \to j \,|\, \theta)$, where θ is the model's parameters. For the SBM, which

is parameterized by the mixing matrix $\mathcal{M}$ and partition z, the probability of an edge existing is $\Pr(i \to j \,|\, \mathcal{M}, z) = \mathcal{M}_{z_i,z_j}$. Generating a network in this way takes $\Theta(n^2)$ time, and is computationally expensive when $n > 10^4$ or so.

The SBM has a large number of parameters, and this greater flexibility is what allows it to produce such a broad variety of large-scale network patterns. In total, there are $1 + n + \Theta(c^2)$ parameters, which correspond to the number of values required to specify the number of groups c, a group label for each node $\vec{z}$ and the groupwise edge densities $\mathcal{M}$, respectively.

4.1.3 Planted partitions

When we generate a synthetic network using the SBM, we have to specify both the partition of nodes $\vec{z}$, and the mixing matrix $\mathcal{M}$. Once specified, we sometimes call these a *planted partition*, since we're "planting" a particular kind of structure inside the random graph. Although the full flexibility of the model can be useful, often it is convenient to make the planted partition very simple, in order to induce a stylized pattern (Section 2) whose structure is controlled by a small number of parameters. Such simplified models are often very useful as a substrate for numerical experiments.

The most common form of a planted partition is a simplified assortative pattern, where we couple together all the off-diagonal (between group) densities using a single parameter. This reduces the $\Theta(c^2)$ free parameters in $\mathcal{M}$ to only $c+1$ parameters: one for the between-group densities q, and c parameters $\vec{p} = p_1, p_2, \ldots, p_c$ for each of the within-group densities, like so:

$$\mathcal{M}^{\text{planted-}c}_{\text{assort}} = \begin{pmatrix} p_1 & q & \cdots & q \\ & p_2 & \ddots & \vdots \\ & & \ddots & q \\ & & & p_c \end{pmatrix} .$$

We can then vary the group sizes $\vec{n}$ and the densities within each community $\vec{p}$ to produce different types of assortative patterns.

The most well-studied planted partition model simplifies this form in four additional ways, producing a *one* parameter model:

1. Equalize the within-group densities: $\forall_i \; p_i = p$. This choice reduces the mixing matrix to just two parameters, p for within-group edges, and q for between-group edges
2. Equalize the group sizes: $\forall_i \; n_i = n/c$.
3. Set $c = 2$ groups.
4. Fix the mean degree to be $\langle k \rangle = d$.

Hence, the generative model becomes

$$\mathcal{M}_{\text{assort}}^{\text{planted-2}} = \begin{pmatrix} p & q \\ & p \end{pmatrix} \quad \text{and} \quad n_1 = n_2 = n/2 \ , \tag{7}$$

and the fourth constraint implies that p and q are no longer independent. That is, in order for the mean degree to remain fixed at d, if we increase the within-group density p, then we must decrease the between-group density to compensate q by the same amount, and vice versa.

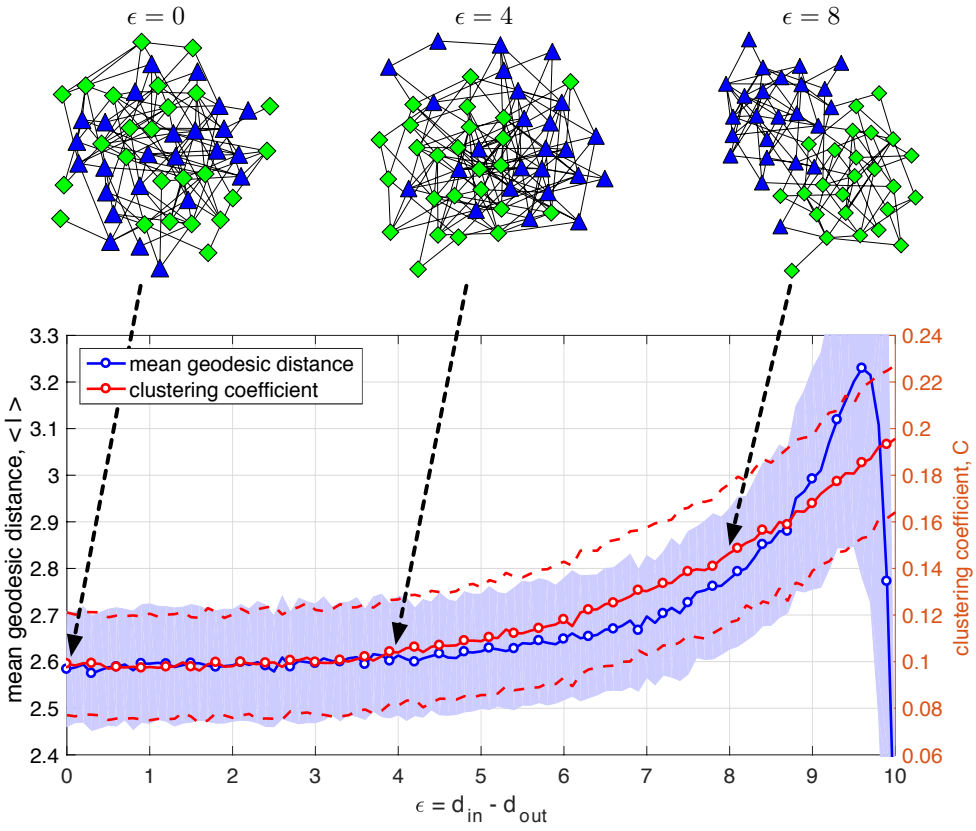
Let's define $p = d_{\text{in}}/n$ and $q = d_{\text{out}}/n$, where d_{in} and d_{out} are twice the within- and between-group degrees of each node (do you see why?). A little algebra can then show that $2\,d = d_{\text{in}} + d_{\text{out}}$. Defining $\epsilon = d_{\text{in}} - d_{\text{out}}$, we can then solve the $2d$ expression to obtain $d_{\text{out}} = d - \epsilon/2$ and $d_{\text{in}} = d + \epsilon/2$. Plugging these into our original expressions for p and q yields

$$p = (d + \epsilon/2)/n \qquad q = (d - \epsilon/2)/n \ .$$

Now, if we fix n and d, we have a simple, one-parameter model in which varying ϵ changes the relative balance of within- vs. between-group edges for each node. When $\epsilon \gg 0$, within-group edges far outnumber between-group edges, and the two groups are very distinct. As $\epsilon \to 0$, the relative difference goes away, and the groups become indistinguishable (as in an ER graph).

Planting partitions for simulation experiments. The value of ϵ naturally varies from $[0, 2d]$ for assortative communities (do you see what happens if $\epsilon < 0$?). Given a choice of ϵ, we can derive the values of p and q, and then generate a synthetic network. By varying ϵ, which quantifies how "crisp" the division is between a pair of communities, we can explore how different measures of network structure, either local or global, depend on a network's community structure.

For example, setting $n = 50$ nodes and the mean degree $d = 5$, the following figure shows how the mean geodesic distance $\langle \ell \rangle$ (left-hand y-axis) and the clustering coefficient C (right-hand y-axis) vary as a function of ϵ. Each point on either line is the average measure over 250 repetitions, and the shaded (dashed) region represents one standard deviation above and below this mean behavior. For three choices of ϵ, a network visualization illustrates the shape of a corresponding graph.
When $\epsilon < 4$ or so, both network measures are relatively stable, even though the relative share of within-group to between-group connections, given by ϵ, is changing. Above $\epsilon = 4$, we see that the clustering coefficient responds more quickly to the increasing division between the communities, while the $\langle \ell \rangle$ rises only slightly. It's not until around $\epsilon = 6$ that the $\langle \ell \rangle$ starts to increase more noticeably, and then it does so very quickly, especially for $\epsilon > 8$. This difference in the responsiveness of $\langle \ell \rangle$ and C to increasingly strong community divisions ϵ illustrates a key insight: $\langle \ell \rangle$ has a more non-linear response to structural changes than C does. In fact, the $\langle \ell \rangle$ peaks just before and then collapses at $\epsilon = 2d$ (do you see why?), while C exhibits only a smooth increase over the entire range.



There are many ways we could use the planted partition model, or more elaborate variations. For instance, we could define a simple planted partition model that represents core-periphery structure (do you see how to do that?) and use it to investigate how a node's mean geodesic distance $\langle \ell \rangle_i$ varies as a function of how sparse the periphery is. Etc. The utility of the planted partition model comes primarily from its ability to be a substrate for other analyses, and any useful specification of $\vec{z}$ and $\mathcal{M}$ can provide great insight about how structure varies, and how it shapes dynamics that run on top of the network.

4.2 The degree-corrected stochastic block model

Just as the stochastic block model is a modular generalization of Erdős-Rényi random graphs, the **degree-corrected stochastic block model** or DC-SBM is a modular generalization of a random graph with specified degree structure.[8] Combining a specified group structure and a specified degree structure makes the DC-SBM the most flexible random graph model in our toolbox, and

[8] Introduced in Karrer & Newman, *Phys. Rev. E* **83**, 016107 (2011) `https://arxiv.org/abs/1008.3926`.

also the most realistic.[9] The DC-SBM was introduced in 2010, and is now widely used in many branches of network analysis.

There are notable differences between the DC-SBM and the SBM. First, the DC-SBM is an undirected *multigraph model*. In the DC-SBM, each adjacency value A_{ij} is a Poisson random variable with a mean that depends both on the degrees k_i, k_j and on their groupwise mixing parameter $\mathcal{M}_{z_i,z_j}$. This makes A_{ij} a counting variable rather than a binary 0 or 1, as in the SBM. To output a simple graph, we can just collapse the multi-edges. Second, because we need to balance the effect of the mixing matrix with the effect of the degree sequence, the DC-SBM requires a little more work to parameterize properly.

Concretely, beyond the number of nodes n, a DC-SBM is specified by a tuple $\theta = (c, \vec{z}, \vec{k}, \mathcal{M})$:

1. c is the number of communities,
2. $\vec{z}$ is a partition of the nodes, by their community membership, where $z_i \in \{1, 2, \ldots, c\}$,
3. $\vec{k}$ is the expected degree sequence (like the Chung-Lu model), and
4. $\mathcal{M}$ is a $c \times c$ mixing matrix, where $\mathcal{M}_{rs}$ counts the number of edges between groups r and s, or, if $r = s$, twice that number, implying $\mathcal{M}_{rs} = \sum_{ij} A_{ij}\delta_{z_i,r}\delta_{z_j,s}$.

Finally, given the partition $\vec{z}$, we define the rth group's size $n_r = \sum_{i=1}^{n} \delta_{z_i,r}$, i.e., the number of nodes with label r.

Crucially, the definition of $\mathcal{M}$ in the DC-SBM is different than in the SBM: in the DC-SBM, $\mathcal{M}_{rs}$ is an edge count, not an edge density. This fact implies that the matrix sum $\sum_{rs} \mathcal{M} = 2m$, because every edge must run between some pair of groups r, s.[10]

It's useful think of the DC-SBM as a two-level model. In the first level, the groups themselves act as nodes in a group-level network, and the mixing matrix $\mathcal{M}$ gives the connectivity among these groups (like an adjacency matrix, but for a multigraph). In the second level, each group node gets expanded into a collection of individual nodes. At this level, a connection between a pair of nodes i, j in the same group $r = z_i = z_j$ will be equivalent to a self-loop for the rth group in the first level (counted by $\mathcal{M}_{rr}$). And, much like the planted partition model for the SBM, each node's degree k_i can be divided into a within-group portion, which contributes to $\mathcal{M}_{rr}$ and a between-group portion, which contributes to $\mathcal{M}_{r\neq s}$. This ideas will be useful when we describe below how to generate a graph from a DC-SBM.

[9] By using explicit parameters for degrees $\vec{k}$, the DC-SBM can use the mixing matrix $\mathcal{M}$ as intended, to represent a network's group-level structure, rather than simply grouping together vertices with similar degrees as the SBM tends to do in practice to create a heavy-tailed degree distribution.

[10] Note: being an edge count does not necessarily imply that $\mathcal{M}_{rs} \leq n_r n_s$, since the DC-SBM generates multigraphs.

4.2.1 Properties of DC-SBM graphs

At a high level, DC-SBM graphs have the following properties:

- By default, an *undirected multigraph*, generalizable to directed edges; obtain a simple network by collapsing multi-edges.
- *Modular* connectivity, governed by the sizes of the c groups and the mixing matrix $\mathcal{M}$.
- The *degree structure* is specified by $\vec{k}$.
- Diameter and mean geodesic distance are typically $O(\log n)$, unless $\mathcal{M}$ has unusual structure.
- The clustering coefficient tends toward $C = O(1/n)$, but depends strongly on group sizes, their internal density specified by $\mathcal{M}$, and the variance of $\vec{k}$.
- The largest connected component (LCC) tends to be $O(n)$ when G is not too sparse.

4.2.2 Generating a DC-SBM network

In the first-level of the DC-SBM, groups act like "super" nodes, and so we can define the "degree" of the rth group as

$$\kappa_r = \sum_s \mathcal{M}_{rs} = \sum_i k_i \delta_{z_i,r} \ , \tag{8}$$

that is, the sum of the degrees of all the nodes in group r. The network at this level is then very much like a (loopy multigraph) configuration model, with a degree sequence $\vec{\kappa}$.

In the model's second level, where we expand the rth group to be a collection of n_r nodes, we now need to allocate the κ_r connections of the group "node" to specific group members, in a way that respects their degrees. Consider one such connection. The probability that it lands on the ith node is given by what we can call its *propensity*

$$\gamma_i = \frac{k_i}{\kappa_{z_i}} \ , \tag{9}$$

which is just the fraction of the group's total degree that belongs to node i.

Given these parameters, the value of each element of the adjacency matrix is

$$\forall_{i>j} \qquad A_{ij} = A_{ji} = \text{Poisson}\big(\gamma_i \gamma_j \mathcal{M}_{z_i,z_j}\big) \ , \tag{10}$$

which implies that the expected number of edges running between i and j is $\gamma_i \gamma_j \mathcal{M}_{z_i,z_j}$. Intuitively, the product $\gamma_i \gamma_j$ plays a similar role here to the degree product $k_i k_j$ in the configuration model. A key difference, however, is that γ_i is a fraction rather than a count, and so the product $\gamma_i \gamma_j$ serves

to pick out of the edge count $\mathcal{M}_{z_i,z_j}$ the number of edges we allocate to i, j.

Given Eq. (10) and a choice of parameters $\theta = (c, \vec{z}, \vec{k}, \mathcal{M})$, we draw a simple graph from the ensemble as follows, in $\Theta(n^2)$ time:

1. compute the group-level degrees κ_r by Eq. (8)
2. compute the node-level propensities γ_i by Eq. (9)
3. initialize an empty graph G with n nodes
4. for each pair $i > j$, draw a Poisson random variable r with mean $\gamma_i\gamma_j\mathcal{M}_{z_i,z_j}$, as in Eq. (10)
5. if $r > 0$, add the undirected, unweighted edge (i, j) to G.

The last step collapses the multi-edges to create a simple network. If a multigraph is desired, replace this step with assigning $A_{ij} = A_{ji} = r$. This simplification step will induce small differences in the generated degree sequence and the specified one, but usually, these differences are negligible. The DC-SBM has more parameters than the SBM, which gives it even greater flexibility in producing a broad variety of large-scale network patterns. In total, there are $1 + 2n + \Theta(c^2)$ parameters, which correspond to the number of values required to specify the number of groups c, a group label for each node $\vec{z}$, the node degrees $\vec{k}$, and the groupwise edge counts $\mathcal{M}$, respectively.

4.2.3 Planted partitions in the DC-SBM

Like the SBM, we can define a planted partition model for the DC-SBM and use it to explore how degree and modular structures interact. As with the SBM, our goal is to develop a one-parameter model that interpolates between more or less strongly modular structure.

To do this, we'll use a slightly different strategy than we did for the SBM. This time, we will define two mixing matrices, one for perfectly assortative communities and one for a random mixing (as in a configuration model), and then to combine these matrices using a parameter λ, like this

$$\mathcal{M}_{rs} = \lambda\,\mathcal{M}_{rs}^{\text{assort.}} + (1-\lambda)\,\mathcal{M}_{rs}^{\text{random}} \quad . \tag{11}$$

When $\lambda = 1$, we use only the assortative planted partition, and as $\lambda \to 0$, we mix in increasing amounts of configuration model randomness, until, when $\lambda = 0$, all of the structure is entirely that of a random graph with specified degrees. The random mixing matrix should have the form of a configuration model for each pair r, s, and hence we say

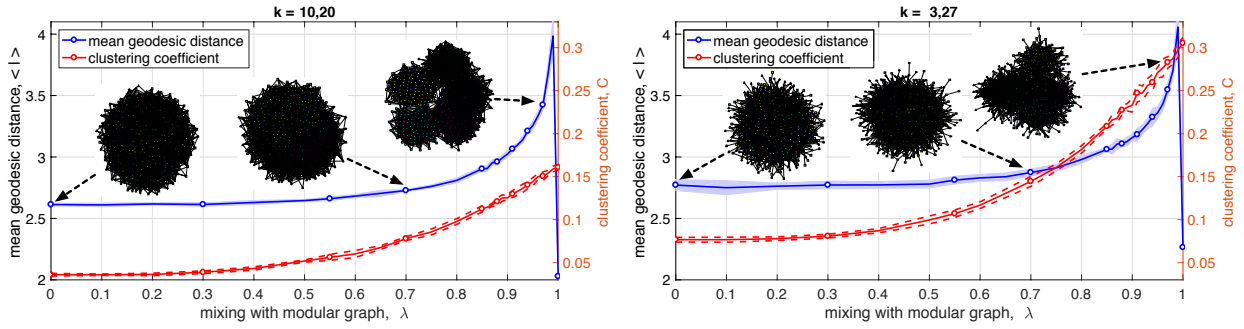
$$\forall_{rs}\ \mathcal{M}_{rs}^{\text{random}} = \kappa_r\kappa_s/2m \quad .$$

While for the perfectly assortative communities, we set the rth community's degree to be entirely within-group:

$$\mathcal{M}^{\text{assort.}} = \begin{pmatrix} \kappa_1 & 0 & \cdots & 0 \\ & \kappa_2 & \ddots & \vdots \\ & & \ddots & 0 \\ & & & \kappa_c \end{pmatrix} , \tag{12}$$

and set the group sizes to be all equal, $n_r = n/c$. Now, we need only to specify the degree structure $\vec{k}$. The values k_i can be drawn from any integer probability distribution we like, e.g., a discrete exponential distribution or a discrete power-law distribution (or anything in between). The one constraint on every degree value is that the maximum degree cannot exceed the size of its group, meaning $k_{\max} \leq n/c$, as otherwise, its propensity score would exceed 1, and we would have the impossible situation of $k_i > \kappa_{z_i}$.

Planting partitions for simulation experiments. The value of λ naturally varies from $[0, 1]$, and across this range, the strength of the modular structure will vary considerably. In this version of our experiment, we will explore how the same network measures vary as a function of λ, for two different choices of a degree distribution: one with low variance, and one with very high variance. For example, consider a $c = 5$ group network on $n = 500$ nodes, where we either (i) set each degree $k_i = 10$ or 20 with equal probability (lower variance), or (ii) set each degree $k_i = 3$ or 27 with equal probability (higher variance). In either case, the mean degree $\langle k \rangle = 15$ is the same, so all that's different between the two settings is the variance, and the structure it induces. Now, as in the SBM experiment, we measure the mean geodesic distance $\langle \ell \rangle$ (lefthand y-axis) and the clustering coefficient C (righthand y-axis) as a function of how strong the community structure is λ, for each distribution.



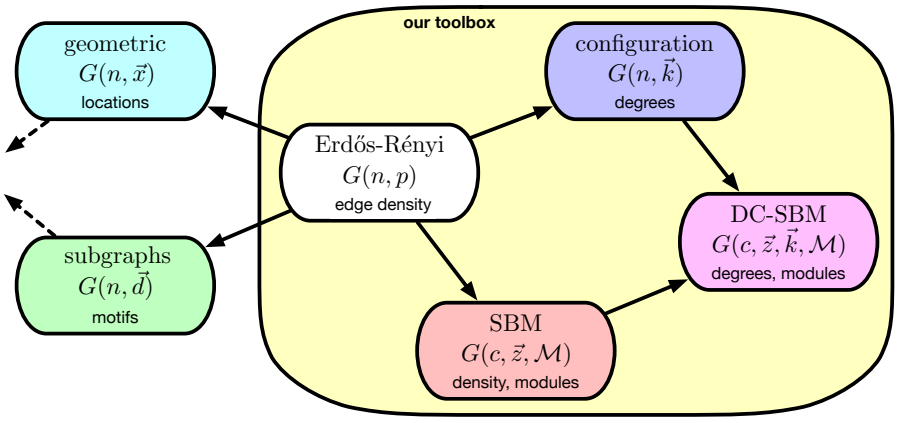
The overall pattern for both degree distributions is similar to what we saw in the SBM: when connections are mostly random (modular structure is weak $\lambda < 0.3$), both network measures are fairly stable, and both increase as the modular structure strengthens beyond this point. A clear

difference, however, is that the clustering coefficient C is far more sensitive to the change in variance than is the average path length $\langle \ell \rangle$: it starts off a little higher, even in the purely random graph extreme ($\lambda = 0$), and increases faster and farther as the network approaches the purely assortative community extreme ($\lambda = 1$). Intuitively, the reason is that the higher degree nodes in the higher-variance case will tend to group together, by virtue of their larger degree giving them many more ways they could group together, and this naturally generates more triangles than when edges are distributed more equitably.

One last thing is worth noting about these experiments: even when a large majority of a graph's edges are drawn from the perfectly assortative planted partition $\mathcal{M}^{\text{assort.}}$, the path-length structure of the network is largely driven by the presence of the $\mathcal{M}^{\text{random}}$ connections. It's not until truly extreme values of $\lambda > 0.9$ do we really see the assortativity dominant the randomness. This pattern is a powerful reminder that it does not take much randomness to make a network look like a random graph, at least in terms of its path lengths.

5 Taking stock of random graph models

With the SBM and DC-SBM, and including the Erdős-Rényi and configuration (or Chung-Lu) models, we now have four random graph models in our toolbox. Together, these models specify ensembles of random graphs based on assumptions of randomness plus parameterized (i) edge density, (ii) degree structure, or (iii) modules. And, these models are generalizations of each other. Both the configuration model and the SBM can be viewed as different kinds of generalizations of the Erdős-Rényi model, one to a setting with degree structure and the other to a setting with modules. Combining these two models is what gives us the DC-SBM. We can visualize these relationships as shown below, where the large bubble covers the models we have learned about so far.



It's easy to imagine adding new models to this picture, where model i connects to model j if the latter is a generalization of the former. For example, in a *random geometric graph*, nodes have locations in a metric space $\vec{x}$ and the probability that a pair i, j is connected depends on their distance $d(x_i, x_j)$.[11] This model also generalizes ER graphs, but does so in a different direction than either the configuration model or the SBM. Further elaborations of this model might add assumptions about degree structure or modules, thereby generalizing the configuration model and the SBM, respectively. Or, we could design a random subgraph model in which we specify the motifs of different types that each node is part of $\vec{d}$.[12] This model would generalize ER graphs in still another direction. The point here is to illustrate that there are many ways to build on top of the basic random graph models in our toolbox, by adding new structural assumptions. Doing so is an active area of research today.

Returning to our four random graph models, we can now take stock of the degree to which each is able to capture or reproduce the kinds of structural patterns often found in real-world networks. The table below presents the story so far for each of the models, for measures like the edge density ρ, the degree distribution $\Pr(k)$, the diameter $\ell_{\max}$ or mean geodesic distance $\langle \ell \rangle$, the clustering coefficient C, reciprocity r (directed networks only), and the existence of a very large connected component. Clearly, random graphs with modular structure, and especially the DC-SBM, do a pretty good job of capturing many of the observed network patterns.

measure	pattern	edge density	degrees	modules, density	modules, degrees
	empirical	*Erdős-Rényi*	*configuration*	*SBM*	*DC-SBM*
edge density ρ	sparse	specified	specified	specified	specified
degrees $\Pr(k)$	heavy tailed	Poisson	specified	Poisson mixture	specified
diameter $\ell_{\max}$	$O(\log n)$	$O(\log n)$	$O(\log n)$	typically $O(\log n)$	typically $O(\log n)$
clustering C	social: many non-social: few	$O(1/n)$	$O(1/n)$	depends on $\mathcal{M}$	depends on $\mathcal{M}$
reciprocity r	high	$O(1/n)$	$O(1/n)$	depends on $\mathcal{M}$	depends on $\mathcal{M}$
large component	$\Theta(n)$	if $\langle k \rangle > 1$	if $\langle k^2 \rangle - 2\langle k \rangle > 0$	depends on $\mathcal{M}$	depends on $\mathcal{M}$

6 Supplemental readings

1. Chapters 14.1 and 14.4 in *Networks*

[11] See Penrose, *Random Geometric Graphs*, Oxford University Press (2003).
`https://books.google.com/books?hl=en&lr=&id=RHvnCwAAQBAJ`

[12] As in Karrer and Newman, *Phys. Rev. E* **82**, 066118 (2010). `https://arxiv.org/abs/1005.1659`

7 Maximizing modularity in the Amazon co-purchasing network

The modularity function Q (Eq. (4)) gives a measure of assortative structure on enumerative vertex attributes. When labels on the vertices are given, we can calculate Q and quantify how well edges fall within the specified groups relative to the (loopy multigraph) configuration model. Different labelings give different scores, and the higher the score, the "better" the communities. But, if we are not given the groups, can't we just search over all possible node labelings to find a partition that produces a high or even the highest possible modularity score? Yes. Yes, we can.

Recall from Eq. (4) that the modularity score Q can be written as a sum over the differences between the fraction of edges that join vertices in the uth module, denoted e_{uu} and the expected fraction, given by a_u^2. A "good" partition—with Q closer to unity—represents groups with many more internal connections than expected at random; in contrast a "bad" partition—with Q closer to zero—represents groups with no more internal connections that we expect at random. Thus, Q measures the cumulative deviation of the internal densities of the identified modules relative to a simple random graph model.[13]

The number of possible partitions is for n vertices is a Bell number of the second kind B_n, which grows super-exponentially with n. It is generally infeasible to exhaustively search this space, and it is known that maximizing Q is NP-hard. In practice, however, many search heuristics perform extremely well on real-world networks. Examples of these strategies include greedy agglomeration, mathematical programming (linear and quadratic), spectral methods, extremal optimization, simulated annealing and various kinds of sampling techniques like Markov chain Monte Carlo. The partition returned by these heuristics is only likely to be a high-scoring partition, and not the best possible partition. These algorithms also say nothing about how many alternative, equally-good partitions they did not return (spoiler: there are often exponentially many).

Many modularity maximizing algorithms run slowly on real-world networks, taking time $O(n^2)$ or slower; a few run very quickly, for instance, taking time $O(n \log^2 n)$ and can thus be applied to very large networks of millions or more vertices. Because they are heuristics, of course, faster algorithms must make more aggressive assumptions about how to search the partition space and thus are less likely to find the globally optimum partition.

To illustrate the kind of things that come out of community detection in real-world data, and modularity maximization in particular, here's a fun example of applying a greedy agglomeration algorithm to a co-purchasing or "recommender" network from `amazon.com`. Amazon sells a wide

[13] It should be pointed out that a high value of Q only indicates the presence of significant deviations from the null model of a random graph. A large deviation could mean two things: the network exhibits (i) genuine modular structure in an otherwise random graph, or (ii) other non-random structural patterns that are not predicted by a random graph. See Section 8 at the end of this file for a brief discussion of this subtlety.

variety of products, particularly books and music, and on each item A's webpage, Amazon lists several other items most frequently purchased by buyers of A. The figure below shows an example of this information.

These recommendations form a directed network in which vertices are items and an item A points to an item B if B was frequently purchased by buyers of A. In our analysis, we convert this directed network into an undirected one. The data we use are from a 2003 snapshot of Amazon's recommendation network, in which we focus on the largest component, which contains $n = 409,687$ vertices and $m = 2,464,630$ edges.

Applying the greedy agglomeration algorithm to this network produces some interesting results. The resulting dendrogram is far too large to draw (since it contains nearly half a million joins), but Fig. 1 (left) shows the modularity score Q over the sequence of merges made by the algorithm. The

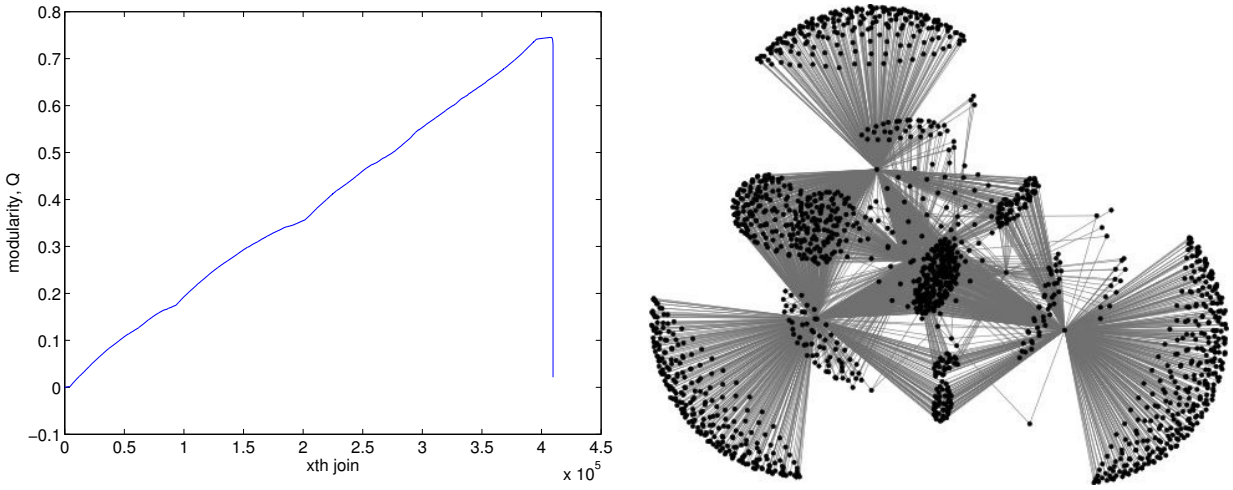


Figure 1: (left) The modularity score Q over the course of the agglomerative algorithm, with a maximum at $Q = 0.745$ for 1684 communities. (right) A visualization of the community structure at maximum modularity, where each vertex is a "super" node containing many items.

Rank	Size	Description
1	114538	General interest: politics; art/literature; general fiction; human nature; technical books; how things, people, computers, societies work, etc.
2	92276	The arts: videos, books, DVDs about the creative and performing arts
3	78661	Hobbies and interests I: self-help; self-education; popular science fiction, popular fantasy; leisure; etc.
4	54582	Hobbies and interests II: adventure books; video games/comics; some sports; some humor; some classic fiction; some western religious material; etc.
5	9872	classical music and related items
6	1904	children's videos, movies, music and books
7	1493	church/religious music; African-descent cultural books; homoerotic imagery
8	1101	pop horror; mystery/adventure fiction
9	1083	jazz; orchestral music; easy listening
10	947	engineering; practical fashion

Table 1: The 10 largest communities in the `amazon.com` co-purchasing network.

maximum value is $Q = 0.745$, which is fairly large as these kinds of calculations go[14] This value occurs when there are 1684 communities remaining, each of which has an average size of 243 items. Fig. 1 (right) gives a visualization of the community structure at this point, where each vertex

[14]Empirically, the maximum modularity score $Q_{\max}$ appears to correlate with n.

represents a group of items. There is plenty of interesting structure to be seen here, including the major communities, smaller "satellite" communities connected to them, and "bridge" communities that connect two major communities with each other.

If we examine the contents of the largest communities in the network, we find that they tend to consist of items (books, music) in similar genres or on similar topics. Table 1 gives informal descriptions of the ten largest communities, which account for about 87% of the entire network. The remainder are generally small, densely connected communities that represent highly specific co-purchasing habits, e.g., major works of science fiction (162 items), music by John Cougar Mellencamp (17 items), and books about (mostly female) spies in the American Civil War (13 items).

8 Trouble with community detection

Finally, there are many caveats about community detection. Here are three.

The first ill omen is the observation that despite a tremendous amount of activity on identifying clusters in networks, for the most part, we have no good explanations of what social, biological or technological processes should cause clusters to exist in the first place, or what functional role they play in those systems. (Presumably, these two things are related.) The best we have is a verbal argument due to Herbert Simon for why modular designs a good way to construct complex systems. The argument goes like this:

Suppose we are tasked with assembling many small, delicate mechanical parts into a beautiful Swiss watch, a masterpiece of complexity. And, suppose that we are interrupted every now and then, perhaps by the arrival of email or the need to have a snack, and than if we are interrupted before we have fully completed the assembly of a single watch, the partially assembled watch falls to pieces and we must start afresh after our break is over. If, on average, we experience one or more interruptions during the long process of assembling a watch, we will never produce very many watches. Now, however, consider the advantage of a modular design to the watch. In this case, the final watch is produced by combining smaller groups of parts or modules. If these modules are themselves stable products, then now we can tolerate interruptions and still produce watches because at worst, we only lose the work necessary to build a piece of the whole, rather than the whole itself. Thus, it seems entirely reasonable to expect that, in the face of "interruptions" of all different kinds, complex systems of all kinds will exhibit a modular or even hierarchical organization.

A second ill omen comes from recent observations that many of the popular global techniques for identifying modules in networks exhibit two undesirable properties: (i) *resolution limits* and (ii) *extreme degeneracies*. The first observes that the technique cannot detect intuitively coherent communities that are smaller than some preferred scale. In general, the mathematics showing

the appearance of this behavior always points to the same root cause, which is the random-graph assumptions built into the techniques' score functions. The second observes that there are an exponential number of alternative, high-scoring partitions, which makes both finding the best partition computationally difficult (in some cases, NP-hard) and interpreting the particular structure of the best partition ambiguous. These two properties make it somewhat problematic apply network clustering techniques in contexts where what we want is to find the *true* but unknown modules. For much more detail about these issues, see Good et al., "The performance of modularity maximization in practical contexts." *Physical Review E* **81**, 046106 (2010).

A third ill omen comes from the world of spatial clustering, a very old and still very active field. Many of the techniques for finding clusters in networks have antecedents in this domain. The general problem can be framed like so: we are handed a data set $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$ and our task is to identify the "natural" clusters of these data such that points in a cluster are closer (via some measure of distance) to each other than they are to all the other points.

However, in a 2002 paper titled "An Impossibility Theorem for Clustering", Jon Kleinberg showed, using an axiomatic approach, that no clustering function f can simultaneously satisfy three highly reasonably and practically banal criteria:

1. *Scale Invariance*: For any distance function d and any $\alpha > 0$, $f(d) = f(\alpha \cdot d)$. Intuitively, this requirement says that if we rescale all the distances between points by some constant factor α, the best clustering should stay the same.

2. *Richness*: Range(f) is equal to the set of all partitions of $\mathbf{x}$, which simply requires that all possible partitions of our data be potentially okay candidates.

3. *Consistency*: Let d and d' be two distance functions. If $f(d) = \Gamma$, and d' is a Γ-transformation of d, then $f(d') = \Gamma$. In other words, if a particular distance function d yields the clustering Γ, then if we make a new distance function d' that reduces all of the distances arising within clusters while increasing all those distances between clusters—that is, we make the clusters internally more dense and move them further away from each other—then we should get the same clustering Γ under d'.

Thus, any practical solution to identifying clusters in spatial data can, at best, only have two of these properties. One consequence of this fact is that we can categorize clustering techniques into broad categories, based on which single criteria they violate. It's unknown if a similar impossibility theorem exists for network clustering.[15]

[15]It is not hard to see that the second and third criteria have natural analogs in the context of network clustering. The first criteria, however, has no clear analog.