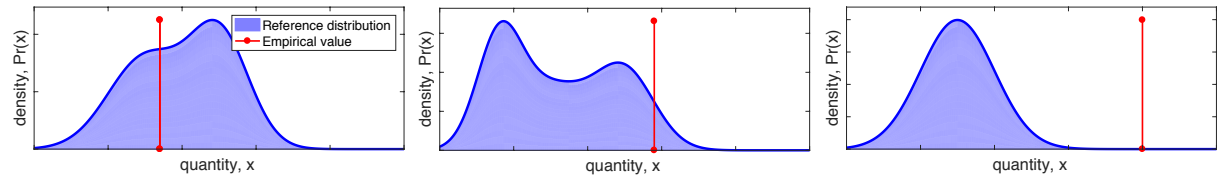
1 Random graphs and reference distributions

Suppose that we've calculated one or more descriptive statistics for a network G. For example, we've calculated the mean degree $\langle k \rangle$, or the degree distribution $\Pr(k)$, or the (global) clustering coefficient C, or the mean geodesic distance $\langle \ell \rangle$. How do we interpret the values these statistics take? Which values are "big" or "small," "typical" or "unusual"?

Random graph models provide one way to answer these questions, by providing a reference point x or a whole reference distribution $\Pr(x)$ for assessing an empirical measurement x.[1]

For example, let's say we obtain a clustering coefficient of C_* for a gene regulatory network G. Using a random graph model M that is parameterized to be *similar* to G in some way, e.g., similar edge density, similar degree distribution, etc., we can use M to produce many "synthetic" (computer generated) networks, and calculate each of their clustering coefficients. These synthetic values represent a *reference distribution* $\Pr(C)$, which encodes how often M will produce networks with different values of C. If the empirical value C_* falls in the "middle" of $\Pr(C)$ (left-hand panel below), then we may conclude that the assumptions that underlie M are sufficient to explain the value of C_* in G. Or, if C_* tends to be on the upper end of the reference distribution, we may conclude that C_* is somewhat unusual but not unrealistic (middle panel). Or, if C_* falls far outside $\Pr(C)$, we may conclude that the assumptions of M are insufficient to explain the value of C_* (right-hand panel).



Here, we will learn about three classes of random graph models, which represent a sequence of increasing structure imposed on the basic property of randomness.[2]

1. **Erdős-Rényi (ER) random graph**:
   the simplest random graph, in which edges are independent and identically distributed with probability p, which makes them very homogeneous.

[1] As mathematical models, random graphs are also extremely useful for analyzing the structural properties of networks in ways that are impractical for real-world networks, whose structure is messier and more complicated. Random graph models provide a way to build our intuition about how certain descriptive statistics should behave, under different assumptions about the structural randomness.

[2] A fourth class of random graph, which we will not discuss in detail, are spatial random graphs, in which the probability that an edge (i, j) exists depends on the pairwise distance $d(i, j)$ of the two nodes in some latent or explicit space, e.g., a shorter distance correlates with a higher connection probability.

2. **Configuration model** (and the Chung-Lu model):
   in which the random graph is conditioned on having a specified degree sequence $\vec{k}$ or a specified degree distribution $\Pr(k)$, which makes them more heterogeneous.

3. **Modular random graphs**:
   in which the random graph is conditioned on having a specified modular structure θ, and possibly also a specified degree sequence $\vec{k}$, etc.

A deeper discussion of the third class is left until Lecture 5, when we explore the idea of community structure. Below, we describe the ER random graph and and the configuration model, and then show how they can be used to test whether density or the degree distribution are sufficient to explain certain structural patterns of real-world networks.

2 The Erdős-Rényi random graph

The **Erdős-Rényi random graph model**, sometimes called an ER graph or simply $G(n,p)$, is the oldest and simplest random graph model. It was most prominently studied by mathematicians Paul Erdős (1913–1996)[3,4] and Alfréd Rényi (1921–1970).

An ER graph, denoted $G(n,p)$, is a simple graph with n vertices, where p is the probability that each simple edge exists, $(i,j) \in E$.[5] Hence, edges are independent and identically distributed (iid) random variables, and the size of the network and the edge probability fully specify the model.

In an adjacency matrix representation, we say

$$\forall_{i>j} \qquad A_{ij} = A_{ji} = \begin{cases} 1 & \text{with probability } p \\ 0 & \text{otherwise .} \end{cases}$$

(Do you see why we need the restriction $i > j$?)

Mathematically, $G(n,p)$ defines an *ensemble*, or a parametric distribution over all graphs $\Pr(G \,|\, p)$. Choosing p parameterizes this distribution, and we draw any particular graph G with probability $\Pr(G \,|\, p)$. This fact means that when we describe the properties of $G(n,p)$ below, we are typically describing *average* properties, and individual instances will tend to fluctuate around these values.[6]

[3] `http://xkcd.com/599/` and `https://en.wikipedia.org/wiki/Erdos_number`

[4] "A mathematician is a machine for turning coffee into theorems."

[5] A closely related random graph model is $G(n,m)$, which places exactly m edges on n vertices, meaning that m is no longer a random variable as it is under $G(n,p)$.

[6] Counter-intuitively, if $0 < p < 1$, there is a non-zero probability of generating *any* graph of size n from $G(n,p)$.

The extremal values of p produce extreme graphs: $p = 1$ produces a fully connected (dense) graph, or a clique, because all $\binom{n}{2}$ possible edges exist; in contrast, $p = 0$ produces an "empty" graph, because no edges exist. But, most real-world networks are what we call "sparse," meaning they have a mean degree $\langle k \rangle$ that is very small compared to n. For $G(n, p)$ to produce a sparse graph, we need only set $p = c/(n-1)$, for a desired mean degree $\langle k \rangle = c$, e.g., $c = 2$ or $c = 4$.

2.1 Properties of ER graphs

ER random graphs are not considered realistic models of networks, but they are a useful baseline random graph model. Nevertheless, for a few descriptive statistics, randomness and low density are sufficient to produce interesting structure. At a high level, ER 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 "homogeneous," and is entirely driven by the edge "density" parameter p; edges are independent and identically distributed (iid).
- The degree distribution $\Pr(k)$ is a Poisson distribution with mean $c = p(n-1)$, which is a low-variance distribution.
- The diameter and mean geodesic distance are $O(\log n)$, making ER graphs "small-world-like," so that short paths exist between most pairs of nodes.
- The clustering coefficient is $C = O(1/n)$, meaning there are very few triangles.
- The largest connected component (LCC) is proportional to the network size $O(n)$ when $c > 1$, and is vanishingly small, containing $O(1)$ nodes, when $c < 1$ (a phase transition at $c = 1$).

2.2 Generating a $G(n, p)$ network

Once n and p have been chosen, the most straightforward way to draw a graph from the ensemble is to loop over pairs of nodes:

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
3. if $r \leq p$, then add the undirected edge (i, j) to G.

Basically, this procedure flips $\binom{n}{2}$ coins, each with bias p. As a result, generating a network in this way takes $\Theta(n^2)$ time, and is computationally expensive when $n > 10^4$ or so.[7]

[7] It's possible to draw from $G(n, p)$ in only $O(n+m)$ time, using an algorithm that is only slightly more complicated than this one. See Batagelj and Brandes, *Phys. Rev. E* **71**, 036113 (2005) `http://vlado.fmf.uni-lj.si/pub/networks/doc/ms/rndgen.pdf`.

2.3 Mean degree

There are two simple ways to calculate the mean degree. On the one hand, note that for any particular node i, there are $n-1$ possible other nodes j, and each of those pairs i, j is connected iid with probability p. Hence, the mean degree is

$$\langle k \rangle = c = \sum_{j=1}^{n-1} p = p(n-1) \ . \tag{1}$$

On the other hand, there are $\binom{n}{2}$ pairs of nodes, and each of them exists with probability p. Hence, the expected number of edges is

$$\langle m \rangle = \sum_{i=1}^{n} \sum_{j>i} p = p\binom{n}{2} = p\left(\frac{n(n-1)}{2}\right) \ . \tag{2}$$

Recall that the mean degree is $\langle k \rangle = 2m/n$. Plugging Eq. (**??**) into this expression yields

$$\begin{aligned} \langle k \rangle &= \frac{2}{n}\left(p\left(\frac{n(n-1)}{2}\right)\right) \\ &= p(n-1) \ . \end{aligned}$$

2.4 Degree distribution

Because edges in $G(n,p)$ are iid random variables, the entire degree distribution has a simple form: it's a Binomial distribution. For a node to have degree k, when we flip the $n-1$ coins that determine which of the other $n-1$ nodes it connects to, we have to get exactly k successes and $n-1-k$ failures. And, there are $\binom{n-1}{k}$ ways to get those k successes. Hence, the degree distribution is

$$\Pr(k) = \binom{n-1}{k} p^k (1-p)^{n-1-k} \ , \tag{3}$$

with parameter p for $n-1$ independent "Bernoulli" trials (coin flips).

Recall that for $G(n,p)$ to be sparse, we set $p = c/(n-1)$, where c is a small positive number for the mean degree. In this parameter regime, in which p is small and G is sparse, the Binomial is well-approximated by a Poisson distribution[8]

$$\Pr(k) = \frac{c^k}{k!} \mathrm{e}^{-c} \ . \tag{4}$$

A Poisson distribution has mean and variance c, runs only over the natural numbers, and is

[8] It's a nice little calculation to derive this answer yourself. Start off by applying a first-order Taylor expansion of the logarithm to simplify $\ln\left[(1-p)^{n-1-k}\right]$, and then let n be large, and simplify.

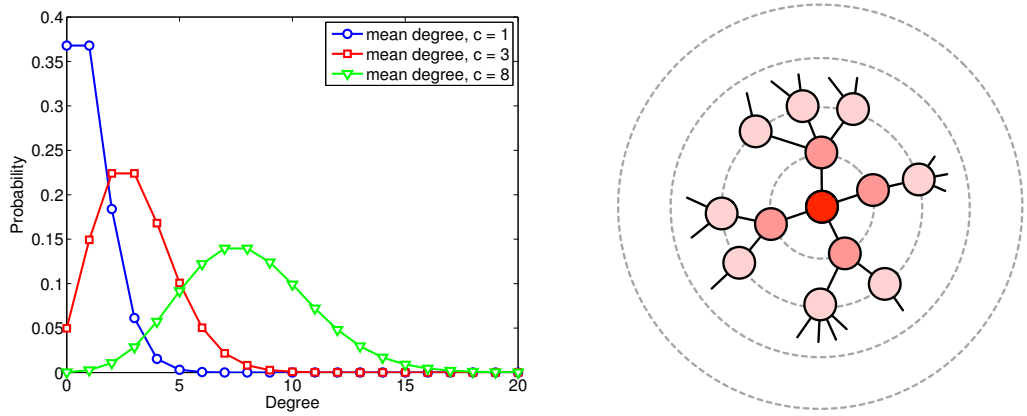


Figure 1: (left) Poisson degree distributions for a few choices of mean degree $c = \{1, 3, 8\}$, showing that as the mean degree increases, the distribution move away from the minimum value of $k = 0$, but remains tightly concentrated around the mean c. (right) A schematic illustrating a "locally tree-like" structure, in which for most nodes, very few or none of their neighbors are connected (few or no loops of short length).

slightly asymmetric. The left-hand panel of Figure 1 shows examples of Poisson distributions, with $c = \{1, 3, 8\}$.

Implications. Recall from Lecture 2 that most real-world networks have heavy-tailed distributions. In contrast, a Poisson distribution is not heavy tailed—it falls off too quickly to produce nodes with degrees much higher than the mean. Hence, ER graphs are poor models of real-world networks.

2.5 Motifs, reciprocity, and the clustering coefficient

The fact that edges in an ER graph are iid makes it easy to calculate the probability of many motifs.

For instance, if we generalize $G(n, p)$ to a directed case, in which both (i, j) and (j, i) each occur with probability p, then the reciprocity coefficient is

$$r = \frac{\text{(number of reciprocated links)}}{\text{(number of links)}} = \frac{(n^2 - n)p^2}{(n^2 - n)p} = p \ , \tag{5}$$

where $n^2 - n$ is the number of possible directed edges (without self-loops), and in the numerator, we require that both (i, j) and (j, i) exist, which happens with probability p^2.

The calculation for the clustering coefficient, on the undirected ER graph, follows a similar logic:

$$C = \frac{\text{(number of triangles)}}{\text{(number of connected triples)}} \propto \frac{\binom{n}{3}p^3}{\binom{n}{3}p^2} = p \ . \tag{6}$$

Implications. When a random graph G is sparse, both $r = O(1/n)$ and $C = O(1/n)$, meaning that the density of these motifs shrinks toward zero in the limit of large graphs. In other words, ER graphs have very few reciprocated links or triangles.[9] The latter fact has a crucial implication: sparse ER graphs are *locally tree-like*, meaning that if we build a tree outward from any particular node in the graph, we rarely encounter a "cross edge" that links between two branches of the tree. The right panel in Figure 1 illustrates this idea.

2.6 Diameter and mean geodesic path length

The locally tree-like structure of a sparse ER graph implies that, if we were to grow that tree outward from any particular node, e.g., using Dijkstra's algorithm or a breadth-first search tree, each new node the tree touches would itself branch out to touch roughly $c-1$ new nodes (do you see why $c-1$ rather than c?). After ℓ such steps, the tree will have roughly touched $(c-1)^\ell$ nodes. The tree will stop growing when it has touched all n nodes. From these observations, we can derive an estimate of a sparse ER graph's diameter:

$$\begin{aligned} (c-1)^{\ell_{\max}} &= n \\ \ell_{\max} \log(c-1) &= \log n \\ \ell_{\max} &= \log_{c-1} n \\ &= O(\log n) \ . \end{aligned} \tag{7}$$

Hence, the diameter of a sparse ER graph is $O(\log n)$. So too is the mean geodesic distance $\langle \ell \rangle$, by a similar argument. This analysis can be made fully rigorous, but we omit the details here.

Implications. The logarithmic diameter of ER graphs implies that information or activation will tend to spread quickly across the network, because a signal needs to only cross a chain of $O(\log n)$ edges to reach *anywhere* in the network. That is, for any pair of nodes i, j, there exists a "short" path between them. If you recall how fast the binary search algorithm is at finding a target element within a sorted array, traversing an ER graph is similarly fast—the length of the longest chain of steps in both cases is $O(\log n)$.

This fact implies that the existence of such short paths can be explained by randomness and sparse connectivity alone, without needing to refer to other structural patterns. In this sense, short paths

[9] In fact, Eq. (**??**) generalizes to a loop of $1 < \ell \leq n$ edges. Do you see how? As a result, ER graphs have few loops of any size.

are "automatic" if a network has some amount of randomness in its connectivity structure. This property can be useful for spreading signals quickly, e.g., to synchronize different parts of a system, but also problematic if the goal is to decouple or insulate different parts of the network.

2.7 What $G(n, p)$ graphs look like

In this section, we'll build some intuition for the shape of ER graphs, and how that structure varies with mean degree c. The figure below shows simple visualizations[10] for networks of varying sizes $n = \{10, 50, 100, 500, 1000\}$ and mean degrees $c = \{0.5, 1.0, 2.0, 4.0\}$. And, to avoid visual clutter, all singleton vertices (degree $k = 0$) are omitted.

In the "ultra-sparse" case of $c < 1$, the networks are composed entirely of small or very small components, mostly perfect trees, and the largest connected component (LCC) contains $O(1)$ nodes. At $c = 1$, many of the little trees have begun to connect, forming a wide variety of larger components, most of which are still trees. However, for $c > 1$, we see the emergence of a very large connected component (in fact, a *giant component*; see the Appendix below), which will contain $O(n)$ vertices. Despite its size, when $c = 2$, we still see some "dust" components, i.e., the same small trees we saw for $c < 1$, around the giant component. In this regime, the LCC displays some interesting structure, being locally tree-like but exhibiting long cycles punctuated by tree-like whiskers.

Finally, when the mean degree is fairly large (here, $c = 4$), the LCC has gobbled up nearly every node and has the appearance of a big hairball, with little apparent structure.[11] Although one cannot see it in these visualizations, the structure is still locally tree-like.

[10] The positions of nodes on the page were assigned via a standard spring-embedder algorithm like the Fruchterman-Reingold force-directed layout algorithm, in this case, implemented in the `yEd` software program.

[11] Visualizations of such networks are sometimes called *ridiculograms*, reflecting the fact that much of the meaningful structure is obscured by the overprinting of the edges on each other and on the nodes. Such figures are surprisingly common in the networks literature.

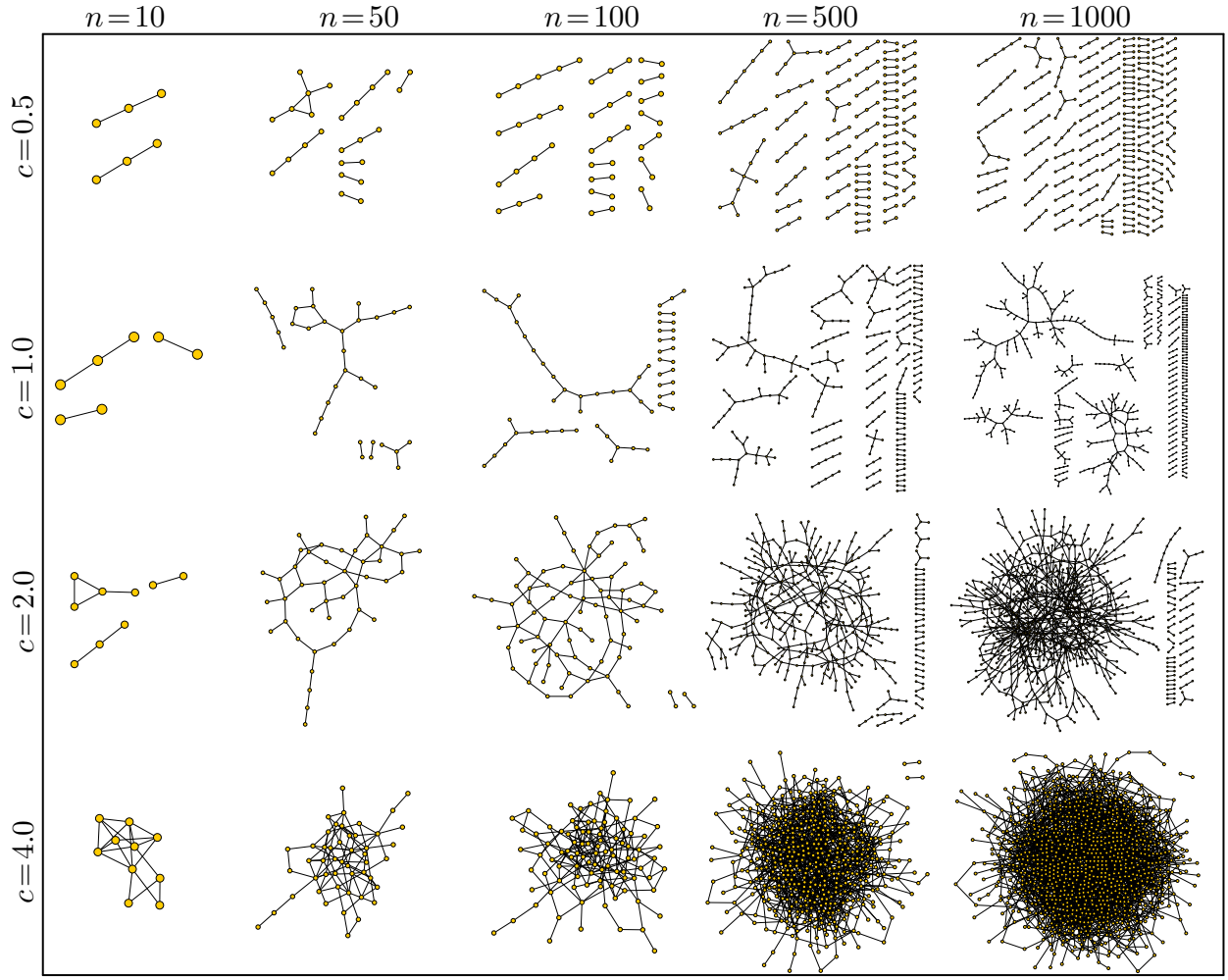


Figure 2: Visualizations of sparse ER graphs $G(n, p)$ for increasing size n and mean degree c.

3 The configuration model

Suppose that instead of merely setting the density of edges p, we fixed the entire degree sequence $\vec{k}$ of a random graph. The **configuration model**, and its close relative the Chung-Lu model, defines such an object: a random graph with a specified degree sequence. These networks are richer in their structural variety than Erdős-Rényi random graphs, because they have more realistic degree structures—indeed, we can even set $\vec{k}$ to be exactly the degree sequence of a real-world network G.

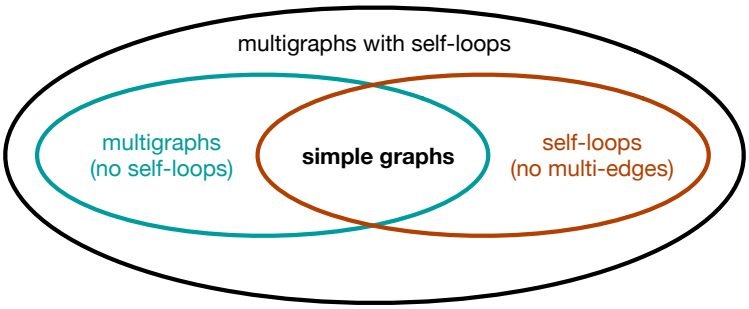
As an undirected adjacency matrix, such a random graph essentially says

$$\forall_{i>j} \qquad A_{ij} = A_{ji} = \begin{cases} 1 & \text{with probability } \propto k_i k_j \\ 0 & \text{otherwise} \end{cases},$$

meaning that the probability that i, j are connected depends on their joint degrees k_i and k_j, rather than the constant value $p = 2m/n(n-1)$ of an ER graph.

And, just as the Erdős-Rényi model allows us to test whether some descriptive statistic's value can be explained by the density of edges alone (plus randomness), the configuration model lets us say whether or not it can be explained by the degree distribution alone (plus randomness).

There are, in fact, *four* configuration models for undirected networks,[12] depending on the target graph properties; specifically, self-loops (yes or no) and multiedges (yes or no). If neither are permitted, then we are dealing with the space of random simple graphs. We note that nearly all of the statements in this section also hold for a configuration model with directed edges, where we separately specify $\vec{k}^{\text{out}}$ and $\vec{k}^{\text{in}}$, but for simplicity we focus on the undirected case.



3.1 Properties of configuration model graphs

Configuration models generate more realistic networks because at least they can capture the heavy-tailed nature of real-world degree distributions, which we saw in Lecture 2. For this reason, they find uses in a variety of network analyses and models, including as mathematical models, as a substrate for simulating network dynamics, and as "null" models for constructing reference distributions for empirical quantities. At a high level, configuration model random graphs have the following properties, which are similar in many ways to those of an ER graph:

- Four flavors: *simple graph* or multigraph, both with or without self-loops.

[12] The subtleties of these distinctions, and their implications for generating and using configuration model random graphs are discussed thoroughly in Fosdick et al., *SIAM Review* **60** (2018) `https://arxiv.org/abs/1608.00607`.

- Connectivity is *specified* by the degree sequence parameter $\vec{k}$, and many patterns vary with the mean $\langle k \rangle$ and variance $\langle k^2 \rangle$ of this sequence.
- The diameter and mean geodesic distance are typically $O(\log n)$, making configuration model graphs "small-world-like," but can shrink further if $\vec{k}$ is extremely heavy-tailed.
- Motif frequencies tend to be $O(1/n)$ when the network is sparse, but can be larger as either $\langle k \rangle$ or $\langle k^2 \rangle$ increases.
- The size of the largest connected component (LCC) tends to be $O(n)$ when G is not too sparse, but a giant component can emergence in sparser settings if either $\langle k \rangle$ or $\langle k^2 \rangle$ is large.

3.2 Generating a configuration model network

Unlike an ER graph, generating a configuration model network is mildly non-trivial. This additional complexity is nearly always worthwhile because of how practically useful they are.

Two facts constrain how we generate a random graph with specified degree structure $\vec{k}$.

Constraint 1: which graph space? First, we must decide which graph space the generated graph will come from, which then tells us which algorithm we use to generate it:

- multigraphs with self-loops: use the Molloy-Reed stub-matching algorithm[13]
- multigraphs with no self-loops: use Fosdick et al. MCMC
- self-loops but no multi-edges: use Fosdick et al. MCMC
- simple graphs: use Fosdick et al. MCMC (specified degrees) or Chung-Lu (expected degrees).

Constraint 2: is it graphical? Second, given a graph space, the degree sequence must be *graphical* within that graph space, meaning that we must be able to pair every "stub" of every edge to some other appropriate stub. What constitutes an appropriate stub depends on that graph space we want the graph to come from.

For instance, the sequence $(1, 1, 10)$ is not graphical as a simple graph, because the degree 10 node will have 8 unmatched "stubs" after connecting to the two degree 1 nodes. On the other hand, this sequence is graphical as a multigraph with self-loops.

[13] Molloy-Reed goes as follows. Create an empty graph G with n nodes and $m = 0$ edges. Allocate an array B of length $\sum_i k_i$. If the length of B is not even, then $\vec{k}$ is not graphical. Then, for every vertex i, write the index i in B exactly k_i times. Take a random permutation of B. Then, for $j = 1 : m$, add the edge $(B[j], B[m + j])$ to the edge set of G. The resulting graph G is then a uniformly random draw from set of multigraphs with self-loops and degree sequence $\vec{k}$. See also `https://en.wikipedia.org/wiki/Configuration_model`.

Given an arbitrary sequence, we can use the Havel-Hakimi algorithm[14] to determine if there exists any simple graph G with $\vec{k}$. More practically, every real-world network A is an existence proof that its degree sequence is graphical, in whatever graph space A comes from. Hence, if we are using a configuration model as a null model, then this constraint is typically automatically satisfied.

3.2.1 The Fosdick et al. MCMC

The most general way to generate a configuration model network is using the Fosdick et al. MCMC,[15] which is a *Markov chain* algorithm that *samples* networks with a specified degree sequence, using a "double-edge swap" to move around in a given graph space.

To start the chain, we must first construct *one* G_0 that has the target graph properties and specified degree sequence $\vec{k}$. If $\vec{k}$ is not graphical, we will know at this step, and can stop. Once we have this initial graph, the algorithm then repeatedly applies a simple function $g : G_t \to G_{t+1}$ to create a sequence of graphs $G_0, G_1, G_2, \ldots, G_t$, for any t we like. If g has the right properties, this graph sequence (the "chain") will be a random walk within the target set $\mathcal{S}$ that eventually visits every member with the same probability. For a sufficiently long chain, the structure of G_t will be independent of G_0 and can we output G_t as the generated random graph, or, if we run the chain sufficiently long between draws, produce as many such random graphs as we like.

Given some $G_t \in \mathcal{S}$, we can generate another $G_{t+1} \in \mathcal{S}$ using a simple *double edge swap* procedure, as follows:

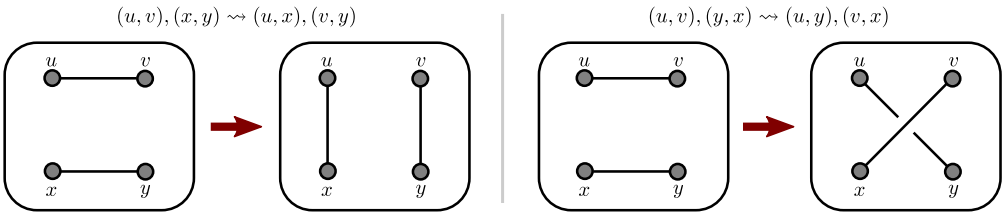
- choose a uniformly random pair of edges $(u, v), (x, y)$
- create G_{t+1} by replacing these edges with either $(u, x), (v, y)$ or $(u, y), (v, x)$ (equal probability)
- finally, if $G_{t+1} \notin \mathcal{S}$ (the sequence exited the target set), revert to G_t and pick new edges.[16]

The idea here is that the double edge swap incrementally randomizes which nodes connect to each other, and so applying it repeatedly can transform any $G_i \in \mathcal{S}$ to any other $G_j \in \mathcal{S}$. Crucially, the double edge swap is *degree preserving*, meaning that the degrees k_u, k_v, k_x, k_y before and after the swap remain the same. Hence, every graph G_t in the sequence will have the same, specified degree

[14] Havel-Hakimi goes as follows. Sort the nodes in decreasing order of their degree. Without loss of generality, let that list be $L = (v_n, v_{n-1}, \ldots, v_1)$, with degrees $\vec{k} = (k_n, k_{n-1}, \ldots, k_1)$. Then, recursively apply the following steps. Take the largest degree vertex v_i and connect it to the k_i vertices immediately to its right in L, i.e., vertices $v_{i-1}, \ldots, v_{i-k_i}$; set $k_i = 0$ and decrement the degrees of each of the right-hand vertices i connected to; finally, remove all vertices from L and $\vec{k}$ that have degree 0. If $\vec{k}$ is graphical, then we will apply this rule at most $n - 1$ times. If at any time $k_i > i - 1$, then there are not enough "stubs" left for i to connect to, and the sequence is not graphical. See also `https://en.wikipedia.org/wiki/Havel-Hakimi_algorithm`.

[15] A complete Python implementation is here `https://github.com/UpasanaDutta98/ConfigModel_MCMC`

[16] Depending on the chosen graph space, there can be other conditions under which a swap is rejected. Fosdick et al., *SIAM Review* (2018) explains.



sequence $\vec{k}$. The third step of the procedure guarantees that each G_{t+1} will also have the target set of graph properties, and hence so will the final output graph.

Convergence and sampling. In the sequence of graphs $G_0, G_1, G_2, \ldots, G_t$ traversed by the Fosdick et al. MCMC algorithm, the first graph G_0 is almost always *not* a uniformly random choice—it is either the particular output of Havel-Hakimi or an empirical network we used as a starting point. So how large should t be before G_t *is* a uniformly random graph with the target properties?

This question is one of *convergence*: how long do we run a Markov chain before G_t is independent of G_0, and hence has "converged" on a uniform distribution over $\mathcal{S}$? And then also, how many steps η_0 should we go between "draws" of graphs from the chain? Instead, we instead rely on heuristics.

Empirical work, based on running the Fosdick et al. MCMC on hundreds of real-world networks of varying size and origin, suggest that for most networks,[17] convergence occurs after $t^* \approx 20m$ double-edge swaps, and that consecutive "draws" can be made about every $\eta_0 = 2m$ steps thereafter.[18]

3.2.2 The Chung-Lu model

Much of the complexity of generating configuration model random graphs stems from requiring that G have *exactly* the specified degree sequence $\vec{k}$. If we relax this constraint, requiring only that G have a degree sequence close to $\vec{k}$, then we can use the Chung-Lu model[19] to generate a simple random graph in much the same way as we generate an ER graph. [20]

[17] These heuristics work best for networks in which the edge density ρ and the maximum degree are not too high. If a network violates these technical constraints, the Markov chain tends to behave in unusual ways, which requires a direct estimation of an appropriate sampling gap η_0.

[18] See Dutta and Clauset, "Convergence criteria for sampling random graphs with specified degree sequences." Preprint, arXiv:2105.12120 (2021). `https://arxiv.org/abs/2105.12120`

[19] Chung and Lu, *Annals of Combinatorics* **6**, 125–145 (2002) `http://math.ucsd.edu/~fan/wp/conn.pdf`

[20] It's possible to draw from the Chung-Lu model in only $O(n+m)$ time, using an algorithm described by Miller and Hagberg, *Proc. Workshop on Algorithms and Models for the Web Graph* (WAW 2011) `https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-11-01482`.

Suppose that we fix a choice of nodes i and j, with degrees k_i and k_j. What is the probability that i, j are connected? For now, let $k_i = 1$, i.e., i only has one edge to connect anywhere. From i's perspective, it has k_j chances for its one edge "stub" to connect to j. And, across the entire network, there are $2m - k_i$ possible stubs where its edge could land (we subtract off k_i because we prohibit self-loops). Hence, the probability that i connects to j is $k_j/(2m - k_i)$. This argument holds independently for each of i's stubs,[21] implying that a probability that any of i's stubs connects to any of j's stubs

$$\text{if} \quad \max_i k_i^2 < 2m \quad \text{then} \qquad p_{ij} = k_i \left(\frac{k_j}{2m - k_i} \right) \approx \frac{k_i k_j}{2m} \ . \tag{8}$$

The constraint on the left-hand side in Eq. (**??**) ensures that $p_{ij} \leq 1$ always.

The Chung-Lu model calculates this probability for each pair i, j and then adds each edge (i, j) to G with probability p_{ij}:

$$\forall_{i>j} \qquad A_{ij} = A_{ji} = \begin{cases} 1 & \text{with probability } p_{ij} \\ 0 & \text{otherwise} \ . \end{cases}$$

The Chung-Lu model does not produce exactly $\vec{k}$, and instead generates networks whose degrees are $\vec{k}$ in expectation, only. In fact, the degree of a node i will be a Poisson-distributed random variable, with mean equal to the specified degree k_i, if the network is sparse and the maximum degree is not too large. And, because we flip one coin for each pair of nodes, Chung-Lu graphs are simple, and take the same amount of time to generate as a similarly sized ER graph. They also generalize naturally to directed networks, when both $\vec{k}^{\text{in}}$ and $\vec{k}^{\text{out}}$ are specified, by setting $p_{ij} \approx k_i^{\text{out}} k_j^{\text{in}}/m$, and dropping the requirement that $A_{ij} = A_{ji}$.

3.3 Descriptive statistics of configuration model random graphs

The precise patterns of various descriptive network statistics, e.g., motif counts like reciprocity or the clustering coefficient, or positional measures like the diameter or the mean geodesic path length, depend on the choice of degree sequence.

In general, most of the measures have similar behavior to those of an ER graph, except that in a configuration model, their precise value can depend on the degree sequence's mean $\langle k \rangle$ and (uncentered) variance $\langle k^2 \rangle$. We omit the details of the mathematical calculations and instead simply state the results, which hold for sufficiently large and sparse networks.[22]

[21] Such an independence assumption is approximately valid for sparse and large networks, because the probability of multiple connections between i and j is vanishingly small. Hence, the difference between this argument and the rigorous one is negligible in most cases of interest.

[22] The equations here are derived under the multigraph with self-loops flavor of the configuration model, and hold for the simple graph flavor in the limit. Details of the derivations are in *Networks* by Mark Newman (2019).

For example, the *expected clustering coefficient* C is

$$C = \frac{1}{n} \frac{\left[\langle k^2 \rangle - \langle k \rangle\right]^2}{\langle k \rangle^3} \ , \tag{9}$$

which is $O(1/n)$ when $\langle k \rangle$ and $\langle k^2 \rangle$ are finite, much like the ER graph.

A giant component, i.e., a largest connected component containing $O(n)$ nodes, appears in the configuration model when $\langle k^2 \rangle - 2\langle k \rangle > 0$ which can be true even when $\langle k \rangle < 1$, if the variance is large enough.

And, the diameter of a configuration model is also $O(\log n)$, just like the ER graph, but can be even smaller, like $O(\log \log n)$ for especially heavy-tailed degree sequences (in which the maximum degree $k_{\max} > \sqrt{n}$).

Finally, the *expected number of common neighbors* n_{ij} for a pair of nodes i, j is

$$n_{ij} = p_{ij} \frac{\langle k^2 \rangle - \langle k \rangle}{\langle k \rangle} \ . \tag{10}$$

Interpretation. The fact that this quantity is proportional to p_{ij} implies that not only are high-degree nodes likely to be connected themselves, they are also likely to have many neighbors in common. Another calculation shows that high-degree nodes also tend to be located on many short paths through the network. Taking these facts together, along with the insight that most real-world networks have heavy-tailed degree distributions, and the most-well-connected nodes have degrees *far* higher than the typical node, we arrive at the intuition that high-degree nodes play a special role in structuring a random graph. But their specialness is no mystery—it's entirely driven by the fact that these nodes simply have *many* more chances for connections, and the randomness of the graph ensures that these many connections spread very widely. It would be a very unusual network structure for high-degree nodes to not be connected to each other, have many common neighbors, and be positioned centrally in the network.

4 Random graphs as null models

Random graphs are remarkably useful as a **null model** for investigating the structure of a real network G. They allow us to quantitatively answer the question

How much of an observed pattern is explained by
{edge density or degrees} alone, under randomness?

To make this work, we must first *parameterize* the random graph model so that the distribution over graphs $\Pr(G)$ that it defines is "centered" on our real network $G_\circ$ in some way. Typically, we

set the model's free parameters by following a statistical estimation procedure, such as maximum likelihood. For the ER model, the maximum likelihood parameterization is simply $p = \langle k \rangle / (n-1)$. For the configuration model, we set $\vec{k}$ equal to the degree sequence of $G_\circ$.

Doing so allows us to interpret the parameterized random graph as a null model for $G_\circ$, i.e., a simple data generating process, with only a few structural assumptions, that generates a reference point or distribution for deciding whether some statistic or other pattern in $G_\circ$ is large or small, typical or unusual, relative to the range of such statistics the null produces.[23] If the empirical value is typical under the null, then we may say that it is "explained" by the null's underlying assumptions, e.g., edge density plus randomness (ER model), or the degree structure plus randomness (configuration model).

4.1 For example

So see how these ideas work in practice, we'll apply them to understand the structural patterns of the metabolic network of *Archaeoglobus fulgidus*,[24] which contains $n = 315$ nodes and 5434 directed edges. Ignoring edge direction yields $m = 3793$ undirected edges, a mean degree of $\langle k \rangle = 24.08$, and a standard deviation $\sigma_k = 28.17$. This edge density value is fairly high, and hence unsurprisingly, the largest connected component contains the large majority of nodes (296, 94%). Similarly, the fact that $\sigma_k > \langle k \rangle$ indicates that the degree distribution is likely heavy-tailed. Let's dig in more.

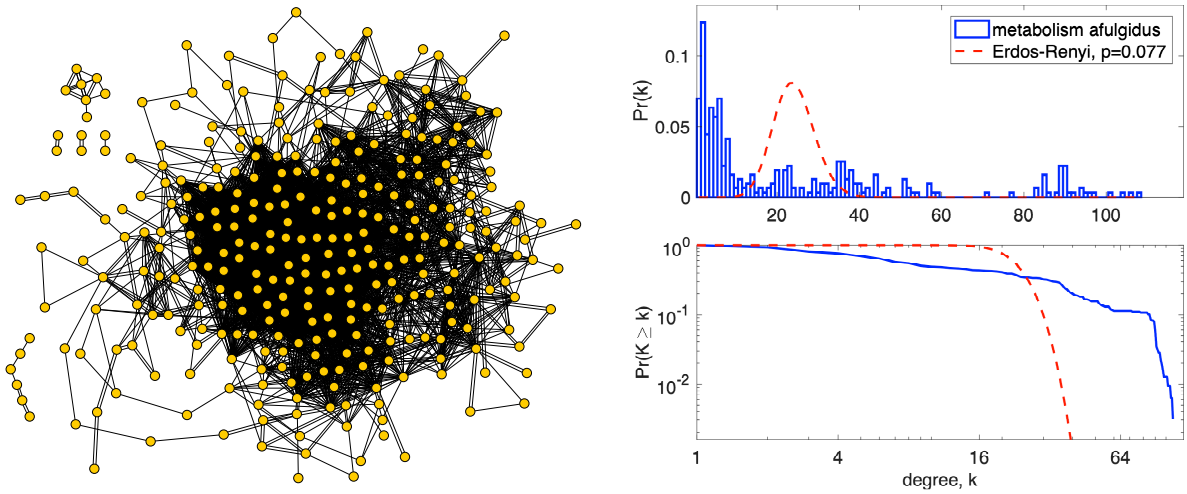
4.1.1 What does edge density explain?

The left-hand figure below shows a ridiculogram for this network—super informative, right? The right-hand figure plots the (undirected) degree distribution of the *Archaeoglobus* network, as a simple normalized histogram and as a CCDF, along with the degree distribution predicted by an ER graph with the same edge density, $p = \langle k \rangle / (n-1) = 0.0766960$. This parameterization allows us to ask whether a comparable ER random graph model qualitatively matches the empirical *Archaeoglobus* network's statistical structure.

Clearly, it does not: the Poisson distribution places far too much density around the mean degree, and far (far!) too little at either low *or* high degrees nodes. The highest degree node in the *Archaeoglobus* network is $k_{\max} = 108$, but the probability of generating such a node under the maximum likelihood ER model is less than 4.3937×10^{-36}, or about 1 time every 1,000,000,000,000,000,000,000,000,000,000,000,000 networks, or basically never. If the ER model were the true data generating process, this node would be rather surprising to see.

[23] Do you see why it makes no sense to interpret the similarity of the empirical $\langle k \rangle$ with the mean degrees of a parameterized ER graph, or any statistic of the empirical $\vec{k}$ with that of a parameterized configuration model network?

[24] A sulfur-eating archae: `https://microbewiki.kenyon.edu/index.php/Archaeoglobus_fulgidus`



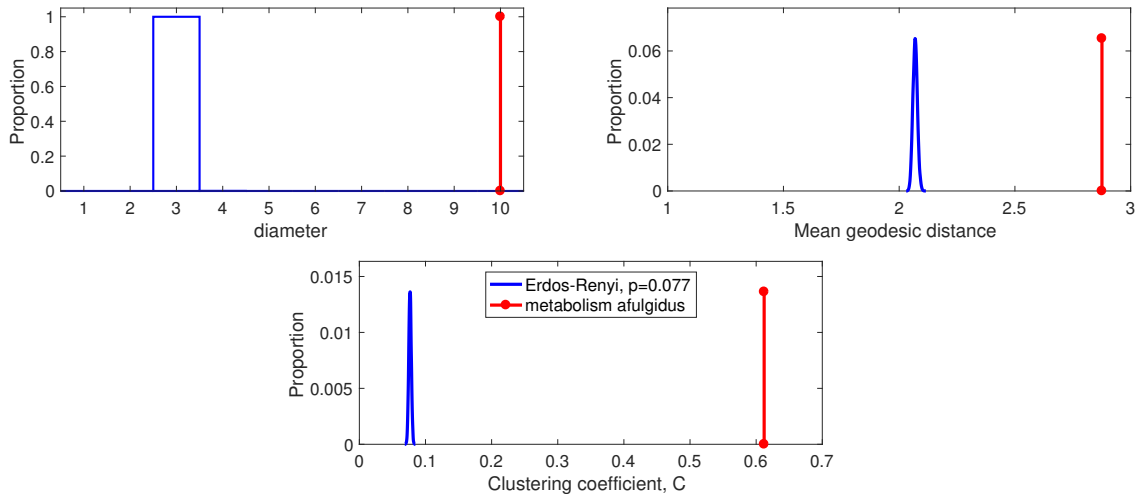
Neither does the edge density explain the path length structure or the clustering coefficient of the *Archaeoglobus* network. The top two plots below compare the empirical diameter $\ell_{\max}$, mean geodesic distance $\langle \ell \rangle$, and clustering coefficient C vs. the corresponding null distributions (tabulated from 10^4 ER networks drawn from the null). In each case, the null distribution dramatically underestimates the value of the *Archaeoglobus* statistic: the empirical diameter is a whopping $\ell_{\max} = 10$, compared to an expected value of 3, the empirical mean geodesic distance is $\langle \ell \rangle = 2.88$, while the null yields only $\langle \ell \rangle = 2.11$ (not *that* far off, all things considered, but still not close enough to explain anything), and the empirical clustering coefficient is $C = 0.61$, while the ER graph yields only $C = 0.077$.

What we learn by observing that the empirical values are far outside the range produced by an ER graph null is that there is substantial structure in the *Archaeoglobus* network that is not explained by edge density alone.

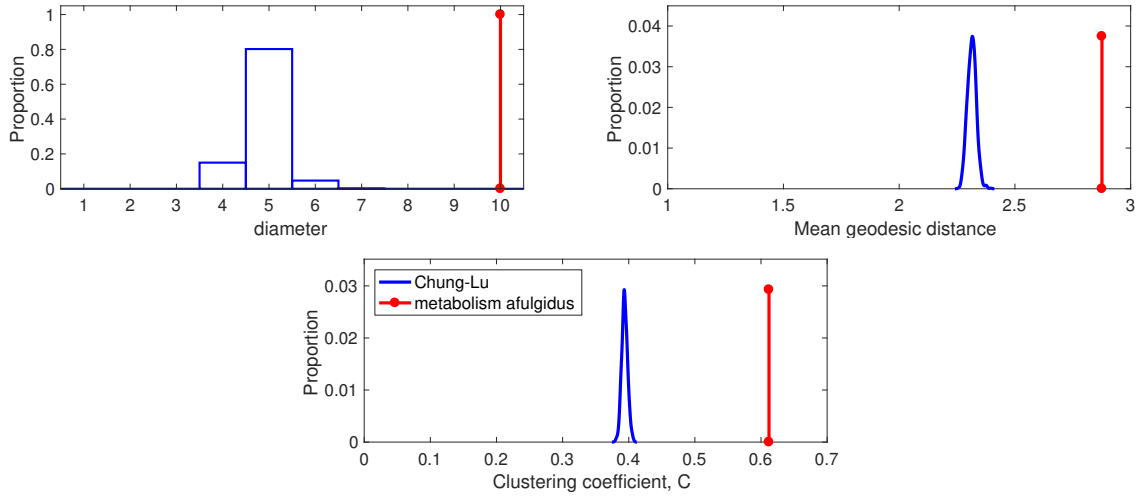
4.1.2 What does degree structure explain?

By parameterizing a random graph to have the same degree structure as the *Archaeoglobus* network, we lose the ability to sensibly compare the degrees between the null and the empirical network. In other words, we cannot compare the degree distributions like we did above. Instead, we focus on the other three measures (figures below), which reveals that the degree structure explains *more* of the empirical patterns than edge density alone, but it doesn't explain any of these statistics completely.

Both the diameter and the mean geodesic distance are larger now (concentrated around $\ell_{\max} = 5$



and $\langle \ell \rangle = 2.31$), but still a ways off. The most improved of the three statistics is the clustering coefficient, which moves up to $C = 0.39$. Clearly, this metabolic network has more structure than can be explained by the degrees alone, but the heavy-tailed degree distribution does explain a substantial portion of the prevalence of triangles in the network.



5 Supplemental readings

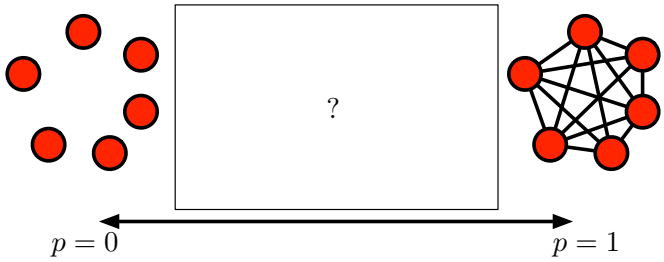
1. Chapters 11 and 12 in *Networks*

Appendix: A phase transition in network connectedness

A particularly interesting property of the Erdős-Rényi random graph, is that it exhibits a *phase transition*[25] in the connectedness of the network $G(n, p)$ as a function of the mean degree $c = p(n-1)$.

When the mean degree $c < 1$, graphs drawn from this model are almost surely disconnected, and composed of $O(n)$ components, the largest of which is still very small, around $O(1)$ nodes, and all of which are mostly "tree-like." But, when the mean degree $c > 1$, although the graph is still disconnected in the strict sense, the largest connected component is now huge, being $O(n)$ nodes. As c cross the *critical value* of $c = 1$, the networks switches from one "phase" to the other.

Phase transitions are interesting because they represent a *qualitative* (very large, and very sudden) change in the fundamental behavior of the system. They are non-linear effects, in which a small change in the parameter c, i.e., going from just below 1 to just above 1, leads to a big change in the system's behavior, i.e., the size of the largest connected component.



Without loss of generality, define $p = c/(n-1)$, i.e., p is just the iid probability that an edge exists. Now, consider the two limiting cases for the parameter p. If $p = 0$ we have a fully empty network with n completely disconnected vertices. Every component in this network has the same size, and that size is a $O(1/n)$ fraction of the size of the network. In the jargon of physics, the size of the largest component here is an *intensive* property, meaning that it is independent of the system size.

On the other hand, if $p = 1$, then every edge exists and the network is an n-clique. This single component has a size $O(n)$, proportional to the size of the network. In the jargon of physics, the size of the largest component here is an extensive property, meaning that it depends on the system

[25] The term "phase transition" comes from the study of *critical phenomena* in physics. Classic examples include the melting of ice, the evaporation of water, the magnetization of a metal, etc. Generally, a phase transition characterizes a sudden and qualitative shift in the bulk properties or global statistical behavior of a system. In this case, the transition is discontinuous and characterizes the transition between a mostly disconnected and a mostly connected networked.

size n.[26] Hence, as we vary p from 0 to 1, the size of the largest component transforms from an intensive property to an extensive one, and this is the hallmark of a phase transition. Of course, it could be that the size of the largest component becomes extensive only in the limit $p \to 1$, but in fact, something much more interesting happens. (When a graph is sparse, what other network measures are intensive? What measures are extensive?)

The sudden appearance of a "giant" component

In this section, we will mathematically show that a "giant" component, i.e., one whose size is extensive, emerges at a single critical value of p.

Let u denote the average fraction of vertices in $G(n,p)$ that do *not* belong to the giant component. Thus, if there is no giant component (e.g., $p=0$), then $u=1$, and if there is then $u<1$. In other words, let u be the probability that a vertex chosen uniformly at random does not belong to the giant component.

For a vertex i not to belong the giant component, it must not be connected to any other vertex that belongs to the giant component. This means that for every other vertex j in the network, either (i) i is not connected to j by an edge or (ii) i is connected to j, but j does not belong to the giant component. Because edges are iid, the former happens with probability $1-p$, the latter with probability pu, and the total probability that i does not belong to the giant component via vertex j is $1-p+pu$.

For i to be disconnected from the giant component, this must be true for all $n-1$ choices of j, and the total probability u that some i is not in the giant component is

$$\begin{aligned} u &= (1-p+pu)^{n-1} \\ &= \left[1-\frac{c}{n-1}(1-u)\right]^{n-1} \qquad (11) \\ &= \mathrm{e}^{-c(1-u)} \qquad (12) \end{aligned}$$

where we use the identity $p=c/(n-1)$ in the first step, and the identity $\lim_{n\to\infty}\left(1-\frac{x}{n}\right)^n = \mathrm{e}^{-x}$ in the second.[27]

[26] Other examples of extensive properties in physics include mass, volume and entropy. Other examples of *intensive* properties—those that are independent of the size of the system—include the density, temperature, melting point, and pressure. See `https://en.wikipedia.org/wiki/Intensive_and_extensive_properties`

[27] We can sidestep using the second identity by taking the logarithms of both sides of Eq. (**??**):

$$\ln u = (n-1)\ln\left[1-\frac{c}{n-1}(1-u)\right] \simeq -(n-1)\frac{c}{n-1}(1-u) = -c(1-u)$$

where the approximate equality becomes exact in the limit of large n. Exponentiating both sides of our approximation then yields Eq. (**??**). This should look familiar.

If u is the probability that i is not in the giant component, then let $S = 1 - u$ be the probability that i belongs to the giant component. Plugging this expression into Eq. (**??**) and eliminating u in favor of S yields a single equation for the size of the giant component, expressed as a fraction of the total network size, as a function of the mean degree c:

$$S = 1 - \mathrm{e}^{-cS} \quad . \tag{13}$$

Note that this equation is transcendental and there is no simple closed form that isolates S from the other variables.[28]

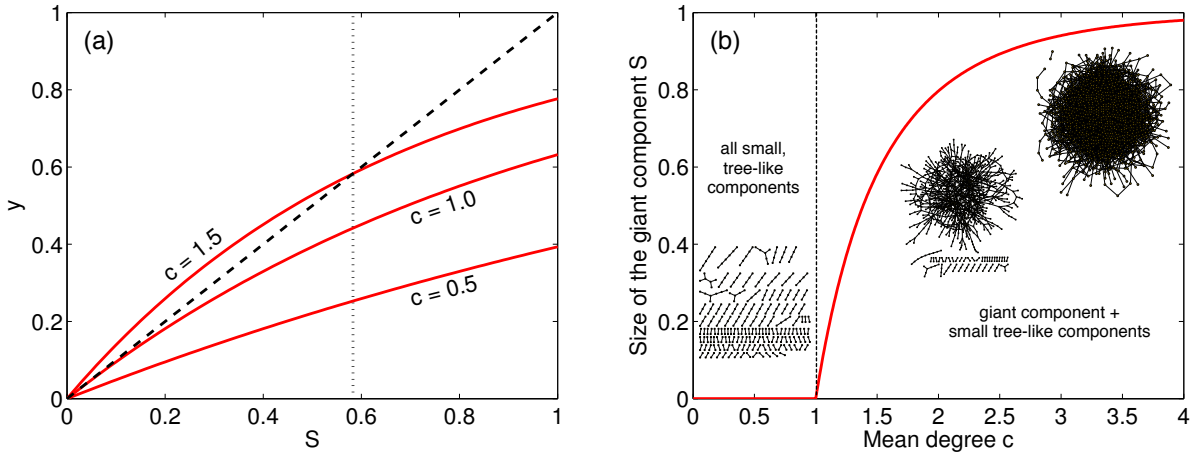


Figure 3: (a) Graphical solutions to Eq. (**??**), showing the curve $y = 1 - \mathrm{e}^{-cS}$ for three choices of c along with the curve $y = S$. The locations of their intersection gives the numerical solutions to Eq. (**??**). Any solution $S > 0$ implies a giant component. (b) The solution to Eq. (**??**) as a function of c, showing the discontinuous emergence of a giant component at the critical point $c = 1$, along with some examples random graphs from different points on the c axis.

We can visualize the shape of this function by first plotting the function $y = 1 - \mathrm{e}^{-cS}$ for $S \in [0, 1]$ and asking where it intersects the line $y = S$. The location of the intersection is the solution to Eq. (**??**) and gives the size of the giant component. Figure 1 (above) shows this exercise graphically. In the "sub-critical" regime $c < 1$, the curves only intersect at $S = 0$, implying that no giant component exists. In the "super-critical" regime $c > 1$, the lines always intersect at a second point $S > 0$, implying the existing of a giant component. The transition between these two "phases" happens at $c = 1$, which is called the "critical point".

[28] For numerical calculations, it may be useful to express it as $S = 1 + (1/c)W(-ce^{-c})$ where $W(.)$ is the *Lambert W-function* and is defined as the solution to the equation $W(z)\mathrm{e}^{W(z)} = z$.

Branching processes and percolation

An alternative analysis considers building each component, one vertex at a time, via a *branching process*. Here, the mean degree c plays the role of the expected number of additional vertices that are joined to a particular vertex i already in the component. The analysis can be made entirely analytical, but here is a simple sketch of the logic.

When $c < 1$, on average, this branching process will terminate after a finite number of steps, and the component will have a finite size. This is the "sub-critical" regime. In contrast, when $c > 1$, the average number of new vertices grows with each new vertex we add, and thus the branching process will never end. Of course, it must end at some point, and this point is when the component has grown to encompass the entire graph, i.e., it is a giant component. This is the "super-critical" regime. At the transition, when $c = 1$, the branching process could in principle go on forever, but instead, due to fluctuations in the number of actual new vertices found in the branching process, it does terminate. At $c = 1$, however, components of all sizes are found and their distribution can be shown to follow a power law.