Chapter 24

Uncertainty quantification and error analysis

As discussed in Ch. 10, network data are necessarily imperfect. Missing and even spurious nodes and edges can create uncertainty in what the observed data tell us about the original network. In this chapter, we dive deeper into tools that allow us to quantify such effects and probe more deeply into the nature of an unseen network from our observations of it.

24.1 Computational and mathematical approaches

We can understand the effects of errors and missing data by computational methods or mathematical models. The computational approach is quite similar in spirit to the null models discussed in Sec. 11.6. There we applied some form of randomization algorithm to G to generate G_{null} whose properties we could compare to G. Now, instead of randomizing a network, we can apply a sampling or error algorithm to the network, then compare statistics of the sampled network to those same statistics on the original network. For example, Martin and Niemeyer [294] perform experiments looking at the random removal or addition of nodes or edges to see how robust different centrality measures are to such errors, finding for instance that degree centrality is quite robust to such errors while eigenvector centrality is more affected by errors such as missing nodes, especially when the nodes that are missing had high centrality. Borgatti et al. [67] and Frantz et al. [169] perform computational studies along similar lines.

Mathematical approaches, on the other hand, model the sampling or error mechanism probabilistically, which can give further insight into the problems we face. Mathematical *uncertainty quantification* enables us to understand the uncertainty in model parameters and summary statistics computed from data. These models can even provide guidance on important questions, such as whether further data collection is necessary. We discuss a variety of mathematical approaches in this chapter.

The advantage of the computational approach is that it can give us intuition about how different statistics are affected by errors and sampling, especially complicated error

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

modes that are difficult to study mathematically. The disadvantage is that they require starting from a known "true" network. Still, these approaches can give us intuition about the statistical estimators themselves which we can use when calculating those statistics on our data.

The advantage of the mathematical approach is that it gives tractable insights that are not as easily seen computationally. Further, those insights can allow us to build statistical models that can extrapolate in various ways from the observed data to the unseen network. The disadvantage is that the probabilistic models are usually limited to simpler forms of errors, for example that edges are independently observed or missing from the data. While these assumptions and approximations are limiting in many ways, the insights and extrapolations they provide are still quite useful.

In general, we encourage practitioners to consider both computational and mathematical approaches to network uncertainty quantification.

24.2 Missing data and its effects

Suppose that nodes present in the true network are absent in the data we have available. What effect does this have on our understanding of the network? What does the degree distribution look like and how does it compare to the original? What about the overall network structure?

One tool we can use for these questions is *percolation*. Percolation usually models simple random sampling, where nodes or links are randomly retained in the network, or equivalently, randomly removed, although it can be extended in various ways to capture bias (for instance, that high-degree nodes are more likely, or less likely, to be observed in the data than other nodes). In fact, percolation has been used in the context of network resilience, where nodes or edges are not missing but damaged or non-functioning, and can even allow us to understand what happens to a network under attack. But for our data analysis purposes, missingness and damage are effectively equivalent.

One interesting conclusion that percolation shows us is that networks undergo *phase transitions* (see also Ch. 22) based on the level of sampling. We discuss one now.

Is the observed network globally connected?

A network is globally connected when it has a giant component, a connected component containing the majority of nodes. We can determine the level of sampling necessary for a giant component to exist by following a simple heuristic: the network is globally connected when a random node i, whose neighbor j belongs to the giant component, is also connected to at least one other node. If this is not the case, the network is globally fragmented. To find the minimum point where this occurs, we write this condition as

$$\langle k_i \mid i \leftrightarrow j \rangle = \sum_{k_i} k_i P(k_i \mid i \leftrightarrow j) = 2, \tag{24.1}$$

where $P(k_i \mid i \leftrightarrow j)$ is the probability that i has degree k_i given it is connected to j. Let's simplify this by using the degree distribution $P(k)$ instead of this conditional

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

probability. From Bayes' theorem and the joint probability $P(k_i, i \leftrightarrow j)$ we have

$$P(k_i \mid i \leftrightarrow j) = \frac{P(k_i, i \leftrightarrow j)}{P(i \leftrightarrow j)} = \frac{P(i \leftrightarrow j \mid k_i)P(k_i)}{P(i \leftrightarrow j)}. \tag{24.2}$$

If we assume the network is uncorrelated and sparse (meaning, we neglect loops), then $P(i \leftrightarrow j \mid k_i) = k_i/(N-1)$ and $P(i \leftrightarrow j) = \langle k \rangle /(N-1)$. Substituting into Eq. (24.2),

$$P(k_i \mid i \leftrightarrow j) = \frac{k_i P(k_i)}{\langle k \rangle}. \tag{24.3}$$

Finally, applying this to Eq. (24.1) we arrive at a succinct expression for global connectivity:

$$\frac{1}{\langle k \rangle} \sum_{k_i} k_i^2 P(k_i) = \frac{\langle k^2 \rangle}{\langle k \rangle} := \kappa = 2. \tag{24.4}$$

Now, what happens when nodes are missing? Assume a fraction p of nodes are removed independently from the network (i.e., each node is independently sampled with probability $1-p$). A node with degree k_0 in the original network will have a new degree k due to sampling, on average,

$$\binom{k_0}{k}(1-p)^k p^{k_0-k}. \tag{24.5}$$

Applied to all nodes, this modifies the original degree distribution into the new distribution

$$P'(k) = \sum_{k_0=k}^{\infty} P(k_0)\binom{k_0}{k}(1-p)^k p^{k_0-k}. \tag{24.6}$$

(Primes denote quantities after sampling.) Now, let's compute the first and second moments, $\langle k \rangle'$ and $\langle k^2 \rangle'$, for this new distribution in terms of the original moments. For the first moment,

$$\begin{aligned}
\langle k \rangle' &= \sum_k k P'(k) \\
&= \sum_k k \sum_{k_0=k}^{\infty} P(k_0)\binom{k_0}{k}(1-p)^k p^{k_0-k} \\
&= \sum_{k_0=0}^{\infty} P(k_0) \underbrace{\sum_k k\binom{k_0}{k}(1-p)^k p^{k_0-k}}_{\text{mean of a binomial distribution}} \\
&= \sum_{k_0=0}^{\infty} P(k_0) k_0 (1-p) \\
&= \langle k_0 \rangle (1-p).
\end{aligned} \tag{24.7}$$

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

A similar calculation for $\langle k^2 \rangle'$ using the second moment of a binomial distribution gives $\langle k^2 \rangle' = \langle k_0^2 \rangle (1-p)^2 + \langle k_0 \rangle (1-p)p$. The sampled network is globally connected when $p \leq p_c$ such that

$$\frac{\langle k^2 \rangle'}{\langle k \rangle'} = \frac{\langle k_0^2 \rangle}{\langle k_0 \rangle}(1 - p_c) + p_c = 2 \tag{24.8}$$

holds, or

$$1 - p_c = \frac{1}{\langle k_0^2 \rangle / \langle k_0 \rangle - 1} = \frac{1}{\kappa_0 - 1}, \tag{24.9}$$

where κ_0 is computed using the original network's degree distribution.

The "critical" sampling value $1 - p_c$ from Eq. (24.9) allows us to understand how much random sampling disconnects a network, before sampling occurs. This p_c is known as the *percolation threshold*, where (in the limit of large sizes) networks undergo a phase transition from globally fragmented to globally connected. And, while this result is somewhat limited by strong assumptions, it allows us to understand how sampling changes the average degree $\langle k \rangle$ (Eq. (24.7)) and even the degree distribution $P(k)$ (Eq. (24.6)).

We now consider a percolation argument in a more complex setting.

24.3 Community structure

In Ch. 12 we discussed communities, how some networks are organized into densely connected groups of nodes called communities, clusters, or modules. One facet of this structure that has attracted interest is overlapping communities (Sec. 12.7), where nodes may belong to multiple groups. (The community structure is now a cover of the nodes, not a partition.)

How does network uncertainty affect our discovery of overlapping communities? Here we explore this question using a mathematical model for overlapping communities along with a (relatively simplistic) modeling assumption of how an overlapping community detection method may perform when nodes are missing.

24.3.1 Modeling overlapping communities

Overlapping communities can be well modeled with a bipartite graph, also known as an affiliation network [485]. This graph contains two types of nodes representing the nodes and the communities in the network. Undirected links represent which node belongs to which community. The graph is characterized by two degree distributions, r_m and s_n, governing the fraction of nodes that belong to m communities and the fraction of communities that contain n nodes, respectively [341, 339]. For simplicity, links are distributed randomly between "node nodes" and "community nodes" following these degree distributions. The average number of communities per node is $\sum_m m r_m := \mu$ and the average number of nodes per community is $\sum_n n s_n := \nu$. We then derive two networks from the bipartite graph by projecting onto either the nodes or the communities. One is the original network between nodes, while the other is a network where each

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

node represents a community and two communities are linked if they overlap, i.e., they share at least one node.

We model missing nodes by (prior to projection) retaining nodes independently with probability p; otherwise, they are removed with probability $1 - p$. Meanwhile, we also model the effects of this sampling on a (hypothetical) community detection algorithm. We assume the algorithm fails to find a community if fewer than a critical fraction f_c of its original nodes remain, the idea being that detection will fail when too little of the original community remains in the sample. These undetected communities are removed from the community network but any member nodes that were sampled are not removed from the node network (Fig. 24.1). A percolation analysis can show us the effects that sampling has on both the network structure and its communities. The giant component in the original network disappears when, due to missing nodes, the sampled network lacks global connectivity; in the community network, it vanishes when the communities become uncoupled (non-overlapping). Can we always detect the overlap, all the way down to the percolation threshold? Or does the overlapping structure disappear (well) before that sampling point?

Before proceeding with analysis, it's worth noting that this model makes two assumptions about the communities: that all interactions within each community exist and are equal, and that there are no differences between individual nodes that share a community—there are no "captains" or "team leaders." While simplistic, these nevertheless can form the basis for more complex analyses as needed.

To understand whether sampling hides overlapping communities, disconnects the network, or both, we determine $S(p)$, the fraction of observed nodes within the giant component, as a function of p, for both the node and community networks. We use four generating functions (Sec. 22.4.1):

$$\begin{aligned} f_0(z) &= \sum_{m=0}^{\infty} r_m z^m, \quad f_1(z) = \frac{1}{\mu} \sum_{m=0}^{\infty} m r_m z^{m-1}, \\ g_0(z) &= \sum_{n=0}^{\infty} s_n z^n, \quad g_1(z) = \frac{1}{\nu} \sum_{n=0}^{\infty} n s_n z^{n-1}. \end{aligned} \tag{24.10}$$

These functions generate probabilities for (f_0) a randomly chosen node to belong to m communities, (f_1) a random node within a randomly chosen community to belong to m other communities, (g_0) a random community to contain n nodes, and (g_1) a random community of a randomly chosen node to contain n other nodes.

To analyze this model we now separately study the two projections (the node and community networks) of the original bipartite graph.

Network

Consider a randomly chosen node A that belongs to a group of size n. Let $P(k \mid n)$ be the probability that A still belongs to a connected cluster of k nodes (including itself) in this group after sampling:

$$P(k \mid n) = \binom{n-1}{k-1} p^{k-1} (1-p)^{n-k}. \tag{24.11}$$

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

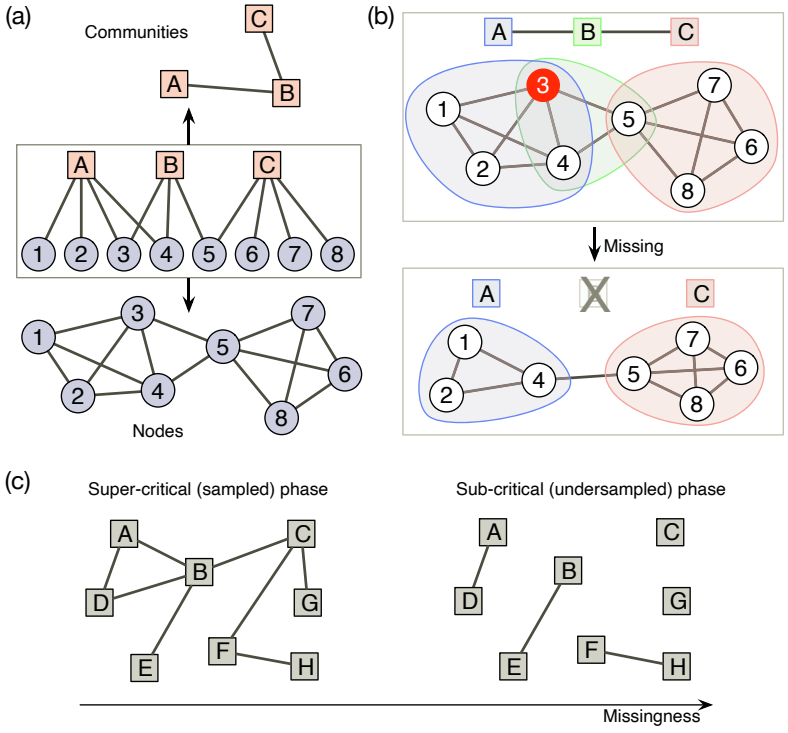

Figure 24.1 Sampling in a network model with overlapping communities. (**a**) Using an affiliation network, we analyze two networks, one representing the linkages between network nodes and a second detailing the overlapping connectivity between the communities themselves. (**b**) Missing nodes (node 3) may prevent communities (community B) from being detected. (**c**) With sufficient missingness, we transition from a well-sampled to an undersampled phase. This can cause (Fig. 24.2) the community network to become disconnected, preventing us from detecting the overlapping community structure, even though the network itself remains connected.

The generating function for the number of other nodes connected to A within this group is

$$h_n(z) = \sum_{k=1}^{n} P(k \mid n) z^{k-1} = (zp + 1 - p)^{n-1} . \tag{24.12}$$

Averaging over community size:

$$h(z) = \frac{1}{\nu} \sum_{n=0}^{\infty} n s_n h_n(z) = g_1(zp + 1 - p). \tag{24.13}$$

The total number of nodes that A is connected to, from all communities it belongs to, is then generated by

$$G_0(z) = f_0(h(z)). \tag{24.14}$$

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

Likewise, the total number of nodes that a randomly chosen neighbor of A is connected to is generated by

$$G_1(z) = f_1(h(z)). \tag{24.15}$$

Before determining S, we first identify the critical sampling point p_c where the giant component emerges. This happens when the expected number of nodes two steps away from a random node exceeds the number one step away, or

$$\partial_z G_0(G_1(z))\big|_{z=1} - \partial_z G_0(z)\big|_{z=1} > 0. \tag{24.16}$$

Substituting Eqs. (24.14) and (24.15) into (24.16) gives $f_0'(1)h'(1)[f_1'(1)h'(1)-1] > 0$ or $f_1'(1)h'(1) > 1$, making the condition for a giant component to exist, since $h'(1) = pg_1'(1)$, be

$$pf_1'(1)g_1'(1) > 1. \tag{24.17}$$

For constant network degrees, $r_m = \delta(m,\mu)$ and $s_n = \delta(n,\nu)$, where $\delta(a,b) = 1$ if $a = b$ and 0 otherwise, this gives $p(\mu-1)(\nu-1) > 1$. If $\mu = 3$ and $\nu = 3$, for example, then the transition occurs at $p_c = 1/4$.

To find S, consider the probability u for node A not to belong to the giant component. A is not a member of the giant component only if all of A's neighbors are also not members, so u satisfies the self-consistency condition $u = G_1(u)$. The size of the giant component is then $S = 1 - G_0(u)$.

Communities

For the community network, we proceed with a similar calculation.

Consider a random community C and then a random member node A. Let $Q(\ell \mid m)$ be the probability that C is connected to ℓ communities, including itself, through node A, who was originally connected to m communities including C:

$$Q(\ell \mid m) = \binom{m-1}{\ell-1} q_1^{\ell-1} (1-q_1)^{m-\ell}, \tag{24.18}$$

where

$$q_1 = \frac{1}{\nu} \sum_{n=0}^{\infty} n s_n \sum_{i=x}^{n} \binom{n-1}{i-1} p^{i-1}(1-p)^{n-i}. \tag{24.19}$$

(Notice that $q_1 = 1$ when $x(n) := \lceil nf_c \rceil = 1$ for all n.) The generating function j_m for the number of communities that C is connected to, including itself, through A is

$$j_m(z) = \sum_{\ell=1}^{m} Q(\ell \mid m) z^{\ell-1} = (zq_1 + 1 - q_1)^{m-1}. \tag{24.20}$$

Once again, averaging j_m over memberships gives

$$j(z) = \frac{1}{\mu} \sum_{m=0}^{\infty} m r_m j_m(z) = f_1(zq_1 + 1 - q_1). \tag{24.21}$$

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

The total number of communities that C is connected to is *not* generated by $g_0(j(z))$ but by $\tilde{g}_0(j(z))$, where the $\tilde{g}_i$ are generating functions for community sizes in the sampled network:

$$\tilde{g}_0(z) = \sum_{n=0}^{\infty} \tilde{s}_n z^n, \qquad \tilde{g}_1(z) = \frac{\sum_{n=0}^{\infty} n\tilde{s}_n z^{n-1}}{\sum_{n=0}^{\infty} n\tilde{s}_n}. \tag{24.22}$$

The probability $\tilde{s}_k$ to have k nodes remaining within a community after sampling is

$$\tilde{s}_k = \frac{\sum_n \binom{n}{k} p^k (1-p)^{n-k} s_n}{\sum_n \sum_{k'=x}^{n} \binom{n}{k'} p^{k'} (1-p)^{n-k'} s_n}. \tag{24.23}$$

The denominator in $\tilde{s}_k$ is necessary for normalization since the community detection algorithm cannot observe communities with fewer than $\lceil n f_c \rceil$ members. Notice that $\tilde{s}_n = s_n$ when $s_n = \delta(n, \nu)$ and $\lceil n f_c \rceil = n = \nu$. Finally, the total number of communities connected to C through any member node is generated by $F_0(z) = \tilde{g}_0(j(z))$ and the total number of communities connected to a random neighbor of C is generated by $F_1(z) = \tilde{g}_1(j(z))$. As before, the community network has a giant component when $\partial_z F_0(F_1(z))|_{z=1} - \partial_z F_0(z)|_{z=1} > 0$ and $S = 1 - F_0(u) = 1 - \tilde{g}_0(j(u))$, where u satisfies $u = F_1(u) = \tilde{g}_1(j(u))$.

24.3.2 Missing data reveals an *inference gap*

For the uniform case with $\mu = 3$, $\nu = 3$, and $f_c > 2/3$, the critical point for the community network is $p_c = 1/2$, a considerably higher threshold than for the node network ($p_c = 1/4$) discussed above. Figure 24.2 shows S for $\mu = 3$ and $\nu = 6$. The *inference gap*, the difference between the critical points for the node and community networks, grows as the community method's detection cutoff increases, covering a significant range of p for the larger values of f_c. Intuitively this makes sense: a high detection cutoff means an algorithm is sensitive and small changes to the community will lead to detection failure. But even if a method can succeed when half of a community is missing, which is impressive, we still see a non-trivial inference gap in Fig. 24.2.

Of course, realistic networks do not have constant degrees. What do these results look like for scale-free networks? Here we take $r_m = \delta(m, \mu)$ as before, but now $s_n \sim n^{-\lambda}$, with $\lambda \geq 2$. (Scale-free group sizes also model scale-free networks, as the degree distribution after projection remains scale-free, with the same exponent, although the maximum degree may increase.) It is known that the global connectivity of scale-free networks is robust to sampling errors when $2 < \lambda < 3$ (meaning that $p_c \to 0$). However, this result also requires that the maximum value K of the degree distribution be large ($K \gg 1$) [111]. Indeed, as we lower λ, we discover that, while our network is more robust under sampling, we are actually *less robust* when detecting the communities (Fig. 24.3)—overlapping structure vanishes earlier for smaller λ! Interestingly, increasing the maximum size of a community $N = \max\{n \mid s_n > 0\}$ does not make the overlapping structure more robust to node sampling.

So that's where partitions come from? When one ponders mechanisms for how community structure can appear in a network, it becomes clear that we should expect

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

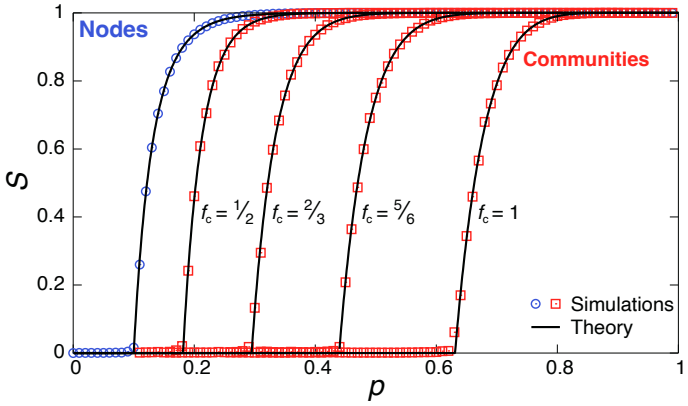

Figure 24.2 The size of the giant component S for the node and community networks as a function of node sampling rate p. Theory and simulations confirm that the network undergoes a transition from coupled to non-overlapping communities well before it loses global connectivity. Symbols represent node (⊙) and community (⊡) networks. Here we used $r_m = \delta(m, \mu)$, $s_n = \delta(n, \nu)$, with $\mu = 3$ and $\nu = 6$. Simulations used 10^6 nodes.

many types of networks to exhibit overlapping communities. Yet in network data we often find high-quality non-overlapping communities. The inference gap revealed here can in part explain this: the overlapping structure is present in the original network but not so easily seen in the sampled network.

24.4 Uncertain networks as probabilistic graphs

Thus far, we have focused most of our attention on the problem of missing nodes (although many site (node) percolation arguments translate to the related bond (edge) percolation problem). Here we go beyond missing nodes or edges by allowing for edges to be uncertain using *probabilistic graphs*. Such probabilistic graphs, while making simplifying assumptions, can capture both missing and spurious edges.

In a probabilistic graph, each edge $e = (i, j)$ is associated with a probability[1] $P(e)$ which we can use to reason about our uncertainties in edges. Assuming edges are independent, we arrive at an expression for the probability of the entire graph which we've encountered several times before,

$$P(G) = \prod_{e \in E} P(e) \prod_{e \notin E} [1 - P(e)]. \tag{24.24}$$

Simple edge sampling can generate a graph, call it G_s, by choosing to include each edge e with probability $P(e)$. Under such a process, what kind of graph statistics can we expect? What is the prevalence of triangles, for instance? What is the *expected* average shortest path length (ASPL)?

[1] A good source for $P(e)$ would be the posterior probabilities for edges estimated from the edge observer model; Sec. 23.3.

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

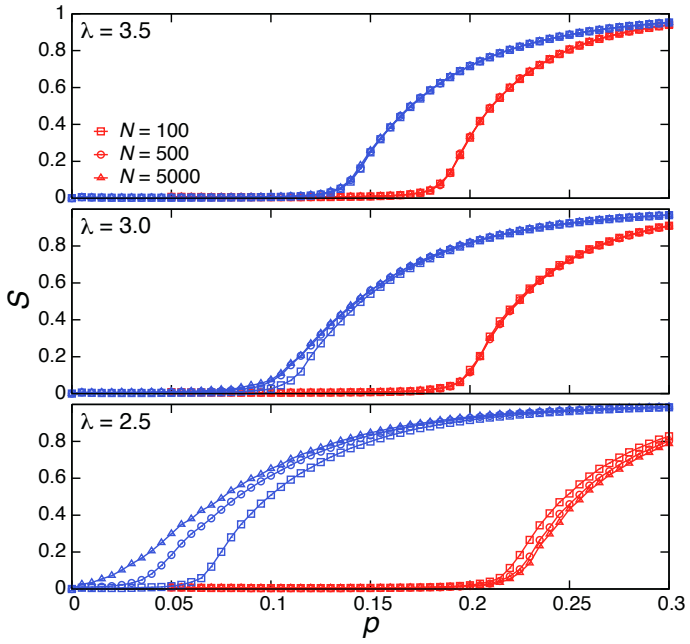

Figure 24.3 Sampling and the community structure of scale-free networks. Here $r_m = \delta(m, 3)$, $s_n \sim n^{-\lambda}$, $f_c = 1/2$, and $N := \max\{n \mid s_n > 0\}$. Increasing N and decreasing λ, measures known to improve the robustness of scale-free networks [111], actually *magnify* the inference gap. Simulations used 10^5 nodes. Figure from [28].

Shortest paths are emphasized in network measures for two reasons. First, they are easy to calculate in a given network using an algorithm such as breadth-first search or (for weighted networks) Dijkstra's algorithm. More importantly, they draw on the belief that heavily used, important paths—say, important for information flow—are short. But for an uncertain graph, we need to account for both the length of the path and *whether it actually exists*. Let ρ_{ij} denote a path between nodes i and j and $E(\rho)$ denote the set of edges comprising a path ρ. Then the probability $P(\rho_{ij})$ that ρ_{ij} exists in our probabilistic graph is

$$P(\rho_{ij}) = \prod_{e \in E(\rho_{ij})} P(e). \tag{24.25}$$

This says nothing about multiple paths; indeed, we expect more than one path can exist between a given pair of nodes. The *most probable path* between nodes i and j is $\rho_{ij}^{(\mathrm{MP})} = \arg\max_{\rho_{ij}} P(\rho_{ij})$. Treating the $P(e)$ as edge weights, we can in principle find ρ_{ij} using Dijkstra's algorithm on the probabilistic graph.

Finding the most probable path can tell us about the existence of paths (although it is a point estimate only), but what about whether the path is actually used? The prior belief is that shorter paths are more likely to be used than longer paths. We can capture this notion by assuming there is a constant "transmission rate" per edge, β, and transmissions along a path of length $|\rho| = \ell$ will then occur at rate β^ℓ, which

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

we call the "permeability." Thus, instead of focusing only on most probable paths, we combine with permeability[2] to find $\rho_{ij}^{(\text{MPP})} = \arg\max_{\rho_{ij}} \left(P(\rho_{ij})\,\beta^{|\rho_{ij}|}\right)$. Most Probable and Permeable (MPP) paths are also found efficiently with Dijkstra's algorithm using these different weights.

With MPP paths we now have an analog for shortest paths in an uncertain network. We can use the average MPP path length in place of the ASPL, and we can use MPP paths for betweenness centrality, allowing us to rank the centralities of nodes in probabilistic graphs.

Lastly, let's consider triangles in a probabilistic graph. Usually these are quantified with transitivity or the clustering coefficient (Ch. 12). The (deterministic) clustering coefficient for node a is given by $c_a = \Delta_a / \binom{k_a}{2}$. For a probabilistic graph, a triangle between nodes a, b, and c will occur with probability $P(e_{ab})P(e_{bc})P(e_{ca})$. Similarly, the probability for a two-path on those nodes through a is $P(e_{ab})P(e_{ac})$. We can use these to define an *expected* clustering coefficient for the probabilistic graph:

$$c_a = \mathbb{E}\left[\frac{T_a}{\tau_a}\right] \approx \frac{\mathbb{E}[T_a]}{\mathbb{E}[\tau_a]}, \tag{24.26}$$

where

$$\begin{aligned} \mathbb{E}[T_a] &= \sum_{\substack{b,c \in N_a, \\ b \neq c}} \left[P(e_{ab})P(e_{bc})P(e_{ca})\right], \\ \mathbb{E}[\tau_a] &= \sum_{\substack{b,c \in N_a, \\ b \neq c}} \left[P(e_{ab})P(e_{ac})\right], \end{aligned} \tag{24.27}$$

(and a better approximation than that given in Eq. (24.26) would incorporate the variances of T and τ, and their covariance).

24.5 Size estimation

One challenge with real network data is trying to use a limited sample of the network to learn about the unseen remainder of the network. Here we describe some strategies for inferring the total number of links,[3] what we call *size estimation*, when only a subgraph has been observed. This is useful both for understanding scientifically how *much* network we're dealing with, and logistically for marshalling our resources—if we are going to pay for experiments, it's good to know if we can expect to find, say, 1% of the data, or half the data.

Suppose we have a sample network $G_{\text{samp}} = (V_{\text{samp}}, E_{\text{samp}})$ and we wish to understand the complete network $G_{\text{full}} = (V_{\text{full}}, E_{\text{full}})$ (with $V_{\text{samp}} \subseteq V_{\text{full}}$ and $E_{\text{samp}} \subseteq E_{\text{full}}$) from the sample. Our sample G_{samp} is the subgraph of G_{full} induced by V_{samp}. We assume G_{full} is generated by some statistical model $P_\theta(G_{\text{full}})$ parameterized by θ and

[2] Pfeiffer and Neville [371] refer to most probable and permeable paths as maximum likelihood handicapped (MLH) paths.

[3] We focus on edges, but estimating the number of nodes is interesting too!

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

that G_{samp} is then sampled from it with probability

$$P_\theta(G_{\text{samp}}) = \sum_{G_{\text{full}} \supseteq G_{\text{samp}}} P_p(G_{\text{samp}} \mid G_{\text{full}}) P_\theta(G_{\text{full}}), \tag{24.28}$$

where p refers to a parameter of the sampling mechanism, which we assume is independent of the network's generating model.

Let us assume for now we know N_{full} and wish to estimate M_{full}. Such a situation is common when we know *a priori* the network nodes but it is prohibitive to confirm, observationally or experimentally, all possible interactions, i.e., all elements of the $N_{\text{full}} \times N_{\text{full}}$ $\mathbf{A}$. (If we did not assume N_{full} in Eq. (24.28), we would need to sum over more than all networks of N_{full} nodes.)

Suppose sampling depends only on the nodes and not on how they are connected. Then the sampling mechanism that appears in Eq. (24.28) factors into $P_p(G_{\text{samp}} \mid G_{\text{full}}) = Q_p(N_{\text{samp}}) n(G_{\text{samp}}, G_{\text{full}})$, where $Q_p(N_{\text{samp}})$ is the probability of sampling the observed nodes and $n(G_{\text{samp}}, G_{\text{full}})$ is the number of ways that the N_{samp} nodes can be sampled from N_{full}. Enumerating the ways the nodes can be sampled is tricky. If nodes are unlabeled and are all degree zero, it becomes simple, $q(G_{\text{samp}}, G_{\text{full}}) = \binom{N_{\text{full}}}{N_{\text{samp}}}$, but obviously this is a huge oversimplification. However, suppose every node is *identifiable*, uniquely labeled and distinguishable (and the labels are the same in G_{samp} and every possible G_{full}). Then there is only one way to choose the observed nodes, and $n(G_{\text{samp}}, G_{\text{full}}) = 1$. This assumption is not universally true, but quite reasonable: proteins, for instance, are all well identified by their open reading frames (ORFs), and we can expect the assumption to hold in many other contexts such as (some) social networks.

Under the assumptions of connection independence and node identifiability, the sampling mechanism is entirely described by $Q_p(N_{\text{samp}})$. For independent node sampling, meaning we now interpret the sampling parameter p as every node is independently sampled with probability p, we have $Q_p(N_{\text{samp}}) = p^{N_{\text{samp}}}(1-p)^{N_{\text{full}} - N_{\text{samp}}}$. Solving $\partial Q / \partial p|_{p=\hat{p}} = 0$ for $\hat{p}$ gives us the MLE $\hat{p} = N_{\text{samp}}/N_{\text{full}}$, which is quite intuitive.

Next, because nodes are identifiable (and not because they are independently sampled) the conditional probability

$$P_\theta(G_{\text{full}} \mid G_{\text{samp}}) = \frac{n(G_{\text{samp}}, G_{\text{full}}) P_\theta(G_{\text{full}})}{\sum_{G'_{\text{full}} \supseteq G_{\text{samp}}} n(G_{\text{samp}}, G'_{\text{full}}) P_\theta(G'_{\text{full}})}, \tag{24.29}$$

which comes from using Bayes' theorem on the factored $P_p(G_{\text{samp}} \mid G_{\text{full}})$, does not depend on p. It only depends on the model for G_{full}. Therefore, we need to make some assumptions about how G_{full} is generated.

Suppose π, the probability for an edge to appear in G_{full}, is well approximated by the density of the sample, i.e.,

$$\pi \approx \hat{\pi} = \frac{2M_{\text{samp}}}{N_{\text{samp}}(N_{\text{samp}} - 1)}. \tag{24.30}$$

From this, we have $M_{\text{samp}} / \binom{N_{\text{samp}}}{2} \approx M_{\text{full}} / \binom{N_{\text{full}}}{2}$, and we can solve for an estimate of

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

the unseen network's size,

$$\hat{M}_{\text{full}} = M_{\text{samp}} \frac{N_{\text{full}}(N_{\text{full}} - 1)}{N_{\text{samp}}(N_{\text{samp}} - 1)}. \tag{24.31}$$

While the assumptions so far are quite strong, and we should be skeptical, nevertheless Eq. (24.31) gives us a straightforward way to perform size estimation. Let's see it in an application.

Example Let's take these results and apply them to HuRI. This network, after removing self-loops, has $N_{\text{samp}} = 8{,}272$ nodes and $M_{\text{samp}} = 52{,}068$ edges. Luck et al. [283] build their experimental protocol around a PPI screening space of approximately 17,500 protein-coding genes: "To increase interactome coverage and generate a reference map of human binary PPIs, we expanded the ORFeome collection to encompass ~ 90% of the protein-coding genome." Indeed, the GENCODE Release 42 Human dataset lists 19,379 protein-coding genes. We use this value for N_{full}, yielding $\hat{M}_{\text{full}} = 285{,}787$ from Eq. (24.31). According to this, HuRI captures $M_{\text{samp}}/M_{\text{full}} = 18.22\%$ of the human interactome!

It's worth exploring some of the limitations of this estimation procedure.

Uncertainty in the number of nodes Some problems may give you information on the nodes of the full network independent of the sample, but many will not. How might our estimates of M_{full} change if we don't know N_{full}? Suppose our uncertainty in the now unknown N_{true} compared to N_{full} is $\epsilon \ll 1$ such that

$$N_{\text{full}} = (1 \pm \epsilon) N_{\text{true}}. \tag{24.32}$$

Then, $\hat{p} = N_{\text{samp}}/N_{\text{full}}$ becomes

$$\tilde{p} = \frac{N_{\text{samp}}}{N_{\text{true}}} = \frac{N_{\text{samp}}}{(1 \pm \epsilon) N_{\text{full}}} \approx (1 \mp \epsilon)\hat{p}. \tag{24.33}$$

(The approximation comes from a Taylor series for small ϵ: $(1 \pm \epsilon)^{-1} = 1 \mp \epsilon + O(\epsilon^2)$.) Using $\tilde{p}$ instead of $\hat{p}$ in

$$\hat{M}_{\text{full}} = M_{\text{samp}} \frac{N_{\text{full}}(N_{\text{full}} - 1)}{N_{\text{samp}}(N_{\text{samp}} - 1)} \approx M_{\text{samp}} \left(\frac{N_{\text{full}}}{N_{\text{samp}}} \right)^2 = \frac{M_{\text{samp}}}{\hat{p}^2} \tag{24.34}$$

shows us the effect of the error ϵ is

$$\tilde{M} = \frac{M_{\text{samp}}}{\tilde{p}^2} = \frac{M_{\text{samp}}}{(1 \mp \epsilon)^2 \hat{p}^2} \approx (1 \pm 2\epsilon) \frac{M_{\text{samp}}}{\hat{p}^2}. \tag{24.35}$$

In other words, an uncertainty of ϵ in the true network's number of nodes leads to an uncertainty of roughly 2ϵ in the estimated number of edges.

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

Uncertainty in measurements of edges Probably your observations will contain errors such as false positives (reported edges not actually present) and false negatives (non-reported edges actually present). The *true* number of edges $M = M_{\text{TP}} + M_{\text{FN}}$, where M_{TP} is the number of true positive edges and M_{FN} is the number of false negative edges. Likewise, the *observed* number of edges $\tilde{M} = M_{\text{TP}} + M_{\text{FP}}$, where M_{FP} is the number of false positive edges. Suppose your experimental or observational process has been validated so it has known *precision* and *recall*:

$$\text{Precision} = \frac{M_{\text{TP}}}{M_{\text{TP}} + M_{\text{FP}}}, \qquad \text{Recall} = \frac{M_{\text{TP}}}{M_{\text{TP}} + M_{\text{FN}}}. \tag{24.36}$$

The precision (or positive predicted value) tells us many edges detected by our observations were true while recall (or true positive rate) tells us how many true edges were detected by our observations. These relate nicely to M and $\tilde{M}$, allowing us to estimate $\hat{M} \approx M$ given $\tilde{M}$ and the observation process's precision and recall:

$$\frac{\text{Precision}}{\text{Recall}} = \frac{M_{\text{TP}} + M_{\text{FN}}}{M_{\text{TP}} + M_{\text{FP}}} = \frac{M}{\tilde{M}} \quad \Rightarrow \quad M \approx \hat{M} = \frac{\text{Precision}}{\text{Recall}}\tilde{M}. \tag{24.37}$$

Other sampling mechanisms What if nodes are not independently sampled at a constant rate? We can model a sampling mechanism where nodes are sampled independently but non-uniformly. Let p_i be the probability that node i is sampled and assume the values of p_i for different nodes i are not equal but are drawn from the same distribution,

$$p_i \sim D(\beta)\ \forall i, \tag{24.38}$$

where β is some parameter(s) for the sampling rate distribution D. Since nodes are still sampled independently, the probability for an edge to be sampled is now $\pi_{ij} = p_i p_j$. Under these assumptions, the expected value of $\hat{p}$ converges to the expected value of p_i and the variance of $\hat{p} \to 0$ for $M_{\text{full}} \to \infty$ making $\hat{p}$ an unbiased and consistent (converges to the true value) estimator. A similar argument holds for π_{ij}, allowing us to proceed with inference. We can even relax this further by assuming that the rate at which node i is sampled depends in some way on i. We capture this by assuming $p_i \sim D_i(\gamma_i; \beta)$, where γ_i parameterizes how the information related to i changes the distribution of p_i. Such information could be related to i's network properties, such as the degree or clustering, or it could be related to non-network attributes. If we assume the network is uncorrelated given these parameters such that $P(\gamma_i, \gamma_j) = P(\gamma_i)P(\gamma_j)$ for edge i, j and nodes are drawn independently given the probabilities p_i, then we can once again show [448] estimator $\hat{p}$ is unbiased and consistent.

Much work continues on estimating the size of networks. In PPI networks such as HuRI, for instance, it's common to estimate the unseen network's size by equating the densities (Eq. (24.30)) but only across edges of a very thoroughly studied subgraph. This is commonly done using *literature curated* data, the set of edges extracted by analyzing preexisting studies; the argument being that those interactions are more heavily investigated and replicated. For HuRI, Luck et al. [283] include this PPI network, which they call "Lit-BM." Luck et al. [283] themselves provide an estimate of 2–11% coverage for HuRI, less coverage than our estimate of 18% but we are not terribly far

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

off. It is on the whole much more probable to overestimate coverage than underestimate it, and we should in general be prepared for such.

24.5.1 Size estimation from capture–recapture

(This approach to size estimation works when you have access to multiple networks, such as from different experiments or temporal snapshots.)

Capture–recapture (also called mark and recapture) is an idea used for population estimation in ecological studies [14, 61]. Imagine you are trying to count the number of animals that live in a given area. You can go out and capture them with traps, but it will not be possible to capture all of them, especially at the same time. How then can you count the population?

At first glance, this sounds like an impossible problem, but there is a lovely way to address it. Suppose we have a population of unknown size M and we *capture* a sample of M_1 individuals from that population. We tag each individual somehow with a marker that we assume will stay affixed, then release the captured sample. Later, we repeat the capture process exactly as before and capture a second sample of size M_2. Let the number of individuals tagged in sample 1 who were *recaptured* in sample 2 be M_{12}. If we make some assumptions, like that tags don't fall of individuals but also that the two captures are independent from one another, then we have a way to infer M.

The *Lincoln–Petersen* (L–P) *estimator*, which was the impetus for capture–recapture, recognizes that if individuals are equally likely to appear in either sample, then the proportion of tagged individuals found in sample 2 should be equal to the proportion of the total population that was tagged in sample 1, or

$$\frac{M_{12}}{M_2} = \frac{M_1}{M} \;\Rightarrow\; \hat{M} = \frac{M_1 M_2}{M_{12}}, \tag{24.39}$$

which we solved for M to estimate the unknown population size $\hat{M} \approx M$. If we also want to compute confidence intervals, it's helpful to have the variance of the estimator [14],

$$\mathrm{Var}\left(\hat{M}\right) = \frac{(M_1+1)(M_2+1)(M_1-M_{12})(M_2-M_{12})}{(M_{12}+1)^2(M_{12}+2)}. \tag{24.40}$$

An estimate with a 95% confidence interval, say, can now be given by $\hat{M} \pm 1.96\sqrt{\mathrm{Var}\left(\hat{M}\right)}$.

Equation (24.39) is a simple, brilliant idea. It can be extended and generalized in many ways, for example going from 2 to K measurements, and considerable work has focused on understanding and improving upon it, especially for small sample sizes. It doesn't work perfectly, though; its assumptions can be restrictive and difficult to validate. For the estimate to be accurate, we need four ingredients: captures are independent, all individuals are equally likely to be captured, population size is constant during captures, and tags remain affixed. These seem innocuous, but imagine yourself a soaking-wet biologist, stomping through the woods in a downpour, trying to find an elk who just ran off with a loose tag—you may be quite skeptical of iid capture probability![4]

[4] Indeed, wildlife biologists and ecologists have long debated the accuracy of such estimates [14].

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

What can the L–P estimator tell us about networks? Our notation in Eq. (24.39) is suggestive. Imagine each edge is a member of the population we are estimating. We observe network edges in one experiment, then repeat the experiment and reobserve them. Using the intersection of the edges, we can estimate the total number of edges M from Eq. (24.39). Thus we have another protocol for size estimation.

Example Let's illustrate the L–P estimator using the Malawi Sociometer Network. This is a dynamic network (Ch. 15) and we can use the edge events (Sec. 15.2) to "simulate" two samples. First, divide the full set of edges into two sets:

$$\begin{aligned} E_1 &= \{(u,v) \mid (u,v,t_i) \in \text{events}, t_i \le t_*\}, \\ E_2 &= \{(u,v) \mid (u,v,t_i) \in \text{events}, t_i > t_*\}, \end{aligned} \tag{24.41}$$

where we take $t_* = \max_i t_i/2$. Then, using $M_1 = |E_1|, M_2 = |E_2|, M_{12} = |E_1 \cap E_2|$ in Eqs. (24.39) and (24.40) gives a size estimate of $\hat{M} = 396.97 \pm 22.12$ edges. In terms of the full number of observed edges, this means we estimate[5] the Malawi Sociometer Network data to capture $87.41 \pm 4.87\%$ of edges. Good coverage.

Probably the most pressing concern for using the L–P estimator on networks is the assumption that all edges are equally likely to be observed. (That edges are observed independently is also important.) This has been shown to not hold in real-world problems, such as estimating edges in the Internet topology [403]. To overcome this, Roughan et al. [403] introduce a "stratified" model by assuming that edges fall into one of C *classes* and the capture probability is different between classes but the same for all edges within a class. We describe this approach now.

First, suppose we know the class of a given edge. If we take K iid measurements, then the model's probability that we observe an edge in class j a total of k times is

$$\Pr(k \mid K, p_j) = \binom{K}{k} p_j^k (1-p_j)^{K-k}, \tag{24.42}$$

where p_j is the observation probability for a j-class edge. The observation probability could be estimated with the MLE

$$\hat{p}_j = \frac{\sum_{i \in C_j} k_i}{|C_j| K}, \tag{24.43}$$

where k_i is the number of times edge i was observed and C_j is the set of links in class j.

But Eq. (24.43) breaks down because in practice we only have the j-class edges that were observed at least once. If we knew all the edges in class j, we would have already solved the size estimation problem!

In other words, we actually have the conditional distribution

$$\Pr(k \mid k > 0, K, p_j) = \binom{K}{k} \frac{p_j^k (1-p_j)^{(K-k)}}{1-(1-p_j)^K}, \tag{24.44}$$

[5] Here we use error propagation [455] on a ratio $r = M/(\hat{M} \pm \delta M)$ to get $r \pm \delta r$.

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

known as the truncated binomial distribution. If we estimate its parameter, we can then estimate the number of hidden ($\Pr(k = 0)$) edges. The MLE $\hat{p}_j$ for Eq. (24.44) satisfies [396]

$$M^{(\text{obs})} K p_j = \left(1 - (1 - p_j)^K\right) \sum_{i=1}^{M^{(\text{obs})}} k_i, \tag{24.45}$$

which we can solve numerically, where $M^{(\text{obs})}$ is the number of observed edges. With $\hat{p}_j$ solved from Eq. (24.45), we can estimate the total number of edges with the MLE

$$\hat{M}_j = \frac{M^{(\text{obs})}}{1 - (1 - \hat{p}_j)^K}. \tag{24.46}$$

Lastly, we can iterate through each class j to derive our total estimate.

So far, we assume the class of a given edge is known, which is not realistic. We can address this using expectation–maximization (EM) to simultaneously estimate edge class and class parameters.

The EM method, which we also saw in the edge observer model (Sec. 23.3), iterates between (E-step) averaging a latent (hidden) variable (in our case, edge classes) and (M-step) finding model parameters by maximizing a likelihood. To use EM, we first extend the model to capture how edges fall into classes. We do this with two new statistical parameters: w_j, the proportion of edges in class j, and $c_j^{(i)}$, the probability that edge i belongs to class j. We describe the model in terms of EM in Alg. 24.1. After fitting, the model can also categorize the edges: we estimate the class of edge i to be $\arg\max_j c_j^{(i)}$, the most probable class. And the number of edges in class j is given by $\hat{M}_j = M_j^{(\text{obs})} \big/ \left(1 - (1 - \hat{p}_j)^K\right)$, and we have our size estimates.

One wrinkle. The number of classes C is now a free parameter. The more classes we have, the more complex the model is, up to the extreme of a single class for every edge. Thus we are forced into a tradeoff between model fit and model simplicity. This is a classic problem in model selection, and one way to tackle it is with the Akaike

Algorithm 24.1 The EM algorithm for estimating the parameters of the multi-class model, where C is the number of classes. Here we use a uniform initialization, but the choice is not too important. For a convergence condition, continue iterating until the total change in $\hat{p}_j$ from one iteration to the next is less than, say, 10^{-6}.

$\hat{p}_j \leftarrow j/(C+1)$, $w_j \leftarrow 1/C$ ▷ *Initialization (uniform)*
while (not converged) **do**
  $c_j^{(i)} \leftarrow \hat{w}_j \Pr(k_i \mid K, \hat{p}_j)$ (Eq. (24.42)) ▷ *E-step*
  for $j = 1$ to C **do** ▷ *Start M-step*
    while (not converged) **do**
      $\hat{p}_j \leftarrow \left(1 - (1 - \hat{p}_j)^K\right) \sum_i k_i c_j^{(i)} \Big/ \left(K \sum_i c_j^{(i)}\right)$
    $\hat{w}_j \leftarrow \sum_i c_j^{(i)} / \left(M^{(\text{obs})} (1 - (1 - \hat{p}_j)^K\right)$
  ▷ *End M-step*

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

Information Criterion (AIC), given by

$$\text{AIC} = 2\eta - 2\ln(\hat{L}), \tag{24.47}$$

where η is the number of parameters and $\hat{L}$ is the maximum likelihood of the model. We can then compare models by plotting AIC as a function of η or, in our case, as a function of C (as C determines the total number of parameters).

In general, size estimation in networks is quite interesting, and attempting to probe beyond the certainty of a limited sample into the unknown is exciting. Unfortunately, progress is usually made by taking on some heavy assumptions, and these should make us skeptical. For example, we usually assume edges are sampled independently. But it may be the sampling mechanism is biased towards motifs or other structures; it's certainly the case that edges won't exist independently in the network. That said, experiments such as assays testing for protein interactions (à la HuRI), may meet this assumption. But, in general, don't be too surprised if your estimates (and CIs) are off by noticeable factors.

24.6 Other approaches

The edge observer model described in Sec. 23.3 also fits nicely into our suite of network error analysis methods. (Indeed, such models are known as *observer error models*.) It attempts to estimate the most likely set of edges from repeated, noisy observations, and its estimates of the probabilities of edges can serve as the base for the probabilistic graphs described in Sec. 24.4. Lastly, like the multi-class capture–recapture method in Sec. 24.5.1, it uses the EM algorithm for parameter fitting.

But the EM algorithm gives us point estimates for our parameters. We may, when interrogating uncertainty, be better served with a full Bayesian treatment—in essence, beginning from the same model but then generating samples from the posterior, $P(\mathbf{A}, \theta \mid \text{data})$ (Eq. (23.8)) using, for example, MCMC. The main difference is that instead of one estimate for the posterior probability for an edge to exist, (Q_{ij}, Eq. (23.25)), we would have a distribution, and we can assess our per-edge uncertainty based on how wide or narrow that distribution turned out to be. Young et al. [503] pursue this in depth.

24.7 Summary

The fundamental challenge of measurement error in network data is capturing the error-producing mechanism accurately and then inferring the unseen network from the (imperfectly) observed data. Computational approaches can give us clues and insights, as can mathematical models. Mathematical models can also build up methods of statistical inference, whether in estimating parameters describing a model of the network or estimating the network's structure itself. But such methods quickly become intractable without taking on some possibly serious assumptions, such as edge independence. Even without addressing the full problem of network inference, we can still explore features of the unseen network, such as its size, using the available data.

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

Bibliographic remarks

For an introduction to uncertainty quantification in general, we can recommend Smith [435] and Sullivan [451]. A long thread of computational studies of network measures under uncertainty includes Borgatti et al. [67], Frantz et al. [169], and Martin and Niemeyer [294] but sociologists have been concerned for far longer; see, for example, Granovetter [189]. Percolation has been one of the main mathematical tools for understanding missing nodes and edges. The percolation condition for a random network (Eq. (24.9)) was first derived by Cohen et al. [111]. Likewise, the calculation showing how sampling may make an overlapping community structure appear to be non-overlapping was studied in Bagrow et al. [28] based on the model and calculation of Newman and Park [339]. Pfeiffer and Neville [371] introduced the probabilistic graph analysis we describe in Sec. 24.4.

Size estimation has been of interest to researchers studying PPI networks for some time [448, 413]. The approach we describe here was introduced by Stumpf et al. [448]. Readers interested in learning more about capture–recapture sampling, which we applied to a dynamic network to estimate our coverage of edges, may start with Amstrup et al. [14] or Böhning et al. [61]. Basic capture–recapture can be extended, and we describe the model of Roughan et al. [403] who allow for different edges to be captured with different probabilities. Their model was proposed in the context of estimating the size of the Internet (at the Autonomous Systems level), a particularly interesting application of size estimation.

Exercises

24.1 What is the probability that a node with degree $k = 1$ appears to have degree $k = 0$ in a network with iid node sampling and a p sampling rate? What can this tell us about missing data?

24.2 Suppose every edge e in a network is sampled with constant probability $P(e) = p$. What is the expected transitivity (Eq. (12.9))?

24.3 Size estimation with capture–recapture requires data from independent experiments or from different time periods. If we take a single network, randomly divide it into two parts in an attempt to simulate two capture periods, and apply the Lincoln–Petersen estimator, we will fail to find useful size estimates. Why?

24.4 (**Focal network**) Use a modularity optimization method to find a high-quality partition of the Malawi Sociometer Network. (For simplicity, take the weighted version and ignore edge weights.)

(a) Report modularity Q, the number of communities, and the mean and median community size.

(b) Now, suppose the network is under-observed, meaning edges are missing. Simulate this missingness with iid edge sampling and a $p = 1/2$ sampling rate. Make a computer function `subsample(G,p)` that takes the network

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press

and the edge sampling rate as input and returns a subsampled copy of the network.

(c) Apply `subsample` independently to the original network 100 times. For each sampled network, reapply the modularity optimization and record its Q. Report the distribution of Q over the subsampled realizations. How does the modularity of the sampled networks compare to that of the original?

24.5 (**Focal network**) Repeat the analysis of Ex. 24.4 but vary the edge sampling rate p and report the mean Q as a function of p. Interpret the dependence between Q and p.

https://doi.org/10.1017/9781009212601.029 Published online by Cambridge University Press