Random Graphs

Generative Models

- Could hope to understand networks in the real world if we had descriptions of the processes that generate them.
- Describe such processes via random algorithms.
- The likelihood of generating a given graph via algorithm induces a probability distribution.
- Examine the properties of generated graphs; try to find models whose properties match those of the networks we see.

Generative Models as Hypotheses

- Given many graph samples, can test the empirical distribution against our model distribution.
- Similarly, can test empirical distribution of an induced statistic against our model distribution.
- Given a single empirical sample can use ordered/centered statistics to reject a model (if our observed statistic is extreme).

Erdos-Renyi random graphs

The most studied and well-known random graph model.

The algorithm

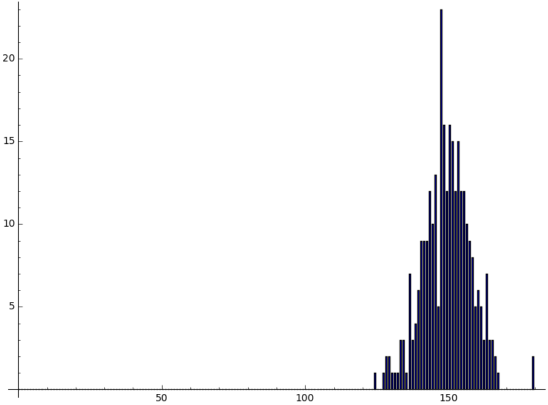
- Choose a number of vertices n.
- Choose a probability p.
- For each possible edge, add it with probability p (and thus omit it with probability $1 - p$.)

The associated probability distribution is often denoted $G(n, p)$.

Movie - Generating a G(n,p) random graph (n=12, p=0.5)

Properties of Erdos-Renyi Random Graphs

- A lot more is know about $G(n, p)$ than we'll discuss here.
- Average degree: $(n-1)p$
- Degree distribution example (n=300, p=0.5):



- Clearly not a good model for sparse networks, or those with fat-tailed degree distributions.

Properties of Erdos-Renyi Random Graphs

- Famous result by Erdos and Renyi: the connectivity of random graphs is strongly controlled by np.
- We'll say a property holds **almost surely** for a sequence of distributions depending on n if the probability of it holding goes to 1 as $n \to \infty$.
- Note that a property holding almost surely is a statement about the sequence of models as we vary n.

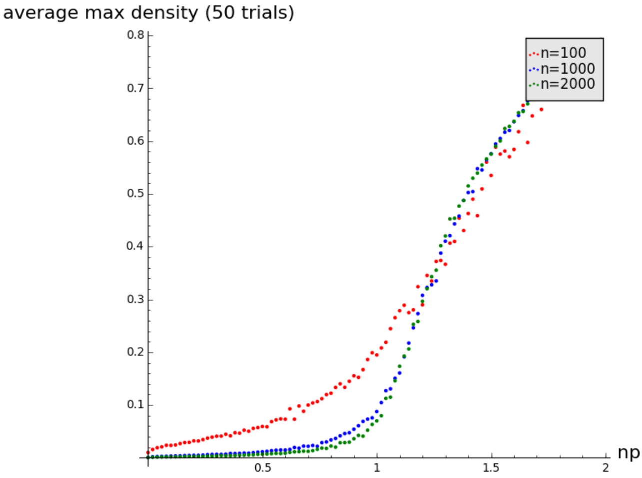
Theorem

In the $G(n, p)$, holding np fixed:

- *If $np < 1$,* **components are small***: for fixed $\alpha > 0$, there are almost surely no connected components of size greater than $\alpha \log(n)$.*
- *If $np > 1$,* **a giant component emerges***: there exists $\beta > 0$ such that almost surely there exists a component of size at least βn.*

Properties of Erdos-Renyi Random Graphs

In applications, would often also care about the speed of convergence:



Zero-One Law for G(n,p) (Fagin)

- Use first-order logic to write down potential properties of graphs, using adjacency and equality as predicates.
- Example 1: to express the property of having an edge: $\exists u \exists v (u \sim v)$.
- Example 2: to express the property of having minimum degree 2: $\forall u \exists v \exists w ((u \sim v) \wedge (u \sim w) \wedge (v \neq w))$

Theorem

Given a fixed first-order sentence S and fixed $p \notin \{0, 1\}$, as $n \to \infty$ either S or $\neg S$ holds almost surely for $G(n, p)$.

FACT: There exists an infinite graph R, called the Rado graph, for which S holds iff it holds almost surely in $G(n, p)$ as above.

The Regularity Lemma (Szemeredi)

- Even if our real life networks are far from the $G(n,p)$ model, we might hope to find modules in these networks such that the interconnections between modules appear random with a given density.
- Szemeredi's regularity lemma says roughly that by taking large enough graphs, we can find lots of modules such that almost all module pairs have close-to-random interconnections.
- Unfortunately, to get reasonable bounds the graphs must be taken to be impractically large.

$G(n, m)$ vs $G(n, p)$

If $m < n$ is a natural number, we have a closely related model $G(n, m)$:

$G(n, p)$ algorithm

Start with n vertices and no edges. Select a missing edge uniformly at random and add it. Repeat m times.

- The number of edges in $G(n, m)$ is always m, whereas the number of edges in $G(n, p)$ varies, but is tightly clustered around $\binom{n}{2}p$.
- $G(n, p)$ is usually easier to reason about, since each edge choice is independent. Given $m \approx \binom{n}{2}p$, the models have similar properties.

Configuration Model

- One of the basic properties of real-world networks we want to replicate is degree distribution.
- Given a listing of the desired degrees of all vertices in a network, we can randomly select a graph with exactly those degrees.
- Here's a naive random algorithm: sample uniformly from the space of all networks with n vertices (i.e. $G(n, 0.5)$), and throw away any sample not having exactly the right degrees. (This is impractical.)
- The configuration model does better, but sacrifices being an exactly uniform sample:

Configuration Model

Configuration model algorithm

Given desired degrees $d_1, \ldots, d_n$ (which must sum to an even number):

- Take n vertices, where vertex i has d_i "stubs" attached.
- Choose 2 distinct stubs uniformly at random. Remove the stubs, and replace them with an edge between those vertices.
- Repeat until all stubs are gone.

Note: This pairing process might create multiedges or loops; if these are not desirable, we can resample until we find a graph without those properties.

Movie? - Configuration model (1,1,2,2,2,3,3)

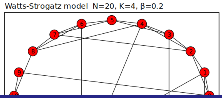
Watts-Strogatz Model

This model attempts to create graphs with high clustering coefficients and low path lengths.

Watts-Strogatz algorithm

Given a desired number of vertices N, average degree K (assumed even), and probability p:

- Construct a circle of N vertices where each vertex is connected to it's K closest neighbors.
- Iterate through the nodes in circular fashion, and for each node i iterate through its edges (i, j) such that $i < j$ in increasing fashion.
- As each edge (i, j) iterated through, replace it with probability p by another edge (i, k) chosen uniformly at random from all missing edges.



Barabasi-Albert Model

This model attempts to replicate real-world power-law degree distributions via a simple mechanism. It also has relatively low path lengths.

Barabasi-Albert algorithm

Given an initial graph size M, a connection number m, and a stopping time T:

- Start with M fully connected nodes.
- Add a new node and iteratively connect it m times to existing nodes.
- Each time, choose the node to connect to weighted by the ratio of its degree to the total degree of the graph (??)
- Repeat T times.

Movie - Barabasi-Albert Model (M = 5, m = 2, T = 100)

Other Ideas

- Directed versions of the models we've discussed also exist.
- Weighted random graphs can be generated by, for instance, choosing a distribution besides Bernoulli for each edge independently.
- Instead of probabilistically choosing edges from $G(n, p)$, we could choose from some geometric lattice or the graph induced by a triangulation.
- Flow sampling: given a vector field on a compact space, we can cut the space into small elements, take each element as vertex, and add weighted directed edges by sampling flow lines starting at a random location and continuing for time T. We can do discretized flow sampling even if our vector field/walk process is random.

End