Learning from Networks

Significance and Random Graphs

Fabio Vandin

October 31st, 2024

Motivation

Assume you computed some feature of a graph, e.g. the clustering coefficient.

Now you want to understand if it provides an *interesting* information.

Published: 04 June 1998

Collective dynamics of 'small-world' networks

Duncan J. Watts & Steven H. Strogatz

Nature **393**, 440–442 (1998) | Cite this article

Table 1 Empirical examples of small-world networks

	L_{actual}	C_{actual}
Film actors	3.65	0.79
Power grid	18.7	0.080
C. elegans	2.65	0.28

Ideas?

One Solution

Compare your observation with the one obtained from a *random network*.

Intuition: you will understand if what you measured is not explained by the characteristics of a random network $\Rightarrow$ it is interesting!

Published: 04 June 1998

Collective dynamics of 'small-world' networks

Duncan J. Watts & Steven H. Strogatz

Nature **393**, 440–442 (1998) | Cite this article

Table 1 Empirical examples of small-world networks

	L_{actual}	L_{random}	C_{actual}	C_{random}
Film actors	3.65	2.99	0.79	0.00027
Power grid	18.7	12.4	0.080	0.005
C. elegans	2.65	2.25	0.28	0.05

Simple Examples

Comparison with Random Networks

We need to describe:

- how do we **compare** the observation in the real network with the observation from a random network?
- how do we **compute** the measure on a random network?

Comparison with Random Networks

Scenario:

- you have measured the clustering coefficient for your network;
- you can compute the clustering coefficient for a random network

What would you do to understand if the clustering coefficient of your network is interesting?

Assessing Significance

We are interested in verifying (actually, *falsifying*) whether simple characteristics of the graph *explain* our measured feature $\mathcal{F}$ (e.g., the clustering coefficient).

We use a framework corresponding to *statistical hypothesis testing*.
Note: we present it for our scenario, not the most general version.

valore di feature F osservato in rete reale

$\Uparrow$

We assume the *null hypothesis* H_0 that f *well conforms* with the distribution observed in a *random graph*

random graph: graph taken uniformly at random among all graphs (with $|V|$ vertices and) with the required simple characteristics.

Example

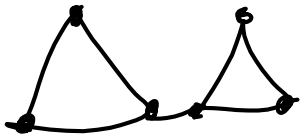
Given a graph $G = (V, E)$, we compute its clustering coefficient $cc(G)$

Question: is $cc(G)$ explained by the number $|E|$ of edges of G?

null hypothesis H_0: $cc(G)$ *well conforms* with the distribution of $cc(G)$ in a *random graph*

random graph: graph taken uniformly at random among all graphs with $|V|$ vertices and $|E|$ edges (or where the expected number of edges is $|E|$).

G:

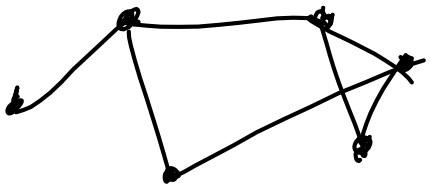


$$\Rightarrow cc(G) = \frac{2}{\binom{6}{3}} = \frac{2}{20} = 0.1$$

random graph con 6 nodi e 7 lati

$\Downarrow$

G':



$$\Rightarrow cc(G') = \frac{1}{\binom{6}{3}} = \frac{1}{20} = 0.05$$

anche G è random graph

Measures of Significance

How do we assess whether the *null hypothesis* H_0 is true or not?

Note: given that we are considering *random graphs*, our observation is (almost) always obtainable in *at least one or few* random graphs

Needed: way to *quantitatively* assess how *likely* it is that the value f of feature $\mathcal{F}$ arises when H_0 is true.

Commonly used measures:

- *z-score*
- *p-value*

z-score

Let f be the measure of the feature $\mathcal{F}$ of interest.
Let $X_{\mathcal{F}}$ be the *random variable* corresponding to the value of the feature $\mathcal{F}$ when the null hypothesis H_0 is true, i.e. in a random graph

Definition

The z-score of H_0 is:

$$\frac{f - \mathbb{E}[X_{\mathcal{F}}]}{\sigma[X_{\mathcal{F}}]}$$

Notes:

- the expectation $\mathbb{E}[X_{\mathcal{F}}]$ and standard deviation $\sigma[X_{\mathcal{F}}]$ are w.r.t. the distribution of random graphs given by the null hypothesis H_0
- the z-score can be positive or negative

p-value

Measures the *probability* of observing a value for feature $\mathcal{F}$ *at least as extreme* as f when the null hypothesis H_0 is true.

Meaning of *"at least as extreme"* depends on the feature and the specific situation.

Example:
Given the clustering coefficient $cc(G)$, *"at least as extreme"* could be:

- $\geq cc(G)$
- or $\leq cc(G)$

p-value (continue)

Definition

The p-value $p(H_0)$ of H_0 is

$$p(H_0) = \mathbb{P}[X_{\mathcal{F}} \text{ is at least as extreme as } f | H_0 \text{ is true}]$$

Notes:

- if we are interested in understanding if the value of f is *higher* than expected (when H_0 is true):
  $p(H_0) = \mathbb{P}[X_{\mathcal{F}} \geq f | H_0 \text{ is true}]$
- if we are interested in understanding if the value of f is *lower* than expected (when H_0 is true):
  $p(H_0) = \mathbb{P}[X_{\mathcal{F}} \leq f | H_0 \text{ is true}]$
- if $p(H_0)$ is small it is unlikely that the observed value f is obtained when H_0 is true

Example

Small Digression

Usually the p-value is used to *reject* a null hypothesis H_0 using the following rule:

Rejection rule: given a value $\alpha \in (0, 1)$, reject H_0 if $p(H_0) \leq \alpha$.

Rejecting H_0: flagging feature f as *significant*

The following (easy to prove) proposition provides the ground for the rejection rule above.

Proposition

If the rejection rule above is used, then $\mathbb{P}[H_0 \text{ rejected} | H_0 \text{ true}] \leq \alpha$.

Small Digression (continue)

Note that in statistical hypothesis testing two errors are possible:

- H_0 is true *but* H_0 is rejected (*type-I error* or *false positive*)
- H_0 is not true *but* H_0 is not rejected (*type-II error* or *false negative*)

The previous proposition shows that the **rejection rule** provides guarantees on false positives.

Recap

Up to know: how to compare the observation in the real network with the observation from random networks.

Still missing: what is the *actual* definition of random graph?

From before: graph taken uniformly at random among all graphs (with $|V|$ vertices and) with the required simple characteristics.

In practice?

Erdős-Rényi Random Graphs

Idea: the simple characteristics that is preserved is the *number of edges* in the graph.

Definition

A random graph from the $G(n, m)$ model is a graph chosen uniformly at random among all graphs with n vertices and m edges.

Note: the vertices are implicitely considered *labelled*

Introduced by Paul Erdős and Alfréd Rényi in 1959.

Example

Erdős-Rényi-Gilbert Random Graphs

Similar to the previous model, but the number of edges is preserved only *in expectation*.

Definition

A random graph from the $G(n, p)$ model is a graph with n vertices and where each edge appears with probability p independently of all other events.

Introduced by Paul Erdős, Alfréd Rényi, and Edgar Gilbert in 1959.

Example

Erdős-Rényi-Gilbert Random Graphs (continue)

Proposition

Let $m(n, p)$ be the number of edges of a random graph from $G(n, p)$. Then

$$\mathbb{E}[m(n, p)] = \binom{n}{2} p$$

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

Informally: if $p = \frac{m}{\binom{n}{2}}$, then $G(n, m)$ and $G(n, p)$ are relatively similar models.

Note:

- the two models are not identical, but several properties are the same
- $G(n, p)$ is usually preferred because it allows for easier analytical results.

Example

What is the expected clustering coefficient of a random graph from $G(n, p)$?

Erdős-Rényi-Gilbert Random Graphs (continue)

However, not all properties can be easily computed analytically!

Note: depending on the analysis/property/measure, we need

- the *expectation* and *standard deviation* (e.g., for z-score)
- the *distribution* (e.g., for p-value)

Example

What Now?

Monte-Carlo Approach

Let f be the measure of the feature $\mathcal{F}$ of interest on your real network.

If you can *generate* a random graph, then you use the following *Monte-Carlo* approach:

1. generate P instances $G_1, G_2, \ldots, G_P$ of a random graph
2. compute the measure of interest on the P instances: $\mathcal{F}(G_1), \mathcal{F}(G_2), \ldots, \mathcal{F}(G_P)$
3. use $\mathcal{F}(G_1), \mathcal{F}(G_2), \ldots, \mathcal{F}(G_P)$ to estimate the z-value (with estimated expectation and standard deviation) or p-value for f

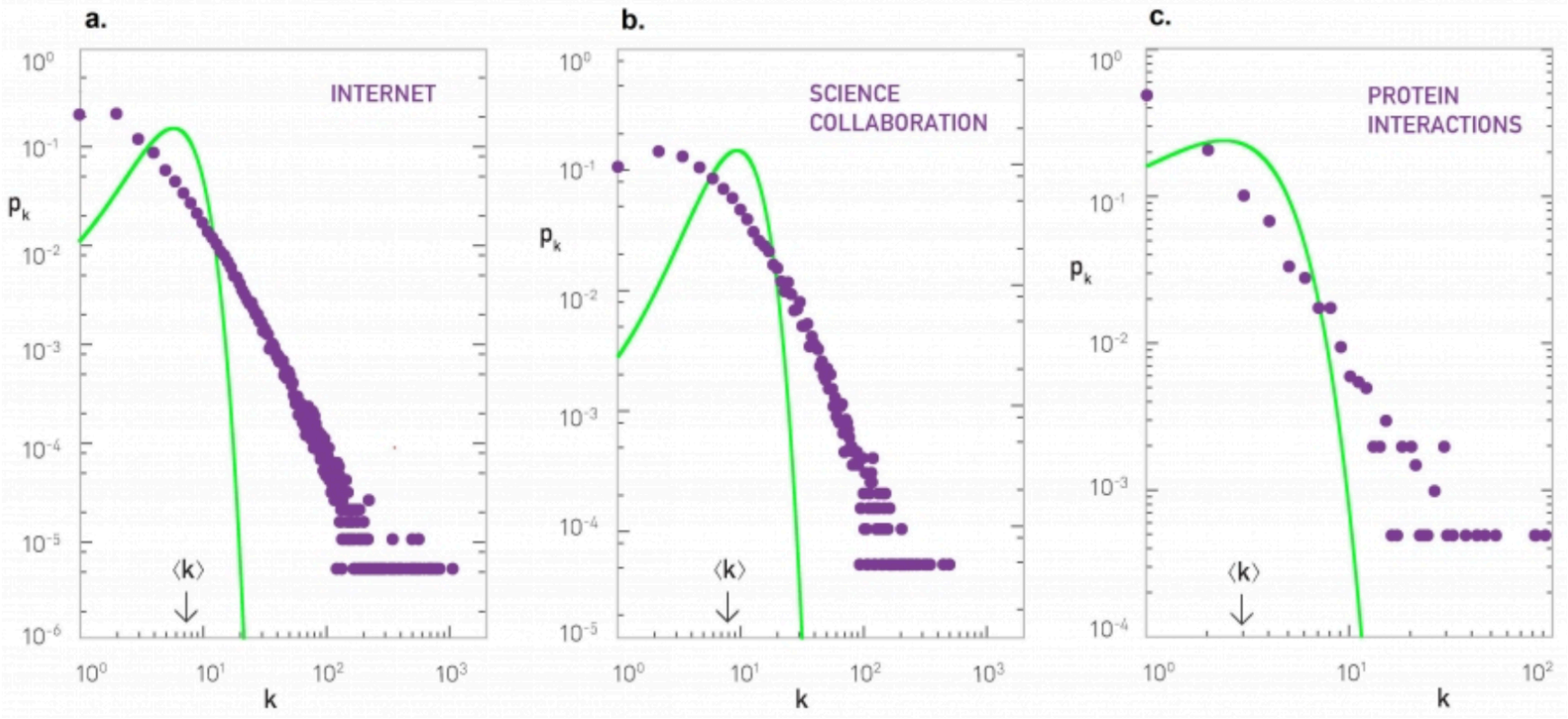
Notes:

- can be very expensive computationally!
- it is embarrassingly parallel

Example

Back to $G(n, p)$

While useful, $G(n, p)$ does not resemble real world networks.



Random Graphs with Given Degree Sequence

Problem: given a graph G, you want to generate a random graph where each vertex has the same degree as in G.

More formally: let

- $V = \{1, \ldots, n\}$
- $d_i = deg(i)$ the degree of vertex i in G

Given: d_i for $i = 1, 2, \ldots, n$
Goal: generate a random graph $G' = (V, E')$ where the degree of vertex i is d_i.

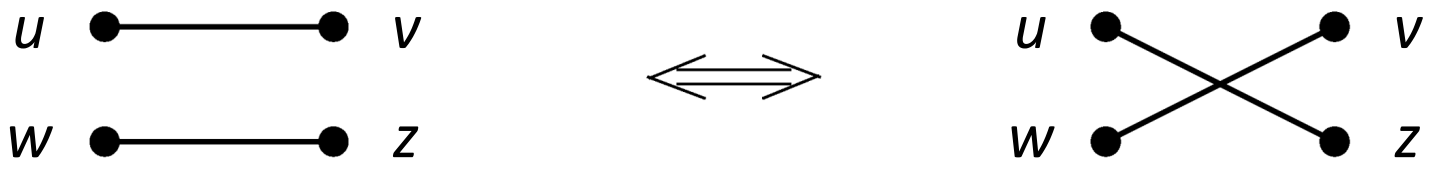
Ideas?

Random Graphs with Given Degree Sequence (continue)

Idea:

- start from G
- *swap* pairs of edges in G while preserving the degree of each vertex
- repeat the previous steps many times

Edge swap operation:



Random Graphs with Given Degree Sequence (continue)

Algorithm `RandomGraphFixedDegrees`(G, k)
Input: graph $G = (V, E)$; $k \in \mathbb{N}^+$
Output: random graph G' where each vertex has the same degree as in G

$E' \leftarrow E$; $G' = (V, E')$;
for $i \leftarrow 1$ *to* k **do**
  sample edges (u, v) and (w, z) from E' uniformly at random;
  if $(u, z) \notin E'$ *and* $(w, v) \notin E'$ **then**
    $E' \leftarrow E' \setminus \{(u, v), (w, z)\} \cup \{(u, z), (w, v)\}$;
return G';

Example

Interpretation

Interpretation (continue)

Analysis

What we want: the output of `RandomGraphFixedDegrees`(G, k) is a graph *chosen uniformly at random* among *all the graphs* with the same degree sequence as G.

Equivalent: *every* graph with the same degree sequence as G is produced in output by `RandomGraphFixedDegrees`(G, k) with the same probability.

Question 1: can `RandomGraphFixedDegrees`(G, k) produce in output every graph with the same degree sequence as G?

Proposition

Every graph with the same degree sequence as G can be obtained by a sequence of edge swap operation.

Proved in Ryser (1957), *Combinatorial properties of matrices of zeros and ones*, Canadian J. Math.

Question 1: can `RandomGraphFixedDegrees`(G, k) produce in output every graph with the same degree sequence as G?
Answer: yes.

Analysis (continue)

Question 2: does every graph with the same degree sequence as G have the same probability to be produced in output by `RandomGraphFixedDegrees`?

Answer: yes, if k is large enough

Intuition

A Different, but Not Correct, Algorithm

Algorithm `RandomGraphFixedDegrees`(G, k)
Input: graph $G = (V, E)$; $k \in \mathbb{N}^+$
Output: random graph G' where each vertex has the same degree as in G

$E' \leftarrow E$; $G' = (V, E')$;
for $i \leftarrow 1$ *to* k **do**
  sample edges (u, v) and (w, z) from E, with $(u, z) \notin E$ and $(w, v) \notin E$, uniformly at random;
  $E \leftarrow E \setminus \{(u, v), (w, z)\} \cup \{(u, z), (w, v)\}$;
return G';

Intuition

Analysis (continue)

Question 2: does every graph with the same degree sequence as G have the same probability to be produced in output by `RandomGraphFixedDegrees`?
Answer: yes, if k is large enough

Question 3: how large should k be?
Answer: empirically, $k \in \Omega(|E|)$ (e.g., $k = 100|E|$)

Related Random Graph: The Chung-Lu Model

Idea: instead of having the *exact* degree sequence, each node has (approximately) the same degree as in G *in expectation*

How?

Let $deg(u)$ be the degree of $u \in V$ in G, and $|E| = m$

For each pair (u, v), let $p_{u,v} = \frac{deg(u)deg(v)}{2m}$.

Random graph G': every edge (u, v) appears with probability $p_{u,v}$ independently of all other events.

Introduced by Chung and Lu (2002), *The average distances in random graphs with given expected degrees*, PNAS.

Analysis

Proposition

Let d_u the degree of u in random graph from the Chung-Lu model. Then

$$\mathbb{E}[d_u] = deg(u)\left(1 - \frac{deg(u)}{2m}\right)$$

Notes

- $\mathbb{E}[d_u] \approx deg(u)$
- some nodes u may have degree fairly different from $deg(u)$ (e.g., low degree nodes)
- the model is (fairly) amenable to analytical results
- it is the underlying model for a commonly used heuristic algorithm to find *communities* in networks