lorenzo.dallamico@unito.it

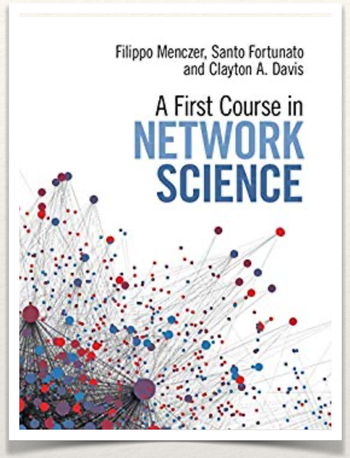
di.unito.it

Lecture 12.ns08

Networks models

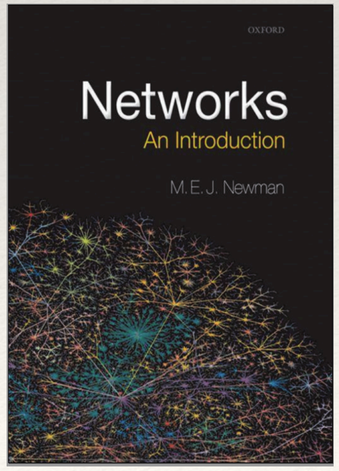
Course: **Complex Networks Analysis and Visualization**
Sub-Module: **NetSci**

References



Chapter 5: Network Models

Section 6.3.4: Stochastic block modeling



Chapter 12

Recap on structural characteristics of real networks

Features of real networks: small-world property

Most real-world networks are small worlds: **short paths**

Network	Nodes (N)	Links (L)	Average path length ($\langle \ell \rangle$)	Clustering coefficient (C)
Facebook Northwestern Univ.	10,567	488,337	2.7	0.24
IMDB movies and stars	563,443	921,160	12.1	0
IMDB co-stars	252,999	1,015,187	6.8	0.67
Twitter US politics	18,470	48,365	5.6	0.03
Enron Email	87,273	321,918	3.6	0.12
Wikipedia math	15,220	194,103	3.9	0.31
Internet routers	190,914	607,610	7.0	0.16
US air transportation	546	2,781	3.2	0.49
World air transportation	3,179	18,617	4.0	0.49
Yeast protein interactions	1,870	2,277	6.8	0.07
C. elegans brain	297	2,345	4.0	0.29
Everglades ecological food web	69	916	2.2	0.55

Features of real networks: high clustering coefficient

- The **clustering coefficient** of a node is the **fraction of pairs of the node's neighbors that are connected to each other:**

$$C(i) = \frac{\tau(i)}{k_i(k_i-1)/2} = \frac{2\tau(i)}{k_i(k_i-1)}$$

where $\tau(i)$ is the number of triangles involving i. Note that in this definition, the clustering coefficient is undefined if $k_i < 2$: a node must have at least degree 2 to have any triangles. However NetworkX assumes $C=0$ if $k=0$ or $k=1$

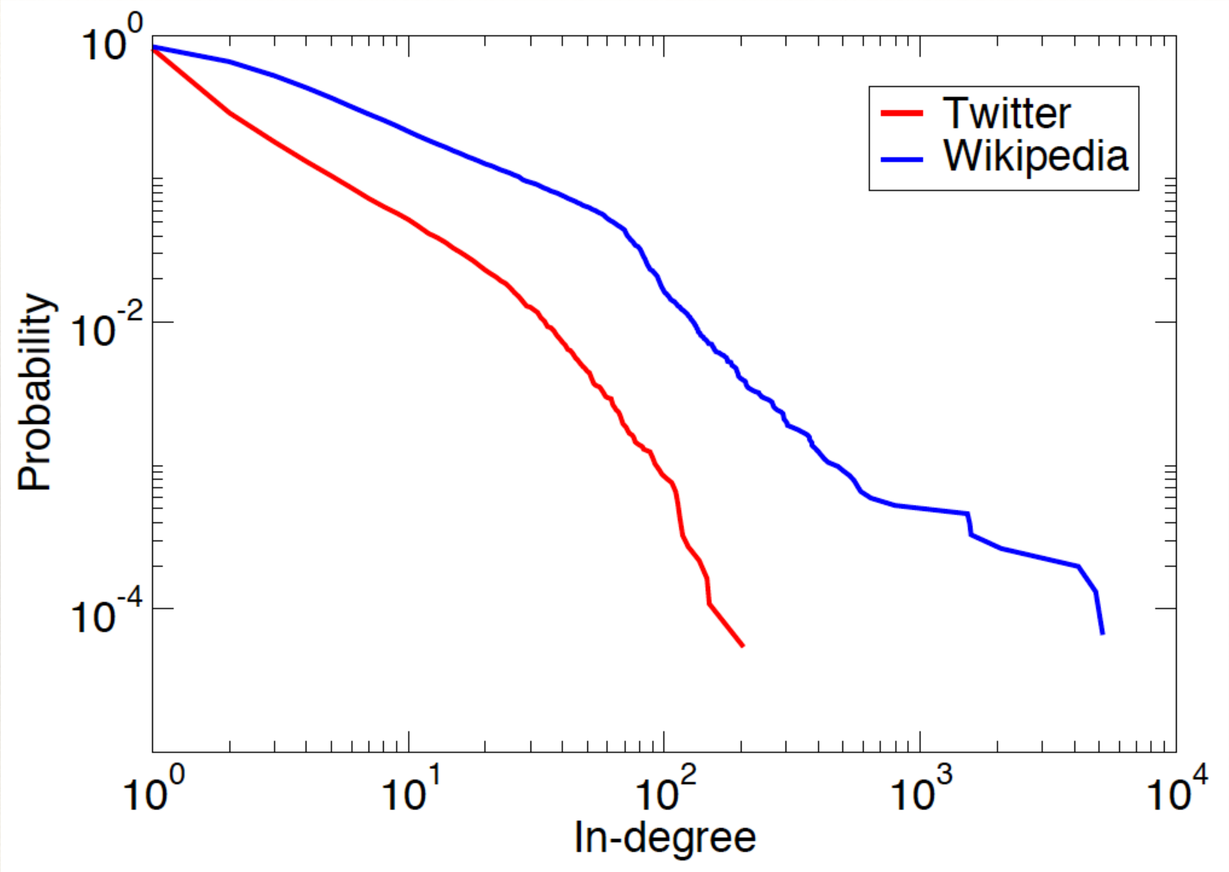
Features of real networks: high clustering coefficient

- Many networks have high clustering coefficients
- Other networks, e.g., bipartite and tree-like networks, have low clustering coefficient

Network	Nodes (N)	Links (L)	Average path length ($\langle \ell \rangle$)	Clustering coefficient (C)
Facebook Northwestern Univ.	10,567	488,337	2.7	0.24
IMDB movies and stars	563,443	921,160	12.1	0
IMDB co-stars	252,999	1,015,187	6.8	0.67
Twitter US politics	18,470	48,365	5.6	0.03
Enron Email	87,273	321,918	3.6	0.12
Wikipedia math	15,220	194,103	3.9	0.31
Internet routers	190,914	607,610	7.0	0.16
US air transportation	546	2,781	3.2	0.49
World air transportation	3,179	18,617	4.0	0.49
Yeast protein interactions	1,870	2,277	6.8	0.07
C. elegans brain	297	2,345	4.0	0.29
Everglades ecological food web	69	916	2.2	0.55

Features of real networks: heterogeneity

- **Heavy-tail distributions:** the variable goes from small to large values
- **Hubs:** nodes with high degree



Features of real networks: heterogeneity

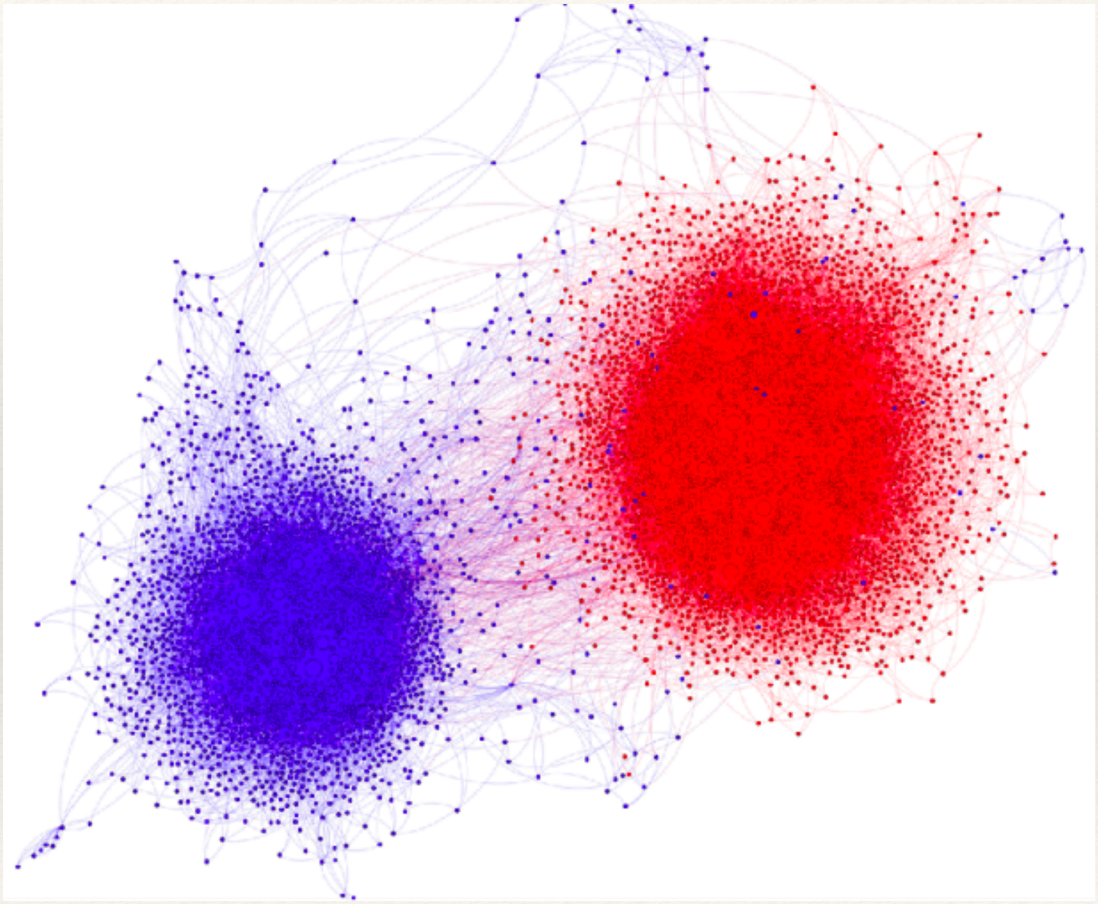
- **Heterogeneity parameter:** a measure of how broad the degree distribution is

$$\kappa = \frac{\langle k^2 \rangle}{\langle k \rangle^2}$$

Network	Nodes (N)	Links (L)	Average degree ($\langle k \rangle$)	Maximum degree (k_{max})	Heterogeneity parameter (κ)
Facebook Northwestern Univ.	10,567	488,337	92.4	2,105	1.8
IMDB movies and stars	563,443	921,160	3.3	800	5.4
IMDB co-stars	252,999	1,015,187	8.0	456	4.6
Twitter US politics	18,470	48,365	2.6	204	8.3
Enron Email	36,692	367,662	10.0	1,383	14.0
Wikipedia math	15,220	194,103	12.8	5,171	38.2
Internet routers	190,914	607,610	6.4	1,071	6.0
US air transportation	546	2,781	10.2	153	5.3
World air transportation	3,179	18,617	11.7	246	5.5
Yeast protein interactions	1,870	2,277	2.4	56	2.7
C. elegans brain	297	2,345	7.9	134	2.7
Everglades ecological food web	69	916	13.3	63	2.2

Community structure

- Tridic closure and weak ties
- Connections as proxy of affinity



Random graphs

- We need random graphs to be able to study networks analytically and get a deeper theoretical sense.
- We might also want to be able to generate a large number of graphs for our studies and real networks are limited
- Finally, we want to know the role played by a given graph property (e.g. the average degree) and we want to be able to tune it.

Erdos-Renyi random graph

Definition

- G(n,p): any two nodes are connected with probability p (Gilbert)
- G(n,L): there are L edges that are randomly placed (Erdos-Renyi)

- G(n,p) is equivalent to G(n, L) if

$$L = Bin\left(\frac{n(n-1)}{2}, p\right)$$

ER graph

- Simple idea: placing links at random between pairs of nodes
- **Algorithm** (Gilbert random network model):
  1. Start with N nodes and zero links
  2. Go over all pairs of nodes; for each pair of nodes i and j, generate a random number r between 0 and 1
     1. If $r < p \Rightarrow i$ and j get connected
     2. If $r > p \Rightarrow i$ and j remain disconnected

Percolation threshold

We compare the size of the giant component with the graph size.
Let G be the giant component: what is the probability that a node belongs to G?

$$\mathbb{P}(i \notin G) = \prod_{i \in N \setminus \{i\}} 1 - p + p\mathbb{P}(j \notin G)$$

Percolation threshold

We rename variables and exploit symmetry

$$p := \frac{d}{n-1}$$

$$\mathbb{P}(i \notin G) := u_i = u$$

We get

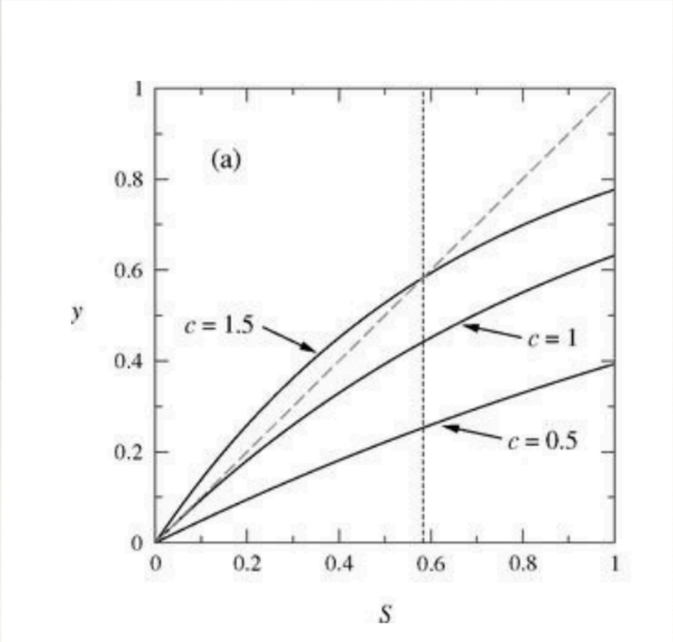
$$u = \left[1 - \frac{d(1-u)}{n-1}\right]^{n-1} \to e^{-d(1-u)}$$

Percolation threshold

The probability to be part of the giant component is s = 1-u

$$\mathbb{P}(i \in G) = 1 - u = s = 1 - e^{-ds}$$

Which can be solved numerically

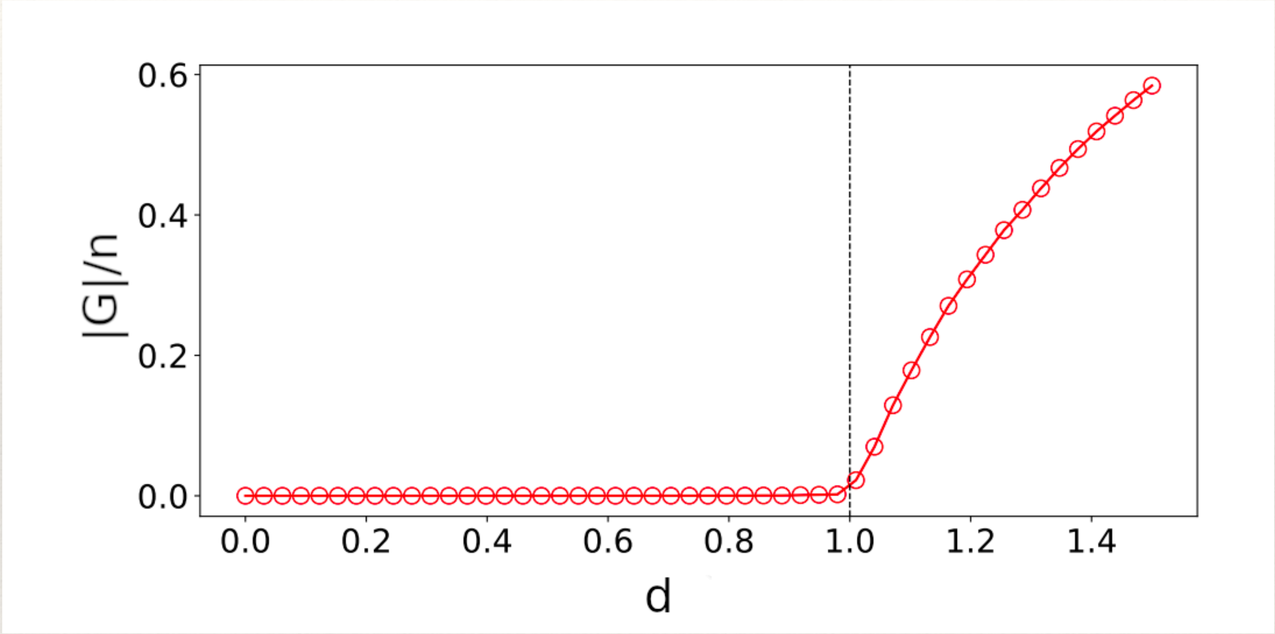


Percolation threshold

We compare the size of the giant component with the graph size.

There is a sharp transition for d = 1

$$\begin{cases} d > 1 \to |G| = O_n(n) \\ d < 1 \to |G| = o_n(1) \end{cases}$$

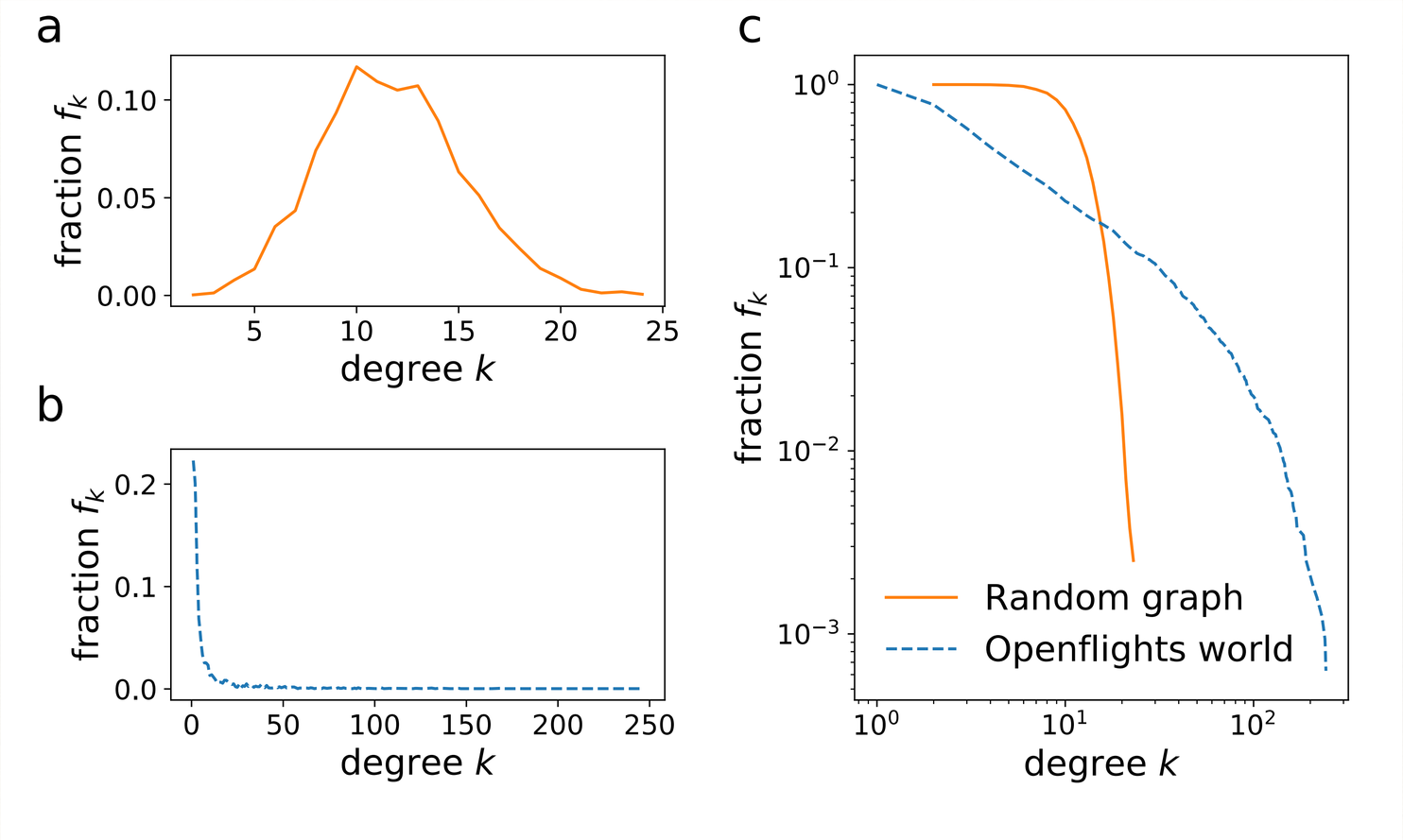


Degree distribution

- **Coin tossing problem:** what is the probability that a coin that yields heads with probability p results in k heads out of N–1 (independent) trials?
- **Binomial distribution:** $$P(k) = \binom{N-1}{k} p^k (1-p)^{N-1-k}$$

- For small p and large N the binomial distribution is well approximated by a bell-shaped curve ⇒ **most degree values are concentrated around the peak, so the average degree is a good descriptor of the distribution**
- $$\langle k \rangle = (n-1)p = d$$

Degree distribution



The degree distribution of random networks **is very different** from the broad distributions of most real-world networks!

Connectedness transition

$$s = 1 - e^{-ds}$$

$$\mathbb{P}(\min\ d_i > 0) \sim e^{-ne^{-d}}$$

If d is sufficiently large, s -> 1. In particular, if d > log(n) the graph is connected with high probability, while if d < log(n) it has isolated nodes with high probability.

Diameter

- **Question:** how many nodes are there (on average) d steps away from any node?
- **Premise:** since nodes have approximately the same degree, let us assume they have all exactly the same degree k
  - At distance $d = 1$ there are k nodes
  - At distance $d = 2$ there are $k(k-1)$ nodes
  - …
  - At distance d there are $k(k-1)^{d-1}$ nodes
- If k is not too small, the **total number of nodes within a distance *d*** from a given node is approximately:

$$N_d \sim k(k-1)^{d-1} \sim k^d$$

Diameter

- **Question:** how many steps does it take to cover the whole network?

$$N \sim k^{d_{max}}$$

$$\log N \sim d_{max} \log k$$

$$d_{max} \sim \frac{\log N}{\log k}$$

- The diameter of the network **grows like the logarithm** of the network size

- **Example:** $N = 7{,}000{,}000{,}000$, $k = 150$

$$d_{max} = 4.52$$

Clustering coefficient

- The clustering coefficient of a node *i* can be interpreted as the probability that two neighbors of *i* are connected

$$C_i = \frac{\text{number of pairs of connected neighbors of i}}{\text{number of pairs of neighbors of i}}$$

- **Question:** what is the probability that two neighbors of a node are connected?

- **Answer:** since links are placed independently of each other, it is just the probability *p* that any two nodes of the graph are connected:

$$C_i = p = \frac{\langle k \rangle}{N-1} \sim \frac{\langle k \rangle}{N}$$

- Since <*k*> is usually a small number, the average clustering coefficient of random networks with realistic values for <*k*> and *N* is **much smaller** than the ones observed in real-world networks

Erdos-Renyi: summary

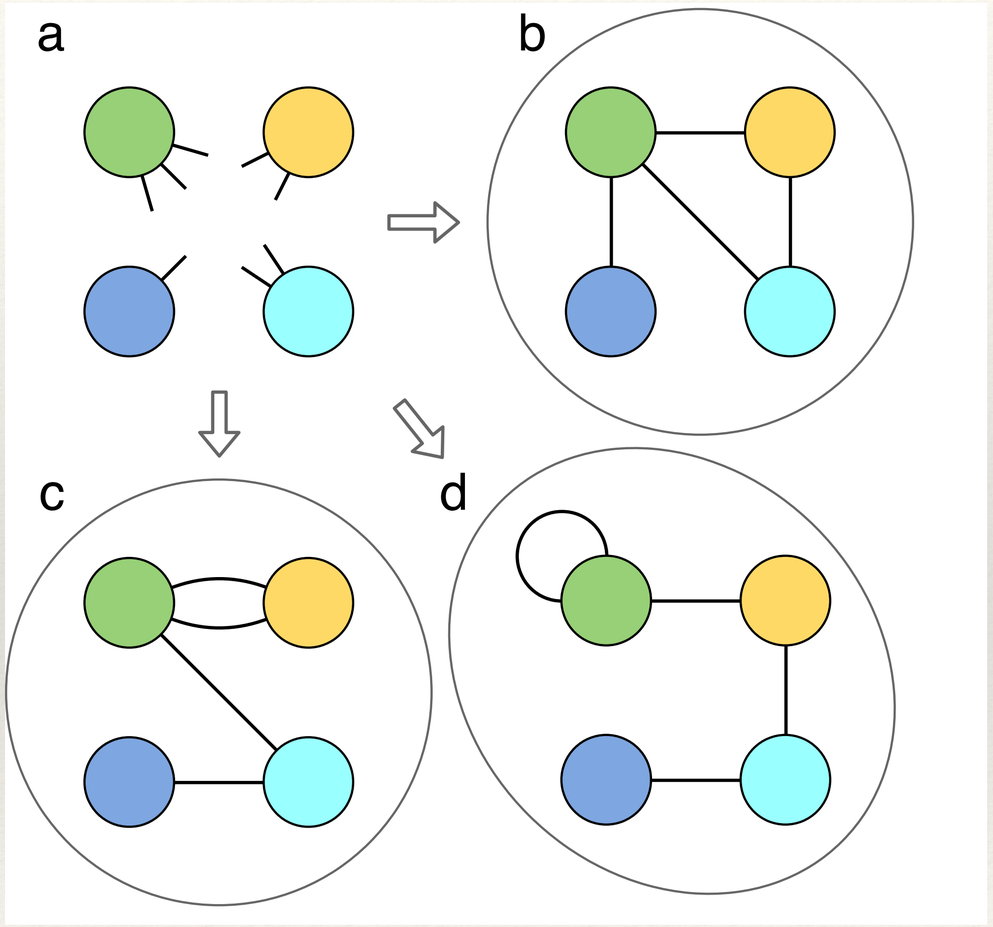
- Links are placed at random, independently of each other
- Distances between pairs of nodes are short (small-world property): **good!**
- The average clustering coefficient is much lower than on real networks of the same size and average degree: **bad!**
- The nodes have approximately the same degree, there are no hubs: **bad!**
- No community structure exists: **bad!**

```
G = nx.gnm_random_graph(N,L) # Erdos-Renyi random graph
G = nx.gnp_random_graph(N,p) # Gilbert random graph
```

The configuration model

The configuration model

- **Problem:** is it possible to build networks with a predefinite degree distribution?
- **Solution:** the configuration model
- **More specific focus:** build networks with a predefinite degree sequence!
- **Degree sequence:** list of N numbers (k_1, k_2, … , k_{N-1}, k_N), where k_i is the degree of node i
- **Principle:** assign a degree to each node (*e.g.*, from the desired distribution or a real network), place as many **stubs** on each node as the degree of the node, and attach pairs of stubs at random

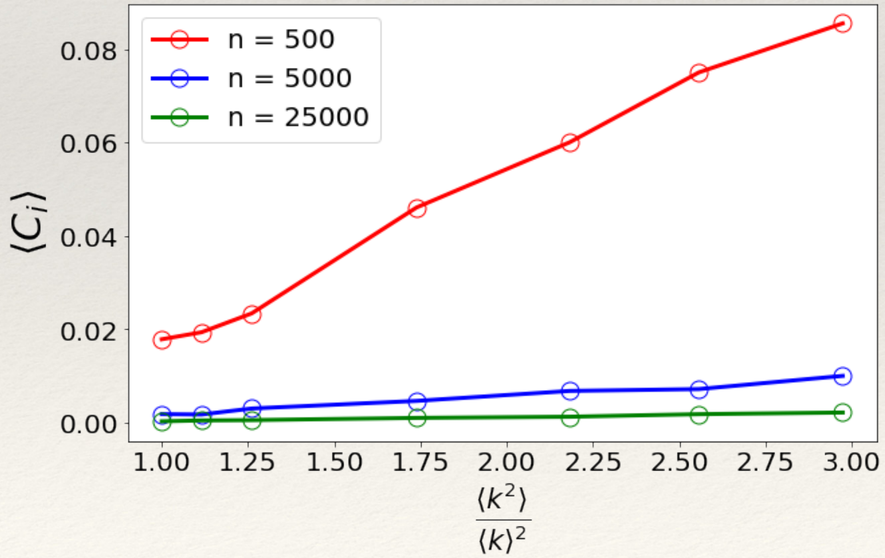


The configuration model

- **Degree-preserving randomization:** generate randomized versions of a given network with the same degree sequence, using the configuration model
- **Why:** useful to see whether a specific property of the original network is determined by its degree distribution alone
  - If the property is maintained in the randomized configurations, then the degree distribution is the main driver
  - If the property is lost in the randomized configurations, other factors must be responsible for it

The clustering coefficient

- Neighbors with high degree are more likely to get connected, so heterogeneity improves the clustering
- Randomness however makes triangle rare, especially for large networks



The random version of the configuration model

- Assign each node a probability p_i
- Generate edges independently (up to symmetry) with probability

$$\mathbb{P}(A_{ij} = 1) = p_i p_j$$

- The expected degree of i reads: **connectivity is not guaranteed**

$$\mathbb{E}[k_i] = \sum_{j \neq i} \mathbb{E}[A_{ij}] \approx p_i n \langle p \rangle$$

Configuration model: summary

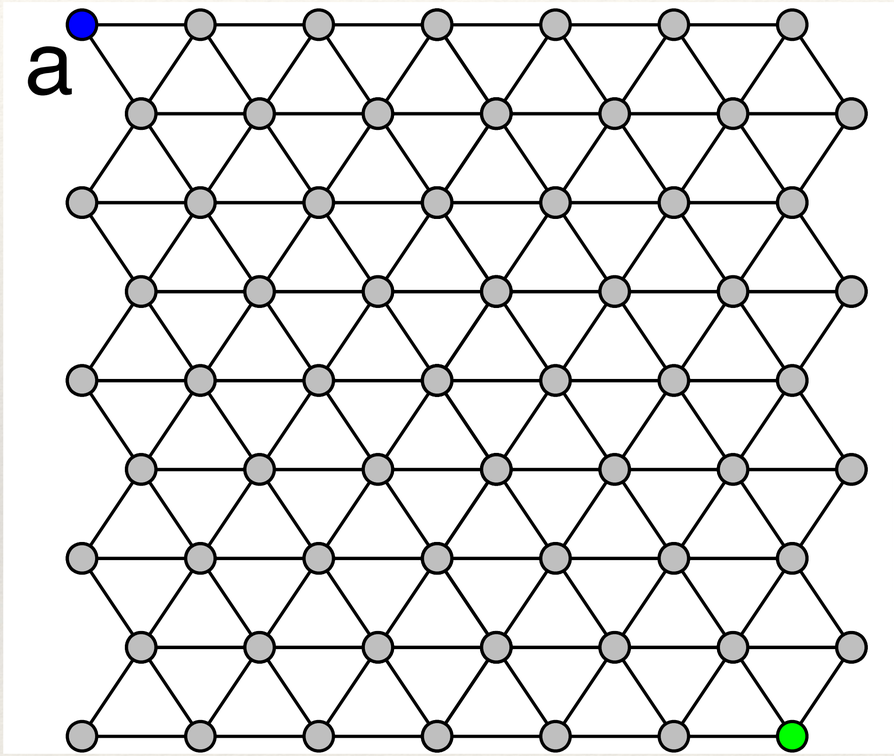
- Links are placed at random, respecting a degree distribution
- Distances between pairs of nodes are short (small-world property), inherited from the Erdos-Renyi: **good!**
- The average clustering coefficient is much lower than on real networks of the same size and average degree: **bad!**
- We may have any degree distribution: **good!**
- No community structure exists: **bad!**

```
# network with degree sequence D
G = nx.configuration_model(D)
```

Small-world networks

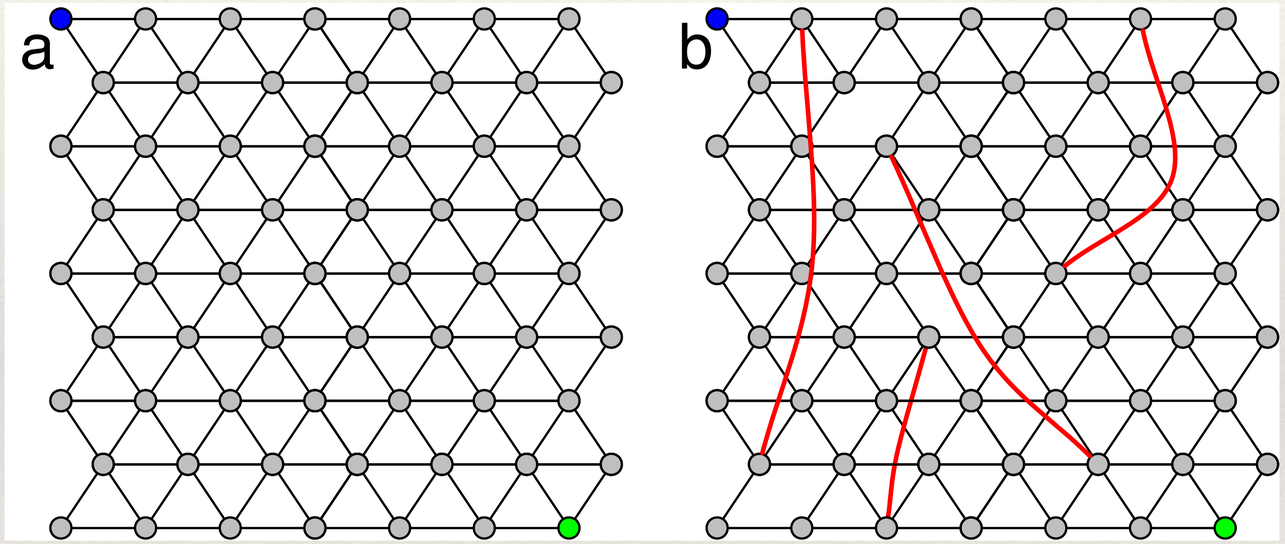
Small-world networks

- **Problem:** randomness gives small diameters and low clustering coefficients
- **Solution:** interpolating between a regular lattice (high clustering) and a random network (small-world property)
- **Clustering coefficient of lattice is high:**
  - The internal nodes have $k = 6$ neighbors, 6 pairs of which are connected
  - $C = 6/[(6*5)/2] = 6/15 = 2/5 = 0.4$
  - Most nodes are internal, so the average clustering coefficient of the network is close to 0.4!



Small-world networks

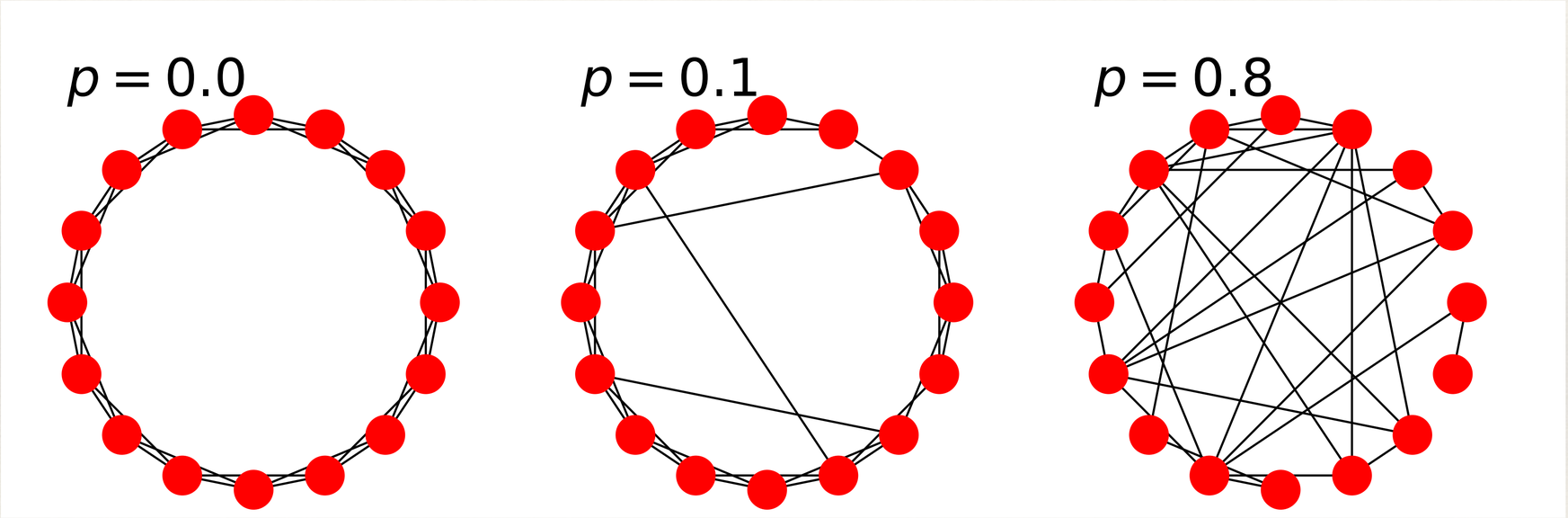
- **Large average shortest path length:** Going from a node to another can take a large number of steps, which grows rapidly with the size of the network/grid



- **Solution? Shortcuts!**

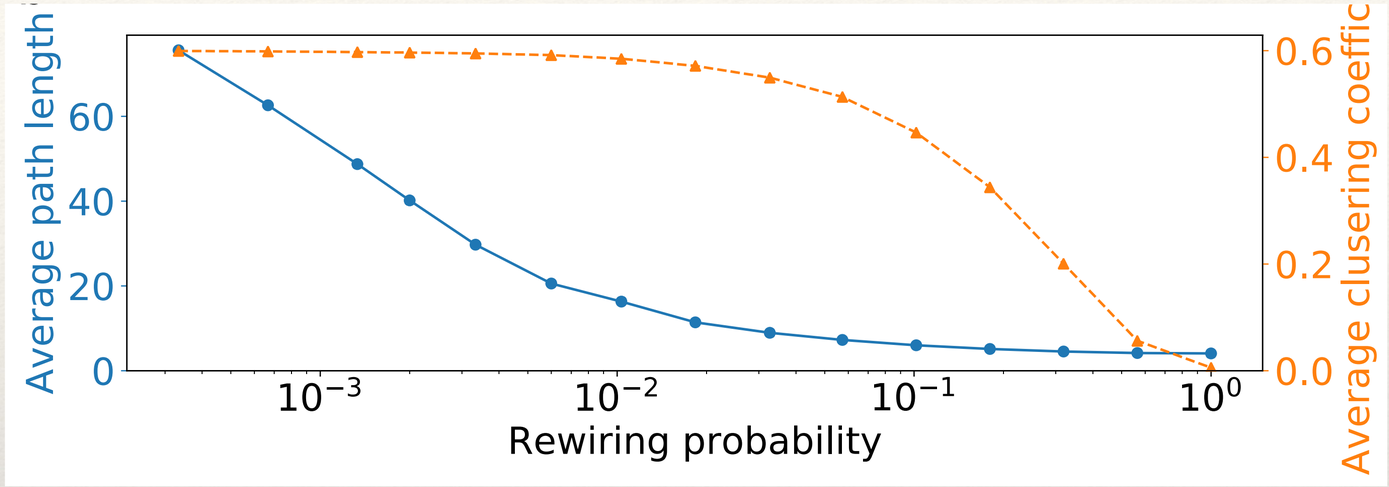
The Watts-Strogatz model

N nodes form a regular ring lattice, with even degree *k*. With probability *p*, each link is rewired randomly



The Watts-Strogatz model

- The expected number of rewired links is $pL = pNk/2$
  - If $p = 0$, no links are rewired: **no change**
  - If p is small, few links are rewired: **the average clustering coefficient stays approximately the same because very few triangles are destroyed, but distances shrink considerably**
  - If $p = 1$, all links are rewired: **the network becomes a random network**



Distances become short already for low values of p; the average clustering coefficient stays high up to large values of p. **There is a range of values of p where the average path length is short and the clustering coefficient is high!**

Degree distribution

What is the degree after the reshuffling?

$$\bar{k} = k - k_{\text{out}} + k_{\text{in}}$$

$$k_{\text{out}} \sim Bin\left(\frac{k}{2}, p\right)$$

$$k_{\text{in}} \sim Bin\left(\frac{(n-1)k}{2}, \frac{p}{n-1}\right)$$

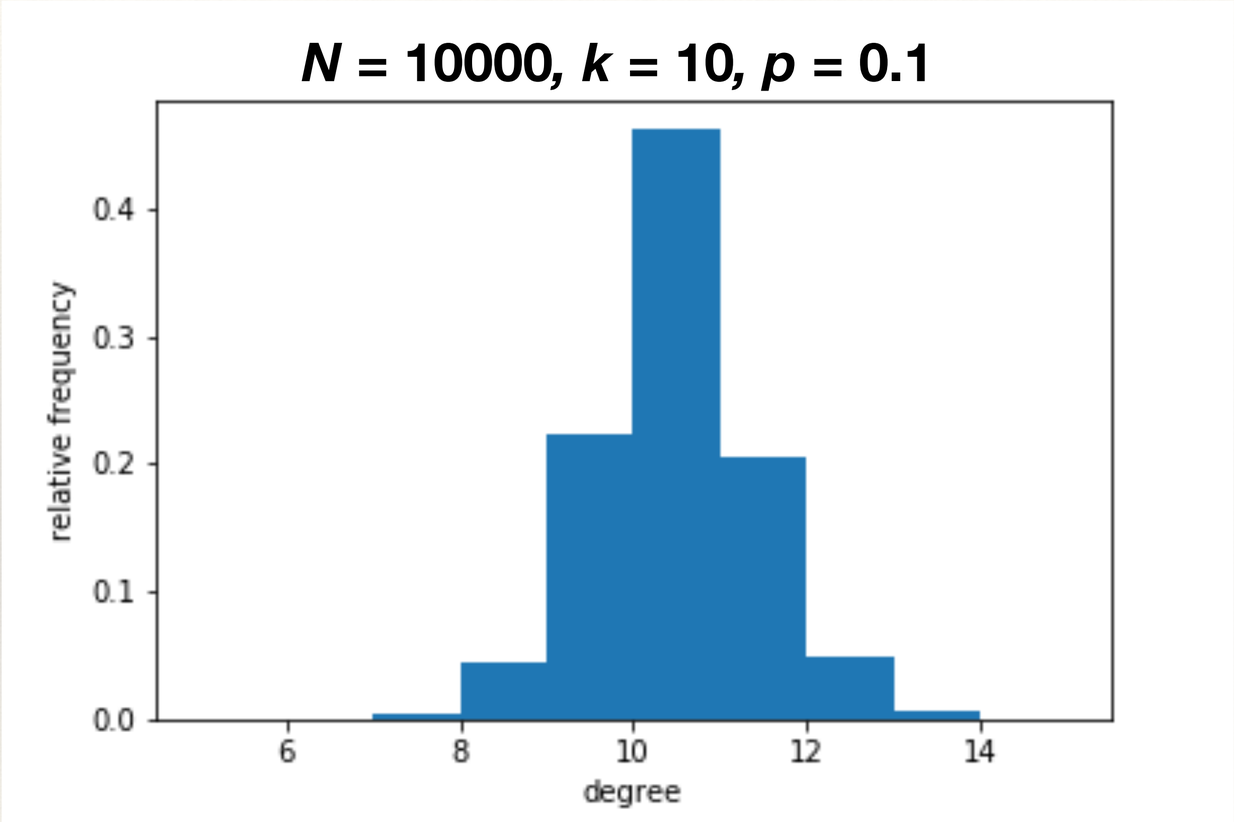
$$\mathbb{E}[\bar{k}] = k$$

$$\mathbb{V}[\bar{k}] \approx \frac{kp(2-p)}{2}$$

So we obtain mean and variance of the degree distribution

Degree distribution

- The degree distribution is peaked as most nodes have the same degree: **no hubs!**
- The Watts-Strogatz model fails to reproduce the broad degree distributions observed in many real-world networks



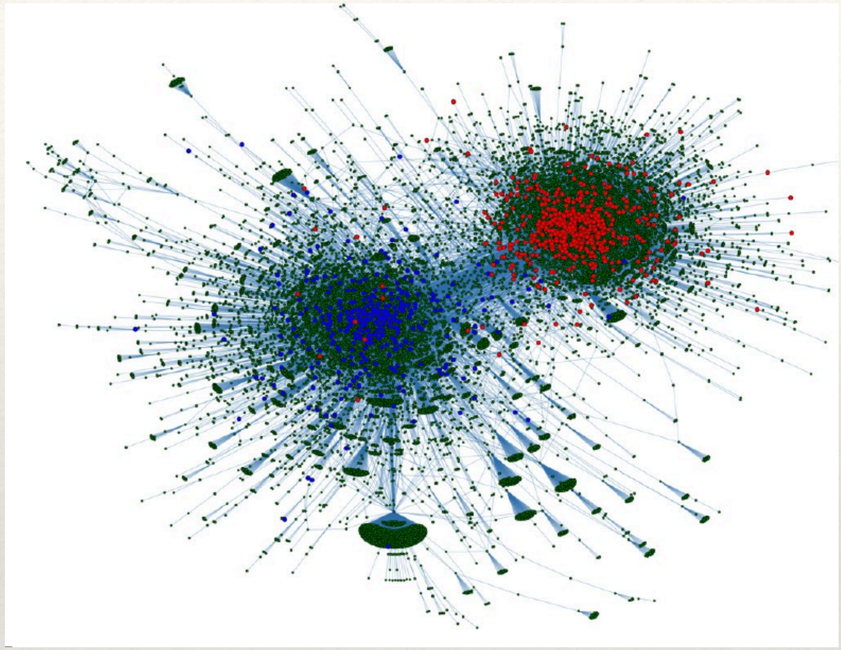
The Watts-Strogatz model: summary

- A regular lattice whose links are randomly rewired, with some probability p
- There is a range of values of the rewiring probability p for which distances between pairs of nodes are short (small-world property) and the average clustering coefficient is high: **good!**
- The nodes have approximately the same degree, there are no hubs: **bad!**
- No community structure: **bad!**

```
# small-world model network
G = nx.watts_strogatz_graph(N,k,p)
```

Random networks with communities

Random networks with communities



Many networks have a community structure: there are groups of nodes that are more tightly connected among themselves than with the others.

How do we generate a network with communities?

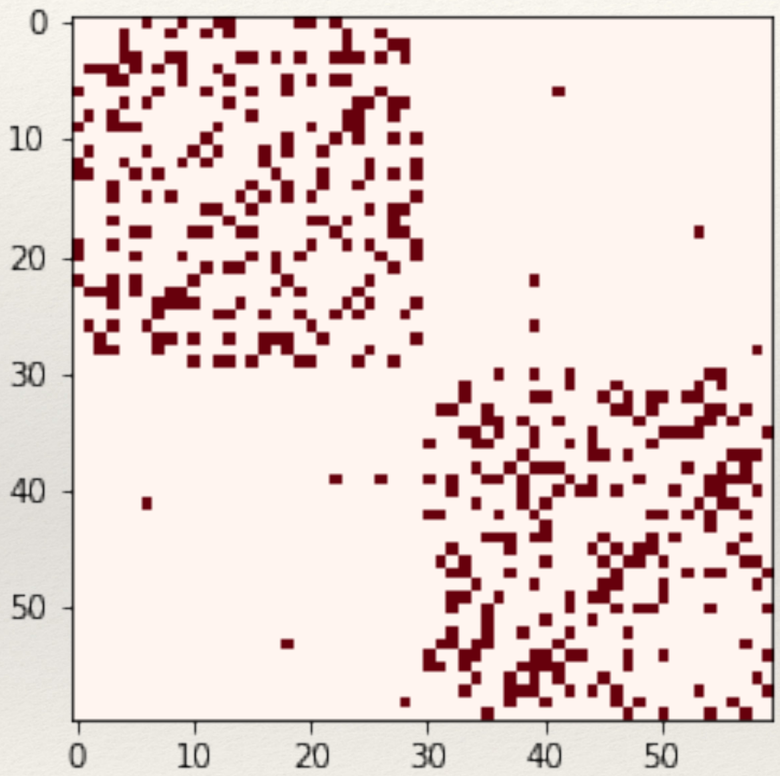
Stochastic block model

Suppose there are two communities. Each node has a label (l = 1 or l = 2). Then, each edge is generated independently at random with the following probability

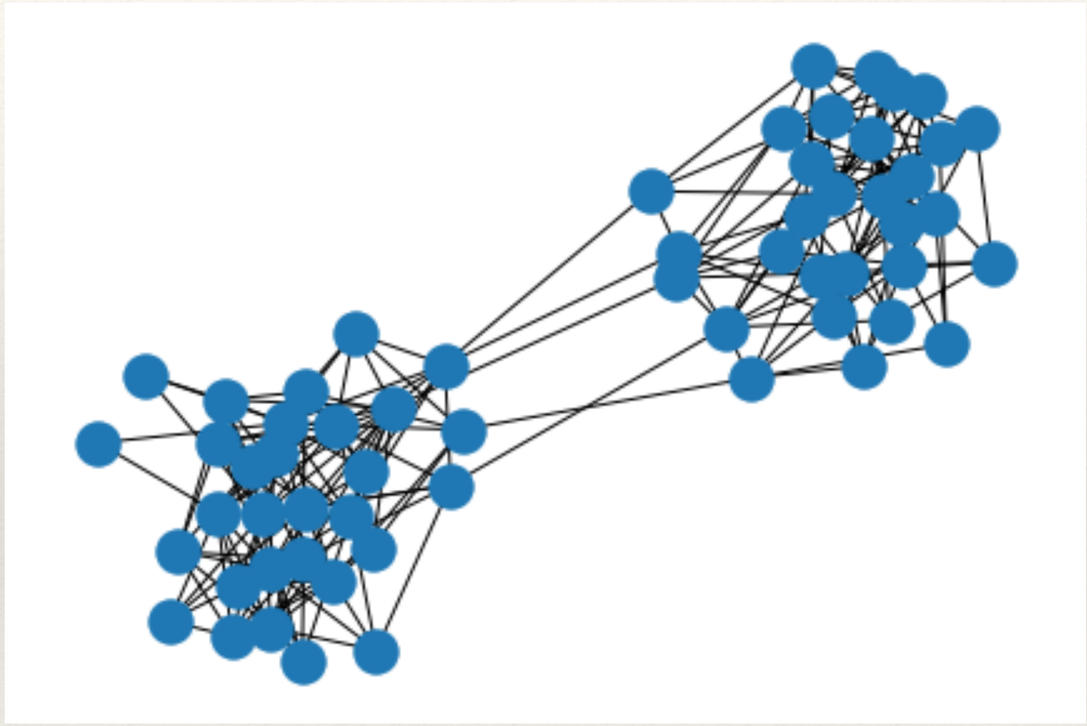
$$\mathbb{P}(A_{ij} = 1) = \begin{cases} p_{\text{in}} & \text{if } l_i = l_j \\ p_{\text{out}} & \text{if } l_i \neq l_j \end{cases}$$

Settin p_in > p_out we obtain the required community structure. This can be extended to multiple classes, but we will talk about this later on.

Stochastic block model



Adjacency matrix



Graph

Stochastic block model: summary

The stochastic block model inherits most of its properties from the Erdos-Renyi

- Links are placed at random, independently of each other but with a community structure.
- Distances between pairs of nodes are short (small-world property): **good!**
- The average clustering coefficient is much lower than on real networks of the same size and average degree: **bad!**
- The nodes have approximately the same degree, there are no hubs: **bad! But**, we can create a mix with the configuration model and obtain the **degree-corrected stochastic block model (good!)**
- Creates a community structure: **good!**

```
G = nx.stochastic_block_model(sizes, probs)
```

Geometric model

Geometric model (optional)

https://arxiv.org/pdf/cond-mat/0203026.pdf

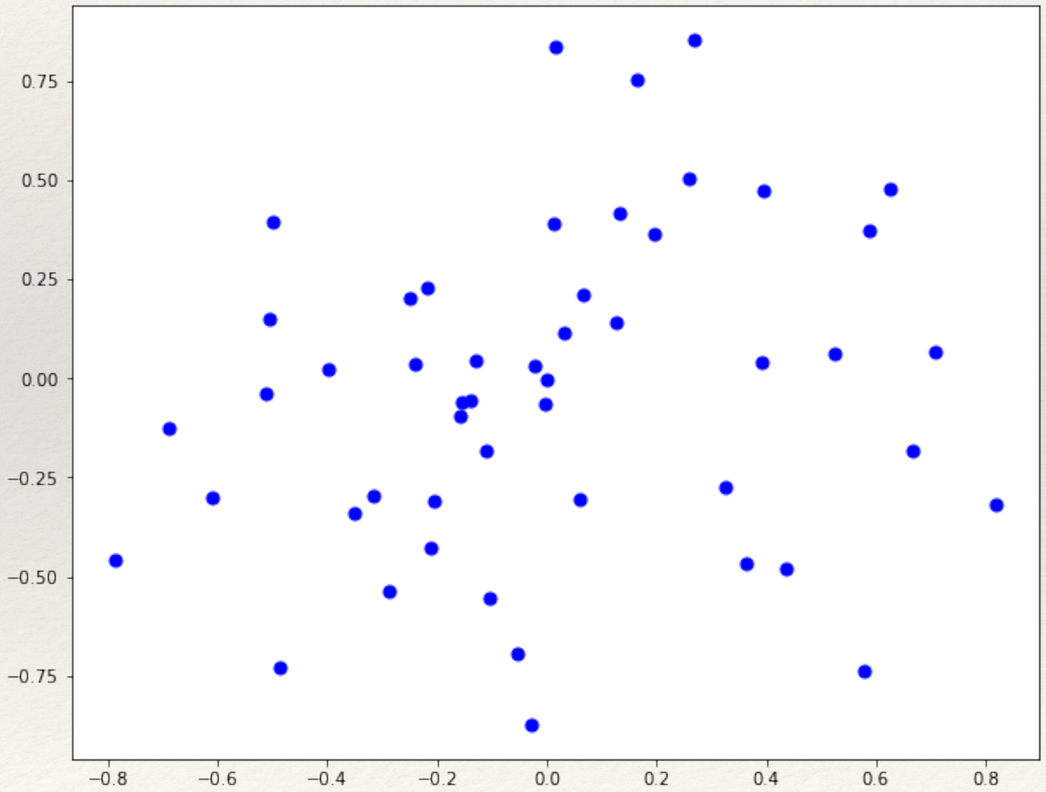
Random models are good to capture the small world effect, but they tend to have **low clustering coefficients**. The problem is due to the fact that edges are generated independently.

Observation: triadic closure is due to opportunity

Idea: introduce a concept of geometrical proximity between nodes

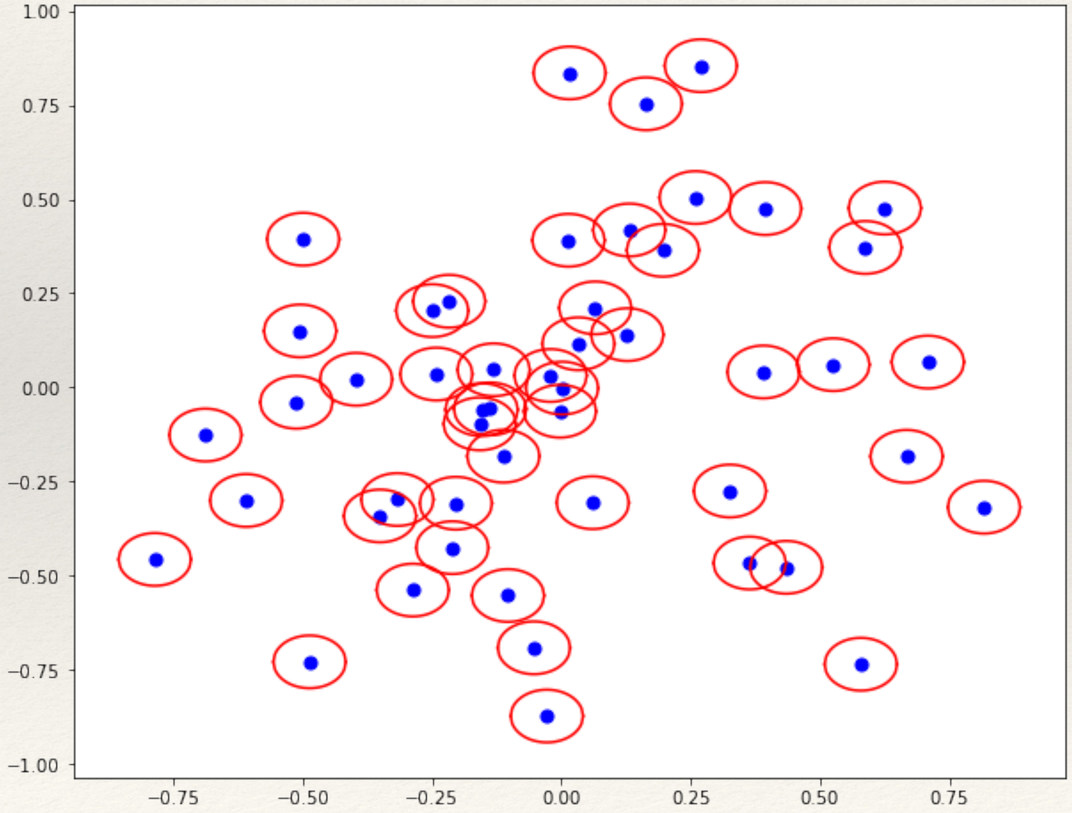
Geometric model (optional)

Example: map each node to a point in a two dimensional space



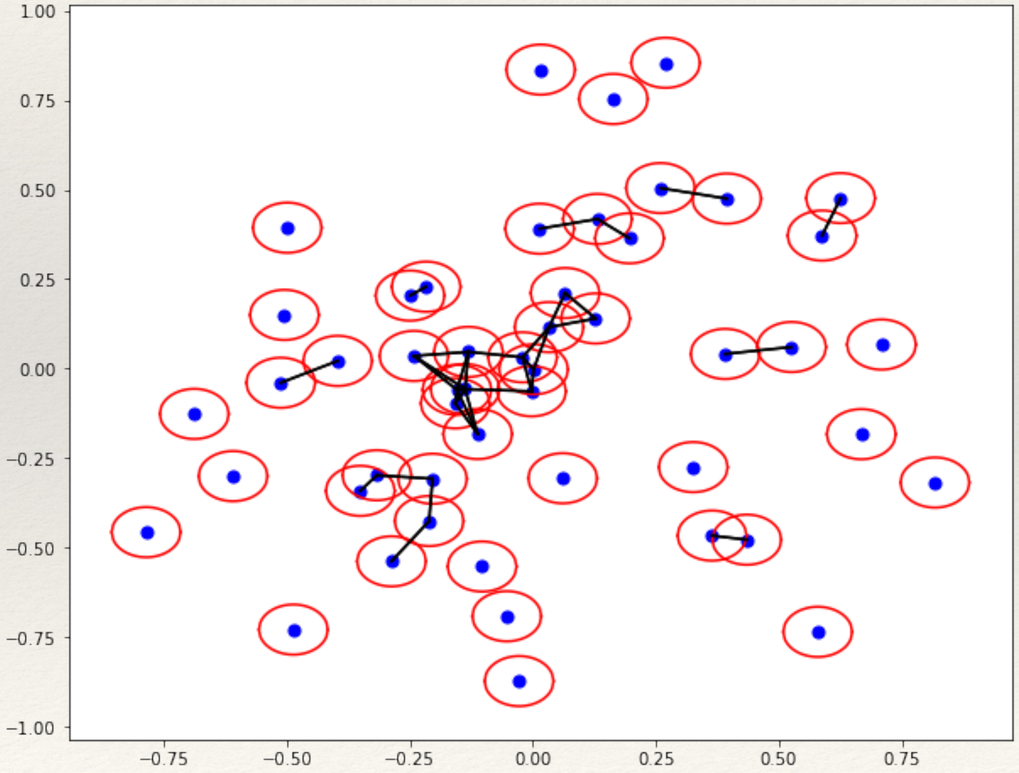
Geometric model (optional)

Example: now draw a circle of radius R around this point



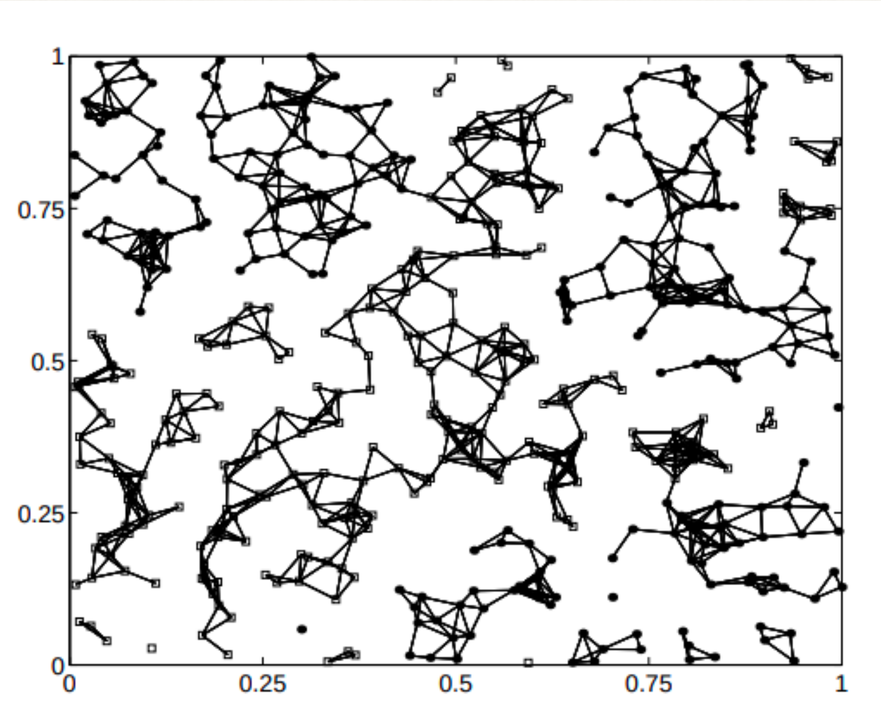
Geometric model (optional)

Example: if two circles overlap then join the two vertices with an edge



Geometric model (optional)

- What we have obtained is a geometrical model in which connections exist based on proximity.



Geometric model (optional)

- The a**verage degree** depends on the value of R which also determines the **percolation threshold** of this model
- This model is capable of reproducing a network with a **high clustering coefficient** hence with many small communities
- This can be generalized to many settings: the input **space dimensions**, its **domain**, the initial distribution of points, the **distance function**, for instance.
- The model was presented as a deterministic one (once the the initial points are set). Softer versions exist, in which the connection probability is a decaying function of the nodes distance.

Geometric model: summary

- Generates the edges according to the distance of the vertices in an embedded space
- Generates networks with a high clustering coefficient. There are several parameters than can be tuned and that we will not explore. Try to make some tests and see what happens!

```
G = random_geometric_graph(n, radius)
```