1 Introduction to Networks

1.1 What are networks?

Intuitively the networks that we encounter share the features that individual elements of the whole system are coupled by links that form a complicated system of interactions that determine the entire structure or the dynamics that the structure produces. However two basic questions are not addressed:

1. What systems can be modelled by networks?
2. What are interesting networks?

If we think about it, almost all systems in nature can be considered as parts that interact by reciprocal forces, so why not model everything as a network? An example from astrophysics can illustrate why the network approach may not necessarily be the right approach for certain systems and should be reserved for those for which it is suitable. Consider a system as illustrated in Fig. 1.1. The figure shows a globular cluster of a couple of thousand stars. All these stars can be considered elements that interact by gravitational force. If we label the stars i then the interaction force between pairs at a given point in time is given the force of gravity

$$F_{ij} = G\frac{M_i M_j}{|\mathbf{x}_i - \mathbf{x}_j|^2} \tag{1.1}$$

where $G = 6.67 \times 10^{-11}\mathrm{m}^3/\mathrm{kg}\,\mathrm{s}^2$ is the graviational constant, M_i and M_j are the masses of stars i and j, and $\mathbf{x}_i$ and $\mathbf{x}_j$ are their positions. So why not consider this a network of nodes i with connections of strength F_{ij}? Why not investigate a system such as this one with network theory? In physics this system is called a many body problem and it seems strange that these systems aren't the prime example for systems suitable for network science. The reason for this can be understood by looking at the network systems from another perspective: Networks are systems in which certain connections **are missing**. In fact, most of the most interesting things about networks emerge because from a set of potential connections some are missing or some are much weaker in effect than others and can be neglected. The important ingredient in networks is that only a subset of potential links are realized.

Let's assume we have a system of N interacting units and let's assume we can measure their interaction strength A_{ij}. Now we make a histogram of these interaction strengths. Let's assume we find a scenario as depicted in Fig.1.2a : The A_{ij} are distributed over some interval. It's difficult to distinguish between interaction **types** and draw a line that could help us distinguish

Figure 1.1: A globular cluster of stars is a typical example of a many body system, not a network system.

between them and potentially neglect a subset of interactions. This scenario is typically seen in the globular cluster system or other many body systems. There are other systems though that might exhibit distributions of interactions that look more like those depicted in Fig. 1.2b and c. In b, the distributions of interactions has two peaks, one centered around zero and the other at some typical value. In a system of this type the structure of interactions would suggest that we could model the system by either setting interactions to "existing" or "not existing". If we were to draw a picture of the system we would draw a network. A similar situation is encountered in Fig. 1.2c. The difference to *b* is that the interactions not centered at zero have a broad distribution so that we may ignore the interactions around zero. However, the range of values for those that are in the right half of the distribution still must be considered. In this case we would model this by a weighted network.

Many systems encountered in network science never really need to be addressed this way because in many systems there is either a link between nodes or there is not, computers are either connected or they are not, facebook users are either friends or they are not, phonecalls are exchanged or they are not. Yet, it's important to keep in mind what certain networks mean, in particular in relation to social networks and operational definitions of networks.

In quantitative science, quantities must be defined operationally: they are defined by the experiment that quantifies them. Let's assume we would like to measure a social tie between dolphins (this is actually being done in research) by measuring how close they approach each

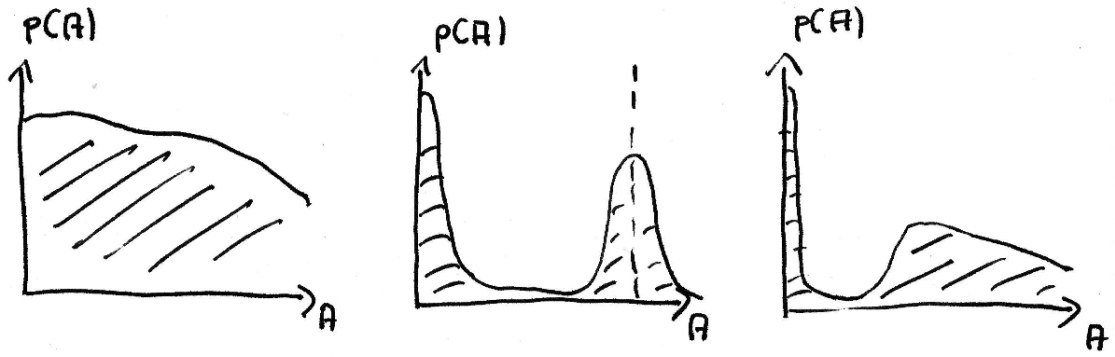


Figure 1.2: Distributions of interactions in many body systems. a) interactions range over a wide range b) basically two different interaction strengths exist, one fraction centeredaround $A \approx 0$ the other fraction centered around a typical value (dashed line) and c) a peak around $A \approx 0$ remains, followed by a gap and the nonzero fraction is distributed broadly.

other. We define a distance d and if a pair of dophins approach each other below that distance for a minimum of number N of times each day we say they have a tie. This generates a network of social ties in a swarm of dolphins.

This network definition depends on the parameters d and N. The network approach is only viable in this system if whatever conclusions we draw are robust agains small changes in these parameters. This is to be expected if for the interactions we measure something as illustrated in Fig. 1.2b and c, but it is doubtfull that this would be the case if interaction strengths were distributed according to Fig. 1.2a. Bottom line:

- Networks have nodes and links
- Interesting networks are those in which potential links are missing
- Always think about how a network is defined and whether the network approach is the most suitable one for your system.

1.2 Notation and graph theoretic origin

Our view of networks is that we've got nodes that are connected by links. Mathematicians call networks graphs because one can draw them. In graphing theory a graph $\mathcal{G}$ is a collection of nodes V and links E, i.e.

$$\mathcal{G} = \mathcal{G}(V, E).$$

The reason why the set of nodes is called V is because nodes are called vertices by mathematicians and links are called edges. We will not use this notation, since our focus is on real networks.

Also the term edge is confusing. Here's a table of typical expressions often used in place of nodes and links:

label	label	field
node	link	network science
vertex	edge	mathematics
actor	tie	social science
site	bond	physics

We will use the terms nodes and links throughout the class.

1.2.1 How to draw networks

A key aspect of networks is that you can gain insight about their structure by drawing them onto a piece of paper but how to draw them doesn't matter. All that counts is their connectivity structure. There's one exception: spatial networks which we will discuss in detail later. In spatial networks each node as a location in a space, most frequently in a two-dimensional space.

1.2.2 The orgin of graph theory - The Königsberg problem

The best way to see the value in abstracting in this way is by looking at the Königsbergproblem. Königsberg is a city in Prussia, formerly in Germany and now part of Russia. A river runs through it and 7 bridges cross to an island in the middle as illustrated in Fig. 1.3. A problem posed a number of centuries ago was:

- Is it possible to cross all the bridges in any arbitrary sequence with crossing each bridge once.

Euler solved this problem by using network (or rather graph theory), realizing that only the topological situation counts. We drew a picture as shown in Fig. 1.3b. Each node represents a piece of land and the links represent bridges. Euler realized that if you cross each bridge exactly once in a given path there are basically two possibilities:

1. The path starts on one node i and ends at the same node i in which case the number of bridges connecting to that node must be even. Likewise for all other nodes because for every path that goes to a node one must be leaving.

2. If we have a path that starts at node i and ends at node j those two nodes can have an odd number of connections, but all the other nodes must have an even number of connections.

If we look at the graph closely, we see that the number of connections to each node is odd and therefore not path exists in which each bridge is crossed exactly once.

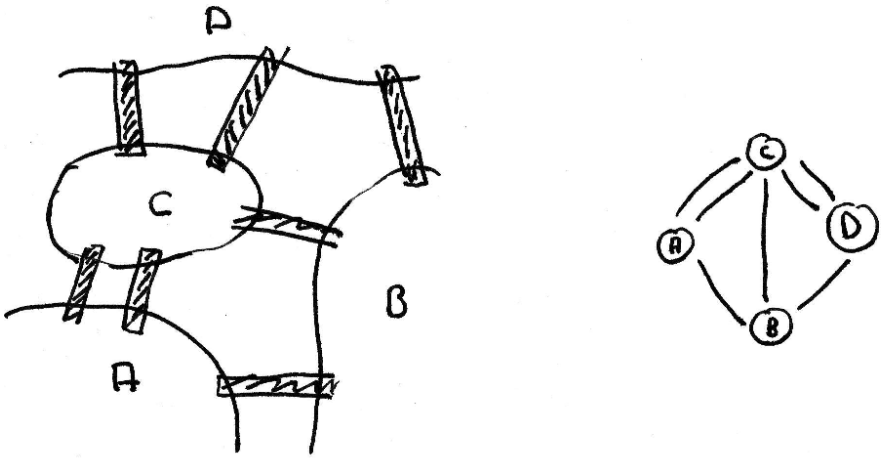


Figure 1.3: The Königsberg seven bridges problem.

1.2.3 Network flavors

Generally networks come in different flavors that share the common feature of having a set of nodes and a set of links. The simplest networks are...

... homogeneous undirected networks

In these networks links have no direction and no weight attached to them. A pair of nodes either is connected by a link or it isn't. Most of the networks that we will be discussing in this course are this type. If we have an undirected network with N nodes without any self-loops (links that connect a node to itself), the maximum number of links is

$$L_u = \frac{N(N-1)}{2}.$$

These networks are used in situations where it is meaningless to assign a direction to links. If for instance the nodes are people and a link denotes if two people have exchanged a hand-shake. These networks are also called symmetric because of the symmetry in links.

Directed networks

These are a bit more complicated. The links have a direction, so if there's a link $i \to j$ that does not imply that there's a link $j \to i$. i might be connected to j but not vice versa. The maximum number of links in such a network is

$$L_d = N(N-1).$$

Directed network are generally encountered if the direction of a link contains important information. For instance, if a link informs if person i has given person j a call. We may expect an

assymmetric situation. Foodwebs are another example of a network that is typically understood to be connected.

Symmetric directed networks

Sometimes we are dealing with situations where we need to model a real network as a directed network, which means there are connections going in both directions for a given node, but if a link goes from i to j it implies a link also goes from j to i. This sounds a little like homogeneous undirected networks. But it's actually slightly different as we will see soon.

Weighted networks

Weighted networks are very useful when the strength of connections plays a key role. in this case we assign a number w_{ij} to a node j to node i. Of course we can have directed or undirected weighted networks. Weighted networks often play a role in traffic networks where the weight quantifies the amount of traffic between nodes, but also in neural networks in which the weight can quantifiy the synaptic strength between neuron. In foodwebs it can quantify the amount of carbon exchanged between two species.

Hypergraphs and bipartite networks

Hypergraphs are a little different than the above. In hypergraphs more than 2 nodes can be connected or linked. Hypergraphs can be mapped on something that looks like a regular network, called a bi-partite network: we make an array of the original nodes i of the hypergraph and an additional row of new nodes that symbolize the links k. Both types of nodes, the original nodes and their linkages, can be connected in a bi-partite graph. This is illustrated in Fig. 1.4.

1.2.4 Trees and acyclic networks

A general feature of networks is that they may contain loops. A loop is a sequence of links that originate at some node i and terminate at the same node. Loops are also called cycles. The simplest loop is a self-loop, connecting a node to itself. If we have a node without self-loops we may nevertheless have loops. For instance triangles $i \to j \to k \to i$ if links exist between each pair in this sequence of nodes. If an undirected network has no loops, the network is called a

Tree

Trees are very interesting and useful. Often they are used as approximations to more complex networks. If all the nodes are part of a tree, the entire network is a tree. If there are multiple trees the network is called a forrest. An interesting feature of trees is that they've got

$$L_T = (N-1)$$

links, if N is the number of nodes. This is best understood if we draw a tree in a convenient layered way, by picking a leaf node (Nodes that only have one connection in a tree are called leaves) at the top, it's children below, the grand children below that, etc until every node is drawn. Each layer contains as many links connecting to the layer above as nodes drawn below, except for the root node which doesn't have a parent.

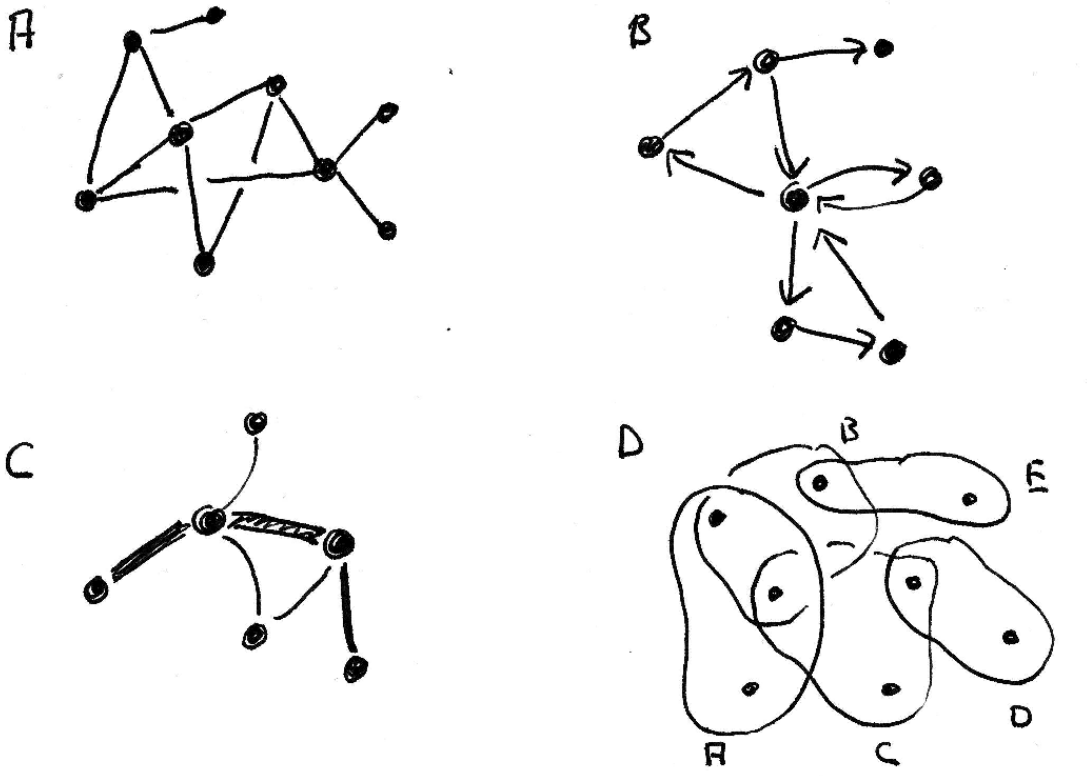


Figure 1.4: Types of networks. A: undirected network, B: Directed Network, C: Weighted (undirected) Network, D: Hypergraph.

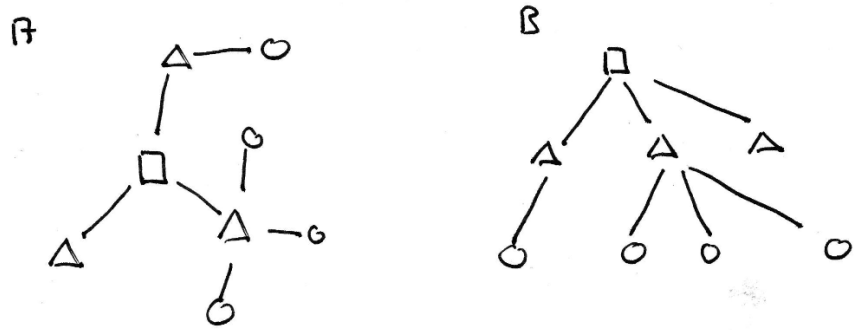


Figure 1.5: A tree is an undirected network without loops. A: an arbitrary way of drawing the tree. B: a multitier way of drawing the same tree, rooted at a chosen node. This way we see that a tree has exactly $(N-1)$ links.

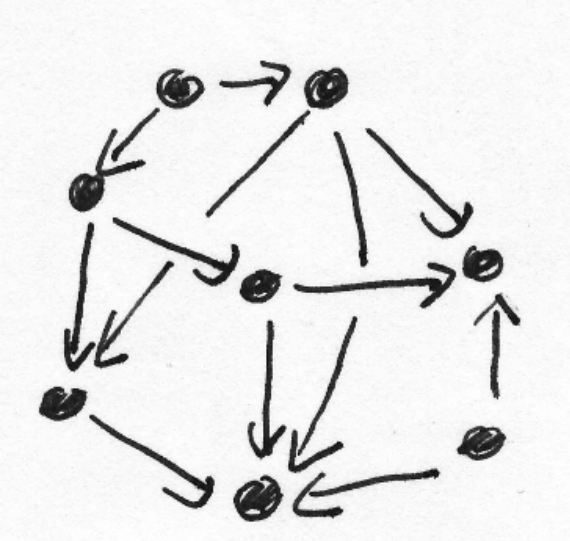


Figure 1.6: Acylcic directed network. This network contains no loops.

Acyclic network

In directed networks the situation is more complicated. It's more difficult to see that a network does not contain cycles. If it doesn't it's called an acyclic network (it's not necessarily a tree). Here's the way you can determine whether a network is acyclic.

1. find a node that contains only outgoing links
2. remove the node and all it's connecting nodes
3. go back to step 1.) if you can continue to remove all nodes this way, the network is acyclic.

1.3 Network representations

It's clear that one way of representing networks is by drawing them. But how can we represent them algebraically? Let's discuss simple undirected networks first.

1.3.1 The adjacency matrix

The most common way of representing a network of N nodes by an $N \times N$ matrix, known as the adjacency matrix A whose elements of which are defined as

$$A_{ij} = \begin{cases} 1 & \text{if nodes } i \text{and } j \text{are connected} \\ 0 & \text{otherwise} \end{cases}$$

This matrix is very simple, having only a bunch of zeros and ones. Note that according to the definition we always have $A_{ij} = A_{ji}$ i.e. the ajacency matrix contains roughly twice as much information as is required for an undirected network. Each link in an undirected network generates two 1 entries in the matrix (except the self loops).

1.3.1.1 Directed networks

Here we define the adjacency matrix as

$$A_{ij} = \begin{cases} 1 & \text{if node } j \text{connects to } i \\ 0 & \text{otherwise} \end{cases}$$

Here we see a difference between symmetric directed and undirected networks. In symmetric directed we have $A_{ij} = A_{ji}$ but each of these counts as one link.

1.3.1.2 Weighted Networks

Weighted networks contain more information on the links, namely the weights w_{ij}. So we can just identify a weighted adjenceny matrix according to

$$W_{ij} = \begin{cases} w_{ij} & \text{if node } j \text{connects to } i \\ 0 & \text{otherwise} \end{cases}$$

Since it doesn't matter which way we label nodes in a network and which way we draw them, it also means that a given network has many adjacency matrices that represent it. Given a matrix A we can form another matrix A' by swapping rows and column of A that describes the network as well. For instance the matrices

$$\begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

describe the same directed network depicted in Fig. 1.7

1.3.2 Link tables

The adjacency matrix is not the only way to represent the matrix. Just looking at one, we see that it may be the most intuitive way to represent a network but certainly not the most efficient and the most useful for certain tasks. Imaging you would like to represent the network of facebook this way. Facebook has about 5×10^8 nodes, so the adjancency matrix of connections in facebook would have about 25×10^{16} elements most of which are 0. Let's assume we were to put this into a computer memory, and we need one bit for each matrix element, we'd need 25 Petabyte to

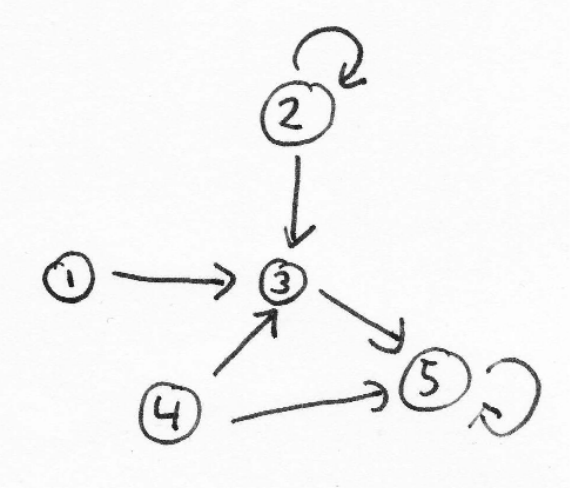


Figure 1.7: Example network.

store the information. But, we'd store many zeros. Alernatively we could represent the same information in a link table, that just codes up the links in pairs (i, j). So for instance a table

$$L = \begin{pmatrix} 1 & 3 \\ 2 & 2 \\ 2 & 3 \\ 3 & 5 \\ 4 & 3 \\ 4 & 5 \\ 5 & 5 \end{pmatrix}$$

encodes the same information as the adjacency matrix given above, it says a link exists between nodes 1 and 3, 2 and 2, etc. Instead of a 5×5 matrix with 25 elements we have a 2×7 matrix. Clearly this is a good way of doing things if the number of links a node has is typically much smaller than the number of possible links. If for instance we take the facebook example assuming that each user has about 100 connections. So the matrix L in that case has $2 \times 100 \times 25 \times 10^8 = 5 \times 10^{11}$ entries. Assuming we need 2 bytes per entry that is 10^{12} bytes which is 1 terabyte which fits on a $50 harddrive.

Weighted networks

In weighted networks we need an additional column in L that keeps track of the weight of a link, for instance:

$$L = \begin{pmatrix} 1 & 3 & 0.1 \\ 2 & 2 & 1.2 \\ 2 & 3 & 3.2 \\ 3 & 5 & 0.5 \\ 4 & 3 & 0.2 \\ 4 & 5 & 9.1 \\ 5 & 5 & 1.2 \end{pmatrix} \quad \text{or} \quad W = \begin{pmatrix} 0 & 0 & 0.1 & 0 & 0 \\ 0 & 1.2 & 3.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 \\ 0 & 0 & 0.2 & 0 & 9.1 \\ 0 & 0 & 0 & 0 & 1.2 \end{pmatrix}$$

1.3.3 Linked lists

Another interesting way to represent a network is by a linked list. Each element in the list contains an array of other elements. In a way, this is the most natural way to represent a network and for some computations it's very useful. The above network is represented like this:

$$\begin{aligned} 1 &: \{3\} \\ 2 &: \{2,3\} \\ 3 &: \{5\} \\ 4 &: \{3,5\} \\ 5 &: \{5\} \end{aligned}$$

It's very intuitive to read this, node one is connected to node 3, node 2 to 2 and 3, etc. It is memory efficient and very useful if the task is to search through a network or run dynamic processes like an epidemic.

Weighted networks

If we have a weighted network then for each link list we need to keep track of the corresponding weights and thus have two lists:

$$\begin{aligned} 1 &: \{3\}, [0.1] \\ 2 &: \{2,3\}, [1.2, 3.2] \\ 3 &: \{5\}, [0.5] \\ 4 &: \{3,5\}, [0.2, 9.1] \\ 5 &: \{5\}, [1.2] \end{aligned}$$

Note that for each row, $\{\cdot\}$ denotes a reference to another node, whereas $[\cdot]$ holds actual values and not references.

1.4 Node degree

The simplest yet most frequently measure used to characterize a node is its degree, the number of connections it has to other nodes. Let's consider undirected networks first. The degree of

node i can be easily compute from the adjacency matrix A:

$$k_i = \sum_j A_{ji}$$

If we sum over all degrees k_i we count all the links twice because all links have two ends

$$\sum_i k_i = \sum_{ij} A_{ij} = 2L \tag{1.2}$$

where L is the number of links in the network. Thus, if we want to compute the average node degree of the entire network it's given by

$$\langle k \rangle = \frac{1}{N} \sum_i k_i = \frac{2L}{N},$$

which is twice the number of links devided by the number of nodes, which is intuitive.

1.4.1 Density of a network

This is also often used for defining the density of a network which is the ratio of links L and the maximum number of links which is $N(N-1)/2$ (in an undirected network), so

$$\rho = \frac{2L}{N(N-1)} = \frac{\langle k \rangle}{N-1} \approx \frac{\langle k \rangle}{N}$$

the mean degree per node or the fraction of links a node has on average normalized by the potential number of neighbors.

1.4.2 Directed Networks

In directed networks the situation is a bit more difficult but manageable. Since $A_{ij} \neq A_{ji}$ in general the number of links leaving a node is not necessarily the same as the number of links arriving at a node. Thus we have in-degree and out-degree

$$k_i^{\text{in}} = \sum_j A_{ij} \quad \text{and} \quad k_i^{\text{out}} = \sum_j A_{ji}. \tag{1.3}$$

The total number of links leaving nodes are the total number of links and they also must be equal to the number of incoming links which is why

$$L = \sum_i k_i^{\text{in}} = \sum_i k_i^{\text{out}} = \sum_{ij} A_{ij}$$

Note that this is different from Eq. (1.2) by a factor of 2. Even if $A_{ij} = A_{ji}$ and thus in-degree and out-degree are the same for each node in a symmetric network, between a pair of nodes there's always also a pair of links.

Because of (1.3), the average in- and out-degree must be the same

$$\left\langle k^{\text{in}} \right\rangle = \frac{1}{N} \sum_i k^{\text{in}} = \frac{1}{N} \sum_{ij} A_{ij} = \frac{L}{N} = \frac{1}{N} \sum_i k^{\text{out}} = \left\langle k^{\text{out}} \right\rangle.$$

1.4.3 Weighted Networks

In weighted networks it also makes sense to just focus on the connectivity, given by the adjacency matrix A (i.e. computing the degree of nodes). Additional insight is gained by using the weight matrix W. For instance we can compute the capacity ϕ_i of a node, adding the weights of its links

$$\phi_i = \sum_j W_{ij}$$

If the network is directed and weighted we have in-capacity and out-capacity

$$\phi_i^{\text{in}} = \sum_j W_{ij} \quad \text{and} \quad \phi_i^{\text{out}} = \sum_j W_{ji}. \tag{1.4}$$

and just like in the situation with degree the average in- and out-capacity must be equal

$$\langle \phi^{\text{in}} \rangle = \frac{1}{N} \sum_i \phi_i^{\text{in}} = \frac{1}{N} \sum_{ij} W_{ij} = \frac{\Omega}{N} = \frac{1}{N} \sum_i \phi_i^{\text{out}} = \langle \phi^{\text{out}} \rangle,$$

where Ω denotes the capacity of the entire network.

1.4.4 Example: The Worldwide Air Traffic Network

This network consists of $N = 4,092$ airports (nodes) and $L = 25,477$ links. The network is weighted and symmetric. The weight w_{ij} quantifies the number of passengers travelling between nodes i and j per day. Using the concepts above we can do a litte statistics

Worldwide Air Traffic Network		
$\langle k \rangle$	12.4521	
$k_{\max}$	345	FRA
$\langle \phi \rangle$	2177.725 passengers/day	
$\phi_{\max}$	158,169 passengers/day	ATL
ρ	0.30438%	
$\langle w \rangle$	248	
$w_{\max}$	28370	Sapporo -> Tokio

We see that the network is not very densely connected. We also see that the mean strength of a connection is almost two orders of magnitude smaller than the maximum strength (ATL, Atlanta Intl. Airport). Tables 1.1 and 1.2 list the top ten airports in terms of degree and capacity. Clearly these are different, airports that are strongly connected are not necessarily the ones that have a strong flux. However, in general a trend exists between degree and capacity. This is shown in Fig. 1.8.

Rank	Airport	k
1	Frankfurt International Apt	345
2	Paris Charles de Gaulle Apt	301
3	Amsterdam	293
4	Munich International Airport	274
5	Atlanta Hartsfield-Jackson Intl Apt	269
6	London Gatwick Apt	245
7	Chicago O'Hare International Apt	216
8	Milan Malpensa Apt	211
9	Dusseldorf International Airport	211
10	London Heathrow Apt	208

Table 1.1: Top ten airports worldwide in terms of their degree k. European Airports have a comparatively high degree.

Rank	Airport	ϕ
1	Atlanta Hartsfield-Jackson Intl Apt	1.581698e+05
2	Chicago O'Hare International Apt	1.364909e+05
3	London Heathrow Apt	1.257413e+05
4	Los Angeles International Apt	1.094880e+05
5	Tokyo Haneda Apt	1.073322e+05
6	Dallas/Fort Worth Intl Apt	1.058395e+05
7	Paris Charles de Gaulle Apt	9.959137e+04
8	Frankfurt International Apt	9.745675e+04
9	Beijing Capital Apt	8.283938e+04
10	Madrid Barajas Apt	8.027420e+04

Table 1.2: Top ten airports worldwide in terms of their capacity ϕ measured in passengers per day. US airports have a comparatively high degree.

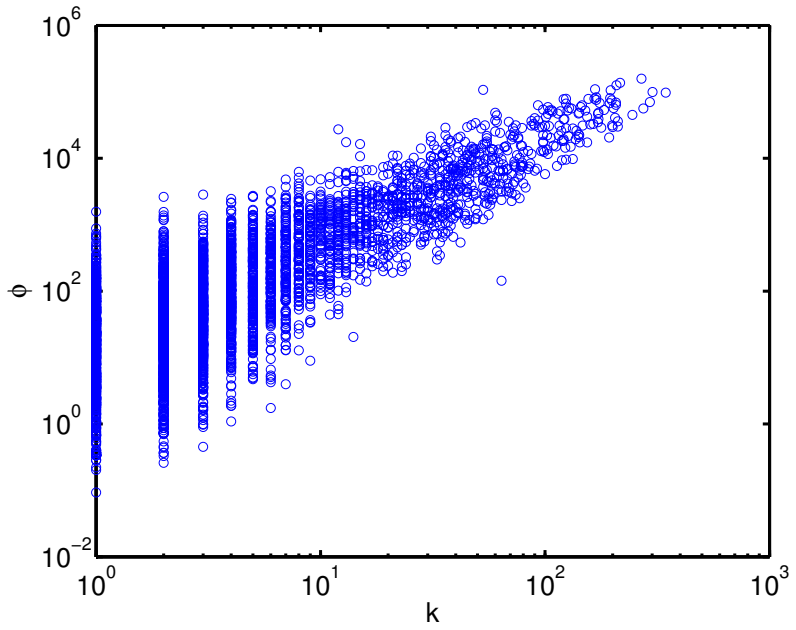


Figure 1.8: Correlation of degree k and capacity ϕ in the worldwide air transportation network.

1.4.5 Example: Florida Foodweb

This is a network of species that interact with one another by eating themselves. We have $N = 122$ species and the number of links in the network is $L = 1767$. This is a directed network. A connection from $i \to j$ exists if carbon is going from species i to species j in other words if species j eats species i.

Florida Foodweb		
$\langle k^{\text{in}} \rangle = \langle k^{\text{out}} \rangle$	14.4836	
$k^{\text{in}}_{\text{max}}$	39	Crocodiles
$k^{\text{out}}_{\text{max}}$	60	Predatory Shrimp
ρ	11.97%	

In comparison to the transportation network this network is much denser and highly connected. Note that although the average in-degree and out-degree are the same as they have to be, the maximum of both quantities are different. Crocodiles have a diverse menu, and predatory shrimp get eaten by lots of other species. A top ten list of predadors and prey is given in the tables below.

Rank	Species	k_{in}
1	Crocodiles	39
2	Raptors	37
3	Predatory Ducks	37
4	Greeb	36
5	Dolphin	34
6	Scianids	34
7	Big Herons	33
8	Loon	33
9	Pompano	33
10	Other Demersal Fishes	31

Table 1.3: Most diverse predators in the florida foodweb.

Rank	Species	k_{out}
1	Predatory Shrimp	60
2	Herbivorous Shrimp	60
3	Pink Shrimp	54
4	Omnivorous Crabs	45
5	Bivalves	43
6	Goldspotted killifish	37
7	Detritivorous Gastropods	37
8	Detritivorous Amphipods	36
9	Herbivorous Amphipods	35
10	Suspension Feeding Polych	35

Table 1.4: most popular prey.

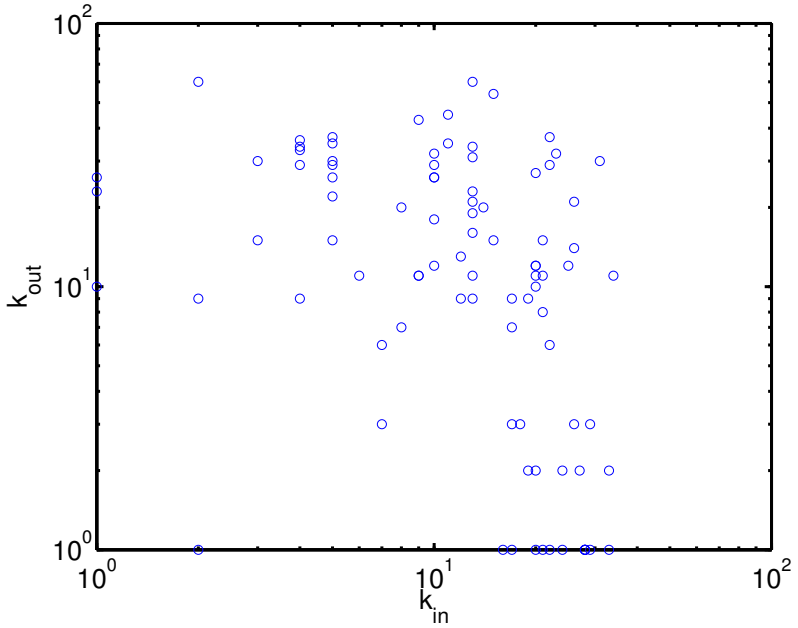


Figure 1.9: Correlation of k_{in} and k_{out} in the Florida foodweb. The data suggest that species don't exist that have both a low in and out degree. Obivously no correlation between out degree and in degree exists.

1.5 Paths

Given a network it is often of interest to determine in what way one can get from one node to another node by a sequence of intermediate nodes. Let's consider undirected unweighted networks first. A path is a sequence of nodes

$$P = n_1, ..., n_k$$

such that each pair n_i, n_{i+1} with $i = 1, ..., k-1$ has a link. Generally some of the nodes (and links) can appear multiple times in a such a sequence.

Components and Paths

Paths make it easy to define compoments of networks. A component of a network is a collection of nodes such that a path exists between any two pairs of nodes. Obviously no path exists between two different components. The situation is a little more complicated in directed networks which will be discussed later.

Self-Avoiding Paths

Those are paths in which nodes or links are only visited once. Paths that do not visit links multiple times but nodes are not self-avoiding.

1.5.1 Length of a path

There are various ways of measuring the length of a path, depending on the type of network and application, various measures are useful.

Hops

The most intuitive measure is just based on the number of hops performed, e.g. in a sequence of k nodes that length is

$$l_{\mathrm{T}} = k - 1.$$

Weighted networks

In weighted network each link has a magnitude w_{ij} and depending on the application this can be translated into distance, most often a reciprocal relation is assume, i.e.

$$d_{ij} = \frac{1}{w_{ij}},$$

which means that the stronger the weight the closer two nodes i and j. A the length of a path P can be computed using this distance

$$l_w = \sum_{i=1}^{k-1} \frac{1}{w_{n_i n_{i+1}}}$$

1.5.2 Paths and the adjacency matrix

For unweighted networks which are determined by their adjacency matrix, this matrix is very useful in computing the number of paths between two chosen nodes. Let's assume we have three nodes i, $j$and k. A path starting a i via j to k exists if there's a link connection i and j and j and k. That means only if

$$A_{ij}A_{jk} = 1$$

Therefore if we sum over j the result is the number of paths that connect i and k

$$n_2(i|k) = \sum_j A_{ij}A_{ji}.$$

The rhs. of this equation is the multiplication of the adjacency matrix with itself, so

$$n_2(i|k) = \left[\mathbf{A}^2\right]_{ik}$$

So the number of paths of length 2 connection i and k are given by element ij of the square of the adjacency matrix. Likewise if we chose a start node i and a terminal node k and would like to know how many paths of length l can be constructed from i to k this is given by

$$n_l(i|k) = \left[\mathbf{A}^l\right]_{ik}. \tag{1.5}$$

Let's look at the example again, depiced in Fig. 1.7. The adjacency matrix is given by

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

So let's count the number of paths connecting nodes that have length 3, to this end we need to look at

$$\mathbf{A}^3 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 1 & 2 & 1 \end{pmatrix}$$

This means that exactly one path exists that goes from node 1 to node 5 that has length 3. and two different ways exist to go from 4 to 5.

1.5.2.1 Loops/cyles

Of particular interest are the number of paths that start at node i and end there as well. According to Eq. (1.5) the number of $i-$loops of length l is given by

$$n_l(i|i) = \left[\mathbf{A}^l\right]_{ii},$$

the diagonal elements of the $l-$th power of the adjacency matrix. The total number of loops of length l in the entire network is given by the sum

$$n_l = \sum_i n_l(i|i) = \mathrm{Tr}\mathbf{A}^l$$

In the above example we see that there are always only two loops for any path length. The reason is clear if I start on nodes 1,3 and 4 I always end up on node 5. The only possibility to loop are the self loops on nodes 2 and 5.

1.5.2.2 Counting triangles in networks

Sometimes it's useful to be able to count triangles in networks. Triangles are often used in interpreting network structures. In friendships a triangle means that two friends of a person are also friends. According to what we discussed above we could count the triangles in an undirected network by

$$n_\Delta = \frac{1}{6}\mathrm{Tr}\mathbf{A}^3 \tag{1.6}$$

One of the things we have to understand is that the operation really counts paths, so a path $i \to j \to k \to i$ is counted as well as $j \to k \to i \to j$ and also $k \to i \to j \to k$ plus the same in both directions. In directed networks, the number of triangles is given by. That's why we have to devide by 6.

$$n_\Delta = \frac{1}{3}\mathrm{Tr}\mathbf{A}^3$$

Let's see if this works. Fig 1.10 depicts a planar triangular graph with 500 nodes. for which the number of triangles were compute using the above formula.

1.5.3 Paths and Eigenvalues

We can use a little linear algebra to relate shortest paths to eigenvalues of the adjacency matrix **A**. Let's first consider an undirected network. In this case the adjacenc y matrix is symmetric, $A_{ij} = A_{ji}$. A symmetric matrix has only real eigenvalues. From linear algebra we now that in this case we can write

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{U}^T \tag{1.7}$$

where **U** is an orthogonal matrix (consisting of the eigenvector basis of **A**) and **D** is a diagonal matrix with eigenvalues on its diagonal. For an orthogonal matrix we have $\mathbf{U}\mathbf{U}^T = \mathbf{1}$, and thus

$$\mathrm{Tr}A = \mathrm{Tr}(\mathbf{U}\mathbf{D}\mathbf{U}^T) = \mathrm{Tr}(\mathbf{D}\mathbf{U}^T\mathbf{U}) = \mathrm{Tr}(\mathbf{D}) = \sum_i \lambda_i$$

where λ_i are the eigenvalues of **A**. Also

$$\mathrm{Tr}(\mathbf{A}^l) = \mathrm{Tr}(\underbrace{\mathbf{U}\mathbf{D}\mathbf{U}^T\mathbf{U}\mathbf{D}\mathbf{U}^T....\mathbf{U}\mathbf{D}\mathbf{U}^T}_{l-\text{times}}) = \mathrm{Tr}(\mathbf{U}\mathbf{D}^l\mathbf{U}^T) = \mathrm{Tr}(\mathbf{D}^l) = \sum_i \lambda_i^l.$$

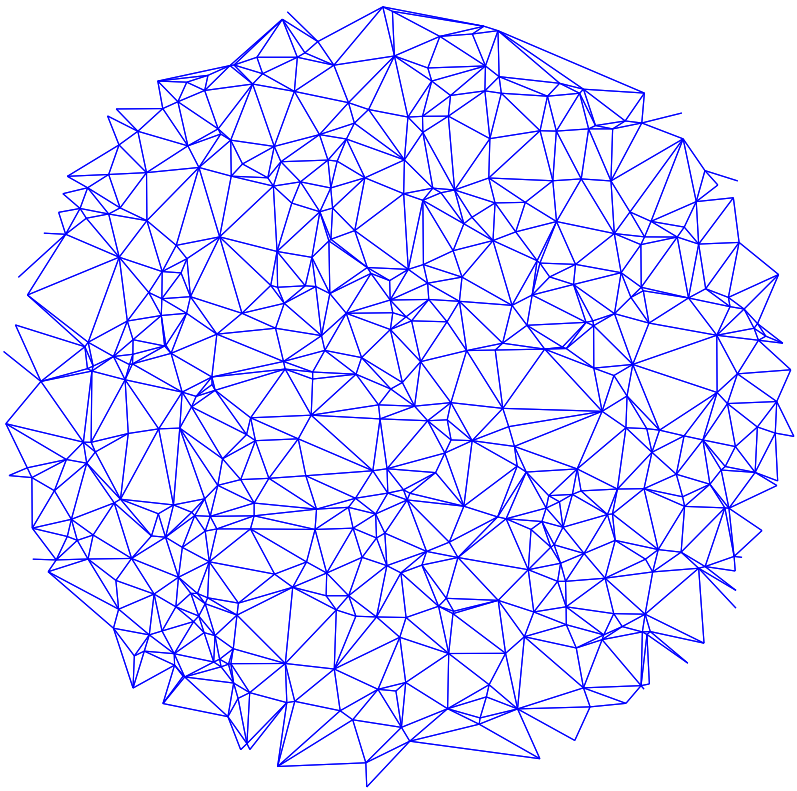


Figure 1.10: Triangular, undirected planar network consisting of 500 nodes. Given the expression 1.6 and the 500×500 adjacency matrix $\mathbf{A}$ we compute quickly that this network has 884 triangles.

And thus we can write

$$n_l = \sum_i n_l(i|i) = \mathrm{Tr}\mathbf{A}^l = \sum_i \lambda_i^l$$

where λ_i is the i^{th} eigenvalue of $\mathbf{A}$. This is very interesting because the eigenvalues, let alone their powers, are not whole numbers but the above sum must end up being a whole number.

1.5.3.1 Directed networks

The above reasoning assumes undirected networks, i.e. symmetric $\mathbf{A}$. For undirected networks, less can be said about the spectrum. However, Something similar to Eq. (1.7) exists called the Schur decomposition which states than any $N \times N$ matrix can be writen as

$$A = UTU^t$$

where T is an upper triangular matrix. The reaoning above then suggest that the trace of A^l is given by

$$Tr(A^l) = \sum_i T_{ii}$$

where T_{ii} are the diagonal elements of the upper triangular matrix T.

1.5.4 Geodesic shorest paths Paths

Obivously, given two nodes i and j, we find typically that many paths of various length exist between the two nodes. Imagining the set of all such paths $\mathscr{P}_{ij}$. We can define a subset $\mathscr{S}_{ij}$ of shortest paths, those that have the minimal number of steps. These are called geodesic paths or simply shortest paths. The union of all shortest paths, i.e. for all pairs of nodes i and j is the set of all shortest paths in the network

$$\mathscr{S} = \bigcup_{i,j} \mathscr{S}_{ij}.$$

Note that in an undirected networks if a path P_{ij} is a shortest path from i to j then it must also be a shortest path from j to i. This is not true for directed networks.

Example: worldwide air transportation network

Say we want to compute a shortest path from Raleigh-Durham Airport (RDU) to Hannover Airport (HAJ) in Germany. The shortest path length from RDU to HAJ is 3, and one of the shortest paths is

$$RDU \rightarrow YYZ \rightarrow HAJ$$

the airport YYZ is Toronto International Airport.

1.5.4.1 Shortest paths in weighted networks

But let's recall that for weighted networks it is more useful to compute distance by inverse weight

$$d_{ij} = \frac{1}{w_{ij}}. \tag{1.8}$$

Using this measure the shortest route has length 5 and is given by

$$RDU \rightarrow ATL \rightarrow ORD \rightarrow LHR \rightarrow MUC \rightarrow HAJ.$$

This often makes sense in networks that are completely connected. Say we have a weighted network in which $w_{ij} > 0$ for every pair of nodes. According to what we discussed earlier, it doesn't really make much sense to regard this as a network because the connectivity is trivial. One exception is shortest paths in fully connected networks.

Example: The world trade network

A good example is the world trade network shown in Fig. **??**. This is a fully connected network of 191 countries (data based on the year 2001). The network is symmetric and weighted, so everything is encoded in the weight matrix W_{ij}. The shortest path computed by quantifying distance of each step according to (1.8). So, although the US, e.g. have direct links to all the other nodes, using the shortest path to other countries often involves intermediate steps. The interpretation of this particular network would be that the likelihood of a trade between two countries may be greater if untermediate steps are involved.

1.5.4.2 Network diameter

Geodesic paths are often use to define the notion of the diameter of the network. say we compute, for all pairs of nodes i and j the set or shortest paths $\mathscr{S}$. There are two definitions that are useful for defining the diameter of the network

- The longest path in the ensemble of shortest paths

$$D_0 = \max(\mathscr{S})$$

- the average shortest path in the ensemble of shortest path

$$\bar{D} = \frac{1}{M}\sum_k P_k$$

  where $P_k \in \mathscr{S}$.

Let's again compare two networks, the Worldwide Air transportation network, and the planar, triangular grid of Fig. 1.10 except for we chose a larger one with 4000 nodes, to be comparable to the WAN. The table shows the same comparison for the US air transportation network and a triangular grid with 400 nodes.

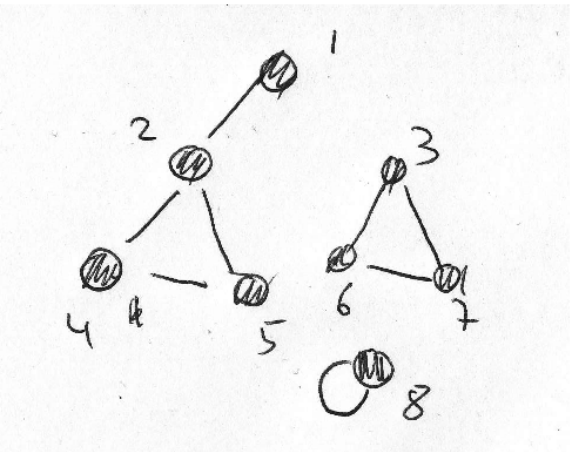


Figure 1.11: Three component undirected network

	Worldwide Airtransport.	Triangular Grid	US Airtransport.	Triangular Grid
	4039	4000	459	450
D_0	18	41	10	16
$\bar{D}$	6.54	21.7	3.65	8.15

We see from this that the average shortest path length in the planar graphs is about a factor of 3 larger than in the real transportation network. This is because of the long range connections and the small world effect which we will discuss in detail later.

1.5.5 Eulerian and Hamiltonian paths

In some applications it is important to search through the network and visit either all nodes or traverse all links. Paths that accomplish this are called Eulerian paths and Hamilonian paths:

- Eulerian Path: visit each link exactly once (like in the Königsberg problem)
- Hamiltonian Path: visits each node exactly once.

1.6 Components

When investigating the stability of networks the concept of components is essential. Components are disconnected subnetworks of a network. It straightforward to use paths as a definition of components. For each pair of nodes i, j in a component of an undirected network a path must exist that goes from i to j. If the pair is in two components now path exists. Say we have the following adjacency matrix. This corresponds to the network illustrated in Fig. 1.11

It represents an undirected network with three components. That's not seen because of the labeling of the nodes.

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

We can write another adj. matrix, relabeling the nodes $5 \to 3$ and then we see that the new adj. matrix has the form

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} A_1 & 0 & 0 \\ 0 & A_2 & 0 \\ 0 & 0 & A_3 \end{pmatrix}$$

with

$$A_1 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}, \quad A_3 = 0.$$

That means the network can be regarded as a direct sum of subnetworks. Note that the submatrices have all the properties that the original matrix has (symmetry, loop structure etc.). For instance we can still compute the number of paths of lengh l by $\mathrm{Tr}A^l$ because

$$A^l = \begin{pmatrix} A_1^l & 0 & 0 \\ 0 & A_2^l & 0 \\ 0 & 0 & A_3^l \end{pmatrix}$$

and

$$\mathrm{Tr}A^l = \sum_k \mathrm{Tr}A_k^l.$$

1.6.1 Components in directed networks

In directed networks, the situation is more difficult because all the links are one-way streets. So getting somewhere does not imply that you can get back.

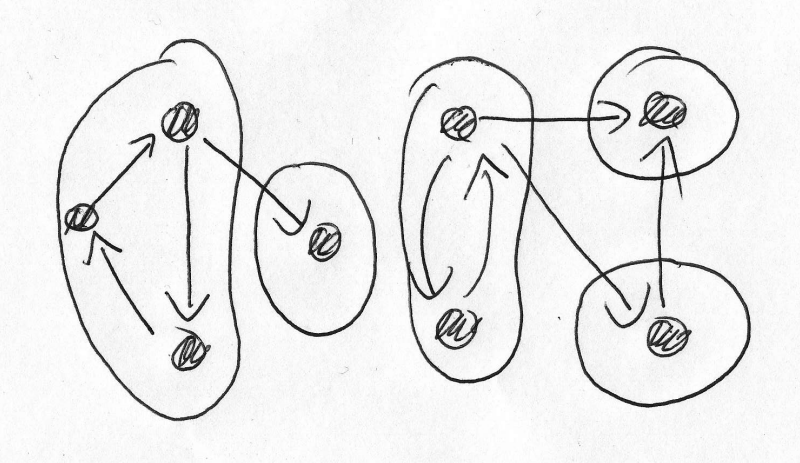


Figure 1.12: Components in directed networks. This network has two weakly connected components and 5 strongly connected components.

1.6.1.1 Weakly connected component

This is just pretending that the links are not directed. It matters only that I can get from one node to the other.

1.6.1.2 Strongly connected components

First we consider that two nodes i and j are strongly connected if there's a path (not necessarily a direct link) from i to j and possiblity a different path from j to i. A strongly connected compoment is a set of nodes such that each pair of them is strongly connected.

1.6.1.3 Out-Component and In-Component

In directed networks it is sometimes of interest to know what part of the network can be reached from a given nodes, and what part of the network can reach a given node. The first is called the out component $C_{\text{out}}(i)$ the second one the in component $C_{\text{in}}(i)$. If I intersect both, I get the subset of nodes that can be reached from i and that can reach i. Therefore the intersection is the strongly connected component of i

$$C(i) = C_{\text{out}} \cap C_{\text{in}}$$

1.7 Bi-Partite networks

Formally, a bi-partite network is an undirected network such that the set of nodes V can be grouped into two disjoint sets V_1 and V_2 and that the links of the network only exist between

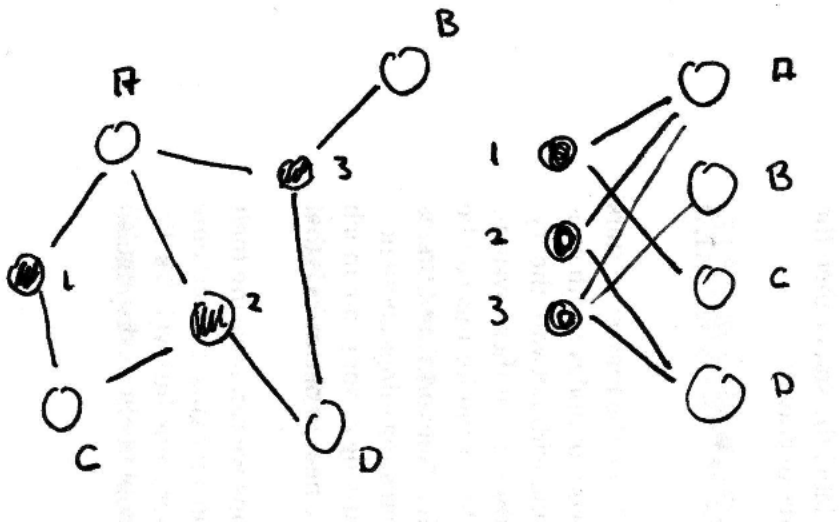


Figure 1.13: A small bi-partite network. The entire network has $N = 7$ nodes. Links only exist between pairs of nodes such that one node is white and on is black. A useful representation of the connectivity is the colinear represensation of the two groups and their linkage in between.

pairs of nodes i and j such that $i \in V_1$ and $j \in V_2$. In other no links exist within V_1 and V_2. In a way that is the opposite of having a network with two disconnected components. Fig. 1.13 illustrates a simple bi-partite network. The best way to display this type of network is a co-linear alignment of the groups of nodes and their links in between.

The adjacency matrix of this particular network looks like this

$$A = \begin{pmatrix} 0 & B \\ B^T & 0 \end{pmatrix}$$

with

$$B = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix}$$

is a 3×4 matrix encoding the links from the perspective of the black nodes and B^T is a 4×3 matrix informing how the white nodes receive links from the black ones. The adjacency matrix of the bi-partite network, has the opposite structure of a network of independent components (i.e. the block-diagonal form). Often it is much more useful to just use the $N_1 \times N_2$ matrix B (denoting the number of nodes in the groups by N_1 and N_2) because it contains all the information. This matrix is called the *incidence matrix*.

1.7.1 Applications

1.7.1.1 Hypergraph expansion

We have previously discussed the concept of hypergraphs, in which subsets of nodes are not pairwise linked by linked by hyperlinks, which connect more nodes than just pairs, see Fig. 1.4. Let's say we have a network of N nodes and we have M groups, or cliques. The hypergraph can visualized as a bi-partite network if we consider the cliques to be nodes of a different type and draw a link from original node i to clique node k if i is in group k. This way we will obtain an $N \times M$ incidence matrix B.

1.7.1.2 Examples

Typical examples appear in recommender systems where we have a pool of people and a pool of things they evaluate. For instance we could have N customers and M products they like, e.g. netflix users and films. Another example could be people (nodes) and restaurants they frequent (groups). Another interesting example from biology are biochemical reactions in which nodes are reactants or metabolites and the groups they participate in are chemical reactions. And yet another well studies system are networks in which the nodes are research authors and the groups the papers they co-author, or the nodes are actors and the groups are films they are part of.

1.7.2 The human disease network

One of the most famous and recent applications of bipartite networks is the construction of the human diseasome, the relation between diseases that are caused by gene malfunction. This network is illustrated in Fig. 1.14. This network was computed from a bipartite network in which nodes are genes and the groups are diseases they cause. We start otu with a set of N genes that are known to cause a set of M diseases. A gene defect i may cause a number of diseases $\alpha_1, ..., \alpha_k$. This generates a bipartite network that can be described by and $N \times M$ incidence matrix B.

As such, B is just a big table, what can we do with it to gain insight about the diseases of the interplay of the genes that cause them?

1.7.3 One-Mode Projections

If we have such a bi-partite network we can produce two types of one-mode projections. We can form and $N \times N$ network of genes and and $M \times M$ network of diseases. Let's assume we have the simplified version of a hypergraph and bipartite networks shown in Fig. 1.15. We have 8 genes labeled 1...8 and 5 diseases labeled A...E. We can generate the $N \times N$ network among genes by setting $A_{ij} = 1$ if two genes i and j cause the same disease. This amounts to connecting all the nodes cause the same disease. All nodes in that cause disease A for example will be connected all-to-all, they form what is called a clique.

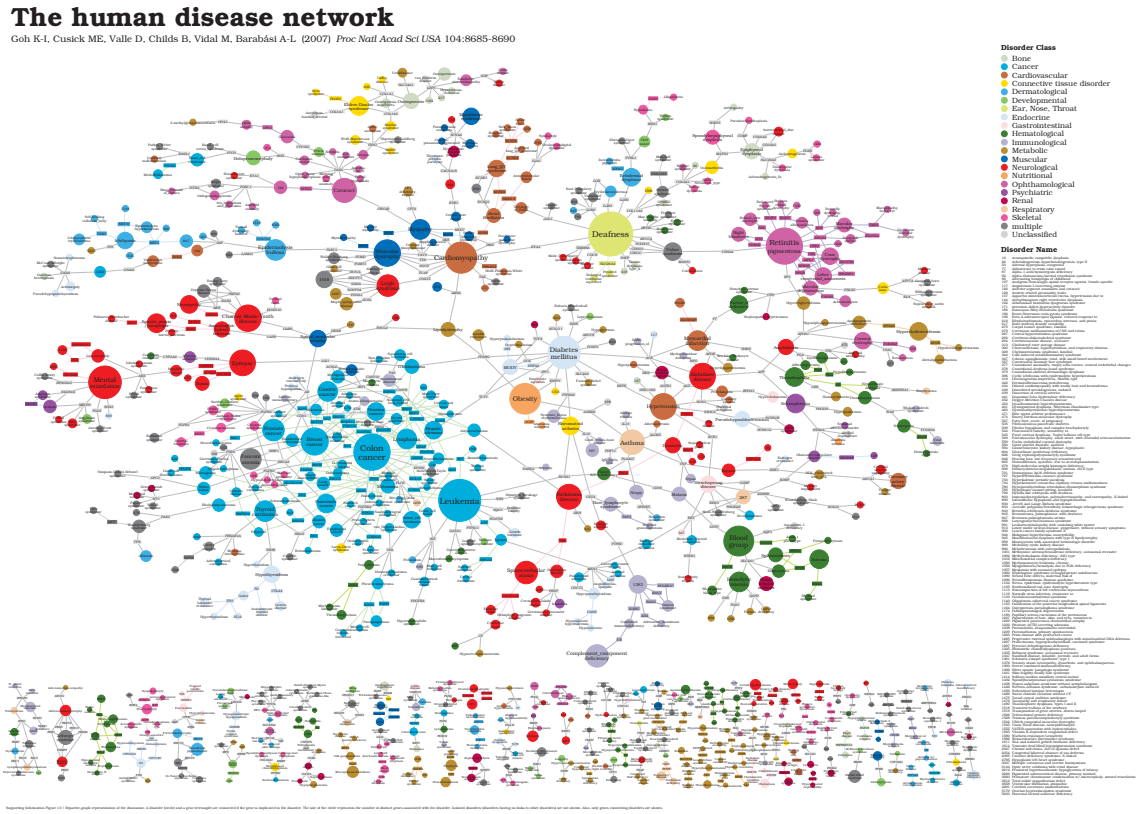


Figure 1.14: The human disease network

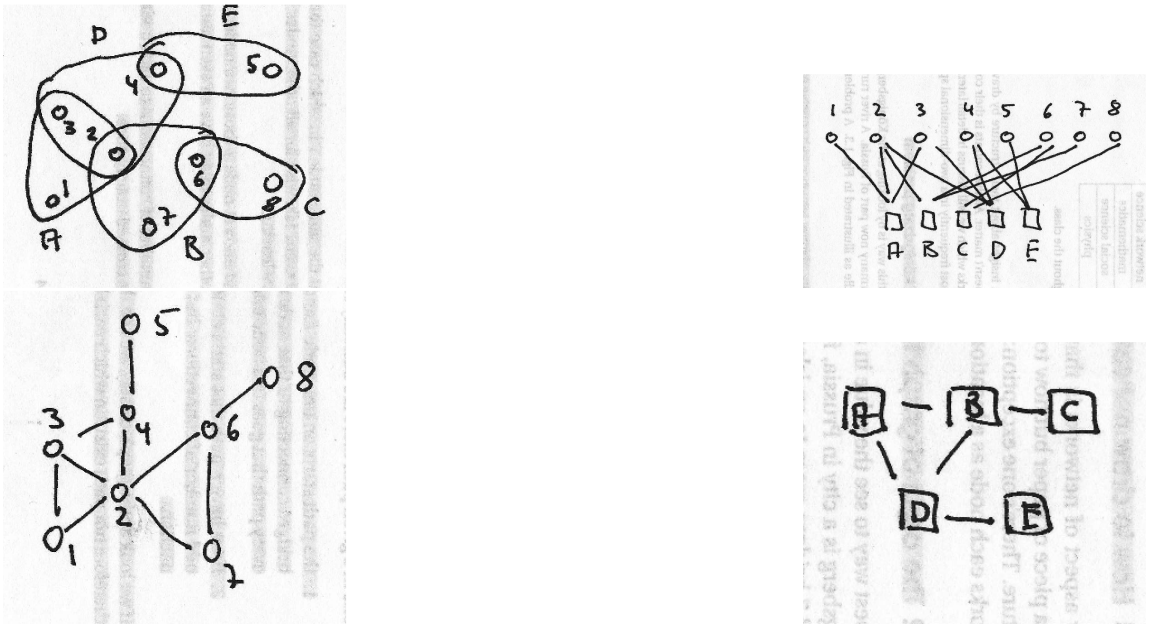


Figure 1.15: top left: a hyper graph consisting of 7 nodes in 5 groups. top right: the equivalent bi-partite networks. lower left: the one-mode projection generating a network among the nodes, lower right: a one-mode projection generating a network among the groups.

Likewise we can form a projection in the other direction. We can generate a network between diseases, connecting the disease α and β if they are caused by at least one common gene. In fact, in this direction it can happen that a disease is caused by multiple genes so we can generate a weighted network $W_{\alpha\beta}$ in which the entries count the number of genes that cause both α and β.

Formally we start out with the $N \times M$ incidence matrix B in which the rows i lavel the genes and the columns α the diseases. For the example in Fig. 1.15 we have

$$B = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Element $B_{i\alpha} = 1$ if gene i can be associated with disease α. Thus a connection between i and j exists if $B_{i\alpha}B_{j\alpha} = 1$ and $B_{i\alpha}B_{j\alpha} = 0$ otherwise. So we can sum over all the diseases to count the number of connections between gene i and j which we define as the weight between i and j

$$W_{ij} = \sum_{\alpha}^{M} B_{i\alpha}B_{j\alpha} = \sum_{\alpha}^{M} B_{i\alpha}B^{T}_{\alpha j} \tag{1.9}$$

and thus

$$\mathbf{W} = \mathbf{B}\mathbf{B}^{T}$$

is a weighted $N \times N$ matrix. Doing this in the example of genetic disease we would expect if W_{ij} is large, then these genes probably have something do to with each other in causing a number of diseases and are thus important. In the above example we get

$$W = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 3 & 2 & 1 & 0 & 1 & 1 & 0 \\ 1 & 2 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 2 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}$$

We see from this and also from Eq. (1.9) that $W_{ij} = W_{ji}$ so as expected we have an undirected weighted networks. What do the weights tell us. Interestingly W has nonzero diagonal elements.

$$W_{ii} = \sum_{\alpha}^{M} B_{i\alpha}B_{i\alpha} = \sum_{\alpha}^{M} B_{i\alpha} = q_i$$

where q_i is the number of diseases that gene i causes.

The other projection direction can also provide interesting information. If $B_{i\alpha}B_{i\beta} = 1$ that means that gene i has causes both diseases α and β which would suggest a relation between those diseases. The number

$$\Omega_{\alpha\beta} = \sum_i^N B_{i\alpha}B_{i\beta} = \sum_i^N B^T_{\alpha i}B_{i\beta}$$

thus counts the similarity between disease α and β by the number of genes that cause it, the matrix

$$\Omega = B^T B$$

is a similarity matrix of all the groups or in our example diseases. So it's an $M \times M$ matrix. For the incidence matrix above we obtain

$$\Omega = \begin{pmatrix} 3 & 1 & 0 & 2 & 0 \\ 1 & 3 & 1 & 1 & 0 \\ 0 & 1 & 2 & 0 & 0 \\ 2 & 1 & 0 & 3 & 1 \\ 0 & 0 & 0 & 1 & 2 \end{pmatrix}$$

Again this is a symmetric matrix. The diagonals Ω_{ii} quantify by how many gene defects cause a disease.

1.7.4 Real Data

The real data set consists of $N = 1777$ genes that are known to cause diseases and a list of $M = 1238$ genetic diseases. The incidence matrix is an 1777×1238 matrix. The incidence matrix has 2673 links, it is therefore very sparse. The density of the incidence matrix is $\rho = 0.12\%$. Fig. 1.16 shows the incidenc matrix. This was constructed from a link table with rows that contain a link from gene to disease $i \to \alpha$, a piece of that table is shown in Tab. 1.5

Now we can perform the one-mode projections, i.e. compute the $N \times N$ gene network and the $M \times M$ matrix of diseases.

1.7.5 The gene network

The gene network has 14982 non-zero entries which means the density is about $\rho = 0.45\%$. Remember that the diagonal elements W_{ii} quantify the number of diseases a defective gene i may play a part in. Fig. 1.17. Some genes play a role in many diseases. For instance the gene TP53 is a tumor suppressor gene. It's failure causes tumor growth.

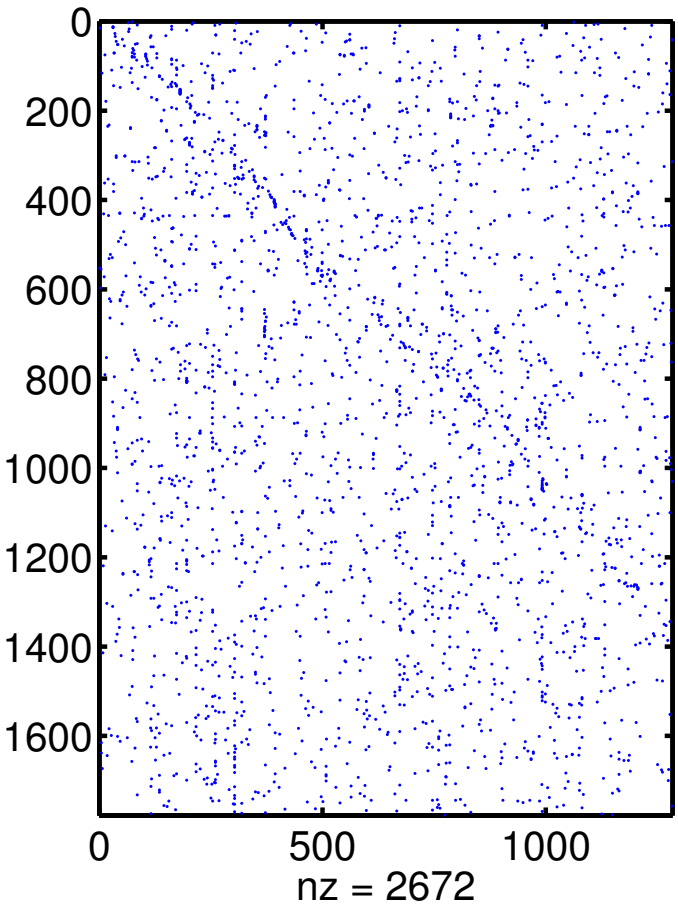


Figure 1.16: The incidence matrix B or the human diseaseome.

Gene	Disease
BMPR1B	Brachydactyly
NTRK2	Obesity
ADAR	Dyschromatosis
SLC5A1	Glucose/galactose malabsorption
RB1CC1	Breast cancer
MEN1	Lipoma
SLC2A4	Diabetes mellitus
CIITA	Rheumatoid arthritis
COL1A1	Caffey disease
ALG8	Congenital disorder of glycosylation

Table 1.5: Genes and Genetic diseases.

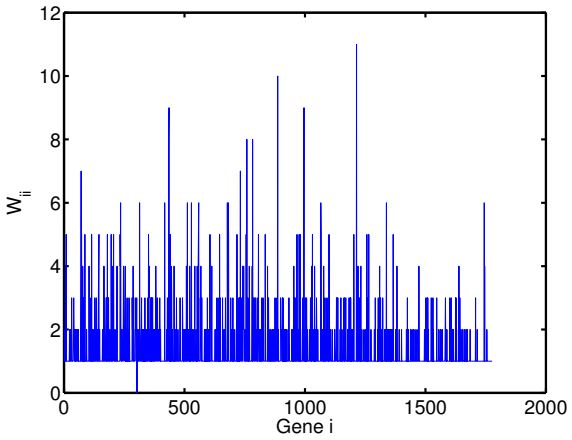


Figure 1.17: The diagonal matrix elements W_{ii} of the one mode projection onto gene space. W_{ii} is shown as a function of gene index i.

Gene	Disease Count	Gene	k
TP53	11	TP53	78
PAX6	10	CCND1	70
PTEN	9	KRAS	68
FGFR2	9	EYA4	64
MSH2	8	RAD54L	60
MEN1	8	PIK3CA	58
FGFR3	8	MSH2	57
LRP5	7	PPARG	56
APC	7	CAV3	50
ARX	6	BRCA2	50

Another interesting questions concerns how genes are linked in the one-mode projection. Which ones are the most connected genes? This could potentially be at the root of the cause of a specific disease. This high degree genes in the one-mode projection network are not necessarily the ones involved in the most groups. We see in the Tables above that there are differences in their ranks. Yet the tumor suppressor TP53 is also a gene with a high degree. We learn from this analysis that this particular gene should be in the focus of research on genetic diseases.

1.7.6 The human disease network

We can proceed along the same lines for the one-mode projection of the bi-partite network onto the $M \times M$ network of human diseases. This network has 3054 links and thus a density of $\rho = 0.18\%$. This network is sparse as well. In this projection we can rank the diseases by the diagonal elements $\Omega_{\alpha\alpha}$ which count how many gene defects can cause the disease. Likewise we

can rank the diseases according to their degree, which would imply that a high degree disease goes together with a number of other diseases. The top ranking diseases are given in the table below:

Disease	Gene Count	Disease	k
Deafness	41	Colon cancer	66
Leukemia	37	Breast cancer	43
Colon cancer	34	Gastric cancer	30
Retinitis pigmentosa	30	Diabetes mellitus	29
Diabetes mellitus	27	Deafness	28
Cardiomyopathy	25	Thyroid carcinoma	26
Mental retardation	24	Pancreatic cancer	26
Blood group	23	Leukemia	26
Obesity	21	Retinitis pigmentosa	25
Breast cancer	19	Ovarian cancer	24

1.8 Centrality

Having seen properties of some networks, it is intuitively clear that some nodes are more essential than others. In transportation networks some nodes are more highly connected than others, in the disease network the gene $TP53$ had a high degree and thus we would think that this gene is very important. The concept of central nodes and links is intimitely linked to the study of network resilience, robustness and susceptibility to targeted attacks. Presumably, failure of central nodes has more dramatic consequences than failure or peripheral nodes (or links).

1.8.1 Degree Centrality

The simplest and most intuitive centrality measure that only requires local information about a node is degree centrality, i.e. centrality according to the degree. The more neighbors a node has the more central it is. Table XX lists the top 10 degree nodes of the US Air Transportation network ($N = 702$) and the top 10 genes in the human diseaseome network (this network is the one-mode projection of the bi-partite network that links genes to genetic diseases).

1.8.2 Eigenvector Centrality

Degree centrality increases proportional to the number of neighbors a node has. It does not distinguish between the types of neightbors. For instance two nodes may have $k = 10$ but one is connected to nodes that also posess a high degree whereas the other may be connected to nodes that all possess a low degree. Eigenvector centrality accounts for this. Let's assume that we have a network described by the adjacency matrix $\mathbf{A}$. Let's assume we denote the centrality of a node i by x_i. And we would like this centrality to be proportional to the sum of the centralities

Rank	degree	Node
1	181.00	Atlanta Hartsfield-Jackson Intl Apt
2	164.00	Minneapolis International Apt
3	154.00	Dallas/Fort Worth Intl Apt
4	143.00	Chicago O'Hare International Apt
5	140.00	Las Vegas McCarran International Apt
6	140.00	Houston George Bush Intercontinental Ap
7	139.00	Cincinnati Northern Kentucky Intl Apt
8	138.00	Denver Intl Apt
9	135.00	Detroit Wayne County
10	109.00	Salt Lake City

Rank	degree	Node
1	70.00	TP53
2	70.00	CCND1
3	65.00	KRAS
4	64.00	EYA4
5	55.00	PPARG
6	54.00	RAD54L
7	51.00	PIK3CA
8	48.00	MSH2
9	48.00	ACE
10	47.00	GJB6

Table 1.6: Rank of nodes in the US Air Transportation network and a genetic network according to degree centrality.

of the neightbors it connects to, so

$$x_i \propto \sum_j A_{ij} x_j$$

We can set both sides equal by introducting a proportionality constant

$$\lambda x_i = \sum_j A_{ij} x_j$$

In matrix form this reads

$$\mathbf{Ax} = \lambda \mathbf{x}$$

where the vector $\mathbf{x}$ is the vector of centralities of the nodes in the network. Of course this equation has generally many solutions. But we would like to have a centrality measure that is non-negative $x_i \geq 0$. Because we are dealing with an undirected network $\mathbf{A}$ is symmetric and it only has non-negative entries. Now we've got this theorem called the Frobenius-Perron theorem that states, that the largest eigenvalue of all the eigenvalues is the only one that has an eigenvector with non-negative entries x_i. So all we need to to is compute the spectrum, look for the largest eigenvalue and the corresponding eigenvector $\mathbf{x}$. The entries of this eigenvector are the eigenvector centralities of the nodes.

A good way to normalize this is by

$$y_i = \frac{x_i}{\langle x \rangle}.$$

This way, if all the nodes generate the same x_i all their centralities are one.

Tab. 1.7 shows the eigenvector centrality of the US-Air transportation network as well as the gene network. Compare to Tab. 1.6. Interesting are of course those nodes that show a deviation from a perfect correlation

Rank	EV-centrality	Node	Rank	EV-centrality	Node
1	10.65	Atlanta Hartsfield-Jackson Intl Apt	1	21.29	EYA4
2	9.96	Chicago O'Hare International Apt	2	20.77	PCDH15
3	9.84	Minneapolis International Apt	3	20.77	USH1C
4	9.80	Cincinnati Northern Kentucky Intl Apt	4	20.77	MYO7A
5	9.73	Dallas/Fort Worth Intl Apt	5	20.77	CDH23
6	9.47	Detroit Wayne County	6	20.65	GJB6
7	9.25	Houston George Bush Intercontinental Ap	7	20.58	GJB3
8	8.99	Newark Liberty International Apt	8	20.57	JAG1
9	8.95	Las Vegas McCarran International Apt	9	20.57	COL11A2
10	8.59	Cleveland Hopkins International Apt	10	20.57	GJB2

Table 1.7: Eigenvector centrality for the two networks.

1.8.3 Closeness Centrality

Degree and eigenvector centrality are based on topological features along and take into account only information in the neighborhood of a node i. Using the concepts of paths and distances (which is global information) we can get a more metric notion of centrality. An analogy to mechanics is usefull. Say we have a three dimensional object defined by a three dimensional region V and say some constant mass density $\rho(\mathbf{x})$. The total mass of the object is given by the integral

$$M = \int_V d^3\mathbf{x}\rho(\mathbf{x}).$$

The center of mass is given by

$$\langle \mathbf{x} \rangle = \frac{1}{M}\int_V d^3\mathbf{x}\mathbf{x}\,\rho(\mathbf{x}).$$

Given some point in the mass $\mathbf{x}_0$ we could define its centrality by the distance to the center of mass

$$C(\mathbf{y}) = |\mathbf{y} - \langle \mathbf{x} \rangle|$$

Now let's assume that the mass density is concentrated at N locations $\mathbf{x}_i$. Mathematically we would write

$$\rho(\mathbf{x}) = \sum_i^N \delta(\mathbf{x} - \mathbf{x}_i)$$

where the funtion $\delta(\mathbf{x})$ is strongly peaked, then

$$C(\mathbf{y}) = \left| \frac{1}{N}\sum_i (\mathbf{y} - \mathbf{x}_i) \right|$$

Likewise we can use the set of shortest paths of node i to all the remaning nodes as a measure of centrality. Let's assume denote by l_{ij} a shortest path from i to j. For all the destination nodes

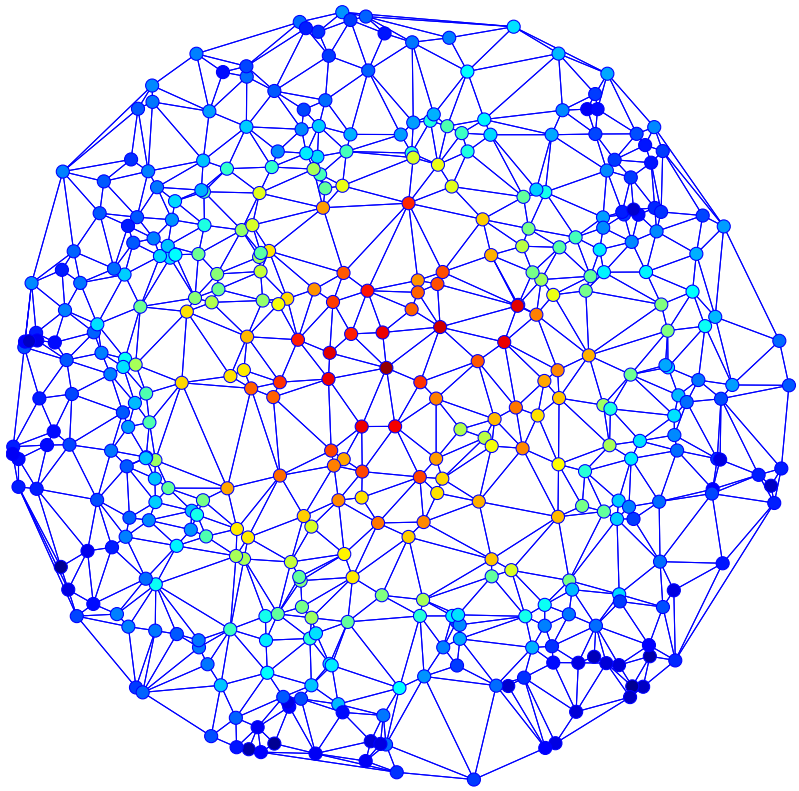


Figure 1.18: Closeness centrality in a planar graph. The nodes are colored according to closeness centrality.

we can compute

$$\langle l(i) \rangle = \frac{1}{N-1} \sum_j l_{ij}$$

assuming that the shortest paths are unique. This is the aerage shortest path from node i to the rest of the network. If that number is small, we are not very far away from everything, which means that we are close to the center, if that number is large, we are far way from the rest of the network. This means that the inverse

$$x_i = \frac{1}{\langle l(i) \rangle}.$$

could be used to quantify centrality in a traditional way. Let's see if this works with a planar graph. Fig. 1.18 shows that in fact nodes with a high closeness centrality are in fact close to the barycenter of the network.

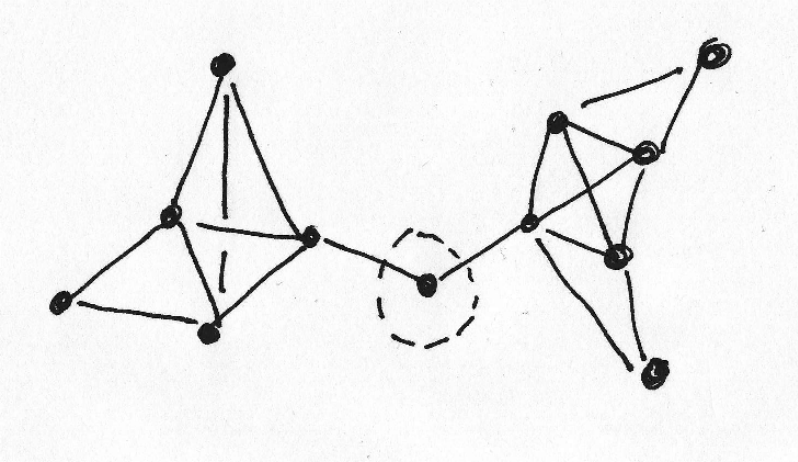


Figure 1.19: The meaning of betweenness centrality. The node in the center has a small degree but a large betweenness centrality.

1.8.4 Betweenness Centrality

A little bit more sophisticated is the concept of betweenness centrality. This measure tries to capture the situation illustrated in Fig. 1.19 in this network, one of the nodes connects two parts of the network. Clearly this node is important in the sense that if it were removed it would disconnected the whole network. Note that a high degree is not required for this function, in fact in the example network other nodes have a more important role. The way to capture this type of centrality is betweenness: the fraction of shortest paths that pass through a node. Let's denote the set of all shortest paths connecting all $N(N-1)/2$ pairs of node i and j by $\mathscr{S}$ and the subset $\mathscr{B}_i$ the subset of shortest paths that pass through node i. The betweenness

$$b_i = \frac{B_i}{S}$$

is defined as the number of elements in the set $\mathscr{B}$ by the total number of shortest paths in the network S. Let's again look at two examples: the US air transportation network and the planr triangular graph. Note that betweenness, just like closeness is small on the border of the planar graph. In the US air transportation network we see that although most of the large airports also have a high betweenness, there are some that have a high betweenness although their degree centrality is small. Those airports for instance that serve as connectors to Alaska.

1.9 Centrality in Weighted Networks

Most of the concepts are generalizable for weighted networks, with some expection.

Rank	Betweenness	Node
1	0.78	Anchorage International Apt
2	0.26	Seattle/Tacoma International Apt
3	0.24	Minneapolis International Apt
4	0.21	Fairbanks International Apt
5	0.18	Denver Intl Apt
6	0.17	Atlanta Hartsfield-Jackson Intl Apt
7	0.16	Bethel Municipal Apt
8	0.13	Dallas/Fort Worth Intl Apt
9	0.12	Las Vegas McCarran International Apt
10	0.11	Houston George Bush Intercontinental Ap

Table 1.8: Betweenness centrality in the US Air transportation network

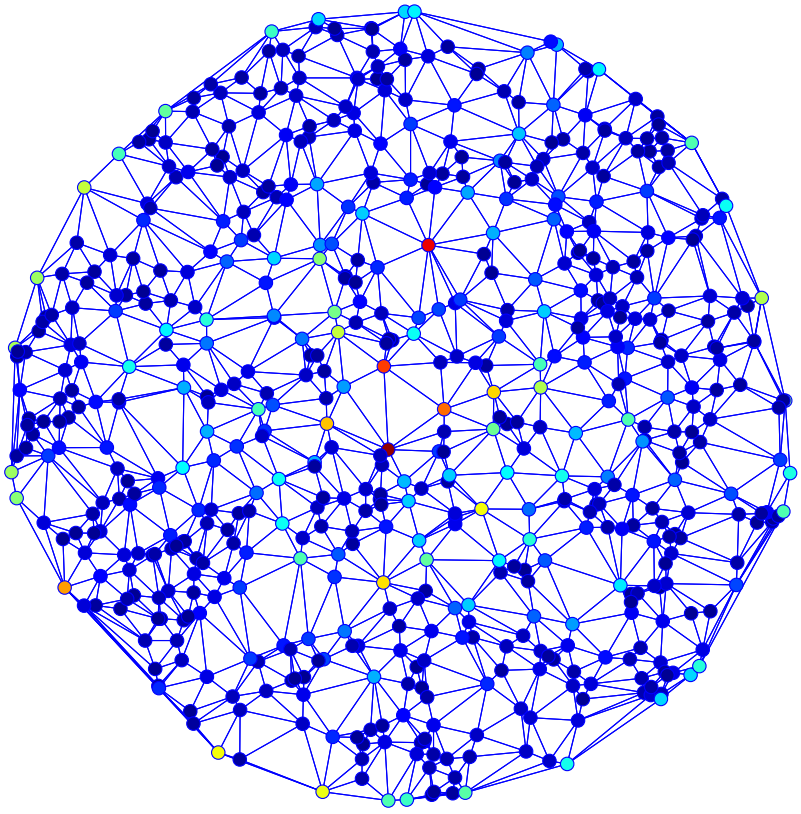


Figure 1.20: betweenness centrality in a planar network.

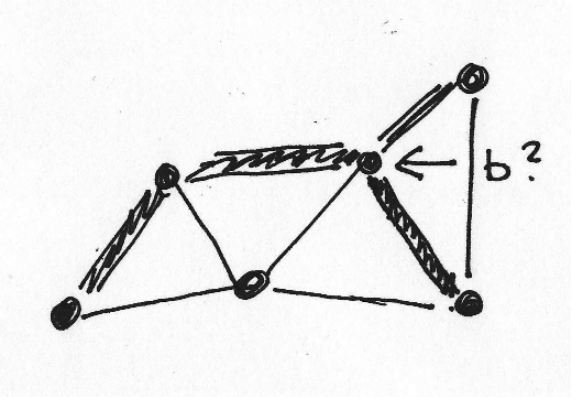


Figure 1.21: Shortest paths in weighted networks are different and thus impact on the betweenness of nodes.

1.9.1 Generalized degree (capacity) centrality

In weighted networks we can associate centrality with the capacity ϕ_i of node i, i.e.

$$\phi_i = \sum_j W_{ij}$$

1.9.2 Eigenvector centrality

If we are dealing with an undirected weighted network, i.e. if $W_{ij} = W_{ji}$ then the Frobenious-Perron Theorem holds as well, which means that we start with

$$\lambda x_i = \sum_j W_{ij} x_j$$

which defines the eigenvector centrality x_i for node i by finding the largest eigenvalue λ of the weight matrix $\mathbf{W}$.

1.9.3 Closeness and betweenness centrality

When we take into acount weights, that can change the effective distance between nodes. Remember that we defined the distance between connected nodes in a weighted networks according to

$$d(i,j) = \frac{1}{w_{ij}}.$$

and the length of a path $n_1, ..., n_k$ with $k-1$ legs is given by

$$l_P = \sum_{i=1}^{k-2} d(n_i, n_{i+1}) = \sum_{i=1}^{k-2} \frac{1}{w_{n_i, n_{i+1}}}.$$

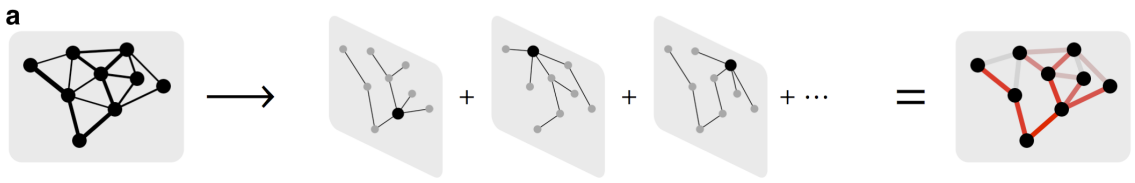


Figure 1.22: Link Salience, the superposition of shortest path trees.

The definition of weighted closeness centrality uses the same equation as in the unweighted networks, i.e.

$$x_i = \frac{1}{\langle l(i) \rangle}.$$

but the average shortest path

$$\langle l(i) \rangle = \frac{1}{N-1} \sum_j l_{ij}$$

of a node i could be quite different since the shortest path from i to some other node j could be different if weights are taken into account. The same argument is true for betweenness centrality in weighted networks. It's still defined as the fraction of shortest paths, but the set of shortest paths change if weights are taken into account.

1.10 Link Betweenness

Although predominantly people focus on nodes and their centrality it is reasonable to also quantify how important individual links are in a network. For example, a large fraction of shortest paths could go through a single link and this link would serve as a bridge, connecting two parts of a network. Therefore, given a link (i, j) the betweenness centrality of that link

$$B_{ij} = \frac{\text{\# of shortest paths going through } ij}{\text{total \# of shortest paths in network}}.$$

is the number fraction of shortest paths that pass through that link. It is convenient to represent this as a matrix $\mathbf{B}$. Let's assume for simplicity that given a pair of nodes i and j that the shortest path between them is unique. Then the total number of shortest paths is $N(N-1)/2$ and thus scales with N^2. Sometimes that is not a good behavior because often the number of shortest paths going to a given link ij scales as N and in the limit $N \to \infty$ betweenness would disappear.

1.11 Link Salience

This, and for other reasons, make it more plausible to use link salience. This is a very simple concept. Given a certain root node i one computes the shortest path tree $T(i)$ to the rest of the network. In a way, this shortest path tree is the union of most effective routes to go from i to

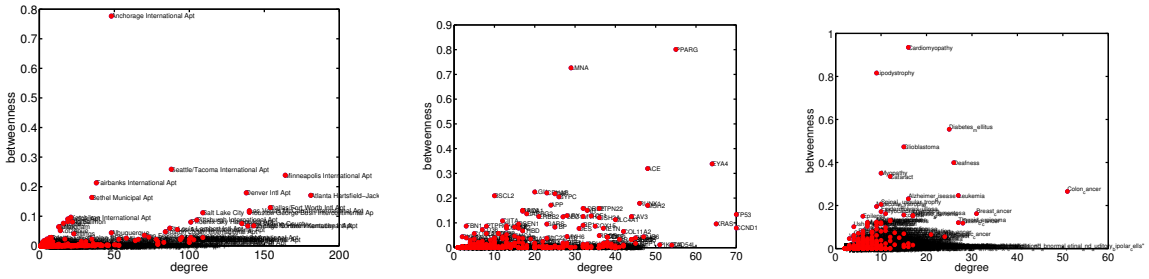


Figure 1.23: Correlation of betweenness and degree centrality in the US air transportation network (left) the human genetic network (center) and the disease network (right).

any other node of the network. We can represent $T(i)$ as a matrix with elements $T_{jk}(i)$ such that $T_{jk}(i) = 1$ is the link jk is in the shortest path tree of node i and zero otherwise. The salience of link jk is then defined as the average accross all shortest path trees

$$S_{jk} = \frac{1}{N}\sum_i T_{jk}(i)$$

This means the salience can be represented as and $N \times N$ matrix $\mathbf{S}$. If $S_{jk} \approx 1$ that means that link jk is important for all nodes, the nodes agree that this link is essential to get anywhere in the network. If $S_{jk} \approx 0$ that means this link essentially does not matter for all nodes i.

1.12 Correlations of centrality measures

We have seen that many of the different centrality measures correlate in some of the networks. For example the top ten table above pretty much contain the same set of nodes in the air transportation network as well as in the gene network. This is not surprising. But this can give interesting insights in case where deviations occur from the overall trend. Let's look at the two networks again to see how betweenness and degree correlate. This is a pair of centrality measures that in the three example network shown in Fig. 1.23. Similar correlation plots for the same networks are shown in Fig.

1.13 Groups

So far we've been focusing on individual nodes or individual links and methods on how to characterize them by centrality measures. Typically there's more structure in networks apart form just single element statistics, i.e. higher level structure that involves more than just single elements. The next level of complexity is about groups of nodes that are connected somewhat differently that the "typical" node in the network.

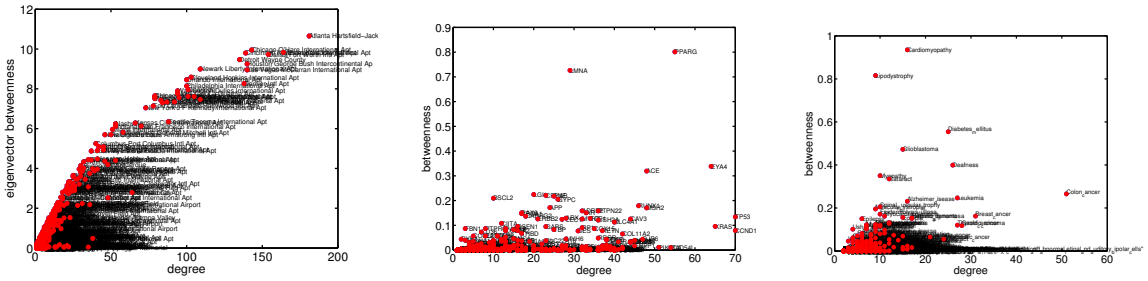


Figure 1.24: Correlation structure of degree centrality and eigenvector centrality.

1.13.1 Cliques

Say, for instance you've got the network depicted in Fig. 1.25, a network of $N = 100$ nodes with a mean degree of $k_0 = 3$. This is a network not very densely connected ($\rho \approx k_0/N$=3%). Just looking at the adjacency matrix or a random graph layout doesn't show much structure. Nevertheless the network contains a subgroup of nodes that are all connected to one another. This is called a clique of size k if k is the number of nodes involved. Mathematically a $k-$clique is thus a set of k nodes such that any given pair of nodes in the clique is connected. Cliques are difficult to find numerically.

1.13.2 k-cores

The concept of cliques is related to k-cores in network. A very useful idea for network layout. k-cores are based on node degree. A k-core is a *maximal* set of nodes such that all nodes in the set are connected to *at least* k nodes in the set. Mathematically we say

$$\Omega_k = \max\{i | i \in V, \sum_{j \in \Omega_k} A_{ij} \geq k\}$$

So let's say I've got 10 nodes in a 5-core, that means each of the 10 nodes must be connected to at least 5 others in the set. There are some interesting things about such k-cores:

- multiple $k-$cores can exist in a network.
- if this is the case, they cannot be connected, because otherwise the union would be a $k-$core, and we are only looking for maximal sets.
- they are very easy to compute numerically. Say you want to compute a $k-$core. First, no nodes with degree less than k can be in it. So
  - you start with the entire network and remove all the nodes with degree $k' < k$
  - then you remove all the dangling links of the nodes that are remaining
  - then you have a remainder network

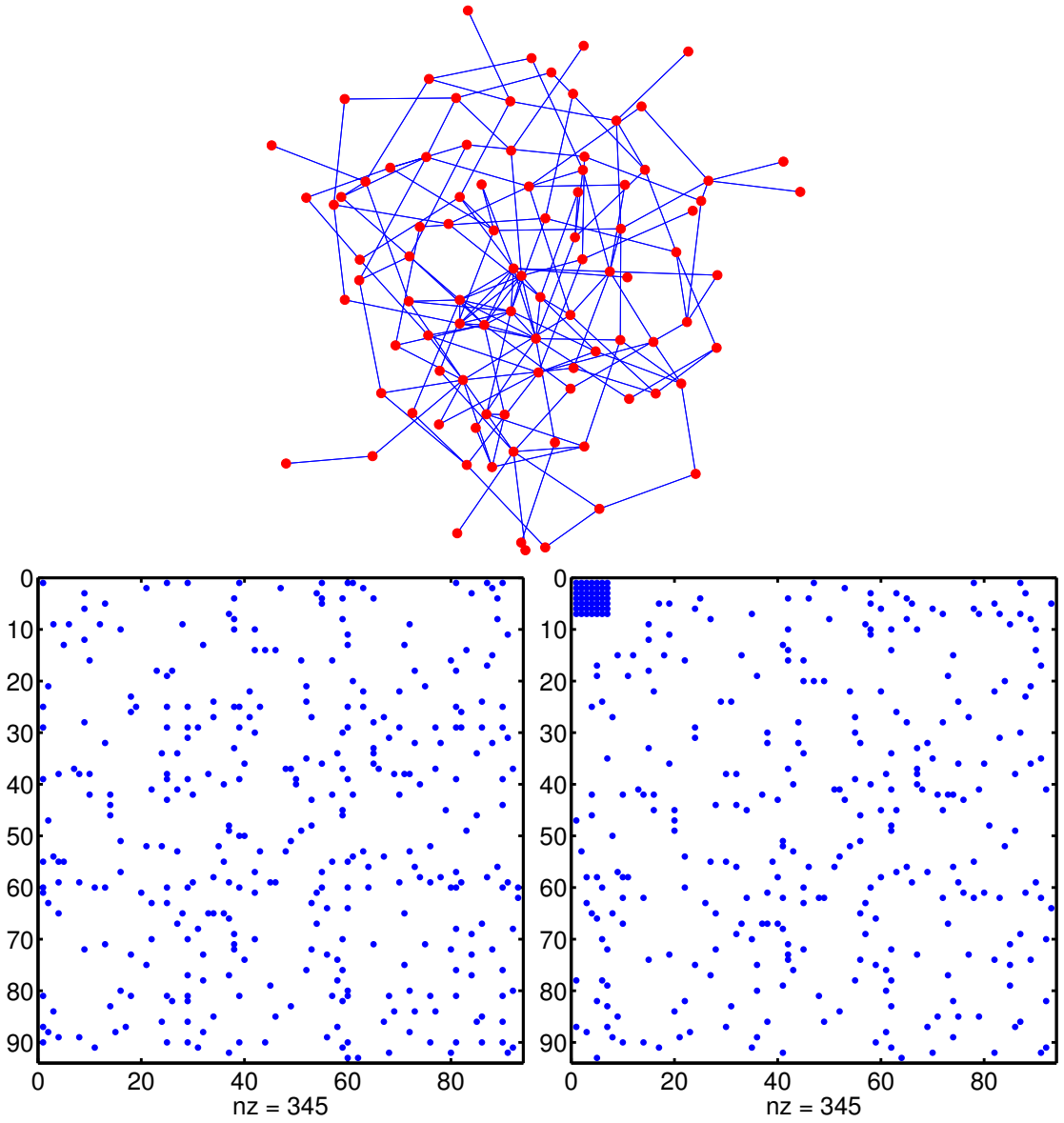


Figure 1.25: Top: A random network with mean degree $k_0 = 3$. Bottom left: The adjacency matrix of the corresponding graph. Bottom right. The same adj. matrix, with nodes labeled differently. We see the network has a subgroup of nodes 7 that are all connected to one another. This group is called a clique. Cliques have different size.

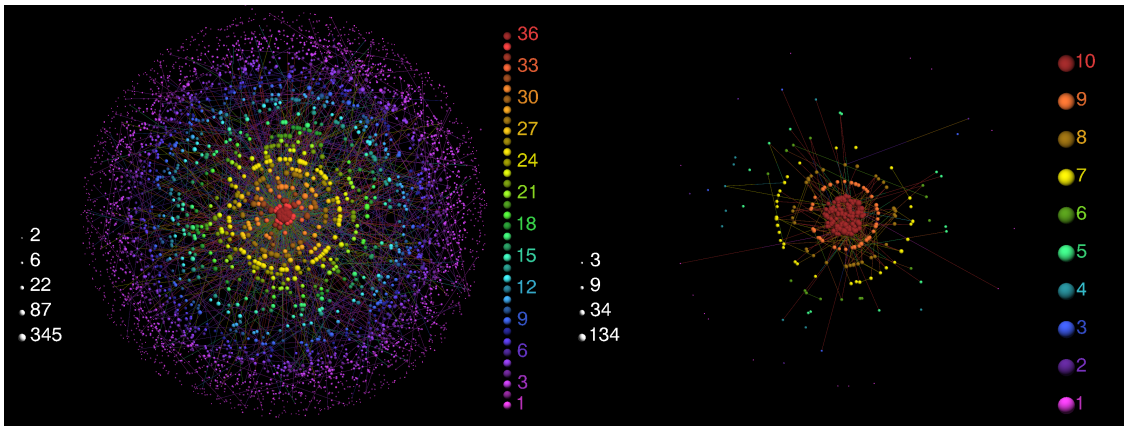


Figure 1.26: k-shell representation of the worldwide air transportation network and the neural network of the organism *C. Elegans*.

– then you go back to step one until you are left with the k-core or a set of k-cores.

You can now do this for all the values of k and compute k shells by successivly plotting the large k cores in the middle and the moving to the periphery. This produces nice layout, see Fig. 1.26.

1.13.3 Transitivity

The basic idea behind transitivity is how reliably we can say that if nodes i and j and nodes i and k are connected whether j and k are connected as well. If that is the case then we have a triangle. In general if $A_{ij} = 1$ and $A_{ik} = 1$ then the nodes i, j, k form a *triplet*. There are only two types of triplets: triangles and non-triangles. Triangles contain 6 paths of length 3 for example $1231, 1321, 2312, 2132, 3123, 3213$ whereas non-triangles, regular triplets, contain two paths of length 2, if e.g. node 2 is in the center of the triplet it's $123, 321$. For the triangles we have $A_{jk} = 1$ for regular triplets $A_{jk} = 0$. Particularly in social networks a large fraction of triplets are triangles, which means if A is friends with B and C then with a high probability B and C are also friends. So one way of measuring the strength of transitivity of an undirected unweighted network is by the fraction of triangles with respect to the entire set of triplets.

$$C = \frac{3 \times \#\text{triangles}}{\#\text{triples}}$$

This is called the clustering coefficient. The reason why we have a factor of three is: each triangle,say i, j, k counts for three triplets, with each of the nodes in the middle. Here we assume that a path i, j, k and k, j, i are the same. if we count these paths not as one triplet, but as two, we have to multplit by 6. So we also have:

$$C = \frac{6 \times \#\text{triangles}}{\#\text{paths of length 2}}. \tag{1.10}$$

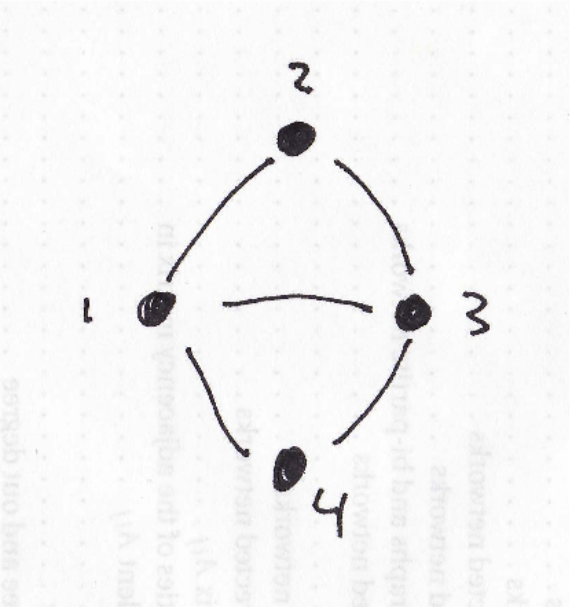


Figure 1.27: Global clustering coefficient C in a network of four nodes. This network has 2 triangles and 16 triplets altogether, so $C = 3/4 = 0.75$.

With these normalizations we have $C = 1$ if all triplets in the network are triangles and $C = 0$ if the network is a tree. Recall that for the number of triangles and the number of paths we only need powers of the adjacency matrix:

$$n_\Delta = \frac{1}{6}\mathrm{Tr}\mathbf{A}^3$$

and the number of paths of length 2 between two given nodes is given by

$$n_2(i,j) = \sum_k A_{ik}A_{kj} = \left(\mathbf{A}^2\right)_{ij}$$

So the total amount of paths of length 2 is given my

$$n_2 = ||\mathbf{A}^2|| - \mathrm{Tr}\mathbf{A}^2$$

where $||\cdot||$ means summing over all matrix elements. Note that we subtract the number of paths of length two that come back to a node, like i, j, i on the same link. This is given by the second term. So all in all we obtain:

$$C = \frac{\mathrm{Tr}\mathbf{A}^3}{||\mathbf{A}^2|| - \mathrm{Tr}\mathbf{A}^2}$$

What does this number mean though? If $C = 1$ that means with certainty the whole network is transitive, e.g. I can infer a connection between node j and k if $A_{ij} = A_{ik} = 1$. If the clustering coefficient is say 0.18 then the likelyhood is only 18%. We can check this in the simple network

Network	N	$\langle k \rangle$	C	C_{ER}	$\langle c \rangle$
US Air Transportation	702	11.81	0.34	0.0170	0.6238
Genetic Network	903	14.97	0.85	0.0165	0.8530
Disease Network	516	5.60	0.67	0.0124	0.6358
C. Elegans Neural	297	14.46	0.18	0.0491	0.2924
Netscientist Collab.	379	4.82	0.43	0.0153	0.7412

Table 1.9: Clustering Coefficient in various networks. C is the clustering coefficient, C_{ER} the clustering coefficient of an random ER network with identical size and link density. and $\langle c \rangle$ is the mean local clustering coefficient.

shown in Fig. 1.27. The total number of triangles is 2 thus the numerator in Eq. (1.10) is 12. The total number of paths of length 2 (non including back/forth paths) are

$$123, 143, 134, 132, 321, 312, 341, 314, 214, 234, 432, 413, 231, 213, 413, 431$$

that is 16. So $C = 3/4 = 0.75$ in this case. Now let's look, again at a planar graph such as the one depicted in Fig. 1.10. When we conmpute the clustering coefficient we get for this particular example $C = 0.3857$. This means, given a node and two neighbors, their likelyhood of being connected is roughly 40%. Can we understand this? Thie planar triangular graph is almost like a triangular grid in which each node has 6 neighbors. IF we pick a node i and one of it's neighbors, then in two out of the 5 remaining neighbors will also be connected to the first one. Thus the probability of picking a triangle is roughly $2/5 = 0.4$ consistent with the numerical measurement of the random triangular lattice. Let's now look at a couple of real networks. Table 1.9 lists the clustering coefficient C for a couple of networks that we discussed.

When comparing these networks the clustering coefficient is useful. But given a single network, what does a value of say $C = 0.45$ mean? Sometimes some insight can be given by comparing a real network with a random network (which we will discuss in detail later) that has the same size and link density. One such network is the Erdos-Renyi network in which one can specify the average degree $\langle k \rangle$ and the number of nodes. Essentially the probability of a link of exisiting is given by $p = \langle k \rangle / (N-1)$.

So if I have a node and two of it's neighbors, the probability of them being connected is entirely random which means it's p. So for instance in the genetic network above this would emply that $C_{ER} = 0.0166$ which is also confirmed by the values shown in Tab. 1.9. That means, the clustering in the real networks above are much higher than expected by chance.

1.13.4 Local Clustering Coefficient

The definition of transitivity as above is a global network property. Often, it is of value to address clustering on a local level. Say we have a node i with degree k_i that means that node has k_i neighbors. Clearly, the maximum number of connections among the neighbors is

$$n_i^{\max} = \frac{k_i(k_i-1)}{2}.$$

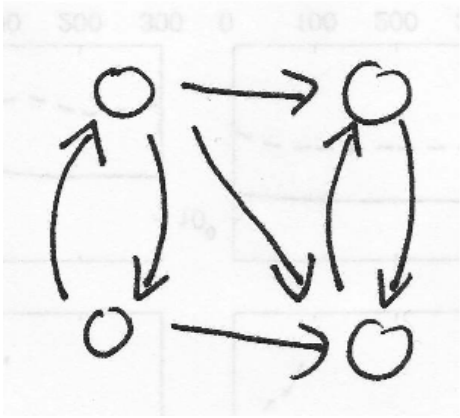


Figure 1.28: A directed network with 7 links, 4 of which are reciprocal yielding a reciprocity of the network of $R = 4/7$.

In this case, we would say that the neighborhood of node i is maximally clustered. A good measure for the local clustering coefficient is thus

$$c_i = \frac{2n_i}{k_i(k_i-1)}$$

where n_i is the actual number of connections among the neighbors of i. Let's look at the simple graph of Fig. 1.27 again. Here we have

$$c_1 = 2/3, c_2 = 1, c_3 = 2/3, c_4 = 1$$

The average local clustering coefficient is also a measure for the clustering of the entire networks

$$\langle c \rangle = \frac{1}{N}\sum_i c_i = \frac{2}{N}\sum_i \frac{n_i}{k_i(k_i-1)}.$$

In the above example this is

$$\langle c \rangle = \frac{1}{4}\left(2 \times \frac{2}{3} + 2\right) = \frac{5}{6} = 0.83$$

Note that this is different from the global clustering coefficient. Usually however, the average local clustering coefficient is used in applications. Both can be quite different, in particular in strongly heterogeneous networks with strong degree variability.

1.13.5 Reciprocity

So far we've been focusing on local structure of undirected and unweighted networks. We will not be discussing clustering in weighted networks although generalizations of C and c_i exist for weighted networks. The idea of local structure in directed networks can be addressed by

reciprocity. This means, if a have a directed link i to j, how likely is it to have a link in the direction j to i. So, for example in Fig. 1.28 we have 7 links and 4 of those are reciprocal. So we would expect the reciprocity to be $R = 4/7$ in this network. In fact we define the reciprocity as

$$R = \frac{\mathrm{Tr}\mathbf{A}^2}{||\mathbf{A}||}.$$

The numerator is the number of self-loops of length 2 that is exactly the number of reciprocal links. and

$$||\mathbf{A}|| = \sum_{ij} A_{ij}$$

is the total number of links in a directed network. Note that a symmetric directed network has $R = 1$ because every link comes in pairs. From which we also learn that

$$\mathrm{Tr}\mathbf{A}^2 = ||\mathbf{A}||$$

for symmetric adjacency matrices. Let's look at our Floria foodweb. We have $N = 122$ nodes. we find $R = 0.0045$ which means that only with a probability of 0.45% a link is reciprocal. That is logical since species tend to not prey reciprocally. But what is the chance of a random network with the same connectivity as the foodweb to have a reciprocal link? The foodweb has 1767 links. this means a density of about 11.87%. Generating a random network with the same density gives a typical reciprocity of that, so by chance this should be 12% roughly. Foodwebs are a factor of 20 below.

1.13.6 Motifs

It turns out that many real world network exhibit small connectivity patterns that appear more than expected by chance that are not just triangles and reciprocal connections. Particularly in protein-protein interaction networks certain structures are more prominent than others. These structures are called motifs. Fig. 1.29 shows all the possible motifs that involve three nodes. Clearly in directed network more motives can be generated with fewer nodes. Some of the motifs have names, like feedforward loop, or three chain or bi-fan. The point is, that some of these small little network motifs occur with a much higher probability in networks than expected by change.

The way this is determined in a real situation is illustrated in Fig. 1.30 for a particular motif, the feedforward loop. First one measures the number of occurances of the motif in the real network. Then one randomizes the network by cutting each link in the middle. This way, if L is the number of links in the network, one gets L arrowheads and L arrow feet. From each of these set of "half"-links we form a random set of links by connecting the links. This way the degrees of the nodes of the original network do not change but their connectivity structure is lost. If in the resulting randomized networks we find substantially less motifs of the type we are investigating, it seems plausible to assume that the funtionality of the original network has something to do with these motifs.

Fig. 1.31 shows a table of data obtained from various network.

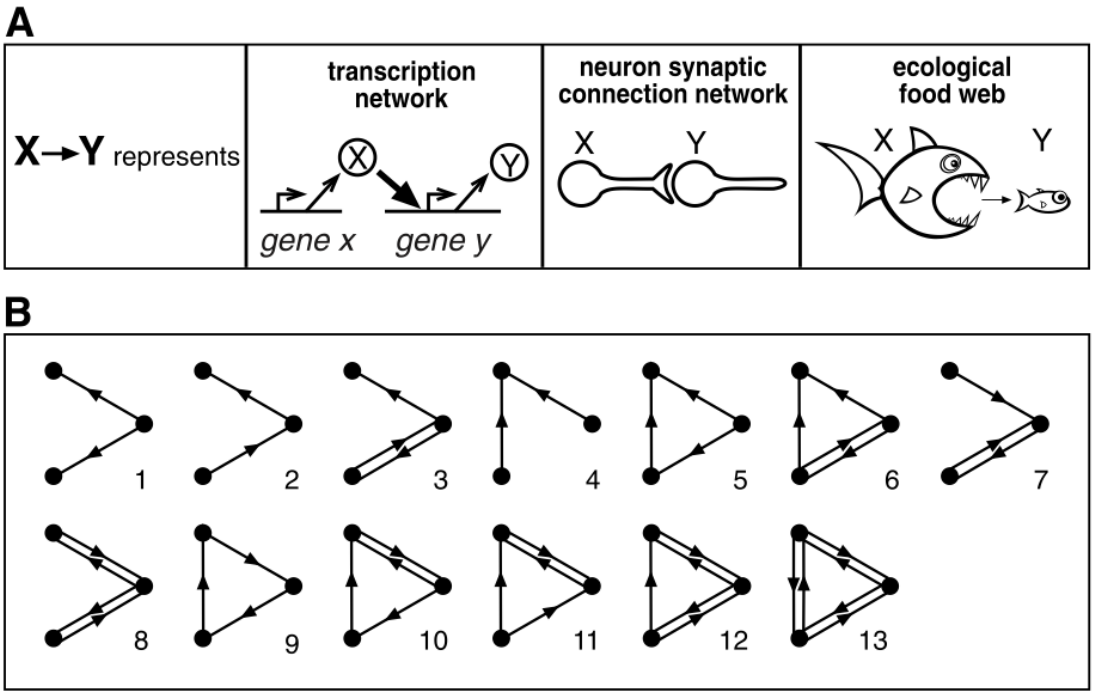


Figure 1.29: Motifs involving three nodes in directed networks. The link direction has a different meaning in different networks. In gene regulation the arrow represents regulatory influence of one gene onto another. In neural networks it represents a synaptic connection and in foodwebs it represents a predator prey relationship.

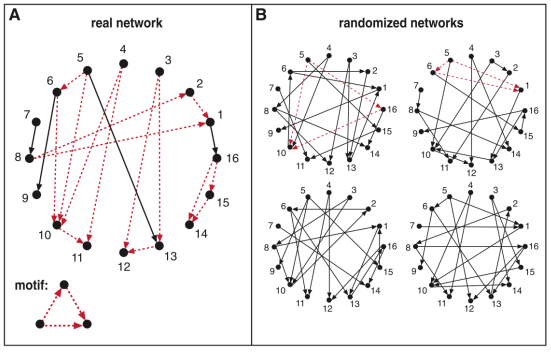


Figure 1.30: Motif measurements. Left: The network represents a real network that has a number of feed forward motifs. Randomizing the links in the network one can determine whether this motif occurs more frequently than expected by chance.

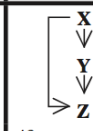
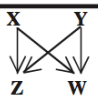
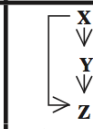
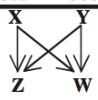
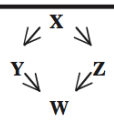
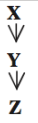
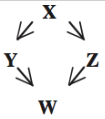
Network	Nodes	Edges	N_{real}	$N_{rand} \pm SD$	Z score	N_{real}	$N_{rand} \pm SD$	Z score	N_{real}	$N_{rand} \pm SD$	Z score
Gene regulation (transcription)					**Feed-forward loop**			**Bi-fan**			
E. coli	424	519	40	7 ± 3	10	203	47 ± 12	13			
*S. cerevisiae**	685	1,052	70	11 ± 4	14	1812	300 ± 40	41			
Neurons					**Feed-forward loop**			**Bi-fan**			**Bi-parallel**
C. elegans†	252	509	125	90 ± 10	3.7	127	55 ± 13	5.3	227	35 ± 10	20
Food webs					**Three chain**			**Bi-parallel**			
Little Rock	92	984	3219	3120 ± 50	2.1	7295	2220 ± 210	25			
Ythan	83	391	1182	1020 ± 20	7.2	1357	230 ± 50	23			
St. Martin	42	205	469	450 ± 10	NS	382	130 ± 20	12			
Chesapeake	31	67	80	82 ± 4	NS	26	5 ± 2	8			
Coachella	29	243	279	235 ± 12	3.6	181	80 ± 20	5			
Skipwith	25	189	184	150 ± 7	5.5	397	80 ± 25	13			
B. Brook	25	104	181	130 ± 7	7.4	267	30 ± 7	32			

Figure 1.31: Motifs in different types of networks.

1.13.7 Structural Equivalence

Another way to detect similarities or structure in networks is by comparing how a pair of nodes connect to the network. Say we compare node i and j. The way they connect to the rest of the network is given by A_{ik} and A_{jk} with $k = 1, ..., N$. We can interpret the rows i and j as two vectors $\mathbf{a}_i$ and $\mathbf{a}_j$ with

$$(a_i)_k = A_{ik} \quad \text{and} \quad (a_j)_k = A_{jk}$$

If the two nodes i and j connect to a large set of common other nodes the dot product of $\mathbf{a}_i$ and $\mathbf{a}_j$ is going to be large and when the share now neighbots it is zero. So the number of common connections is given by the dot product or the overlap

$$n_{ij} = \mathbf{a}_i \cdot \mathbf{a}_j$$

So we can define structural equivalence of two nodes as being

$$\sigma_{ij} \propto n_{ij}$$

However we need to take into account that not all vectors have the same length i.e. not all nodes have the same degree. For instance if the two nodes have both a degree of 5 and connect to the same other nodes we should have σ_{ij} just as large as when both nodes had degree 50. Thus it is

a good idea to normalize the vectors to unit length so

$$\sigma_{ij} = \frac{\mathbf{a}_i \cdot \mathbf{a}_j}{|\mathbf{a}_i||\mathbf{a}_j|} = \frac{\sum_k A_{ik}A_{kj}}{\sqrt{\sum_k A_{jk}^2}\sqrt{\sum_k A_{ik}^2}}$$

If we are dealing with unweighted networks we have $A_{jk}^2 = A_{jk}$ and likewise of i so then

$$\sigma_{ij} = \frac{\left(\mathbf{A}^2\right)_{ij}}{\sqrt{k_i k_j}}$$

the normalization is the geometric mean of the degree.

1.13.7.1 Pearson Correlation

Let's assume we start with the number of connections two nodes share

$$n_{ij} = \sum_k A_{ik}A_{kj}$$

Now, even in a totally random network, two arbitrary nodes are going to have $n_{ij} > 0$, just by chance. So we could approach the problem by asking how much more similar are two nodes than they would be by chance. Let's assume the nodes have degree k_i and k_j. Let's assume i choses one of the other $N-2$ neighbors by chance. The probability that this node is also chosen by j is $k_j/(N-2) \approx k_j/N$. Now i picks another neighbor randomly. Again, the prob. that j has also picken that one is k_j/N. So the total expected $\langle n_{ij} \rangle = k_i k_j/N$ which we should subtract from the above equation so

$$\sigma_{ij} = \sum_k A_{ik}A_{kj} - k_i k_j/N$$

In a totally random network we would have $\langle \sigma_{ij} \rangle = 0$. Now

$$\begin{aligned}
\sigma_{ij} &= \sum_k A_{ik}A_{kj} - k_i k_j/N \\
&= \sum_k A_{ik}A_{kj} - \frac{1}{N}\sum_k A_{ik}\sum_k A_{jk} \\
&= \sum_k A_{ik}A_{kj} - N\left(\frac{1}{N}\sum_k A_{ik}\frac{1}{N}\sum_k A_{jk}\right) \\
&= \sum_k A_{ik}A_{kj} - N\langle A_i \rangle \langle A_j \rangle
\end{aligned}$$

where $\langle A_i \rangle = k_i/N$ is average entry in row i. Thus

$$\begin{aligned}
\sigma_{ij} &= \sum_k \left[A_{ik}A_{kj} - \langle A_i \rangle \langle A_j \rangle\right] \\
&= \sum_k \left(A_{ik} - \langle A_i \rangle\right)\left(A_{jk} - \langle A_j \rangle\right)
\end{aligned}$$

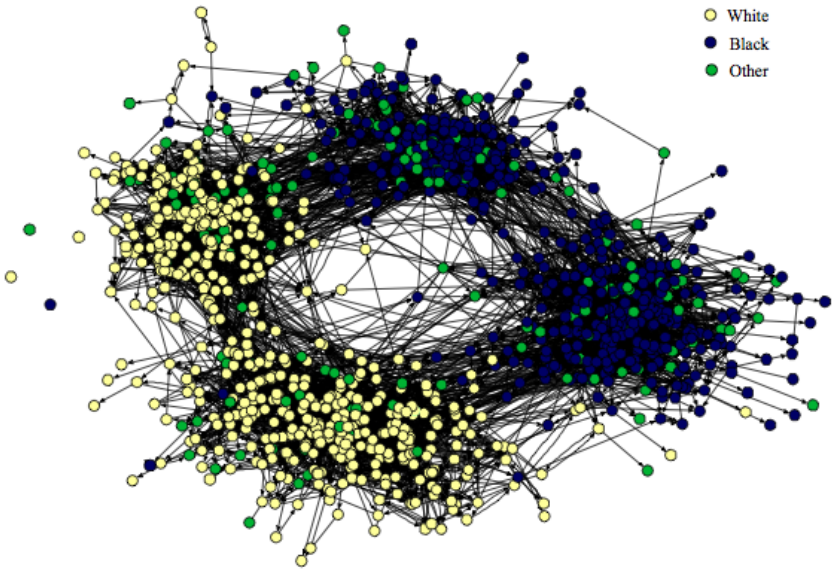


Figure 1.32: Relationships between individuals of different races.

This is the covariance of the two rows corresponding to nodes i and j. This can also be normalized by the variances

$$\sigma_i = \sqrt{\sum_k (A_{ik} - \langle A_i \rangle)^2}$$

so

$$r_{ij} = \frac{\sum_k (A_{ik} - \langle A_i \rangle)\left(A_{jk} - \langle A_j \rangle\right)}{\sqrt{\sum_k (A_{ik} - \langle A_i \rangle)^2}\sqrt{\sum_k (A_{jk} - \langle A_j \rangle)^2}}$$

This correlation coefficient is another measure of how similar the connectivity structure of nodes i and j is.

1.14 Homophily and Assortative Mixing

This idea is realted to what was discussed previously when we discussed clustering and the idea that if node A is connected to B and C then with a prob. higher than chance B and C are also connected. This idea can be generalized in the following way. Often we have, in addition to their linkage, more information about the nodes. For instance, if the network resembles a system of friendships among students we can invistigate how properties of individual nodes such as race, sex, age, nationality determine a networks connectivity. A typical example of assortative mixing is shown in Fig. 1.32 which shows that inter-racial couples are less frequent than inter racial

couples. That means, if I have a given link in the network, the probability that the two nodes it connects are of the same group is higher than chance.

Say I have a network consisting of m different groups with N_m nodes in each group such that

$$N = \sum_m N_m.$$

The relative size of the groups is

$$c_m = N_m / N.$$

If I were to construct a link randomly, the probability of this connecting groups m and m' is given by there relative size

$$p_0(m, m') = c_m c_{m'}.$$

and that a link would be within a group would be

$$p_0(m, m) = c_m^2$$

If we find that in a real network the probability the actual fraction of links within a group m larger, i.e.

$$p(m, m) > p_0(m, m)$$

this is called assortative mixing. If

$$p(m, m) < p_0(m, m)$$

we speak of dissortative mixing.

1.14.1 Mixing by groups

Mathematically we can describe the fact that each node is in a single group by a mapping from the vertices V into the set of groups G. Let's say we've got N nodes and M groups, and each node belongs to exactly one group. We can for an $N \times M$ matrix $\mathbf{G}$ such that element $G_{i\alpha} = 1$ if node i belongs to group α. Note that

$$N_\alpha = \sum_i G_{i\alpha}$$

is the number of nodes in group α. Also, because every node is in exactly one group we have

$$\sum_\alpha G_{i\alpha} = 1.$$

We can use the matrix G to determine whether two nodes i, j are in the same group. Using

$$D_{ij} = \sum_\alpha G_{i\alpha} G_{j\alpha}$$

we have $D_{ij} = 1$ if nodes i and j are in the same group (not if $i = j$). We could also write in matrix notation

$$\mathbf{D} = \mathbf{G}^t \mathbf{G}.$$

The matrix **D** is a block diagonal matrix, with blocks corresponding to the different groups. Let's now count the number of links internal to groups:

$$\begin{aligned} L_{\text{internal}} &= \sum_{ij} A_{ij} D_{ij} \\ &= \sum_{\alpha} \sum_{ij} G_{i\alpha} A_{ij} G_{j\alpha} \\ &= \sum_{\alpha} \left(G^t A G \right)_{\alpha\alpha} \\ &= \operatorname{Tr} E \end{aligned}$$

where

$$E = G^t A G.$$

The elements of matrix E are the number of links between groups

$$E_{\alpha\beta} = \sum_{ij} G_{i\alpha} A_{ij} G_{j\beta}.$$

In general we are interesting in fractions of links. The total number of links in the network is given by

$$L = \|\mathbf{A}\| = \sum_{ij} A_{ij}$$

Say we measure a fraction by

$$l_{\text{internal}} = \frac{\text{Tr}\mathbf{E}}{\|\mathbf{A}\|} = 0.75\%$$

What does that mean? What should we compare this to? If we want to show that there are significantly more links between individuals of the same group then we compare to a random scenario in which we assume that an individuals links is placed at random. What is the number of links that exit a group α? that is given by

$$\begin{aligned} L_{\alpha} &= \sum_{\beta} E_{\alpha\beta} = \sum_{\beta} \sum_{ij} G_{i\alpha} A_{ij} G_{j\beta} \\ &= \sum_{ij} G_{i\alpha} A_{ij} \sum_{\beta} G_{j\beta} \\ &= \sum_{i} G_{i\alpha} \sum_{j} A_{ij} \\ &= \sum_{i} k_i G_{i\alpha} \end{aligned}$$

Let's say we pick one of these links at one end and ask how likely the other link is going to be also in α. That is just the ratio

$$L_{\alpha} / \|\mathbf{A}\|$$

So we expect to have

$$E^0_{\alpha} = L_{\alpha} \times \frac{L_{\alpha}}{\|\mathbf{A}\|}$$

links internal to groups if links are wired at random. So we get

$$\begin{aligned} E^0 &= \frac{1}{\|\mathbf{A}\|}\sum_{\alpha}\sum_{i,j} k_i k_j G_{i\alpha} G_{j\alpha} \\ &= \sum_{ij}\frac{k_i k_j}{\|\mathbf{A}\|}\delta_{ij} \end{aligned}$$

as the expected number of links internal to groups if their links are connected at random. So if the groups significantly determine the way things are linked we define the difference

$$Q = \sum_{ij}\left(A_{ij} - \frac{k_i k_j}{L}\right)\delta_{ij}$$

with

$$\delta_{ij} = \sum_{\alpha} G_{i\alpha} G_{j\alpha}.$$

The quantity Q is called modularity. and the matrix

$$B_{ij} = A_{ij} - \frac{k_i k_j}{L}$$

is known as the modularity matrix. The modularity matrix has the interesting property of

$$\sum_j B_{ij} = 0.$$

If $Q = 0$ then the system is randomly mixed. If on the other hand $Q > 0$ then there's more internal links than expected by chance and the groups mix assortatively. If $Q < 0$ then they mix dissortatively. In practical applications one devided the by the total number of links so

$$Q = \frac{1}{L}\sum_{ij}\left(A_{ij} - \frac{k_i k_j}{L}\right)\delta_{ij}$$

For the network of red and blue nodes we get a modularity of about $Q = 0.38$.

1.14.2 Node and link averages

Let's assume we have some quantity x_i associated with each node in a network. Obviously it sometimes makes sense to compute the average over all nodes

$$\langle x\rangle_N = \frac{1}{N}\sum_i x_i$$

In networks it's sometimes also meaning full to average over the set of links, obviously if for instance the links are weighted we would compute something like the average weight

$$\langle w\rangle_L = \frac{1}{L}\sum_{ij} w_{ij}.$$

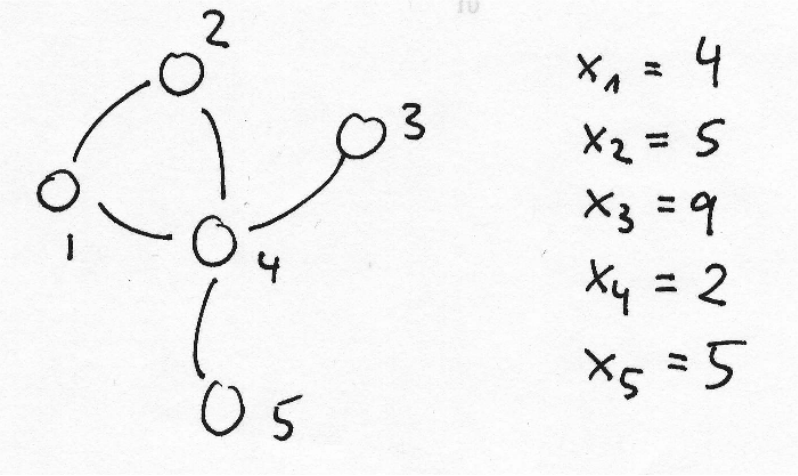


Figure 1.33: A simple network with scalar values attached to each node.

Sometimes we would like to average node properties over the ensemble of links. For example we would like to know what is the average of the quantity x across the set of links. It sounds like this would be the same as averaging over the set of nodes but it is not. Let's look at the example in Fig. 1.33. We have 5 nodes and values $x_1 = 4,5,9,2,5$ associated with each node. Then obviously

$$\langle x \rangle_N = \frac{1}{5}(4+5+9+2+5) = \frac{25}{5} = 5$$

Let's now build the set of links and treat those as an ensemble. In other words let's pick a link at random from the network and pick a random end and ask what's the average. The set of links (x_i, x_j) is given by

$$\begin{array}{lcl} (i,j) & : & (x_i, x_j) \\ (1,2) & : & (4,5) \\ (2,1) & : & (5,4) \\ (2,4) & : & (5,2) \\ (4,2) & : & (2,5) \\ (1,4) & : & (4,2) \\ (4,1) & : & (2,4) \\ (3,4) & : & (9,2) \\ (4,3) & : & (2,9) \\ (4,5) & : & (2,5) \\ (5,4) & : & (5,2) \end{array}$$

The average value of x_i accross the ensemble of links is thus

$$\langle x \rangle_L = \frac{1}{10}(2 \times 4 + 2 \times 5 + 9 + 4 \times 2 + 5) = \frac{40}{10} = 4$$

which is different from the average over nodes. This means that if we pick a node at random the average of x is different from when we pick a link and one of it's connected nodes at random. From the above table we see that

$$\langle x \rangle_L = \frac{1}{L} \sum_i k_i x_i$$

This is intuitively clear because each node i contributes k_i copies of x_i to the set of links. We can also write this as

$$\langle x \rangle_L = \frac{\sum_i k_i x_i}{\sum_i k_i}.$$

An interesting consequence of this is if we let the quantity x_i be the degree of the node k_i. That means that the average degree of a node connected to a randomly chose link is

$$\langle k \rangle_L = \frac{\langle k^2 \rangle_N}{\langle k \rangle_N} \geq \langle k \rangle_N$$

and is always larger than the average computed across the set of nodes.

1.14.3 Mixing by scalar quantities

How can we determine if nodes of propoertiy x_i are likely to be connected to nodes with the same x_i or in a similar range. Again we focus on a link $i \leftrightarrow j$ that connects two nodes i and j. First let's compute the mean value of x_i for all link:

$$\langle x \rangle = \frac{1}{L} \sum_i k_i x_i$$

Let's now look at the covariance of x by summing over all the links in the network

$$\begin{aligned} C(x) &= \frac{1}{L} \sum_{ij} (x_i - \mu) A_{ij} (x_j - \mu) \\ &= \frac{1}{L} \sum_{ij} A_{ij} x_i x_j - \mu^2 \\ &= \frac{1}{L} \sum_{ij} A_{ij} x_i x_j - \frac{1}{L^2} \sum_{ij} k_i k_j x_i x_j \\ &= \frac{1}{L} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{L} \right] x_i x_j \end{aligned}$$

which looks almost like modularity, except for the factor $x_i x_j$ instead of δ_{ij}. This is a covariance. Deviding by the variance

$$V(x) = \frac{1}{L}\sum_i k_i (x_i - \mu)^2$$

gives the assortativity coefficient

$$r = \frac{C(x)}{V(x)} = \frac{\sum_{ij}\left[A_{ij} - k_i k_j/L\right] x_i x_j}{\sum_{ij}\left[k_i\delta_{ij} - k_i k_j/L\right] x_i x_j}.$$

This number is restricted to the range $[-1,1]$. If $r = 1$ the network is perfectly assortative and if $r = -1$ is is perfectly dissortative. if $r = 0$ then the values of x_i at the end of links are uncorrelated.

1.14.4 Mixing by degree

If our quantity of interest is the degree itself the assortatitivy coefficient of degree is

$$r = \frac{C(x)}{V(x)} = \frac{\sum_{ij}\left[A_{ij} - k_i k_j/L\right] k_i k_j}{\sum_{ij}\left[k_i\delta_{ij} - k_i k_j/L\right] k_i k_j}.$$

1.15 Software

For most of the networks that we will look at we will use the software package Gephi. Gephi displays network and can do basic computation, filtering and ranking of nodes and links in network. In order for Gephi to work, it needs to have access to network data that is provided in network files. There are different standards for writing network data into a file.

1.15.1 Comma separated link table (csv)

this is the easies format for a network that Gephi can read. Say we have a network of N nodes and L links. We can store the connectivity in of the network in a table in which each row represents a link from i to j, so each row just contains the comma separated indices of the origin and the destination of the link. Gephi will ask whether this is supposed to be a directed or an undirected link. if the links have weights, these can be given as a third value in each row.

1.15.2 GML

This is a fairly old but widely used standard for encoding graphs. The problem with only providing a link table is that it is difficult to provide information about nodes. For instance labels, locations and other node specific information. The link table is really only about links. the GML (graph mark up language) and other standards solve this problem this way.

The data contains two blocks of information

- Block 1: Information about nodes

- Block 2: information about links

The whole GLM standard is a nested structure of objects. Fig. XX shows a typical example of such a structure. The outmost object is the

- GRAPH: this can hold some information, such as whether the graph is directed or undirected

GRAPH contains a list of

- NODE objects. These contain some information about the nodes. The most important is the ID, that uniquely identifies the node. Additional information is the LABEL and the X,Y, and Z coordinates.

GRAPH also contains a list of

- EDGE objects. These are more complicated, they contain a SOURCE and a TARGET property. The values are the Node IDs. There's also te VALUE tag, that encodes the weight of the link.

1.15.3 Graph Document Format (GDF)

This is very similar and has become for popular, because it's easier to generate files in this format. This format has also two blocks, one corresponding to nodes and one to their links. Each block starts with a definition of what a node and a link is:

- `nodedef> ......`
- `edgedef>......`

All that is required for the node-definition is the NAME, which is also used as an id, and for the link definition node1 and node 2. So For instance this is a simple defintion of a network

```
nodedef> name INT
1
2
3
4
edgedef> node1, node2
1,3
4,4
2,1
3,1
```

The clue about this syntax is that one can provide additional information and in the definition line say what that would be so this here is another example:

```
nodedef> name VARCHAR,age INT,label VARCHAR
```

```
john, 23, idiot
mary,3,newborn
horst,66,guy
inge,66,horsts wife
edgedef> node1, node2
john,mary
horst,horst
horst,inge
horst,john
```

1.16 Algorithms

1.16.1 Recursive Algorithms

Many algorithms useful network analysis are most effectively programmed rescursively. That means, the algorithm generally is defined by a function that calls itself until a particular condition is met. So for instance if we want to compute the power of 2, i.e. 2^k we could do this in a for loop

- `x = 2`
- `for i = 1 to k-1; x=2*x; endfor`

Alternative we could write a function that calls itself recursively

- `function x=power2(k)`
  - `if(k), x=2*power2(k-1); else x=2; end`
- `end`

This function works backwards.

1.16.2 Finding Components in a network

Although we discussed earlier what components of a network are (each node in a component must be reachable by any other node in the component), we have not discussed how to find components in a network given by the adjacency matrix A. We can do this recursively by burning through the network.

1. `label all nodes as belonging to no component`
2. `chose a random initial node`
3. `associate it with component 1`

4. `chose all it's neighbors that have not been associated with a component`
    a) `if there are no such neighbors you are done with component 1`
    b) `for each of the neighbors go back to step 3`
5. `remove all the nodes that belong to component 1 from the network`
6. `go back to 2`

this process burns succesivly through all the components as is illustrated in Fig. XX

1.16.3 Finding Shortest paths

Finding the shortest paths is almost as easy as finding component. First, let's assume that we have a network that consists of only one component, so we can reach any node from any other node. Finding the shortest paths is very intuitive when using Dijkstra's algorithm. It works like this. First we pick a root node from which we would like to compute the shortest paths to all the other nodes. We define a distance D that we associate with each node, which is to measure the distance to the root node. So we fix $D_i = 0$ if i is the root node and $D_j = \infty$ for all the other nodes. Now comes the algorithm

1. Find the neighbors of the root node, call this U_i
2. Increase the temporary distance D to $D+1$;
3. Compare all the neighbor distances D_j of nodes $j \in U_i$ to D (at the first step all the neighbors have $D = \infty$)
4. If $D < D_j$ then set $D_j = D$, and record the parent node i for all the nodes in U_i, If no such nodes exist, exit
5. Treat all the remaining neighbors as new root nodes and go back to step 1

Successively this algorithms goes through the network in shells from neighbors to next neighbors and so forth, but it will only proceed to neighbors that are "farther away". As is shown in Fig. XX.

1.16.4 Randomizing a network

When we talked about assortativity we had to compare real networks to those that we obtain if we randomize their connectivity keeping the degrees of the nodes the same. Often this needs to be done with a computer. Intuitively it's clear that we we cut each link into two ends we get two replicas of the set of nodes each one having the same set of open links as shown in Fig. XX. Graphically we randomize by just connecting links randomly from left to right. In an algorithm this is best done by permutating a link list. Say we have an undirected graph represented by a link list

i	j
i_1	j_1
i_2	j_2
i_3	j_3
...	...

Then all we have to do to randomize the network keeping the degree the same is to shuffle the second column in the link list. The degree of each node stays the same this way because say $i_1 = i_5 = i_7 = 9$ corresponds to node 9 with degree 3, then shuffling the second column is not going to change the degree. The only thing we have to make sure of is that no self-links are generated by shuffling and multiple links between a given pair of nodes. If self- and multiple links are permitted, that's ok. If not then the shuffling can generate a self-loop and multiple links. However, in large sparsely connected networks that usually does not impact that statistics. If one has to make sure that no self- or mutliple links emerge by randomization one needs to be a bit more careful in randomization: One can do this sequentially. Starting with node i and degree k_i one picks from the set of link stubs k_i that belong to different nodes. This avoids self-links and multiple links and one marks these links as taken. then one proceeds to the next node $i+1$ and does the same until all stubs are taken. If the algorithm runs into a dead end (one is only left with stubs belonging to one node), one starts over again. It is not guaranteed that this converges in acceptable times.

2 Statistical Analyis of large scale networks

In this chapter we will be discussion applications of the analysis presented in previous chapters to large scale networks. By large scale networks we mean networks that have thousands of nodes such that a focus on single nodes or links and their role in the entire network is negligible. Rather the information on the structural and topological features is given by the distributions of properties such as the distribution of degree or other centrality measures. Also, some of the interesting properties of complex networks only emerge in large scale networks. Examples of large scale networks are given in Table XX

2.1 The small world phenomenon

On of the most celebrated properties of many real world networks is their small world property. A network is small world if, loosely speaking, the shortest path between two randomly chose nodes is much smaller that one would expect.

2.1.1 The shortest path on a lattice

Let's first consider the behavior that we are used to when wee think of lattices. Let's assume that we have a two dimensional lattice with $N \times N$ nodes that has a linear size L. if we randomly pick two nodes i and j we can compute a shortest paths that connects them. That path is most likely going to be $d < L$. The question is now how the typical path length scales with the size of the system. In the two dimensional lattice we will find that

$$d \sim L$$

which also means that

$$d \sim N^{1/2}$$

because

$$N \sim L^2$$

In fact in a $d-$dimensional lattice we find that the typical shortest path between two nodes scales with the number of nodes

$$d \sim N^{1/d}.$$

Fig. XX illustrates this, again, for the great planar triangular network. The mean diameter is was computed for network sizes ranging from 10 to 1,000 nodes. The networks are shown in the figure, as well as the mean shortest path as a function of N the solid line indicates the scaling

$$d \sim \sqrt{N} \tag{2.1}$$

As expected the actual scaling is consistent with the planar architecture of the network. Let's now compare this to another network. Let's keep the average degree

$$k_0 \approx 6$$

of the random planar graphs and generate a completety random network with the same degree for each of the chose network sizes choses previsouly for the planar graphs. Fig XX shows that the behavior 2.1 is no longer visible in the random network. In fact we will later show that in the random graph we typically find a scaling relation

$$d = \sim \log N$$

that means if we square the number of nodes, we only double the mean shortest distance! We can thus expect in networks that appear to have no metric imprint on the topology to see this logarithmic scaling.

This notion was tested experiementally by one of Milgrams famous experiements in the 1960s in which we wanted to measure how many steps it takes by means of aquaintance links in a social network to go from one randomly chosen person to another.

2.1.2 Milgrams experiment - Six degrees of separation

In its first experiment, Milgram asked randomly selected people in Nebraska to send letters to a distant target individual in Boston, identified only by his name, occupation and rough location. The letters could only be sent to someone whom the current holder knew by first name, and who was presumably closer to the final recipient. Milgram kept track of the paths followed by the letters and of the demographic characteristics of their handlers. Although the common guess was that it might take **hundreds** of these steps for the letters to reach their final destination, Milgram's surprising result was that the number of links needed to reach the target person had an average value of just **six**. More recently, a similar experiment conducted by Dodds on e-mail exchanges successfully reproduced Milgram's experiment, but capitalizing on the globalization of the Internet. The e-mail passing messages, indeed, completed enough chains as to allow for their through statistical characterization. Milgrams experiment then coined the phrase *"six degrees of separation"*. Fig. X show the distribution of actual path lengths in a similar experiement performed in 2003 not using actual mail, but email. The distributions of path lengths is consistent with those measure by Milgram 50 years ago.

2.1.3 How many degrees of separation today?

Milgrams experiment largery focused on the United States in the 60s. The overall population of the US at that time was about 180 million. If the relation

$$d \sim \log N$$

is true for a social network then we have

$$d = A \log N$$

where A is a proportionaly constant that we can estimate with Milgrams experiement.

$$A = \frac{d = 6}{\log N = 27.32} = 0.218$$

So worldwide he have now 7×10^9 people so we would expect the mean degree of separation to be

$$d = 0.218 \times \log 7 \times 10^9 = 0.218 \times 32.704 = 7.1296$$

not a significant increase!

2.1.4 Geocaching Experiment

Here's another, more personal, example: Geocaching.

2.2 Topology and statistics

2.2.1 Basic Statistics

The easiest way to get an overall impression of the structure of a large scale network is by computing averages of the various quantities we discussed so far, e.g. (assuming an undirected and unweighted network):

- mean degree $\langle k \rangle$
- relative size of the largest component $s = N_{gc}/N$
- mean shortest path $\langle l \rangle$ (also known as the diameter of the network)
- the clustering ceofficient C (the fraction of 2 step paths that are triangles)
- the average local clustering coefficient $\langle c \rangle$
- etc.

Typically, these numbers are best understood in comparison to other networks. Table 2.1 provides these quantities for a number of known networks.

Network	N	L	$\langle k \rangle$	s	$\langle l \rangle$	C	$\langle c \rangle$
Film Actors	449,913	25,516,482	113.43	0.980	3.48	0.20	0.78
Physics Coauthorships	52,909	245,300	9.27	0.838	7.57	0.15	0.34
Power Grid	4,941	6,594	2.67	1.0	18.99	0.10	0.08
Internet	10,697	31,992	5.98	1.0	3.31	0.035	0.39
Metabolic Network	765	3,686	9.64	0.996	2.54	0.09	0.67
Protein Interactions	2,115	2,240	2.12	0.689	6.80	0.072	0.071

Table 2.1: Examples of global statistical features of large scale networks.

2.2.2 Statistical Distributions

More insight can be gained by investigating not only mean values but entire distributions of centrality measures for instance. Simply put, we have a long list, a set, of quantities and we measure histograms. So for instance we may have a large network with N nodes and L links we can ask, what's the fraction of nodes n_k that have a degree $k_i = k$. Mathematically what we do is we form the ensemble of degrees

$$\{k_1, ..., k_N\}$$

and the just count all the ones that have a degree k, N_k and then simply compute

$$n_k = N_k / N.$$

And then we plot n_k vs. k. This is particularly easy because the degree is a discrete variable, $k_i \in \mathbb{N}$. Not all measures are discrete, for example the weight w_{ij} of a link may be best considered a rational number, or the eigenvector centrality, or other quantities. Let's take the example of links. Let's say we have L links and again form the set

$$\{w_i, ..., w_L\}$$

where we now label the links with just one index (unlike we usually do with two indices). Now if w_i are non integers, with certainty each weight appears only once in the ensemble. Therefore we cannot ask what's the fraction of links that has a weight w. We rather have to ask what's the number $L(w, \Delta w)$ of links that are in the interval

$$[w, w + \Delta w]$$

or

$$[w - \Delta w/2, w + \Delta w/2].$$

and the fraction $l(w, \Delta w)$ is just defined according to

$$l(w, \Delta w) = \frac{L(w, \Delta w)}{L}$$

We can now cover the whole range of weights by successive intervals

$$[w_k, w_k + \Delta w_k]$$

with

$$\Delta w_k = w_{k+1} - w_k$$

not necessarily the same and $w_{k+1} > w_k$ and $w_0 = 0$. This way we make sure that every weight falls into one interval.

2.2.2.1 Normalization

If we are counting discrete elements such as degree the normalization just reads

$$1 = \sum_{k=0}^{\infty} n_k$$

In the continous case we have

$$1 = \sum_{k=1}^{\infty} l(w_k, \Delta w_k)$$

2.2.2.2 Continuous approximations and densities

The quantity $l(w_k, \Delta w_k)$ is the fraction of links that falls into the range $[w_k, w_k + \Delta w_k]$. We can define the probability density

$$p(w_k) = \frac{l(w_k, \Delta w_k)}{\Delta w_k}$$

The normalization condition is given by

$$1 = \sum_{k=1}^{\infty} p(w_k) \Delta w_k \approx \int_0^{\infty} p(w) dw.$$

Sometimes we assume that the underlying probability density $p(w)$ is continuous but we only have a finite sample and access to the empirical density

$$p(w_k) = \frac{l(w_k, \Delta w_k)}{\Delta w_k}$$

Then we hope that the larger the sample the better the agreement

$$p(w_k) \approx p(w).$$

2.2.3 Degree Distribution

The most prominent and usually the first distribution one investigates in undirected unweighted networks is the degree distribution

$$p_k = \text{probability that a node has degree } k$$

2.2.4 Mean degree

Remember that if we have a set of degrees

$$\{k_1, ..., k_N\}$$

we compute the mean degree as

$$\langle k \rangle = \frac{1}{N} \sum_i k_i$$

Intepreting the relative frequency n_k as the probability p_k that a node as a link we can also of course write

$$\langle k \rangle = \sum_k k\, p_k$$

2.2.5 Higher Moments

While the mean $\langle k \rangle$ informs about a typical value, the variance

$$\mathrm{Var}(k) = \left\langle (k - \langle k \rangle)^2 \right\rangle \geq 0$$

gives a measure of how much a degree is expected to be away from the mean. Since

$$\begin{aligned} \left\langle (k - \langle k \rangle)^2 \right\rangle &= \left\langle k^2 - 2k \left\langle (k - \langle k \rangle)^2 \right\rangle + \langle k \rangle^2 \right\rangle \\ &= \left\langle k^2 \right\rangle - 2 \langle k \rangle^2 + \langle k \rangle^2 \\ &= \left\langle k^2 \right\rangle - \langle k \rangle^2 \geq 0 \end{aligned}$$

the second moment tells us a bit about the variability in the distribution p_k. Note also that because of this the second moment is always larger or equal to the square of the first moment. The second moment can be computed either by

$$\left\langle k^2 \right\rangle = \frac{1}{N} \sum_i k_i^2$$

or using the probability p_k:

$$\left\langle k^2 \right\rangle = \sum_k k^2\, p_k$$

2.2.6 Coefficient of Variation

A useful quantity that to measuring the variability in a quantity is the coefficient of variation. This, unlike the variance has no units and quantifies the relative variability in the data (i.e. in units of the mean)

$$CV(k) = \frac{\sqrt{\left\langle (k - \langle k \rangle)^2 \right\rangle}}{\langle k \rangle}.$$

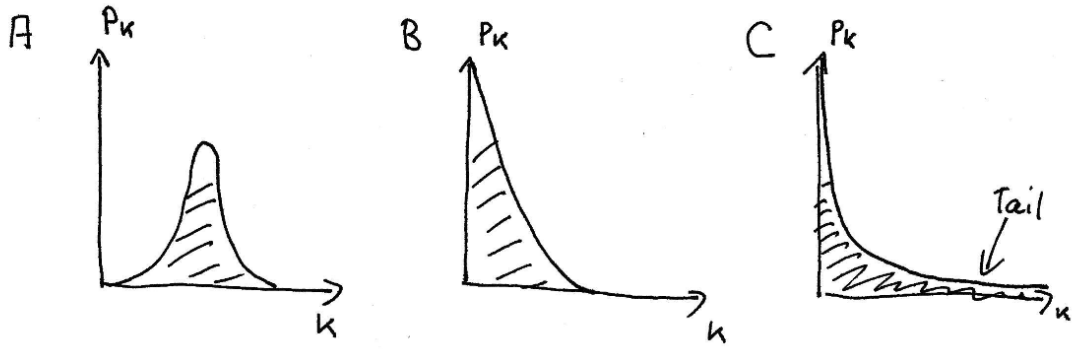


Figure 2.1: Three classes of degree distributions $p(k)$.

2.3 Types of degree distributions

Generally we can classify degree distributions that occur in real systems into three categories. Although finer categorizations exist, broadly speaking there are these types, as illustrated in Fig. 2.1.

1. The variability in degree is small and centered around a typical value
2. The degree distributions decreases as k is increased, the larger the degree the small the likelyhood of occurance. The curve drops off sufficiently fast such that degrees are not observered that are orders of magnitude larger than the expected degree.

   a) if for instance say $p(k)$ behave like

   $$p(k) \sim e^{k^2/\sigma^2}$$

   for large k, then the probability of occurance of a value $k = 10\sigma$ is e^{-100} with is pretty much never.
3. The degree distribution decreases asymptotically but does so weakly, such that a long tail in the distribution exists. An example is a power law

   $$p(k) \sim \frac{1}{k^2}$$

   Again if we were to ask how does the probability decrease if we go to a value of $10k$ that is going to be $1/100$ which is small but a lot bigger than e^{-100}. It turns out as we will see, that this makes a huge difference.

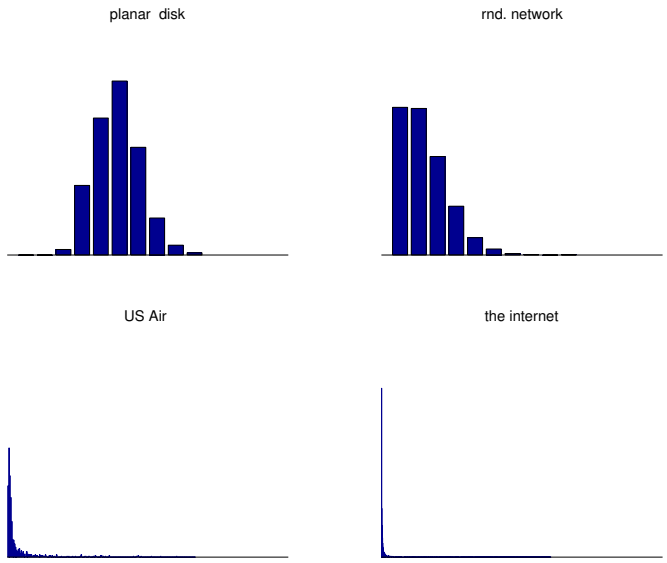


Figure 2.2: $p(k)$ of four different network, the planar network, a random network, the US Air transportation network and the snapshot of the internet.

2.3.1 Examples

Fig. 2.2 show three types of examples of degree distributions. The first one is $p(k)$ for a planar random triangular graph. The second is a random network, the third the US air transportation network and the fourth is a snapshot of the internet. The planar graph has nodes with degree that fluctuate a little around the average.

2.3.2 Treating the degree as a conitnuous variable

Of course the degree is a discrete variable $k = 0, 1, 2, \ldots$. and the distribution is on this discrete sets of values, such that p_k is the probability of a node having a degree k and

$$\sum_{k=0}^{\infty} p_k = 1$$

In many cases the range over which acutal degrees in a large scale network are distributed is very large ranging over many orders of madgnitude such that the spacing between degrees (which is trivially equal to 1) is much less that this range. in this case we can think of the degree distribution as a function over a continuous domain of numbers for example

$$p(k) = p_k \Delta k$$

with $\Delta k = 1$ and

$$\int_0^\infty p(k)\,dk = 1$$

Strickly speaking this is only symbolic because there's not $\Delta k \to 0$ happening. The logic behind this is merely that one can use methods from continuous variable calculus instead of having to deal with sums which is more cumbersome in many cases. The intuition about this is that if we have nodes with a degree that varies between say 100 and 10000 we really do not care whether 452 nodes have degree 1402 or whether 452 nodes have degree 1401. They statistical structure doesn't change. Because of this we will think of $p(k)$ for those networks that have a broad distribution as a continuous quantity.

2.3.2.1 Log-Log plots and log linear plots

One way to distinguish in particular between a fast and a slow decrease for large values of k, for instance if we want to distinguish between a power-law behavior

$$p(k) \sim \frac{1}{k^{1+\gamma}}$$

and for instance an exponential decrease

$$p(k) \sim e^{-\alpha k}$$

is to plot $p(k)$ on a double logarithmic plot. that is plotting $\log p$ vs. $\log k$. If we are dealing with a power-law we get a line

$$\log p = \log A - (1+\gamma)\log k$$

which is a line with negative slope $1+\gamma$. Therefore, we can also measure the exponent γ if the resulting plot is indeed a line or approximately a line. Fig. XX shows the different $p(k)$ on a double logarithmic scale. We see that the two real networks seem to be following a linear shape and we might conjecture that the degree distribution is a power law. We see also that in these plots there is a lot of statistical wiggle for large values of k which makes an estimation of the exponent γ difficult.

2.3.3 Cumulative distributions

Therefore it is sometimes better, not to plot the relative frequency p_k or alternatively speaking the probability density function $p(k)$ but rather the probability $P(k_i \geq k)$ that the degree of a node i is greater than a reference value k. Mathematically it is stragihtforward to compute this probability if we have the pdf (probability density function) $p(k)$. Assuming for a moment that k is a continuous variable then

$$P(k_i \geq k) = \int_k^\infty p(k')\,dk'.$$

so therefore $P(k_i \geq 0) = 1$ and and $P(k_i \geq \infty) = 0$.

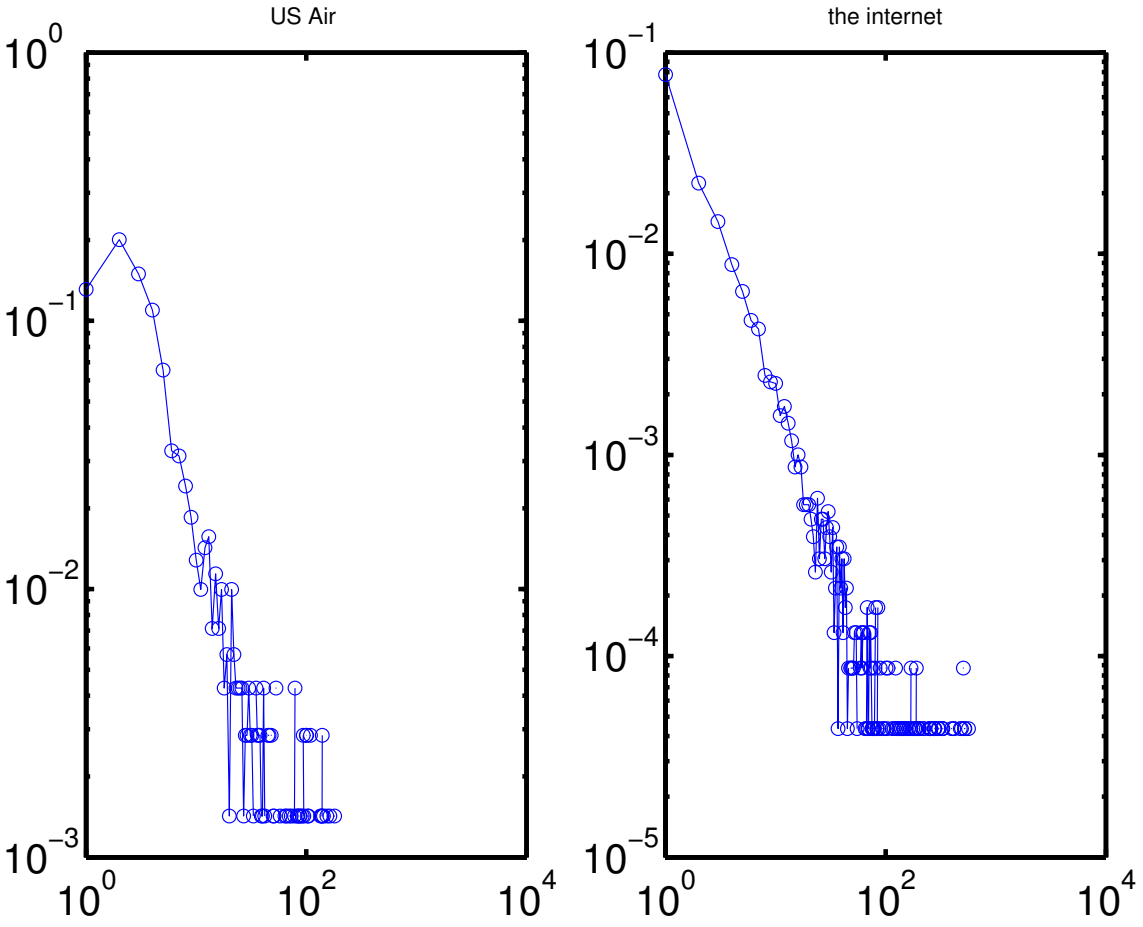


Figure 2.3: Degree distributions on a log-log scale can reveal a power law structure.

Let's now be more precise and account for the fact that k is discrete and we have a relative frequency p_k. Then the probability that the degree of a node is larger than k is given by

$$P_k = \sum_{l=k}^{\infty} p_l$$

One way of computing this easily from data is the following. Let's assume we have a set of nodes indexed i and each has a degree k_i. We can sort the array of degrees such that

$$k_1 \leq k_2 \leq ... \leq k_i \leq .... \leq k_N.$$

If we pick a reference value k_{m+1}. It will be somewhere in this sequence

$$k_1 \leq k_2 \leq ... \leq ... \leq k_m \leq k_{m+1} .... \leq k_N.$$

Then m degrees in the sequence are smaller or equal to k_m. Thus, the probability that a nodes degree is smaller than k is approximately given by

$$Q_m = \frac{m}{N}$$

and the probability that a node has a degree higher than k is given by

$$P_m = 1 - \frac{m}{N}$$

Thus plotting vector

$$k_m = \{k_1, k_2, ..., k_N\}$$

against the vector

$$P_m = 1 - \frac{1}{N}\{1, 2, 3, 4...., N\}$$

that is the pair

$$(k_m, P_m)$$

will approximate the probability that a node will have a degree larger than k_m. This is easily computed by just sorting the degrees of all nodes and plotting them against the array

$$1 - 1/N, 1 - 2/N, ..., 1 - m/N, ...0$$

Fig. 2.4 shows this estimated cumulative probability for the different networks. The nice thing about this procedure is that we make use of every data point we have. What can we learn from this procedure? We can also see if $p(k)$ is a power law. If for instance

$$p(k) = A/k^{(1+\gamma)} \quad \text{for} \quad k > k_0$$

then

$$P(k > k_0) = A \int_{k_0}^{\infty} \frac{dk}{k^{1+\gamma}} \sim \frac{1}{k^{\gamma}}$$

which means that this is also a power law and should be a straght line on a log log scale. Thus we see that the internet has a power-law degree distribution. The ship network, which we thought might have actually curves down, so is probably not a true power law.

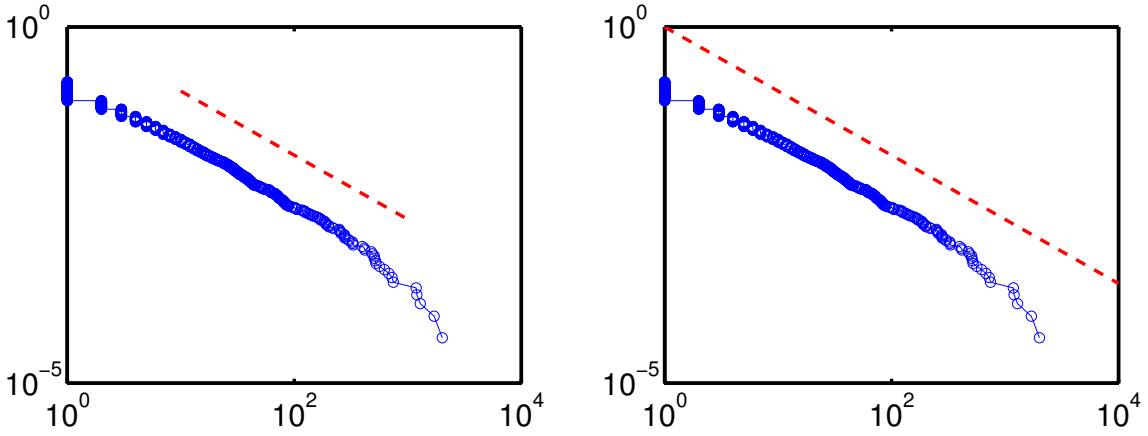


Figure 2.4: The cumulative degree distribuion $P(k_i > k)$ for the US air transportation network. Both networks show a range on which the function is a power law with exponents $\mu \approx 1$ indicated by the red dashed line.

2.3.4 Comparing different degree distributions

Sometimes it is essential to compare the degree distributions of different networks to investigate whether they have the same shape or not. In principle, we can just plot them in one figure. However, since the mean degree is very different in different networks, the curves for $p(k)$ or $P(k)$ even though they may have a similar shape, do not coincide. Therefore one often rescale the degree k by the median or mean degree and plots the distribution of those rescaled quantities.

2.4 Scale free networks

So what's all the fuss about power-law degree distributions. What are the implications? They are profound, it turns out. To understand this, we have to understand power-law in statistical distributions in general. Let's assume that we have a probability distribution for a random variable $X > x_0 > 0$ and let's say we call this probability distribution $p(x)$. One of the most fundamental properties of probability is the law of large number. a consequence of which is the following. Say we draw a sequence of random numbers from $p(x)$ and let's call this sequence

$$S_n = \{X_1, ..., X_n\}$$

Like rolling a die n times. Then we can compute the average

$$\langle X \rangle_n = \frac{1}{n} \sum_i X_i$$

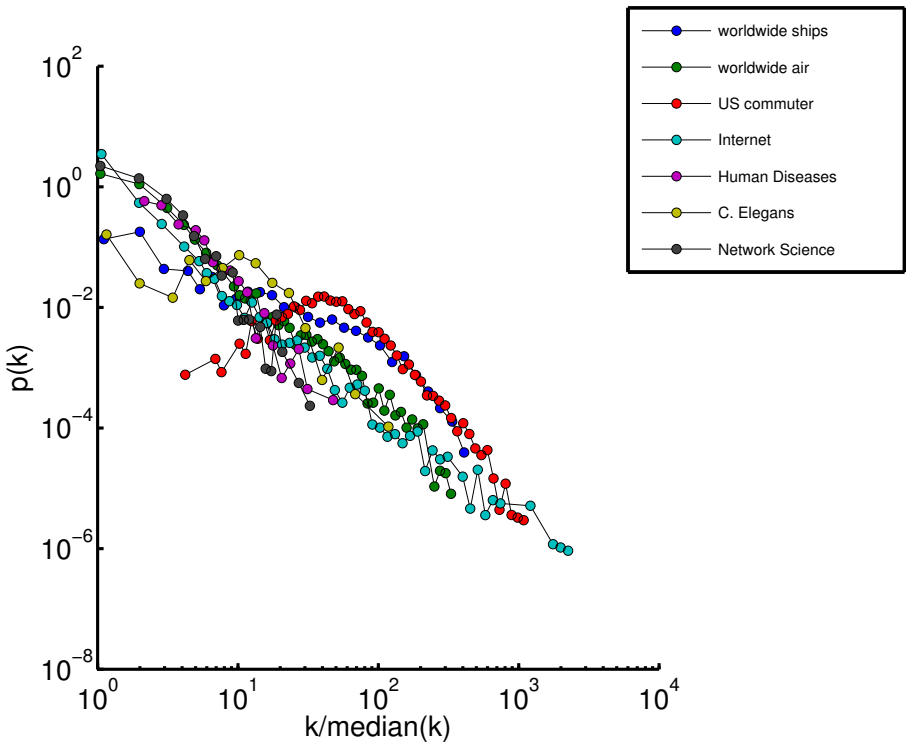


Figure 2.5: Power-law or power-law-like degree distribution $p(k)$ of various networks. In order to compare the shape, the pdf is plotted against k devided by the the median.

Then one of the things we would expect is that as we increase the sample size, this average converges to some constant number

$$\lim_{n\to\infty} \langle X\rangle_n = \langle X\rangle$$

That is just plain intuition of course this limit is

$$\langle X\rangle = \int_{x_0}^{\infty} dx\, x\, p(x).$$

Now let's assume we have a power law

$$p(x) = \frac{\mu}{x^{1+\mu}}$$

one the intervale $[1,\infty]$, so

$$\int_1^{\infty} dx\, p(x) = 1.$$

What would be the average that we would get if we were to compute the sum above? Let's see

$$\int_1^{\infty} dx\, x\, p(x) = \mu \int_1^{\infty} \frac{dx}{x^{\mu}} = \frac{\mu}{\mu-1}$$

if $\mu > 1$. If on the other hand $0 < \mu \leq 1$ then the integral diverges so the average is infinite. We may expect this never to occur in reality and just be a mathematical peculiarity. But it does have real significance. Because it means, even if I have a finite sample S_n computing the average

$$\langle X\rangle_n = \frac{1}{n}\sum_i X_i$$

simply does not converge before I reach the full sample size n. Although the sum of course does give a number, it carries no information because if I were to extend my sample, say double it I'd get a different number. Fig. XX illustrates this. Let's first assume we have a sample of random numbers drawn from an exponential probability distribution

$$p(x) = \alpha e^{-\alpha x}.$$

We can check that

$$\int_0^{\infty} dx p(x) = 1$$

The expectation value is given by

$$\begin{aligned}\langle x\rangle &= \int_0^\infty dx\, x\, p(x)\\ &= \alpha\int_0^\infty dx\, x\, e^{-\alpha x}\\ &= \alpha\frac{d}{d(-\alpha)}\int_0^\infty dx e^{-\alpha x}\\ &= \alpha\frac{d}{d(-\alpha)}\frac{1}{\alpha}\\ &= \alpha\frac{1}{\alpha^2} = \frac{1}{\alpha}\end{aligned}$$

Now we draw N random numbers from this distribution and compute the empirical average

$$\langle X\rangle_N = \frac{1}{N}\sum_{i=1}^{N} X_i$$

as a function of N. In Fig. XX we see that this average converges to the theoretical mean, if $\alpha = 1$ this is $\langle x\rangle = 1$. After a few hundred samples we are pretty close. If we do this for a power law described above for an exponent $\mu < 1$ the empirical average does not converge. In fact, with regular repitions some number are drawn that are as large as the entire sum producing large excursions of the average. As the sample size increases these excursions do not stop and sustain not yielding a plausible average value. A consueqence of the large tail in the power law. Averages hav thus no meaning when power law distributions are involved.

2.5 Weighted Network

Another important source of scale-free-ness is seen in many weighted, real world networks. Recall that these are networks that have a weight W_{ij} associated with each link. This adds another layer of complexity and the first thing one can look at, is the statistics of the weights given by the distribution $p(w)$ and the capacity or strength of a node

$$p(\phi) \quad \text{where} \quad \phi_i = \sum_j W_{ji}.$$

In many real world networks, the capacity as well as the weights exhibit scale free distributions that have no well defined mean or variance.

Fig. 2.6 show the weight and capacity distributions of various real world networks that all exhibit a broad distribution in both quantities.

2.5.1 Correlations of observables

Another universal feature is the connection between degree and capacity. In particular, transportation networks show this correlation. One can easily see this correlation when plotting the

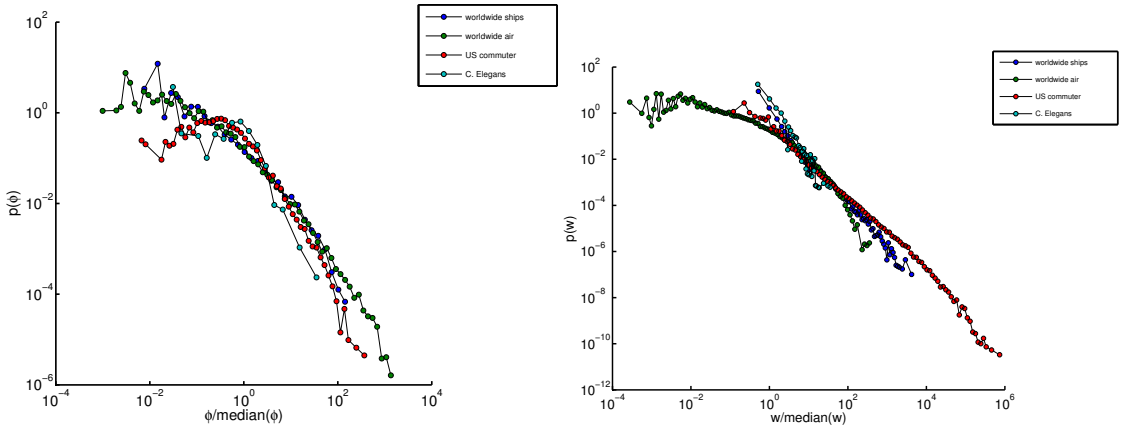


Figure 2.6: Probability density functions of link weights $p(w)$ (left) and of node capacities $p(\phi)$ in real world weighted networks.

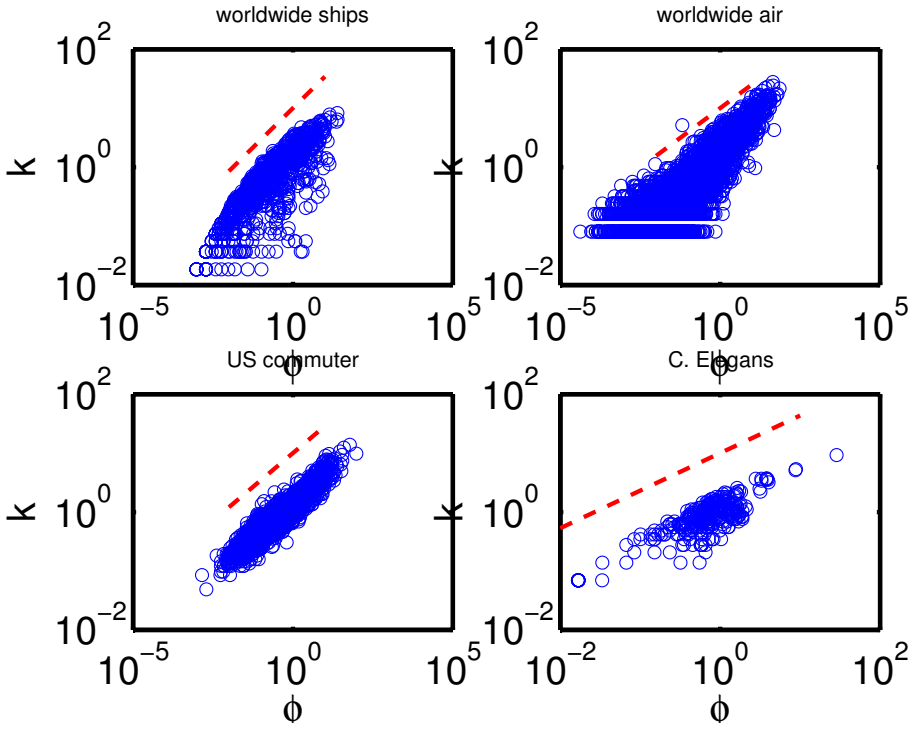


Figure 2.7: Capacity degree correlations in weighted networks.

degree k_i or a node against the capacity ϕ_i. Although scatter plots like this show, as their name suggests, a lot of scatter, an overall trend can be seen in such plots. For instance, in many networks one observes a typical scaling relation

$$k \sim \phi^{\xi}$$

where the exponent $0 < \xi < 1$. For instance many transportation networks have a scaling exponent $\xi \approx 0.6$ which means that if I double the total amount of traffic going into a node I don't double the degree. That in turn means that on average the links must become stronger.

3 Network Models

We have seen in previous chapters and sections how real world networks can be characterized using centrality measures and in large networks their statistics. In this chapter we will investigate network models. Those are networks that are designed according to some basic rules such that their properties capture some of the properties or real world networks.

When we visualize some networks, particularly large networks they look random in the sense that we cannot predict whether two nodes are linked or not. Therefore the most important class of models used in complext network research are networks that possess some sort of randomness or random compontent in their construction mechanism to mimic the degree of unpredictability in real world networks.

3.1 Random Networks

Before we discuss specific random networks we have to get some of the fundamental ideas right. Obviously a random network g is a realization of some process $\mathscr{G}$. The process or rule generates a random network

$$\mathscr{G} \rightarrow g$$

The rule is fixed, but it can generate a number of different random network g_1, g_2.... Just like the fixed process "rolling a die" generates different outcomes $1,2,3,4,5,6$. The foundation of random network theory is that we want to deal with something that is "fixed" so instead of considereing single network g we consider the rule that generates them, $\mathscr{G}$. This is why: Let's assume we have some rule $\mathscr{G}$ that generates a network g. Typically, we cannot infer the rule $\mathscr{G}$ from just a single g as much as we cannot infer the rule "throwing a die" from getting a 3 on the top face of a die. Because, however, we cannot analyse rules so well and we would like to infer them from networks, we have to look at ensembles G of networks that are generated from the rule. For instance if one throws a die many times, make two observations: 1.) we only get the numbers $1-6$ and we get them with equal probability of $1/6$. IF we have that we have all that we need to know. So for networks we have an ensemble of networks

$$G = \{g_i\}_{i=1,....\Omega}$$

where Ω is the total number of possible networks, sometimes that is infinite, and an assosiated proability measure $p(g_i)$ for each network that tells us how likely it is to observe g_i.

3.1.1 Ensemble averages

If we have a process $\mathcal{G}$ and the set of networks it can generate G plus the probability $p(g)$ for each $g \in G$ we can compute averages. Let's say we have a function F that computes some property of a network g, taking it as an argument, for instance the diameter or the maximum degree or the global clustering coefficient, so

$$F = F(g)$$

is some quantity compute from g. There are two possibilities for computing an average

1. We either generate an long sequence $g_1, ...g_n$ using the process $\mathcal{G}$ and compute

$$\langle F \rangle = \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} F(g_i).$$

2. We can make use of the probability of $p(g)$ and the set of all possible outcomes

$$\langle F \rangle = \sum_{i=k}^{\Omega} F(g_k) p(g_k).$$

Note the important difference between 1.) and 2.) is that in the first version we have to sample and infinite numbner of times, and in the second we just have to some over the possible number of networks g, the drawback being that we need to know the probability $p(g)$ for every $g \in G$.

3.1.1.1 Example

Let's look at the following process. We generate a network of 4 nodes and 4 links in which each of the links is drawn from the total set of $N(N-1)/2 = 6$ possible links at random. So we have the following link set

$$\begin{aligned} 1 &\leftrightarrow 2 \\ 1 &\leftrightarrow 3 \\ 1 &\leftrightarrow 4 \\ 2 &\leftrightarrow 3 \\ 2 &\leftrightarrow 4 \\ 3 &\leftrightarrow 4 \end{aligned}$$

So how many networks are in G? That is equal to the number of possibilities to pick 4 links from the set of 6 links so

$$\Omega = \left(\begin{array}{c} 6 \\ 4 \end{array} \right) = \frac{6!}{4!2!} = 15.$$

And if everything is picked at random the probability for each of the 15 networks to be generated by the process is

$$p(g) = \frac{1}{\Omega} = \frac{1}{15}$$

So, for instance the adjacency matrix

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

corresponds to one of those networks generate this way. In fact this network has a triangle and one dangling leaf node (3). In G there are 12 networks of this type and 3 networks in which the 4 nodes are connected in a 4-loop. Let's name all these network $G(4,4)$ meaning that we have 4 nodes and 4 links. What would be the average degree $\langle k \rangle$ expected for a node in a network drawn from this ensemble? It's given by

$$\langle k \rangle = \frac{1}{15} [3 \times 2 + 12 \times k_\Delta]$$

the term 3×2 corresponds to the three networks that have a four-loop and every node has degree 2 and the second term corresponds to 12 networks that have a triangle in which the mean degree of each network is given by

$$k_\Delta = \frac{1}{4} (1 + 2 \times 2 + 3) = 2$$

so

$$\langle k \rangle = \frac{1}{15} [3 \times 2 + 12 \times 2] = 2.$$

That was complicated and obvious. Each of the networks has 4 nodes and 4 links to the average degree of every network has to be $2N/L = 2$.

But how about the clustering coefficient? In the 12 networks that have a triangle the global clustering coefficient is $C_\Delta = 1/3$. In the three networks with a 4-loop the clustering coeff. is 0. So the mean clustering coeff. for the entire ensemble G is given by

$$\langle C \rangle = \frac{1}{15} \left[3 \times 0 + 12 \times \frac{1}{3} \right] = \frac{4}{15} < \frac{1}{3}.$$

This is important, because if we were to generate a network g by the process $\mathscr{G}$ and the clustering coefficient of that specific g will not be equal to the clustering coeff. of the entire ensemble.

3.2 The ensemble $G(N, L)$

The example above is a special case of $G(N, L)$ the ensemble of networks that have N nodes and $0 \le L \le L_{\max}$ links where $L_{\max} = N(N-1)/2$. The number of networks in $G(N, L)$ is given by

$$\Omega = \begin{pmatrix} L_{\max} \\ L \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} N \\ 2 \end{pmatrix} \\ L \end{pmatrix}.$$

Typically, this is a large number, For $N = L = 10$ we have $\Omega = 3,190,187,286$.

3.3 Erdős-Rényi Random Networks

Very much related to $G(N,L)$ is the ensemble $G(N,p)$. This ensemble is generated in a slightly different way. Instead of fixing the number of links L we take the entire set of links and pick each with probability p. Generally, every network g constructed this way may have different number of links L. the ensemble $G(N,p)$ is called the ER-network. How many networks are in this set? Since any combination of links has a finite probability the entire set has

$$\Omega = 2^{L_{\max}} = 2^{N(N-1)/2}.$$

The probability of generating a network g with L links is not uniform accross G, for instance the probability of generating a network with no links and a network with all links present is very small compared with networks with intermediate values since a lot more combinations exist of this type. What is the probability of generating a network with L links? This can be obtained by the following reasons. First, the probability of picking L links and not picking the remaining $L_{max} - L$ links is

$$p^L(1-p)^{L_{max}-L} = p^L(1-p)^{\binom{N}{2}-L}$$

But we can permute the L links so we get

$$P(N,L) = \binom{\binom{N}{2}}{L} p^L(1-p)^{\binom{N}{2}-L}$$

where $P(N,L)$ is the proability that a network g in $G(N,p)$ has L links. This is a binomial distribution, if we sum over L we see that the distribution is normalized

$$\sum_{L=1}^{L_{max}} P(N,L) = 1$$

With the same argument we can show that the degree distribution of a network $g \in G(N,p)$ is given by

$$p(k) = \binom{N-1}{k} p^k(1-p)^{N-1-k}$$

3.3.1 Mean degree

If we want to compute the mean degree we can do the following. Given a network with L links the mean degree is

$$k = \frac{2L}{N}$$

of this particular network. The mean degree with respect to the entire ensemble is therefore

$$\langle k \rangle = \frac{2\langle L \rangle}{N}$$

The mean number of links $\langle L \rangle$ is computed from the distribution $P(N,L)$, which is the mean of the binomial distribution, so

$$\langle L \rangle = L_{max} p$$

so the mean degree is

$$\langle k \rangle = \frac{2L_{\max} p}{N} = p(N-1).$$

which makes sense. This is also what we obtain if we use $p(k)$ given above.

3.3.2 The local clustering coefficient

Remember that the local clustering coefficient is the probability that two neighbors of a node are also connected. Since the probability of any link being there is p that is also the probability that two neighbors of a node are connected to the local clustering coefficient is

$$\langle c \rangle = p.$$

3.3.3 Diameter of $G(N,p)$

The diameter of an ER network can be roughly estimated just using the mean degree k_0 and the network size. If we take l steps away from a given reference node, then the number of nodes we can reach is rouchly given by

$$n(l) \approx k_0^s$$

At some point $n(l)$ is going to be of the order N and saturate, the diameter is then approximately the number of steps we need to take until this happens so

$$k_0^l \approx N$$

which means that

$$l \approx \frac{\log N}{\log k_0} \sim \log N$$

This alone gives a handwaving argument for the small world nature of many real world network.

3.4 The Poisson Random network

This is an important type of network obtained from $G(N,p)$ in the limit $N \to \infty$. In many situation we have large networks in which the typical degree

$$\langle k \rangle = p(N-1)$$

is fixed, even if we keep increasing N. So a meaningful limit is $N \to \infty$ and $p \to 0$ keeping the mean degree fixed. If we let p become very small, the binomial distribution

$$p(k) = \left(\begin{array}{c} N-1 \\ k \end{array} \right) p^k (1-p)^{N-1-k}$$

becomes

$$p(k) = \frac{\langle k \rangle^k}{k!} e^{-\langle k \rangle}$$

which is the Poisson distribution.

3.4.1 Implications

So we see that the ER Networks exhibit both, the small world effect and a vanishing local clustering coefficient has we increase N keeping the mean degree k_0 fixed. We might conclude from this that maybe both phenomena go hand in hand, that vanishing local clustering coefficient implies the small world effect and vice versa. We will shortly learn that this is not so. In fact, the small clustering coefficient is actually a significant drawback of the ER network. Most networks that possess small world characteristics do in fact have a large clustering coefficient. Also, the Poissonian degree distribution is not a good model for real world network which often have broad, e.g. power law degree distributions. It is therefore plausible to develop a model that can generate an arbitrary degree distribution. Which lead directly to the...

3.5 Configurational Model

The idea behind this is the following. Say we have good reason to believe that some function $p(k)$ is a good model for a real world network that exhibits a degree sequence

$$S = \{k_1, ..., k_N\}$$

One way of generating an ensemble of random networks with a degree distribution consistent with S based on the assumption that $p(k)$ is a good model for te real network is by using the following process $\mathscr{G}$:

1. Generate a random degree sequence for N nodes

   $$k_i, i = 1, ...N$$

2. The total number of link stubs is given by

   $$L = \sum_i k_i$$

   and if we connect the stubs (neglecting the emergence of self loops and multi-links) we have

   $$L = \frac{1}{2} \sum_i^N k_i$$

   links in the entire network

3. We now randomly connect the stubs as discussed earlier and we get a random network that is statistically indistiguishable from a network that has a degree distribution consistent with $p(k)$.

3.5.1 The probability of two nodes being connected

In the ER network, the probabilty that a pair of nodes i and j are connected is constant, p. In the configurational model this is different. If node i's degree is k_i let's first pick one of the k_i stubs. If we randomly connect it to the other stubs then with probability

$$q = k_j/(2L-1)$$

this one link connects to j. Adding all these probabilities this gives

$$p_{ij} = \frac{k_i k_j}{2L-1} \approx \frac{k_i k_j}{2L}$$

3.5.2 Self-loops and multi-links

One of the annoying properties of the configurational model is that by randomly connecting stubs we autmatically create self-loops and multiple links between nodes. If we want to ingnore those, we have to make sure that in the limit of large networks their effect is negligible. One can show, using the equation above that the number of multilinks and self-loops is proportional to

$$\left(\frac{\langle k^2\rangle - \langle k\rangle}{\langle k\rangle}\right)^2 = \text{const.}$$

which means, it does not increase with network size as lonk as the moments are constant. Thus the probabiliy of finding a self or multi-link vanishes when $N \to \infty$.

3.5.3 Generating a configuration without self- or multiple links

One way of going around this is the following. Let's say we have a degree sequence

$$S = \{k_1, ..., k_N\}$$

and this sequence is drawn from a distribution $p(k)$. We can now make use of the expression

$$p_{ij} = \frac{k_i k_j}{2L-1}$$

and say that this is the probability that a link exists between i and j and

$$q = 1 - p_{ij}$$

that there is no such link. We can then define that

$$p_{ii} = 0$$

and only throw the dice for the link i and j once. The drawback is that the resulting network may not have the original degree sequence, buy an ensemble generated like this has a $p(k)$ consistent with S. That is actually not quite true, only for large networks it is true. Nevertheless this is a better way of generating networks with a specfied degree distribution.

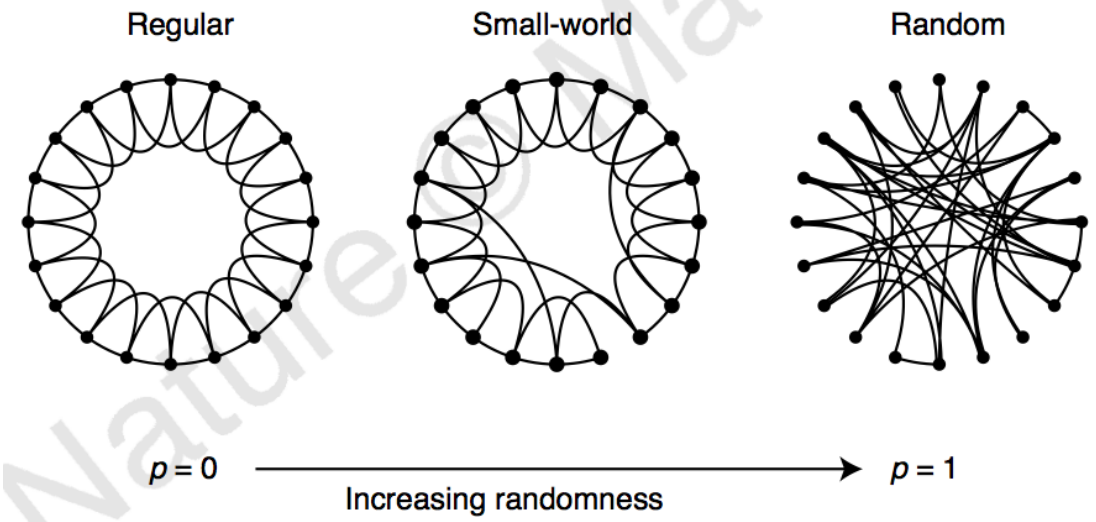


Figure 3.1: Watts-Strogatz small world network.

3.6 Watt-Strogatz Small World Networks

In one of the most cited scientific articles in network science Duncan Watts and Stephen Strogatz designed a model in which they showed that one can have a situation in which the local clustering coefficient is high and nevertheless the network is small world. The idea of the model is to find a way of going from one well known extrem to another well known extreme using one parameter. Extrem one is a regular linear lattice on a ring in which are coupled to their m nearest neigbors, so we have $L = mN$ links in the network. This regular lattice is not a small world network because the number of hops that separate two randomly chosen nodes scales with the system size, so is of order N. Likewise if we calculate the local clustering coefficient, because we have a lattice, we typically get a clustering coefficient which is comparatively large

$$C_0 = O(1)$$

because the local neighborhood is well connected in a regular lattice.

On the other hand in an ER network with the same number of nodes and links, we have a mean degree of

$$k_0 = m$$

and an average clustering coefficient of

$$c \approx \frac{m}{N}$$

which for large networks becomes very small, if say $m = 4$ and $N = 1000$. The idea of the Watts Strogatz modell is this. We introduce a probability $0 \leq p \leq 1$ and for each of the mN links in

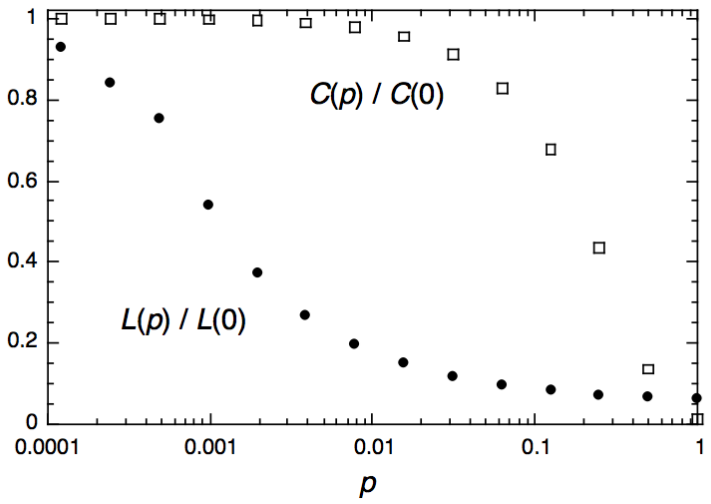


Figure 3.2: Clustering Coefficient and Diameter in Watts-Strogatz small world network.

Network	N	L	$\langle l \rangle$	$\langle c \rangle$	c_{ER}
Film Actors	449,913	25,516,482	3.48	0.78	2.5×10^{-4}
Physics Coauthorships	52,909	245,300	7.57	0.34	1.7×10^{-4}
Internet	10,697	31,992	3.31	0.39	5×10^{-4}
Metabolic Network	765	3,686	2.54	0.67	0.0125
Protein Interactions	2,115	2,240	6.80	0.071	0.001

Table 3.1: Small world networks

the network we do this. We rewire one end of the link from a local neighbor to a randomly chosen node in the network, thereby creating a shortcut to potentially distant locations. This way, if $p = 0$ we have our lattice and of $p = 1$ we have a random, ER like mode. The question is of course how does the diameter of the network and the clustering coefficient depend on p. The fundamental result of Watts and Strgatz was even if p is extremely small, the diameter $L(p)$ decreases very rapidly with p whereas the local clustering coefficient does not. In fact, $L(p)$ very quickly drops and unless p becomes very large $C(p)$ remains constant. This resolved the issue of many real networks having a large local clustering coefficient yet a small diameter.

3.7 Static scale free networks

One way to generate scale free network with a power-law degree distribution is just using the configuration model discussed above. Generating a degree sequence

$$S = \{k_1, ..., k_N\}$$

from a scale-free degree distribution $p(k)$ and linking nodes i and j according to the probability

$$p_{ij} = \frac{k_i k_j}{2L-1}.$$

However, a power law in $p(k)$ has the tendency to create lots of self loops because the hubs tend to connect to themselves effectively decreasing the degree.

3.7.1 The Model by Goh et al.

Several other recipes to construct static scale-free networks, based on assuming that a node has some intrinsic properties, have been proposed in the physics community. In one model, each node i is assigned a weight (or fitness) $f_i = i^\alpha$, where $i = 1, 2, ..., N$ is the node index and α is a tunable parameter in [0, 1). Two different nodes, i, j are selected with probabilities equal to the normalized weights

$$p_i = \frac{w_i}{\sum_k w_k} \quad \text{and} \quad p_i = \frac{w_i}{\sum_k w_k}$$

respectively, and are connected if there is not already a link between them.

The process is repeated until mN links are made so that $< k >= 2m$. When $\alpha = 0$ one obtains a ER random graph. When $\alpha > 0$ the graph obtained has a power law degree distribution

$$p(k) \sim \frac{1}{k^{1+\mu}}$$

with an exponent $\mu = 1/\alpha$. Thus, by varying α in [0, 1) one obtains an exponent in the range$1 < \mu < 2$.

3.8 Preferential Attachment Models

So far we've been discussing ways to analyse models and how to generate networks that resemble real networks to some extend. An entirely different class of models goes a bit further by providing basic network growth mechanism some of which yield networks that exhibit properties of real networks. These are in this case emergent properties. One very famous mechanism can explain the powerlaws we've been seeing in degrree distributions of many networks

$$p(k) \sim \frac{1}{k^{1+\mu}}.$$

Most of them go back to fairly old ideas. The first person to systematically address this question has Herbert Simon in the 60's. He borserved that power-law distributions in general are seen everywhere, biology, sociology and economics. He invented the idea of cumulative advantage and motivated this by economic ideas and coined the principle of

- The rich get richter phenomenon based on
- cumulative advantage "the more money i've got the more money I make"

He explained this using the example of work frequencies in text. He assumed

1. the $k+1$th worh in a sequence of words is a word previously chosen in the sequence with a probability proportional to the number it has appeared already
2. at a constant rate new words appear

3.8.1 The Price Model (1965)

In 1965 Price adapted these ideas to networks. He wanted to understand the structure of the network generated by scientific articles that cite one another This model adapts the two concepts of

1. growth
2. cumulative advantage

to understand the network of citation in scientific articles. Price noticed that the number of citations a paper gets exhibit a power law distribution. Given a snapshot of all the scientific articles at a given time, one can think of those as a directed network, in which each node is a paper that cites other paper by connecting to them. Therefore if paper i cites paper j there's a link from i to j, so $A_{ji} = 1$ (but $A_{ij} = 0$). So we have a picture as illustrated in Fig. 3.3.

The in-degree

$$q_i = \sum_j A_{ij}$$

is the total number of citation a paper has and the out-degree

$$k_i = \sum_j A_{ji}$$

is the number of paper that paper i cites. The first think one must notice is that the out degree k has a typical size, say 10-20 citations, whereas the in-degree can be very small or very large. Also, when a new paper is added, it comes in with it in-degree into the network, and by connecting it changes the out-degree of the network. At a given point we have

$$\sum_i q_i = \sum_i k_i = L$$

the number of references in the whole body of papers. The idea of Price was the following, making the following assumptions

- When a new paper comes in, it does so by citing and average number $c = \langle k \rangle$ other papers. This number does not have to be constant for every paper that comes in, but it should have an average c.
- A paper i cites another paper with a higher probability if that paper has a high in degree q_j, meaning that paper j has already lots of citations. This is the cumulative advantage.

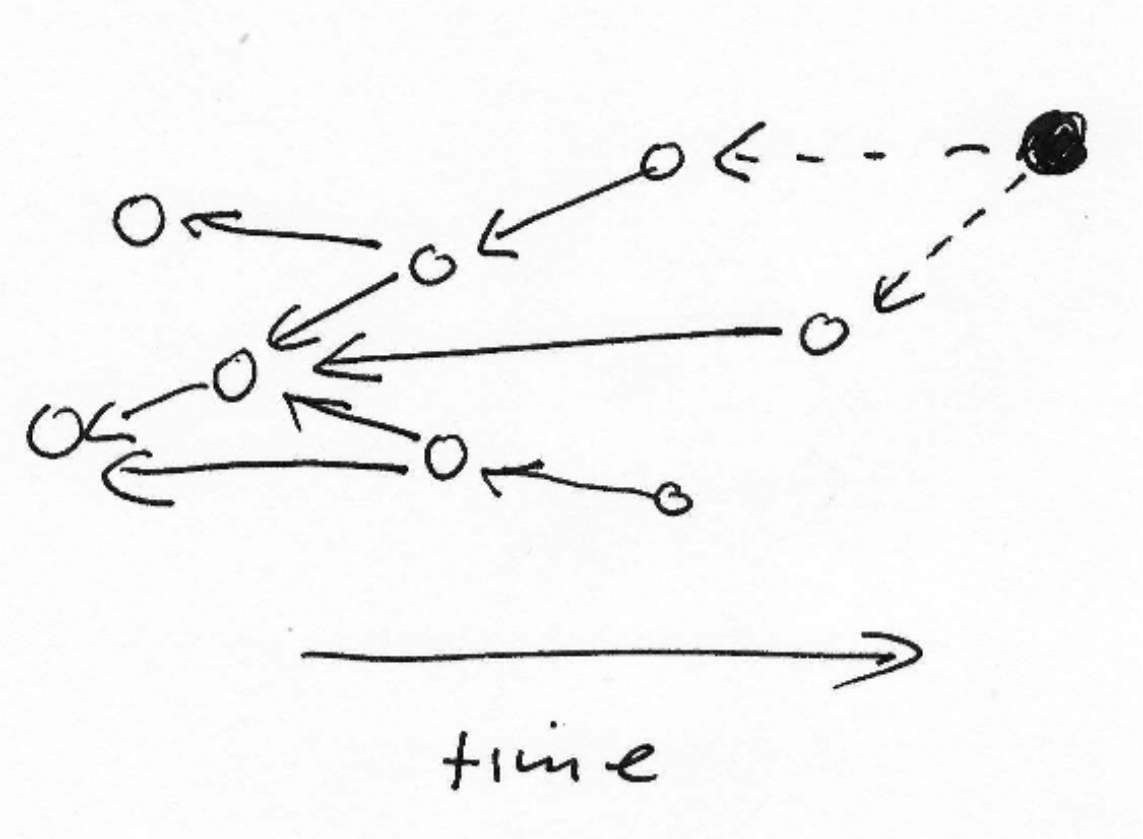


Figure 3.3: The network of scientific papers can be considered as a directed graph. If a new paper appears it has cites a typical number c of papers in the set and generates new connections to other nodes.

In general we have to pick the out-degree from a distribution

$$p(k) \quad \text{with} \quad \langle k \rangle = c$$

the simplest choice being

$$p(k) = \delta_{kc}$$

where δ is the Kronecker delta but we could also chose

$$p(k) = \frac{c^k}{k!} e^{-c}.$$

which also has

$$\langle k \rangle = c$$

Let's assume at time t we have n papers that are available for citation and let's label them j. Then according to the second rule we should have

$$p(j) = f(q_j)$$

where $p(j)$ is the probability of citing paper j. So we have to have

$$\sum_j p(j) = 1$$

and we need an increasing function $f(q)$ taking into account the accumulative advantage. Price made the following asssumtion

$$f(q) \propto (a + q)$$

where a is a nonzero constant. This is supposed to model that also papers that haven't been cited at all have a probability of being cited. So for instance the ratio between a paper that has no citation and one that has 1 citation is given by

$$\frac{f(0)}{f(1)} = \frac{a}{1+a}$$

If $a = 1$ then $f(1) = 2f(0)$ if $a = 10$ then $f(0) \approx f(1)$. From this it follows that

$$p(j) = \frac{a + q_j}{\sum(a + q_j)} = \frac{a + q_j}{n(a + \langle q \rangle)} = \frac{a + q_j}{n(a + c)}.$$

In order to obtain the distribution of in-degrees $p(q)$ we have to do computations that we will not get into here. The basic idea is this. If we have at time t a network of n nodes and a distribution of in-degrees given by $p(q, t)$. Then by adding a new node, we change that distribution to $p(q, t + 1)$ and we can write down what is known as a Master-equation for the evolution of the distribution of in degrees. Let's call n the number of papers in the set at time t. Then $np(q, t)$

are the number of papers that have q citations and $(n+1)p(q,t+1)$ are the number of papers with q citations afterwares. We can then write

$$(n+1)p(q,t+1) = np(q,t) + w(q-1)\,p(q-1,t) - w(q)\,p(q,t).$$

What is this? This is a balance equation. At t we have $np(q,t)$ papers with q citations. Some of these will get cited by the paper that we add at time t and become papers that have then an in-degree of $q+1$. How many papers, on average will that be? that will be the the number of papers with degree q i.e. $np(q,t)$ times the probability that each receives a new citation times the number of citations added to the system which is

$$w(q) = \frac{c(a+q)}{n(a+c)}$$

so

$$w(q) \times np(q,t) = \frac{c(a+q)}{(a+c)}\,p(q,t)$$

Likewise some nodes become papers of in-degree q, those that at time t had degree $q-1$. The number of those are $np(q-1,t)$ and the likelihood of them being promoted to q is

$$w(q-1) = \frac{c(a+q-1)}{n(a+c)}$$

so alltogether we get

$$(n+1)p(q,t+1) = np(q,t) + \frac{c(a+q-1)}{a+c}\,p(q-1,t) - \frac{c(a+q)}{a+c}\,p(q,t).$$

This is a complicated equation, but it can be solved. The solution is a bit tedious. The key result is that when one tries to investigate the limit $t \to \infty$ or instead finding a solution $p(q,t)$ that is time independent one finds

$$p(q) \sim \frac{1}{q^{1+\mu}}$$

with

$$\mu = 1 + \frac{a}{c}$$

This is the key. The price model naturally yields a power-law of in degree in the directed network of paper-citations with only the two ingredients growth and cumulative advantage. The exponent is determined by the two parameters a and c in general one assumes $a \ll 1$ and $c \approx 10-20$ which means that $\mu \approx 1$. For most of the models we saw power-laws in we mesaureed $\mu = 2$ which would imply $a = c$. This would be fitting. The problem is that we have not really a way for measuring a, whereas c is no problem. Shortly we will discuss a natural system that generates $\mu = 2$. First let's breifly discuss how to generate such a scale free network in a computer.

3.8.2 Price model Numerically

To investigate the price model numericall one first fixes the parameters a and c to say $a = 1$ and $c = 10$. We then start with say $n = 50$ nodes that are all unconnected. We also have to specify the distribution for out-degrees k, $p(k)$. At every step we

1. increase $n \to n+1$
2. draw a random number from $p(k)$ for the out degree of the new node
3. for each of the k new links we attach with probability
$$p_j = \frac{a+q_j}{na+c}$$
to node j
4. return to step 1

The only difficulty in this process is that we have to compute p_j for all the nodes at every step which may take a long time if we want to generate large networks. So there's a better way.

3.8.2.1 Krapivsky-Redner-Method

when we attach a new link let's do one of two things

1. with probability ϕ we attach the link proportional to the in-degree
$$Q_j = \frac{q_j}{\sum_j q_j} = \frac{q_j}{nc}$$
2. with probability $1-\phi$ we attach uniformly among all the j target nodes
$$P_j = \frac{1}{n}$$

Then the probaiblity of connecting to j is

$$p_j = \phi\frac{q_j}{nc} + (1-\phi)\frac{1}{n}$$

Now let's pick

$$\phi = \frac{c}{c+a}$$

in this case we get

$$p_j = \frac{a+q_j}{na+c}$$

which is what we want. This is a clear advantage. All we have to keep track of is the list of arrowheads of the links. If we pick from that list uniformly we will pick nodes attached to the arrowheads proportional to their in-degree.

3.8.3 The Barabasi-Albert Model

The Price model is a model for a directed network in which we distinguish between in and out degree. A similar model was developed by Barabasi and Albert 34 years later for undirected networks. This model starts with a small set of m nodes. At each step a new node is added with degree m and attaches to the nodes that already exist in the network propotional to their degree

$$p(i,j) = \frac{k_j}{\sum_j k_j}.$$

This is a special case of the price model and it can be shown that the key feature of this model is an asymptotic power-law in the degree distribution

$$p(k) \sim \frac{1}{k^3}.$$

4 Network resilience

The topic of network resilience is related to the question:

- What happens to the network if we successively remove some network elements, i.e. nodes or links, randomly or in some informed fashion?

Basically we can distinguish between two different procedures, either we remove links or we remove nodes and their links.

By informed, we mean we could remove elements by their rank, e.g. terrorists may target strongly central nodes, e.g. those that have a high degree or high betweenness centrality.

4.1 Measuring impact

The whole topic about network resilience is a global topic. We would like to know what happens to a network globally. The two things that measure something globally that come to mind are:

1. The diameter of the network. Recall that this is the mean shortest path d of the network.
2. The components of the network

If we start removing nodes, we would expect

- The diameter of the network to increase, because by removing elements we remove possible shortest paths of the network
- The overall number of components to increase because when we elements we may separate connected components.

These are qualitatively correct ideas, and we could immediately test the on the computer. Before we do so, let's look at some of the theory that resembles the backbone of many of these studies.

4.2 Percolation Theory

The origin of network resilience is percolation theory that originally was developed for d-dimensional lattices, generally $d \geq 2$. Let's look at a two-dimensional lattice. Percolation theory asks the following questions. In the limit of an infinite lattice, what is the fraction of

- nodes (site percolation) or

- links (bond-percolation)

that has to be removed in order to find a path from one end of the lattice to the other end. Of course, it is clear that this depends on which elements of the lattic I remove. So let's assume we remove them randomly. Also in this case the answer may depend on the particular random choice of elements we remove.

It turns out however that in the limit of infinite system size one can show that for a given fraction of remove elements, the probability that one can find a path from one end to the other is either 0 or 1. The key question is form which fraction p_c or elements has to be removed. This turns out to be an intensely difficult question to anwers, except for in a few simple cases. For example for bond percolation on a quare lattice we have $p_c = 1/2$. That means half the bond need to be removed. for $p > p_c$ there is not path trough the system, for $p < p_c$ there is. The critical value p_c is called the percolation threshold.

For site percolation in 2d no analytic result exists the numerical value so far obtained is

$$p_c = 0.592746$$

No one knows why. For the triangular lattice the situation is reversed the site percolation threshold is $p_c = 1/2$ and the bond percolation threshold is

$$p_c = 0.347296$$

In networks the question is related to the question: how many elements do we have to remove in order for the network to fall apart. Recall that many networks possess a giant component when they are sufficiently connected and below a certain connectivity the giant component disappears.

4.3 Percolation in Network

4.3.1 Complete networks

Let's first consider fully connected networks. Let's start with a network that has N nodes and $L = N(N-1)/2$ links. Let's now remove a fraction of p nodes. What is the critical p such that the giant component disappears. Coming from a fully connected network and removing a fraction of p links is equivalent to starting with no links and adding $q = 1-p$ links. So a complete network with a fraction p removed is equivalent to and ER network with connectivity parameter q.

And we learned earlier that the critical mean degree $k_c = 1$ for an ER network. And since

$$q(N-1) = k$$

we have

$$q_c = \frac{1}{N-1}$$

And therefore

$$p_c = 1 - \frac{1}{N-1} = \frac{N-1-1}{N-1} = \frac{N-2}{N-1}$$

So we have to remove almost all nodes, leaving

$$q_c \approx \frac{1}{N}.$$

4.3.2 Configurational Model

Let's a assume we have a network with degree distribution $p(k)$ and mean degree $\langle k \rangle$. One of the things we have to recall is that the probability distribution for the degree of a neighbot is given by

$$q_1(k) = \frac{kp(k)}{\langle k \rangle},$$

with the mean degree

$$\langle k \rangle_1 = \frac{\langle k^2 \rangle}{\langle k \rangle} \geq \langle k \rangle .$$

Let's recall briefly, how this unual result came about. Let's say you are a node in the configurational model and you look at one of your links. In the configurational model, this link of yours is going to be connected to one of the $2L-1$ link-stubs in the rest of the network with equal probability. Now pick a single node that has degree k so has k stubs. with probability

$$\frac{k}{2L-1}$$

your link is going to be connected to exactly this node. How likely is it that your link is going to be connected to any link of degree k. There are $np(k)$ nodes of this type to this probability is

$$q(k) = \frac{k}{2L-1} \times N \times p(k) = \frac{N}{2L-1} \times k \times p(k) = \frac{kp(k)}{\frac{2L-1}{N}} = \frac{kp(k)}{\langle k \rangle}.$$

On average, your friends have a higher connectivity than you do. We will need this result in order to understand the percolation transition in the configurational model.

Sometimes we are interested in the so called excess degree distribution, the pdf that the degree of the node is k but not counting the link on which I arrived. That's simply a shift to the right

$$q_e(k) = q(k+1) = (k+1)p(k+1)/\langle k \rangle$$

Let's assume we remove a fraction p of nodes from a network and let's define

$$\phi = 1 - p$$

as the fraction of nodes that are still in the network. Lets define u as the probability that a node does NOT is not connected to the giant component through a chosen neighbor. Let's focus on a

node of degree k. Then the probability that this node is not connected at to the giant component is u^k. What is the prob. that the node has degree k? It's $p(k)$ so the probability of any node not being connected to the GC is

$$g_0(u) = \sum_{k=0} p(k)u^k.$$

So the probability that the node IS connected to the GC is given by

$$1 - g_0(u).$$

We are looking at a situation in which we have removed a fraction of $1-\phi$ nodes. Obviously we are only investigating those nodes that have not been removed, which is ϕ. The other ones are disconnected already by removal. So the total fraction of nodes that belong to the giant component are given by

$$S = \phi\left[1 - g_0(u)\right].$$

This isn't helpful yet, because we still have to compute the quantity u which is the probability that a node is NOT connect to the GC via a chosen neighbor. Again there are two ways of NOT being connected to the GC:

1. Either the chosen neighbor is not part of the GC and is present (with prob. ϕ)
2. or the neighbor has been removed (with prob. $1-\phi$).

If the neghbor, let's label it A is still around and has degree $k'+1$ then the probability of that neighbor itself not being connected to the GC is $u^{k'}$. So adding both probs. gives

$$Q(k') = 1 - \phi + \phi u^{k'}.$$

What is the probability that the neighbor has degree k'. It's given by the excess-degree distribution $q(k')$. Thus, the probability that the original node is not connected to the GC is given by the expectation values of Q

$$\sum_{k'=0} (1 - \phi + \phi u^{k'}) q(k'+1)$$

but this is also the probability of not being connected to the GC which is u, so

$$u = 1 - \phi + \phi \sum_{k=0} u^k q(k+1)$$

which we write as

$$u = 1 - \phi + \phi g_1(u)$$

where

$$g_1(u) = \sum_{k=0} u^k q(k+1).$$

Now we have two equations:

$$S = \phi\left[1 - g_0(u)\right],$$

and

$$u = 1 - \phi + \phi g_1(u)$$

We can use the second one to compute u, plug that in to the first and get $S(\phi)$, the size of the giant component. Now we can solve this graphically. trivially $u = 1$ is a solution because

$$g_1(1) = 1$$

One can show that $0 \leq g_1(u) \leq 1$ and that $g_1(u)$ is monotionically increasing with u. Then three scenarios can happen.

1. If $\phi > \phi_c$ then a solution $u^\star < 1$ exists which means not everything does NOT belong to the GC
2. If $\phi < \phi_c$ then $u = 1$ is the only solution, which means no GC exists and
3. $\phi = \phi_c$ is the critical point when things fall apart.

The third condition is true when the derivative of the rhs. is 1 so when

$$1 = \phi_c \left| \frac{dg_1(u)}{du} \right|_{u=1}$$

So

$$\frac{dg_1(u)}{du} = \frac{d}{du}\sum_{k=0} u^k q(k+1) = \frac{1}{\langle k \rangle}\sum_{k=0} k(k+1)p(k+1) = \frac{\langle k^2 \rangle - \langle k \rangle}{\langle k \rangle}$$

and thus

$$\phi_c = \frac{\langle k \rangle}{\langle k^2 \rangle - \langle k \rangle} = \frac{1}{\kappa - 1}$$

with

$$\kappa = \frac{\langle k^2 \rangle}{\langle k \rangle} > 1.$$

Thus, the percolation transition in the configurational model is given by only the moments of the degree distribution.

4.3.2.1 Example

So let's look at a Poisson Random Network (ER in the limit of large N) then we have

$$p(k) = \frac{k_0^k}{k!} e^{-k_0}$$

where k_0 is the mean degree. Then

$$\langle k \rangle = k_0$$

and

$$\langle k^2 \rangle = k_0(k_0+1)$$

so that

$$\phi_c = \frac{k_0}{k_0^2 + k_0 - k_0} = \frac{1}{k_0}$$

Therefore, if for instance the mean degree is 4 then even if I remove 3/4 of the nodes, my network is still well connected. If the mean degree is 10 I can remove 90% of the nodes and onlythen reach the critical point.

4.3.3 Percolation in scale-free networks

Remember that many scale free networks have a degree distribution

$$p(k) \sim \frac{1}{k^{1+\mu}}.$$

with an exponent $0 \leq \mu \leq 2$ for instance the BA network has $\mu = 2$. And remember that for these exponents we have

$$\langle k^2 \rangle = \infty$$

According to the reasoning above this would imply that

$$\phi_c = 0$$

which means that in scale free networks I can keep removing links and never destroy the giant component. Intuitively this is consistent withthe idea that these networks have a hub and spoke structure, and it is very unlikely that by removing nodes randomly I hit the hubs that connect everything.

5 Dynamical Processes on Networks

5.1 The simplest model

The idea of percolation is useful for trying to understand under which circumstances contagion processes can spread through a network. If we have a network that has a giant component, we know that we have to remove a fraction

$$f = 1 - \phi_c$$

in order for that giant component to fall apart and thus keep a contagion spread thoughout the entire network. This picture however is slightly oversimplifying what is going on. Let's look at epidemics of individuals. People become infectious and can spread a disease to their neighbors. For many phenomena there's a typical period in which people are infectous. Effectively that means that if a node i has k_i neighbors, then for each link there is a transmission probability $0 \leq T \leq 1$ to spread the disease across each link.

How can we determine if a disease spreads through a network. This will be the case if every transmission accross a link, on average generates more than 1 transmission acrross subsequent links. The ratio of the two is called the reproduction number

$$R = \text{average number of secondary infections caused by one transmission.}$$

So given there is a transmission from node i to node j, then

$$R = T \times \kappa > 1$$

where κ is the expected degree of node j. This is equal to the excess distribution, because j is the neighbor of i and has an average degree to nodes (except for i) given by κ. Earlier we learned that this is related to the average ndoe degree by

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

So the critical transmission probability is given by

$$T_c = \frac{1}{\kappa} = \frac{\langle k \rangle}{\langle k \rangle^2 - \langle k \rangle}$$

So for an ER network we find

$$T_c = \frac{1}{\langle k \rangle}$$

which makes sense. But for a scale free network with

$$p(k) \sim \frac{1}{k^{1+\mu}} \quad \text{with} \quad \mu \leq 2$$

we find

$$T_c = 0$$

This means that arbitrarily low transmission probability T will generate an epidemic in a scale free network. This is because with probabiliy 1 a transmission will be to a hub and from there it goes everywhere. Of course this is only strictly true in an infinite system. For real scale-free networks T_c is finite but will decrease with N.

5.2 General Epidemic Models

To get a better idea about spreading phenomena we need to look at models for spreading phenomena that capture more of the dynamics. This will also enable us to extract specifically what the effects will be that are induced by network topologies. So let's first look at models that describe the dynamics of epidemics in single populations.

5.2.1 The SIS model

This model assumes that we have a population with N individuals where N is large enough so that fluctuations can be neglected. We also assume that individuals can either be susceptible (S) on infectious (I) so that at any point in time we have

$$S + I = N,$$

individuals do not enter or leave the population. We assume that generally two reactions can occur

$$\begin{aligned} S + I &\xrightarrow{\alpha} 2I \\ I &\xrightarrow{\beta} S \end{aligned}$$

The first reaction represents the transmission, an infected interacts with a susceptible and after that reaction we have 2 infecteds. The second equation represents that infecteds recover after a certain typical time. The parameters are

$$\beta: \quad \text{the recovery rate of an individual}$$

This means that $T = 1/\beta$ is the time an individual remains infectious before becoming S again. E.g. this could be $T = 3$ days. The parameter α is the transmission rate, for example something like 3 infections per day caused by one individual. To that the quantity

$$R_0 = \alpha \times T = \alpha/\beta$$

is the average number of infectious caused by one infected individual in a susceptible population.

Let's now assume that each of the reactions induce a change in the number of susceptibles and infecteds. So that

$$\begin{aligned} S(t+\Delta t) &= S(t) + \Delta S_1(t) + \Delta S_2(t) \\ I(t+\Delta t) &= I(t) + \Delta I_1(t) + \Delta I_2(t) \end{aligned}$$

For the first reaction we assume that

$$\Delta I_1 = \Delta t \times \alpha \times S \times \frac{I}{N}$$

each of the S susceptible (the S factor above) can become infected by interacting with an infected individual (the factor I/N). Likewise because of reaction 1 we have

$$\Delta S_1 = -\Delta t \times \alpha \times S \times \frac{I}{N}$$

Reaction two induces a change

$$\Delta I_2 = -\Delta t \times \beta \times I = -\Delta S_2.$$

Now we can combine these to obtain the differential equations

$$\begin{aligned} \frac{dS}{dt} &= -\alpha SI/N + \beta I \\ \frac{dI}{dt} &= \alpha SI/N - \beta I \end{aligned}$$

Note that this means that

$$dN/dt = 0$$

as expected. So we can use $S = N - I$ to reduce the set of two odes to one

$$\frac{dI}{dt} = \alpha(N-I)I/N - \beta I.$$

Now we can use the relative fraction of infecteds $x = I/N$ which yields

$$dx/dt = \alpha x(1-x) - \beta x.$$

This is the SIS model. Note that $x = 0$ and

$$x^\star = 1 - \frac{1}{R_0}$$

are the stationary solutions of this where $R_0 = \alpha/\beta$. It turns out that the non-trivial fixed point above is stable if $R_0 > 1$ and unstable if $R_0 < 1$ (in which case $x = 0$) is stable. So the SIS has a threshold property, only if $R_0 > 1$ the system evolves into a state where we have a stable and constant fraction of infecteds $x^\star$. This is called the endemic state. If for instance the basic reproduction number $R_0 = 4$ this implies that $x^\star = 0.75$ which would imply that 75% of the population is infectious.

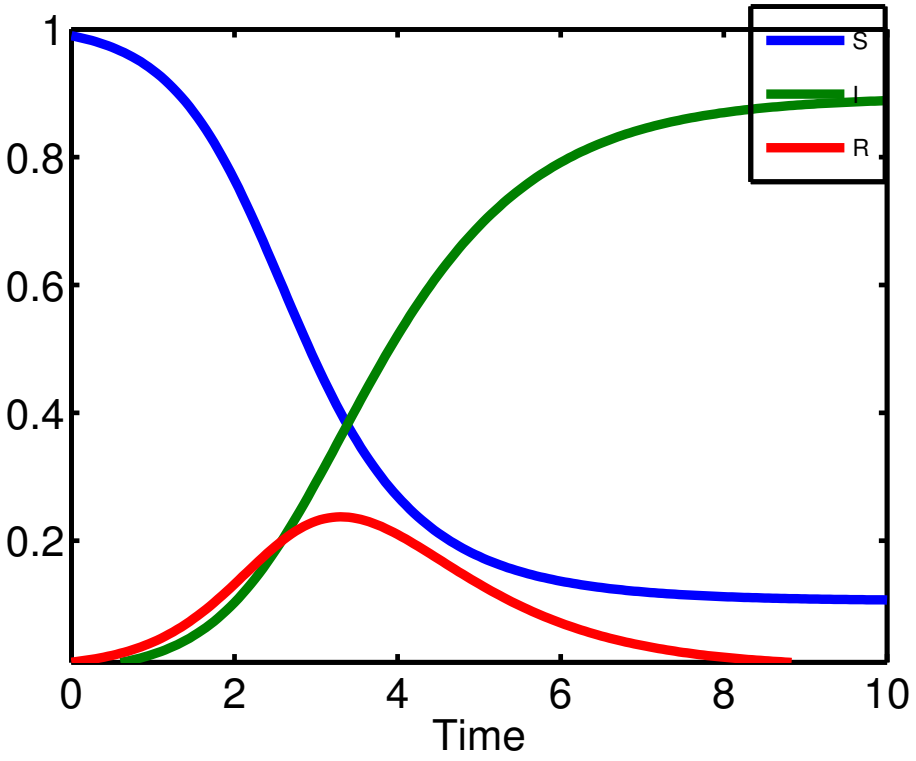


Figure 5.1: Dynamics of the SIR model for $R_0 = 2.5$.

5.2.2 The SIR model

This is the second most simple model to capture the time-course of an epidemic. It has different dynamic ingredients. Instead or going back to the susceptible class, infected become immune and go to a recovered class in which they cannot catch the disease a second time, so we have

$$S \to I \to R$$

which is why this model is called the SIR model. So we have the two equations

$$\begin{aligned} S + I &\xrightarrow{\alpha} 2I \\ I &\xrightarrow{\beta} R \end{aligned}$$

and the conservation of people says that

$$N = R + I + S.$$

We can do the same reasoning as above and we obtain two differential equations for S and I that read

$$\begin{aligned}\frac{dS}{dt} &= -\alpha SI/N \\ \frac{dI}{dt} &= \alpha SI/N - \beta I\end{aligned}$$

We do not need the ODE for R because we can compute it from $R = N - I - S$. Using relative variable $x = I/N$ and $y = S/N$ these become

$$\begin{aligned}dx/dt &= \alpha xy - \beta x \\ dy/dt &= -\alpha xy\end{aligned}$$

We immediately see that susceptibles can only decrease because $x, y \geq 0$ and $dy/dt \leq 0$. We can also see that if we start the system for $x(0) = \epsilon$ and $y(0) = 1 - \epsilon$ initially the infecteds will behave according to

$$dx/dt \approx (\alpha - \beta)x$$

which yields an exponential increase if $R_0 > 1$. It turns out the the nonlinear set of ODEs above cannot be solved analyitcally. But we can solve it in the computer to see what the behavior is. We see in Fig. that for R_0 the shape of $x(t)$ exhibits a typical epidemic peak. The larger R_0 the larger that peak. Note also that after the epidemic we always have a fraction of susceptibles that never caught the disease.

5.2.3 Stochasticity in the SIR model

One key drawback of the classes of models above is that they are deterministc, i.e. if we start with some initial condition, the system will always evolve the same way. This is certainly not the case for small populations where individual interactions shape the dynamics. One way to introduce stochasticity is this. Let"s think of the system evolving in a state space spanned by the dimensions S and I. At everypoint the system is going to be in a state given by

$$\mathbf{X} = (S, I).$$

Then reaction 1 induces a step in state space

$$(S, I) \rightarrow (S - 1, S + 1)$$

and reaction 2 induces a step in state space given by

$$(S, I) \rightarrow (S, I - 1).$$

If the system is in state (S, I) we assume that we cannot predict whether reaction 1 occurs or reaction 2 or no reaction at all. What we will make assumption on are the probabilities, or propensities of each reaction occuring. Let's denote the probability of reaction 1 by

$$w_+ \approx \Delta t \times \alpha \times I \times S/N$$

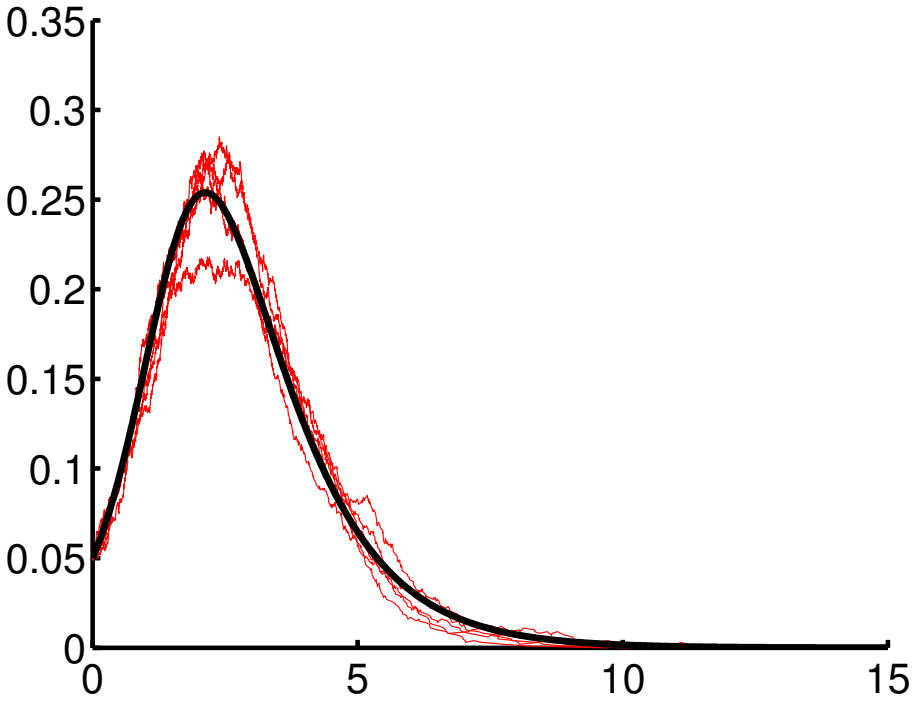


Figure 5.2: Stochastic SIR model.

and by

$$w_- \approx \Delta t \times \alpha \times I$$

the prob. that reaction 2 occurs, both in a small interval Δt. The probability that nothing occurs is

$$\begin{aligned} w_0 &= 1 - w_- - w_+ \\ &= 1 - \Delta t \times r(\mathbf{X}) \end{aligned}$$

where

$$r(\mathbf{X}) = \alpha IS/N - \beta I$$

is the propensity that either reaction 1 or 2 will occur. w_0 is the probability that nothing happens in the time interval Δt. So what's the probability that either reaction 1 or 2 occur in the time intervale $[T, T+\Delta t]$. We can split the time T into n intervales, $T = n\Delta t$. Then this probability

$$\begin{aligned} p(T,\mathbf{X}) &\approx w_0^n \Delta t\, r(\mathbf{X}) \\ &\approx (1 - \Delta t \times r(\mathbf{X}))^n \Delta t\, r(\mathbf{X}) \\ &\approx (1 - T/n \times r(\mathbf{X}))^n \Delta t\, r(\mathbf{X}) \end{aligned}$$

If we let $n \to \infty$ this becomes

$$p(T,\mathbf{X}) = r(\mathbf{X}) e^{-Tr(\mathbf{X})} dt$$

This means the times the system stays in state $\mathbf{X}$ is a random variable T with an exponential distribution given by the above equation. After this time T the system goes into one of the two new states with probability

$$p_+ = \frac{w_+}{w_+ + w_-} = \frac{\alpha IS}{\alpha IS + \beta NI}$$

and

$$p_- = \frac{w_-}{w_+ + w_-} = \frac{\beta NI}{\alpha IS + \beta NI}$$

So we can simulate the process the following way

1. We initialize the system in a state $\mathbf{X} = (S, I)$
2. we compute the total exit rate $r(\mathbf{X})$ for that state
3. we draw a random waiting time T from the exponential distribution and increment time by T
4. we pick one of the directions the system evolves along by $p_\pm$
5. we go back to 2.)

A simulation of such a stochastic SIR model is shown in Fig. 5.2.

5.2.4 Spatial models

The models above capture the dynamics in single populations in which we can assume that the probability of interaction is the same between any chosen pair of individuals. This is not very realistic because we all have a high probability of interacting with our friends neighbors and people in our neighborhood. The first type of models that try to account for this are spatial lattice models, most of them are two-dimensional. The nodes in the network are ordered on 2-d lattices and can have any of the three states S, I and R.

The dynamics is generated in a similar fashion as in the compartmental models above. on each lattice a node can make two transitions

$$S \rightarrow I$$

and

$$I \rightarrow S$$

for the SIS model. We assume that at every discrete time step n the probability that a node that is infected makes a transition to S is

$$0 \leq \beta \leq 1$$

Susceptible nodes make a transition to the infected state with a probability which is propotional to the parameter α and the fraction of infecteds in the neighborgood

$$p_i = \alpha \frac{n(I)}{k}$$

where k is the total number of neighbors and $n(I)$ the fraction of infecteds. This is very easy to simulate. There are two choice, either one updates all the nodes at once or randomly and sequentially.

5.2.4.1 SIR on a lattice

This can be done in a similar way. The only difference is that infecteds become recovered with probability β. Fig. 5.3

5.2.5 Generalizations to networks

Whatever works on a lattice also works on a network. The key generalization is the infection probability of a node. For every susceptible node i we say that the probability of infection is

$$p_i = \alpha \frac{n(I)}{k_i}$$

where again $n(I)$ are those nodes of the k_i neighbors that are infected.

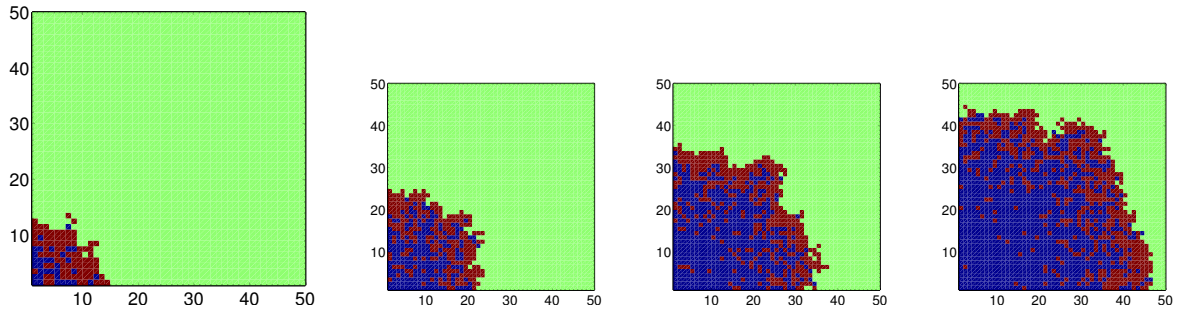


Figure 5.3: Three snapshots of the SIR model on a lattice. Green denotes susceptible, red infected and blue recovered.

5.3 Metapopulation Network Models

The above models are designed around the concept of nodes being single individuals and their links transmission pathsways between them. We also mentioned that a different type of network model exists in which each node is a population and the coupling the flux of individuals between these locations labeled m. So in addition to reactions

$$\begin{aligned} S_m + I_m &\xrightarrow{\alpha} 2I_m \\ I_m &\xrightarrow{\beta} R_m \quad (\text{or} S_m) \end{aligned}$$

we assume that individuals can move around between locations

$$X_n \xrightarrow{w_{nm}} X_m$$

where X_m is a place holder for any type of individual and w_{mn} is a rate of hopping between places. So before tackling this problem we need to understand movements on networks. The easiest type of random movement is a random walk, in which each individual hops between nodes of networks.

5.3.1 Diffusion and random on network

Let's assume the simplest scenario. We have $m = 1, ..., M$ locations and we consider an individual that hops between them, hopping from m to n only occurs if there's a link $A_{nm} = 1$. For simplicity we consider only symmetric networks, so being able to hop from n tpo m implies I can hop the other direction, too. The difficulty is again to translate topology into dynamics. We will see that there are different ways of doing things.

5.3.1.1 Model 1

Let's assume a random walker is placed with some probability on the set of nodes, say $p_0(n)$. And at discrete timesteps $t = 1, 2, ...$ the walker hops to one of the connected nodes. then the

probability of finding the walker at node m at time $t = 1$ and at n at time $t = 0$ is given by

$$p(m,1,n,0) = w(m|n)p_0(n)$$

where $w(m|n)$ is the conditional probability of jumping from n to m. We assume that $w(n|n) = 0$. The proability of the walker being at m at time $t = 1$ is given by

$$p_1(m) = \sum_n p(m,1;n,0) = \sum_n w(m|n)p_0(n)$$

We can write this as a matrix equation in which $p_1(m)$ is the m-th element of a vector $\mathbf{p}_1$so the eq. become

$$\mathbf{p}_1 = \mathbf{W}\mathbf{p}_0$$

where the matrix $\mathbf{W}$ has elements $W_{nm} = w(m|n)$ Likewise the probability of finding the walker at location n at time $t = 2$ is given by

$$\mathbf{p}_2 = \mathbf{W}p_1 = \mathbf{W}^2\mathbf{p}_0$$

and in general

$$\mathbf{p}_{t+1} = \mathbf{W}\mathbf{p}_t$$

It might occur that the probability that the walker is at a given location does not depend on time which implies that

$$\mathbf{p}^\star = \mathbf{W}\mathbf{p}^\star$$

so the stationar pdf $p^\star(m)$ is an eigenvector with eigenvalue 1 of the probability matrix $w(m|n)$. Not that since $w(m|n)$ is a probability it has to be normalized so it has to fullfill

$$\sum_m w(m|n) = 1,$$

the walker has to jump somewhere.

In the simplest model of a random walker on a network we assume that

$$w(m|n) \propto A_{mn}$$

which means that

$$w(m|n) = \frac{A_{nm}}{k_n}$$

this also in general implies that

$$w(m|n) \neq w(m|n).$$

Then the above equation becomes

$$p_{t+1}(n) = \frac{1}{k_n}\sum_m A_{nm}p_t(m).$$

and the equation for the stationary distribution is given by

$$k_n p^\star(n) = \sum_m A_{nm} p^\star(m).$$

We can check that this is this has a trivial stationar solution

$$p^\star(n) = \frac{1}{N}$$

so the walker is everywhere with equal probability.

How quickly does is that stationary distribution reached? Let's look at

$$\mathbf{p}_t = \mathbf{W}^t \mathbf{p}_0$$

We can diagonalize so

$$W = S^{-1} D S$$

in which D is diagonal and contains the eigenvalues. So

$$S p_t = D^t S p_0$$

The matrix D^t has diagonal elements λ_i^t wher λ_i is the i-th eigenvalue of W. It can be shown that the largest eigenvalue is 1, corresponding to the stationary solution and that all other eigenvalues are real and $\lambda_i < 1$, so in D^t the term that relaxes least fast is

$$e^{-t/\tau}$$

where

$$\tau = -\frac{1}{\lambda_{max}}$$

After this time, the proability has pretty much relatex to $p^\star$.

5.3.2 Solving random walks on a computer

There are two ways to simulate this type of hopping process on a network. Starting a a certain node n_0 we can compute the random sequence

$$n_0, n_1, ... n_t$$

All we have to do is at each location $n_t = m$ we determine the next location according to

$$w(m|n) = \frac{A_{mn}}{k_n}.$$

If we fix the position n_0 we have an initial probability distribution

$$p_0(n) = \delta_{n,n_0}$$

and if we want to compute $p_t(n)$ we use equation

$$\mathbf{p}_t = \mathbf{W}^t \mathbf{p}_0$$

So

$$\begin{aligned} p_t(n) &= \sum_m (W^t)_{nm} p_0(m) \\ p_t(n) &= \sum_m (W^t)_{nm} \delta_{m,n_0} \\ &= \left(W^t\right)_{n,n_0} \end{aligned}$$

5.3.3 Model 2

This however is not the only way to model hopping on a network. Let's look at a system in continous time. Let's assume that in a small time interval Δt a walker can jump from on location to another or remain at the location that it is at.The probability of finding a walker at location

$$p(n, t+\Delta t) = p(n,t) + \Delta t \left[\sum_m w(n|m) p(m,t) - w(m|n) p(n,t) \right]$$

the first term increases the probability because the walker could be coming from one of the locations m and the second term decreases the probablity because the walker could be leaving n to go somewhere else. So this yiels

$$\partial_t p(n,t) = \sum_m \left(w(n|m) p(m,t) - w(m|n) p(n,t) \right)$$

This is a master equation and $w(n|m)$ is the probability rate of going from m to n. Setting the left hand side to zero we can determine the stationary pdf by the equation

$$w(n|m) p(m) = w(m|n) p(n)$$

this is called detailed balance. in Equilbirum there's as much probability flux from m to n as the other way around. What's important here is that $w(n|m)$ is not a probability, but a proability rate. Hence, it does not have to be normalized. So let's assume the rate of hopping from m to n is constant and equal to γ if a link exists between n and m and zero otherwise. In this case we have

$$w(n|m) = \gamma A_{nm}$$

This also means that

$$w(n|m) = w(m|n)$$

which also means that

$$p(m) = p(n)$$

so the stationary distribution is constant, as before. The master equation becomes

$$\begin{aligned}\partial_t p(n,t) &= \gamma \sum_m A_{nm} p(m,t) - A_{mn} p(n,t) \\ &= \gamma \sum_m A_{nm} p(m,t) - k_n p(n,t) \\ &= \gamma \sum_m D_{nm} p(m,t)\end{aligned}$$

with

$$D_{nm} = A_{nm} - k_n \delta_{nm}$$

this matrix is called the network Laplacian. When we sum over it's rows we get zero

$$\sum_m D_{nm} = 0$$

We can again write the dynamics of $p(n,t)$ i vector notation and get

$$\partial_t \mathbf{p}(t) = \gamma \mathbf{D} p(t)$$

with the solution

$$p(t) = e^{\gamma \mathbf{D} t} \mathbf{p}(0).$$

One can show that the largest eigenvalue of $\mathbf{D}$ is zero, corresponding to the stationary solution, and the largest nonzero eigenvalue ios nevative. This largest eigenvalue determines the rate at which the equilibrium is reached and

$$\mathbf{p}(t) \approx e^{-\gamma |\lambda_1| t} \mathbf{p}(0)$$

The relaxation time of te random walk, the time an ensemble of walkers need to read everything in the network is given by

$$T = \frac{1}{\gamma |\lambda_1|}.$$

It seems a little suprising that a walker in equilibrium is as likely to be at a high degree node than at a small degree node. Because one would expect the walker to hit at high-degree node more freuqnetly than a node of low degree. This is true, but the walker doesn't spend much time on a hub. One can show from the master equation that the typical waiting time at a node is given by

$$T_n = \frac{1}{k_n}$$

which is small for nodes of high degree.

5.3.3.1 Random walks on weighted networks

Above we assumed that

$$w(n|m) = \gamma A_{nm}.$$

and that the links are unweighted. of course we can easily generaloze this to weighted network with elements W_{nm} and we could just define

$$w(n|m) = W_{nm}$$

Typically we do not have good data on the probability rates $w(n|m)$ in real mobiltiy patterns, what we do have is the equilbirium flux

$$F_{nm} = w(n|m)N_m = w(m|n)N_n$$

If we divide this by

$$\mathcal{N} = \sum_m N_m$$

we get

$$f_{nm} = w(n|m)p_m = w(m|n)p_n = f_{nm}$$

where N_m is the number of people in population m. Given $N_n$and F_{nm} we can compute $w(n|m)$, however. So if we define

$$w(n|m) = \gamma \frac{F_{nm}}{N_m}$$

and generate diffusion according to

$$\partial_t p_n = \sum_m w(n|m)p_m - w(m|n)p_n$$

we have a system with and equilibrium distribution

$$p_n = \frac{N_n}{\mathcal{N}}.$$

5.3.4 SIS Dynamics on a network

Let's now combine random walks on a network with epidemics or in general contagion phenomena. So in addition to the epidemic reactions we have a set of equations

$$S_n \xrightarrow{w_{mn}} S_m$$

and

$$I_n \xrightarrow{w_{mn}} I_m$$

What are the equations that govern these types more models. Let's assume that we have $m = 1,..M$ populations with N_m individuals. And let's assume that in each population we can describe the dynamics by ordinary SIS dynamics, i.e.

$$\begin{aligned} dI_n/dt &= \alpha \frac{S_n I_n}{N_n} - \beta I_n \\ dS_n/dt &= -\alpha \frac{S_n I_n}{N_n} + \beta I_n \end{aligned}$$

Now let's assume that in a small time interval Δt individuals are exchanged between places. Let's denote by $\Delta t \times F_{nm}$ the number of individuals that travel from m to n in a small time interval, so F_{nm} is the flux of individuals from m to n. If the overall population is in equilibrium, then we must have

$$F_{nm} = F_{mn}$$

This travel induces a change in the number of infecteds and susceptibles in population n.

$$\Delta I_n = \gamma \sum_m \Delta t \times \left[F_{nm} \times \frac{I_m}{N_m} - F_{mn} \times \frac{I_n}{N_n} \right]$$

Some leave and some come. So the above equations become

$$\begin{aligned} dI_n/dt &= \alpha \frac{S_n I_n}{N_n} - \beta I_n + \gamma \sum_m \times \left[F_{nm} \times \frac{I_m}{N_m} - F_{mn} \times \frac{I_n}{N_n} \right] \\ dS_n/dt &= -\alpha \frac{S_n I_n}{N_n} + \beta I_n + \gamma \sum_m \times \left[F_{nm} \times \frac{S_m}{N_m} - F_{mn} \times \frac{S_n}{N_n} \right] \end{aligned}$$

We can use relative concentrations

$$x_n = I_n/N_n, \quad y_n = S_n/I_n$$

to obtain

$$\begin{aligned} dx_n/dt &= \alpha x_n y_n - \beta x_n + \frac{\gamma}{N_n} \sum_m [F_{nm} x_m - F_{mn} x_n] \\ dy_n/dt &= -\alpha x_n y_n + \beta x_n + \frac{\gamma}{N_n} \sum_m \left[F_{nm} y_m - F_{mn} y_n \right] \end{aligned}$$

which we can also write as

$$\begin{aligned} dx_n/dt &= \alpha x_n y_n - \beta x_n + \frac{\gamma}{N_n} \sum_m F_{nm} (x_m - x_n) \\ dy_n/dt &= -\alpha x_n y_n + \beta x_n + \frac{\gamma}{N_n} \sum_m F_{nm} \left(y_m - y_n \right) \end{aligned}$$

bcause in equilibrium $F_{nm} = F_{mn}$. We can show that

$$\frac{d}{dt}(I_n + S_n) = 0$$

which means that $y_n = 1 - x_n$. So we have

$$dx_n/dt = \alpha x_n(1-x_n) - \beta x_n + \frac{\gamma}{N_n}\sum_m F_{nm}(x_m - x_n)$$

Now let's see if an epidemic can spread through the entire system. Obviously $x_m = 0$ for $m = 1, 2, ..., M$ is a stationary solution. We can assume that if we perturb the system a little away from the disease free state so if

$$x_m(0) \approx \epsilon$$

then the system is approx. linear

$$dx_n/dt \approx (\alpha - \beta)x_n + \frac{\gamma}{N_n}\sum_m F_{nm}(x_m - x_n)$$

We can write this as

$$dx_n/dt \approx \sum_m G_{nm}x_n$$

with

$$G_{nm} = \left[(\alpha - \beta) - \frac{\gamma}{N_n}\sum_m F_{mn}\right]\delta_{nm} + \gamma\frac{F_{mn}}{N_n}.$$

This is easier if we make use of

$$F_{mn} = w_{mn}N_n$$

so we've got

$$G = (\alpha - \beta)\mathbf{1} - \gamma\mathbf{D}$$

where

$$D_{nm} = \gamma w_{nm} - \phi_n\delta_n$$

and

$$\phi_n = \sum_m w_{mn}$$

5.4 Social Dynamics

5.4.1 Spread of rumours

The SIR model consists of the basic mechanism triggered by the two reactions

$$\begin{aligned} S + I &\xrightarrow{\alpha} 2I \\ I &\xrightarrow{\beta} R \end{aligned}$$

We can also think of this as a mechanism for rumour spreading. In fact this has been used in this conext. However, in a slightly modified version. We make the identification

- S: susceptible, i.e. the rumour is new to me

- I: active rumour spreader
- R: knows the information, but no longer is interested in spreading it

Then we have the following reaction for the transmission of the rumour

$$S + I \xrightarrow{\alpha} 2I$$

so the first reaction of the SIR model, but instead of the spontaneous transition to the passive state R we assume that this transition is also triggered by and interaction

$$\begin{aligned} I + I &\xrightarrow{\beta} I + R \\ I + R &\xrightarrow{\beta} 2R \end{aligned}$$

This doesn't make much of a difference one would expect. But let's see. Of course we still have conservation of individuals

$$N = I + S + R.$$

The change in each class in a small time interval is

$$\begin{aligned} \Delta S &\approx -\Delta t \alpha IS/N \\ \Delta I &\approx \Delta t \left[\alpha IS/N - \beta I^2/N - \beta IR/N\right] \\ \Delta R &\approx \Delta t \left[\beta I^2/N + \beta IR/N\right] \end{aligned}$$

which means we have a set of differential equations

$$\begin{aligned} dx/dt &= x\alpha y - \beta x^2 - \beta x(1-x-y) \\ dy/dt &= -\alpha xy \end{aligned}$$

which we can write

$$\begin{aligned} dx/dt &= (\alpha+\beta)xy - \beta x \\ dy/dt &= -\alpha xy \end{aligned}$$

The only difference to the ordinary SIR model is that we have a factor $\alpha+\beta$ in the first eq. instead of just α. We can expect the dynamics to look the same. In the SIR model we new, that an epidemic occured if the ratio

$$R_0 = \frac{\alpha}{\beta} > 1.$$

Here this turns into

$$R_0 = \frac{\alpha+\beta}{\beta} > 1$$

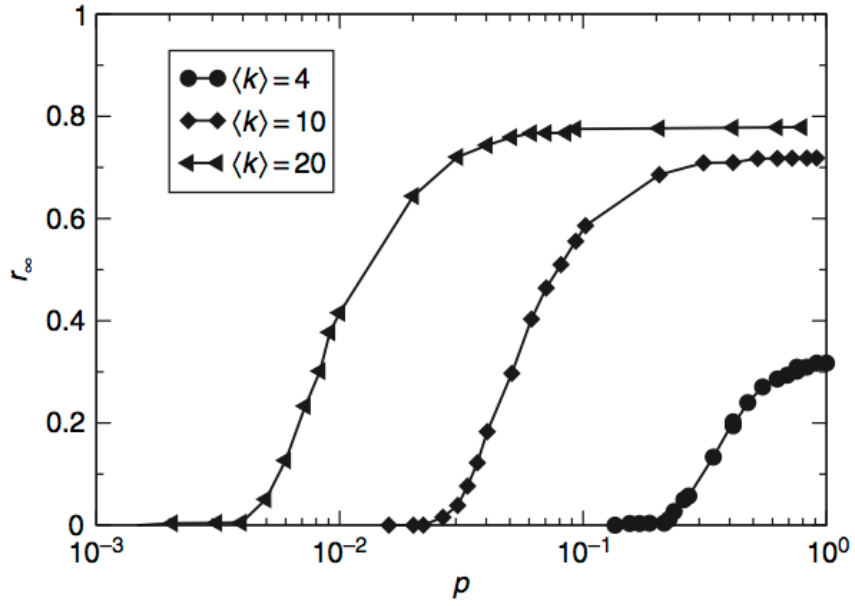


Figure 5.4: Rumors on a small network.

which means

$$\frac{\alpha}{\beta} > 0$$

so the rumour model exhibits an "epidemic" for arbitrarily small rate parameters α and β. Something we would also expect to happen in a network. One can also compute the final number of people how once spread the news, this is given by

$$r_\infty = 1 - e^{-(\frac{\alpha+\beta}{\beta})r_\infty}$$

5.4.1.1 Rumor spread in small world networks

We can use the equations

$$S + I \xrightarrow{\alpha} 2I$$

and

$$\begin{aligned} I + I &\xrightarrow{\beta} I + R \\ I + R &\xrightarrow{\beta} 2R \end{aligned}$$

and model those on a network in discrete time as usual. We make this as simple as possible:

1. We initiall infect a single node i
2. We pick one of its neighbors as given by the adjacency matrix, so

$$p_j = \frac{A_{ji}}{k_i}$$

3. then we have the choice:
   a) If node j is in the susceptible class, it becomes infected.
   b) If node j is infected or recovered, node i recovers
4. We pick another infected node at random and repeat the process

We would expect that this process spreads the rumor throughout a considerable fraction of the network. Note however that f we have a strong local clustering around a node, what can happen is that many nodes become infected and have infected neighbors, everytime we pick one of those, they recover and can exstinguish the rumor spread. On the other hand, if we have a network that is small world we can expect the rumour to spread far away. This is in fact seen.

In networks the process that does not have a threshold in a well mixed population, exhibits a clear cut phase-transition in small worlds. Fig. 5.4 illustrates the asymptotic value of r_∞ as a function of the rewiring probability p.

5.4.2 The Ising Model

One of the most important conceptual models in the context of social influence is the Ising Model. In this model, we assume that each node i in a network can be in one of two states

$$s_i = \begin{cases} 1 \\ -1 \end{cases}$$

These two states can represent two opposing political opinions. Now let's assume a network adjacency matrix quantifies the channels of interactions, $A_{ij} = 1$ means that nodes i and j can interact and according to the interaction change their state

$$s_i(t+1) \to -s_i(t)$$

based on their neighborhood. Let's assume that for each node, it is benefitial to adopt the opinion of its neighbors. Let's look at a pair s_i and s_j and the quantity

$$h_{ij} = -\frac{1}{2} s_i A_{ij} s_j.$$

Obviously, if these nodes are uncoupled this quantity is zero. If $A_{ij} = 1$ then

$$h_{ij} = \begin{cases} -1 & \text{if} \quad s_i = s_j \\ 1 & \text{if} \quad s_i = -s_j \end{cases}$$

Thus, if aligning s_i with s_j is benefitial, a negative h_{ij} is benefitial. Of course, node i is connected to many neighbors in general, and flipping it will align it with some and not others. But we can compute the net alignment by

$$h_i = -\frac{1}{2} s_i \sum_j A_{ij} s_j$$

We can rewrite this as

$$h_i = -\frac{1}{2}(k_+ - k_-)$$

where k_+ is the number of i's neighbors with $s_i = s_j$ and k_- with $s_j = -s_i$. Thus flipping i will induce a change in h_i of magnitude

$$\Delta h_i = -\frac{1}{2}(k_- - k_+) + \frac{1}{2}(k_+ - k_-) = -(k_- - k_+)$$

Thus, if

$$k_- > k_+$$

it is benefitial for i to switch because it lowers the overall agreement with i's neighborhood. The overall agreement in the entire network can be computed by

$$H = -\frac{1}{2} \sum_{ij} s_i A_{ij} s_j = -\frac{1}{2} \mathbf{s}^T \mathbf{A} \mathbf{s}$$

In physics this is the energy of a spin system that can model the behavior of magnetic spins. Clearly the lowest possible "energy" is when all spins are alighned in which case

$$H = -\frac{1}{2} \sum_{ij} A_{ij} = -L$$

where L is the number of links in the system.

We can now simulate this model on a computer by the following algorithm:

1. Pick a node at random
2. compute the change in "agreement" of i were to change it's state Δh_i
   a) if $\Delta h_i \leq 0$ flip the spin
   b) if $\Delta h_i > 0$ do not flip the spin
3. go back to 2.

This process will eventually go into the state that corresponds to full agreement in the set of nodes. How can we measure this? Simply by

$$M = \frac{1}{N} \sum_i s_i$$

so that $M = \pm 1$ if everyone agrees, and if the spins are random then $M \approx 0$. This model is not so interesting as such but becomes interesting in the following way:

Let's assume that each node has a probability to flip it's state, even if

$$\Delta h_i > 0$$

This is equivalent to going with the minority opinion. So we can set the system up

$$p(s_i \to -s_i) = \begin{cases} 1 & \text{if} \quad \Delta h_i \leq 0 \\ q_i & \text{if} \quad \Delta h_i > 0 \end{cases}$$

and we can chose

$$q_i = \exp(-\Delta h_i / T)$$

in which $T > 0$ is a parameter. If $T \to 0$ then $q_i = 0$ which is the case described above. If $T \gg \Delta h_i$ then $q_i \approx 1$ which means that I flip the spin with probability 1 as well. The question is, how does the overall opinion in the population look like as a function of the parameter T

$$M(T) = ?$$

This depends very much on the adjacency matrix A. In a regular two dimensional grid, the system exhibits a second order phase transition, for

$$T > T_c$$

we find

$$M \approx 0$$

but for $T < T_c$

$$|M| > 0$$

Can we compute whether a phase transition exists in complex networks? The general answer to this question is "no". We can do it numerically but alrealy the 2-d lattice is very complicated. A handwaving approximation suggests that the transition in the configurational model is

$$T_c \propto \frac{\langle k^2 \rangle}{\langle k \rangle}$$

so divergent in scale-free networks.

5.4.3 The Voter Model

Very much related to the Ising model is the voter model. Again, the state of a node is given by

$$s_i = \begin{cases} 1 \\ -1 \end{cases}$$

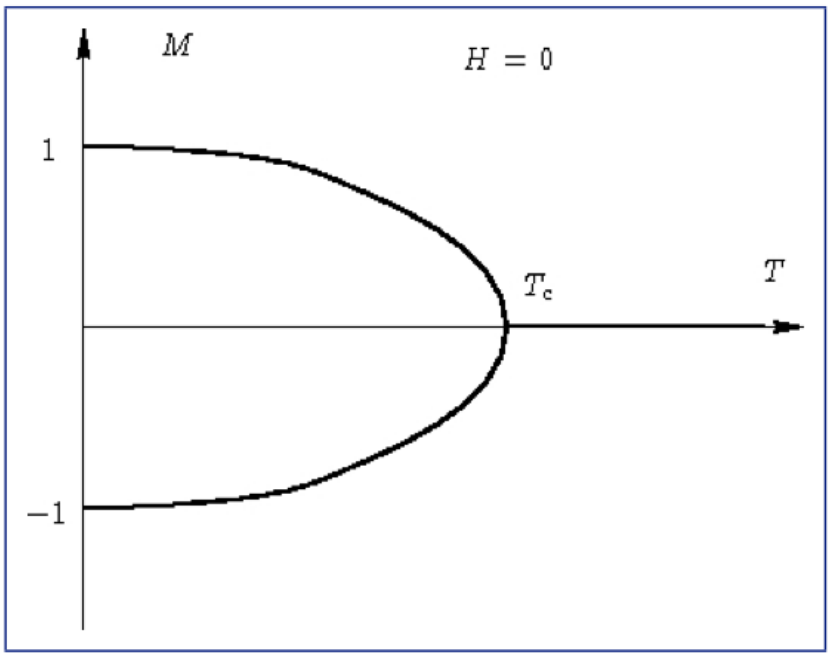


Figure 5.5: Phase transition of the Ising Model in a 2-d lattice.

and transitions can be made according to

$$s_i(t+1) \rightarrow -s_i(t).$$

In the voter model, each node picks one of it's neighbors at random and aligns its opinion to that node so if i is the node of interest and j one of i's neighbors then

$$s_i(t+1) = s_j(t).$$

Also in this model, the key question is if the entire system reaches a state in which consense exists throuought the entire population. It turns out that simulating the voter model on a 2-d lattice generates regions of consent that are separated by interfaces that separate these regions or so-called active bonds which are defined as having two nodes of opposing spin on each end. The number of interface decreases over time which can be understood in a one dimensional lattice. The borderes fluctuate in either direction until two borders meet and merge, getting rid of one border. Thus the number of active bond $n_A(t)$ decreases over time. In a one-dimensional lattice like this

$$n_A(t) \sim 1/\sqrt{t}.$$

In fact one can show but it is slightly involved that

$$n_A(t) \sim \begin{cases} t^{-1/2} & d = 1 \\ 1/\log t & d = 2 \\ a - bt^{-d/2} & d > 2 \end{cases}$$

which means that in more than 2 dimensions there will always be active links. What happens in a small world? Because of the small probability of having a link to a distant region, what can happen is that a region of one opinion and seed into a region of another opinion such that no overall consent is reached. The function $n_A(t)$ reaches a plateau.

5.4.4 The majority rule

This is very similar to the voter model, but involves groups of nodes. At each time step, on picks a group of r nodes and computes the opinion of that group. The group size is usually taken as random for instance taken from a binomial distribution, Poisson distribution.

$$q = \sum_{i=1}^{r} s_i$$

and then one updates all nodes in that group

$$s_i(t+1) = \begin{cases} 1 & \text{if} \quad q > 0 \\ 0 & \text{if} \quad q < 0 \end{cases}.$$

The groups are interpreted as discussion groups that form a consensus.

5.4.5 Neural networks: The Hopfield model

The Ising model can be used in the context of social networks in which the spins s_i represent the opinion polarity of a node s_i. The application range is much larger. In fact the concept was was of the first models in computational neuroscience and plays a role in one of the most fundamental models for memory.

In the Hopfield model wei have a set of neurons labeled i and they are interconnected by symmetric weighted links $w_{ij} = w_{ji}$ which resemble the synaptic input to from neuron i to j and vice versa. There are only two possible states for neurons

$$\begin{aligned} s_i &= 1 \quad \text{active} \\ s_i &= 0 \quad \text{inactive} \end{aligned}$$

The activity of a neuron at time-step $n+1$ is determined by the input that neuron gets from all the other neurons. This input is

$$h_i = \sum_j w_{ij} s_j$$

If this input exceeds a threshold, neuron i will become active so that

$$s_i(n+1) = \sigma\left[h_i - \theta_i\right]$$

where

$$\sigma(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}$$

and θ_i is the threshold of neuron i. It can be shown, that if we update all the neurons according to the rule given above, that the function

$$\begin{aligned} E &= -\frac{1}{2}\sum_{ij} s_i w_{ij} s_j + \sum_i \theta_i s_i \\ &= -\frac{1}{2}\mathbf{sWs} - \theta^T \mathbf{s} \end{aligned}$$

It can be shown that this function is non-increasing as afunction of time and is bounded from below, so the evolution dynamics always approaches a local minimum of E in which a subset of neurons are active. What we are saying is that

$$E(\mathbf{s}(n+1)) \leq E(\mathbf{s}(n))$$

The clue about the Hopfield model is that one can now chose or "learn" the appropriate weights between neurons, such that the network can recall specific patterns in response to an input. For example, let's define a state

$$x_i$$

to be a state to be recovered by the dynamics of the network. So we want for any initial condition $s_i(0)$ that

$$\lim_{n\to\infty} s_i(n) = x_i$$

This way we can say that the neural network recovered the memory which corresponds to the activity x_i. We can chose

$$w_{ij} = x_i x_j - x_i^2 \delta_{ij}$$

If we plug this into the energy we get

$$\begin{aligned} E &= -\frac{1}{2}\sum_{ij} s_i x_i x_j s_j + \sum_i x_i s_i + \sum_i \theta_i s_i \\ &= -\frac{1}{2}\left(\sum_i s_i x_i\right)^2 + \sum_i (\theta_i + x_i) s_i \\ &= -\frac{1}{2}(\mathbf{s}\cdot\mathbf{x})^2 + (\mathbf{s}\cdot\mathbf{x}) + \theta\cdot\mathbf{s} \end{aligned}$$

This has a minimum at

$$\mathbf{s} = \mathbf{x}$$

Let's say we want to store multiple patterns in the network. One way of doing this is by

$$w_{ij} = \frac{1}{M}\sum_\alpha x_i^\alpha x_j^\alpha - \left[\sum_\alpha x_i^\alpha x_i^\alpha\right]\delta_{ij}$$

We can try this in a compute and see how succesfull the network recovers a given memory state **x**.

5.4.6 Social Impact Theory

The psychological theory of social impact describes how individuals feel the presence of their peers and how they in turn influence other individuals. The impact of a social group on a subject depends on the number of individuals in the group, on their convincing power, and on the distance from the subject, where the distance may refer either to spatial proximity or to the closeness in an abstract space of personal relationships. The starting point is a network of N nodes. Each node i is characterized by an opinion

$$s_i = \pm 1$$

and by two parameters that estimate the strength of the nodes action its neighbors:

- persuasiveness p_i : capability to convince someone to change
- supportiveness q_i: capability to convince someone to keep their opinion.

These parameters are assumed to be random numbers, and introduce a disorder that is responsible for the complex dynamics of the model. The stength of coupling of nodes is given by a weighted network w_{ij}. Then we can fomulate the total influence experienced by node i as

$$I_i = \frac{1}{2}\sum_j P_j w_{ij}(1 - s_i s_j) - \frac{1}{2}\sum_j Q_j w_{ij}(1 + s_i s_j)$$

If $s_j = -s_i$ that means if i has the opposing opinion then $s_i s_j = -1$ and the term contributed to I_i by node j is $w_{ij}P_j$ if on the other hand $s_j = s_i$ the impact is $-w_{ij}Q_j$. This means the supporters try to decrease the social impact. I is thus a measure of influence to switch the dynamics. The dynamics of opinion is governed by

$$s_i(t+1) = -\sigma(s_i I_i)$$

For example if the opinion is $s_i = +1$ and $I_i > 0$, then $s_i(t+1) = -1$. If $I_i < 1$ then $s_i(t+1) = s_i$ in general $s_i(t+1) = s_i(t)$ if $I_i < 1$ and flipping occurs when $I_i > 1$.

5.4.7 Axelrod Model

In the Axelrod model each agent is endowed with a certain number F of cultural features defining the individual's attributes, each of those assuming any one of Q possible values. So for each node i we have $q_f(i) \in [1, Q]$. The Voter model is thus a particular case of Axelrod's model, with $F = 1$ and $Q = 2$. The model takes into account the fact that agents are likely to interact with others only if they already share cultural attributes, and that the interaction then tends to reinforce the similarity. The precise rules of the model are therefore the following:

- at each time step, two neighboring agents are selected i and j with $A_{ij} = 1$
- They interact with probability

  $$p(i,j) \propto \sum_{k=1}^{F} \delta_{q_k(i),q_k(j)}$$

  proportional to the number of features (or attributes) for which they share the same value.
- one of the features for which they differ is chosen, and one of the agents selected at random adopts the value of the other.

The local convergence rules can lead either to global polarization or to a culturally fragmented state with coexistence of different homogeneous regions. As F increases, the likelihood of convergence towards a globally homogeneous state is enhanced, while this likelihood decreases when q increases. Indeed, when more features are present (at fixed q), there is a greater chance that two neighboring agents share at least one of them, and therefore the probability of interaction is enhanced. At larger q instead, it is less probable for each feature to have a common value in two agents, and they interact with smaller probability. In the case of Watts–Strogatz (two-dimensional) small-world networks, the disorder is shown to favor the ordered (homogeneous) state. In other words, the existence of a culturally fragmented phase is hindered by the presence of hubs and is no longer possible.

5.4.8 Boolean Networks

One of the most powerful network dynamical systems are boolean networks, which are very much related to the Hopfield network of neurons. We will first discuss the general idea.

A boolean network consists of nodes labeled $i = 1, ..., N$ that can be in one of two binary states

$$s_i \in \{0, 1\}$$

The states of all the nodes is updated synchronously according to some rule which we can write as

$$\mathbf{s}(t+1) = \mathbf{F}(\mathbf{s}(t))$$

everything interesting is encoded in the function $\mathbf{F}$. A few things are are entirely independent an generally true for all $\mathbf{F}$:

- Because each state of a node is binary, the state of the system can be described by a binary sequence

  $$\mathbf{s} = 01101011001$$

  for example. And the entire state space is finite having

  $$\Omega = 2^N$$

  different states.

- Because F is a deterministic function if at some point in time, say t_0

  $$\mathbf{s}(t_0) = \mathbf{s}_0$$

  then

  $$\mathbf{s}(t_0 + 1) = F(\mathbf{s}(t_0)) = \mathbf{s}_1$$

  should the system evole to the state s_0 at some later time

  $$\mathbf{s}(t_0 + T) = \mathbf{s}_0$$

  then also

  $$\mathbf{s}(t_0 + T + 1) = \mathbf{s}_1$$

  This means however, that it will cycle through the intermediate states again, only to arrive at $\mathbf{s}_0$ again at time $2T$.

- The only other alternative is that

  $$s_1 = \mathbf{s}_0 = F(\mathbf{s}_0)$$

Consequenlty the system can only have cycles of fixed points. One may argue that T could be infinitely large, but since the state space is finite this cannot happen.

5.4.8.1 Setting up a boolean network.

Let's now focus on individual nodes i. Let's assume each nodes next state is determined by the input from $K < N$ other nodes that node i "reads". Now we have to specify what nodes node i reads, this is of course determined by the adjaceny matrix A_{ij} which for row i has only K nonzero values.

What is the total number of different inputs a node can get. Since the input nodes are also bindary, this number of different inputs is2^K. There's an easy way to do the bookkeeping for this. say we have $K = 3$. We first write down all the different input states and what node i does with it.1

$$\begin{aligned} 7:111 &\rightarrow 0 \\ 6:110 &\rightarrow 0 \\ 5:101 &\rightarrow 0 \\ 4:100 &\rightarrow 1 \\ 3:011 &\rightarrow 1 \\ 2:010 &\rightarrow 1 \\ 1:001 &\rightarrow 1 \\ 0:000 &\rightarrow 0 \end{aligned}$$

This specific rule is defined by its output on the $2^K = 8$ different inputs. The output is a bindary sequence

$$00011110 \doteq 30$$

How many such rules are possible for 2^K inputs? Well

$$2^{2^K} = 256$$

So, how we've got everything we need and we can write it in a table with N rows in which the first column is the index of node i, the next K columns the inputs to node i and the last column the number that corresponds the rule R_i which compute the output of i based on the input. Such a table could look like this:

node	input 1	input 2	input 3	rule
1	4	15	2	12
2	1	3	2	16
3	3	5	3	1
...	...	...	...	...

5.4.9 Dynamics

For smallish network, one can try to understand the dynamics of such as system by network visualization techniques. Instead of visualizing the network say of $N = 13$ nodes, one visualizes

the $2^N = 4096$ states the system can be in as nodes. Since under the dynamic rule every state gets mapped onto exactly one other state, one draws a directed link from every state to the next state.

Note although each state gets mapped onto exactly one state, multiple (or zero) states can get mapped onto a state. For every state we can compute the pre-image, the states that get mapped onto it and visualize them as branches. Since eventually the system has to go into a fixed point or a perioded cycle, this must be at the core of the treelike structure generated in this way.

5.4.9.1 Random boolean network dynamics

We can now see what random networks of this type do. We can fix $N = 13$ $K = 3$ chose each nodes input randomly form the 12 remaining nodes and choose among the $2^{2^3} = 256$ different rules randomly and see what the dynamics of such a system looks like.

5.4.10 Boolean Networks and gene regulation

A very similar idea is use to model gene regulation, basically the Hopfied network for genetics. In this model, each gene can be expressed of silenced, corresponding to the states $s_i = 1$ and 0, respectively. Genes regulate each other, they can either repress or promote expression of other genes. The simplest way to model this is by saying that

$$J_{ij} = \begin{cases} +1 & \text{if } j \text{ promotes } i \\ -1 & \text{if } j \text{ represses } i \\ 0 & \text{if } j \text{ has no impact on } i \end{cases}$$

Then we define the dynamics as

$$\begin{aligned} s_i(t+1) &= 1 \quad \text{if} \quad \sum_j J_{ij} s_j(t) > 0 \\ s_i(t+1) &= 0 \quad \text{if} \quad \sum_j J_{ij} s_j(t) < 0 \\ s_i(t+1) &= s_i(t) \quad \text{if} \quad \sum_j J_{ij} s_j(t) = 0 \end{aligned}$$

This is a special case of a boolean network.

5.4.10.1 Example Cell

see slides

5.5 Synchronization of oscillators

One of the most interesting and also most difficult to study network systems are those in which each node is an oscillator. And the key questions is if because of the coupling through network connections oscillators synchronize and under what conditions.

5.5.1 Oscillators

Let's look at some examples of oscillators. The simplest one is the harmonic oscillator, described by the following equation

$$\ddot{x} = -kx$$

This is a second order linear differential equation which we can write as

$$\begin{aligned} \dot{x} &= v \\ \dot{v} &= -kx \end{aligned}$$

We can easily see that

$$\begin{aligned} x(t) &= \cos \omega t \\ v(t)/\omega &= -\sin \omega t \end{aligned}$$

with

$$\omega = \sqrt{k}.$$

This is just a simple circular movement which we can also describe by

$$\theta(t) = \omega t$$

or

$$\dot{\theta} = \omega$$

This oscillator could for instance describe the dynamics of a clock with angular frequency ω.

5.5.2 Coupled Oscillators

Let's first understand what happens when two oscillators are coupled. Let's look at two oscillators that are governed by the following two equations

$$\begin{aligned} \dot{\theta}_1 &= \omega_1 \\ \dot{\theta}_2 &= \omega_2 \end{aligned}$$

Now let's assume that both oscillators interact, such that they are try to align the angular velocities, for instance if

$$\begin{aligned} \theta_1 > \theta_2 &: \quad \text{this will increase } \dot{\theta_2} \text{and decrease } \dot{\theta}_1 \\ \theta_1 < \theta_2 &: \quad \text{this will increase } \dot{\theta_1} \text{and decrease } \dot{\theta}_2 \end{aligned}$$

This is accomplished by

$$\begin{aligned} \dot{\theta}_1 &= \omega_1 + K \sin(\theta_2 - \theta_1) \\ \dot{\theta}_2 &= \omega_2 + K \sin(\theta_1 - \theta_2) \end{aligned}$$

where K is the coupling strength. The synchronmized state corresponds to

$$\begin{aligned}\theta_1(t) &= \omega t \\ \theta_1(t) &= \omega t + \alpha\end{aligned}$$

where ω is the frequency in the synchronized state and α is a potential phase difference. If we plug this into the equations of motion we get

$$\begin{aligned}\omega &= \omega_1 + K\sin\alpha \\ \omega &= \omega_2 - K\sin\alpha\end{aligned}$$

So that

$$\omega = \frac{1}{2}(\omega_1 + \omega_2)$$

and

$$\frac{1}{2K}(\omega_1 - \omega_2) = \sin\alpha < 1$$

This is only a solution if

$$K > \frac{1}{2}\delta\omega.$$

This means two things. For a given frequency difference we have to increase the coupling beyond a critical value. Right at the critical value the phase lag between the oscillators is $\alpha = \pi/2$. If the oscillators have the same natural frequency, the oscillate in phase. One can also show that this synchronisation is stable.

A generalization to multiple oscillators is straighforward, we can write

$$\dot{\theta}_i = \omega_i + \frac{K}{N}\sum A_{ij}\sin(\theta_j - \theta_i)$$

where now the matrix A_{ij} is the adjacency matrix and K determines the coupling strength between oscillators i and j and is generally taken to be symmetric. The problem with this model is that it is very difficult to show if synchronized states exist and whether they are stable. This model is called the Kuramoto model for coupled oscillators.

For a fully connected network $A_{ij} = 1$ one can show that all the oscillators phase synchronize if $\omega_i = \omega$. If the individual frequencies are drawn from a distribution $p(\omega)$, one can show that the system synchronizes if

$$K > \frac{2}{\pi p(\omega_0)}$$

where ω_0 is the mean of ω.