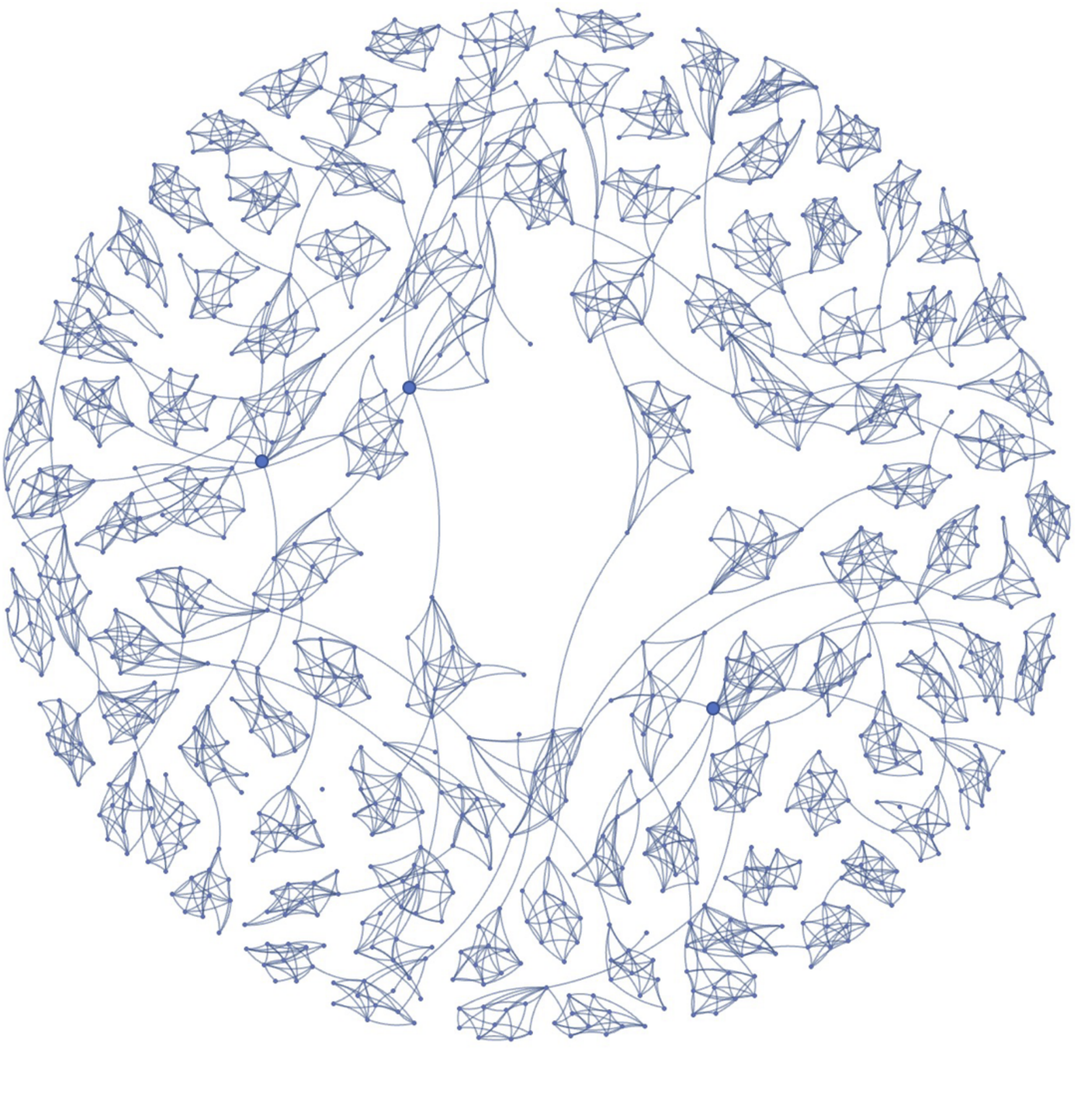
C5.4 Networks

Renaud Lambiotte and Pete Grindrod

November 5, 2022



Contents

Course Overview

Network Science provides generic tools to model and analyse systems in a broad range of disciplines, including biology, computer science and sociology. This course aims at providing an introduction to this interdisciplinary field of research, by integrating tools from graph theory, statistics and dynamical systems. Most of the topics to be considered are active modern research areas.

This year course has been altered to incorporate some new material on dynamically evolving networks and the analysis of scaling properties of growing networks.

Learning Outcomes: Students will have developed a sound knowledge and appreciation of some of the tools, concepts, models, and computations used in the study of networks. The study of networks is predominantly a modern subject, so the students will also be expected to develop the ability to read and understand current (2010-2022) research papers in the field.

Course Synopsis: roughly week by week
1. PRELIMINARIES. Probability theory; Renewal processes; Random walks; Power-law distributions; Matrix algebra; Markov chains.

2. BASIC STRUCTURAL PROPERTIES OF NETWORKS. Network Definitions; Degree distribution; Measures from walks and paths; Clustering coefficient; Centrality Measures; Spectral properties.

3. RANDOM GRAPH MODELS. Random Graphs; Erdös-Rényi random graph; Stochastic Block model; Configuration model; Small World model; Growing network with preferential attachment; Max likelihood calibration

4. COMMUNITY DETECTION: spectral method; Modularity

5. DYNAMICALLY EVOLVING NETWORKS: Markov chains of random graphs, application to triadic closure, via a mean field analysis

6. CONSENSUS DYNAMICS: Dynamics on networks

7. RANDOM WALKS : Discrete-time random walks on networks; PageRank; Models of epidemics

8. SCALING PROPERTIES OF GROWING NETWORKS: sequentially combining networks to form large growing networks.

1 PRELIMINARIES

1.1 Probability

Probability theory plays an important part in defining different classes of networks (and fitting networks to data), the dynamics of time-dependent networks, as well the dynamics of walks over networks. For those who have not studied probability in any great depth we begin with some basic foundations. Those who are interested further should consult [41], which is freely available online and is a *life-changing* book. Bayesian probabilities are subjective: we can each estimate our own priors based on our own knowledge and experience. In certain very well defined cases (such as the drawing playing cards from a pack, all coloured balls from an urn) we might all agree. Bayesian probabilities are usually conditional, and so they may be updated as more information becomes available to us (via Bayes' rule, given below). The important idea is that we all update the probabilities in a consistent way.

We deploy a conditional notation:

$$P(X|Y),$$

which denotes the probability that a proposition X is true given that the proposition Y is true. Sometimes Y may denote a list of possibly inter-dependent propositions and sometimes it may be null, in which case we just write $P(X)$, the probability that X is true in any and all circumstances (thatis, with no conditions).

In all our usage we will always have

$$P(X|Y) = 1 - P(\text{not}X|Y).$$

Here $\text{not}X$ is the logical opposite proposition of X.

Then we may combine successive conditional probabilities by multiplication

$$P(X \text{ and } Y) = P(X|Y).P(Y) = P(Y|X).P(X).$$

And so on...

$$P(A, B \text{ and } C) = P(A \text{ and } B|C).P(C) = P(A|B, C).P(B|C).P(C).$$

These two features of probability mat be derived from basic requirements of consistency [41].

Rearranging the alternative expressions for $P(X \text{ and } Y)$ we obtain **Bayes' rule**:

$$P(X|Y) = P(Y|X).P(X)/P(Y).$$

We usually say that $P(X)$ is the prior probability of X, and that $P(X|Y)$ is the posterior probability of X, the probability of X after we know that Y is true. In most applications Y will denote some set of observations (or some given data) while X will denote some proposition of interest.

If X and Y are independent then $P(Y|X) = P(Y)$. In that case $P(X \text{ and } Y) = P(X).P(Y)$. Strangely this multiplication of independent probabilities is usually taught first in schools.

Let $O(X|Y) = P(X|Y)/P(\text{not}X|Y)$ on $[0, \infty)$ denote the odds of X given Y, and so on. Then Bayes' rule can be rewritten as

$$O(X|Y) = \frac{P(Y|X)}{P(Y|\text{not}X)}.O(X).$$

This gets rid of the division by $P(Y)$: we can see how the prior odds are updated to become the posterior odds by the ratio of the probabilities of Y given X and not$X$. The ratio is usually called a Bayes factor.

Bayes' rule is often ignored by many (non-mathematical) professionals who should know better.

A person is the victim of knife crime in a New York neighbourhood [2]. The police fan out over the area and after one hour they arrest a man, taking him to the police station. He is found to have a knife hidden about his person. Does that fact make him more or less likely to be guilty? The Police Chief immediately lets him go free. Let X= "The suspect is the perpetrator", and $P(X) = p$ at the time of arrest (p is subjective). Let Y= "The suspect has a knife". About 10% of all men in the particular neighbourhood carry a knife, so $P(Y|\mathrm{not}X) = 1/10$. Only 1/1000 perpetrators would be crazy enough to still have a knife on them an hour after after committing a knife crime. So we have

$$O(X|Y) = \frac{1/1000}{1/10}.\frac{p}{(1-p)}. = \frac{1}{100}.\frac{p}{(1-p)}.$$

This Bayesian machinery works whether we are considering the probabilities of events with discrete possible outcomes or of continuous varying events, such as unknown real parameters.

In the former case we may have uncertainty over a **discrete random variable**, which might be an integer (the count of a given type of instance) or a categorical variable (such as gender, or eye colour) that is restricted to a discrete number of alternatives. The variable in question may take a number of mutually exclusive distinct values. Where there is a finite number of such alternatives, say $\{\alpha_1, \ldots, \alpha_K\}$, this is usually described by a multinomial distribution. This is a set of probabilities $\{p_k | k = 1, \ldots, K\}$ that sum to unity, and p_k is the probability that the variable is equal to α_k, has a type-k outcome. The distribution when there are just two types of outcome (heads or tails for some possibly biased coin, for example) is well known with probabilities p and $(1-p)$. Multiple experiments/observations give rise to combined outcomes distributed by the Binomial distribution as follows. If we have n independent experiments with exactly k type-1 results and $n-k$ type-2 results then the probability mass distribution is given by

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

because $p^k(1-p)^k$ is the probability of each specific sequence of outcomes occuring with exactly k type-1 results; and there are $\binom{n}{k}$ such sequences.

In the latter case we uncertainty over the value of some **continuous random variable**, θ, within some admissible set Ω, that may be analysed from observations (data). Suppose that our prior distribution for such a $\theta \in \Omega$ is some non-negative function $P(\theta) = f(\theta)$ and that we observe a piece of new data, D. Then we have

$$P(\theta|D) = P(D|\theta)f(\theta)/P(D) \propto P(D|\theta)f(\theta),$$

since $P(D)$ is just a normalising constant and plays no active roll in distributing θ. In fact we can always normalise $P(D|\theta)f(\theta)$ by making sure that the integral over admissible θ values is equal to unity. So the posterior distribution is given by

$$P(\theta|D) = \frac{P(D|\theta)f(\theta)}{\int_\Omega P(D|\theta)f(\theta)\, d\theta}. \tag{1}$$

The terms $P(D|\theta)$ should be thought of as a *model*. We must assume some model form for possible values of the data D given the possible values of θ. As modellers we will impose that model based on our experience and our own assumptions (our prior subjective pre-judgement).

Maximum likelihood estimation refers to finds the best value, the most likely value of θ, say θ^*, given the data D:

$$\theta^* = \text{argmax}_\theta P(D|\theta)f(\theta).$$

It provides one way, the mode, to summarise the posterior distribution for θ.

One might prefer the expected value (the mean) of the posterior distribution:

$$\langle\theta\rangle = \frac{\int_\Omega \theta P(D|\theta)f(\theta)\, d\theta}{\int_\Omega P(D|\theta)f(\theta)\, d\theta},$$

but this is polluted by the extreme values and the behaviour of the tails of the distribution. The mode is usually easily found using calculus (especially in the case of Gaussians) or by some optimisation via hill walking, gradient, methods.

We will aways use the notation $\langle . \rangle$ to denote the expected value of the expression inside. Higher moments are calculated similarly:

$$\langle\theta^k\rangle = \frac{\int_\Omega \theta^k P(D|\theta)f(\theta)\, d\theta}{\int_\Omega P(D|\theta)f(\theta)\, d\theta}, \; k = 1, 2, \ldots .$$

In particular we have the variance

$$\sigma_\theta^2 = \langle(\theta - \langle\theta\rangle)^2\rangle = \langle\theta^2\rangle - \langle\theta\rangle^2.$$

When we are confronted with empirical data, a crucial step is to find the parameter values that best reproduce the data, given a model. There exist many different approaches to parameter fitting. The use of maximum likelihood is surely the most popular.

Firstly though, in practice we do not always bother to normalise probability distributions. If we have a distribution we can always divide it by a constant so that is integrates or sums to unity (whenever we need to). It is only important that it is non negative. In Bayes' theorem we will usually ignore the denominator write (1) simply as

$$P(\theta|D) \sim P(D|\theta)f(\theta).$$

Even more permissively, sometimes we will consider distributions (especially as priors) which cannot be summed (they have an infinite mass): these are termed *improper distributions,* $f(\theta)$. Nevertheless they still possess maxima and must still be nonnegative, so maximum likelihood methods (usually gradient search numerical methods, or detremined via calculus etc) can be applied for such distributions.

Consider a sequence of m observations $\{x_i\}$ $(i = 1, 2, \ldots, m)$. Suppose that we are trying to fit a certain model for the distribution of possible values of the individual observations, say $q(x|\theta)$, whose parameter vector, denoted by θ, is assumed to have a support in Ω, and our prior distribution $f(\theta)$ is a constant, meaning any possible admissible value of $\theta \in \Omega$ is equally likely.

Maximum likelihood dictates that the parameter values are chosen to maximise the probability with which the model generates the observed data. To this end, we calculate $p(\theta|\{x_i\})$, which is related to $p(x_i|\theta)$ by Bayes' law, as above. We have the posterior,

$$p(\theta|\{x_i\}) = \frac{p(x_i|\theta)}{p(x_i)}.$$

In maximising this quantity, and thus estimate θ, we can simply ignore the denominator since By definition, the probability of observing certain data is fixed, and it does affect the optimisation of θ. Then, $p(\theta|\{x_i\})$ and $p(x_i|\theta)$, the likelihood of the data given θ, are proportional to each other, and the locations of their maxima coincide.

As an example, consider the model in which each individual x_i independently obeys the same **model** distribution, say $q(x|\theta)$. (As modellers we can propose or assert a model of our choice give our experience and knowledge.) Then **likelihood of the data**, $\mathcal{L}$ is given by

$$\mathcal{L} = \prod_{i=1^m} q(x_i|\theta).$$

Note (as we saw earlier) the multiplication of probabilities can only be valued when each successive observation is independent of the others.

In almost all practical circumstances we take logs and rather than maximise $\mathcal{L}$ we will maximise $\log \mathcal{L}$:

$$\log \mathcal{L} = \sum_{i=1^m} \log(x_i|\theta).$$

This is because when we try to multiply hundreds or thousands of probabilities together we may get sufficiently small num bers to be below machine accuracy, since the largish data sample outcomes are more and more diversified and thus all occurrences (even at the maximum likelihood) become unlikely. If in doubt, then, one is always advised to take logarithms; and one should avoid models for which $q(x|\theta) = 0$ anywhere relevant.

1.2 Matrix Algebra

We begin with some basic considerations from linear algebra.

A self-adjoint matrix, or Hermitian matrix, A, is such that $A = A^*$, the complex conjugate transpose of A. If A is real then this means $A = A^T$. Every self-adjoint matrix is a normal matrix, meaning that is diagonalisable. Indeed the spectral theorem says that any real self-adjoint matrix can be diagonalised by a unitary matrix: $U^T.A.U$ is diagonal, where U is unitary, and that the resulting diagonal matrix has only real entries. This implies that all eigenvalues of a self-adjoint matrix are real, and that it has a full set of linearly independent eigenvectors.

Exercise Show that all of the eigenvalues of a self-adjoint matrix, A are real.

Non-negative matrices play a very important role within many theories including network theory, unsupervised discrimination, mathematical economics and Markov chain theory. Nonnegative symmetric matrices are especially interesting since their spectrum is real. Many courses on linear algebra do not emphasise the properties of nonnegative matrices, but within analytics of many areas of business they occur quite often.

The **Perron-Frobenius theorem** gives us a ready way to estimate the spectral radius of nonnegative matrices and other things besides. It was proved by Oskar Perron and Georg Frobenius and says that a real square matrix with strictly positive entries has a unique largest real eigenvalue and that the corresponding eigenvector has strictly positive components, and it also asserts a similar statement for certain classes of nonnegative matrices, called irreducible nonnegative matrices.

A non-negative matrix A is irreducible if for every pair of indices i and j, there exists a natural number k, depending on (i, j), such that $(A^k)_i j > 0$. If A is real and $A > 0$ then it is trivially irreducible (we will see later what this condition means when A is an adjacency matrix). We state the following standard result without proof.

Perron-Frobenius Theorem Let A be an irreducible non-negative $n \times n$ matrix with spectral radius $\rho(A) = r > 0$. Then the following statements hold.
a) r is an eigenvalue of the matrix A, called the Perron-Frobenius eigenvalue.
b) The Perron-Frobenius eigenvalue, r, is simple. Both right and left eigenspaces are one-dimensional.
c) A has a left eigenvector and a right eigenvector with eigenvalue $r$whose components are all positive.
d) The only eigenvectors whose components are all positive are those associated with the eigenvalue r.
e) The Perron-Frobenius eigenvalue is bounded above and below by the maximum and minimum row sums of A respectively:

$$\min_{i=1,..,n} \sum_{j=1}^{n} A_{ij} \leq r \leq \max_{i=1,..,n} \sum_{j=1}^{n} A_{ij}.$$

Similarly r is bounded above and below by the maximum and minimum column sums of A.

Exercise. Suppose an $n \times n$ matrix A is nonnegative and the spectral radius of A is given by the Perron-Frobenius eigenvalue, r. Let $\alpha \in (0, r)$. Then consider

$$(I - \alpha A)^{-1}.$$

Show that if this matrix is strictly positive then A is irreducible. Is the converse true? Show that if the matrix

$$\exp(A) = I + A + A^2/2! + A^3/3! + \dots$$

is strictly positive then A is irreducible. Is the converse true?

The **singular value decomposition** of a matrix is an extremely useful factorisation of a general matrix that finds uses in all sorts of fields: matrix approximation, the definition of pseudo inverses, total least squares, and many other applications including signal processing and state space embedding methods.

Let M denote an $m \times n$ matrix whose entries are real or complex. The singular value decomposition of M is a factorization of the form

$$M = U\Sigma V^*,$$

where U is an $m \times m$ unitary matrix; Σ is an $m \times n$ diagonal matrix with nonnegative real numbers on the diagonal; and V^* denotes the conjugate transpose of the $n \times n$ unitary matrix V.

The diagonal entries of Σ are known as the singular values of M. A common convention is to list the singular values in descending order.

The m columns of U and the n columns of V are called the left-singular vectors and right-singular vectors of M, respectively. The ideas of singular value decomposition and the eigenvalue decomposition are closely related. In particular, we have both

$$MM^* = U(\Sigma\Sigma^*)U^*, \;\; M^*M = V(\Sigma^*\Sigma)V^*,$$

and these relations describe eigenvalue decompositions. Thus the left-singular vectors of M are eigenvectors of MM^*, and right-singular vectors of M are eigenvectors of M^*M. Hence the non-zero singular values of M (found on the diagonal entries of Σ) are the square roots of the non-zero eigenvalues of both M^*M and MM^*.

The singular value decomposition can define the pseudo-inverse of a matrix. Indeed, the pseudoinverse of the matrix M with singular value decomposition is $M^+ = V\Sigma^+U^*$ where Σ^+ is the pseudonverse of Σ, formed by replacing every non-zero diagonal entry by its reciprocal and transposing the resulting matrix. Working numerically one only "inverts" those singular values greater than some small tolerance.

The singular value decomposition can be applied to any $m \times n$ matrix, whereas eigenvalue decomposition can only be applied to certain classes of square matrices.

In the special case that $m = n$ and M is a normal matrix, the spectral theorem says that it can be unitarily diagonalised using a basis of eigenvectors, so that it can be decomposed UDU^*, with a unitary matrix U and a diagonal matrix D. Thus if M is also positive semi-definite, then the eigenvalue decomposition is also a singular value decomposition. Yet the eigenvalue decomposition and the singular value decomposition will differ for all other matrices.

Exercise
Suppose A is normal and invertible. Then there is a unitary U such that $A = U\Lambda U^T$ and Λ is diagonal containing the eigenvalues of A. Let $f : \mathbb{R} \to \mathbb{R}$ be any function that is well defined at all of the eigenvalues of A. Define $f(A) = Uf(\Lambda)U^T$ where $f(\Lambda)$ is diagonal; with f applied to each corresponding element of Λ.

a) Show that if Q is any polynomial

$$Q(x) = q_0 + q_1x + ... + q_mx^m$$

then

$$Q(A) = q_0I + q_1A + ... + q_mA^m.$$

b) Similarly show that

$$Q(A - I) = q_0I + q_1(A - I) + ... + q_m(A - I)^m = UQ(\Lambda - I)U^T,$$

and

c) that

$$Q(A)^{-1} = UQ(\Lambda)^{-1}UT.$$

Exercise
Suppose A is normal and its spectral radius is $\rho(A) < 1/\alpha$ for some $\alpha > 0$. Then consider $(I - \alpha A)^{-1} = U(I - \alpha\Lambda)^{-1}U^T$. Show that this is the geometric series $S = I + \alpha A + \alpha^2A^2 + \alpha^3A^2 + \dots$

Suppose that A is a real, non-negative, normal, $n \times n$ matrix and let $s = (1, 1, \dots, 1)^T \in \mathbb{R}^N$. Then the vector $As = (d_1, d_2, ..., d_n)^T$ contains the row (and hence the column) sums of A. Let $D = \mathrm{diag}(d_1, d_2, ..., d_n)$ be the diagonal matrix of A's row sums.

The (combinatorial) **Laplacian** associated with A, denoted by L, is the symmetric matrix

$$L = D - A.$$

Note the sign convention here: it is inherited from that of the discrete graph Laplacian (continuum mechanicists might well wish to write L as the negative of this).

Then, by design, we have

$$L.\mathbf{s} = 0,$$

and for all $\mathbf{w} = (w_1, \dots, w_n)^T \in \mathbb{R}^n$ we have the quadratic form:

$$\mathbf{w}^T.L.\mathbf{w} = \frac{1}{2}\sum_{i,j=1}^{m}(w_i - w_j)^2 A_{ij}. \tag{2}$$

Please prove this. We have

$$\mathbf{w}^T.L.\mathbf{w} = \sum_{i=1}^{n} d_i w_i^2 - \sum_{i,j=1}^{n} A_{ij} w_i w_j = \sum_{i,j=1}^{n} A_{ij} w_i^2 - \sum_{i,j=1}^{n} A_{ij} w_i w_j.$$

But, using the symmetry of A, we have the first term,

$$\sum_{i,j=1}^{n} A_{ij} w_i^2 = \frac{1}{2}\sum_{i,j=1}^{n} A_{ij}(w_i^2 + w_j^2).$$

Hence the result.

Thus L is always positive-semidefinite, and if $\mathbf{w}$ is any eigenvector of L corresponding to the zero eigenvalue of L then its components must be the same for any pair (i, j) where $A_{ij} > 0$.

1.3 Markov chains

Markov chains are stochastic dynamics on n states in discrete time. A state may be the position in a network having n vertices such that the process represents a random walk on the network. Alternatively, a state may be the number of infected people, between 0 and $n - 1$, in a structureless population of $N - 1$ individuals. In both cases, we number the states as $1, 2, ..., n$. The state at time t. $(t = 0, 1, \dots)$, which is a random variable, is denoted by X_t .

In a stochastic process on N states in general, state X_{t+1} may depend on all preceding states (the full history) of the dynamics, i.e., $X_0, X_1, ..., X_t$. Under the Markov assumption, the conditional probability to observe a state at time $t + 1$ only depends on the state at time t. Such a discrete stochastic process is called the **Markov chain**.

Among the class of Markov chains, we are often interested in the stationary ones, in which the conditional state-transition probability does not depend on t: it has fixed (matrix of) transition probabilities,

$$p(X_{t+1} = j | X_t = i) \equiv T_{ij}.$$

Processes satisfying both properties, Markovianity and stationarity, are called stationary Markov chains. Because any realisation of the process visiting state i must go somewhere, including itself, at the next time step, we obtain

$$1 = \sum_{j=1}^{n} T_{ij}.$$

A stationary Markov chain is fully described by an initial state and an $n \times n$ transition matrix $T = (T_{ij})$. The probability that state i is visited at time t, denoted by $p_y(t)$, evolves according to

$$p_j(t+1) = \sum p_i(t) T_{ij} \ \ (1 \leq j \leq N).$$

It should be noted that $\sum_{i=1}^{n} p_i(t) = 1$ for any t, if the initial condition is properly normalised. The evolution equation may be compactly rewritten in vector form as

$$p(t+1) = p(t)T,$$

where $p(t)$ is the row vector $(p_1(t), \ldots, p_N(t))$.

We also have

$$p(t) = p(0)T^t. \tag{3}$$

A Markov chain is composed of different types of states. By definition, the process does not escape from an absorbing state once it has been reached. State i is absorbing if and only if $T_{ii} = 1$, which implies that $T_{ij} = 0$ for any $j \neq i$. A group of states forms an ergodic set if it is possible to go from i to j for any states i and j in the set and if the process does not leave the set once the process has reached it. An absorbing state is thus an ergodic set composed of a single state. Finally, a state is called a transient state if it is not a member of an ergodic set.

We denote the non-negative stationary density by $p^* = (p_1^*, \ldots, p_n^*)$, where $p_i^* = \lim_{t\to\infty} p_i(t)$ for $1 \leq i \leq n$: clearly we must have

$$p^* = p^*T.$$

Therefore, the stationary density is the left eigenvector of T with eigenvalue equal to unity. If the entire set of the n states is ergodic, T is guaranteed to have such an eigenvalue of unity: p^* is simply the Perron-Frobenius left eigenvector and 1 is the Perron-Frobenius eigenvalue. This observation is consistent with the fact that all elements of the Perron-Frobenius vector are positive. In addition, the discrepancy of $p(t)$ from p^* decays exponentially as $\propto |\lambda_2|^t$, where λ_2 is the second largest eigenvalue of T in terms of the modulus. In words, the second largest eigenvalue governs the relaxation time of the iterate. More generally, the speed of convergence is determined by the difference or ratio between λ_2 and $\lambda_{\max}$, with the latter being equal to unity in the current case. Therefore, we often call $1 - \lambda_2$ the spectral gap. A Markov chain with a large spectral gap converges rapidly.

Markov chain theory also allows us to answer other types of questions. For example, how long on average do the dynamics need to reach a certain state? What is the probability of ending in a certain absorbing state, depending on the initial condition?

1.4 Renewal Processes

Let us consider a system where events take place in a discrete and apparently random fashion. Those events may be emails arriving in a mail box, or atoms colliding in a gas. Such systems are often modelled, as a first order approximation, by a **Poisson process**, also called the homogeneous Poisson process. The Poisson process assumes that the events are independent of each other, that the rate at which the events take place is constant over time and that time is continuous. These assumptions are often violated in empirical data. For instance, in the case of emails, their reception certainly depends on the time of the day and on the day of the week. In addition, emails are often not independent processes; an email may trigger a discussion thread between two users, causing a cascade of emails. Yet, the Poisson processes are advantageous in their simplicity, which allows us to exactly calculate their properties and make them serve as a baseline model.

The Poisson process is defined as follows. Consider a time window of duration Δt and the probability q that an event takes place within time Δt. By definition, the event rate is given by $\lambda = q/\Delta t$. A Poisson process is specified by the rate $\lambda$for infinitesimally small Δt. For λ to be well-defined, $q \to 0$

must be satisfied as $\Delta t \to 0$. Consistent with this requirement, we do not allow multiple events to occur in a time window when Δt is sufficiently small.

We derive two key properties of Poisson processes.

(1) **The distribution of inter-event times**, i.e., time between consecutive events: Let $p(n,t)$ be the probability of observing n events in time window $[0,t]$. By definition, for small Δt, we have

$$q = p(1, \Delta t) = \lambda \Delta t,$$

and

$$1 - q = p(0, \Delta t) = 1 - \lambda \Delta t.$$

For any $n \geq 1$, we obtain

$$p(n, t + \Delta t) = p(n,t)(1 - \lambda \Delta t) + p(n-1, t)p(1, \Delta t).$$

This holds true because, if there are$n$ events in time window $[0, t + \Delta t]$, either there are n events in $[0,t]$ and no event in $[t, t+\Delta t]$, or there are $n-1$ events in in $[0,t]$and one event in $[t, t+\Delta t]$,. In the limit as $\Delta t \to 0$ we have

$$\frac{dp(n,t)}{dt} = \lambda p(n-1,t) - \lambda p(n,t). \tag{4}$$

For $n = 0$, $p(0,t)$ is the probability that no event occurs in $[0,t]$, and so we have $p(0,0) = 1$, so we obtain

$$\frac{dp(0,t)}{dt} = -\lambda p(n,t),$$

so

$$p(0,t) = e^{-\lambda t}.$$

The probability that the first event occurs in $[t, t+\Delta t]$ is given by $p(0,t) - p(0, t+\Delta t)$. This implies that the inter-event time between two consecutive events, denoted by τ, is distributed according to the density

$$\psi(\tau) = -\frac{dp(0,\tau)}{d\tau} = \lambda e^{-\lambda \tau}.$$

So the inter-event time of a Poisson process is distributed according to this exponential distribution.

The mean inter-event time is given by

$$\langle \tau \rangle = \int_0^\infty \tau \psi(\tau) d\tau = \frac{1}{\lambda}.$$

In Poisson processes, different interevent times τ are independent of each other because event times before the last event time do not affect the time to the next event since then. This property is called the renewal property of a Poisson process. Poisson processes satisfy a stronger property, i.e., having no memory in the sense that

$$p(\tau > t_1 + t_2 | \tau > t_2) = p(\tau > t_1).$$

This indicates that the length of time, t_2, for which we have waited without an event does not affect the time of the next event. The time to the next event starting from $t = t_2$, that is t_1, is independent of t_2 and obeys $\psi(t_1)$.

(2) **The distribution of the number of events** observed within a given time window. Using Eq. (4) recursively, with $p(n.0) = 0$ for $n \geq 1$, we obtain

$$p(n,t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}.$$

Therefore, the probability of observing n events in $[0,t]$ obeys the Poisson distribution with mean and variance equal to λt. The Poisson distribution is a limiting case of the binomial distribution when the number of trials is very large and the expected number of successes remains fixed. This interpretation is consistent with the discrete-time formulation of the Poisson process because in $[0,t]$, there are $t/\Delta t$ trials in each of which an event occurs with small probability q. Therefore, the number of events in $[0,t]$ is distributed according to the binomial distribution whose mean is equal to $(t/\Delta t) \times q = \lambda t$.

1.5 Random walks and diffusion in discrete time

The Poisson processes provide a basic model for modelling temporal events, when random events take place. Random walk processes are its counterpart for modelling trajectories in space: when and where random events take place. Random walk processes are a standard tool to emulate diffusion on networks and also to extract information from the structure of networks, as we will show later. In this section, we derive some basic properties of random walk processes in their simplest setting, when they take place on a one-dimensional space (the real line) in discrete time.

In each discrete time step, a walker performs a jump whose length and direction are random variables. The probability density of transition is denoted by $f(r)$. In other words, the probability that the walker located at x arrives in the interval $[x+r, x+r+\Delta r]$ in one jump is equal to $f(r)\Delta r$. The normalisation condition is given by $\int_{-\infty}^{\infty} f(r)dr = 1$.

Our aim is to derive the density of the probability density that the walker is located at x after $t$steps, denoted by $p(x;t)$. Under the assumption that jumps are inde-pendent events, we obtain the following master equation:

$$p(x;t) = \int_{-\infty}^{\infty} f(x-x')p(x',t-1)dx'$$

because the probability of visiting x at time t is the prob-ability of having visited x' at time $t-1$, and performing a jump of displacement $x - x'$.

The Fourier transform transfers a convolution into a product. So we have

$$\hat{p}(k;t) = \hat{f}(k)\hat{p}(k;t-1),$$

where

$$\hat{g}(k) = \int_{-\infty}^{\infty} g(x)e^{-ikx}dx$$

denotes the Fourier transform of the function $g(x)$.

If the walker known to be at $x = 0$ at $t = 0$ then we have $p(x;0) = \delta(x)$, the Dirac distribution; and so $\hat{p}(k;0) = 1$. Putting this all together, we have

$$\hat{p}(k;t) = [\hat{f}(k)]^t.$$

Using the inverse Fourier transform, the formal solution of the random walk in the time domain is given by

$$\hat{p}(x;t) = \frac{1}{2pi} \int_{-\infty}^{\infty} [\hat{f}(k)]^t e^{-ikx}dk,$$

This solution depends on the function, $\hat{f}(k)$. However, the asymptotic behaviour of the random walk as t grows only depends on some of its properties.

When the first two moments of the structure function are finite (the mean and variance of f), the solution converges to the Gaussian profile

$$p(x;t) = \frac{1}{(2\pi Dt)^{1/2}} e^{-\frac{(x-vt)^2}{4Dt}},$$

with a variance controlled by a constant D growing linearly in time.

Alternatively the appearance of a Gaussian distribution is anticipated by the use of the Central Limit Theorem, since the walk is made up of the sum of a number of independent steps, each drawn from a distribution, f, itself having a finite variance.

Spatial processes that yield distributions having a variance which grows linearly in time, and have no memory, are all general forms of diffusion.

If f has no finite variance (it is infinite) the spatial process is not diffusive: large jumps are possible (F might be a Cauchy distribution for example) . In that case a *Lévy flight* is often the consequence, giving a distribution $p(x;t)$ that has a fat and growing tail. These processes are often observed in the analysis of stock market equity data movements and that of other financial instruments.

1.6 Power-law distributions

We have seen the emergence of two types of distributions in stochastic processes, the exponential distribution in the case of Poisson processes, and the Gaussian distribution in the case of random walk processes. Another type of distribution, the power-law distribution, plays a central role in network science and in the theory of complex systems in general.

In this section we overview properties of power-law distributions and raise some flags in order to properly use them when modelling complex systems.

We explain a power-law distribution for continuous variables, keeping in mind that most of the observations generalise to the case of discrete variables. Consider the Pareto distribution given by

$$p(x) = Cx^{-\alpha} \;\; (x > x_{\text{min}}),$$

where $\alpha > 1$ is the power-law exponent of the distribution, $x_{\text{min}} > 0$ is the minimum value taken by the random variable and $C = (\alpha - 1)x^{\alpha-1}$ is the normalisation constant, so that

$$\int_{x_{\text{min}}}^{\infty} p(x)dx = 1.$$

Other more general power-laws distributions are, by definition, those that are asymptotically (for large x) the same as this up to a normalisation constant.

Power-law distributions mainly differ from the exponential and Gaussian distributions by the significant mass of probability carried by their "fat" tail, at the large values of x. The exponential and Gaussian distributions have a characteristic scale such that the probability of observing instances many times larger than this scale is negligible. In contrast, under a power-law distribution, a vast majority of instances exhibits small values while few but non-negligible instances produce very large values.

Power-law distributions are associated with a broad heterogeneity in the system and are said to have a fat or long tail, because the tail of the distribution is much more populated than in exponential-like distributions. Power-law distributions are typically found in the wealth of individuals, populations of cities, the frequency of words in text, sales of books and music, citations that a scientific paper receives and so forth. This is summarised in many applications in [12].

Since the advent of the Pareto distribution, and the associated Zipf's law, power-law distributions have been studied over a century. We stress that fat tails are also present in distributions with out a power-law tail. Examples include stretched exponential distributions and log-normal distributions.

The moments of power-law distributions are given by

$$\langle x^\beta \rangle = \int_{x_{\min}}^{\infty} x^\beta p(x) dx = \frac{\alpha - 1}{\alpha - 1 - \beta} x_{\min}^{\beta}.$$

This moments for which $\beta > \alpha - 1$ are divergent and do not exist. For example, the sample mean for the power-law distribution with $\alpha = 2$ diverges as we accumulate samples.

The Cauchy distribution (occurring for Levy flight spatial processes - see the box below) is of the form $\sim 1/(1 + x^2)$ and is asymptotic to $p(x)$, above, where $\alpha = 2$. This it has no expected value and no finite variance. Hence if individual independent samples from the Cauchy distribution are summed (aggregated) to model a memoryless stochastic spatial process, the Central Limit Theorem does not apply (it only applies when increments are drawn for a distribution with a finite variance) and hence there can be no longer-term, larger-scale, diffusion-like behaviour (see box below).

Memoryless spatial processes: Fickian diffusion and Lévy flights in continuous time
Let $X = L^p(\mathbb{R})$ for $p \geq 1$. Let $f \in L^1(\mathbb{R})$ denote a real nonnegative function with unit mass, and r be a positive real parameter. For continuous time $t > 0$ we define the "similarity" function

$$F_t(x) = \frac{1}{t^r} f\left(\frac{x}{t^r}\right),$$

and the corresponding family mappings S_t, evolving any initial distribution, $u \in X$, at time $t = 0$, via the convolutions

$$S_t.u = F_t * u, \quad u \in X.$$

Then for any well-defined, memoryless, time-dependent evolution we must have $S_{t_a+t_b}.u = S_{t_b}.S_{t_a}.u$, for all $t_a, t_b > 0$. This requires that S satisfies the semigroup property: $S_{t_a+t_b} = S_{t_a}.S_{t_b}$ for $t_a,\ t_b > 0$. Taking Fourier transforms we see that this last is true if and only $\hat{F}_t(k) = e^{tg(k)}$, for some even function $g(k)$ vanishing at $k = 0$, and tending to $-\infty$ for k large. On the other hand, given the definition of F_t, a direct calculation of the transform shows that $\hat{F}_t(k) = \hat{f}(t^r|k|)$.

Putting these together we must have $g(k) = -\mu.|k|^{1/r}$, for some constant $\mu > 0$ (without loss of generality we take $\mu = 1$) so that, $\hat{f}(t^r k) = e^{-t|k|^{1/r}}$.

If $r = 1/2$ we recover the usual Fickian (diffusion) process: $F_t(x)$ is the fundamental solution of the diffusion equation. The semigroup S_t is the Gauss-Weierstrasse semigroup.

If $r = 1$ we obtain $F_t(x) = 1/\left(\pi t(1 + (x/t)^2)\right)$, the standard Cauchy distribution. The semigroup S_t is called the Poisson or Cauchy semigroup.

These two examples are very well known [13, 14] with the general case $1/2 \leq r \leq 1$ treated in

[13]: in fact, for r outside of this range f is not non-negative so we may rule out such cases.

Variations on Lévy motions and the corresponding fractional partial differential equations have been very widely proposed for use within pricing models in finance applications, extending the Black-Scholes method . Almost all of the resulting analysis seems to resort to numerical (sampling) methods. The semigroup property ensures that there is no memory dependence of solutions so the adoption of this class of dispersive process does not invite infinite opportunities for arbitrage.

In many applications Lévy flight methods are developed with controlled truncations of the Cauchy distribution.

Other properties of power-law distributions include the following:

- Power-law distributions are scale-invariant because they satisfy

  $$p(c_1 x) = c_2 p(x)$$

  for large x, where c_1 and c_2 are constants. This implies that multiplying the variable, or equivalently, changing the unit in which it is measured, does not affect properties of the system.

- Power-law distributions conveniently take the form of a straight line in a log-log plot because

  $$\log p(x) = \log C - \alpha \log x.$$

  When testing if empirical data are power-law distributed, it is instructive (but not conclusive) to plot their distribution on the log-log scale.

Although we have introduced power-law distributions for a real stochastic variable x, it is easy to see how the above idea applies to distributions for integer valued stochastic variables, which are normalised by sums rather than by integrals. If such distributions decay as a power law then moments are calculated through series, which may or may not converge.

Many networks exhibit power-law distributions in their network properties, such as the (integer) distribution for vertex degree which is defined in Section 2 below.

Exercise
Take an electronic version of a large book, measure the number of occurrences of each word and then plot the distribution of these numbers. Observe the behaviour of the distribution for large values (very popular words). Plot the "Zipf plot" of the data, that it is the relation between the rank of the word and the number of occurrences of the word. Any connection between the Zipf plot and the distribution?

1.7 Entropy, information and similarity measures

The entropy of a random variable, denoted by H, is a measure of its uncertainty and quantifies how much we know about a variable before observing it. After the observation, we get rid of the uncertainty and thus gain information H about the system. For a discrete random variable X, that can take one of n possible values (with the process existing in one of exactly n states), x, each with probability $p(x)$ the entropy is defined as

$$H(X) = -\sum_x p(x) \log p(x).$$

The maximum value $H(X) = \log n$ is realised when $p(x)$ is the uniform density, $(p(x) = 1/n)$. The minimum value $H(X) = 0$ is realised when X is deterministic, where $p(x) = \delta_{x,x_0}$ for a specific value x_0, and δ is Kronecker delta. Small H is complete order and certianty: larger H corresponds to more and more disorder and uncertainty.

The joint entropy $H(X, Y)$ of a pair of discrete random variables with joint distribution $p(x, y)$ is defined as

$$H(X, Y) = -\sum_x \sum_y p(x, y) \log p(x, y).$$

Similarly the conditional entropy $H(Y|X)$ is defined as

$$H(Y|X) = -\sum_x \sum_y p(x, y) \log p(y|x) = \sum_x p(x) H(Y|X = x),$$

and refers to the entropy of Y conditioned on the value of X and is averaged over all possible values of X.

The joint entropy and conditional entropy are related by the chain rule:

$$H(X, Y) = H(X) + H(Y|X).$$

This states that the total uncertainty about X and Y is simply the uncertainty about X, plus the average uncertainty about Y once X is known.

What does the knowledge of one variable tell us about another one? The conditional entropy $H(Y|X)$ addresses this question. More precisely, mutual information $I(X, Y)$ is defined as the amount of information gained on X by knowing the value of Y as follows:

$$I(X, Y) \equiv H(X) - H(X|Y) = H(X) + H(Y) - H(X, Y).$$

If Y is perfectly informative in the sense that it tells us everything about X, mutual information reduces to the entropy of X because $I(X, Y) = H(X) - H(X|Y) = H(X)$. Mutual information is rewritten as

Mutual information may be rewritten as

$$I(X, Y) = \sum_x \sum_y p(y, x) \log \frac{p(y, x)}{p(x)p(y)},$$

and is clearly symmetric.

Mutual information measures the cost of assuming that two variables are independent when they are in fact not. Mutual information captures non-linear correlations between random variables, in contrast to linear quantities such as the Pearson correlation coefficient; since the nature of the distribution around any nonlinear relationship is subsumed within the joint distribution, $p(y, x)$.

2 BASIC STRUCTURAL PROPERTIES OF NETWORKS

In this chapter we give an introduction to some basic ideas an Model Exampled concepts within network science.

2.1 Network Definitions

A network is a system made of vertices connected by links. Links can be undirected or directed, and un-weighted or weighted. In the mathematical literature, a network is called a "graph". It is defined as

$$G = (V, E),$$

where V is a set of vertices (also called vertices) and E is a set of edges (also called links).

The number of vertices and that of edges are denoted by n and m throughout. Each edge is defined by a pair of vertices, i.e., $e = (v, v') \in E$. In the case of undirected networks, the order of v and v' does not matter. In the case of directed networks, (v, v') indicates a link from v to v', and if $(v, v') \in E$ and $(v', v) \in E$, the two vertices are reciprocally connected. In the case of weighted networks, edges are also assigned with a weight function, characterising the importance or weight of the link.

In order to efficiently store networks and to carry out computations, it is necessary to use appropriate data structure. Each representation emphasises a certain aspect of the network and is amenable to certain types of computational or mathematical operations. For sparse graphs it may be easier tio just keep a list of present edges.

A network can be represented by the corresponding $n \times n$ adjacency matrix. Being adjacent means that two vertices are directly connected by an edge.

In the case of unweighted networks, the entries of the **adjacency matrix** are given by

$$A_{ij} = 1 \text{ if } (v_i, v_j) \in E; \quad A_{ij} = 0 \text{ otherwise.}$$

In general we will **not allow** self to self connections (self-loops), or double edges. So A is always binary and has all zeros all along its diagonal.

Sometimes we will write G_A to denote the network with adjacency matrix A, implicitly defining the vertex set.

A **walk** through a network is an ordered sequence of links where the ending vertex for each link is the starting vertex for the next link (see subsection 3.1 below). The length of a walk is the number of separate sequential edges.

Let A and B be two adjacency matrices defined on a common set of n vertices. Then $(AB)_{ij}$ counts the number of walks from v_i to v_j taking one edge from G_A, from v_i to v_k for some $k \neq i, j$, followed by one edge from G_B, from v_k to v_j. This is the exactly equal to number of vertices v_k that are linked from v_i in G_A; AND that are linked to v_j in G_B.

Similarly for $k \geq 1$, the entry $(A^k)_{ij}$ counts the number of walks from v_i to v_j of exact length k, in G_A.

We say a vertex v_i is walk-connected, or just "connected", to a vertex v_j in G_A iff there exists a path from v_i to v_j in G_A .

We say G_A is a **strongly connected** network if any pair of vertices are connected.

Then G_A is strongly connected iff and only if for all $i \neq j$ there is an integer k such that $(A^k)_{ij} > 0$; and hence G_A is strongly connected iff A is irreducible (see the condition of the Perron-Frobenius theorem above).

The **degree of a vertex** in G_A is the number of edges that connect to that particular vertex For an undirected graph the degree of vertex is simply the ith row/column sum of the A. The degree distribution, $P(d)$, represents the probability that a randomly chosen vertex has a particular degree, d.

We will usually let d_i denote the degree of vertex v_i.

It is clear that the sum of the degrees of all of the vertices is equal to twice the number of edges, $2m$. This simple result is called the "hand shake lemma".

For directed networks, we distinguish the in-degree, i.e., the number of links incoming to the vertex, and the out-degree, i.e., the number of links outgoing from the vertex. They are given by column sums of A and the row sums of A, respectively.

For directed graphs both the in-degrees and the out-degrees must sum to m separately. And A is not necessarily symmetric (it can be so iff all directed edges are reciprocated).

A network is called a **regular network** if all vertices have the same degree.

Vertex degree is the most basic measure of the *centrality*, or importance, of a vertex in a network. See later section for more details.

A majority of networks across different domains possesses long-tailed **degree distributions** $P(d)$ (see Figure 3 for examples). In many situations, their tail is described by a power-law, i.e.,

$$P(d) \sim d^{-\gamma},$$

where γ is typically between two and three. Because the maximum degree is equal to $n - 1$, this scaling law approximately holds true up to a certain cutoff degree, above which $P(d)$ rapidly decays to zero.

For an undirected graph, the average degree, denoted by $\langle d \rangle$, is given by

$$\langle d \rangle = \sum_d dP(d) = 2m/n.$$

The **friendship paradox** is a phenomenon, in which, anecdotally, the average number of friends of a friend is greater than the average number of friends of an individual. This is purely a mathematical consequence that always arises unless every vertex has the same degree. The paradox originates from the fact that vertices with a large degree contribute disproportionately to the average degree of a friend, as they have a higher probability of being friends than do low degree vertices.

Consider the situation shown where the network has $N = 6$ vertices with degrees 1,2,3,1,4, and 1 respectively and the average degree of a randomly selected individual is equal to 2.

To calculate the average number of friends of a friend, we have instead to perform a weighted average, accounting for the fact that a vertex with degree d will appear d times in the calculation of the average degree of friends. In this case the weighted average degree of friends is equal to

$$2.67 = \frac{(1^2 + 2^2 + 3^2 + 1^2 + 4^2 + 1^2)}{(1 + 2 + 3 + 1^+ 4 + 1)}.$$

In general, for sufficiently large and random graph the mean degree of a neighbour is given by

$$\frac{\langle d^2 \rangle}{\langle d \rangle}$$

which is $\geq \langle d \rangle$ (since the varaince of the degree distribution is given by $\sigma^2 = \langle d^2 \rangle - \langle d \rangle^2 \geq 0$).

A **clique** is a graph where all of the edges are present: every vertex is connected to every other vertex. The adjacency matrix for a clique contains all ones except for the main diagonal, which contains zeros (as we do not allow edges connecting a vertex to itself). We will usually denote this adjacency matrix by **1**.

If $G = (V, E)$, then the complementary graph is given by $G' = (V, E')$, where E' is the complement of E, and is the set of all admissible edges not in E. So considering G_A, then $\mathbf{1} - A$ is the adjacency matrix for the complementary graph, G'_A.

3 RANDOM GRAPH MODELS

We often wish to generate graphs which are drawn from a distribution, with suitable properties.

An undirected **random graph** is a probability distribution defined over the set of all possible graphs. Usually we will talk about an equivalent probability distribution being defined over the set of all such possible adjacency matrices (symmetry, binary, with zeros along the diagonal).

We could very laboriously list all possible candidates for A and assign each a probability, $P(A)$. Note that for n vertices there are exactly $2^{\frac{n(n-1)}{2}}$ distinct possible adjacency matrices for indirected graphs (why?). More usually we will have some rule that determines whether each element of the upper triangle part A, or various groups of the upper triangle elements of A, are equal to one.

For example we mights define a graph where each link (each element of A) is present independently with some given fixed probability $p \in [0, 1]$. Then when we wish to generate a graph from this random network we simply run around each element in the upper triangular part of A and set it to 1 with independent probability p. This random graph, on n vertices, is called an **Erdös-Rényi graph**, and it is usually denoted by $G(n, p)$. See Figure 1

More generally we might state that any edge (v_i, v_j) is present independently with a given probability p_{ij} (the individual edge probabilities could all be different). In that case we have an **expected value** for A, denoted by $\langle A \rangle$, with (i, j)th term, given by $\langle A \rangle_{ij} = p_{ij}$.

Of course $\langle A \rangle$ is symmetric and if we wish to generate a graph from this random graph we again simply run over all of the elements in the upper triangular part of A and make it equal to one with the corresponding independent probability p_{ij}; and then impose the fact that A is symmetric if it is required.

Let S denotes the set of all real valued, symmetric matrices, having zero diagonal elements and all non-diagonal elements taking values in [0,1], so that they are probabilities. Then for any random graph we must always have $\langle A \rangle \in S$.

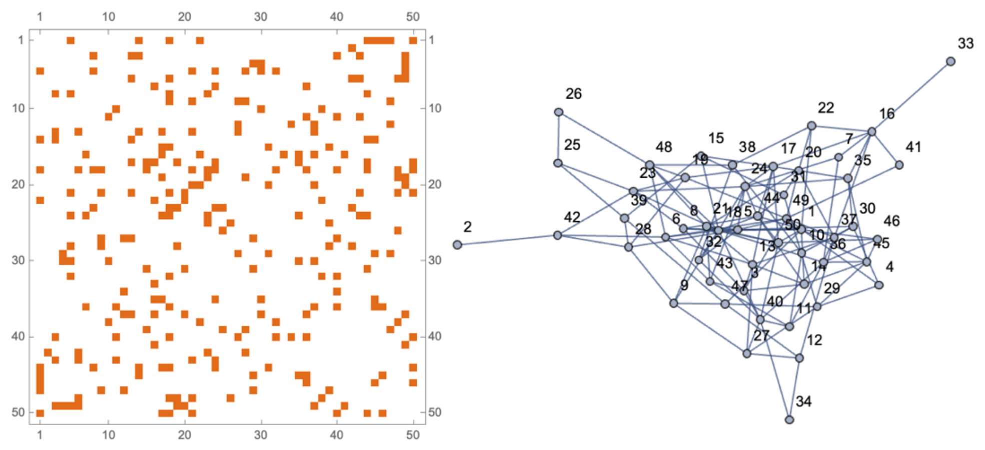


Figure 1: Example of an Erdös-Rényi graph: adjcency matrix and network plot.

The expected value of the adjacency matrix for an Erdös-Rényi graph is simply $p\mathbf{1} \in S$.

An important class of such graphs are **Stochastic Block Model**. Here $\langle A \rangle \in S$ is assumed to have has a block structure, so that all possible edges between pairs vertices within the same block are all present independently with a common given probability (just as for Erdös-Rényi graphs), while all possible edges between pairs of vertices within distinct blocks are all present independently with a common (given) probability depending only on the relevant pair of blocks. Again though, each edge is present independently of the presence of any and all others.

Core periphery graphs are often modelled via a Stochastic Block Model, with edges between vertices in the core set being more likely to occur, and edges between vertices in the periphery being less likely or even impossible.

Stochastic Block Model Example
Consider a model for an undirected graph where there are two disjoint blocks of vertices, block B_1 containing n_1 vertices and block B_2 containing n_2 vertices. Edges $e_{v_i,v_j},\ (i \neq j)$, are present independently with the following given probabilities $(p_1, p_2, p_{1,2})$:

$$p(e_{v_i,v_j}) = p_1 \ \ v_i \in B_1,\ v_j \in B_1,$$

$$p(e_{v_i,v_j}) = p_2 \ \ v_i \in B_2,\ v_j \in B_2,$$

$$p(e_{v_i,v_j}) = p_{1,2}\ v_i \in B_1,\ v_j \in B_2.$$

This could model the friendship network in a college where vertices are students and B_1 and B_2 are sets of science undergraduates and humanities undergraduates respectively. m_1 and n_2 are quite large.

What is the expected degree of a random vertex in each B_i? What is the expected degree of a random student from $B_1 \cup B_2$? what is the expected number of (two way) edges?

Fix the probabilities and generate instances of the adjacency matrix A for this stochastic block model, with vertices $1, \ldots, n_1$ in B_1, and vertices $n_1 + 1, \ldots, n_1 + n_2$ in B_2 for large n_1 and n_2. What in $\langle A \rangle$? See Figure 2

Random graphs can be thought of as models that are often used to generate observable instances of graphs with certain (expected or likely) properties. The probabilistic rule (the model) is usually a recipe that that uses random processes to determine whether every edge is present (the elements in the upper triangular part of A).

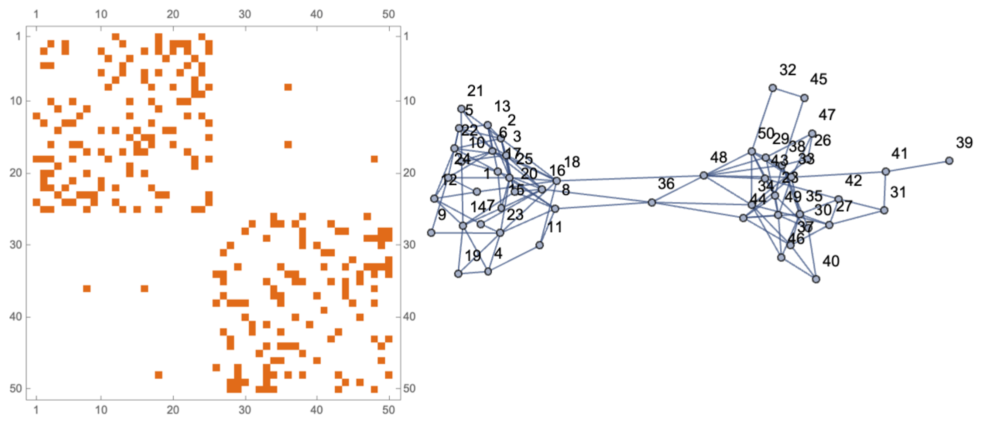


Figure 2: Example of a Stochastic Block Model: adjacency matrix and network plot. There are two two blocks with $p_1 = p_2 = 0.2$ and $p_{12} = 0.002$.

One might ask whether we consider graphs where the edges are NOT independent.

Random Triangles Example
Consider the graph on $n = 20$ vertices where each closed triangle of distinct vertices, (v_i, v_j, v_k), is present with an independent probability $p = 0.02$. This graph is a union of randomly generated triangles. Is this graph edge independent?

There are $m = n - 2$ possible triangles that contain any single edge, say (v_i, v_j). Thus each edge has a probability equal to

$$1 - (1 - p)^m = 0.3049\ldots$$

of being present within this graph.

Now suppose that we know that we observe that the edge (v_i, v_k) is present. What is the probability, knowing this, that edge (v_i, v_j) $(i \neq j \neq k)$ is present too?

The triangle (v_i, v_k, v_j) is one of the m triangles containing the edge (v_i, v_k). So the triangle (v_i, v_k, v_j) has an updated probability, q say, of being present as a result of this new information. In fact $q = p/(1 - (1 - p)^m) > p$, since, applying Bayes' Theorem to update the odds on the edge (v_i, v_k), we have

$$\frac{q}{1-q} = \frac{P((v_i, v_k)|(v_i, v_j, v_k))}{P((v_i, v_k)|\text{not}(v_i, v_j, v_k))} \times \frac{p}{1-p} = \frac{1}{1-(1-p)^{m-1}} \times \frac{p}{1-p}.$$

Now all of the other $(m-1)$ triangles involving (v_i, v_j) still have a probability p of being present. So now the edge (v_i, v_j) has a posterior probability equal to

$$1 - (1 - p)^{m-1}(1 - q) = 0.3372\ldots$$

of being present. Hence our knowledge of one edge, (v_i, v_k), being present, has changed our estimation of whether a second distinct edge, (v_i, v_j), can be present.

Returning to the simplest case of an undirected random graph with edge independence (with no self-loops and no multiple edges), if we know $\langle A \rangle \in S$, the matrix of independent edge probabilities, then for that random graph and any admissible symmetric binary adjacency matrix (with zero diagonal),

A, we have the distribution

$$P(A) = \prod_{i=1}^{N-1} \prod_{j=i+1}^{N} \langle A\rangle_{ij}^{A_{ij}} (1 - \langle A\rangle_{ij})^{(1-A_{ij})}.$$

This is just the product of the corresponding probabilities for each edge in A being present or not being present.

Hence the edge-independence assumption together with the knowledge of the expected value, $\langle A\rangle \in S$, is enough to determine the probability distribution, $P(A)$, over the set of admissible adjacency matrices.

So far we have met Erdös-Rényi graphs and their generalisation to stochastic block model graphs. Both are edge-independent, so we can just examine the probability that each edge is present.

The Erdös-Rényi model is usually seen as a model for sparse networks, where the total number of links scales linearly with the number of vertices, n. In $G(n, q)$, every link exists independently with the same probability, q Therefore, the probability of generating a network with an exact total of m links, from the $(n-1)n/2$ possibilities, is given by the binomial distribution:

$$p(m) = \binom{n(n-1)/2}{m} q^m (1-q)^{n(n-1)/2-m}.$$

The expected number of links is given by $qn(n-1)/2$. Similarly, because a vertex is independently adjacent to any other vertex with probability q, the degree distribution is given by

$$p(d) = \binom{n-1}{d} q^d (1-q)^{n-1-d}.$$

In practice we would not wish that the average degree of the vertices $\langle d\rangle = q(n-1)$ should depend on N. Therefore, we usually employ a small value of q, more precisely, $q \propto 1/n$. In the limit of large networks, where $q = \langle d\rangle/(n-1)$ is sufficiently small, the binomial degree distribution above is well approximated by the Poisson distribution

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

Several properties of the Erdös-Rényi random graph can be derived thanks to the independence of links. Although difficult to derive, the average distance between pairs of vertices in the Erdös-Rényi random graph is given by

$$\approx \frac{\log n}{\log\langle d\rangle}.$$

In general the degree distribution $p(d)$, above, for the Erdös-Rényi model decays much faster for large d than that for observed graphs (in social media, bioinformatics and other applications).

In Figure 3 we can see that many observed degree distributions decay much more slowly than that for the Erdös-Rényi graphs.

Those observed graphs tend to have degree distributions with *fat tails* that very often decay as power law decay (at least until some cut off). For this reason the Erdös-Rényi model in its simplest form is not very practical as a model of real world phenomena. It is rather popular within combinatorics: but is useful mainly as a "null hypothesis" within applied network theory — merely a way of generating graphs with some little structure (just the desired edge density) to be contrasted with a real-world observed graph. We can do rather better.

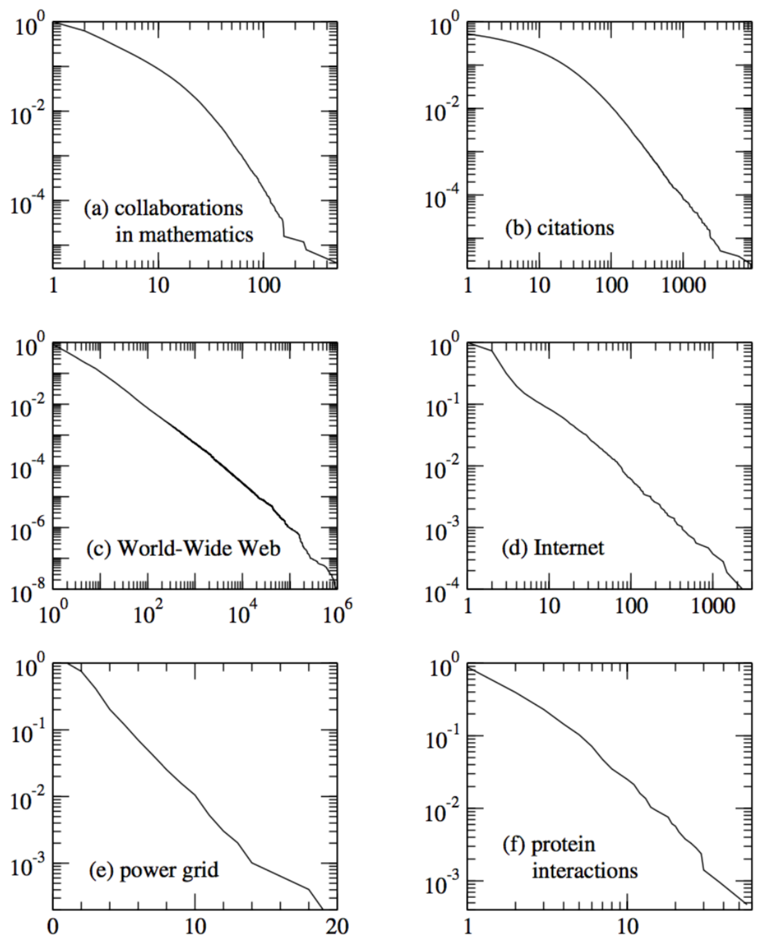


Figure 3: Cumulative degree distributions for six different networks. The horizontal axis for each panel is vertex degree d (or in-degree for the citation and Web networks, which are directed) and the vertical axis is the cumulative probability distribution of degrees, i.e., the fraction of vertices that have degree greater than or equal to d. The networks shown are: (a) a collaboration network of mathematicians; (b) citations between 1981 and 1997 to all papers cataloged by the Institute for Scientific Information; (c) a 300 million vertex subset of the World Wide Web, circa 1999; (d) the Internet at the level of autonomous systems, April 1999; (e) the power grid of the western United States; (f) the interaction network of proteins in the metabolism of the yeast. Of these networks, three of them, (c), (d) and (f), appear to have power-law degree distributions, as indicated by their approximately straight-line forms on the doubly logarithmic scales. Taken from [4]

The **configuration model** is a generalisation of the Erdös-Rényi random graph to the case of an arbitrary but given degree for each vertex.

It is used to inspect the effect of heterogeneous degree distributions because it does not have more specific features such as high clustering. The model is defined as a random graph in which all possible configurations appear with the same probability under the constraint that vertex v_i has degree d_i $(1 \leq i \leq n)$. The degree sequence $\{d_i\}$ is sometimes observed within a real world network or often generated by a given degree distribution $p(k)$ under the constraint that the sum of the degrees is an even number to satisfy the handshaking lemma.

To generate an instance of the configuration model for a given degree sequence, we first create exactly d_i "stubs" (half-edges) at each vertex v_i. Then we randomly select pairs of stubs one by one to connect them as an edge, as long as the tentatively connected pair does not form a multiple edge or self-loop.

In fact, we must avoid the case in which the link creation stops reaches an impasse. For example, if there remain three vertices which have 1, 2, and 3 unused stubs, we have to create three more links because there are six stubs remaining. However, we cannot do that without a self-loop or multiple edge.

Consider a large network generated from a configuration model with a given degree sequence. A stub emanating from v_i is connected to v_j with probability $d_j/2m$ because the number of stubs in the network is equal to $2m = \sum_{i=1}^{n} d_i$ (this is called the handshake lemma).

Because v_i owns d_i stubs, the expected number of links between v_i and v_j is given by

$$\langle A_{ij} \rangle = \frac{d_i d_j}{2m}, \tag{5}$$

where $\langle A_{ij} \rangle$ represents the mean of the adjacency matrix element.

So far we have met Erdös-Rényi graphs and their generalisation to the stochastic block model and the configuration model. We will meet some more models later.

3.1 Measures derived from walks and paths

A **walk** is defined as a succession of adjacent vertices such that one can travel from the start vertex to the end vertex by traversing links. A **path** is a walk where each vertex is visited only once (with the possible exception that the walk may end at the vertex where it begins - becoming a **cycle**).

Walks are used for constructing dynamical processes on networks (e.g., random walks) and measurements such as the Katz centrality (see below), where all possible walks from one vertex to another are exhaustively counted. Paths are particularly useful when considering the shortest travelling route from one vertex to another. In the network science literature, which is rooted in statistical physics, authors tend not to distinguish walks and paths. Here, however, we will do so.

We have already seen that the number of walks of a certain length can be obtained from powers of the adjacency matrix. The adjacency matrix provides the number of walks of length 1 between two vertices. In general, the number of walks of length l is given by the elements of A^l.

To identify paths from a vertex to another requires some more effort. The **distance** between a pair of vertices, v_i and v_j, denoted by $\delta(v_i, v_j)$, is defined as the smallest number of edges in a path necessary to go from v_i to v_j.

One way is to calculate:

$$\delta(v_i, v_j) = \min_{l \geq 1} \{ l | (A^l)_{ij} > 0 \}.$$

For undirected networks, the distance defined in this way satisfies the axioms that a distance measure should satisfy: non-negativity (i.e., $\delta(v_i, v_j) \geq 0$), coincidence (i.e., $\delta(v_i, v_j) = 0$ if and only if $v_i = v_j$), symmetry (i.e., $\delta(v_i, v_j) = \delta(v_j, v_i)$) and triangle inequality (i.e., $\delta(v_i, v_j) \leq \delta(v_i, v_k) + \delta(v_k, v_j)$).

For directed networks, the symmetry is broken because a shortest path from v_i to v_j is not generally the same as that from v_j to v_i.

For both undirected and directed networks, $\delta(v_i, v_j)$ can be efficiently calculated by **Dijkstra's algorithm**, as follows.

For a fixed v_i, first initialise the distance from v_i by setting $\delta(v_i, v_i) = 0$ and set tentative distances $\delta(v_i, v_j) = \infty (j \neq i)$. Second, set $d(v_i, v_i) = 1$, where v_j is a neighbour of v_i. Third, we declare that v_i has been visited. Fourth, consider each neighbour of v_j except v_i. If a neighbour, v_l, has a tentative distance value from v_i larger than two, then we reset it to $\delta(v_i, v_l) = 2$. When all neighbours of v_j are exhausted, we declare that v_j has been visited. Fifth, we select an unvisited vertex with the smallest tentative distance value (equal to 2 at the first iteration) and we inspect its all neighbours one by one (resetting to $\delta = 3, 4, \ldots$ at each successive iteration). We repeat the same procedure to determine the distance from v_i to every other vertex.

This procedure sweeps out through the graph from v_i. For large and sparse graphs this is far more efficient that calculating the successive powers, A^l (requiring n^2 calculation at each value of l, many of them resulting in zero).

For undirected networks, the average distance for a network is defined by tye vaeregae ove all pairs of distinct vertices:

$$L = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{i=1}^{i-1} \delta(v_i, v_j).$$

In many real networks, L is remarkably small as compared to the number of vertices, N. For example, a Face-book network composed of $n \sim 7.2 \times 10^8$ active users, with 6.9×10^{10} friendship links, yielded $L \sim 4.7$.

The **diameter** of a networks is defined by the longest paths between any pair of vertices:

$$D = \max_{u,v,\in V} \delta(u, v).$$

In undirected networks, we defined two vertices to be connected if there exists a path between them. Connectedness is an equivalence relation because it is reflexive , symmetric, and transitive. Intuitively, a connected component is an island within which one can travel from any vertex to any other along a path. There is no path between vertices in different components. Connected components impose limitations on any dynamical process taking place on the network. In epidemic processes, for example, the existence of distinct components implies that certain regions of the net- work are never infected, independently of the model of epidemic dynamics and its parameters.

In directed networks, symmetry is not satisfied because the existence of return paths cannot be guaranteed. Therefore, the concept of connectedness is more complex, and the notions of strong and weak connectedness are distinguished. vertices u and v are said to be strongly connected if there exist reciprocal paths. Two vertices u and v are said to be weakly connected if there exists a path between u and v in a network where the direction of the links is discarded. Both strong and weak connectedness is an equivalence relation and induces strongly and weakly connected components, respectively. For example, a strongly connected component is a maximum set of vertices in which each pair of vertices is strongly connected. Strong connectedness implies weak connectedness but not vice versa.

3.2 The clustering coefficient and small worlds

The observable relatively small diameters of some large sparse networks brought about the concept of **small world** networks.

Empirical networks are quite often abundant in triangles, i.e., mutually connected three vertices. The amount of triangles in a network is quantified by the **clustering coefficient**. It is defined through the local clustering coefficient:

$$C_i \equiv \frac{\text{number of triangles including the } i\text{th vertex}}{d_i(d_i - 1)/2)},$$

which measures the abundance of triangles in the neighbourhood of the ith vertex v_i. The denominator gives the normalisation such that $0 \le C_i \le 1$.

In a social friendship network C_i measures how many pairs of your friends are themselves friends.

If any possible pair of the neighbours of v_i are adjacent to each other, to form a triangle, then $C_i = 1$. If no pair of neighbours of v_i are adjacent to each other, then $C_i = 0$. The clustering coefficient, denoted by C, is defined as the average of C_i over the vertex set:

$$C \equiv \sum_{i=1}^{N} C_i.$$

Note that $0 \le C \le 1$.

What is C for the Erdös-Rényi graph $G(n, p)$?

There are many examples of networks that have a high C coefficient.

For example, consider a directed circular lattice with N vertices arranged like the hours on a clock, ith all pairs of vertices separated by less than or equal to $k < (n-1)/2$ places being connected with an edge.

What is C in this case? Each vertex has degree $K = 2k$, with k connections in the clockwise direction and k connections in the clockwise direction. The denominator of C_i is thus $k(2k-1)$, while there are exactly $k(k+1)/2$ pairs of adjacent vertices that are not directly connected (show that this is true).

Thus (since, by symmetry, all vertices have the same clustering coefficient) we have

$$C = C_i = \frac{3(k-1)}{2(2k-1)}.$$

Note that some authors write $K = 2k$, the degree of each vertex in the lattice, so that

$$C = C_i = \frac{3(K-2)}{4(K-1)}.$$

In either representation as k becomes large then $C \to 3/4$.

This 3/4 value is really quite high, even though N might be very large, since there are many local triangles in the circular lattice.

The **small-world model** due to Watts and Strogatz [3] is created by choosing at random a fraction p of the edges in the graph and moving one end of the edge to a new location, chosen uniformly at random.

There is a slight variation on the model in which some shortcuts are added randomly between vertices, but no edges are removed from the underlying one-dimensional lattice. See Figure 4.

If the rewiring probability p is small then the number of adjusted edges is small and so C remains close to $3/4$.

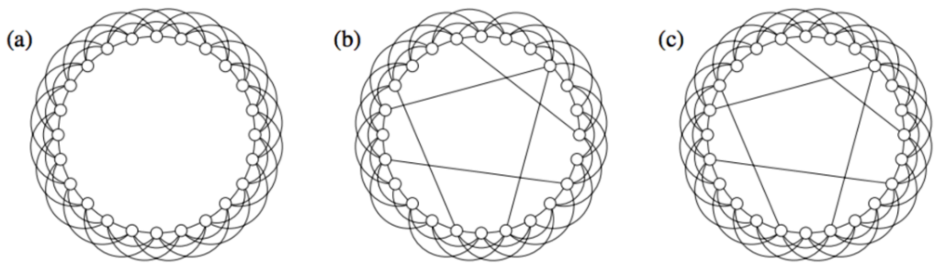


Figure 4: (a) A one-dimensional lattice with connections between all vertex pairs separated by k or fewer lattice spacing, with $k = 3$ in this case. (b) The small-world model of Watts and Strogatz is created by choosing at random a fraction p of the edges in the graph and moving one end of each to a new location, also chosen uniformly at random. (c) A slight variation on the model in which shortcuts are added randomly between vertices, but no edges are removed from the underlying one-dimensional lattice. Taken from [4].

If the rewiring probability p is very close to one then almost all if the edges are rewired: the network is thus equivalent to an Erdös-Rényi graph. There are k^2 edges in total. Any possible edge is thus present with probability $q = 2k^2/n(n-1)$. Thus $C \to q = 2k^2/n(n-1)$ as $p \to 1$.

The rewiring process is designed to reduce the diameter of the resultant network to be low while keeping the clustering coefficient high. Without it the diameter would be $\sim n/2k$ (why?): so if k is fixed and n is large one has to jump around the network k places at a time. Each rewiring edge reduces this estimate [3].

A high density of triangles is often associated to the mechanism of triadic closure (see also section 5, below), that is the tendency for wedges (paths of length 2, like open jaws) to form closed triangles. Wedges are then associated to unstable structures, later turning into more stables ones. This mechanism is at the heart of several methods for edge prediction. Say that you have a snap-shop of a social network, at a certain time. In order to predict the links that will be created at future steps, a simple, but very efficient strategy consists in considering pairs of vertices that are not (yet) connected, but belong to several wedges. So friends of friends tend to become friends. See for instance [8].

Small world networks (where there is high local clustering but a small diameter) are often observed in social networks. The small diameter is reflected in the famous "Six degrees of separation" phenomena: please read this `https://en.wikipedia.org/wiki/Six_degrees_of_separation`.

3.3 Preferential attachment

A broad range of networks grow in time in terms of both the number of vertices, n, and the number of edges, m. Examples include citations in science, web graph and networks of airports. Network growth is a type of temporal fluctuations of networks. Understanding mechanisms of network growth definitely helps one to understand temporal dynamics of networks.

In this section, we study a popular growing network model with the preferential attachment mechanism, that has played a pivotal role in the entire network science. The model was proposed by Barabási and Albert [24, 25], which we call the BA model, while the model had been known for longer time. The model is an instance of a family of multiplicative stochastic models, starting around a century ago with the Polya urn model and the Yule process. Historically, the mechanism of preferential attachment was also identified qualitatively by the sociologist Robert Merton, who called it the Matthew effect, after a passage in Biblical Gospel of Matthew. The Yule process was studied by the economist Herbert Simon, interested in the distribution of wealth, who showed that it produces power-law distributions. This work inspired the Price's network model.

The BA model produces a network according to the following steps:

1. Prepare n_0 initial vertices each of whose degree is at least one. A typical choice is the clique (where a link exists between every pair of nodes).
2. Add a new vertex with $m < .n_0$ half-edges to the existing network. This simplest case is where $m = 1$. Suppose that the existing network has n' nodes ($n' = n_0$, initially, but we shill apply this step iteratively) with degrees d_i for $i = 1, \dots, n'$. Then the probability that each half-edge connects to v_i is specified by

$$\Pi(d_i) = \frac{d_i}{\sum_{j=1}^{n'} d_j} \quad i = 1, \dots, n.$$

This indicates that any existing vertex receives a new edge with the probability proportional to its present degree, hence the name of preferential attachment. If $m > 1$ this must be applied under the constraint that we avoid multiple edges, although this constraint is non-essential. Strictly, we also have to decide on whether or not to update the relevant degree, d_i, values used in $\Pi(d_i)$ when one of the m links has been added. However, this decision is again immaterial.
3. Add the nodes one by one until we have n nodes according to step 2.

As will we show below, the BA model produces networks with a power-law degree distribution $P(d) \propto d^{-3}$.

In early stages, nodes have similar values of d, which are equal to or slightly larger than m. However, once $\{d_i\}$ becomes somewhat heterogeneous, the heterogeneity will self-reinforce owing to the preferential attachment mechanism. The degree distribution of the BA model can be derived in different ways. Here we proceed via master equations. Denote by $p(d, t_i, t)$ the probability that a node v_i that has joined at time-step t_i has degree d at time-step d. The master equation for $p(d, t_i, t)$ is given by

$$p(d, t_i, t+1) = \frac{d-1}{2t} p(d-1, t_i, t) + \left(1 - \frac{d}{/}2t\right) p(d, t_i, t)$$

because d increases by one with probability $m\Pi(d) \approx d/2t$ and does not change with probability $1 - d/2t$ in a time step. When n is large, we wish to obtain the asymptotic solution

$$P(d) = \lim_{t\to\infty} \frac{\sum_{t_i} p(d, t_i, t)}{t}.$$

The normalisation factor $1/t$ comes from the fact that there are $t + m \approx t$ nodes at time t.

The vertex that joins at time $t_i = t+1$ has been absent at time t, such that $p(d, t+1, t) = 0$. By using this and summing the master equation over t_i, we obtain

$$\sum_{t_i=1}^{t+1} p(d, t_i, t+1) = \frac{d-1}{2t} \sum_{t_i=1}^{t} p(d-1, t_i, t) + \left(1 - \frac{d}{/}2t\right) \sum_{t_i=1}^{t} p(d, t_i, t).$$

But

$$p(d) \approx \sum_{t_i=1}^{t} p(d, t_i, t)/t = \sum_{t_i=1}^{t} p(d, t_i, t+1)/(t+1)$$

and

$$p(d-1) \approx \sum_{t_i=1}^{t} p(d-1, t_i, t)/t,$$

so we have

$$(t+1)p(d) = \frac{d-1}{2t} tp(d) + \left(1 - \frac{d}{2t}\right) tp(d).$$

Hence

$$p(d) = \frac{d-1}{d+2} p(d-1) \;\; d \geq m+1,$$

which yields

$$p(d) \propto \frac{1}{d(d+1)(d+2)} \propto d^{-3}.$$

So the degree distribution has a "fat", powerlaw, tail for large d. This d^{-3} behaviour was what we set out to demomnstrate.

An alternative argument, using a differential equation, is as follows [2].

Consider the situation, as before, where a graph grows by having a single vertex added to the existing graph over each unit of time, that is attached to exactly m of the existing vertices. As before, the probability that it becomes attached to any vertex is assumed to be given by the relative degree of the vertex. Suppose that at t = 0 we start with a very small number, say n_0, of sparsely connected vertices, and that new vertices are added randomly at an expected rate of one per unit time. After integer time t we will have added approximately t vertices and there will be approximately mt edges connecting the $t + n_0$ vertices.

Let $d_i(t)$ denote the expected degree of the ith added vertex, the one that was added at around time i. Then for $t \geq i$, $d_i(t)$ grows as successive vertices (and hence edges are added), with vertex i wins new attachments at a rate proportional to its current relative degree. We have

$$\frac{d}{dt} d_i(t) = (\text{rate at which new edges are added}) \times P(\text{new edge attaches to vertex } i),$$

so that

$$\frac{d}{dt} d_i(t) = m \frac{d_i(t)}{2mt} = \frac{d_i(t)}{2t}$$

(since at time t the sum of all vertex degrees is the twice the number of edges, which is thus $\approx 2mt$), and it must also satisfy the initial condition $d_i(i) = m$.

So directly we have

$$d_i(t) = m(t/i)^{1/2}.$$

Hence for any value of d chosen larger than m, at time t we will have $d_i(t) < d$ if and only if $i > tm^2/d^2$. So there are approximately $n_0 + t - tm^2/d^2$ vertices with degrees less than d. Therefore there is a fraction

$$1 - \frac{tm^2}{d^2(t + n0)}$$

of the vertices that have degrees less than d. For large time the distribution forgets n_0 and this is simply

$$1 - \frac{m^2}{d^2}.$$

Note that this doesn't make sense for $d < m$ since all vertices have degree greater than or equal to m.

This is the cumulative distribution for the vertex degrees (valid for large enough d where $d \geq m$). The density distribution (the derivative of the cumulative with respect to d) is thus $P(d) \propto d^{-3}$, as required.

It can also be shown that the clustering coefficient is given by

$$C \approx \frac{m-1}{8} \frac{(\log n)^2}{n}$$

which tends to zero for large n. Some extensions of the BA model can realise a non-vanishing C value as $n \to \infty$.

The BA preferential attachment model contains some rather unrealistic ingredients, as a new vertices must have access to information about the whole network in order to decide which vertex to connect to. This limitation can be solved by using local mechanisms, such as redirection and copying, that essentially lead to preferential attachment together with other desirable features. In particular, copying processes allow to generate scale-free net- works with a high density of cliques of different sizes; see [27].

We will generalise this model to stochastic block model preferential attachment in section 8.2, below.

Exercise.
Generate numerically networks according to the preferential attachment model and verify the theoretical predictions for the degree distribution.

3.4 Centrality

Centrality measures aim to quantify the *importance* of vertices in a network. The simplest one is the degree (i.e., degree centrality), with which hubs are considered to be important. The degree centrality is effective in various situations but not always. This observation has motivated the introduction of different types of centrality measures. In this section, we explain some of them.

The **closeness centrality** and **betweenness centrality** are popular centrality measures based on the distance between pairs of vertices.

The **closeness centrality** for vertex v_i is defined by

$$\text{closeness}_i = \frac{n-1}{\sum_{j=1, j\neq i}^{N} \delta(v_i, v_j)}.$$

It is the reciprocal of the mean distance to all other vertices. The closeness centrality is well-defined only for connected networks

The **betweenness centrality** is defined as the fraction of the shortest paths passing through the vertex in question. This quantity is averaged over all possible pairs of vertices. The betweenness of the ith vertex is defined by

$$\text{betweenness}_i = \frac{2}{(n-1)(n-2)} \sum_{j=1, j\neq i}^{n} \sum_{l=1, l\neq i}^{j-1} \frac{\sigma_{jl}^{I}}{\sigma_{jl}}$$

where σ_{jl} is the number of the shortest paths connecting the jth and lth vertices, and σ_{jl}^{I} is the number of such shortest paths that pass through the ith vertex. The convention is that we regard the summand on the right- hand side to be zero when σ_{jl} is equal to zero (i.e., when the jth and lth vertices are in different connected components). The summation excludes the shortest paths that start or end at the ith vertex because it is obvious that such a path does not go through the ith vertex. The normalisation factor $2/((n-1)(n-2))$ comes from the combinations of j and l, whereas it is often neglected in practice. Betweenness is employed more in fields of computer science than in mathematical approaches, largely as it does not yield to algebra. The next concept will do so.

Given an adjacency matrix, A, we have seen that the number of walks from the ith vertex to the jth vertex with exactly l steps is given by the $(A^l)_{ij}$.

Supposing that short walks are more important than long walks in mediating, e.g., communication and infectious diseases, the we scale the importance of each walk of length l (for $l \geq 0$) by a factor of α^l, where we have $0 < \alpha < 1$. Then, the weighted sum of the number of walks from the ith to the jth vertices of various lengths is given by the (i, j)th element of the geometric series

$$I + \alpha A + \alpha^2 A^2 + \alpha^3 A^3 + \ldots = (I - \alpha A)^{-1};$$

providing that the sum converges.

Note that the walks of length zero also contribute to the counting with weight one.

The **Katz centrality** of the ith vertex is defined by

$$\mathrm{Katz}_i = \sum_{j=1}^{n} [(I - \alpha A)^{-1}]_{ij}.$$

In other words, the weighted sum of the number of walks starting from the ith vertex is summed over all destination vertices. If $\alpha = 0$, then $\mathrm{Katz}_i = 1$ for all i. Therefore, we are interested in making α large to diversify the values of Katz_i.

In fact, as intuitively understood f, $(I - \alpha A)^{-1}$ diverges for a large α. This occurs when an eigenvalue of $I - \alpha A$ hits zero for the first time as α is increased from zero. Therefore, the Katz centrality is well-defined when α is smaller than the inverse of the largest eigenvalue of A; which is the Perron-Frobeneus eigenvalue, the spectral radius of A, $\rho(A)$.

A well-known centrality measure for directed networks is the **PageRank**, which was first introduced for ranking webpages and later adopted in a variety of applications. The PageRank is defined as the stationary density of a discrete-time random walk, particularly on directed networks.

We will introduce it later on, after introducing the concept of random walks on networks. In contrast with the previous metrics, either defined in terms of shortest paths or number of paths passing by a vertex, PageRank is a typical recursive metric, based on the circular idea that: a vertex is important if it receives connections from many important vertices. As we will see, this relation leads to an eigenvector problem. Similar arguments lead to other centrality measures, such as **Eigenvector centrality**.

3.5 Spectral properties

A broad range of dynamical and structural properties of networks is characterised by spectral properties of a matrix describing the network. Depending on the problem at hand, we often use the adjacency matrix (denoted by A), the (combinatorial) Laplacian matrix (denoted by L) or the normalised Laplacian matrix (denoted by $\tilde{L}$).

Spectral properties of networks have been studied in detail, and various bounds are available. In this section, we present a summary of basic spectral properties of undirected networks.

The Laplacian, which we have already met in section 1.2, using matrix notation, and the normalised Laplacian are defined by

$$L_{ij} = d_i\delta_{ij} - A_{ij} \iff L = D - A;$$
$$\tilde{L}_{ij} = \delta_{ij} - \frac{A_{ij}}{\sqrt{d_i d)j}} \iff \tilde{L} = I - D^{-1/2}AD^{-1/2}.$$

Recall that D is the $n \times n$ diagonal matrix whose (i, i)th element is equal to d_i. The eigenvectors of the three matrices are the same for regular networks.

Both Laplacian matrices are symmetric and their eigenvectors $\mathbf{u}_l$ $(1 \leq l \leq n)$ form an orthonormal basis such that the inner products satisfy $\mathbf{u}_l^T\mathbf{u}_{l'} = \delta_{ll'}$. Any vector $\mathbf{x} \in \mathbb{R}^n$ can be decomposed as

$$\mathbf{x} = \sum_{l=1}^{n} a_l\mathbf{u}_l,$$

where $a_l = \mathbf{x}^T\mathbf{u}_l$.

For the adjacency matrix, A, it is customary to order the eigenvectors from the largest λ_1 to the smallest λ_n , whereas the eigenvalues are usually ordered from the smallest to the largest for the Laplacian matrices.

The two Laplacian matrices always have a zero eigenvalue. In fact, the corresponding eigenvector for L and $\tilde{L}$ is given by $\mathbf{u}_1 = (1, \ldots, 1)^T$ and $\mathbf{u}_1 = (\sqrt{d_1}, \ldots, \sqrt{d_n})^T$ respectively.

In undirected networks, the zero eigenvalue, $\lambda_1 = 0$, is an isolated eigenvalue and all the other eigenvalues are positive such that $0 = \lambda_1 < \lambda_2 \leq \ldots \leq \lambda_N$ if the network is connected. In this case, the smallest nonzero eigenvalue of the Laplacian matrix, λ_2, determines the relaxation time of diffusion and synchronisation dynamics induced by L, and is often called the spectral gap. The corresponding eigenvector, $\mathbf{u}_2$, is called the Fiedler vector. In general, the number of connected components is given by the number of zero eigenvalues of L or $\tilde{L}$. Therefore, the network is connected if and only if $\lambda_2 > 0$.

3.6 Fitting models to data

When we are given a network, usually as a list of edges, with the vertices labelled in no particular order, we often wish to represent that data according to a particular model. This is a very practical problem and here we consider some approaches based on maximum likelihood methods and also investigate some consequences of having errors in te given data. The latter problem tens to be ignored by many practitioners, but it is real. The problem essentially arises when a network is formed by observing some underling evidence of relationships between entities (represented by vertices) and interpreting some such relationships as (undirected or directed) edges or not; and then presenting they data to the network analysis. False positives are edges which are in thee network which really should not have been there. False negatives are missing edges in the data : edges which are not observed or accepted as such, and thus are not present.

We consider some inverse problems: how should we represent the observed data within a given class of networks?

Whenever a scientist seeks to impose some structure on a collection of observed data, he or she learns something. A wonderful example is provided by the periodic table of elements: by wrapping elements around with respect to their atomic numbers with a period of eight one finds that elements with similar properties form up the columns, instantly validating the idea. This told us something about the structure of atoms, yet it also enabled the prediction the properties of elements that had still to be observed.

A good test of any inverse is to generate an actual network using the class of network model of interest and then shuffle the vertices randomly. Then apply the inverse method to the "observations" and see if it returns (un-shuffles) the observed data back to the correct form (as it was generated).

3.7 Calibration for undirected range dependent models

Consider first undirected **range depenedent random networks**. Here we have a network whers the vertices are arranged in a simple ordered list $i = 1, \ldots, n$ and vertices i and j are connected by an edge with independent probability

$$p_{ij} = f(k) \quad k = |i - j|,$$

where f is a monotonically decreasing function taking values in (0,1), and $k = |i - j|$ is called the range of the possible edges. Intuitively pairs vertices of vertices with a smaller range are relatively more likely to be connected. The expected value $\langle A \rangle >$ is a Toeplitz matrix.

Typically we will choose $f(k) = ab^k$, where a and b are constants in (0,1), or more simply

$$f(k) = \alpha e^{-\eta k^2} / (1 + \alpha e^{-\eta k^2}),$$

for some $\eta\alpha > 0$. We will choose the latter here to keep the algebra simple. Later we will generalise this model to undirected networks, but let us keep this undirected for now.

To generate such a network with independent edge probabilities we simply visit each edge and accept its presence with probability $f(k)$ - essentially filling in the upper triangular part of A and then imposing symmetry.

Now if we are given the edge data A_{ij} , a binary adjacency matrix, the given indices i will usually not be in the right order. The vertex list is scrambled. So we have to fit the parameters in f and determine the correct ordering (intro the range dependent concept).

Let $q = (q_1, ..., q_n)^T$ denote some permutation function for the index set $i = 1, ..., n$. Then since we assume independence of the individual super-diagonal matrix elements within our random adjacency matrix model, the probability of observing the data, given the permutation $\mathbf{q}$, can be formed by the product of the probabilities for each element, after the indices have been permuted. We have the conditional "likelihood": a product over possible edges:

$$\mathcal{L} = \prod_{i<j} f(|q_i - q_j|)^{A_{ij}} (1 - f(|q_i - q_j|))^{1 - A_{ij}}$$

So

$$\log \mathcal{L} = \sum_{i<j|A_{ij}=1} \log f(|q_i - q_j|) + \sum_{i<j|A_{ij}=0} \log(1 - f(|q_i - q_j|)).$$

Now comes a trick:

$$\log \mathcal{L} = \sum_{i<j|A_{ij}=1} \log \frac{f(|q_i - q_j|)}{1 - f(|q_i - q_j|)} + \sum_{i<j} \log(1 - f(|q_i - q_j|)).$$

But the second term does not depend on $\mathbf{q}$, so it can be discarded in maximising $\log \mathcal{L}$. So we just maximise a product of log-odds over the observed edges:

$$\log \mathcal{L} = \sum_{i<j|A_{ij}=1} \log \frac{f(|q_i - q_j|)}{1 - f(|q_i - q_j|)}.$$

Substituting $f(k) = \alpha e^{-\eta k^2}$, we have

$$\log \mathcal{L} \propto \sum_{i<j|A_{ij}=1} (q_i - q_j)^2 = -\frac{1}{2} \sum_{i,j=1}^{n} L_{ij} (q_i - q_j)^2.$$

Here L is the Laplacian matrix associated with A, that we met earlier. This follows directly from (2) in section 1.2.

So we must minimise

$$-\log \mathcal{L} \propto \mathbf{q}^T A \mathbf{q}.$$

How best to choose a permutation $\mathbf{q}$ to achieve this.

Next we make a huge simplifying step. We *relax* the problem to consider a nearby problem that has a much easier solution.

Above we sought a vector $\mathbf{q}$ taking integer values (and indeed all integer values in $\{1, \ldots n\}$) and then we wished to reorder the vertices in the increasing order given by the corresponding elements of $\mathbf{q}$ (the reordering equivalent to making the permutation $\mathbf{q}$). Notice that if we add any fixed constant onto all elements of $\mathbf{q}$, or we multiply $\mathbf{q}$ by any positive constant, then this has no effect on the reordering. We just reorder by sorting vertices by the values of the corresponding elements, q_i.

We will minimise $-\log \mathcal{L}$ with respect to $\mathbf{q} \in \mathbb{R}^n$, and we will reorder the vertices in increasing order of their corresponding real elements of $\mathbf{q}$. Now, since multiplying the solution, $\mathbf{q}$, by any positive constant, or adding a fixed constant onto all of its elements, makes no difference, we must find a real $\mathbf{q}$ to satisfy

$$||\mathbf{q}|| = 1 \text{ and } \mathbf{q}.(1, 1, ..., 1)^T = 0.$$

But we have seen that this functional is minimised by taking $\mathbf{q}$ to be the Fiedler eigenvector. Since A is self adjoint the Fiedler eigenvector orthogonal to $(1, 1, ..., 1)^T$, the eigenvector corresponding to the Null space. So we re-order the given vertices of the graph in the order of the increasing elements of the Fiedler eigenvector.

TRY THIS OUT: generate a large Range Dependent Graph: find the Fieldler eigenvector (for the giant component) Check that the ordering suggested by the Fieldler eigenvector components is as that (or close to that) used to generate the RDG. Add in some (5%) false positives (extra edges that are extremely unlikely): what happens? Or create false negatives (delete 5% of existing edges) :what happens?

Example: a proteome - protein-protein interaction (PPI) networks
Here we illustrate the methods above with applications originally published twenty years ago (see [2] and the references therein).

Since the time of Gregor Mendel, biologists have been attempting to understand how genes determine biological properties. How the "genotype" (the organism's full set of genetic information) relates to the "phenotype" (the organism's features and functionalities). Differences in genes largely explain biological diversity. DNA forms an organisms genetic signature and may be viewed as a linear string where each character is one of the four nucleotide bases: C,A,T,G. It is arranged as a number of one-dimensional lattices (chromosomes). Certain contiguous chunks of DNA, that satisfy known constraints, may code for genes. Genes are important because they in turn code for proteins. Proteins are linear strings of amino acids, from an alphabet of 20 characters, but, unlike DNA, these strings fold into complicated 3D shapes, capable of interacting with each other in a myriad of ways.

Proteins are three-dimensional objects, and if two proteins are said to interact this means that they can physically combine. Experiments can be conducted where every possible pair of proteins in the cell can be tested to see if a mutual interaction takes place. The resulting protein-protein interaction (PPI) network is simply an undirected graph whose the nodes are proteins and the edges denote observed interactions. The emergence of such data (admittedly often containing both false-positives and false- negatives) raised a number of intriguing network-theoretic questions: how should such networks be characterised? How did they evolve? Where (relatively to the whole) are the evolutionarily old and young proteins located? and do they contain small subnetworks of proteins that work together to produce common ends (cellular functions)?

Yeast two hybrid (Y2H) experiments allow biologists to measure, in a pairwise fashion, whether yeast proteins interact. The two hybrid system is based on the premise that many eukaryotic transcriptional activators consist of two physically discrete modular domains. The DNA binding domain of the transcription factor is expressed as a hybrid protein fused to protein P1 (the "bait"), while the activation domain is fused to protein P2 (the"prey"). The domains act as independent modules: neither alone can activate transcription. Only if proteins P 1 and P 2 interact will the activation domain be in the proper position to activate transcription of the reporter gene. PPI networks obtained this way are very noisy, experimental limitations are believed to result in a rate of at least 50% for both the false negative (missing interactions) and false positive (spurious interactions) rates.

In Figure 5 we plot the undirected (symmetric) sparse adjacency matrix of a PPI network for yeast based on the original (rather arbitrary) ordering given in data. Here, a dot in row i and column j indicates an interaction between proteins i and j. In this case there are $n = 1048$ proteins and 1029 interactions.

In Figure 6 we show how the PPI network from Figure 5 looks when it is reordered according to the Fiedler vector, $\mathbf{q}$, determined via the SVD. In linear algebra terms, we have applied a symmetric row/column permutation to the adjacency matrix. We see that the network as being made up largely of local interactions with relatively few long-range links, as proposed in range dependent networks. The blocks correspond to small groups of proteins likely top work together in creating cellular functions for the organism.

3.8 Calibration for general directed stochastic block models

Given an observed directed graph we wish to impose relaxed versions of some stochastic block model (SBM). The outcome will be a partition of the n vertices in the graph to exactly one of the corresponding into K subsets, called "blocks", by integers $\{1, 2, .., K\}$. This partition is defined more exactly by a mapping $z : \{1, ..., n\} \to \{1, ..., K\}$, such that the ith vertex lies within the kth block, where $k = z(i)$.

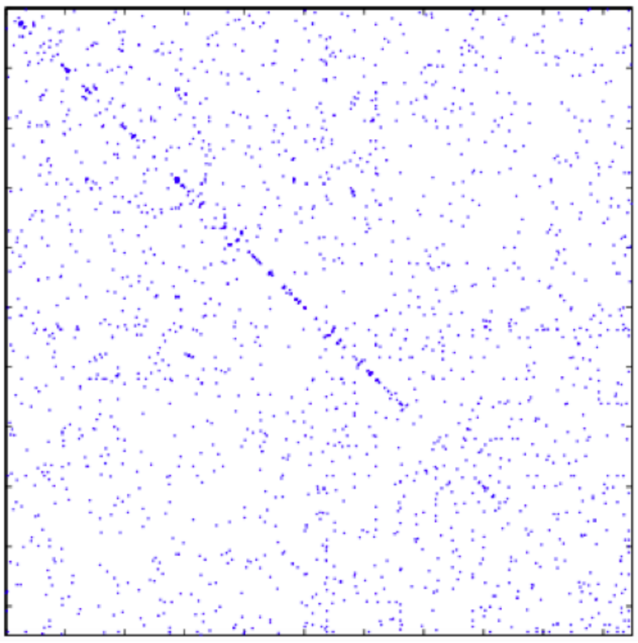


Figure 5: Original yeast PPI data adjacency matrix.

For $k = 1, .., k$, we let

$$n_k = |\{i | k = z(i)\}|, \quad k = 1, ..., K \tag{6}$$

be the number of vertices assigned to the kth block.

The model is completely defined by the partition along with a $K \times K$ probability matrix, P, containing where the $(k, \tilde{k})$th element, $p_{k,\tilde{k}} \in [0, 1]$, denoting the probability that an edge connecting a given vertex within block k to a given vertex within block $\tilde{k}$ is actually present within the network.

Suppose that we know both P and also the partition. Let $e_{i,j}$ denote the possible edge from vertex i to vertex j. Let E denote the actual edge set for the given observed graph. Then, given P and assuming the edges are present independently, the likelihood of the given graph is given by

$$\mathcal{L} = \prod_{e_{i,j} \in E} p_{z(i),z(j)} \cdot \prod_{e_{i,j} \notin E} (1 - p_{z(i),z(j)}),$$

This implies

$$\log \mathcal{L} = \sum_{e_{i,j} \in E} \log \left(\frac{p_{z(i),z(j)}}{1 - p_{z(i),z(j)}} \right) + \sum_{e_{i,j}} \log(1 - p_{z(i),z(j)}).$$

The first term is just the sum of the log-odds for each edge in E, while the second term sums over all possible edges, and may be rewritten as

$$\sum_{k=1}^{K} \sum_{\tilde{k}=1}^{K} n_k n_{\tilde{k}} \log(1 - p_{k,\tilde{k}}),$$

which is independent of z (other than via the n_ks). This is the same log-odds trick that we used in the last subsection.

This suggests the following two-step iterative algorithm.

If we are given z then we may estimate the n_k's directly using (6) and then estimate the $p_{k,\tilde{k}}$'s (and

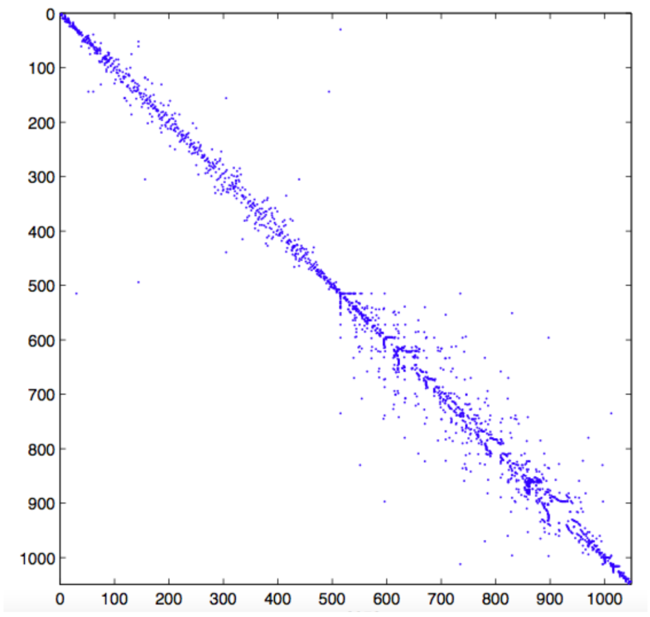


Figure 6: Reordered yeast PPI data adjacency matrix.

thus P) using Laplace's rule of succession [41],

$$p_{k,\tilde{k}} = \frac{1 + m_{k,\tilde{k}}}{2 + n_k(n_{\tilde{k}} - \delta_{k,\tilde{k}})}, \tag{7}$$

where $m_{k,\tilde{k}}$ is the total number of edges in E that go from a vertex in block k to a vertex block $\tilde{k}$ (the Kronecker delta arises since we do not allow any self-connecting loop edges).

Notice even if $m_{k,\tilde{k}} = 0$ we will still estimate $p_{k,\tilde{k}} = 1/(2+n_k(n_{\tilde{k}} - \delta_{k,\tilde{k}})) > 0$, and, similarly, even if all such edges are present ($m_{k,\tilde{k}} = n_k(n_{\tilde{k}} - \delta_{k,\tilde{k}})$) we will estimate $p_{k,\tilde{k}} = 1-1/(2+n_k(n_{\tilde{k}} - \delta_{k,\tilde{k}})) < 1$. Thus we avoid asserting any *certainty* even when there is no observed evidence to the contrary.

Alternatively if the n_k's and P are known, then we may update z so as to maximise the log-odds likelihood,

$$\log \hat{\mathcal{L}} = \sum_{e_{i,j} \in E} \log \left(\frac{p_{z(i),z(j)}}{1 - p_{z(i),z(j)}} \right). \tag{8}$$

This is usually very efficient since the networks we consider are sparse and this is a sum of log-odds over extant edges.

In order to maximise $\hat{\mathcal{L}}$ in (8) we proceed as in [39] (which in turn follows [5]) where a maximum likelihood approach was found to achieve the highest accuracy. We begin with the vertices randomly allocated across the K blocks, as an *initial allocation*. Then we consider each vertex in turn, keeping all of the others fixed within their present blocks, moving it into whichever block results in the greatest increase in $\hat{\mathcal{L}}$. We sweep through the vertices a number of times (less that five appears always sufficient) until none are re-allocated. This results in a near optimal solution, though the solution achieved certainly depends somewhat on the initial allocation.

Example: determining the partition
Here we consider a specific example, seeking z so as to maximise (8), where we have $K = 4$, and

$$P = \begin{pmatrix} 5.\,10^{-7} & 0.05 & 0.05 & 0.075 \\ 5.\,10^{-7} & 0.05 & 5.\,10^{-7} & 5.\,10^{-7} \\ 5.\,10^{-7} & 0.05 & 0.05 & 0.075 \\ 5.\,10^{-7} & 5.\,10^{-6} & 5.\,10^{-6} & 5.\,10^{-6} \end{pmatrix}.$$

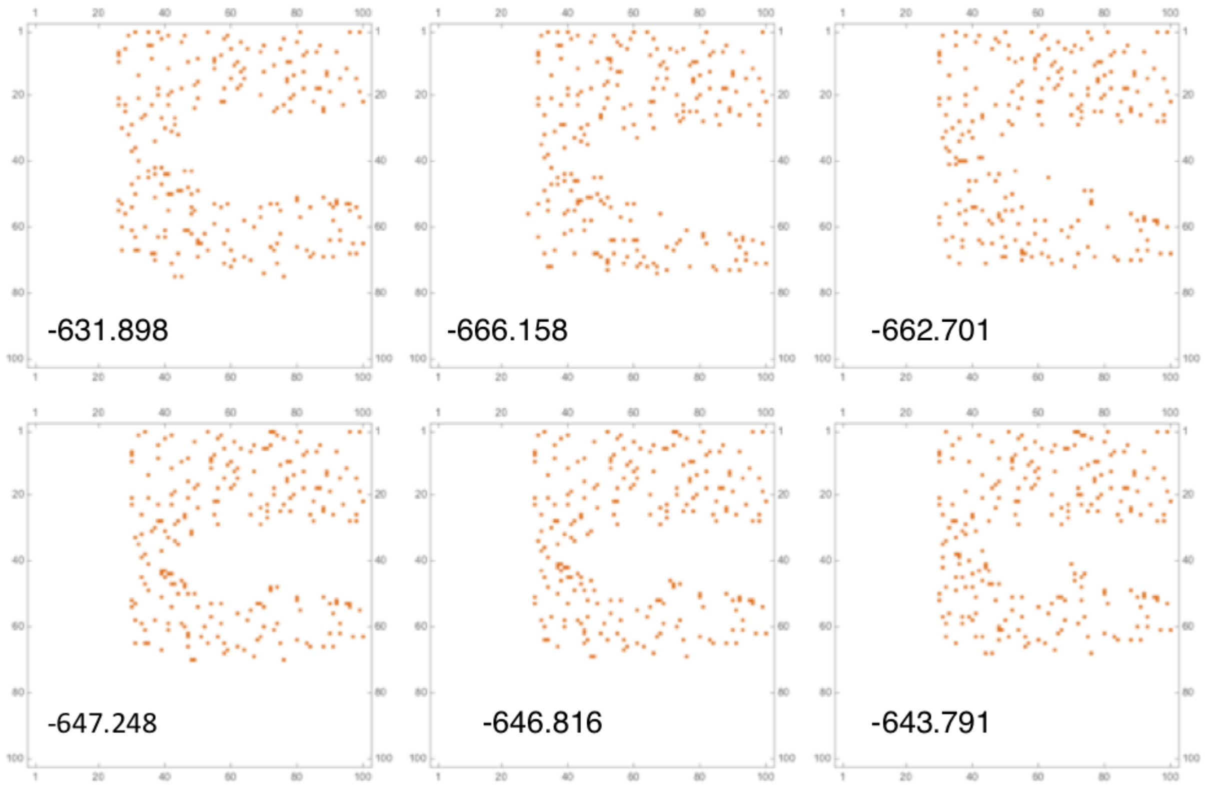


Figure 7: The data (top left) generated by a suitable core-periphery stochastic block model ($n = 100$, with 25 vertices in each of the four blocks, and the number of observed edges is $m = 224$) showing the natural log likelihood $\hat{\mathcal{L}}$, given in (8). We show five distinct attempts to maximise $\hat{\mathcal{L}}$ converging from distinct initial allocations.

We actually generated a block model with $n = 100$ and 25 vertices within each block, where the number of observed edges was $m = 224$ (out of 9900 possible edges).

In Figure 7 we show five results achieved for the block allocation step using distinct initial allocations, alongside of the original data. We also give the value of $\hat{\mathcal{L}}$ achieved in each case.

See [40] for further ideas within this area of analysis.

3.9 Sensitivity to false negative and false positive errors

How sensitive is this allocation step to edge set? Since $\hat{\mathcal{L}}$ is a sum over edges, we normalise it by dividing by $m = |E|$. Then we may successively delete edges, producing false negatives (edges we have not observed that should have been so), or indeed we can successive add in false positive edges (edges which are observed that should not have been so), and are not necessarily consistent with the original model.

Consider the impact on the log-odds/m, as m varies, shown in Figure 8. Obviously the existence of false negative edges (the deletion of actually edges) is less damaging than the addition false positive edges (edges which are mostly very low probability according to the model, P).

In fact, given the estimates made for n_k's the $p_{k,\tilde{k}}$'s we can calculate the expected log-odds for a single

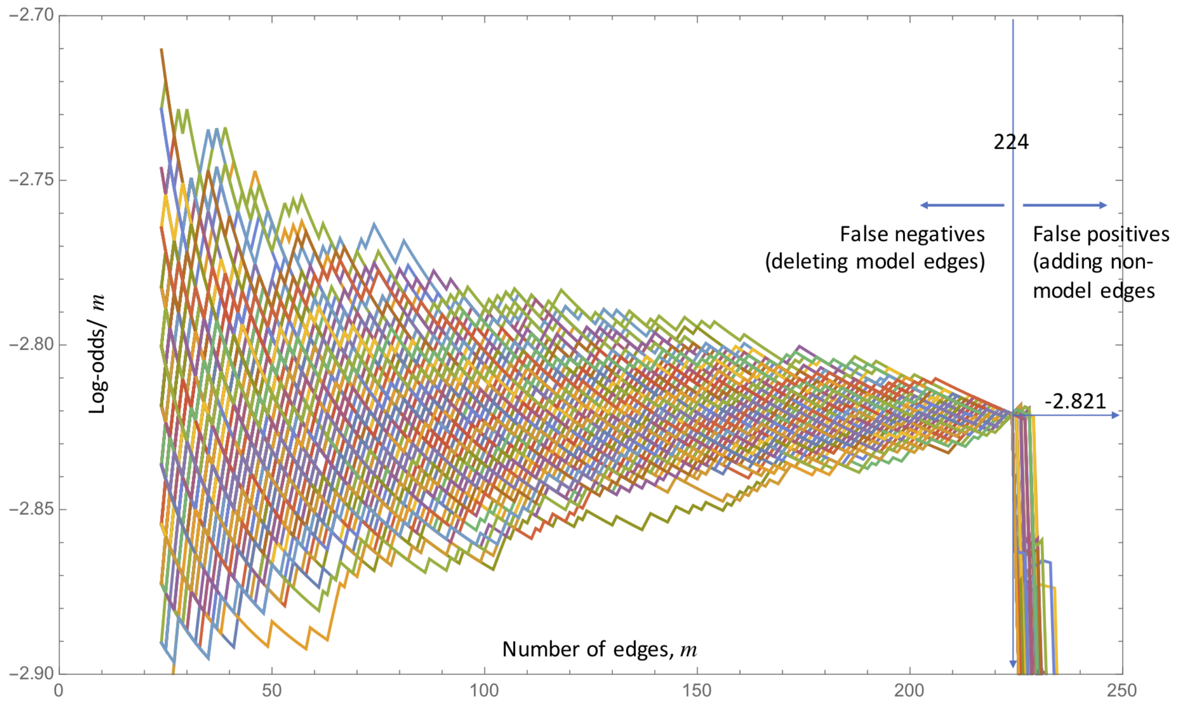


Figure 8: The change in $\hat{\mathcal{L}}/m$, as successive edges in E are deleted (producing false negatives), and m decreases; and as successive new edges are added to E (producing false positive edges, which are mostly inconsistence with the model, p), and thus m increases. We show the envelopes made up of 100 instances of sequential deletions and sequential additions, respectively. To the left we are sampling fewer *true edges* drawn from a distribution of trie edge log-odds values (and then taking the average), so the phenomena reflects the central limit theorem.

true edge or a single false (positive) edge:

$$\langle \log(p/(1-p)) \mid \text{true edge} \rangle = \frac{\sum_{k=1}^{K} \sum_{\tilde{k}=1}^{K} n_k n_{\tilde{k}} p_{k,\tilde{k}} \log(p_{k,\tilde{k}}/(1-p_{k,\tilde{k}}))}{\sum_{k=1}^{K} \sum_{\tilde{k}=1}^{K} n_k n_{\tilde{k}} p_{k,\tilde{k}}},$$

and

$$\langle \log(p/(1-p)) \mid \text{false edge} \rangle = \frac{\sum_{k=1}^{K} \sum_{\tilde{k}=1}^{K} n_k n_{\tilde{k}} (1-p_{k,\tilde{k}}) \log(p_{k,\tilde{k}}/(1-p_{k,\tilde{k}}))}{\sum_{k=1}^{K} \sum_{\tilde{k}=1}^{K} n_k n_{\tilde{k}} (1-p_{k,\tilde{k}})}.$$

For the above SBM example, in Figure 8 we have

$$\langle \log(p/(1-p)) \mid \text{true edge} \rangle = -2.82... \;\; \langle \log(p/(1-p)) \mid \text{false edge} \rangle = -8.97... \, .$$

In summary we see that then instance of even a few false positive edges is very corrosive for $\hat{\mathcal{L}}/m$, the overall log-likelihood per edge (the consequences of erroneous observations/interpretations), whereas even a largish number (50%) of false negatives (possible true edges which were not observed) do not make very much difference to the the overall log-likelihood per edge (since the remainder are consistent with the model).

4 COMMUNITY DETECTION

Many networks exhibit community structure. Community structure implies that the network is composed of groups (communities) of vertices that are densely connected within the same group (the same community) and yet are relatively sparsely connected across different groups (communities). It is often termed a "network of networks" structure. We have seen how models such a Stochastic Block models can be deployed give rise to such structure.

Suppose we are give a network: how do we best divide it up into sub communities that are relatively densely connected internally and relatively sparsely connected from community to community.

There are many algorithms aiming to detect community structure in a given network in the absence of predefined labelling of vertices. In this section, after giving an introduction on the related problem of graph partitioning, we introduce community detection methods based on the notion of **modularity**.

The problem of **graph partitioning** has a long tradition in computer science and has important applications for parallel or distributed computation. It consists in dividing the vertices of a network into a predefined number of groups such that the number of edges between groups is minimised. Problems of this type can be solved in polynomial time, but with a prohibitive complexity of n^{c^2}, where n is the number of nodes and c the number of groups. For practical applications, approximate methods have been developed, among which the popular spectral partitioning method, due originally to Fiedler. That divides up the vertices using the eigenvalues and eigenvectors of the combinatorial Laplacian.

4.1 A spectral method

Consider a case where we consider the simplest instance of the problem, with $c = 2$, thus consisting in finding the best bi-partition of the vertices in a strongly connected undirected network, such that the number of edges between the two communities is minimised.

Let A be the $n \times n$ adjacency matrix of our undirected graph, as usual. For a partition into two communities, denoted by B_1 and B_{-1}, let s_i denote the community that vertex i belongs to: so that $v_i \in B_{s(i)}$.

By definition, given any bi-partition, the number of edges R running between the two communities of vertices, also called the cut size, is given by

$$R = \frac{1}{2} \sum_{i,j|s(i)\neq s(j)} A_{ij},$$

Let $\mathrm{s} = (s_1, \dots, s_m)^T$. Then $\mathrm{s}^T.\mathrm{s} = n$, the number of vertices, since all elements of s are equal to ± 1.

The expression

$$\frac{1}{2}(1 - s_i s_j)$$

is equal to zero if v_i and v_j are in the same community and equal to one if they are in different communities.

Thus

$$R = \frac{1}{4} \sum_{i,j} (1 - s_i s_j) A_{ij},$$

which, using

$$\sum_{i,j} A_{ij} = \sum_{i,j} \delta_{i,j} d_i = \sum_{i,j} s_i s_j \delta_{i,j} d_i,$$

implies

$$R = \frac{1}{4} \sum_{i,j} s_i s_j (d_i \delta_{i,j} - A_{ij}).$$

Here, as usual, d_i is the degree of vertex v_i and $\delta_{i,j}$ is the Kronecker delta.

We can write this in matrix (quadratic) form as

$$R = \mathbf{s}^T L \mathbf{s}$$

where L is the combinatorial Laplacian introduced earlier in section 3.5.

If we *relax* the requirement that s be a vector of elements equal to ± 1, and instead allow s to be real and assign the vertices simply by the sign of the corresponding s_i, then we see that R is minimised by choosing s to be (a suitably normalised version of) the first eigenvector corresponding to the first non-zero eigenvalue, called the Fiedler eigenvalue (see section 3.5). The smallest eigenvalue of L of course is always zero, but for a connected graph it is simple and its eigenvector contains elements that are all equal to one, so is not a useful solution here. The Fiedler eigenvector minimises R subject to $\sum_i^n s_i = 0$.

Relaxation is often used in numerical; methods, to reduce hard combinatorial problems to linear algebra and spectral theory.

Hence one simply takes the Fiedler eigenvector (corresponding to the smallest non-zero eigenvalue) of the combinatorial Laplacian. Note that its elements must sum to zero (why? the eigenvectors of a self-adjoint matrix are orthogonal). One partitions the vertices according to whether the correspond elements of the Fiedler eigenvector are positive or negative.

As an example we take a Stochastic Block Model (SBM) on $n = 20$ vertices with two blocks, each of size 10 vertices (containing vertices 1 thru 10, and vertices 11 thru 20, respectively) with the intra-block edge probability equal to 0.5 and the inter-block edge probability equal to 0.2. Note that these probability values are not too dissimilar. If we reduce the inter-block probability to 0.1 the problem become too easy.

In Figure 9 we show the results for this method for a particular instance of the above SBM: in this case two vertices are misclassified. Below we will introduce an alternative approach (maximising) which is more accurate.

The advantage of this spectral method is that it is relatively inexpensive in terms of computations and really just requires us to locate the Fiedler eigenvector. It therefore deploys well worn ideas from numerical linear algebra. It is fast and reliable, when n become very large.

How might we generalise this if we require $c > 2$? What what is the optimum value for c? Here is an alternative method that generalises well.

4.2 Modularity

In Figure 10 we show a large connected component of an online friendship network for citizens of Bristol, a historical port and city in the UK (it is the Twitter "reciprocated mentions" network for

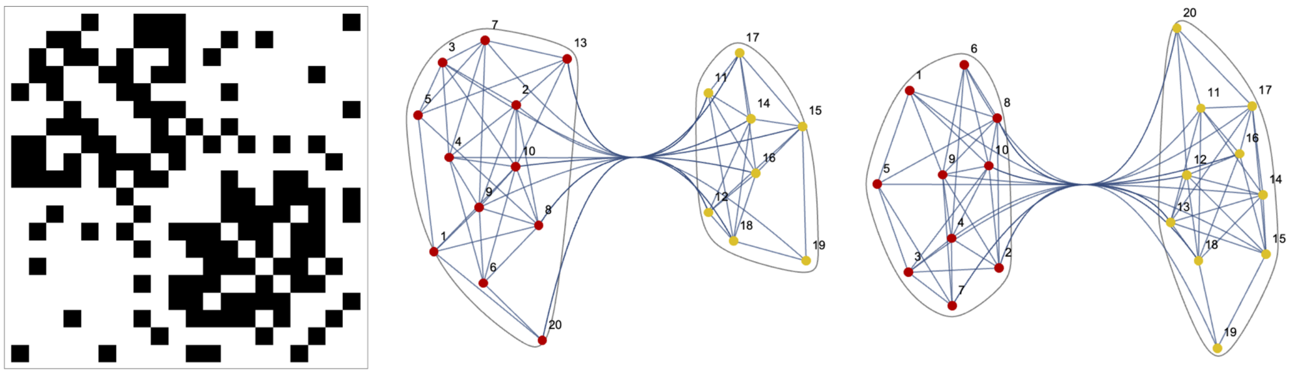


Figure 9: Left: the $n \times n$ ($n = 20$) adjacency matrix for the SBM network: black entries indicate 1s (see text). Centre: bi-partition via the Fiedler vector - two vertices misclassified. Right: bi-partition via maximum modularity (see section 4.2) - no vertices misclassified.

citizens of Bristol [28]). This is partitioned into 74 communities. Here there are 2892 vertices, with average degree 3.14. Similar data is available for the largest 10 cities in the UK [10].

In the graph partitioning problem, so far we have implicitly have had to fix the number of groups. For instance, the spectral partitioning method described before does not provide ways to determine these quantities; this is an input of the algorithm. The community detection problem relaxes these constraints and aims at finding the best partition of a network into communities, whichever their number of their size. The idea is that the structures present in the network should guide the algorithm to the right partition.

Modularity, denoted by Q, is a quantity introduced to measure the goodness of the partitioning of a network into communities. Like the cut size, this quantity is often used as an objective function to be optimised in order to uncover the best partition of a network. The main ad- vantage of modularity over other quality functions for node partitioning is that it allows us to compare partitions made of different numbers of communities. Let us consider a subset of vertices, denoted by CM (standing for a "ComMunity"). The underlying idea of modularity is to compare the number of edges connecting vertices within CM with the expected number of edges in an appropriate null model. Under the configuration model, we saw earlier in (5) that the probability that nodes v_i and v_j are adjacent is given by

$$P_{ij} = \frac{d_i d_j}{2m}.$$

Other choices for the null model P_{ij} have also been considered. We quantify the contribution of CM to Q as

$$\sum_{v_i, v_j \in CM} \left(A_{ij} - \frac{d_i d_j}{2m} \right).$$

Let us now consider a partition of the network into n_{CM} disjoint communities. The cth community $(c = 1, 2, ..., n_{CM})$ is denoted by CM_c.

Modularity is defined as a properly normalised sum of over all communities:

$$Q = \frac{1}{2m} \sum_{c=1}^{n_{CM}} \left[\sum_{v_i, v_j \in CM_c} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \right].$$

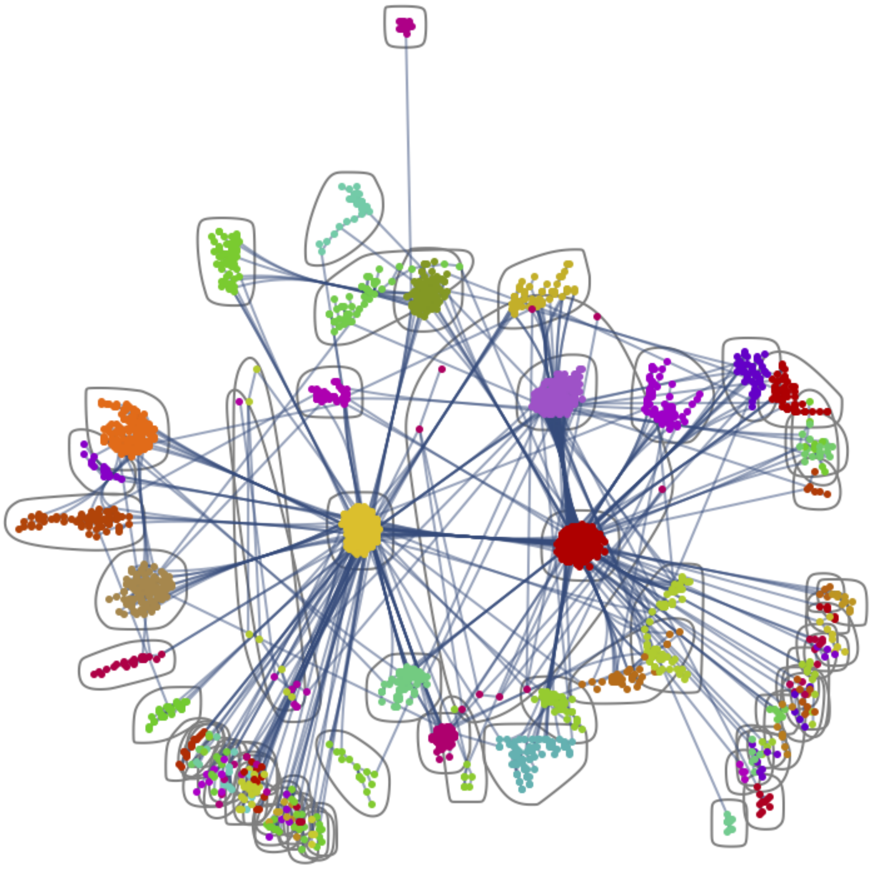


Figure 10: An observed friendship network, formed via Twitter reciprocated mentions, for $n = 2892$ citizens of Bristol [28]; partitioned into 74 communities. This is the optimum partition in maximising modularity (see section 4.2), according to the Mathematica implementation.

We may also write this as

$$Q = \frac{1}{2m} \sum_{ij}^{n} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \delta(s_i, s_j),$$

where s_i denotes the community that v_i belongs to: so that $\delta(s_i, s_j) = 1$ iff v_i and v_j belong to the same community, and is equal to zero otherwise.

Now the idea is to maximise Q over all possible partitions of the vertices. However, the problem is not that simple, because there are so many possible partitions, as we will discuss later.

If, for a moment we restrict the partitions to be a bi-partition, then this Modularity method usually does as well as, or better than, the inexpensive spectral method, see Figure 9.

The Bristol example in Figure 10 uses the modularity maximisation method encoded within the commercial Mathematica package. It partitions Bristol into 74 communities. Any more or any fewer communities, even when tweaked and optimised, would result in smaller values of Q.

Modularity ranges in $[-0.5, 1]$. The trivial partition into one large community always yields

$$Q = \frac{1}{2m} \sum_{ij}^{n} \left(A_{ij} - \frac{d_i d_j}{2m} \right)$$

$$= \frac{1}{2m} \left(2m - \frac{\sum_{i=1}^{n} d_i \sum_{j=1}^{n} d_j}{2m} \right)$$

$$= \frac{1}{2m}\left(2m - \frac{2m \times 2m}{2m}\right) = 0.$$

Exercise Write a function that takes a graph and its partition as an input and returns its modularity. Create an image graph the and colour the vertices by the various elements in the partition (as in Figure 9 and Figure 10)?

4.3 Spectral optimisation of modularity

Optimising the modularity partition of a network into communities is far from trivial, as it was proved to be NP-hard (there are c^n ways of producing a c-element partition), and so various approximate optimisation methods have been designed.

In this section, we present an approach based on the spectral properties of the network, similar in spirit to the spectral partitioning method. For the sake of simplicity, we consider the division of a network into just two communities (division into more communities can be then obtained recursively). Our aim is thus to find the best bipartition of the network, that is the bipartition that optimises modularity. As before, we denote a potential such division by an in- dex vector s with elements as above, satisfying

$$\delta(g_i, g_j) = \frac{1}{2}(s_i s_j + 1).$$

where g_i and g_j are the communities containing v_i and v_j. So thjen

$$Q = \frac{1}{4m}\sum_{ij}^{n}\left(A_{ij} - \frac{d_i d_j}{2m}\right)(s_i s_j + 1) = \frac{1}{4m}\sum_{ij}^{n}\left(A_{ij} - \frac{d_i d_j}{2m}\right) s_i s_j,$$

where as usual, we have used the handshaking lemma.

Hence in matrix form we have the quadratic form:

$$Q = \frac{1}{4m}\mathbf{s}^T B \mathbf{s},$$

where B is the real symmetric matrix

$$B_{ij} = A_{ij} - \frac{d_i d_j}{2m}.$$

This is called the modularity matrix. All rows (and columns) of the modularity matrix sum to zero, which implies that the vector $(1, 1, 1, \dots)^T$ is an eigenvector with eigenvalue zero, just as is the case with the Laplacian. Unlike the Laplacian however, the eigenvalues of the modularity matrix are not necessarily all of one sign and in practice the matrix usually has both positive and negative eigenvalues.

Equation is the equivalent of that above for the cut size and similar matrix methods can be applied to modularity optimisation (in this case maximise Q, rather than minimise R). By direct analogy, we write s as a linear combination of the normalised eigenvectors $\mathbf{u}_i$ of B:

$$\mathbf{s} = \sum_{i=1}^{n} a_i \mathbf{u}_i \;\text{ with } a_i = \mathbf{u}_i^T \mathbf{s},$$

so that

$$Q = \sum_{i=1}^{n} \beta_i a_i^2,$$

where β_i is the eigenvalue of B corresponding to the eigenvector $\mathbf{u}_i$.

Note that $\mathbf{s}$ is normalised:

$$n = \mathbf{s}^T.\mathbf{s} = \sum_{i=1}^{n} a_i^2.$$

Assume that the (necessarily real) eigenvalues are labeled in decreasing order $\beta_1 \geq \beta_2 \geq \ldots \beta_n$.

Modularity optimisation thus becomes equivalent to choosing the quantities a_i^2 so as to place as much as possible of the weight in the terms corresponding to the largest (most positive) eigenvalues.

As with ordinary spectral partitioning, in the previous subsection, this would be a simple task if our choice of $\mathbf{s}$ were unconstrained (apart from normalisation): we would just choose $\mathbf{s}$ proportional to the leading eigenvector $\mathbf{u}_1$ of the modularity matrix. But the elements of $\mathbf{s}$ are restricted to the values ± 1.

As before, we *relax* the problem good approximate solutions can be obtained by choosing $\mathbf{s}$ to be as close to parallel with $\mathbf{u}_1$ as possible, which is achieved by setting s_i equal to the sign of the ith element of $\mathbf{u}_1$.

This is our first and simplest algorithm for modularity-based community detection: we find the eigenvector corresponding to the most positive eigenvalue of the modularity matrix B and divide the network into two groups according to the signs of the elements of this vector. Importantly, the separation between positive and negative entries determines the optimal sizes of the communities. Moreover, finer divisions are obtained by applying the algorithm re- cursively. The method thus produces a set of partitions, with an increasing number of partitions. Modularity can then be used to find, among those, the best partition, and thus determine the right number of modules. The magnitudes of the elements of the eigenvector $\mathbf{u}_1$ also contain useful information about the network, indicating, the "strength" with which vertices belong to the communities into which they are placed.

4.4 Louvain method to optimisation of modularity

The spectral method presented in the previous section is divisive, as it proceeds by divising the network into smaller parts until reaching an optimal partition. The implementation of optimising Modularity in the Mathematica package works that way (but is not discussed openly). One of its limitations is computational, as its implementation requires the estimation of eigenvectors of the modularity matrix, which is prohibitively expensive for systems much beyond 10^4 nodes, in general. For such systems, alternative methods have been designed, often based on greedy, agglomerative principles. In this section, we describe one such method, the Louvain method, implemented in many libraries and packages.

The Louvain method consists of two phases, which are iteratively repeated, until a local maximum of modularity is obtained. The algorithm begins with an undirected weighted graph having n vertices to which an index between 1 and n has been randomly assigned. It is then designed as follows.

First phase: local optimisation. The initial partition consists of placing each vertex into a separate community, this partition is therefore composed of n singleton communities. We then consider the first vertex, v_1, and calculate the modularity variation obtained by removing v_1 from its community and placing it in the community of one of its adjacent neighbours, v_j say. This variation is therefore calculated for each of the neighbours of v_1 and it is then moved to the community where this increase is maximum, but only if this maximum increase is positive. If all the increases are negative, then v_1 is put back into its original community. This process is applied sequentially, that is the process is then reapplied to all the vertices repeatedly until no vertex is moved during a complete iteration. The first phase is then finished. We stress the fact that there are generally several iterations (i.e. after vertex v_n, one returns to v_1, and so on) and this phase ends when a local maximum of modularity is reached, meaning that no individual movement can increase the modularity. After this first phase, the network of n vertices has been divided in a partition P having n_c communities. If $n > n_c$, meaning if the first phase has grouped some vertices, then the algorithm continues to the second phase, if not the algorithm is finished and the result is the partition P.

Second phase: merging of vertices. The second phase consists in constructing a new graph whose vertices are the n_c communities discovered during the first phase. The weight of the edges between two of these new vertices is given by the sum of the weights of the edges which existed between the vertices of these two communities. The links which existed between the vertices of a same community create loops over the community in the new graph. Once this second phase is finished, it is possible to reapply the first phase of the algorithm on the weighted graph and to iterate.

A single combination of the two phases is usually called a “pass”. The first phase consists in finding a local optimum, where each vertex can only be linked to one community in its direct neighbourhood. The second phase consists in aggregating the vertices, such that the application of the first phase on the aggregate graph will lead to collective movements of vertices at a higher level.

This repetition of passes reminds us of the concept of self-similarity of complex network and naturally constructs a hierarchy of communities. The output of the algorithm is therefore a set of partitions, one per pass, such that the average size of the communities and the modularity increase from one pass to another. By construction, the partition found after the last pass is the one maximising modularity, and it is the main outcome of the algorithm, but the hierarchy provided by the algorithm can also be exploited to characterise the hierarchical structure of the network.

4.5 Limitations of modularity optimisation

Methods based on modularity maximisation suffer from several drawbacks.

First, by construction, they are not capable of uncovering overlapping communities, often observed in empirical networks.

Second, Q exhibits a resolution limit, because using Q it is impossible to detect dense clusters of nodes that are smaller than a certain scale. The resolution limit originates from the dependency of the null model on $2m$. The dependency decreases when the number of links, m, is increased. Then, modularity maximisation tends to favour larger communities. In the limit $m \to \infty$, the null model is neglected and modularity optimisation simply uncovers the connected components. Modularity-based methods implicitly favour communities having a certain size, depending on the size of the entire network, not only on its internal structure.

Third, the modularity landscape is usually extremely rugged and degenerate such that there exists an exponential number of alternative, high-scoring partitions. AS one approaches the actual optimum, using an interactive method, the final few iterations yield very marginal increments. See Figure 11, for example, considering Bristol’s reciprocated Twitter mentions network, considered earlier in Figure 10

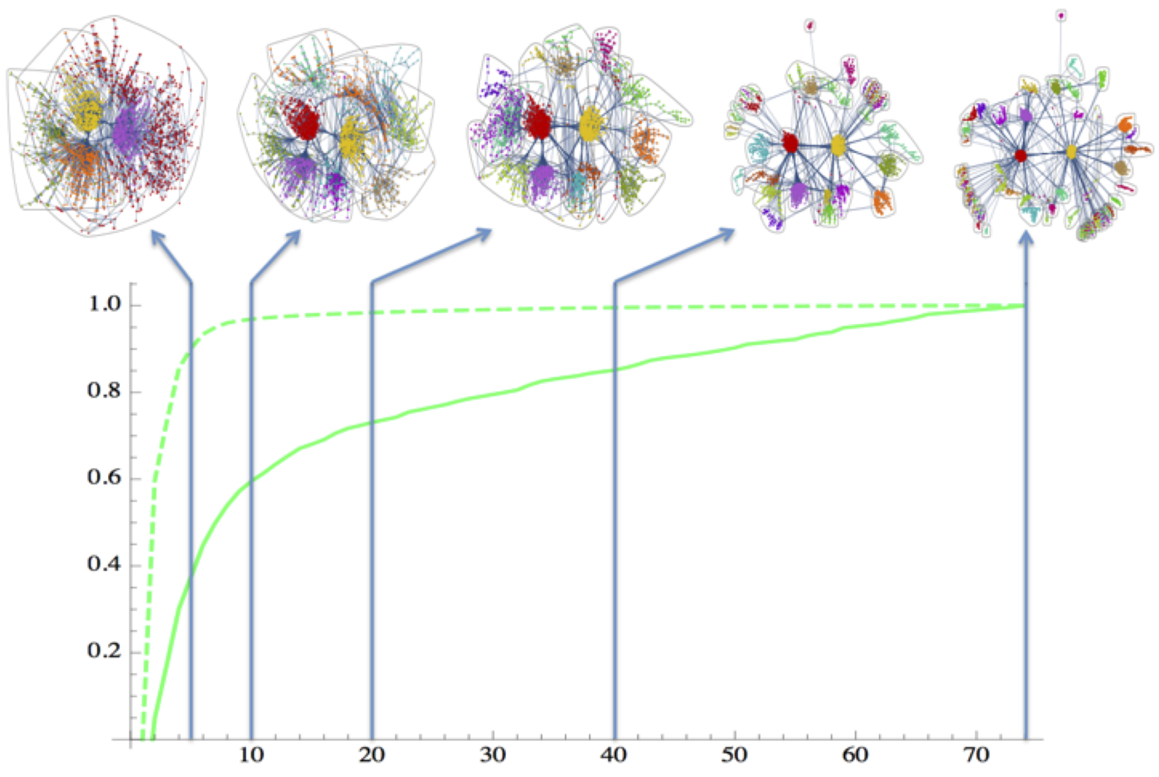


Figure 11: Successively increasing the size of the partition for Bristol's reciprocated Twitter mentions network, considered in Figure 10. The dashed line denotes the relative modularity Q. Clearly the final gains are highly marginal — perhaps we are working too hard. The solid line denotes the entropy. Taken from [28]

Finally, although modularity allows us to compare partitions of the same network, it is by no means intended to compare modularity values of different networks. Therefore, Q should not be used as an absolute measure of the modularity of a network. For instance, the modularity of the best partition of a random network tends to $Q = 1$ when the network is sufficiently large, whereas this network is by no means modular.

5 DYNAMICALLY EVOLVING NETWORKS

5.1 Katz Centrality for Observed Evolving Networks

Let A be an $n \times n$ symmetric adjacency matrix, for an undirected graph, as usual (no loops, no double edges).

Consider the Katz centrality matrix, that we met before,

$$Q = I + \alpha A + \alpha^2 A^2 + \alpha^3 A^3 + \ldots = (I - \alpha A)^{-1}.$$

Recall that this is a weighted count of all the walks between pairs of vertices, with a weight of α^l for a walk of length l.

Here α must be small enough so that the geometric sum exists. To be precise we must have $\alpha\rho(A) < 1$, where $\rho(A)$ is the spectral radius (the Peron-Frobeneous eigenvalue) of A.

Here we wish to generalise the notion of vertex-to-vertex Katz centrality that we defined earlier for static networks, to apply to evolving (sequential) two way communications networks.

We follow [2]. We consider evolving networks that represent time stamped peer-to-peer communications. In dealing with such networks and we have to consider walks through them with successive edges being at the same or later time steps.

Sending messages across such networks is obviously asymmetric, even when each individual communication is symmetric. If I talk with you today and you talk with Ramona tomorrow, then I could have sent a message via you to Ramona, but she could not have sent a message to me. So unlike static networks, the passage of time produces asymmetries and good senders/transmitters are not necessarily the same as good lis- tens/receivers.

Suppose that we wish to find out who is a good transmitter of information through an evolving network, and who is a good receiver, who has access to all the information being passed around an evolving network. Perhaps we wish to make some intervention. For a static network these properties were the same, and vertices could be rank ordered by the row/column sums of the symmetric centrality matrix. Yet now this will not be the case here.

Consider an evolving network (represented by its sequentially evolving adjacency matrix), say $\{A_k\}$, for $k = 1, 2, \ldots, K$ on an enumerated set of vertices, $\{1, 2, \ldots, n\}$. Each of the A_ks is an $n \times n$ symmetric adjacency matrix, for an undirected graph, as usual (no loops, no double edges). We will say the evolving network is equal to A_k at time step t_k.

A dynamic walk of length m from vertex i_1 to vertex i_{m+1} is a sequence of edges $(i_1, i_2), (i_2, i_3), \ldots (i_m, i_{m+1})$ and a non-decreasing sequence of times

$$t_{r_1} \leq t_{r_2} \leq \ldots \leq t_{r_m},$$

such that the edge (i_j, i_{j+1}) is traversed at time t_{r_j}. This is equivalent to saying that

$$(A_{r_j})_{i_j i_{j+1}} = 1.$$

Hence the dynamical walk is made up of m successive edges, each drawn from the evolving network at some suitable non-decreasing set of time steps. In this definition we can use successive edges at the same time or at later times.

The time-dependent context offers a rich variety of alternatives. In some circumstances it may be appropriate to allow at most one edge to be used per time point, so the constraint on the time sequence would involve strict inequality $t_{r_1} < t_{r_2} < \ldots < t_{r_m},$. For example, at a formal evening ball, each time the band plays a dance, you may partner-up with an acquaintance or sit it out. (No cutting in is allowed!). How could you spread a rumour from partner to partner?

Alternatively, it may be appropriate to force exactly one edge to be used per time point. Analogous definitions of dynamic paths and dynamic trails could be made.

Here we will keep to our original definition $t_{r_1} \leq t_{r_2} \leq \ldots \leq t_{r_m},$, so that any number of the edges could be selected for a single time step, but the ordering of the edges **cannot go back in time**.

The dynamic walk concept proceeds via the observation that the matrix product

$$A_{r_1}A_{r_2}\ldots A_{r_m}$$

has (i, j)th element that counts the number of dynamic walks of length m from vertex v_i to vertex v_j, where the jth step of the walk takes place at time t_{r_j}. This is really a simply corollary of a result we considered earlier, when we had walks in a network and then add an edge from a "next" network. So recall that each time we take a successive step, we can count the possible walks by post-multiplying by the right adjacency matrix.

Consider all of the possible dynamical walks from v_i to v_j that take exactly $m_k \geq 0$ edges at each time step $k = 1, ..., K$. Such a walk is of total length $m = m_1 + ... + m_K$, where the walk traverses exactly $m_k \geq 0$ edges from A_k. The number of such walks is given by the (i, j)th term of the matrix product

$$A_1^{m_1}A_2^{m_2}\ldots A_K^{m_K}.$$

Just as in the static network case we want to count up all such walks that are possible (by varying the m_k), though we will discount each walk according to α^m, where m is the total length.

It follows that the matrix product

$$(I + \alpha A^1 + \alpha^2 A_1^2 + ...)(I + \alpha A_2 + \alpha^2 A_2^2 + ...)...(I + \alpha A_K + \alpha^2 A_K^2 + \ldots)$$

contains all such terms. So its (i, j)th term counts all possible walks of all possible combinations of edges taken from the successive time steps, suitably discounted for total length.

Thus the matrix product

$$Q = (I - \alpha A_1)^{-1}(I - \alpha A_2)^{-1}\ldots(I - \alpha A_K)^{-1}$$

is called the Katz centrality matrix for the evolving network.

In order that Q is well defined the infinite sums (the inverses deployed must all exist). This requires that $\alpha < 1/\rho(A_k)$ for all $k = 1, 2, ..., K$. Hence α is bounded above by the inverse of the maximum spectral radius of the adjacency matrices.

It is non symmetric in general as matrix multiplication is not commutative.

Let $\mathrm{s} = (1, 1, \ldots, 1)^T$. The row sums,

$$\mathbf{b} = Q.\mathrm{s},$$

count the the outward influence of each vertex within the network (the weighted sum of walks from each vertex to anywhere). These are called "broadcast-centralities". The column sums,

$$\mathbf{r} = Q^T.\mathrm{s},$$

count the the inwards influence of the network upon each vertex (the weighted sum of walks from anywhere to each vertex). These are called “receive-centralities”.

Corporate email Communications as an Evolving Network
One widely studied time-dependent interaction data set in the public domain lists the email activities of 151 Enron employees (see `http://www.cs.cmu.edu/enron/`).

We first summarise all of the employee-employee email interactions over a period of 1138 consecutive days. This leads to a sequence of 1138 symmetric adjacency matrices, each of dimension 151×151.

Figure 12 shows the total number of edges per day. This gives rise to 1138 single day adjacency matrices, we know that many of them are empty. We contrast the centrality vectors, $\mathbf{b}$ and $\mathbf{r}$, with the total vertex degrees (the sum of degrees over all days).

Figure 13 shows the broadcast-centralities and receive-centralities. Interestingly we know the actual identities of the individuals in this data set as it is a matter of public record (due to the legal proceedings). Obviously those with high degrees have a likelihood of having high centralities. But it depends how they are distributed within the network, and when (earlier or later in the sequences).

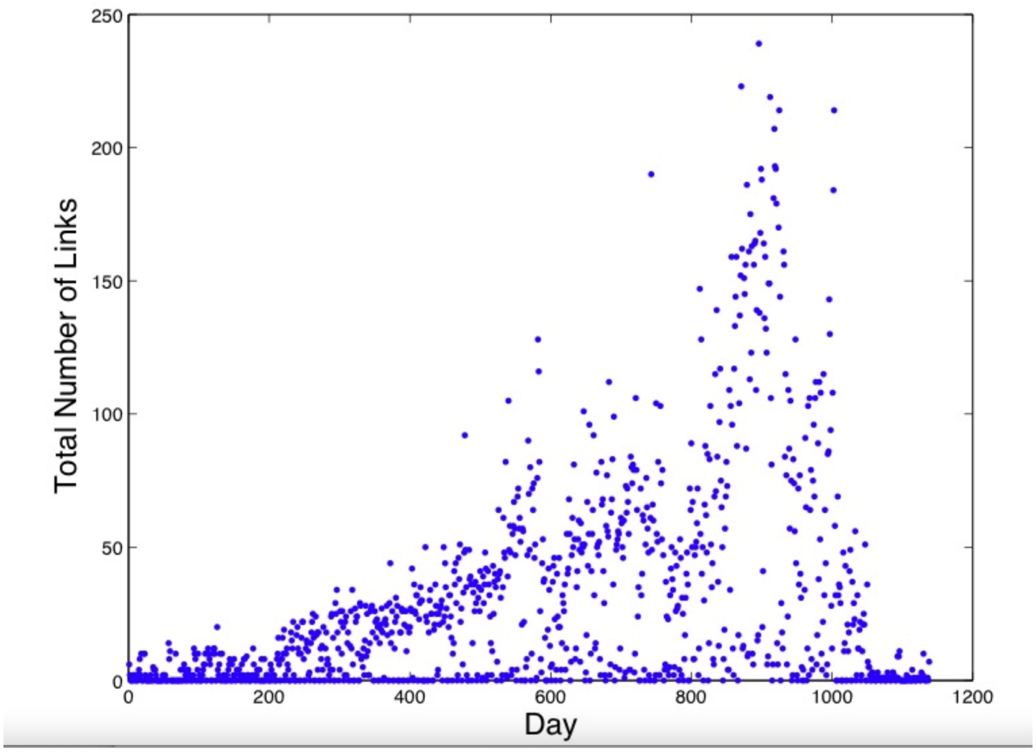


Figure 12: Daily edge counts in an observe evolving network.

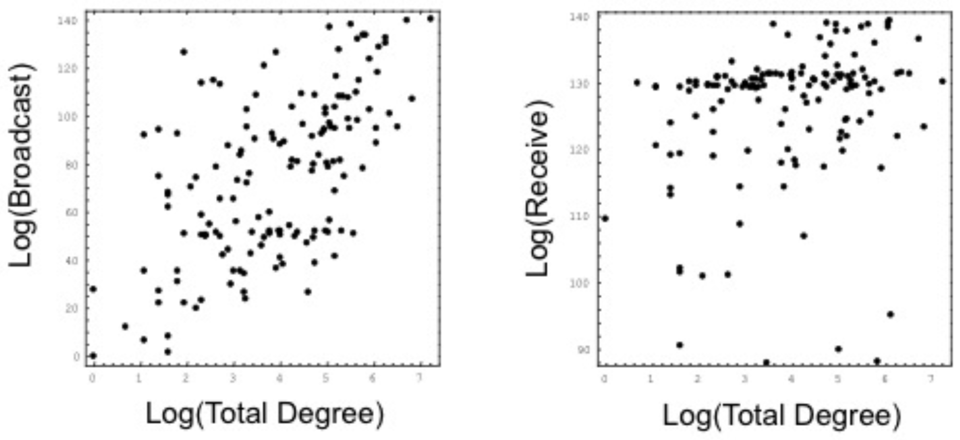


Figure 13: Vertex scatterplots of broadcast-centrality, $\mathbf{b}$, and receive-centrality, $\mathbf{r}$, against total vertex degree. Some vertices with small total degrees are of high centrality: influencing everybody or being influenced by everybody over time.

Example: A Mobile Telephone Call Network We consider the 52-week "Reality Mining" data set as an evolving network with $n \times n$ adjacency matrices $K = 52$ and $n = 106$. Specifically we used the data on voice calls made between pairs of subjects (106 members of MIT faculty) to derive the weekly adjacency matrices. We depict the first 48 weeks of the sequence below. It is very sparse at both the beginning and the end of the sequence, which represents an academic year starting in July. See Figure 14.

So for Q to be well defined, $1/\alpha$ must be larger that the spectral radius (the Perron Frebeneous eigenvalue) of each of the A_k (larger than 17.456 in this example, which occurs when the evolving network is relatively less sparse).

Next in Figure 15 we plot source (broadcast)-centralities held in $\mathbf{b}$ versus sink(receive)-centralities held in $\mathbf{b}$. There are clearly some vertices that are much better as broadcasters/sources than receivers/sinks and vice versa.

5.2 Nonlinear Dynamics: an Evolving Network Model

A network that evolves over time can be generated by a nonlinear Markov model. We follow [11].

Consider a population of n individuals connected through a dynamically evolving undirected network representing person-to-person voice calls or online chats. Let $A(t)$ denote the $n \times n$ binary symmetric adjacency matrix for this network at time t, having a zero diagonal (you cannot talk to yourself). The edges if $G_{A(t)}$ are appearing and disappearing over time

Let S denote the set of real, symmetric, $n \times n$ matrices with all elements taking values in [0,1], having a diagonal containing only zeros. Then at any time the expected value for the adjacency matrix B of an undirected random graph, denoted by $\langle B \rangle$, lies in S. We have considered random graphs earlier in section 3. If we assume that the edges in G_B are independent then we may generate an instance by examining each possible edge, (i, j), in turn and having it present with probability $\langle B \rangle_{ij}$ (since the graph undirected, so B (and $\langle B \rangle$) is symmetric, and we only need to consider the under triangular

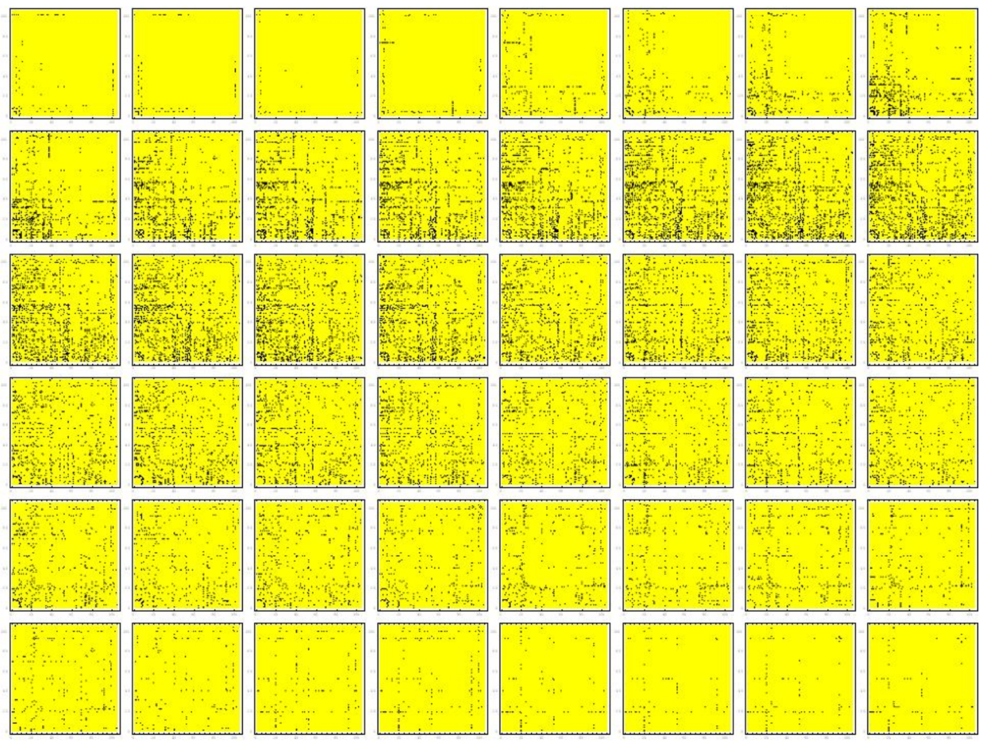


Figure 14: Weekly edge counts in an observe evolving network.

part of $\langle B \rangle$).

Our simplest model for $G_{A(t)}$ assumes that it is known up until the present moment, t and then at future times $A(t+t+\delta t)$, and this the graph, is a stochastic object, representing a random graph (see section 3) defined by a conditional probability distribution over A (the set of all possible adjacency matrices). Each edge within this network will be assumed to evolve independently over time, though it is conditionally dependent upon the current network, so multiple edges may be conditional upon some common parts of the current structure. Rather than model a full probability distribution over A for future network evolution, conditional on its current structure, say $P(A(t+\delta t)|A(t))$, it is enough to specify its expected value $\langle A(t+\delta t)|A(t)\rangle$, a matrix containing all edge probabilities, from which edges may be generated independently.

Now we specify our conditional model for the stochastic network evolution via an equation of the form

$$\langle A(t+\delta t)|A(t)\rangle = A(t) + \delta t F(A(t))$$

valid as $\delta t \to 0$. Here the real matrix-valued function F is assumed to take values in S, and thus it is symmetric, it has a zero diagonal, and all elements within the right hand side must be in [0,1] (since they are independent edge probabilities).

By convention we use the rule into determine the independent edges in the upper triangular part of $\langle A(t+\delta t)|A(t)\rangle$, and then impose symmetry.

We shall fix F more specifically, and write it in the form

$$F(A(t)) = -A(t) \circ \omega(A(t)) + (\mathbf{1} - A(t)) \circ \alpha(A(t)).$$

Here $\mathbf{1}$ denotes the adjacency matrix for the clique, where all possible $n(n-1)/2$ edges are present (all elements are ones except for zeros on the diagonal), so that $\mathbf{1} - A(t)$ denotes the adjacency matrix for the graph complement of $A(t)$; $\omega(A(t))$ and $\alpha(A(t))$ are both real non-negative symmetric matrix

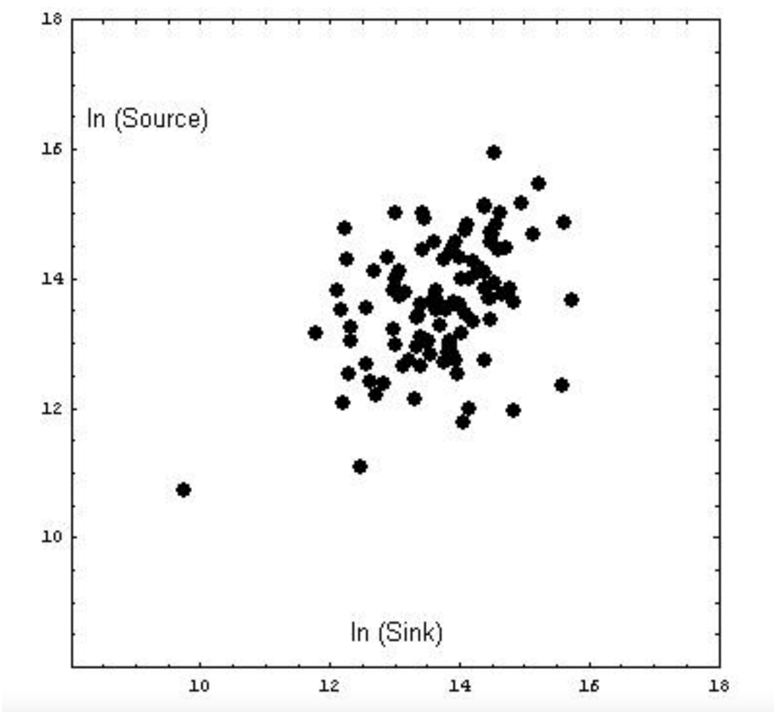


Figure 15: Vertex scatterplots of (source) broadcast-centrality, $\mathbf{b}$ versus (ink) receive-centrality, $\mathbf{r}$.

functions containing conditional edge death rates and conditional edge birth rates respectively; and $\circ$ denotes the **Hadamard product**, or element-wise matrix product, where one simply multiplies the corresponding terms in each matrix together.

One may think of $\alpha(A(t))$ as a matrix of instantaneous birth rates for edges: those edges that are not in $A(t)$ and instead are in $\mathbf{1} - A(t)$. Similarly one may think of $\omega(A(t))$ as a matrix of instantaneous death rates for edges: those edges in $A(t)$ and not in $\mathbf{1} - A(t)$. The Hadamard products simply picks out the required rates for each edge, since both $A(t)$ and $\mathbf{1} - A(t)$ are binary.

To simplify matters further let us consider a discrete uniform time version of the above evolution. Let $\{A_k | k = 1, \dots K\}$ denote an ordered sequence of adjacency matrices (all binary, symmetric with zero diagonals) representing a discrete time evolving network with value A_k at time step t_k. Then we shall assume that the edges evolve (appearing and disappearing) independently from time step to time step with each new network conditionally dependent on the previous network. A first order model is given by an iterative process (a process Markov, see previous notes):

$$\langle A_{k+1} | A_k \rangle = A_k \circ (\mathbf{1} - \tilde{\omega}(A_k)) + (\mathbf{1} - A_k) \circ \tilde{\alpha}(A_k).$$

Here $\tilde{\omega}(A_k) \in S$ and $\tilde{\alpha}(A_k) \in S$ is a real nonnegative symmetric matrix function containing the conditional death and birth probabilities in [0,1], respectively, valid over each time step.

As before the edge independence assumption implies that the conditional distribution $P(A_{k+1}|A_k)$ can always be reconstructed from the expected value $\langle A_{k+1}|A_k \rangle$.

In the sociology literature the simplest form of nonlinearity occurs when people introduce their friends to each other. So if two non-adjacent people are connected to a common friend at step k then it is more likely that those two people will be directly connected at step $k + 1$. To model this **triadic closure** dynamic we may employ

$$\tilde{\omega}(A_k) = \gamma \mathbf{1},$$

so that all current edges have the same step-to-step death probability, $\gamma \in [0, 1]$; together with

$$\tilde{\alpha}(A_k) = \delta\mathbf{1} + \epsilon A_k^2 \circ \mathbf{1}.$$

Here $\delta > 0$ and $\epsilon > 0$ are positive constants and such that $\delta + \epsilon(n - 2) < 1$. This condition ensures that

$$A_k \circ (\mathbf{1} - \tilde{\omega}(A_k)) + (\mathbf{1} - A_k) \circ \tilde{\alpha}(A_k) \in S$$

for all possible values of A_k (why?). The off-diagonal elements $(A_2)_{ij}$ count the number of mutual connections that person i and person j have at step k . The Hadamard multiplication in $\tilde{\alpha}(A_k)$ simply zeros the main diagonal in that term.

It is thus clear that for these definitions both $\tilde{\alpha}(A_k)$ and $\tilde{\omega}(A_k)$ take values in S as required.

We have the dynamic

$$\langle A_{k+1}|A_k\rangle = A_k \circ (1 - \gamma)\mathbf{1} + (\mathbf{1} - A_k) \circ (\delta\mathbf{1} + \epsilon A_k^2 \circ \mathbf{1}). \tag{9}$$

The resulting dynamical equation is ergodic, meaning that over long time, although it it may visit all possible states, such instances are weighted and its average behaviour over time can be described by its expected evolution. In this case the solution is destined to spend most of its time close to states where the density of edges means that there is a balance between edge births and deaths.

In Figure 16 we depict the evolution of this system via the observed edge density, the fraction of edges present in simulations of A_k versus time step k. At each step we generated A_{k+1} from its expected value given by (9): then iterated as required. Cleary the system moves towards one or another quasi stable state. In fact all of these simulations evolve from identical initial conditions. We say “quasi” because in theory, at any time step, we have a distribution for $P(A_{k+1}|A_k)$. So the next graph could jump to anywhere (jump to the empty graph or the clique in the extreme): but some next graphs are much more likely, of course.

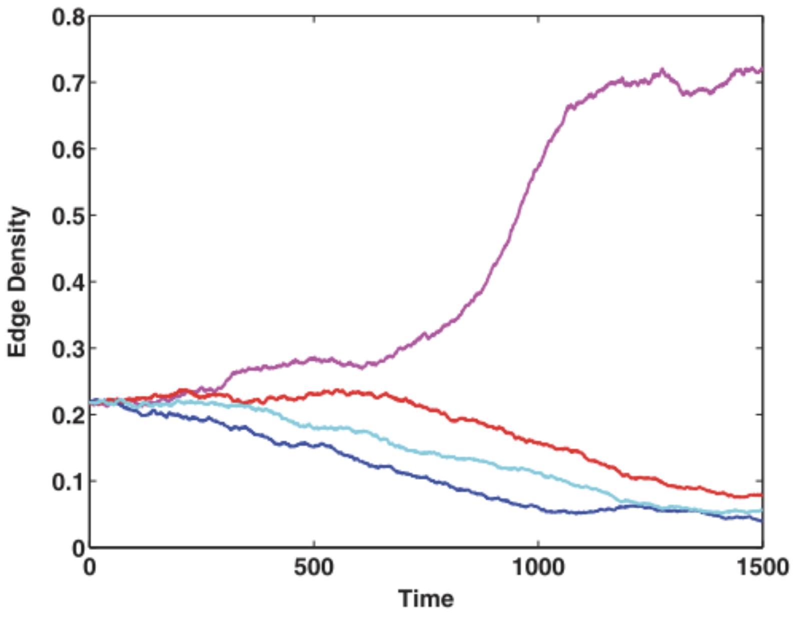


Figure 16: Multiple simulated evolutions of the stochastic dynamic given by (9): we show the observed actual edge density (analogous to the edge probability, p, for an Erdös-Rényi random graph). Here $(n, \delta, \epsilon, \gamma) = (100, 0.0004, 0.0005, 0.01)$.

A mean field approach can be applied by approximating each A_k with its own expectation, which (by symmetry) may be assumed to be of the form $p_k\mathbf{1}$, an Erdös-Rényi random graph, with edge density p_k. We may make such an assumption since no vertices are preferred by the model and the model is invariant under permutations of the vertices.

In the *mean field dynamic* approximation one writes $\langle A_{k+1}|A_k\rangle = p_{k+1}\mathbf{1}$ and replaces the right hand side of the evolution equation (9) with its expected value, using

$$\langle A_k\rangle = p_k\mathbf{1},$$

and

$$\langle A_k^2 \circ \mathbf{1}\rangle = (n-2)p_k^2\mathbf{1}.$$

This last is true since for any two distinct selected vertices: (a) each of the $(n-2)$ possible paths of length 2 between them involves two independent edges (each with probability p_k), and such edge are independent from one another, so such a path has a probability of p_k^2; and (b) since no two such paths can share any edges we may sum those $(n-2)$ probabilities to obtain the expected number of such paths, and hence the expected value of the corresponding term in A_k^2, as $(n-2)p_k^2$.

Hence from (9) we obtain mean field evoltion

$$p_{k+1} = p_k(1-\gamma) + (1-p_k)(\delta + (n-2)\epsilon p_k^2).$$

If δ is small and $\omega < \epsilon(n-2)/4$ then this nonlinear iteration has three fixed points: two of them stable, at $\delta/\gamma + O(\delta^2)$ and $1/2 + \sqrt{1/4 - \gamma/\epsilon(n-2)} + O(\delta)$; and one unstable in the middle at $1/2 - \sqrt{1/4 - \gamma/\epsilon(n-2)} + O(\delta)$. See figure 17 for example.

Thus the extracted mean field behaviour is bistable.

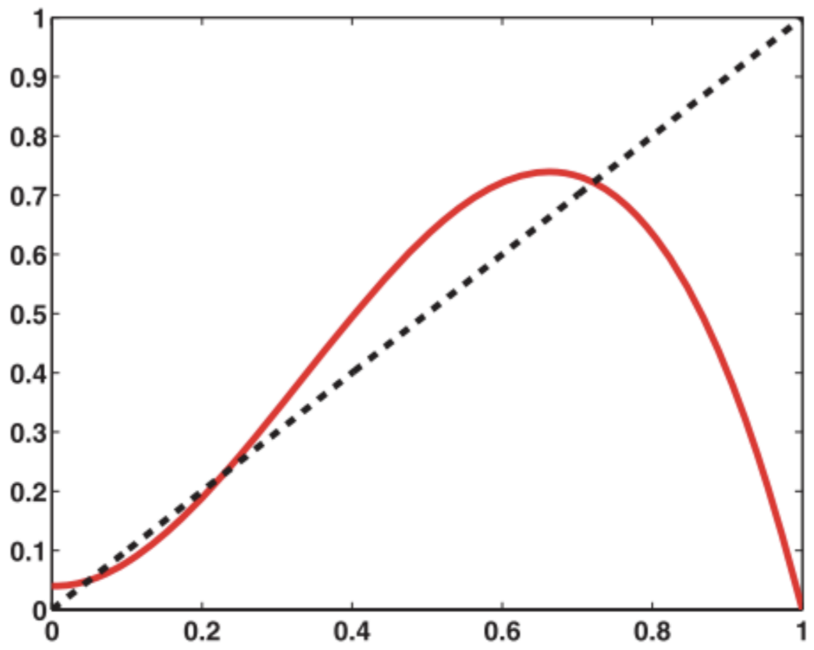


Figure 17: Plot of p (dashed) and $(1-p)(\delta + (n-2)\epsilon p^2)/(1-\gamma)$ (solid) versus p. The fixed points are at 0.049, 0.229 and 0.721: compare these values to those quasi steady states in Figure 16. Here $(n, \delta, \epsilon, \gamma) = (100, 0.0004, 0.0005, 0.01)$.

In practice an observer of data from such a system (9) would see the edge density of such a network approaching one or other stable mean field equilibria, and jiggling around it for a very long time, without any awareness that another type of orbit or pseudo stable edge density could exist.

That is why models are so important, as they imply possibilities or consequences that we have not yet seen.

Notice that the mean field dynamic is deterministic, whereas in practice the actual dynamic in (9) is stochastic. There is always a small chance in (9) that $A_{k+1} = 0$ or $A_{k+1} = \mathbf{1}$ at the next step.

Direct comparisons of transient orbits generated from the full stochastic Markov model (9) with the mean field dynamic, incorporating triadic closure, are very good indeed over short to medium time scales. See Figure 18.

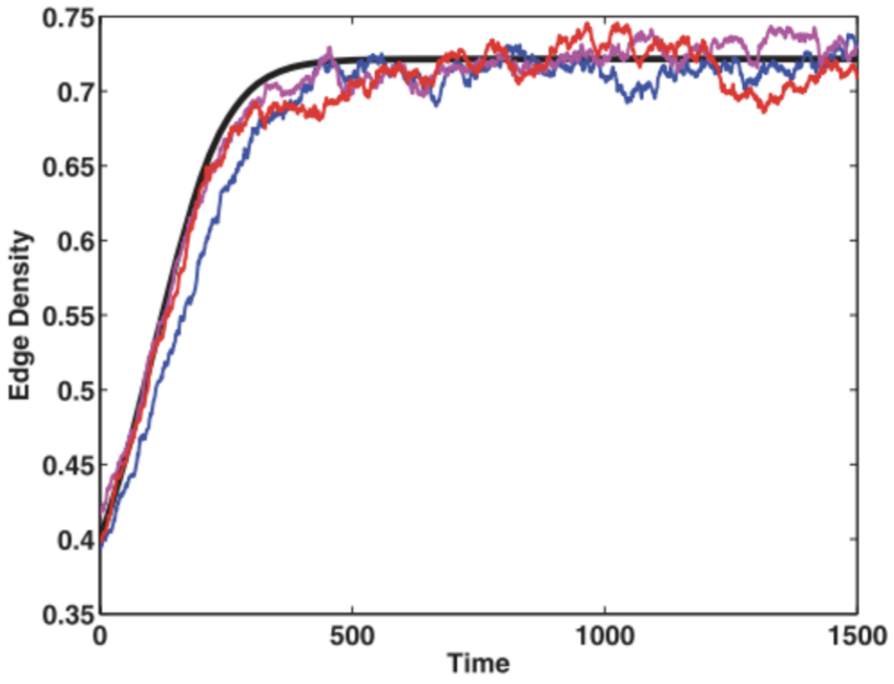


Figure 18: Jagged curves show simulations of edge density extracted for three independent stochastic paths from the Markov chain (9) using the same model parameters. Solid curve shows the smooth mean field evolution.

In fact, the initial condition in Figure 16 was deliberately chosen to be close toe the unstable rest point for the mean field evolution equation, so as to observe divergent trajectories. Yet though we have captured the nonlinear effects well in, the stochastic nature of the full model must eventually cause orbits to diverge from the deterministic stability seen in mean field approximation.

The model in this section can represent friendships a new year cohort of undergraduate school or college where few of the n students know any of the others, and where we measure the social graph of student friendships weekly. Assume $\delta \to 0$. If triadic closure (introducing friends of friends) never gets going then we are destined to approach the lower equilibrium where random friendships form at a rate close to γ and rate that existing friendships expire. The mean field model applies. If there are drinks and cheese parties early on in the first term, the system may get shifted to run close to the upper, more sociable, equilibrium.

6 DYNAMICS ON NETWORKS

One of the main motivations for identifying modular structures in networks is that they provide a simplified, coarse-grained description for the system structure.

Think for instance of a social network, in which we might be able to decompose the system into groups of people such as circles of friends. We may then represent the system in terms of the interactions between these different groups of people, thereby reducing the complexity of our description. The hope is to not only arrive at a more compact structural description, but that the found modules can be interpreted as the 'building blocks' of the system with a functional meaning.

In general this functionality is expressed in the ways by which dynamics is constrained by the underlying structure. Think of flows of passengers in the underground, flows of ideas in citations networks, or flow of information in the social network example. To properly understand such systems, we indeed to consider the dynamics that acts on top of an underlying structure structure. In this section, we will provide an overview of the interplay between structure and dynamics in complex networks by considering (linear) consensus dynamics. First, we describe dynamics with a separation of time-scales and discuss how such a time-scale separation can be a direct consequence of the network structure. Second, we discuss how the presence of particular symmetries in a network can give rise to in- variant subspaces in the dynamics that can be precisely described by graph partitions.

Here is a short summary of the notations used in this section. For simplicity, in the following we consider mainly undirected, connected graphs with n vertices and m edges, as usual.

Our ideas extend to directed graphs, however, which we will outline as we go along. The topology of a graph is encoded in the weighted (non-negative) adjacency matrix $A \in \mathbb{R}^{n\times n}$, where the weight of the edge between vertex i and vertex j is given by A_{ij}. Note that $A = A^T$ for an undirected graph.

The weighted out-degrees (or strengths) of the nodes are given by the vector out-degrees $\mathbf{d} = A.\hat{1}$, where $\hat{1}$ denotes the n-vector of ones. For any vector $\mathbf{x} = (x_1, \dots, x_n)^T$, we define diag($\mathbf{x}$) to be the diagonal matrix X with elements $X_{ii} = x_i$ and zero otherwise. We thus define the diagonal matrix of degrees as D =diag(d).

Recall also that combinatorial graph Laplacian is defined as $L = D - A$. It is symmetric (self adjoint) and positive semi-definite, with a simple zero eigenvalue when the graph is connected.

6.1 Consensus dynamics

Consensus has been one of the most popular and well- studied dynamics on networks. This is due to both its analytic tractability as well as its simplicity in approximating several fundamental behaviours.

For instance, in socio-economic domains consensus provides a model for opinion formation in a society of individuals. For engineering systems, it has been considered as a basic building block for an efficient distributed computation of global functions in networks of sensors, robots, or other agents. To define a standard consensus dynamics, con- sider a given connected network of n nodes and adjacency matrix A. Let us endow each node with a scalar state variable $x_i \in \mathbb{R}$, which is a function of time t. Let $\mathbf{x}(t) = (x_1(t), \dots, x_n(t))^T$.

Suppose $v_i$and v_j are neighbours. Then v_i sees a difference of $(x_j - x_i)$ in their states. If this is positive (respectively negative) then v_i should increase (respectively decrease) its states so as to close the gap.

The (average) **consensus dynamics** on such a network is then defined by the autonomous differential equation:

$$\dot{\mathbf{x}} = -L\mathbf{x}. \tag{10}$$

Note that in coordinate form this simply amounts to $\dot{x}_i = \sum_j A_{ij}(x_j - x_i)$, that is, the state of each vertex adjusts so that the difference to its neighbours is reduced. The name of these dynamics de-rives from the fact that for any given initial system state $\mathbf{x}_0 = \mathbf{x}(0)$, the differential equation above will drive the state to a global 'consensus state' in which the state variables of all nodes are equal.

Since $\hat{1}^T.L = 0$, $\hat{1}$ is an eigenvector of L with zero eigenvalue, we see that

$$\hat{1}^T.\dot{\mathbf{x}} = 0 \rightarrow \hat{1}^T.\mathbf{x} = \text{ constant.}$$

Mathematically, this means that $x_i \rightarrow x^*$ for all i, as $t \rightarrow \infty$, where $x^* = \hat{1}^T.\mathbf{x}_0/n$ is the arithmetic average of the initial node states. Intuitively, this dynamics may be interpreted as an opinion formation process on a network of agents, who will in the absence of further inputs eventually agree on the same value: namely, the average opinion of their initial states.

The rate of convergence is limited by the second smallest eigenvalue of L, the Fieldler eigenvalue, λ_F say. with

$$\mathbf{x}(t) = x^*\hat{1} + O(e^{-\lambda_F t}).$$

To see this simply expand $\mathbf{x}(t)$ out in terms of the orthonormal eigenvectors of L, and observe that the resulting scalar equations decouple. Note that, of course, if the network is NOT strongly connected, so 0 is not a simple eigenvalue for L, then we can solve this equation on each connected component independently, whence $\lambda_F > 0$.

We illustrate this with an example in Figure 19.

As we will show this process is the dual of a random walk process taking place on a network.

Exercise. Write a code to simulate consensus dynamics on a network, and verify that the dynamics asymptotically converges towards the state $x^*\hat{1}$.

6.2 Time-scale separation in dynamical systems

Before discussing time-scale separation in the context of a dynamical process acting on a network, let us explain the concept of time-scale separation with a generic example first.

Consider the following simple dynamical system:

$$\frac{dx}{dt} = f(x, y),$$

$$\epsilon^{-1}\frac{dy}{dt} = g(x, y),$$

where $\epsilon << 1$ is a small positive constant. Note that, the above implies that $x(t)$ changes much more rapidly than $y(t)$. Indeed, dy/dt will be proportional to $\epsilon g(x, y)$, which is small by construction. Alternatively, simply define a new, slow time variable $\tau := \epsilon t$ and note that the above can be written as

$$\frac{dx}{dt} = f(x, y),$$

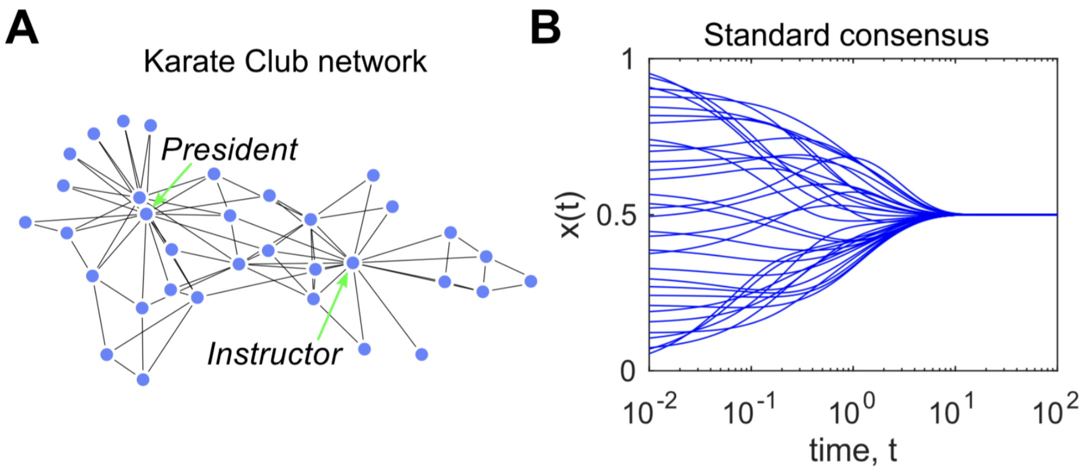


Figure 19: Illustration of a consensus dynamics on the small, often studied, Karate Club network (`https://en.wikipedia.org/wiki/Zachary%27s_karate_club`), originally analysed by Zachary. A: the network. B: Consensus dynamics on the Karate club network starting from a random initial condition. As discussed in the text, as time progresses the states of the individual nodes become more and more aligned, and eventually reach a consensus value, equal to the arithmetic average of the initial condition.

$$\frac{dy}{d\tau} = g(x, y).$$

There is a separation of time-scales in the dynamics, where y evolves according to the 'slow' timescale τ, and x according to the much faster timescale, t

This time-scale separation can be exploited for the analysis of a system in various ways. On the one hand, if we are mainly interested in the short term (fast) behaviour of the system above, we may effectively treat y as a fixed parameter and ignore its time evolution, leading to an effective one-dimensional system description. Indeed for the so called *singular perturbation*, $\epsilon \to 0$, y and x will effectively be decoupled.

On the other hand, if we are mainly interested in the long term behaviour of the system, then it is y we are most interested in. Let us assume, that x(t) converges to some fixed point $x^*(y)$ on the fast timescale, t, as a function of y which is effectively constant on that timescale. Then on the slow time scale we have

$$\frac{dy}{d\tau} = g(x^*(y), y).$$

Using this simplification will, of course, lead to some errors when comparing to the actual time-evolution of y, especially for an initial transient period when f is a long way from zero. However, it allows us again to focus on a simpler one-dimensional system, facilitating a simpler analysis.

This general idea is the basis for center manifold theory and the asymptotic method of multiple timescales [33]. To summarise, the separation of time-scales acts in such a way that it almost decouples the system in two different regimes.

6.3 Time-scale separation in networks with consensus dynamics

Let us now discuss how the above ideas can be trans-lated into the context of networks on which a diffusion or consensus dynamics is acting. For simplicity we will de- scribe the results here in the context of consensus, though translating these ideas to diffusion processes is straight- forward.

For any initial condition $\mathbf{x}_0$, standard linear systems theory tells us that the solution to 10 is given by

$$\mathbf{x}(t) = \exp(-Lt)\mathbf{x}_0,$$

where $\exp(\Delta)$ denotes the matrix exponential. Writing the solution in this way disguises however the time-scales present in the evolution of x as these gets mixed via the network interactions. In order to reveal the characteristic time-scales present in the system we can make use of a spectral decomposition of L. Let us denote the eigenvectors of the Laplacian by $\mathbf{v}_i$, i.e., $L\mathbf{v}_i = \lambda_i \mathbf{v}_i$, and assume that we have ordered the eigenvalues (and associated eigenvectors) in increasing order $0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$.

We may now decompose the Laplacian via

$$L = \sum_i \lambda_i \mathbf{v}_i \mathbf{v}_i^T,$$

and write

$$\mathbf{x}(t) = \sum_i \exp(-\lambda_i t)\mathbf{v}_i \mathbf{v}_i^T \mathbf{x}_0.$$

In this format the time-scales of the process become apparent. They are dictated by the eigenvalues of the Laplacian matrix: each eigenvector (or eigenmode) decays according to a characteristic time-scale $\tau_i = 1/\lambda_i$. Hence, if there are large differences between the eigenvalues, we will have a time-scale separation. More precisely assume that there is a group of k small eigenvalues $\{0, ..., \lambda_k\}$, which are well separated from the remaining eigenvalues in the sense that $\lambda_k << \lambda_{k+1}$. Then after some time, $\tau_0 \sim 1/\lambda_{k+1}$, the eigenmodes associated with teh larger eigenvalues become negligible and the system can be effectively described by the k smallest eigenmodes. More technically, the k first eigenvectors form a dominant invariant subspace of the dynamics.

The main point of the discussion above is that if there is a separation of time-scales, there exists a lower dimensional description of the dynamics on the network after a specific time-scale τ_0. A natural question is thus how this time-scale separation and the lower-dimensional description of our dynamics is related to the network structure.

As an example let us consider a network composed out of k modules, only weakly coupled to the other. To simplify our exposition let us consider the case of a graph with k modules, whose adjacency matrix is dominated by k blocks, corresponding to the k modules, and can be written as:

$$A = A_{\text{structure}} + A_{\text{random}} = \begin{pmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_k \end{pmatrix} + A_{\text{random}}.$$

Here A_{random} may be interpreted as a realization from an Erdös-Rényi random graph, with unstructured, sparse connectivity (just a "noise term"); the A_i are the adjacency (sub-)matrices of the individual clusters which have much higher internal connectivity.

How does the structure present affect the spectrum and the eigenvalues of the cor- responding Laplacian $L = L_{\text{structure}} + L_{\text{random}}$?

Let us first consider the case where $L_{\text{random}} = 0$, so the graph consists of k disconnected components. Then we will have $\lambda = 0$ with algebraic (and geometric) multiplicity k, and the associated eigenspace can be spanned by eigenvectors of the form $\mathbf{c}(j)_i = 1$ if vertex i is in component j and zero otherwise.

To gain insight into the case where $L_{\text{random}} \neq 0$, we can appeal to matrix perturbation theory and random matrix theory, respectively. For a network of the form above, the Davis-Kahan theorem provides bounds on the (angular) distance between the subspace Y spanned by $\{\mathbf{c}(j)\}$, and the subspace Y' spanned by the corresponding eigenvectors of L associated with the smallest eigenvalues. On an intuitive level, the Davis-Kahan theorem states that if the noise level is not too high, then $Y \approx Y'$. The implication is that the dominant invariant subspaces will be commensurate with the structural decomposition of the network in terms of the block-vectors. Hence the long-term dynamics will directly reflect the structural decomposition of the network. In other words, the time scale separation in such a networked system takes an intuitive meaning: quasi-consensus is reached more quickly within each block, while global consensus is only reached on a longer time scale.

To illustrate the above discussion let us consider here a numerical example. The network displayed in Figure 20A consists of 3 groups with 100 nodes each, and is structured as outlined in our discussion above. As can be seen clearly in Figure 20B the dynamics becomes effectively low dimensional after around $t = 0.05$ and can be well approximated by the dominant eigenmodes.

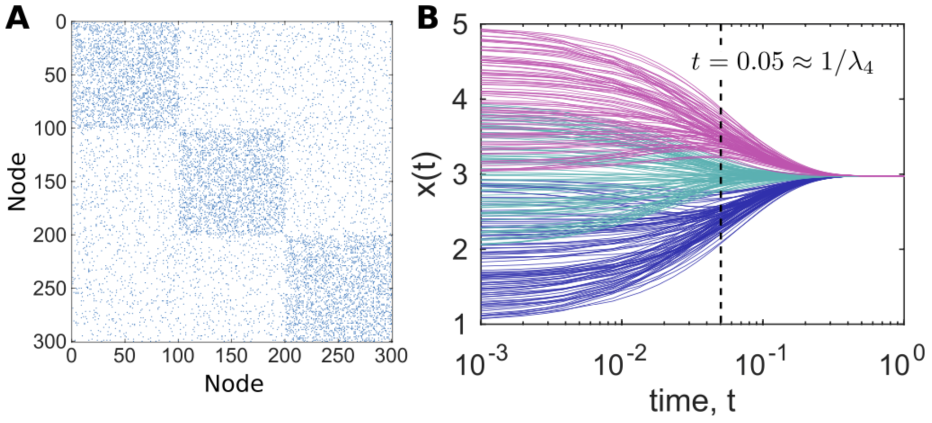


Figure 20: llustration of a consensus dynamics on a structured network. A: Adjacency matrix (unweighted) of a structured network with 3 modules/groups, as discussed in the text. B: Consensus dynamics on this network displays a time-scale separation until around $t = 0.05$, approximate consensus is reached within each group (groups indicated by colour); then a consensus is reached between the groups. Note that for the shown network $\lambda_4 = 18$, in good agreement with our discussion above.

7 RANDOM WALKS

Consensus dynamics has important applications, in connection to the fields of decentralised algorithms and synchronisation. Another important dynamical process on networks is diffusion, aiming at modelling how an entity randomly explores the underlying structure. In this section, we provide a detailed analysis of random walk processes on networks. As we will show, in certain set-tings, random walks can be seen as a dual process of the consensus dynamics described in the previous section.

7.1 Discrete-time random walks on networks

Let us consider a walker diffusing on an undirected network. At each step, the walker located at a certain vertex selects one edges connected to the vertex at random and jumps to an adjacent node. This process is equivalent to a Markov chain, and is described by the $n \times n$ transition matrix T. The probability that the walker visits the ith node after t steps, $p_i(t)$, is given in by

$$p(t) = p(0)T^t,$$

where $p(t) = (p_1(t), \ldots, p_n(t))$. Thus us just (3) in Section 1.3.

The solution, involves products of matrices and can be simplified with the use of a graph Fourier transform. The underlying idea is to decompose the sig-nal in an adequate base of vectors, such that the matrix products take the form of algebraic products for ampli-tudes associated to the different dynamical modes. To work out this idea, let us first note that the transition matrix of the random walk is given by

$$T_{ij} = A_{ij}/d_i,$$

representing the probability that the walker transits from v_i node to v_j. The transition matrix, T, is in general asymmetric, except if the underlying graph is regular (A is symmetric). Nonetheless, its spectral properties can be directly derived from those of the symmetric matrix

$$\tilde{A}_{ij} = \frac{A_{ij}}{\sqrt{d_i d_j}},$$

whose properties are essentially equivalent to those of the normalised Laplacian. By applying the usual spectral (orthonormal) decomposition to $\tilde{A}$, we obtain

$$\tilde{A} = \sum_{i=1}^{n} \lambda_i \mathbf{u}_i . \mathbf{u}_i^T$$

.

Now becasue

$$T_{ij} = \sqrt{d_i} \tilde{A}_{ij} / \sqrt{d_i} \text{ equivalently } T = D^{-1/2} \tilde{A} D^{-1/2}$$

, where $D = \text{diag}(d_1, \ldots, d_n)$, as usual, it is easy to check that the right eigenvectors of T are given by

$$\mathbf{w}_i^R = D^{-1/2} \mathbf{u}_i,$$

and the left eigevectors of T are given by

$$\mathbf{w}_i^L = D^{1/2} \mathbf{u}_i.$$

Note that we write both the right and left eigenvectors as column vectors: so taking the transpose of $\mathbf{w}_i^L$ we have:

$$(\mathbf{w}_i^L)^T.T = \lambda_i(\mathbf{w}_i^L)^T,$$

while we also have

$$T.\mathbf{w}_i^R = \lambda_i \mathbf{w}_i^R.$$

In this case we have the decomposition:

$$T = \sum_{i=1}^{n} \lambda_i \mathbf{w}_i^R.(\mathbf{w}_i^L)^T.$$

(Take care here : we are using T for the transition matrix, T, and also as a superscript to denote the transpose of a vector! Sorry.) Note also that: $(\mathbf{w}_i^L)^T.\mathbf{w}_j^R = \delta_{ij}$ from the orthonormality to the $\mathbf{u}$'s.

We may now solve for the row vector $p(t) = (p_1(t), \ldots, p_n(t))$:

$$p(t) = p(0).T^t = \sum_{i=1}^{n} \lambda^t (\mathbf{w}_i^L)^T (p(0).\mathbf{w}_i^R).$$

The eigenvalues λ_j of the transition matrix are in the interval [-1,1]. The mode with $\lambda_j = 1$ corresponds to the stationary density, and we thus write $(\mathbf{w}_j^L)^T = p^*$. The right eigenvector of that corresponds to this mode is $\mathbf{w}^R \propto (1, ..., 1)^T$.

All of the other modes for which $-1 < \lambda_j < 1$ decay to 0. The eigenvalue $\lambda_j = 1$ is the largest-magnitude eigenvalue, and the Perron–Frobenius theorem guarantees that all elements of $\mathbf{w}_j^L$ and $\mathbf{w}_j^R$ are positive.

Similar results hold for directed networks, although we cannot take advantage of the symmetric structure of the matrix $\tilde{A}$ in general, though similar considerations apply.

By letting t be large but finite, we obtain the row vector:

$$p(t) \approx (\mathbf{w}_{max}^L)^T (p(0).\mathbf{w}_{max}^R) + \lambda_2^t (\mathbf{w}_2^L)^T (p(0).\mathbf{w}_2^R).$$

Here $\lambda_{max} = 1$ and λ_2 denotes the second largest eigenvalue in absolute magnitude.

Hence the second-largest eigenvalue of T governs the relaxation time. More generally, the relaxation speed is determined by the ratio between $|\lambda_2|$ and $\lambda_{max} = 1$. The difference $1 - |\lambda_2|$ is often called the "spectral gap". A large spectral gap entails fast relaxation.

The *Cheeger inequality* gives useful bounds on $|\lambda_2|$. The *Cheeger constant*, which is also sometimes called the "conductance", is defined by

$$h = \min_S \left\{ \frac{\text{number of edges that connect } S \text{ and its complement, } \bar{S}}{\min\{|S|, |\bar{S}|\}} \right\},$$

where S is a subset of the vertices in the network, and $\bar{S}$ is the complement.

The Cheeger inequality is

$$\frac{h^2}{2} < 1 - |\lambda_2| \leq 2h,$$

so a small Cheeger constant, h, implies a small spectral gap, $1 - |\lambda_2|$; and hence slower relaxation. This result is intuitive, because one can partition a network with a small value of h into two well-separated communities such that it is difficult for random walkers to cross from one community to the other. Note that there are various versions of Cheeger constants and inequalities. These results are in line with our discussion on spectral methods for community detection in the previous chapter.

The power method is a practical numerical method to calculate the stationary density of a random walk in a directed network.

7.2 Application: PageRank

A well-known centrality measure for directed networks is the PageRank, which was first introduced for ranking webpages and later adopted in a variety of applications. The PageRank is defined as the stationary density of a discrete-time random walk, particularly on directed networks.

As a reminder, we considered discrete-time random walks on undirected networks. In this section, we start by looking at discrete-time random walk on directed networks, which is a representative Markov chain.

Consider a **directed network** and a random walker in discrete time. In each step, the walker located at the ith node jumps to one of the out-neighbours selected at random. The transition matrix, i.e., the probability that the walker moves from the ith node to the jth node is given by

$$T_{ij} = A_{ij}/d_i^{out}.$$

Although we focus here on the case of unweighted networks, the following analysis may be easily generalised to the weighted case.

The time evolution of the density of the random walk is driven as before. The stationary density essentially defines the PageRank. The PageRank states that node v_i is important if is receives many links, the links entering v_i emanate from important nodes, and a node v_j sending a directed link in to v_i has a small out-degree. The last condition says that the total importance of v_j is shared among its out-neighbours. This circular relationship, i.e., a node is important if it is connected to important nodes, leads to an eigenvalue problem.

The naive use of the stationary density, says,

$$p^* = (p_1^*, \dots, p_n^*),$$

is in general not appropriate because p^* is not unique when there are multiple absorbing states and the stationary density is equal to zero for transient nodes. Recall, the stationary state uniquely exists if and only if the network is strongly connected, which is rare in empirical directed networks. To circumvent these problems, mathematical tricks have been proposed to make the dynamics ergodic even when the underlying network is not strongly connected. The most popular method consists in allowing walkers to randomly *teleport* to other nodes. Random walks with teleportation are driven by the rate equation

$$p_i(t+1) = \alpha \sum_{j=1}^{n} p_j(t) T_{ji} + (1-\alpha)\mathbf{b}_i, \tag{11}$$

where the "preference" row vector $\mathbf{b} = (b_1, \dots, b_n)$ is non-negative and sums to unity; and $\alpha \in (0,1)$ is the probability of teleportation at any given time step; so b_i is the probability of a teleporter landing at v_i, post teleportation. In the case of web browsing, teleportation is interpreted as a jump to a new webpage without following a hyper-link.

The stationary state is formally given by

$$p_i^* = b_i + \sum_{l=1}^{\infty} \sum_{j=1}^{n} b_j (T_{j,i}^{l} - T_{j,i}^{l-1}). \tag{12}$$

This expression clearly shows the non-local nature of the PageRank because it is made of walks of all length l. A small α value gives a low credit to longer walks. As in the case of the Katz centrality, the stationary density may radically change when α is modified.

In matrix form (11) is re-write as an dynamic equation for the row vector $p(t+1)$ as

$$p(t+1) = \alpha p(t)T + (1-\alpha)\mathbf{b},$$

and thus the solution at steady state in (12) may be calculated in the form

$$p^* = (1-\alpha)\mathbf{b}(I - \alpha T)^{-1}.$$

If we take the transpose of this last equation one obtains a matrix equation for the column vector, p^{*T}, and the similarity to Katz centrality is again apparent.

7.3 Models of epidemic processes

Epidemic processes are probably the most studied dynamical processes on networks, both for static and temporal networks. Many of these investigations are motivated by their applications to infectious diseases of humans and animals, viral and other information spreading on social networks, and computer viruses. In this section, we first present classical models of epidemic spreading, and their behaviour in the mean-field, before consider- ing two types of models on networks: meta-population models and spreading on contact networks.

The susceptible-infected-susceptible (SIS) model, the susceptible-infected-recovered (SIR) model and the susceptible-infected (SI) models are probably the most frequently studied epidemic processes. These models are named after the types of state that each node assumes: the susceptible (in short, healthy), infected and recovered states, and admitted transitions between the states. They are called compartmental models, where compartment is a synonym of state. We focus on stochastic versions of these models, which are usually studied when considered on networks. The transition rates of the three models, which fully define the models, are summarised as follows

The SIS model assumes just two processes. When a susceptible node interacts with an infected node, the susceptible node contracts infection and moves into the infected state at rate β. In other words, the probability that a susceptible node gets infected in small time Δt is equal to $\beta\Delta t$. If a susceptible node is adjacent to k_I infected neighbours, the transition rate is equal to $k_I\beta$. An infected node recovers at rate μ irrespectively of the states of the neighbours. Once an infected node recovers, it transits back to the susceptible state. Therefore, a node may contract infection multiple times during a single run of the SIS model.

Consider the mean-field population, that is, , the complete graph on n veryices, where each vertex pair is connected with each other with a normalised weight of $1/N$. Denote the fraction of susceptible and infected nodes at time t by $S(t)$ and $I(t)$, respectively. The dynamics for lafrge N are:

$$\frac{dS(t)}{dt} = -\beta I(t)S(t) + \mu I(t)$$

$$\frac{dI(t)}{dt} = \beta I(t)S(t) - \mu I(t).$$

Clearly this c onserv es mass:

$$\frac{d}{dt}(S(t) + I(t)) = 0.$$

So $S(t) + I(t) = 1$ for all tiume. At eqaulibrium, either $I^* = 0$ (no infection ever present) or else we have the possibility

$$S^* = \frac{\mu}{\beta} \quad I^* = 1 - \frac{\mu}{\beta}.$$

This is intuitive in that the magnitude of infection is large if the infection rate β is large or the recovery rate μ is small. In addition, $I^* > 0$ holds true if and only if

$$\frac{\beta}{\mu} > 1.$$

We say that the epidemic threshold in terms of $\frac{\beta}{\mu}$ is equal to unity in the well-mixed population. Below this value the infection always dies out. As we will see, the epidemic threshold depends on the structure of the underlying network in general.

In the SIR model, infection events occur in the same manner as in the SIS model. The only difference to the SIS model is that when an infected node recovers at rate μ, it transits to the recovered state, not back to the susceptible state. A recovered node does not infect others or is not reinfected. The recovered state can also be interpreted as the removed or dead state because a dead node would not infect or be infected by others. In contrast to the SIS model, infectious nodes are eventually extinct in the SIR model even if the infection rate is high. For an arbitrary initial condition, the final state consists of susceptible and recovered nodes, but not infected nodes. We typically start the SIR model from a single infected node or a small fraction of infected nodes in the background of the susceptible population. The primary interest is in the final size, i.e., the number of recovered individuals when the dynamics have terminated.

The SIR model is suitable for describing the response of a population to a triggering event, such as the viral spreading of a tweet in Twitter. Because such one-shot epidemic dynamics are relevant to many real phenomena, the SIR model and its variants are probably more frequently used than the SIS model unless a slow time scale set by births and deaths of individuals comes into play; birth and death events make the SIR model similar to extensions of the SIS model.

The SIR dynamics for the well-mixed population are described by

$$\frac{dS(t)}{dt} = -\beta I(t)S(t)$$

$$\frac{dI(t)}{dt} = \beta I(t)S(t) - \mu I(t).$$

$$\frac{dR(t)}{dt} = \mu I(t)$$

Summing we defdice that $S(t) + I(t) + R(t) = 1$ for all time.

When $dI(t)/dt > 0$ at $t = 0$, the number of infectious nodes first increases to a macroscopic ($O(N)$) number. In this case, we regard that an outbreak has occurred. Otherwise, initially infected nodes do not trigger secondary infections on a visible scale.

Now we introduce metapopulation models for epidemic spreading. In short, the model assumes a network of subpopulations, not that of individuals.

A subpopulation hosts individuals, and individuals move from one subpopulation to an adjacent subpopulation. A metapopulation model network consists of $\tilde{N}$ vertices , called subpopulations, a and links between pairs of subpopulations, and N particles, which we call individuals. An individual can be anything that is mobile and typically represents a human or animal individual. A subpopulation is a container of individuals, corresponding to a household, office, city, airport, country and so on, where interactions can take place.

The (weighted) adjacency matrix of the metapopulation model is denoted by $\tilde{A}$ and fixed over time. For simplicity, we assume that it is an undirected network: $\tilde{A}$ is symmetric. A edge between two subpopulations allows flows of individuals between them and may be weighted. A metapopulation model can be regarded as a coarse-grained network of individuals and is practical because detailed connectivity between individuals is often unknown, whereas connectivity between subpopulations may be more accessible. At least to estimate.

Metapopulation models are typically defined by two ingredients: a rule for the mobility of the individuals, and a rule for their interactions on the vertices. We first focus on the former aspect. Denote by N_i ($1 \leq i \leq \tilde{N}$), the number of individuals in the ith subpopulation, which varies over time t.

For any t, $\sum_{i=1}^{\tilde{N}} N_i = N$ is satisfied.

The simplest assumption for the mobility of individuals is to assume that each individual performs an independent continuous-time random walk from subpopulation to subpopulation. In other words, an individual moves to a neighbouring subpopulation with probability $D\Delta t$ in short time Δt. The time to the next movement obeys the independent exponential distribution with mean $1/D$. An individual moves from the ith subpopulation to a neighbouring jth subpopulation with probability

$$D\Delta t\tilde{A}_{ij}/\tilde{d}_i,$$

where

$$\tilde{d}_i = \sum_{j=1}^{\tilde{N}} \tilde{A}_{ij},$$

is the (weighted) degree of the ith subpopulation. The number of individuals in each subpopulation, $N_i(t)$, is approximated to be a continuous variable when N is large (there are many exchanges). The master equation for N_i is given by

$$\frac{N_i}{dt} = -DN_i + D\sum_{j=1}^{\tilde{N}} N_j \frac{\tilde{A}_{ji}}{\tilde{d}_j} = -D\sum_{j=1}^{\tilde{N}} N_j L'ji.$$

where $i = 1, \dots, \tilde{N}$ and L' denotes the random walk normalised Laplacian matrix, as usual.

By setting the left-hand side to zero, we obtain the equilibrium density of individuals as

$$N_i^* = \frac{\tilde{d}_i N}{\langle\tilde{d}\rangle\tilde{N}}. \tag{13}$$

Note that $N/\tilde{N}$ is the average population density per subpopulation.

The second ingredient of metapopulation models is the rule of interaction between the agents. Importantly, the population is assumed to be structureless within each subpopulation, and thus well-described by a mean-field, all-to-all process. In the following, we focus on a SIS model on the metapopulation model.

We denote by $N_{S,i}$ and $N_{I,i}$ the numbers of susceptible and infected nodes in the ith subpopulation, respectively. We assume that the diffusion rates for susceptible and infected individuals, D_S and D_I, respectively, are possibly different and that each pair of individuals in the same subpopulation interacts at a constant rate. The master equations are thus given by

$$\frac{dN_{S,i}}{dt} = -\beta N_{S,i}N_{I,i} + \mu N_{I,i} - D_S N_{S,i} + D_S \sum_{j=1}^{\tilde{N}} N_{S,j} \frac{\tilde{A}_{ji}}{\tilde{d}_j}$$

$$\frac{dN_{I,i}}{dt} = +\beta N_{S,i}N_{I,i} - \mu N_{I,i} - D_I N_{I,i} + D_I \sum_{j=1}^{\tilde{N}} N_{I,j} \frac{\tilde{A}_{ji}}{\tilde{d}_j}.$$

To determine the epidemic threshold, we consider the situation in which $N_{I,i}$ $(1 \leq i \leq \tilde{N})$ is infinitesimally small. As long as susceptible individuals move (i.e., $D_S > 0$), then the equilibrium for the susceptible individuals is given by their stated state in (13), in the absence of any infection:

$$N^*_{S,i} = \frac{\tilde{d}_i N}{\langle \tilde{d} \rangle \tilde{N}}$$

Now consider the $N_{I,i}$ dynamics for $N^*_{I,i}$ very small, substituting this equilibrium value for $N^*_{S,i}$, we have

$$\frac{dN_{I,i}}{dt} = +\beta \frac{\tilde{d}_i N}{\langle \tilde{d} \rangle \tilde{N}} N_{I,i} - \mu N_{I,i} - D_I N_{I,i} + D_I \sum_{j=1}^{\tilde{N}} N_{I,j} \frac{\tilde{A}_{ji}}{\tilde{d}_j}. \tag{14}$$

This equation is linear in the $N_{I,i}$. In matrix form we can write (14) as

$$dN_I dt = BN_I,$$

where $N_I = (N_{I,1}, \ldots, N_{I,\tilde{N}})^T$. The equilibrium solution at $N_I = 0$ becomes unstable as soon as the rightward most eigenvalue of B becomes positive — whence very small perturbations away from zero will grow exponentially. This condition determines the epidemic threshold for this system.

In practice this always provides a condition on β/μ that depends on the distribution of the degrees within the metapopulation network.

For example, in the limit $D_I = 0$ (invectives are immobilised), let us consider (14, which decouples. The epidemic threshold is determined by the subpopulation having the largest degree, denoted by $\tilde{d}_{max}$. The condition for endemicity is given by

$$\frac{\beta}{\mu} > \frac{\langle \tilde{d} \rangle \tilde{N}}{\tilde{d}_{max} N}.$$

In fact though, even if this condition is met, only the subpopulations having the largest degrees accommodate infected individuals.

8 SCALING PROPERTIES OF GROWING NETWORKS

We follow [18] and focus on graphs which grow by successively combining various smaller graphs. In concept this reverse of the community detection process of partitioning larger networks into small densely connected sub-communities, called modules, considered earlier.

Instead, here we will successively combine graphs, potentially creating composite high-modularity graphs, and consider the behaviour of various functions that may be defined over them. While including a variety of graph combination mechanisms, we will consider only well behaved functions that are insensitive to a small number extra edges to be added-in to the combination. In many applications to social, organisational, biological, or urban networks we observe scaling laws that govern the behaviour of functional properties of graphs, or properties of dynamical systems that are defined over the graphs, as the underlying graphs grow [12, 19, 20, 21].

In that context it is important to differentiate between two distinct types of scaling law behaviour, which should not be conflated (yet may be related). There is the scaling behaviour exhibited within a single large graph: by the decay of its degree distribution, for example, in scale-free graphs, which are often formed by mechanisms such as preferential attachment [**?**, **?**]. There is also the scaling law exhibited by the evolution of graph properties as the graphs themselves grow, which is observed when we examine and compare such objects of very different sizes (and thus different stages of growth), such as cities, organisations, and organisms [12, 19, 20, 21]. We are primarily focused on the latter here, though we will allow graphs to grow by successive combination of components, including via preferential attachment-type mechanisms.

In subsection 8.2 we deal with the situation where we combine multiple copies of a single random graph. In that case, within any realisation, each addition might be distinct from all others, yet all conforming to the same random (sub)graph model. In one step this creates large graphs whose modules are themselves generated as random graphs, and which necessarily must posses scaling laws (for growth and asymptotic decay).

A realistic model might include a sophisticated combination mechanism that would allow for both the additional graphs to be incorporated and also the associated inter-graph attachments. To achieve this the commutative operation used to combine any two random graphs may itself, of course, include some biased random elements. It would still result in a new, combined, random graph.

8.1 Scaling laws for successively combined graphs

Let $\mathcal{S}$ denote the set of all undirected graphs on a finite number of vertices. For each graph, $S \in \mathcal{S}$, we let $V(S)$ denote its vertex set and $E(S)$ denote its edge set. Let $\diamond$ denote a commutative binary operation over $\mathcal{S}$.

For example $\diamond$ might denote the disjoint union of two graphs, $\biguplus$, as in [15] .

Alternatively we might extend the definition of $\mathcal{S}$ so that each graph in $\mathcal{S}$ also has a distinguished vertex, then $\diamond$ might denote the disjoint union, except for the two distinguished nodes, which are to be identified together as the one new distinguished vertex, inheriting neighbours from both original graphs (this is a kind of *star* combination of the two graphs).

For any positive integer $n \in \mathbb{Z}^+$ let nS denote the $\diamond$ operation on n successive copies of S:

$$nS = S \diamond S \diamond ... \diamond S = S \diamond (n-1)S, \quad n = 2, 3, ...$$

[Warning: in this section we will be using n to denote the number of successive graphs combined together, and not the number of vertices in any given graph, as we have previously.]

A Banach space is a complete normed vector space. Think of the real numbers $\mathbb{R}$.

Let $Q : \mathcal{S} \to \mathcal{B}$ be a function over $\mathcal{S}$ taking values in a Banach space, $\mathcal{B}$. We are interested in functions which are relatively *well behaved* with respect to the edge set of S: in particular functions where a small relative change in the edge set may only result in a small change in Q. Since $\mathcal{S}$ contains discrete objects we will assume the following condition: there exists $C > 0$ such that for any graph $S_1 \in \mathcal{S}$, if we add some new edges to produce a new graph, S_2, then

$$||Q(S_2)||_{\mathcal{B}} - ||Q(S_1)||_{\mathcal{B}} \leq C\frac{|E(S_2/S_1)|}{|E(S_1)|}.$$

Here $|E(S_2/S_1)| = |E(S_2)| - |E(S_1)|$ is the change in the number of edges. As S_1 and $E(S_1)$ get larger the addition of a fixed number of edges become a smaller relative change, and thus the change in Q tends to zero in the limit.

This rules out many functions, Q, which count graphs properties. For example if Q counts the connected components of a graph, then even for the very largest disconnected graph, the addition a single edge can reduce Q by one. This restriction is important within applications, since the data specifying graphs may contain some errors in either the observation or the interpretation of edges: with both false positives and false negatives. Many properties of graphs may not be robust to such errors.

We define a **scaling function**, $H : \mathcal{B} \times \mathbb{Z}^+ \to \mathcal{B}$, for Q and $\diamond$, if it exists, to be a mapping that describes how the function Q behaves under the $\diamond$-combination of multiple isomorphisms of any graph in $S \in \mathcal{S}$, that satisfies

$$Q(nS) = H(Q(S), n), \tag{15}$$

for all $S \in \mathcal{S}$ and all $n \in \mathbb{Z}^+$.

We are interested in setting out a sufficient condition on the pair $(\diamond, Q)$ for the existence of a scaling function,H,

The following result is a generalisation of that given in [15] (where $\mathcal{B} = \mathbb{R}$ and $\diamond$ represented the disjoint union of two graphs).

Suppose that for all $S_0 \in \mathcal{S}$ there exists $C(S_0) > 0$ such that Q satisfies

$$||Q(S_1 \diamond S_0) - Q(S_2 \diamond S_0)||_{\mathcal{B}} \leq C(S_0)||Q(S_1) - Q(S_2)||_{\mathcal{B}} \text{ for all } S_1,\ S_2 \in \mathcal{S}. \tag{16}$$

for all S_1 and $S_2 \in \mathcal{S}$. Then there exists a scaling function, H, satisfying (15).

This is proved as foolows.

On the contrary if for any S_1 and S_2 in $\mathcal{S}$ such that $Q(S_1) = Q(S_2)$ we also have $Q(nS_1) = Q(nS_2)$ for all $n \geq 1$, then the scaling function H may be constructed in a pointwise fashion, without any ill-definition. For any pair $(\hat{Q}, n)$, such that $\hat{Q}$ is in the range of Q we may select any S for which $Q(S) = \hat{Q}$ and assign the corresponding value $Q(nS)$ to $H(\hat{Q}n)$), without any inconsistency.

So , if (15) fails then there must exist a pair of graphs S_1 and S_2 in $\mathcal{S}$ and a smallest integer $n^* > 1$, such that $Q(nS_1) = Q(nS_2)$ for $n = 1, ..., n^* - 1$, that is, Q cannot distinguish between nS_1 and nS_2

for $n < n^*$; and yet $Q(n^*S_1) \neq Q(n^*S_2)$, so that Q does distinguish between n^*S_1 and n^*S_2. Such a situation clearly would make any desired scaling function, H, ill-defined.

Secondly, we will assume (16) together with the converse of (15) and establish a contradiction.

Assuming the converse of (15), then there exist S_1 and S_2 in $\mathcal{S}$ such that $Q(nS_1) = Q(nS_2)$ for $n = 1, ..., n^* - 1$ and yet $Q(n^*S_1) \neq Q(n^*S_2)$. Then, by using the commutativity of $\diamond$, we have

$$||Q(n^*S_1) - Q(n^*S_2)||_{\mathcal{B}} = ||Q(S_1\diamond(n^* - 1)S_1) - Q(S_2\diamond(n^* - 1)S_2)||_{\mathcal{B}}$$
$$= ||Q(S_1\diamond(n^* - 1)S_1) - Q(S_2\diamond(n^* - 1)S_1) + Q(S_2\diamond(n^* - 1)S_1) - Q(S_2\diamond(n^* - 1)S_2)||_{\mathcal{B}}$$
$$\leq ||Q(S_1\diamond(n^* - 1)S_1) - Q(S_2\diamond(n^* - 1)S_1)||_{\mathcal{B}} + ||Q(S_2\diamond(n^* - 1)S_1) - Q(S_2\diamond(n^* - 1)S_2)||_{\mathcal{B}}.$$

Employing (16) twice we have

$$||Q(n^*S_1)-Q(n^*S_2)||_{\mathcal{B}} \leq C((n^*-1)S_1)||Q(S_1)-Q(S_2)||_{\mathcal{B}}+C(S_2)||Q((n^*-1)S_1)-Q((n^*-1)S_2)||_{\mathcal{B}}.$$

But both of these terms vanish, so $Q(n^*S_1) = Q(n^*S_2)$, which is the required contradiction.

Hence the required result.

Remark. The expression (16) is a Lipschitz-type condition.

Remark. The argument given in [15] may be extended to show that the functional equation (15), together with the initial condition $H(Q, 1) = Q$, has solutions in the form of both power law growth and power law decay to a constant, $H(Q, n) = n^{-\alpha} + \beta(1 - n^{-\alpha})$, for constants $\alpha \in \mathbb{R}$, $\beta \in \mathcal{B}$. This is achieved by applying (15) twice over, so that $H(Q, nm) = H(H(Q, m), n)$, and deploying an ansatz.

If the binary combination operation is not commutative, then we will require Q to be Lipshitz in both arguements of $\diamond$, since the proof above theorem relies on this, and we replace it by the following result.

Suppose that for all $S_0 \in \mathcal{S}$ there exists $C(S_0) > 0$ such that Q satisfies

$$\max\{||Q(S_1\diamond S_0)-Q(S_2\diamond S_0)||_{\mathcal{B}}, ||Q(S_0\diamond S_1)-Q(S_0\diamond S_2)||_{\mathcal{B}}\} \leq C(S_0)||Q(S_1)-Q(S_2)||_{\mathcal{B}}, \quad (17)$$

for all $S_1,\ S_2 \in \mathcal{S}$. Then there exists a scaling function, H, satisfying (15).

8.2 Scaling laws for random graphs

In this section we wish extend the above considerations to apply to random graphs, as some natural graph-building operations have a stochastic elements: for example, *preferential attachment* constructions [**?**].

As we have seen random graph W is a probability distribution, $P_W(S)$ defined over $\mathcal{S}$. Let $\mathcal{W}$ denote the set of all random graphs. $\mathcal{W}$ is a complete metric space under the L^1 metric: $d(W_1, W_2) = \sum_{S\in\mathcal{S}} |P_{W_1}(S|X) - P_{W_2}(S|X)|$.

A random graph may be described by a probability distribution, or else, more succinctly, by the random process which generates them [26], for example, by defining an independent probability for every possible edge. Alternatively, a circular "range-dependent" random lattice on K vertices, arranges those vertices in a circle, like the hours of the clock, with the *longer range*, more distant, vertices being successively less likely to be connected [2], similar to the well known "small world" networks [3].

Next we use $\square$ to denote a commutative binary operation over $\mathcal{W}$, so that $W_1 \square W_2$ is composite random graph, formed according to some commutative mechanism, that is conditional on both W_1 and W_2 (as well as X).

As before, we let $nW = W\square ... \square W$ denote the combination of n copies of W.

Now let $Q : \mathcal{W} \to \mathcal{B}$ be a given mapping taking values in a Banach space, $\mathcal{B}$. Then we may extend the arguments given in the above to yield the following result.

If Q and $\square$ are such that for all $W_0 \in \mathcal{W}$, there is a constant $C(W_0) > 0$, and for all W_1 and $W_2 \in \mathcal{W}$, we have

$$\max\{||Q(W_1\square W_0) - Q(W_2\square W_0)||_{\mathcal{B}}, ||Q(W_0\square W_1) - Q(W_0\square W_2)||_{\mathcal{B}}\} \leq C(S_0)||Q(W_1) - Q(W_2)||_{\mathcal{B}}, \tag{18}$$

then a scaling function $H : \mathcal{B} \times \mathbb{Z}^+ \to \mathbb{R}$ exists, satisfying

$$Q(nW) = H(Q(W), n),$$

for all $W \in \mathcal{W}$ and all $n \in \mathbb{Z}^+$.

Note that this scaling law refers to the Q-valued property (in $\mathcal{B}$) of the random graph nW as it grows with n.

A Preferential Attachment Stochastic Block Model (PASBM)
Using this result we may introduce a novel form of random graph, with familiar parentage, constructed as follows.

We generate successive sub-component graphs, all from the same random Erdös-Rényi graph, $G(K, p)$, with parameters K and p fixed. At each successive step we have an "existing combination graph" (ECG) and a new Erdös-Rényi graph to be added. At the first step the ECG is just a single graph drawn from $G(K, p)$.

At each step we combine the two graphs by introducing an edge between a vertex within the ECG selected with probability proportional to the present ECG's vertex degree distribution, and a vertex within the new Erdös-Rényi graph, $G(K, p)$, again selected with probability proportional to the observed vertex degree distribution. After n steps we have a random graph over Kn vertices, where the successive n Erdös-Rényi graphs have been preferentially attached. In Figure 21, we show such a graph, where $(K, p, n) = (10, 0.6, 100)$. So, for the PASBM graph, the combination operation, $\square$, involves (degree-biased) random preferential attachment. Note if $K = 1$, then we preferentially attach a single new vertex to the ECG at each step, and thus we reduce to the usual preferential attachment model [**?**, **?**].

For any graph growing in this way (via PASBM), and for any well behaved function of such a graph, there will exist a scaling function. The latter is highly likely to be a powerlaw in n, the number of basic random graphs being successively combined.

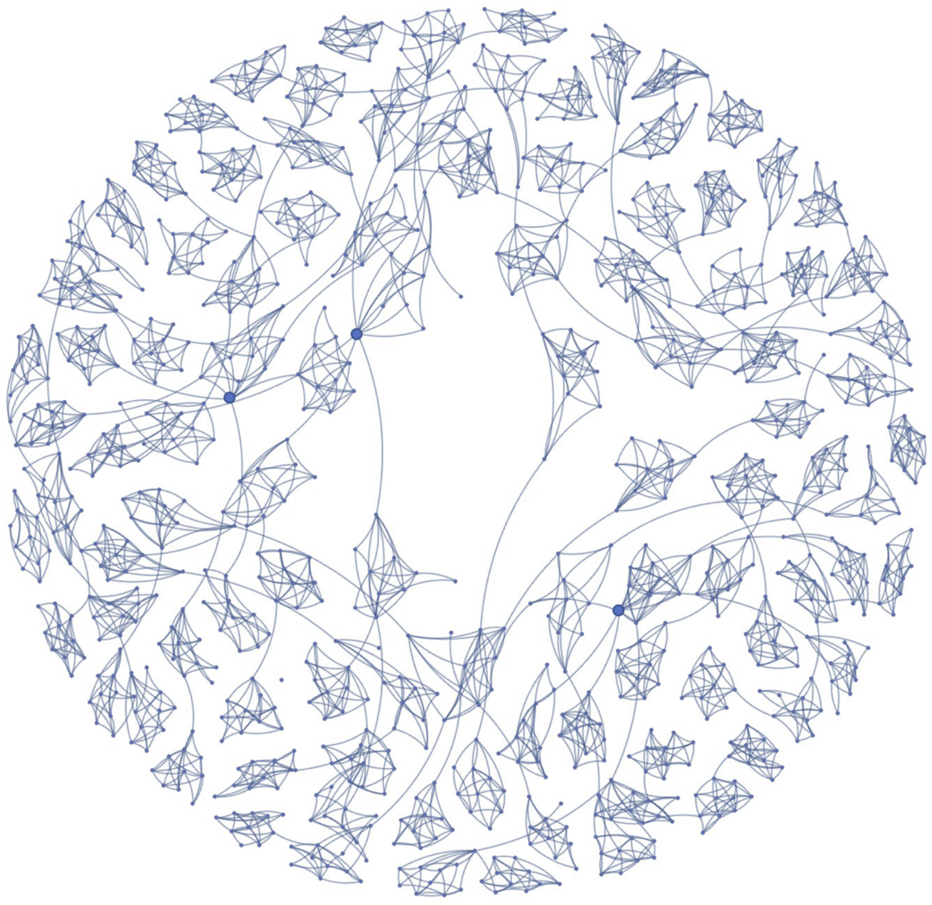


Figure 21: An instance of a Preferential Attachment Stochastic Block Model (PASBM) random graph with $(K, p, n) = (10, 0.6, 100)$. The three vertices with equal highest degree (equal to 11) are highlighted.

8.2.1 Some alternative random graph combination operations

Our characterisation of $\square$ as a commutative binary operation over $\mathcal{W}$, leaves much room for further thought. Beyond disjoint union and preferential attachment, which are both types of aggregation, we now discuss some alternatives well worth considering as prospective models for system growth, since all will yield scaling laws, with respect to growth, whenever suitably behaved functions, Q, are observed.

Mechanistic aggregative combination

So far we have combined two random graphs via (biased and random) **preferential attachment**. In that case we first took the disjoint union and then selected a single vertex from each component sub-graph, according to a probability distribution proportional to the (sub-graph) vertex degrees, and then joined them with a single new edge.

Similarly, we might have first taken the disjoint union and then selected a single vertex from each component sub-graph, according to a probability distribution proportional to the r power of the vertex (sub-graph) degrees, and joined them with a single new edge. As $r \to 0$ this is equivalent to randomly choosing a vertex within each graph, with equal probabilities, and joining them: this might be termed **non-preferential attachment**. As $r \to \infty$ this is equivalent to picking a vertex with the maximum degree within each graph and then joining them: this might be termed **most popular vertex attachment** (a random element is necessary only if there exists more than one vertex with maximum degree in either graph).

We may generalise these mechanisms to include up to $J > 1$ connected edges between the random sub-graphs, each drawn independently with repetition: this might be termed called **multiple-edge preferential attachment/non-preferential attachment/most popular vertex attachment**.

These are all *aggregative* operations, since they embellish the disjoint union of the sub-components, maintaining the component vertices.

We have already mentioned **star attachment**. This applies where each of the graphs drawn from a random graph additionally has a unique *distinguished* vertex. Two such graphs are combined by first taking their disjoint union, and then followed by identifying both components' distinguished vertices together as one single distinguished vertex in the combined graph, retaining all of the edges from both of the component graphs.

Product combination

Hierarchical network models [29] have been proposed two decades ago, where a deterministic self-similar network is constructed iteratively, by successively combining copies of the same graph. This was motivated by the failure of the other simple scale-free models to produce and maintain high clustering coefficients [30]. The combination operator can be extended to produce a hierarchical model combining successive random graphs. The random graph must have a distinguished, "core", vertex (as with star attachment), the none core vertices are termed "peripheral". At each new iteration/level multiple copies of (in our case) a random graph are attached, with edges connected all peripheral nodes to the core vertex from an instance at the previous iteration/level.

In this constriction, the iterated operator, $\Box$, is not commutative. However the scaling theorem from section 8.2 still applies, under the extended condition given in 18, guaranteeing a scaling law for certain functions, under the corollary given in section 8.2.

Hierarchical models combining random graphs are analogous to statistical self-similar fractals, rather than deterministically self-similar fractals.

A hierarchical approach also underpins the Kronecker graphs [31], introduced over a decade ago, with graphs combining via a product; and that non-commutative combination operation calls for the the corollary from section 8.2 in order to guarantee scaling laws as graphs grow iteratively.

Directly combining probability distributions

Let $h : [0,1]^2 \to \mathbb{R}^+$ be any commutative mapping of two variables. Then for any two random graphs W_1 and W_2 in $\mathcal{W}$, and all graphs $S \in \mathcal{S}$, we define $\Box$ by

$$P_{W_1 \Box W_2}(S|X, W_1, W_2) = \frac{h(P_{W_1}(S|X), P_{W_2}(S|X))}{\sum_{S' \in \mathcal{S}}}.$$

If, in addition, $h(x, x) = x$ for $x \in [0, 1]$ then $W \square W = W$.

This type of definition may actually be impractical in many ways, as it seems non-constructive. Yet one can imagine sampling (using MCMC for example [32]) graphs from such a combined probability distribution, albeit relatively inefficiently.

8.3 Summary

We have shown that there are a wide number of methods by which one might successively combine random graphs so as to create a growing random graph. Regardless of that chosen combination operator, $\square$, which may itself include some random elements, there is a strong possibility that observable (summary) performance measures (taking values in some Banach space) will posses a scaling law, as the underlying graph grows. Such scaling might control unbounded growth or the decay to some asymptotic constant. The deciding factor will be whether the function of interest, denoted by Q above, is *well behaved* meaning that it satisfies a Lipschitz-type condition under alternative combinations.

By making these, quite general, considerations we have created a framework that subsumes *aggregative combination* processes and also *product combination* processes, including Hierarchical and Kronecker models. We also introduced the particular class of preferential attachment stochastic block model graphs.

Power laws are an important sub-class of scaling laws, for both growth and decay. Their relative preponderance within analyses of aggregative processes is well known. They are of course solutions of the general scaling law functional equation (though exponential scaling laws can also exist).

The importance of all this lies in considering combinations of random graphs, and allowing the combination operator to be quite general. It is thus applicable far beyond the standard applications to the aggregative growth of highly modular graphs in modelling cities, organisation and organisms.

9 Centrality in continuous time

9.1 Matrix Valued Functions

We have already seen various matrix valued functions of matrices: A^2, $(I-\alpha A)^{-1}$, $ExpA$ and so on.

Before going any further we pause to consider some properties of matrix valued functions that will be useful. In particular we would wish to be able to define fractional powers of matrices, as well as the matrix logarithm.

These last are multivalued, so we wish to take their primary value, while other functions, such as eA or the resolvent $(I-\alpha A)^{-1}$ that we have met, simply need to be well-defined.

There is a number of ways to define matrix valued functions and we shall focus on the simplest. The book [34] is an excellent source of further reading. It is game changing. Or consult the notes at `http://eprints.maths.manchester.ac.uk/2067/1/paper.pdf`.

Suppose A in an $n \times n$ matrix with complex valued elements. Then it can be written in its **Jordan canonical form**: where Z is non-singular and

$$A = ZJZ^{-1},$$

$J = \mathrm{diag}(J_1, J_2, ..., J_q)$ is a **block diagonal matrix** with each J_k corresponds to an eigenvalue λ_k, has dimension $n_k \times n_k$, and is of the form

$$J_k = \begin{pmatrix} \lambda_k & 1 & 0 & & 0 \\ & \lambda_k & 1 & & \\ & & \lambda_k & & \\ & & & \ddots & \\ 0 & & & & \lambda_k \end{pmatrix}.$$

The eigenvalues λ_k are all distinct, with multiplicities n_k, which sum to n.

For any complex function f, at least $n_k - 1$ derivatives, denoted by $f^{(1)}$, $f^{(2)}$, ..., $f^{(n_k-1)}$, we define

$$f(J_k) = \begin{pmatrix} f(\lambda_k) & f^{(1)}(\lambda_k) & f^{(2)}(\lambda_k)/2! & \dots & f^{(n_k-1)}(\lambda_k)/(n_k-1)! \\ 0 & f(\lambda_k) & f^{(1)}(\lambda_k) & \dots & f^{(n_k-2)}(\lambda_k)/(n_k-2)! \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & 0 & \dots & f(\lambda_k) \end{pmatrix}.$$

Let f be defined on the spectrum of A and let A have the Jordan canonical given above. Then we define

$$f(A) = Zf(J)Z^{-1} = Z\mathrm{diag}(f(J_k))Z^{-1}.$$

This definition yields a matrix $f(A)$ that can be shown to be independent of the particular Jordan canonical form.

In the case of multivalued functions, such as $f(t) = \sqrt{t}$ or $\ln t$, is is implicit that a single branch has been chosen in each sub-block, $f(J_k)$.

There are many others ways to define matrix valued functions.

Recall that the adjacency matrix A of any undirected graph is real and symmetric. Therefore, $A = A^T$ and $AA^T = A^TA$ so A is normal (by definition) and thus unitarily diagonalisable (J is diagonal, as is $f(J)$ – if it is well-defined).

Put simply, for our own purposes, we must make sure that f is well defined at all of the eigenvalues of A.

For example, consider $f(z) = (1-\alpha z)^{-1}$, for some real $\alpha > 0$. This has a simple pole at $z = 1/\alpha$. By the P-F theorem the eigenvalues of an adjacency matrix are contained in a disc of radius $r = \rho(A) > 0$, theP-F eigenvalue. So provided we have $1/\alpha > r$ we will be fine and $f(A)$ exists. This is equivalent to assuming α is small enough ($< 1/r$). We saw this before when we wanted the geometric series to converge (for Katz centrality):

$$f(A) = (1 - \alpha A)^{-1} = I + \alpha A + \alpha^2 A^3 ...$$

Let A have no eigenvalues on $\mathbb{R}^-$ (the closed negative real axis). We need the following definition. **Principal log**: $B = \log A$ denotes the unique B such that $e^B = A$ and $\arg \mu_i \in (-\pi, \pi)$for every eigenvalue μ_i of B.

Exercise Let A be the adjacency matrix for an undirected graph. Then A is diagonalisable. How many square roots does A have (that is matrices B for which $B^2 = A$)? What about fractional powers of A, denoted by A^β, for $\beta \in [0, 1]$?

We can define the Principal pth root as follows. For integer $p > 0$, $B = A^{1/p}$ is the unique B such that $B^p = A$ and $\arg \mu_i \in (-\pi/p, \pi/p)$, for every eigenvalue μ_i of B.

Siumilarlty we define the **Principal power**. For $s \in \mathbb{R}$, the principal power is defined as $A^s = e^{s \log A}$, where $\log A$ is the principal logarithm.

Exercise Let A be the adjacency matrix for an undirected graph. We write $A = U\Lambda U^T$, where U is a unitary matrix with columns given by eigenvectors of A, and $\Lambda = \text{diag}(\lambda_1, \lambda_2, ...\lambda_n)$ is a real diagonal matrix of eigenvalues. Show that for all polynomials, P, of any degree there exists a polynomial R of degree less than n such that $P(A) = UR(\Lambda)U^T$, where $R(\Lambda) = \text{diag}(R(\lambda_1), R(\lambda_2), ...R(\lambda_n))$.

9.2 Katz Centrality revisited with age discounting

Suppose that we receive more and more data as time evolves. Then it is necessary to update Katz centrality, Q, for successive time steps. Suppose for ease that each time step is of duration δt.

We write

$$Q(T) = (I - \alpha A_1)^{-1}(I - \alpha A_2)^{-1}...(I - \alpha A_K)^{-1}$$

where $T = K\delta t$ and δt is the uniform time step. Then if new information A_{K+1} corresponding to the time interactions within the time step $(K\delta t, (K + 1)\delta t]$, we have

$$Q(T + \delta t) = Q(T)(I - \alpha A_{K+1})^{-1}.$$

So Q is updated by a single (sparse) matrix solve.

A discount for aged information may be introduced as follows. This is important as it means Q will be successively less dependent on older data, by down-weighting those dynamic walks by their total age (the total time since starting). This is achieved by discounting non-trivial paths at each time step:

we have $Q = I + (Q - I)$, where I counts the trivial walk (staying stationary up to now), while $Q - I$ counts walks with at least one edge from some past time step.

So we may rewrite our update as

$$Q(T + \delta t) = (I + e^{-b\delta t}(Q(T) - I))(I - \alpha A_{K+1})^{-1}1.(7)$$

Here $b > 0$ and any path of length m starting out exactly r time steps ago will be discounted with a weighting equal to $\alpha^m e^{-br\delta t}$.

In fact it has been shown that if we exploit this form of Q to forecast some element of the future behaviour of the evolving network then there is an optimal range of values for the discount parameter. If we take $b = 0$ then all of the history appears equally within Q and may affect the forecast, but if the evolving network is changing rapidly, or is even non-stationary, this will not be a good thing. On the other hand if we take b to be very large, then Q "lives in the moment", and there is no memory at all (we forecast tomorrow using only today's data): hence forecasts are likely to be extremely sensitive when the sequence is voilatile.

This age discounting form is extremely useful in applications and some experimentation should be made with b depending on the desired use of Q.

9.3 Katz Centrality revisited with continuous time

A problem with chunking an evolving network into discrete time-steps is that within each time-step we allow walks to traverse connected edges in any order. But had we taken a smaller time-step one edge might occire at a step before another and then they may only be traversed in time-order. For this reason one may want tio try smaller and smaller time-steps. Here we consider the limit.

Suppose we can observe our evolving network over continuous time rather than discrete time. In effect we have a time dependent adjacency matrix $A(t)$ where the elements switch from zero to one, or vice versa. It is defined for $t \geq 0$ (or on a suitable interval) taking values in A, and we shall assume that it is continuous from the right (so that we can take limits).

Suppose first that we somehow sample from $A(t)$ so as to discretise the continuous time evolving networks and try to calculate Q as in teh last section.

Fix $\delta t > 0$. Then we might define A_k to be the union of all of the connections existing at any time within the half open interval $((k-1)\delta t, k\delta t]$. Then as $\delta t \to 0$ while $K \to \infty$, we will have $A_K \to A(K\delta t^+)$ and we recover an instantaneous sample.

Now consider Q as in the last section, where we write $T = K\delta t$:

$$Q(T + \delta t) = (I + e^{-b\delta t}(Q(T) - I))(I - \alpha A_{K+1})^{-1}.$$

We cannot yet take the limit as $\delta t \to 0$ in this equation just as it is. If we reduce δt by dividing into m equal sub-steps then we obtain m new resolvent terms each in the form $(I - \alpha A_{K'})^{-1}$ as we update from T to $T + \delta t$. If $A(t)$ is constant between T and $T + \delta t$ then it is clear that this limit cannot converge. We would repeatedly update by multiplying by new resolvent terms for intervals that are arbitrarily small.

To get around this we must re-scale (renormalise) our formula so that it make sense in the limit. We modify our equation to become

$$Q(T + \delta t) = (I + e^{-b\delta t}(Q(T) - I))(I - \alpha A_{K+1})^{-\delta t}.$$

Then as $\delta t \to 0$ we are updating with successive terms that tend to the identity.

Notice that now we are employing the fractional power of a matrix, in this case fractional powers of the resolvents $(I - \alpha A)^{-1}$, which is itself a matrix valued function of a matrix.

We have introduced functions of matrices in an earlier section. We wish to define the logarithm of a matrix, $\log(I - \alpha A)$: $A(t)$ is always normal and non-negative with spectral radius assumed to satisfy $r = \rho(A) < 1/\alpha$, as usual. So the spectrum of $I - \alpha A$, which is also normal, is real and lies to the right of zero along the positive axis in the complex plane. Thus, under this assumption, the matrix logarithm and hence all powers less than or equal to one, both positive and negative, of $(I - \alpha A)$ exist.

We have

$$Q(T + \delta t) = (I + e^{-b\delta t}(Q(T) - I)) \exp(-\delta t \log(I - \alpha A_{K+1})).$$

Now we can allow $\delta t \to 0$ in order to obtain an ordinary differential equation (ODE) for a continuously defined centrality matrix:

$$\frac{dQ(T)}{dt} = -Q(T) \log(I - \alpha A(T)) + b(I - Q(t)).$$

Note all eigenvalues of $I - \alpha A(T)$ are real and lie between 0 and 1. So the log term is itself negative. Hence if $A(T)$ is constant for a while the first term encourages the exponential positive growth of Q.

In order to focus on vertex properties, we can take row and column sums in the matrix Q(T) to define the dynamic broadcast and dynamic receive centrality vectors just as before:

$$b(T) = Q\mathbf{s}, r(T) = Q^T\mathbf{s},$$

where $\mathbf{s} = (1, 1, ..., 1)^T$.

It is interesting to note that the receive centrality satisfies its own vector-valued ODE:

$$r'(t) = b(\mathbf{s}(t) - r(t)) - (\log(I - \alpha A(t)))^T r(t),$$

with $r(0) = \mathbf{s}$; which is a factor of n smaller in dimension than the matrix equation for Q and hence cheaper to calculate. By contrast, it is not possible to disentangle the broadcast centrality in this way. Intuitively, this difference arises because the node-based receive vector $r(t)$ keeps track of the overall level of information flowing into each vertex and this can be propagated forward in time as new links become available. However, the broadcast vector $b(t)$ keeps track of information that has flowed out of each vertex: it does not record where the information currently resides and hence we cannot update it based on $b(t)$ alone. So, with this methodology, real-time updating of the receive centrality is fundamentally simpler than real-time updating of the broadcast centrality.

This ODE approach dramatically enhances the existing snapshot-based paradigm for network centrality in terms of both data-driven simulation and theoretical analysis. Our framework fits very naturally into the context of online or digital recording of human interactions. The ODE setting conveniently avoids the need to discretise the network data into predefined time windows, an approach that can introduce inaccuracies and computational inefficiencies. By defining a continuous time dynamical system, we can solve with an off-the-shelf **adaptive time-stepping numerical ODE solvers**, so that time discretisation is performed "under the hood", in a manner that automatically handles considerations of accuracy and efficiency. In this way we may deal adaptively with dramatic changes in network behaviour.

We may take large time-steps when $A(t)$ is almost 0, as Q returns towards I. and small time-steps when $A(T)$ *explodes* into life.

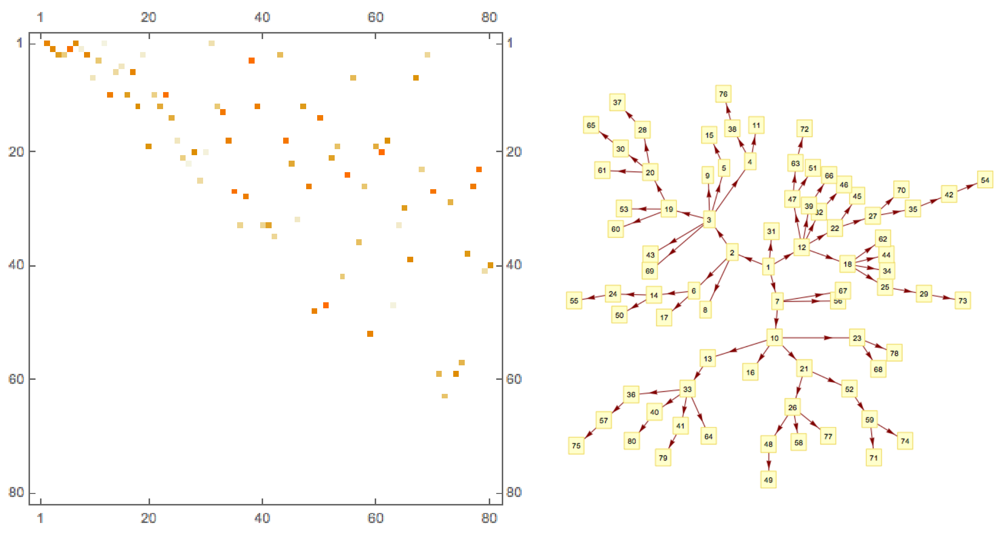


Figure 22: Here is the basic directed graph, as a directed tree on $n = 80$ vertices: adjacency matrix (left) with flows; and visualisation (right).

10 Trees and cycles in directed graphs

Inosculation is a natural phenomenon in which trunks, branches or roots of two trees grow together. It is biologically similar to grafting and such trees are referred to in forestry as "gemels", from the Latin word meaning "a pair". The branches first grow separately in proximity to each other until they touch. At this point, the bark on the touching surfaces is gradually abraded away as the trees move in the wind. Once the cambium of two trees touches, they sometimes self-graft and grow together as they expand in diameter.

The term inosculation is also used in the context of plastic surgery, as one of the three mechanisms by which skin grafts take at the host site. Blood vessels from the recipient site are believed to connect with those of the graft in order to restore vascularity.

10.1 A tree

Here we generated a directed graph on $n = 80$ vertices, in the from of a connected tree (no cycles) containing only directed flows (represented by random random independent variables in (0,10)), moving out from the centre at vertex v_1 towards various twigs. So every node, v_i $(i \geq 2)$, has in-degree of one with an in-flow coming from an a vertex, v_j, having a lower index, $j < i$. See Figure 22

10.2 Helmholtz-Hodge Decomposition

The Helmholtz-Hodge, or Hodge, decomposition represents any flow on a directed graph (with vertex set V and ordered edges in $V \times V$) as a sum of a potential flow and a divergence free flow (that is, a rotational flow). We may calculate the Hodge potential at each vertex, and in this case of our tree we have a potential flow out from vertex 1 towards the twigs, while the divergence free (rotational flow) is equal to zero (since there are no cycles anyway).

We should associate the potential flow with *trophic* or periphery flows, and the divergence free flow with core, circulatory flows.

We briefly recap the Hodge-Decomposition, based on [35].

Let A denotes the usual binary adjacency matrix. Let $B \geq 0$ denote the corresponding real-values weighted adjacency matrix with positive flow whenever the corresponding terms in A is one.

Let $W = (A + A^T)/2$ the symmetric part of A. Let $F = B - B^T$ denote the net flows.

Then the Hodge decomposition is a unique way to write

$$F_{ij} = W_{ij}(\phi_i - \phi_j) + F_{ij}^{Circ}.$$

Here $\phi : V \to \mathbb{R}$ is the Hodge potential and we use ϕ_i to denote the value of the Hodge potential at vertex v_i. F_{ij}^{Circ} denotes the divergence free flow on each edge, meaning that it satisfies

$$\sum_{j=1}^{N} F_{ij}^{Circ} = 0.$$

Using this last condition we must have

$$\sum_{j=1}^{N} F_{ij} = \sum_{j=1} L_{ij}\phi_j,$$

where L is the usual graph Laplacian

$$L = -W + \text{Diag}(W.\mathbf{1}),$$

where $\mathbf{1} = (1, 1, ..., 1)^T$.

Since $L.\mathbf{1} = 0$, we need one extra condition: say, $0 = \sum_{j=1}^{N} \phi_j$. Then ϕ is completely determined, and F_{ij}^{Circ} is the difference between F and the gradient flow.

10.3 Perturbing our tree

Now returning to our tree example, we select two twig-end nodes: v_{37} and v_{73}. We join them together by adding a single edge, from v_{37} to v_{73} with a flow equal to 4.99279.

Now we can see from Figure 23 that there is a single long, 12 vertex, cycle:

$$\{v_1, v_2, v_3, v_{19}, v_{20}, v_{28}, v_{37}, v_{73}, v_{29}, v_{25}, v_{18}, v_{12}, v_1\}.$$

Recalculating the Hodge potential we may compare the resulting potential directly, with those for the unadulterated tree; see figure 24 where blue values are for the tree and the red values are for the grafted tree.

The resulting decomposition of the tree with the extra edge (and cycle) now has a non-trivial flow around the 12-cycle, is shown in Figure 25.

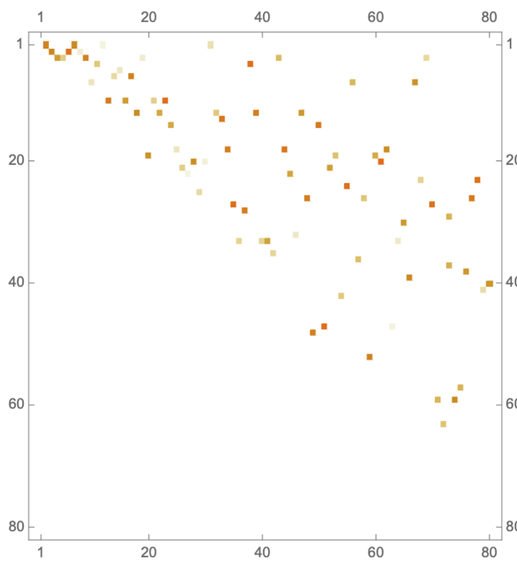
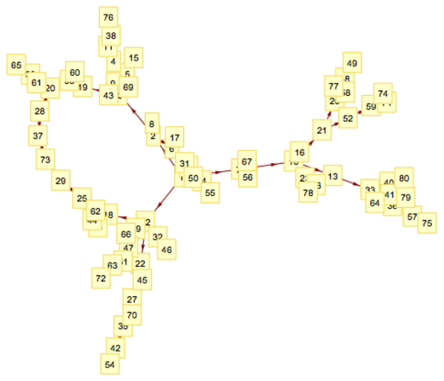



Figure 23: An extra directed edge is added from v_{37} to v_{73}: in the the 37th row and 73rd column in the weighted adjacency matrix, B (left), resulting in a single 12 vertex cycle (right). Compare with that shown in Figure 1: there is a single extra edge at (37,73).

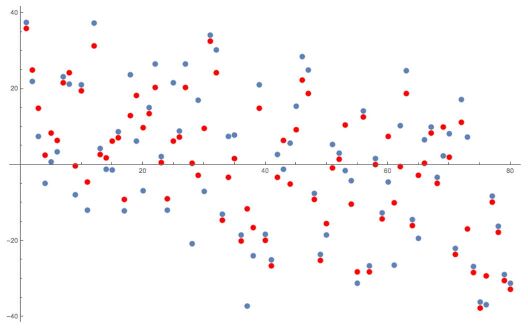


Figure 24: Hodge potentials for each vertex of the directed tree graph (blue) and for the directed tree graph now amended to contain an extra edge - and thus a long cycle (red values).

10.4 Breaking cycles - un-grafting

Next we consider working this process in reverse. Since trees are such easy graphs to deal with, for all sorts of calculations, we consider how best to break cycles within a given graph, by selecting and removing particular edges. This is the opposite of the above grafting process.

Consider the following example

$$A = \begin{pmatrix} 0&0&0&0&1&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&1&0&0&1&0\\ 0&0&1&0&0&0&0&0\\ 1&0&0&1&0&1&0&0\\ 0&0&0&0&1&0&0&0\\ 1&0&0&0&0&0&0&0\\ 1&0&0&1&0&0&1&0 \end{pmatrix} \quad B = \begin{pmatrix} 0&0&0&0&5.715&0&0&0\\ 0&0&0&6.651&0&0&0&0\\ 0&0&0&0.390&0&0&2.356&0\\ 0&0&2.250&0&0&0&0&0\\ 6.897&0&0&0.3282&0&1.215&0&0\\ 0&0&0&0&1.511&0&0&0\\ 5.144&0&0&0&0&0&0&0\\ 3.574&0&0&8.096&0&0&5.891&0 \end{pmatrix}$$

In this case we may use the Hodge decomposition to split the flow into a potential flow and a diver-

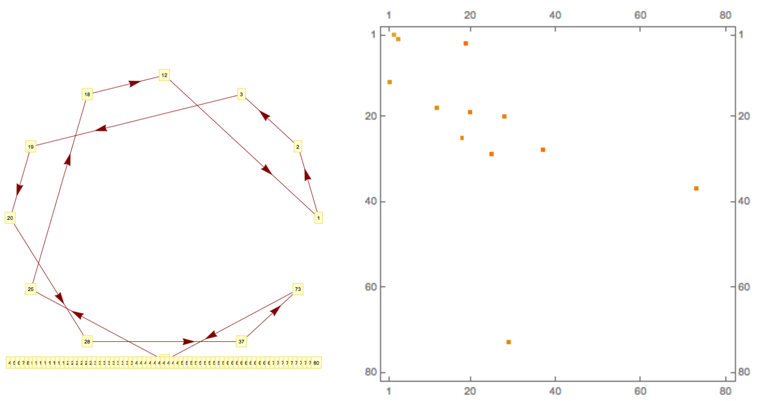


Figure 25: Here we show only the divergence free, circulatory, flow.

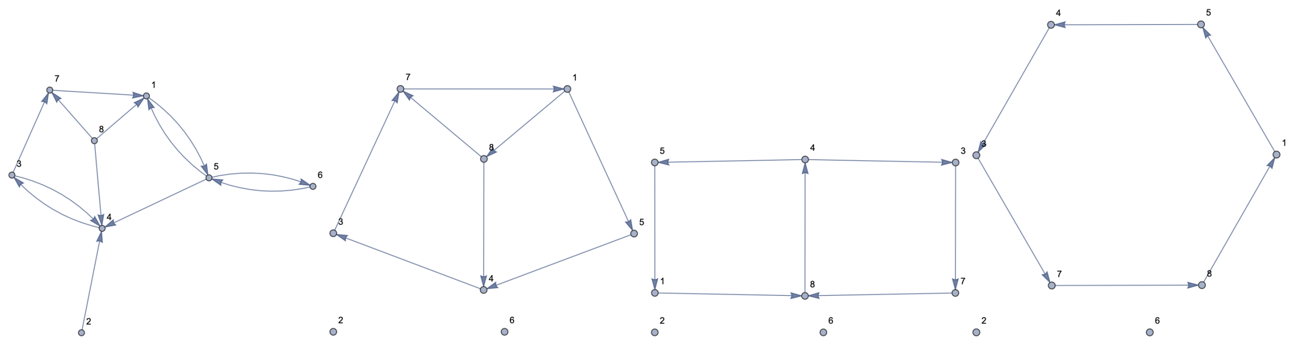


Figure 26: From the Left: i) the original network flow; ii) the associated divergence free flow, following the Hodge decomposition; iii) the divergence flow following the deletion of the directed edge v_7v_1; iv) the divergence flow following the further deletion of the directed edge v_8v_4.

gence free flow: the latter is

$$F^{Circ} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0.407 & 0 & 0 & 3.113 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2.691 & 0 \\ 0 & 0 & 2.691 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.407 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3.520 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2.284 & 0 & 0 & 0.829 & 0 \end{pmatrix}$$

We take the Euclidean norm of this last (defined to equal the sum of the squares of the elements): we have

$$\beta = 42.80.$$

In Figure 26 we show the original network flow and the associated divergence free flow, following the Hodge decomposition. We will examine how the successive deletion ("cutting") of edges reduces this norm of the divergence free part of the flow.

Next we evaluate the sensitivity of β to the deletion of each edge in turn. We find that the deletion of edge v_7v_1 reduces β the most, so that, post deletion, we have $\beta = 13.16$. The divergence free flow is also shown in Figure 26. Next we find that deleting the further edge v_8v_4 reduce β by the most, so that, post that second deletion, we have $\beta = 0.12$. The divergence free flow is also shown in Figure 26. Finally the removal of any of the remaining edges around the remaining single cycle (within the divergence free flow) will result in a zero divergence free flow ($\beta = 0.0$). The remaining part of the original flow will be a potential flow, on a tree.

References

[1] E.T. Jaynes, Probability theory: the Logic of Science, 2002, `https://bayes.wustl.edu/etj/prob/book.pdf`.

[2] P. Grindrod (2014), Mathematical Underpinnings of Analytics, Oxford University Press, Oxford. ISBN: 9780198725091.

[3] Watts, D. J.; Strogatz, S. H. (1998), Collective dynamics of 'small-world' networks, Nature. 393 (6684): 440–442.

[4] Newman, Mark E.J., The structure and function of complex networks, SIAM review 45.2 (2003): 167-256.

[5] B. Karrer and M.E. Newman (2011), Stochastic block models and community structure in networks. Phys. Rev. E 83, 016107.

[6] A. Elliott, A. Chiu, M. Bazzi, G. Reinert, and M. Cucuringu (2020), Core–periphery structure in directed networks. Proceedings of the Royal Society A, 476, 20190783.

[7] F. Tudisco and D.J. Higham (2019), A nonlinear spectral method for core-periphery detection in networks, SIAM J. Math. Data Sci. Vol. 1, No. 2.

[8] David Liben-Nowell, Jon Kleinberg, The link prediction problem for social networks. Journal of the Association for Information Science and Technology 58.7 (2007): 1019-1031.

[9] Grindrod P. and Lee T. E. 2016, Comparison of social structures within cities of very different sizesR. Soc. open sci. 3150526150526

[10] P. Grindrod and T.E. Lee (2016), Data from: Comparison of social structures within cities of very different sizes, Dryad, `https://datadryad.org/stash/dataset/doi:10.5061/dryad.2gf23`.

[11] P. Grindrod, D.J. Higham, M.C. Parsons, Bistability through triadic closure, Internet Math. 8(4) (2012), `https://www.internetmathematicsjournal.com/article/1526`.

[12] G. West (2017), Scale: the universal laws of growth, innovation, sustainability, and the pace of life in organisms, cities, economies, and companies. London, United Kingdom: Penguin Press.

[13] Feller, W., (1971), An Introduction to Probability Theory and its Applications, Vol II, Wiley New York, 1971.

[14] McBride, A.C. (1987), Semigroups of linear operators: an introduction, Pitman Research Notes in mathemtics 156, Longman.

[15] P. Grindrod and D.J. Higham (2018), High modularity creates scaling laws, Scientific Reports, Vol. 8, No: 9737 `https://www.nature.com/articles/s41598-018-27236-0`.

[16] M.E.J. Newman (2010), Networks: An Introduction, Oxford University Press, Oxford, NewYork.

[17] V.D. Blondel, J.-L. Guillaume, R. Lambiotte and E. Lefebvre (2008), Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory & Experiment, Vol. 2008.

[18] Peter Grindrod, Scaling laws for properties of random graphs that grow via successive combination, Journal of Complex Networks, Volume 10, Issue 3, June 2022, cnac024, https://doi.org/10.1093/comnet/cnac024

[19] L.M. Bettencourt, J. Lobo, D. Helbing, C. Kühnert and G.B. West (2007). Growth, innovation, scaling, and the pace of life in cities, Proc. of the Nat. Acad. of Sciences, 104(17), 7301–7306.

[20] D.J. Higham, M. Batty, L.M.A. Bettencourt, D.V. Greetham & P. Grindrod (2017), An overview of city analytics, Roy. Soc. Open Sci. 2017, https://doi.org/10.1098/rsos.161063.

[21] G. West, J. Brown and B. Enquist (1999), The Fourth Dimension of Life: Fractal Geometry and Allometric Scaling of Organisms, Science, Vol 284, Issue 5420, pp. 1677-1679.

[22] C. Lee and D.J. Wilkinson (2019), A review of stochastic block models and extensions for graph clustering. Appl Netw Sci 4, 122. https://doi.org/10.1007/s41109-019-0232-2.

[23] E.N. Gilbert (1959), Random graphs, Annals of Mathematical Statistics, 30 (4): 1141–1144.

[24] A.-L. Barabasi and R. Albert (1999), Emergence of scaling in random graphs, Science 286 (5439): 509-512.

[25] B. Bollobas, O. Riordan, J. Spencer and G. Tusnady (2001), The degree sequence of a scale-free random graph process, Random Structures and Algorithms 18 (3): 279290.

[26] B. Bollobás (2001). Random Graphs, 2nd Ed., Cambridge University Press.

[27] Lambiotte, R., et al. Structural transitions in densifying networks. Physical review letters 117.21 (2016): 218301

[28] P. Grindrod and T.E. Lee (2016), Comparison of social structures within cities of very different sizes, Roy. Soc. Open Sci. 3150526150526, https://doi.org/10.1098/rsos.150526.

[29] E.B. Ravasz and A.L. Barabási (2003), Hierarchical organization in complex networks, Physical Review E. 67 (2).

[30] J. Noh (2003), Exact scaling properties of a hierarchical network model, Phys. Rev. E 67 (4).

[31] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos and Z. Ghahramani (2010), Kronecker Graphs: An Approach to Modelling Networks, UK Journal of Machine Learning Research 11, 985-1042.

[32] T. Tao (2016), An improved MCMC algorithm for generating random graphs from constrained distributions. Network Science, 4(1), 117-139. doi:10.1017/nws.2015.35.

[33] P. Grindrod (1991), Patterns and Waves: Theory and Applications of Reaction-diffusion Equations (Oxford Applied Mathematics & Computing Science S.),OUP.

[34] N.J. hIgham, Matrix Valued Functions,

[35] Fujiwara Y., Islam R. (2020) Hodge Decomposition of Bitcoin Money Flow. In: Pichl L., Eom C., Scalas E., Kaizoji T. (eds) Advanced Studies of Financial Technologies and Cryptocurrency Markets. Springer, Singapore. .

[36] A. Bovet and P. Grindrod (2020), The Activity of the Far Right on Telegram, ResearchGate preprint, `https://tinyurl.com/y8472s7x`.

[37] P. Grindrod (2018), On human consciousness: A mathematical perspective, Network Neuroscience 2018 2:1, 23-40.

[38] B. Karrer and M.E. Newman (2011), Stochastic block models and community structure in networks. Phys. Rev. E 83, 016107.

[39] A. Elliott, A. Chiu, M. Bazzi, G. Reinert, and M. Cucuringu (2020), Core–periphery structure in directed networks. Proceedings of the Royal Society A, 476, 20190783.

[40] F. Tudisco and D.J. Higham (2019), A nonlinear spectral method for core-periphery detection in networks, SIAM J. Math. Data Sci. Vol. 1, No. 2.

[41] E.T Jaynes (Ed. G.L. Bretthorst) (2003), Probability Theory: The Logic of Science. Cambridge University Press.