C5.4 Networks

R. Lambiotte

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 is a mathematical course, with theoretical results and derivations, but with real-world applications in mind, as will be stressed at different points of this manuscript. Several parts of these notes are based on the introductory chapters of "A guide to temporal networks", N. Masuda and R. Lambiotte, World Scientific 2016.

CONTENTS

I. MOTIVATION

Graphs and networks have a long history, dating to the seminal works of Euler and the famous story of the seven bridges of Konigsberg. Since then, networks have become a central toolbox in a range of disciplines, from social systems to the Web, passing by neuroscience. Within the framework of Network Science, a system is modeled as a set of nodes, representing the individual units of the system, and a set of links, representing the dyadic relationships between these units. Once this abstraction has been operated, standardized tools can then be applied to otherwise very different systems.

There exist many libraries and software to analyse and visualise networks. It is important to emphasise here that the main aim of this course is not to learn how to use these tools (even if you are expected to learn how to use them along the way), but to understand the principles behind them. In this spirit, tutorials will cover a mixture of analytical problems and computational ones. For the latter, useful libraries to manipulate networks are NetworkX and iGraph. For network visualisation, Gephi is a popular solution. For those interested in solving the problems in Python, the installation of jupyter notebooks (and standard Python libraries) can be found here: http://jupyter.org/install.html . Note that problem sheets can be solved in the language of your choice and that libraries are ported in a variety of languages (including Matlab, C, C++, etc.).

First, one should keep in mind that a network, for instance a social network, is only an abstraction of the original social system, because it simplifies the essence of a social interaction by a binary value (exists or not), without accounting for the richness of social interactions, with different intensities (je l'aime un peu, beaucoup, etc.) and different natures (my friends are not my enemies; acquaintances are not the same as family members, etc.). One should also keep in mind that most network measurements are subject to measurement errors and missing data, and it is thus important to ensure that observations are robust to noise. See for instance "Subnets of scale-free networks are not scale-free: Sampling properties of networks", by Michael P. H. Stumpf, Carsten Wiuf and Robert M. May for a discussion. With these caveats in mind, here are some problems that we should be able to solve at the end of this course:

- Quantify the structural properties of nodes, for instance to estimate their importance or their centrality.

- Quantify the structure of networks, in particular their connectivity patterns and their statistical properties.

- Build random models of networks, in order to test the importance of structural factors and develop statistical tests.

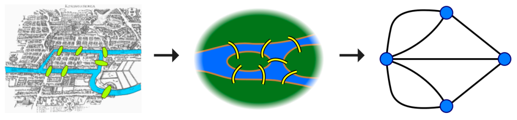

FIG. 1. Map of Konigsberg in Euler's time showing the actual layout of the seven bridges, highlighting the river Pregel and the bridges. Graphs can be used to remove unnecessary (geographical) details, in order to highlight the connectivity patterns between different areas. Taken from Wikipedia.

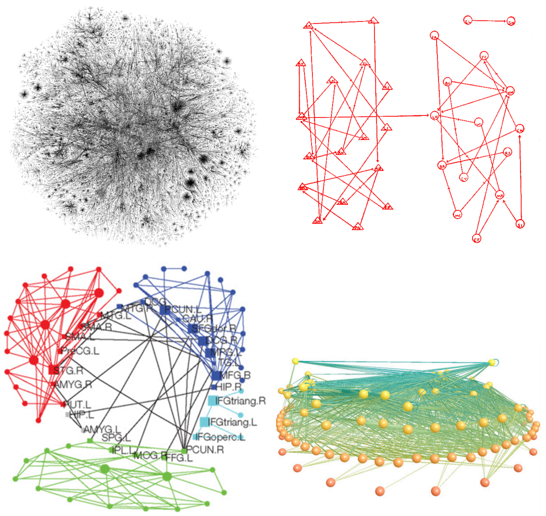

FIG. 2. Graphical representation of the network structure of the Internet (nodes are "Class C subnets"), the seminal sociogram of Moreno (nodes are kids in a primary school), a functional brain network (nodes are brain regions) and a foodweb (nodes are competing species).

- Extract groups of similar nodes, such as social circles, that is cluster the nodes into communities or modules.

- Identify the structural properties that affect the function and dynamics of the system, for instance the spreading of information or of a virus on the network.

We illustrate in Figure 3 such results on the so-called Zachary Karate-Club network, representing the social interactions between members of a Karate club, and one of the most studied networks due to its small size and to its so-called community structure.

Practical organisation: I plan to cover the material of one chapter per lecture. As you will see, there are suggested problems at the end of several sections. There will be a strong overlap between them and those of the problem sheets.

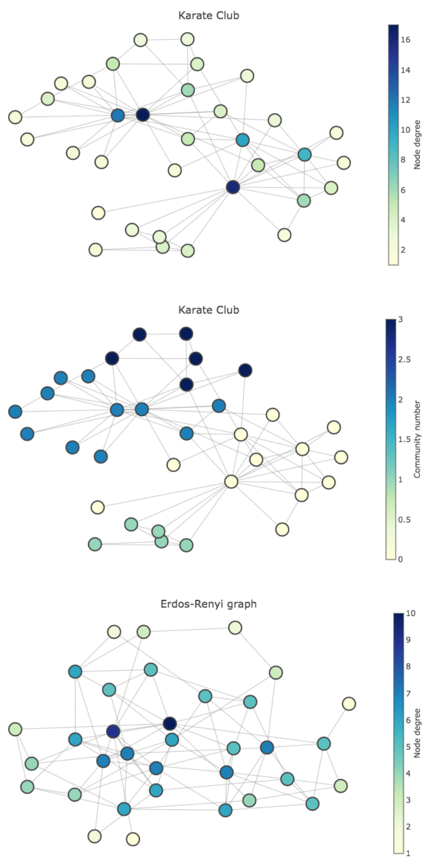

FIG. 3. First two figures: graphical representation fo the so-called Karate Club network. Node colors correspond to the node degree and its participation in a community, as estimated by the Louvain algorithm respectively. In the bottom figure, visualisation of a realisation of an Erdős-Rényi with the same original connectivity than the original network.

II. READING LIST

There exist several books and reviews covering different aspects of network science and graph mining. Here is a non-exhaustive list covering some aspects covered in this course.

- Solid piece of work on spectral properties of networks: *Spectral Graph Theory*, Chung, F.R.K. (1997, American Mathematical Society, Providence).
- Comprehensive book on networks. Excellent entry point: *Networks — An Introduction*, Newman, M. E. J. (2010). (Oxford University Press, Oxford).
- Review on epidemic spreading on networks: *Epidemic processes in complex networks*, Pastor-Satorras, R., Castellano, C., Van Mieghem, P. and Vespignani, A. (2015), *Reviews of Modern Physics* **87**, pp. 925–979.
- Bible of random networks: *Random Graphs, Second Edition*, Bollobás, B. (2001, Cambridge University Press, Cambridge).
- Exhaustive review of community detection on networks: *Community detection in graphs*, Fortunato, S. (2010). *Physics Reports* **486**, pp. 75–174.
- Nice introductory book on the computational aspects of Pagerank: *Google's PageRank and beyond: The Science of Search Engine Rankings*, Langville, A. N. and Meyer, C. D. (2006, Princeton University Press, Princeton).
- Short overview of recent on dynamics on network: *Dynamical systems on networks: A tutorial*, Porter, M. A. and Gleeson, J. P. (2014), Preprint: arXiv:1403.7663v2.
- Recent book with a focus on kernels methods and similarity measures: *Algorithms and Models for Network Data and Link Analysis*, Frans Fouss, Marco Saerens, Masashi Shimbo (2016, Cambridge University Press)

In addition, for the reader interested to know more on power-laws:

- Newman, M. E. J. (2005). Power laws, Pareto distributions and Zipf's law, *Contemporary Physics* **46**, pp. 323–351.
- Clauset, A., Shalizi, C. R. and Newman, M. E. J. (2009). Power-law distributions in empirical data, *SIAM Review* **51**, pp. 661–703.

and, for an introduction to stochastic processes, I would recommend:

- Feller, W. (1971). *An Introduction to Probability Theory and Its Applications, Volume II, Second Edition* (John Wiley & Sons).
- Klafter, J. and Sokolov, I. M. (2011). *First Steps in Random Walks — From Tools to Applications* (Oxford University Press, Oxford).
- P.L. Krapivsky, S. Redner, *A Kinetic View of Statistical Physics*, Boston University, E. Ben-Naim,

III. MATHEMATICAL TOOLBOX

In this section, we introduce mathematical tools used in the subsequent chapters. They are a collection of elementary probability theory and linear algebra, and somewhat advanced materials on stochastic processes.

A. Probability

1. Discrete variables

A random variable is a variable that takes its value stochastically. A discrete random variable X is defined on a set S of possible values x such that $p(x) \geq 0$ for any $x \in S$ and $\sum_{x\in S} p(x) = 1$, where $p(x)$ is the probability that X takes value x; we use p to denote the probability throughout these notes. A common example is a fair dice for which $S = \{1, 2, 3, 4, 5, 6\}$ and $p(x) = 1/6$ for each $x \in S$. If one throws the dice many times, the fraction of times with which one observes 1 tends to 1/6.

A probabilistic event is specified by a certain subset of possible values in X. In the previous example, the event that a dice produces an odd number is represented as the event $X \in \{1, 3, 5\}$. When two events X_1 and X_2 are mutually exclusive, i.e., no value x belongs to both sets, we obtain

$$p(X_1 \text{ or } X_2) = p(X_1) + p(X_2). \tag{1}$$

Information about one event may inform the probability of another event. For instance, knowing that the dice produces an odd number increases the probability of $X = 1$ and decreases the probability of $X = 2$ to zero. Such information is quantified by the conditional probability. The conditional probability of X given Y is

$$p(X|Y) = \frac{p(X \text{ and } Y)}{p(Y)}. \tag{2}$$

By swapping X and Y in Eq. (2), we obtain $p(Y|X) = p(X \text{ and } Y)/p(X)$. By combining this equation with Eq. (2), we obtain the Bayes rule for conditional probabilities:

$$p(X|Y) = \frac{p(Y|X)p(X)}{p(Y)}. \tag{3}$$

Two events X and Y are said to be independent if the probability that X occurs is not affected by whether Y has occurred and vice versa. In other words,

$$p(X|Y) = p(X|\text{not } Y) = p(X). \tag{4}$$

When two events are independent, the probability that both events occur is the product of the probabilities that each event occurs, i.e.,

$$p(X \text{ and } Y) = p(X)p(Y). \tag{5}$$

In terms of the values of X and Y, we obtain

$$p(x, y) = p(x)p(y), \tag{6}$$

where $p(x, y)$ is the joint probability that $X = x$ and $Y = y$. The marginal distribution, i.e., the probability that $X = x$ regardless of the value of Y, is obtained by

$$p(x) = \sum_y p(x, y). \tag{7}$$

In principle, the random variable X can be either numerical (e.g., $1, 2, 3$) or non-numerical (e.g., white, red, black). In the former case, mostly relevant in these notes, there exist different types of tools to characterise its properties. For instance, the expected value, or average, is defined as

$$\langle x\rangle = \sum_x x p(x). \tag{8}$$

We use $\langle\cdot\rangle$ to denote the mean throughout these notes. The nth moment of X is defined by

$$\langle x^n\rangle = \sum_x x^n p(x), \tag{9}$$

where n is typically a positive integer, generalising Eq. (8). The second moment $\langle x^2\rangle$ is related to the variance as follows:

$$\sigma^2 = \langle (x - \langle x\rangle)^2\rangle = \langle x^2\rangle - \langle x\rangle^2, \tag{10}$$

where σ is the standard deviation. Moments can be generalised to the case of multiple random variables, often to evaluate correlations between them. A familiar measure of linear dependence between two variables is the Pearson correlation coefficient defined by

$$\rho_{X,Y} = \frac{\langle (x - \langle x\rangle)(y - \langle y\rangle)\rangle}{\sigma_X \sigma_Y}, \tag{11}$$

where σ_X and σ_Y are the standard deviations of X and Y, respectively.

Here is a short list of frequently used discrete distributions:

1. The Bernoulli distribution takes only two possible values, 0 or 1, i.e., failure or success, with probabilities $1 - p$ and p respectively. The mean $\langle x\rangle = p$ and the variance $\sigma^2 = p(1 - p)$.

2. The binomial distribution describes the outcome of n independent and identically distributed random variables generated by the Bernoulli distribution with parameter p. The probability that exactly m successes are observed is given by

$$p(m) = \binom{n}{m} p^m (1 - p)^{n-m}, \tag{12}$$

where $0 \leq m \leq n$. Note that $p^m(1-p)^{n-m}$ is the probability that a particular sequence containing exactly m successes is realised, and

$$\binom{n}{m} = \frac{n!}{m!(n-m)!} \tag{13}$$

is the number of sequences of length n that possess exactly m successes. We obtain $\langle m \rangle = np$ and $\sigma^2 = np(1-p)$.

3. The geometric distribution is defined via the waiting time before a success is observed, in a sequence of independent and identically distributed random variables obeying the Bernoulli distribution. The geometric distribution is defined as

$$p(m) = (1-p)^m p, \tag{14}$$

where $m = 0, 1, \ldots$. The factor $(1-p)^m$ corresponds to m consecutive failures, and p to the success on the $(m+1)$th trial. We obtain $\langle m \rangle = (1-p)/p$ and $\sigma^2 = (1-p)/p^2$.

4. The Poisson distribution is given as the limit of the binomial distribution as $n \to \infty$ while the mean np tends to a constant λ (therefore $p \to 0$). The Poisson distribution is given by

$$p(m) = \frac{m^\lambda e^{-\lambda}}{m!}, \tag{15}$$

where $m = 0, 1, \ldots$. We obtain $\langle m \rangle = \sigma^2 = \lambda$.

Ex.III.1 : Using the computer language of your choice, calculate the mean and variance of a Bernouilli process, as a function of p.

2. Continuous variables

Continuous random variables describe variables that take any value in a continuum of values, typically any real values or non-negative real values. Continuous random variables are set by their probability density function, $f(x) (\geq 0)$, defined such that the probability of observing any value between a and b is equal to

$$p(a \leq X \leq b) = \int_a^b f(x)\mathrm{d}x, \tag{16}$$

where $\int_{-\infty}^{\infty} f(x)\mathrm{d}x = 1$.

Most operations for discrete random variables are easily transferred to continuous random variables via replacement of sums by integrals. For instance, the joint probability density function $f(x, y)$ for continuous random variables satisfies

$$p(a \leq X \leq b, c \leq Y \leq d) = \int_a^b \int_c^d f(x, y)\mathrm{d}x\mathrm{d}y. \tag{17}$$

The moments of the distribution are given by

$$\langle x^n \rangle = \int_{-\infty}^{\infty} x^n f(x)\mathrm{d}x. \tag{18}$$

If two random variables are independent, their joint distribution factorises into the product of their marginals:

$$f(x, y) = f(x)f(y). \tag{19}$$

Finally, it is often practical to focus on the cumulative probability $F(x)$, defined as the probability that the variable takes a value smaller than x:

$$F(x) = \int_{-\infty}^{x} f(x')\mathrm{d}x'. \tag{20}$$

By definition, $F(-\infty) = 0$ and $F(\infty) = 1$. We also often use the complementary cumulative probability, also called the survival probability or survival function, given by

$$\tilde{F}(x) = \int_x^{\infty} f(x')\mathrm{d}x' = 1 - F(x). \tag{21}$$

Classical distributions for continuous random variables include the following ones:

1. The uniform distribution takes a constant probability on interval $[a, b]$, i.e.,

$$f(x) = \frac{1}{b-a} \quad (a \leq x \leq b). \tag{22}$$

We obtain $\langle x \rangle = (b-a)/2$ and $\sigma^2 = (b-a)^2/12$.

2. The exponential distribution is defined by

$$f(x) = \lambda e^{-\lambda x} \quad (x \geq 0). \tag{23}$$

Its cumulative distribution is given by

$$F(x) = 1 - e^{-\lambda x} \quad (x \geq 0). \tag{24}$$

We obtain $\langle x \rangle = 1/\lambda$ and $\sigma^2 = 1/\lambda^2$.

3. The Gaussian or normal distribution is defined by

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (-\infty < x < \infty), \tag{25}$$

where μ is the average and σ^2 is the variance. The Gaussian distribution exhibits a bell shape. The Gaussian distribution can been seen as a continuous limit of the binomial distribution. The binomial distribution with n trials, each with probability p, converges to the Gaussian distribution with mean np and variance $np(1-p)$ owing to the central limit theorem. In particular for this reason, the Gaussian distribution is frequently observed in empirical data.

Ex.iii.2 : Given an arbitrary continuous distribution determined by the cumulative probability $F(x)$, design a computational method that generates the corresponding random variables.

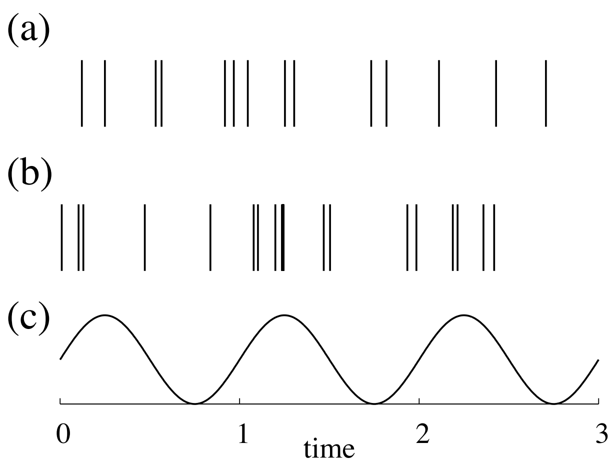

FIG. 4. Homogeneous and non-homogeneous Poisson processes. (a) An event sequence generated by the (homogeneous) Poisson process with $\lambda = 5$. (b) An event sequence generated by the non-homogeneous Poisson process with the sinusoidal rate shown in (c), i.e., $\lambda(t) = 5(1 + \sin 2\pi t)$.

B. 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 λ 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. An event sequence generated by a Poisson process is shown in Fig. 4(a).

Let us 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, we obtain

$$\begin{aligned} q =& p(1,\Delta t) = \lambda \Delta t, \\ 1-q =& p(0,\Delta t) = 1 - \lambda \Delta t, \end{aligned} \tag{26}$$

when Δt (and hence q) is small. For any $n \geq 1$, we obtain

$$\begin{aligned} p(n,t+\Delta t) =& p(n,t)p(0,\Delta t) + p(n-1,t)p(1,\Delta t) \\ =& p(n,t)(1-\lambda\Delta t) + p(n-1,t)\lambda\Delta t. \end{aligned} \tag{27}$$

Equation (27) 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 $[0,t]$ and one event in $[t,t+\Delta t]$. This equation relates the probability of a system at a certain time to that at a previous time, and hence is an example of master equation, which we will encounter many times in the following.

In the limit $\Delta t \to 0$, Eq. (27) is reduced to

$$\frac{\mathrm{d}p(n,t)}{\mathrm{d}t} = \lambda p(n-1,t) - \lambda p(n,t). \tag{28}$$

For $n = 0$, we obtain

$$\frac{\mathrm{d}p(0,t)}{\mathrm{d}t} = -\lambda p(0,t), \tag{29}$$

which results in

$$p(0,t) = e^{-\lambda t}. \tag{30}$$

To derive Eq. (30), we have used the initial condition $p(0,0) = 1$, i.e., no event has occurred at $t = 0$. Because $p(0,t)$ is equal to the probability that no event occurs in $[0,t]$, the probability that the first event occurs in $[t,t+\Delta t]$ is given by $p(0,t)-p(0,t+\Delta t)$. Equation (30) implies that the inter-event time between two consecutive events, denoted by τ, is distributed according to

$$\psi(\tau) = -\frac{\mathrm{d}p(0,\tau)}{\mathrm{d}\tau} = \lambda e^{-\lambda\tau}. \tag{31}$$

The inter-event time of a Poisson process is distributed according to the exponential distribution. The mean inter-event time is given by

$$\langle\tau\rangle = \int_0^\infty \tau\psi(\tau)\mathrm{d}\tau = \frac{1}{\lambda}. \tag{32}$$

In Poisson processes, different inter-event times τ are independent of each other because event times before the last event time t do not affect the time τ to the next event since t. 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

$$p(\tau > t_1 + t_2 | \tau > t_2) = p(\tau > t_1). \tag{33}$$

Equation (33) indicates that the length of time, t_2, for which we have waited, actually without an event, does not affect the time of the next event. The time to the next event starting from $t = t_2$, i.e., 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. (28) recursively, we obtain

$$p(n,t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} \tag{34}$$

for any $n \geq 0$. Therefore, the probability of observing n events in $[0,t]$ obeys the Poisson distribution with mean and variance equal to λt. As discussed in Section III A 1, 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 $(t/\Delta t) \times q = \lambda t$.

A method to generate an event sequence obeying a Poisson process is to generate events one by one by independently drawing the inter-event time τ according to Eq. (31). An alternative method when the final time $t_{\max}$ is specified is to first draw the number of events n in $[0, t_{\max}]$ according to the Poisson distribution with parameter $\lambda t_{\max}$. Then, distribute each of the n events independently and uniformly on $[0, t_{\max}]$. The second method exploits the memoryless property of Poisson processes.

Let us introduce two extensions of Poisson processes. The first is non-homogeneous (also called inhomogeneous) Poisson processes, in which the event rate $\lambda(t)$ is time-dependent. In other words, an event occurs in $[t, t+\Delta t]$ with probability $\lambda(t)\Delta t$. This model is motivated by the fact that event rates seem to vary over time in a majority of empirical data. An event sequence generated by a non-homogeneous Poisson process is shown in Fig. 4(b). In this example, the rate is modulated sinusoidally as shown in Fig. 4(c). For a non-homogeneous Poisson process, Eq. (34) is extended as

$$p(n,t) = \frac{\Lambda(t)^n}{n!} e^{-\Lambda(t)}, \tag{35}$$

where

$$\Lambda(t) = \int_0^t \lambda(t')\mathrm{d}t'. \tag{36}$$

The distribution of inter-event times, conditioned by the last event at $t = 0$, is given by

$$\psi(\tau) = \lambda(\tau) e^{-\Lambda(\tau)}, \tag{37}$$

which extends Eq. (31). It should be noted that Eq. (37) is properly normalised, i.e., $\int_0^\infty \psi(\tau)\mathrm{d}\tau = 1$.

The second extension of Poisson processes, called renewal processes, considers a general distribution of inter-event times, $\psi(\tau)$. The renewal property dictates that different inter-event times are independent of each other and drawn from the same distribution $\psi(\tau)$. When $\psi(\tau) = \lambda e^{-\lambda\tau}$, we recover a Poisson process. When $\psi(\tau) = \delta(\tau - 1)$, events periodically happen at all integer times. To obtain the time of the nth event or the number of events in a given time period, we need to sum independent random variables generated according to $\psi(\tau)$. In that case, it is convenient to study the problem in a frequency domain and to consider the Laplace transform, related to the Fourier transform defined below.

Ex.III.3 : Take at random 10^4 numbers in the interval $]0,1[$, and plot the histogram of the $10^4 - 1$ lengths of the resulting intervals.

C. Random walks and diffusion

The Poisson processes provide a basic model for modelling temporal events, i.e., when random events take place. Random walk processes are its counterpart for modelling trajectories in space, i.e., 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 (i.e., 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 $\int_{-\infty}^{\infty} f(r)\mathrm{d}r = 1$.

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 independent events, we obtain the following master equation:

$$p(x;t) = \int_{-\infty}^{\infty} f(x-x')p(x';t-1)\mathrm{d}x' \tag{38}$$

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

Equation (38) for the entire range of x is more easily solved in the Fourier domain. The Fourier transform is defined by

$$\hat{p}(k;t) \equiv \int_{-\infty}^{\infty} p(x;t)e^{-ikx}\mathrm{d}x, \tag{39}$$

The original function is recovered through the inverse Fourier transform given by

$$p(x;t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} \hat{p}(k;t)e^{ikx}\mathrm{d}k. \tag{40}$$

Probability $p(x;t)$ is thus a combination of the oscillatory functions e^{ikx}, which form a base in the space of functions. The Fourier mode $\hat{p}(k;t)$ is the projection of $p(x;t)$ onto this base. The Fourier transform of $f(x)$, $\hat{f}(k)$, is called the structure function of the random walk. The Taylor expansion around $k=0$ yields

$$\begin{aligned}\hat{p}(k;t) =& \langle e^{-ikx}\rangle \\ =& 1 - ik\langle x\rangle - \frac{1}{2}k^2\langle x^2\rangle + O(k^3). \end{aligned} \tag{41}$$

Equation (41) implies that the moments of $p(x;t)$ are obtained from the derivatives of $\hat{p}(k;t)$ at $k=0$.

The Fourier transform transfers a convolution, such as Eq. (38), to a product. For this reason, working in the Fourier domain is often recommended when dealing with problems involving summations of random variables. Equation (38) is equivalent to

$$\hat{p}(k;t) = \hat{f}(k)\hat{p}(k;t-1). \tag{42}$$

If the walker is initially located at $x=0$, such that $p(x;0)=\delta(x)$, which translates to $\hat{p}(k;0)=1$, we obtain

$$\hat{p}(k;t) = \left[\hat{f}(k)\right]^t. \tag{43}$$

Using the inverse Fourier transform (Eq. (40)), the formal solution of the random walk in the time domain is given by

$$p(x;t) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\left[\hat{f}(k)\right]^t e^{ikx}\mathrm{d}k. \tag{44}$$

This solution generally depends on the details of the structure 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 solution converges to the Gaussian profile

$$p(x;t) = \frac{1}{(2\pi Dt)^{1/2}}e^{-\frac{(x-vt)^2}{4Dt}} \tag{45}$$

with a variance growing linearly with time.

Ex.III.4 : Take a RW on a discrete one-dimensional line. Assuming that the walker has, at each step, a probability 1/2 to go to the left and 1/2 to go to the right. Explore by numerical simulations how the probability distribution evolves over time, and verify the accuracy of Eq.(45). Provide a simple metric to test the “Gaussianity” of the distribution.

D. 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, i.e., 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.

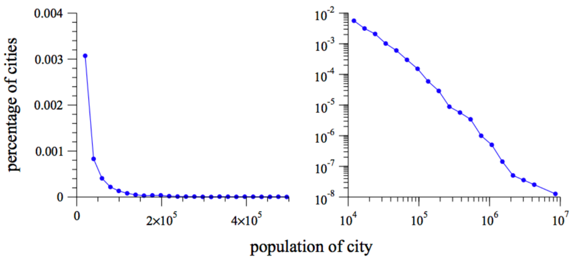

FIG. 5. Left: histogram of the populations of all US cities with population of 10 000 or more. Right: another histogram of the same data, but plotted on logarithmic scales. The approximate straight-line form of the histogram in the right panel implies that the distribution follows a power law. Data from the 2000 US Census.. Taken from Adamic, Lada A. ”Zipf, power-laws, and pareto-a ranking tutorial.” Xerox Palo Alto Research Center, Palo Alto, CA, http://ginger. hpl. hp. com/shl/papers/ranking/ranking. html (2000).

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

$$p(x) = Cx^{-\alpha} \quad (x \geq x_{\min}), \tag{46}$$

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

$$\int_{x_{\min}}^{\infty} Cx^{-\alpha}\mathrm{d}x = 1. \tag{47}$$

Other power-law distributions are by definition asymptotically (i.e., for large x) the same as Eq. (46) 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 tail, i.e., 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. 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 without a power-law tail. Examples include stretched exponential distributions and log-normal distributions.

The moments of power-law distributions are given by

$$\langle x^\beta \rangle = \int_{x_{\min}}^{\infty} x^\beta p(x)\mathrm{d}x = \frac{\alpha-1}{\alpha-1-\beta} x_{\min}^\beta \quad (\beta < \alpha - 1). \tag{48}$$

The moments for $\beta \geq \alpha - 1$ are divergent. In particular, the mean $\langle x \rangle$ does not exist for $1 < \alpha \leq 2$, and the variance does not exist for $2 < \alpha \leq 3$. These features impact various structural and dynamical properties of complex systems including networks, as we will see throughout these notes. When $\alpha \leq 1$, the distribution is ill-defined because $\int_{x_{\min}}^{\infty} p(x)\mathrm{d}x$ is divergent such that $p(x)$ cannot be normalised. When a moment, $\langle x^\beta \rangle$, diverges, its empirical measurement diverges as the number of samples increases and $\langle x^\beta \rangle$ with β only slightly smaller than $\alpha - 1$ converges very slowly. Both the divergence and slow convergence of moments are due to the appearance of extreme values. For example, the sample mean for the power-law distribution with $\alpha = 2$ diverges as we accumulate samples.

In a majority of empirical data, the distribution can be close to Eq. (46) only in a certain range of the variable. However, key observations such as the divergent moments hold true as long as a distribution behaves the same as Eq. (46) when $x \to \infty$ up to a normalisation constant. For example, the Cauchy distribution given by $p(x) = 1/\left[\pi(1+x^2)\right]$ is qualitatively the same as Eq. (46) with $\alpha = 2$ as $x \to \infty$. It should also be noted that the tail of an empirical distribution ceases to be a power-law beyond a certain scale because of the finiteness of the system. The finite size effect typically leads to exponential cutoffs. Therefore, the power-law regime, if present, usually dominates for values that are neither too small nor too large.

The heterogeneity of power-law distributions is often associated with the presence of inequalities in the system. What fraction w of the total wealth is held by a certain fraction of the richest people when the wealth distribution is given by Eq. (46)? To answer this question, let us first calculate the fraction of the people whose wealth is at least x_0:

$$p(x \geq x_0) = \int_{x_0}^{\infty} C x^{-\alpha} \mathrm{d}x = \left(\frac{x_0}{x_{\min}}\right)^{-\alpha+1}. \tag{49}$$

The fraction of wealth held by these richest people is given by

$$w(x_0) = \frac{\int_{x_0}^{\infty} x \cdot C x^{-\alpha}\mathrm{d}x}{\int_{x_{\min}}^{\infty} x \cdot C x^{-\alpha}\mathrm{d}x} = \left(\frac{x_0}{x_{\min}}\right)^{-\alpha+2} = \left[p(x \geq x_0)\right]^{\frac{\alpha-2}{\alpha-1}}, \tag{50}$$

where we have assumed that $\alpha > 2$ so that the average wealth is finite. Equation (50) neither depends on x_0 nor $x_{\min}$ explicitly, and it provides a direct relation between $w(x_0)$ and $p(x \geq x_0)$. This relation is often called the "80-20 rule", anecdotally meaning that 80% of the wealth is in the hands of the richest 20%. More precisely, setting $p(x \geq x_0) = 0.2$, $w(x_0) = 0.2^{(\alpha-2)/(\alpha-1)}$ can take any value between 0.2 and 1 depending on the value of α. In the limit $\alpha \to \infty$, the system does not exhibit a power-law tail, and we obtain $w(x_0) = 0.2$. In this case, the system is egalitarian. As α decreases, the tail of the distribution becomes fat and inequality grows. In the extreme situation with $\alpha \to 2$, the total wealth belongs to an infinitesimally small fraction of the richest people. In the econometrics literature, the measurement of this effect in empirical data can be done with the Gini coefficient.

Other properties of power-law distributions include the following:

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

  $$p(c_1 x) = c_2 p(x) \tag{51}$$

  for large x, where c_1 and c_2 are constants. Equation (51) 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 Eq. (46) is equivalent to

  $$\log p(x) = \log C - \alpha x. \tag{52}$$

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

As a side note, Tauberian and Abelian types of theorems help us understand power-law tails of probability distributions and functions in general. They are particularly useful for analytically understanding the long-term behaviour of stochastic dynamics when power-law statistics come into play. In short, Tauberian and Abelian theorems are the inverse of each other. The Tauberian theorem for the Laplace transform, i.e. related to the Fourier transform, is stated as follows :

Consider a function $f(t)$ whose asymptotic behaviour is given by $f(t) \approx t^{\rho-1}$ $(\rho > 0)$ for large t. The Laplace transform of $f(t)$ near $s = 0$ is given by $\hat{f}(s) \approx \Gamma(\rho) s^{-\rho}$, where $\Gamma(\rho)$ is the gamma function.

Ex.III.4 : Take an electronic version of a large book (e.g. the Bible), 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. Plot the Zipf plot of the data, that it is the relation between the rank and the number of occurrences of the words. Any connection between the Zipf plot and the distribution?

E. Maximum likelihood

The previous sections provide mechanisms by which certain families of distributions emerge. 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 different approaches to parameter fitting. The most popular one is probably the maximum likelihood method.

Consider a sequence of observations $\{x_i\}$ $(i = 1, 2, \ldots)$. We are trying to fit a certain model whose parameter set is denoted by θ and is assumed to have a finite support for simplicity. 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

$$p(\theta|\{x_i\}) = p(\{x_i\}|\theta)\frac{p(\theta)}{p(\{x_i\})}. \tag{53}$$

By definition, the probability of observing certain data, $p(\{x_i\})$, is fixed, and it does affect the optimisation of θ. Moreover, in the absence of other information, it is convenient to assume that any values of θ are equally likely such that the prior distribution $p(\theta)$ is a constant. Then, $p(\theta|\{x_i\})$ and $p(\{x_i\}|\theta)$ are proportional to each other, and the locations of their maximum coincide. Therefore, it suffices to maximise $p(\theta|\{x_i\})$ in terms of θ.

As an example, consider the model in which each x_i independently obeys the same exponential distribution. The likelihood of the data is given by

$$\mathcal{L}(\{x_i\}|\lambda) = \prod_{i=1}^{n} p(x_i|\lambda), \tag{54}$$

where $p(x|\lambda) = \lambda e^{-\lambda x}$ and n is the number of observations. To find the value of λ that maximises the likelihood, we conventionally maximise the logarithm of $\mathcal{L}$. The maximum of

$$\log \mathcal{L}(\{x_i\}|\lambda) = \log \prod_{i=1}^{n} p(x_i|\lambda) = n\log\lambda - \lambda\sum_{i=1}^{n} x_i \tag{55}$$

is obtained via

$$\frac{\partial}{\partial\lambda}\log\mathcal{L}(\{x_i\}|\lambda) = 0, \tag{56}$$

which leads to

$$\hat{\lambda} = \frac{1}{\frac{1}{n}\sum_{i=1}^{n} x_i} = \frac{1}{\langle x\rangle}. \tag{57}$$

The maximum likelihood estimation is easy if the log likelihood takes an analytical form and its maximum is explicitly computed. Otherwise, we resort to numerical methods such as the expectation-maximisation algorithm.

There are also other situations in which likelihood maximisation needs to be done carefully. For example, suppose that we are fitting the power-law distribution, Eq. (46), to data. Usually, a power-law distribution provides a good fit of empirical data in a regime excluding small x values. Therefore, we regard $x_{\min}$ as the point where the power-law regime starts, which we are interested in estimating in addition to the power-law exponent α. The log likelihood of the data under the power-law distribution is given by

$$\log\mathcal{L}(\{x_i\}|\alpha, x_{\min}) = n\log\left(\frac{\alpha-1}{x_{\min}}\right) - \alpha\sum_{i=1}^{n}\log\left(\frac{x_i}{x_{\min}}\right). \tag{58}$$

Setting $\partial\log\mathcal{L}/\partial\alpha = 0$ yields the maximum likelihood estimator given by

$$\hat{\alpha} = 1 + \frac{n}{\sum_{i=1}^{n}\log\left(\frac{x_i}{x_{\min}}\right)}. \tag{59}$$

However, finding the optimal $x_{\min}$ value is not a straightforward exercise because changing values of $x_{\min}$ also changes the number of observations, n, falling within the assumed power-law regime, i.e., $x \geq x_{\min}$. The likelihood monotonically decreases with increasing n because the probability of observing an additional data point is always smaller than unity. Therefore, the maximum likelihood in terms of $x_{\min}$ is obtained by a trivial solution $\hat{x}_{\min} = \max_i x_i$, yielding $n = 0$. Other techniques must be used to estimate $\hat{x}_{\min}$. The minimisation of goodness-of-fit statistics, such as the Kolmogorov-Smirnov test, measuring the distance between the cumulative distribution of the empirical data and that of the model, is one such possibility.

Ex.III.5 : Write a code that takes as an input a sequence of real value numbers and returns the best exponential distribution for the intervals. Verify the accuracy of the prediction (including its absence of bias).

F. 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, entropy is defined as

$$H(X) = -\sum_x p(x) \log p(x). \tag{60}$$

If X can take one of n states, we obtain $0 \le H(X) \le \log n$. The maximum value $H(X) = \log n$ is realised when $p(x)$ is the uniform density, i.e., when $p(x) = 1/n$ for all x. The minimum value $H(X) = 0$ is realised when X is deterministic, i.e., $p(x) = \delta_{x,x_0}$ for a specific x_0, where δ is Kronecker delta. In the latter case, we know the value of X before observing it, hence the lack of uncertainty.

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

$$H(X,Y) = -\sum_x \sum_y p(x,y) \log p(x,y). \tag{61}$$

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

$$\begin{aligned} H(Y|X) &= -\sum_x \sum_y p(x,y) \log p(y|x) \\ &= -\sum_x p(x) H(Y|X=x), \end{aligned} \tag{62}$$

and refers to the entropy of Y conditioned on the value of X and averaged over all possible values of X. The joint entropy and conditional entropy are related by the chain rule:

$$H(X,Y) = H(X) + H(Y|X). \tag{63}$$

Equation (63) 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:

$$I(X,Y) = H(X) - H(X|Y) = H(X) + H(Y) - H(X,Y). \tag{64}$$

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

$$I(X,Y) = \sum_x \sum_y p(x,y) \log \frac{p(y,x)}{p(x)p(y)}. \tag{65}$$

Equations (64) and (65) show that mutual information is symmetric, i.e., $I(X,Y) = I(Y,X)$. 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 (see Figure 6).

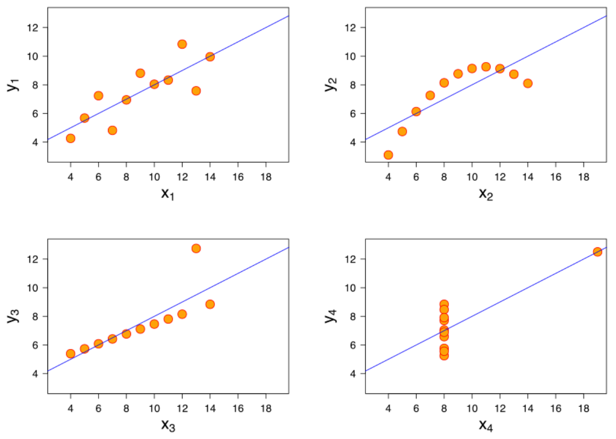

FIG. 6. Anscombe's quartet: all four sets are identical when examined using simple summary statistics, including their Pearson coefficient, but vary considerably when graphed

There exist many situations when we have to compare two networks defined on the same set of nodes, for instance in the case of temporal networks. Mutual information can serve to this end by specifying a distribution $p(x)$ that summarises a network. Other commonly used similarity measures include the Pearson correlation coefficient and the Jaccard index. The Pearson correlation for random variables is given by Eq. (11) and adapted for a list of pairwise observations $\{(x_i, y_i); 1 \le i \le n\}$ as follows:

$$\frac{\sum_{i=1}^n (x_i - \langle x \rangle)(y_i - \langle y \rangle)}{\sqrt{\sum_{i=1}^n (x_i - \langle x \rangle)^2 \sum_{i=1}^n (y_i - \langle y \rangle)^2}}, \tag{66}$$

where $\langle x \rangle = \sum_{i=1}^n x_i/n$ and $\langle y \rangle = \sum_{i=1}^n y_i/n$. The Jaccard index for two sets S_1 and S_2 is defined by

$$\frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}, \tag{67}$$

where $|\cdot|$ denotes the number of elements in the set. The Jaccard index takes the largest value, 1, when $S_1 = S_2$. It takes the smallest value, 0, when S_1 and S_2 do not have any common element.

Ex.III.6 : After reading Information *Theory and Statistical Mechanics*, by ET Jaynes, calculate the maximum-entropy prediction of the following system:
A traditional die (with 6-faces) is biased. The average value is 5, instead of $3,5$. Calculate the probability to observe a value 6.

G. Matrix algebra

Matrices are a standard representation of networks. Properties of matrices are crucial in order to describe linear dynamical systems and at the core of several algorithms to extract structural information from networks. In this section, we provide a short, practical summary of results from linear algebra, emphasising what will be used in later chapters.

Consider an $N \times N$ matrix A. A vector and scalar value λ satisfying

$$Au = \lambda u \tag{68}$$

are called the eigenvector and eigenvalue, respectively. There are at most N eigenvalues and associated eigenvectors. If A is a symmetric matrix, i.e., $A_{ij} = A_{ji}$ $(1 \le i, j \le N)$, all the eigenvalues λ_i $(1 \le i \le N)$ are real. In addition, the eigenvectors u_i associated with different eigenvalues λ_i are orthogonal, i.e., $\langle u_i, u_j \rangle = 0$ if $i \neq j$, where $\langle , \rangle$ is the inner product. Matrix A may have duplicated eigenvalues. Even in this case, we can select the set of N eigenvectors such that the orthogonality is respected.

Matrix A is decomposed as

$$A = \sum_{\ell=1}^{N} \lambda_\ell u_\ell u_\ell^\top, \tag{69}$$

where $\top$ represents the transposition. The validity of Eq. (69) is verified by multiplying an arbitrary eigenvector u_i to both sides of Eq. (69). Due to the orthogonality of the eigenvector, we obtain $Au_i = \lambda_i u_i$, assuming that the eigenvectors are properly normalised such that

$$\langle u_\ell, u_{\ell'} \rangle = \delta_{\ell\ell'}. \tag{70}$$

By combining Eqs. (69) and (70), we obtain

$$A^n = \sum_{\ell=1}^{N} \lambda_\ell^n u_\ell u_\ell^\top. \tag{71}$$

We are often interested in the extremal eigenvalue such as the largest eigenvalue of a symmetric matrix A, i.e., $\lambda_{\max}$. The Perron-Frobenius theorem guarantees that when all elements of A are strictly positive, $\lambda_{\max}$ is the isolated (i.e., not duplicated) largest eigenvalue. In addition, all elements of the corresponding eigenvector $u_{\max}$, called the Perron-Frobenius vector, have the same sign. Any other eigenvector u_ℓ does not show this property because, due to the orthogonality $\langle u_\ell, u_{\max} \rangle = 0$, some of the elements in u_ℓ must have the opposite signs. The Perron-Frobenius theorem also holds true for asymmetric matrices. In the asymmetric case, the statement that the largest eigenvalue is isolated is replaced by that of the modulus, or the absolute value of the eigenvalue. Matrices appearing in network analysis are often sparse, with a majority of elements being zero. The Perron-Frobenius theorem is also applicable in this situation if matrix A is primitive, i.e., if all elements of A are non-negative and all elements of A^n are positive for some integer $n > 0$. If an undirected network of interest is connected as a single component, which is usually the case in theoretical studies, matrices representing the network are usually primitive (with the exception of so-called bipartite graphs), such that the Perron-Frobenius theorem can be used.

The power method is a computationally efficient method to calculate $\lambda_{\max}$ and $u_{\max}$ of a given matrix. To do this, we start with an (almost) arbitrary initial vector x and repeat multiplying A. By multiplying x to both sides of Eq. (69), we obtain

$$x(1) \equiv Ax = \sum_{\ell=1}^{N} \lambda_\ell u_\ell \langle u_\ell^\top, x \rangle \tag{72}$$

By repeating the multiplication of A on both sides of Eq. (72), we obtain

$$x(n) \equiv A^n x = A^n x(n-1) = \sum_{\ell=1}^{N} \lambda_\ell^n u_\ell \langle u_\ell^\top, x \rangle. \tag{73}$$

If $\lambda_{\max}$ is the isolated eigenvalue, as in the case of the primitive matrix, $\lambda_{\max}^n \gg \lambda_\ell^n$ for any other eigenvalue λ_ℓ for large n. Then, in Eq. (73), all but the one term corresponding to $\lambda_{\max}$ is negligible on the right-hand side as $n \to \infty$. After many iterations, we can obtain the largest eigenvalue $\lambda_{\max}$ by looking at how much each element of $x(n)$ grows by one iterate and the corresponding eigenvector $u_{\max}$ from $x(n)$. In practice, we normalise $x(n)$ in each iterate to avoid the elements of $x(n)$ to become very large or small.

Ex.III.7 : Implement the power-method and test it on some examples.
Ex.III.8 : Calculate numerically the distribution of eigenvalues of a random symmetric matrix A of size 1000, where each entry is an independent Bernouilli random variable, subject to the constraint that $A_{ij} = A_{ji}$ to ensure symmetry.

H. Markov chains

Markov chains are stochastic dynamics on N states in discrete time. A state may be the position in a network having N nodes 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 $(t = 0, 1, \ldots)$, 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 (i.e., 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. In other words, a discrete-time stochastic process verifying

$$p(X_{t+1} = i_{t+1}|X_t = i_t, \ldots, X_1 = i_1, X_0 = i_0) \\ = p(X_{t+1} = i_{t+1}|X_t = i_t), \quad (74)$$

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:

$$p(X_{t+1} = j|X_t = i) \equiv T_{ij}. \quad (75)$$

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

$$\sum_{j=1}^{N} T_{ij} = 1. \quad (76)$$

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 $p_i(t)$, evolves according to

$$p_j(t+1) = \sum_{i=1}^{N} p_i(t) T_{ij} \quad (1 \le j \le N). \quad (77)$$

It should be noted that $\sum_{i=1}^{N} p_i(t) = 1$ for any t, if the initial condition is properly normalised. Equation (77) is compactly rewritten as

$$p(t+1) = p(t)T, \quad (78)$$

where $p(t) = (p_1(t) \ \cdots \ p_N(t))$. Equation (78) yields

$$p(t) = p(0)T^t. \quad (79)$$

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 stationary density by $p^* = (p_1^*, \ldots, p_N^*)$, where $p_i^* = \lim_{t\to\infty} p_i(t)$ $(1 \le i \le N)$ and hence $\sum_{i=1}^{N} p_i^* = 1$. Substitution of $p_i(t) = p_i(t+1) = p_i^*$ $(1 \le i \le N)$, which holds true in the limit $t \to \infty$, in Eq. (77) yields

$$p^* = p^* T. \quad (80)$$

Therefore, the stationary density is the left eigenvector of T with eigenvalue unity. Because

$$T \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}, \quad (81)$$

which is a consequence of Eq. (76), T is guaranteed to have an eigenvalue of unity. If the entire set of the N states is ergodic, one can go from i to j for any i and j. In this case, p^* is unique, and iterates of Eq. (79) starting from an almost arbitrary initial condition converge p^* except in special cases.

Then, the eigenvalue of unity is in fact the largest eigenvalue of T in terms of the modulus (i.e., absolute value). Therefore, p^* is the Perron-Frobenius vector. This observation is consistent with the fact that all elements of the Perron-Frobenius vector are positive (Section III G). In addition, Eq. (71) adapted to the case of asymmetric matrices dictates that the discrepancy of $p(t)$ from p^* decays exponentially as $\propto |\lambda_{\text{2nd}}|^t$, where λ_{2nd} 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_{2nd} and $\lambda_{\max}$, with the latter being equal to unity in the current case. Therefore, we often call $1 - \lambda_{\text{2nd}}$ 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?

Ex.III.9 : Construct the 3×3 transition matrix of a Markov chain (or your choice) and describe its properties.

I. Branching processes

A branching process is a Markov process in which each individual produces some (possibly zero) individuals and then dies, each of the new individuals undergoes reproduction, and so forth (Fig. 7). In network theory, branching processes are a useful tool for understanding network generation and epidemic processes on networks. In network generation, we start from a given node and explore its neighbours, neighbours of neighbours, and so on to expand the network under investigation. In epidemic processes, an initially infected node typically propagates infection to a certain number of neighbouring nodes, each of which then infects some others, and so on. In both cases, the number of nodes that a node newly recruits usually depends on the node and hence can be considered as a random number, as assumed in branching processes.

The Galton-Watson process is a prototypical branching process model defined as follows. Fix the distribution of

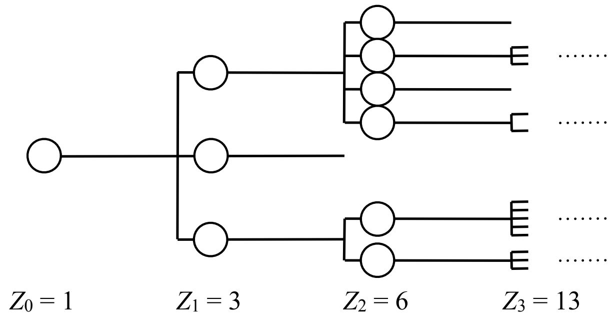

FIG. 7. Schematic of the Galton-Watson branching process.

the number of offspring, $\{p(n)\}$, where $p(n)$ is the probability that an individual reproduces n individuals. The number of individuals in the tth generation is denoted by Z_t (Fig. 7). First, there is initially one individual, i.e., $Z_0 = 1$. Second, this individual generates offspring whose number Z_1 is drawn from $\{p(n)\}$. Third, each of the Z_1 individuals in the first generation produces offspring whose number independently obeys distribution $\{p(n)\}$. The individuals born in this stage, which total Z_2, define the second generation. We repeat this procedure to define further generations until the process gets extinguished. The extinction may not occur, in which case the number of individuals grows indefinitely.

The extinction requires $p(0) > 0$. In other words, an individual does not produce any offspring with a positive probability. If $p(n)$ for large n values is large, the population would grow rather than shrink. In fact, the mean number of offspring, i.e., $\langle n \rangle \equiv \sum_{n=0}^{\infty} np(n)$ is the main determinant of a branching process. If $\langle n \rangle \leq 1$, a realisation of the process will always die out for sufficiently large t, except in the deterministic case $n = \langle n \rangle = 1$ such that each individual always yields exactly one offspring. In particular, $E[Z_t] = \langle n \rangle^t \to 0$ as $t \to \infty$. If $\langle n \rangle > 1$, $E[Z_t]$ exponentially grows and individual realisations of the process may exponentially grow as well.

We denote by q the probability that the process starting from one individual eventually dies out and, as we now show, $q = 1$ when $\langle n \rangle \leq 1$. If an individual produces n individuals, then the process will die out with probability q^n because of the independence of the sub-processes starting from n individuals. Therefore, we obtain the recursive relationship

$$q = \sum_{n=0}^{\infty} p(n) q^n. \tag{82}$$

Equation (82) always has $q = 1$ as a solution. It has a solution with $q < 1$ if and only if $\langle n \rangle > 1$. To show this, we use the fact that the solution is the intersection of

$$y = f_1(q) \equiv q$$

and

$$y = f_2(q) \equiv \sum_{n=0}^{\infty} p(n) q^n$$

. Because $\langle n \rangle > 1$, it suffices to consider the case $p(0) + p(1) < 1$. If $p(0) = 0$, $q = 0$ is a solution because

$$f_2(0) = p(0) = 0 = f_1(0).$$

If $p(0) > 0$, we obtain $0 < f_2(0) < 1$. Because $f_1(1) = f_2(1) = 1$, and

$$\mathrm{d}f_2(q)/\mathrm{d}q = \sum_{n=1}^{\infty} np(n) q^{n-1} > 0$$

and

$$\mathrm{d}^2 f_2(q)/\mathrm{d}q^2 = \sum_{n=2}^{\infty} n(n-1)p(n) q^{n-2} > 0$$

when $0 < q \leq 1$, $y = f_1(q)$ and $y = f_2(q)$ cross in $0 < q \leq 1$ if and only if $\mathrm{d}f_2(q)/\mathrm{d}q > 1$ at $q = 1$. This condition is equivalent to $\langle n \rangle > 1$. In this case, the process grows exponentially with probability $1 - q$.

Ex.III.10 : Consider a Galton-Watson process with $p(0) = a$, $p(2) = 1 - a$ and $p(i) = 0$ otherwise. Estimate numerically the evolution of $E[Z_t]$ as a function of a.

Application : Branching processes are often used to model cascades in social media. For instance, cascades of retweets on Twitter can be represented by trees and modelled by a Galton-Watson, or variations around it. From a practical point of view, the initial structure of a cascade can be fed into a machine learning framework to predict their future success, or the cascade can be used to calibrate the parameters of a branching process for such a prediction. See for instance:
Cheng, Justin, et al. "Can cascades be predicted?." Proceedings of the 23rd international conference on World wide web. ACM, 2014.
Kobayashi, Ryota, and Renaud Lambiotte. "TiDeH: Time-Dependent Hawkes Process for Predicting Retweet Dynamics." ICWSM. 2016.

IV. BASIC STRUCTURAL PROPERTIES OF NETWORKS

In this chapter, we give an introduction on relatively basic methods for network science.

A. Definition

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

$$\mathcal{G} = (V, E), \tag{83}$$

where V is a set of nodes (also called vertices) and E is a set of links (also called edges). The number of nodes and that of links are denoted by N and M throughout this book. Each link is defined by a pair of nodes, 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 nodes are reciprocally connected. In the case of weighted networks, links are also assigned with a weight function, characterising the importance or weight of the link. An undirected and unweighted network with $N = 5$ nodes and $M = 6$ links is shown in Fig. 8.

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. We introduce two major representations.

A network can be represented by the corresponding $N \times N$ adjacency matrix. Being adjacent means that two nodes are directly connected by a link. In the case of unweighted networks, the entries of the adjacency matrix are given by

$$A_{ij} = \begin{cases} 1 & \text{if node } v_i \text{ is adjacent to node } v_j, \\ 0 & \text{otherwise.} \end{cases} \tag{84}$$

If the network is weighted, A_{ij} can take positive values different from one, representing the weight of the link. In general, undirected and directed networks will yield symmetric and asymmetric adjacency matrices, respectively. The adjacency matrix of the network illustrated in Fig. 8 is given by

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \end{pmatrix}. \tag{85}$$

An adjacency matrix representation is useful for formulating and theoretically analysing structure of, and

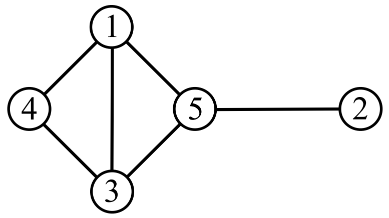

FIG. 8. An undirected and unweighted network.

dynamical processes on networks. In particular, it is amenable to tools from linear algebra such as the analysis of eigenvectors and eigenvalues. A drawback of this representation is its memory cost because a network possessing N nodes requires $O(N^2)$ elements for storage. A majority of networks found in the real world and generated from models are sparse such that most elements of the adjacency matrix are equal to zero. Therefore, it is often preferable to use the data structure called sparse matrix. Its advantage is a significant gain in memory and faster computations because operations involving zeros are not executed.

In fact, a matrix formulation is often unsuitable even if sparse matrix representations are employed. This is the case when, for example, the computation of shortest paths is involved or all links have to be scanned repeatedly. A representation of static networks alternative to the adjacency matrix is called the link list. In this link-centric approach, a graph is described as a list of pairs of nodes, each corresponding to a link in the network, as follows:

$$\{(u_1, v_1), (u_2, v_2), \ldots, (u_M, v_M)\}. \tag{86}$$

When the network is directed, we interpret Eq. (86) as representing directed links from u_i to v_i ($1 \le i \le M$). The link list of the network shown in Fig. 8 is given by

$$\{(v_1, v_3), (v_1, v_4), (v_1, v_5), (v_2, v_5), (v_3, v_4), (v_3, v_5)\}. \tag{87}$$

Link lists have the additional advantage of being efficiently used for link randomisation and numerical simulations of dynamics on sparse networks.

Ex.IV.1 : There exist several repositories of network data, of different type (including unweighted, weighted, directed, temporal, etc.). Explore Konect (http://konect.uni-koblenz.de/) or ICON (https://icon.colorado.edu/), download and import the graphs by using NetworkX.

B. Degree distribution

The degree is defined as the number of links incident to a node. We denote the degree of the ith node by k_i.

For undirected networks, the degree is given by

$$k_i = \sum_{j=1}^{N} A_{ij} \left(= \sum_{j=1}^{N} A_{ji} \right). \qquad (88)$$

A network is called regular if all nodes have the same degree, i.e., $k_i = k_j$ for all i and j.

For directed networks, we distinguish the in-degree, i.e., the number of links incoming to the node, and the out-degree, i.e., the number of links outgoing from the node. They are given by $k_i^{\text{in}} = \sum_{j=1}^{N} A_{ji}$ and $k_i^{\text{out}} = \sum_{j=1}^{N} A_{ij}$, respectively. These numbers are basic measures of the centrality, or importance, of a node in a network. See section (IV E) for more details.

Each link has two endpoints and hence contributes to the degree of two nodes by one each. Therefore, we obtain

$$\sum_{i=1}^{N} k_i = \sum_{i=1}^{N}\sum_{j=1}^{N} A_{ij} = 2M \qquad (89)$$

for undirected networks. Equation (89) is called the handshaking lemma. It implies that the sum of the degrees of all the nodes in any undirected network is an even number. For directed networks, the handshaking lemma is given by $\sum_{i=1}^{N} k_i^{\text{in}} = \sum_{i=1}^{N} k_i^{\text{out}} = M$.

The degree distribution of a network is the frequency distribution of the degree and denoted by $p(k)$. A majority of networks in different domains possesses long-tailed degree distributions. In many situations, their tail is described by a power-law, i.e.,

$$p(k) \propto k^{-\gamma}, \qquad (90)$$

where γ is typically between two and three. Because the maximum degree is equal to $N - 1$, Eq. (90) approximately holds true up to a certain cutoff degree, above which $p(k)$ rapidly decays to zero. The average degree, denoted by $\langle k \rangle$, is given by

$$\langle k \rangle = \sum_{k} k p(k). \qquad (91)$$

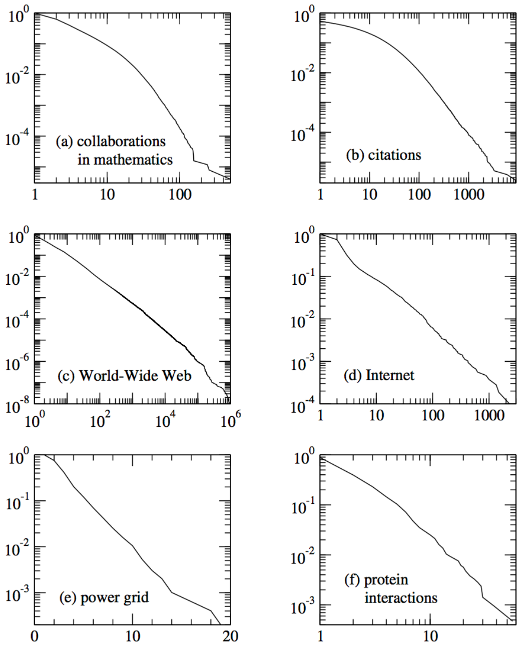

FIG. 9. Cumulative degree distributions for six different networks. The horizontal axis for each panel is vertex degree k (or in-degree for the citation and Web networks, which are directed) and the vertical axis is the cumulative probability distribution of degrees, i.e., the fraction of vertices that have degree greater than or equal to k. 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 Newman, Mark EJ. "The structure and function of complex networks." SIAM review 45.2 (2003): 167-256.

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 node has the same degree. The paradox originates from the fact that nodes with a large degree contribute disproportionately to the average degree of a friend, as they have a higher probability of being friends than low degree nodes do. Consider the situation shown in Fig. 10, where we pretend not to know the actual connection between nodes. This is in fact the definition of the configuration network model (Section V B). The network has $N = 6$ nodes and the average degree of a randomly selected individual is equal to

$$\frac{1}{6}(1+2+3+1+4+1) \equiv \langle k \rangle = 2. \qquad (92)$$

To calculate the average number of friends of a friend, we have instead to perform a weighted average, accounting for the fact that a node with degree k will appear k times in the calculation of the average. The weighted average degree is equal to

$$\frac{(1\times 1 + 2\times 2 + 3\times 3 + 1\times 1 + 4\times 4 + 1\times 1)}{(1+2+3+1+4+1)} \approx 2.67, \qquad (93)$$

where we have allowed self-loops and multiple edges for simplicity. In general, for sufficiently large and random networks, the mean degree of a neighbour is given by

$$\sum_k k \times \frac{k p(k)}{\sum_{k'} k' p(k')} = \frac{\langle k^2 \rangle}{\langle k \rangle}. \qquad (94)$$

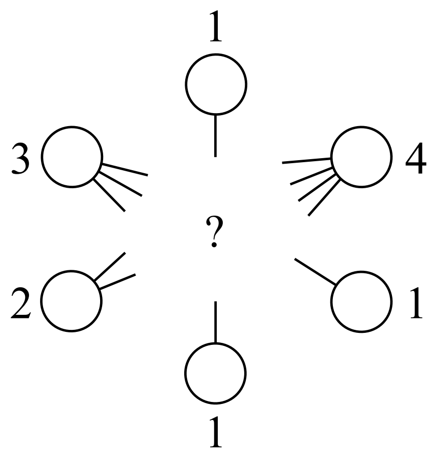

FIG. 10. Friendship paradox. A network with $N = 6$ nodes is shown. The numbers represent the nodes' degrees. The expected degree of a neighbour of a randomly selected node is equal to ≈ 2.67, which is larger than the mean degree of the network, $\langle k \rangle = 2$.

Ex.IV.2 : Plot the degree distribution of the networks imported in the previous exercice.

C. Measures derived from walks and paths

A walk is defined as a succession of adjacent nodes such that one can travel from the start node to the end node by traversing links. A path is a walk where each node is visited only once (with the possible exception that the walk may end at the node where it begins). Walks are used for constructing dynamical processes on networks (e.g., random walks) and measurements such as the Katz centrality, where all possible walks from one node to another are exhaustively counted. Paths are particularly useful when considering the shortest travelling route from one node 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 distinguish walks and paths.

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 nodes. In general, the number of walks of length ℓ is given by the elements of A^ℓ.

To identify paths from a node to another requires more effort. The distance between nodes v_i and v_j, denoted by $d(v_i, v_j)$, is defined as the smallest number of hops in a path necessary to go from v_i to v_j. For undirected networks, the distance defined in this way satisfies the axioms that a distance measure should satisfy: non-negativity (i.e., $d(v_i, v_j) \geq 0$), coincidence (i.e., $d(v_i, v_j) = 0$ if and only if $v_i = v_j$), symmetry (i.e., $d(v_i, v_j) = d(v_j, v_i)$) and triangle inequality (i.e., $d(v_i, v_j) \leq d(v_i, v_\ell) + d(v_\ell, v_j)$). 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, $d(v_i, v_j)$ can be calculated by Dijkstra's algorithm. For a fixed v_i, first initialise the distance from v_i by setting $d(v_i, v_i) = 0$ and $d(v_i, v_j) = \infty (j \neq i)$. Second, set $d(v_i, v_j) = 1$, 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 $d(v_i, v_\ell) = 2$. When all neighbours of v_j are exhausted, we declare that v_j has been visited. Fifth, we select an unvisited node with the smallest tentative distance value and inspect its all neighbours. We repeat the same procedure to determine the distance from v_i to every other node.

For undirected networks, the average distance for a network is defined by

$$L = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1}^{i-1} d(v_i, v_j). \tag{95}$$

In many real networks, L is remarkably small as compared to the number of nodes, N. For example, a Facebook network composed of $N \approx 7.2 \times 10^8$ active users with 6.9×10^{10} friendship links yielded $L \approx 4.7$. The diameter is defined by

$$D = \max_{u,v \in V} d(u, v). \tag{96}$$

In undirected networks, two nodes are said to be connected if there exists a path between them. Connectedness is an equivalence relation because it is reflexive (i.e., v is connected to itself), symmetric (i.e., if u is connected to v, v is connected to u) and transitive (i.e., if u is connected to v and v is connected to w, u is connected to w). Intuitively, a connected component is an island within which one can travel from any node to any other along a path. There is no path between nodes 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 network are never infected, independently of the model of epidemic dynamics and its parameters.

In directed networks, symmetry is not satisfied because the existence of a path from u to v does not guarantee the existence of a path from v to u. Therefore, the concept of connectedness is more complex, and the notions of strong and weak connectedness are distinguished. Nodes u and v are said to be strongly connected if there exist a path from u to v and a path from v to u. Two nodes 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

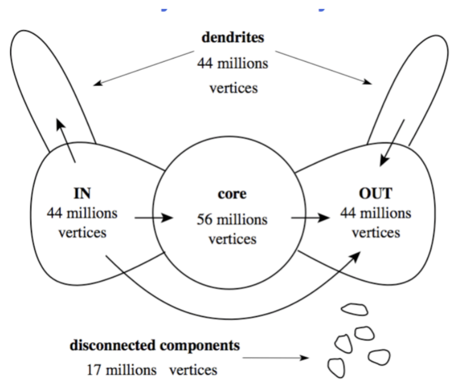

FIG. 11. When networks are directed, the notion of connected is much more rich than in the case of undirected networks. Illustration of the connectivity patterns of a sub-part of the Web.

set of nodes in which each pair of nodes is strongly connected. Strong connectedness implies weak connectedness but not vice versa.

Ex.IV.3 : Use the implementation of Dijkstra in NetworkX to search shortest paths in a network.

D. Clustering coefficient

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

$$C_i \equiv \frac{\text{number of triangles including the } i\text{th node}}{k_i(k_i-1)/2}, \quad (97)$$

which measures the abundance of triangles in the neighbourhood of the ith node v_i. The denominator gives the normalisation such that $0 \le C_i \le 1$. If any pair of the neighbours of v_i is adjacent to each other to form a triangle, $C_i = 1$. If no pair of neighbours of v_i is adjacent to each other such that the neighbourhood of v_i is star-like, $C_i = 0$. The clustering coefficient, denoted by C, is defined as the average of C_i over the network, i.e.,

$$C \equiv \frac{1}{N}\sum_{i=1}^{N} C_i. \quad (98)$$

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

(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 [415, 411] 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 [323, 288] in which shortcuts are added randomly between vertices, but no edges are removed from the underlying one-dimensional lattice

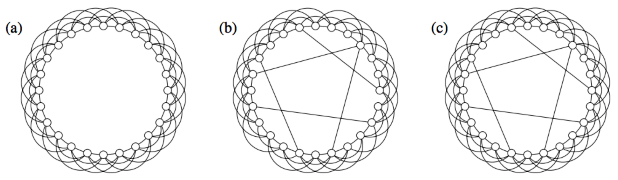

FIG. 12. (a) A one-dimensional lattice with connections between all vertex pairs separated by k or fewer lattice spacing, with k = 3 in this case. (b) The small-world model of Watts and Strogats is created by choosing at random a fraction p of the edges in the graph and moving one end of each to a new 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 Newman, Mark EJ. "The structure and function of complex networks." SIAM review 45.2 (2003): 167-256.

Ex.IV.4 : Generate graphs of the model c from Figure (13) and calculate the dependence of the network diameter and clustering coefficient on the number of shortcuts.

Application: A high density of triangles is often associated to the mechanism of triadic closure, that is the tendency for wedges (paths of length 2) 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 link prediction. Say that you have a snapshop 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 nodes that are not (yet) connected, but belong to several wedges. One such metric for ranking potential edges is the so-called Adamic-Adar. See for instance: Liben-Nowell, David, and Jon Kleinberg. "The link prediction problem for social networks." journal of the Association for Information Science and Technology 58.7 (2007): 1019-1031.

E. Centrality

Centrality measures aim to quantify the importance of nodes 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.

1. Closeness centrality

The closeness centrality and betweenness centrality are popular centrality measures based on the distance between pairs of nodes. The closeness centrality for node v_i is defined by

$$\text{closeness}_i = \frac{N-1}{\sum_{j=1;j\neq i}^{N} d(v_i, v_j)}, \tag{99}$$

which is the inverse of the mean distance from node v_i to any other node. The closeness centrality is well-defined only for connected networks.

2. Betweenness centrality

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

$$\text{betweenness}_i = \frac{2}{(N-1)(N-2)} \sum_{j=1;j\neq i}^{N} \sum_{\ell=1;\ell\neq i}^{j-1} \frac{\sigma_{j\ell}^{i}}{\sigma_{j\ell}}, \tag{100}$$

where $\sigma_{j\ell}$ is the number of the shortest paths connecting the jth and ℓth nodes, and $\sigma_{j\ell}^{i}$ is the number of such shortest paths that pass through the ith node. The convention is that we regard the summand on the right-hand side of Eq. (100) to be zero when $\sigma_{j\ell}$ is equal to zero (i.e., when the jth and ℓth nodes are in different connected components). The summation excludes the shortest paths that start or end at the ith node because it is obvious that such a path does not go through the ith node. The normalisation factor $2/[(N-1)(N-2)]$ comes from the combinations of j and ℓ, whereas it is often neglected.

3. Katz centrality

Given an adjacency matrix, A, the number of walks from the ith node to the jth node with ℓ steps is given by the (i, j) element of A^ℓ. Supposing that short walks are more important than long walks in mediating, e.g., communication and infectious diseases, we scale the importance of each walk of length ℓ $(\ell \geq 0)$ by a factor of α^ℓ, where $0 < \alpha < 1$. Then, the weighted sum of the number of walks from the ith to the jth nodes of various lengths is given by the (i, j) element of

$$I + \alpha A + \alpha^2 A^2 + \cdots = (I - \alpha A)^{-1}. \tag{101}$$

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

The Katz centrality of the ith node is defined by

$$\text{Katz}_i = \sum_{j=1}^{N} \left[(I - \alpha A)^{-1}\right]_{ij}. \tag{102}$$

In other words, the weighted sum of the number of walks starting from the ith node is summed over all destination nodes. If $\alpha = 0$, then $\text{Katz}_i = 1$ for all i. Therefore, we are interested in making α large to diversify the values of Katz_i. In fact, as intuitively understood from Eq. (101), $(1 - \alpha A)^{-1}$ diverges for a large α. This occurs when an eigenvalue of $I - \alpha A$ hits zero for the first time as α is increased. Therefore, the Katz centrality is well-defined when α is smaller than the inverse of the largest eigenvalue of A.

4. 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. We will introduce it more in section VIII B, 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 node, PageRank is a typical recursive metric, based on the circular idea that: a node is important if it receives connections from many important nodes. As we will see, this relation leads to an eigenvector problem. Similar arguments lead to other centrality measures, such as Eigenvector centrality.

Ex.IV.5 : Take an undirected network and measure the correlation between different centrality measures. The correlation can either be estimated with the centrality values (Spearman) or with their associated ranking (Kendall).

Application: Centrality measures are often used to predict the influence of nodes in a network. For instance, in social networks, a combination of centrality measures can be used to predict the future impact of a user. In epidemiology, nodes with a high centrality are often targeted, e.g. by means of vaccination, in order to slow down the progress of a disease. Similarly, in marketing, central users can be targeted to seed viral campaigns. See for instance: Benchmarking Measures of Network Influence, Aaron Bramson and Benjamin Vandermarliere; or Going Viral, Karine Nahon and Jeff Hemsley.

F. 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 and normalised Laplacian are defined by

$$L_{ij} = k_i\delta_{ij} - A_{ij}, \tag{103}$$

$$\tilde{L}_{ij} = \delta_{ij} - \frac{A_{ij}}{\sqrt{k_i k_j}}. \tag{104}$$

The three matrices are connected by the following relationships:

$$L = D - A, \tag{105}$$

$$\tilde{L} = D^{-1/2} L D^{-1/2} = I - D^{-1/2} A D^{-1/2}, \tag{106}$$

where D is the $N \times N$ diagonal matrix whose (i,i) element is equal to k_i and I is the $N \times N$ identity matrix. Both Laplacian matrices are symmetric and their eigenvectors u_ℓ $(1 \le \ell \le N)$ form an orthonormal basis such that $\langle u_\ell, u_{\ell'}\rangle = \delta_{\ell\ell'}$. Any N-dimensional vector x can be decomposed as

$$x = \sum_{\ell=1}^{N} a_\ell u_\ell, \tag{107}$$

where

$$a_\ell = \langle x, u_\ell\rangle. \tag{108}$$

For the adjacency matrix, 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 $u_1 = (1 \ \cdots \ 1)^\top$ and $u_1 = (\sqrt{k_1} \ \cdots \ \sqrt{k_N})^\top$, respectively. In undirected networks, the zero eigenvalue, $\lambda_1 = 0$, is an isolated eigenvalue and all the other eigenvalues are positive such that $0 = \lambda_1 < \lambda_2 \le \cdots \le \lambda_N$ 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, as in our discussion on Markov chains, and is often called the spectral gap. The corresponding eigenvector, 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 $\lambda_2 > 0$.

The eigenvectors of the three matrices are the same for regular networks. For regular networks, the eigenvalues of the three matrices are related by

$$\lambda_i(L) = \langle k\rangle - \lambda_i(A), \tag{109}$$

$$\lambda_i(\tilde{L}) = 1 - \frac{\lambda_i(A)}{\langle k\rangle}, \tag{110}$$

where the argument specifies the matrix. For non-regular networks, these matrices have different spectral properties. Different bounds and results exist for the eigenvalues of these matrices. Notably, the spectrum of the normalised Laplacian, $\tilde{L}$, satisfies

$$0 = \lambda_1 \le \lambda_2 \le \cdots \le \lambda_N \le 2. \tag{111}$$

The largest eigenvalue verifies the equality $\lambda_N = 2$ if and only if the network is bipartite.

Ex.IV.6 : Compute the eigenvalues of the adjacency matrix, the (combinatorial) Laplacian matrix and the normalised Laplacian matrix of an undirected network. Verify that the Frobenius-Perron theorem is verified.

Application: As we will see more in detail below, spectral properties of networks can be used to reveal important structural properties, and are at the core of several algorithms. For degree centrality, the dominant eigenvector of the normalized Laplacian is essentially equivalent to Pagerank. Likewise, the dominant eigenvector of the adjacency matrix is called eigenvector centrality. The second dominant eigenvector is also associated to important structural patterns, that is the presence of botllenecks and communities in the network.

V. MODELS OF NETWORKS

When analysing structural patterns of empirical networks, it is important to compare their properties with those of appropriate reference points, often produced by models of networks. We distinguish two families of models. The first is random graph models in which links are random variables with certain constraints. The most fundamental model of random graph is the Erdős-Rényi model (Section V A), and other examples include the configuration model (Section V B). These models provide neither an explanation for the values taken by the parameters nor the reason for certain constraints to emerge in an empirical network. Instead, they have nice mathematical and statistical properties. For this reason, random graphs provide a useful baseline, or null model, for deciding whether patterns observed in empirical data are significant. In practice, if a value of a measurement observed in empirical data is significantly different from the expected value for the random graph model, the model does not represent the process behind the empirical data.

The second class of models is mechanistic models, whose goal is to understand the mechanisms leading to certain structures observed in empirical networks. In general, such models are defined by simple rules on how nodes and links are created or destroyed in the course of time. Examples include the growing network model (Section V D). Comparison between networks generated by the model and empirical networks allows us to identify potential forces having driven the evolution of the empirical networks.

A. Erdős-Rényi random graph

One of the simplest random graph models is the Erdős-Rényi random graph, introduced by Hungarian mathematicians Erdős and Rényi in 1959 . The model, also called the Poisson or binomial random graph, is denoted by $\mathcal{G}(N,q)$ and has two parameters, the number of nodes, N, and the probability q that a link exists between a pair of nodes. The self-loops are excluded. For each pair of nodes, consider a Bernoulli process that determines whether or not they are connected by a link. In fact, $\mathcal{G}(N,q)$ does not represent a single network, but a random ensemble of them in the probabilistic sense. Any network without multiple edges or self-loops is generated by the random graph $\mathcal{G}(N,q)$ as long as $0<q<1$. However, the probability that the model generates a target network, or a similar network, may be tiny. When studying the Erdős-Rényi random graph model or other network models defined as a random ensemble of networks (in fact, most models are so), a major aim is to predict the average behaviour of certain network metrics and, if possible, their variance.

In $\mathcal{G}(N,q)$, every link exists independently with the same probability. Therefore, the probability of generating a network with M links in total is given by the binomial distribution as follows:

$$p(M)=\binom{\frac{N(N-1)}{2}}{M}q^M(1-q)^{\frac{N(N-1)}{2}-M}, \tag{112}$$

where $N(N-1)/2$ is the maximum total number of links in a network. The expected number of links is given by $qN(N-1)/2$. Similarly, because a node is independently adjacent to any other node with probability q, the degree distribution is given by

$$p(k)=\binom{N-1}{k}q^k(1-q)^{N-1-k}, \tag{113}$$

which is the binomial distribution.

The Erdős-Rényi model is usually seen as a model for sparse networks, where the total number of links scales linearly with the number of nodes, N. Equivalently, the average degree of the nodes $\langle k\rangle = q(N-1)$ should not 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 $q=\langle k\rangle/(N-1)$ is sufficiently small to make $\langle k\rangle$ converge to a positive constant, the binomial degree distribution given by Eq. (113) is well approximated by the Poisson distribution

$$p(k)=\frac{\langle k\rangle^k}{k!}e^{-\langle k\rangle}. \tag{114}$$

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

$$L\approx\frac{\log N}{\log\langle k\rangle} \tag{115}$$

for q larger than $O(\log N/N)$ as $N\to\infty$, which ensures that the network is connected.

The clustering coefficient is given by

$$C=\frac{\binom{N}{3}q^3}{\binom{N}{3}q^2}=q=\frac{\langle k\rangle}{N-1}. \tag{116}$$

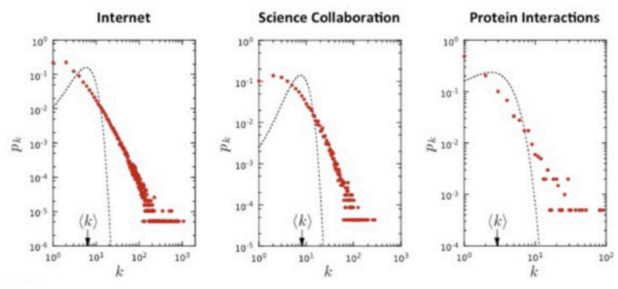

FIG. 13. Comparison between the degree distribution of 3 real-world networks and the corresponding Erdős-Rényi model with the same average connectivity.

When $\langle k \rangle$ does not depend on N, we obtain $C \to 0$ as $N \to \infty$. This calculation can be generalised to the counts of loops or cliques of larger size, leading to a similar observation: the density of such structures decays to zero as $N \to \infty$. This property has important implications in the study of dynamics on random networks, as the network has a locally tree-like structure. If one explores a generated network around a node, the structure is well approximated by a tree, and finding a cross link between two branches of the tree is extremely rare.

The Erdős-Rényi random graph also exhibits a phase transition. Let us consider the size (i.e., number of nodes) of the largest connected component in the network as a function of the mean degree $\langle k \rangle$. When $\langle k \rangle = 0$, the network is trivially composed of N disconnected nodes. In the other extreme of $\langle k \rangle = N - 1$, each node pair is adjacent such that the network is trivially connected. Between the two extremes, the network does not change smoothly in terms of the largest component size. Instead, a giant component, i.e., a component whose size is the largest and proportional to N, suddenly appears as $\langle k \rangle$ increases, marking a phase transition. In general, phase transitions represent qualitative changes in the behaviour of complex systems due to non-linearity. They imply that small changes of a parameter may have drastic consequences on the organisation and dynamics of a system.

The sudden emergence of a giant component is shown as follows. Consider the probability that a randomly chosen node does not belong to the giant component, denoted by u. If a giant component is absent in the network, we obtain $u = 1$. Otherwise, $u < 1$. By definition, if node v_i does not belong to the giant component, it must not be adjacent to any node v_j that belongs to the giant component. Therefore, for an arbitrary node v_j, node v_i is either not adjacent to v_j, which occurs with probability $1 - q$, or adjacent to v_j with the extra condition that v_j does not belong to the giant component, which occurs with probability qu. The probability that v_i does not belong to the giant component via v_j is thus equal to $1 - q + qu$. Because there are $N - 1$ candidate nodes as v_j, we obtain

$$u = (1 - q + qu)^{N-1}. \tag{117}$$

By applying

$$\lim_{N\to\infty} \left(1 - \frac{x}{N}\right)^N = e^{-x} \tag{118}$$

to Eq. (117), we obtain

$$u = e^{-\langle k \rangle (1-u)}. \tag{119}$$

in the limit $N \to \infty$.

The probability S that a node belongs to the giant component is equal to $1 - u$. Substitution of this relation to Eq. (119) yields

$$S = 1 - e^{-\langle k \rangle S}. \tag{120}$$

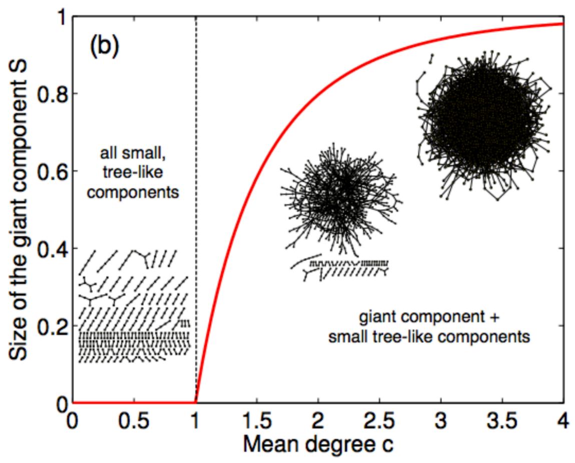

FIG. 14. Illustration of the different regimes of the Erdős-Rényi model.

The solutions of Eq. (120) are given by the intersection of $y = S$ and $y = 1 - e^{-\langle k \rangle S}$ in $0 \le S \le 1$. $S = 0$ is always a solution. Another solution exists if and only if $1 - e^{-\langle k \rangle S}$ grows faster than S at $S = 0$. Because $\mathrm{d}(1 - e^{-\langle k \rangle S})/\mathrm{d}S|_{S=0} = \langle k \rangle$, the critical point above which the giant component emerges is given by $\langle k \rangle = 1$. In the sub-critical regime $\langle k \rangle < 1$, no giant component exists, and the network is composed of a multitude of small components. In the super-critical regime $\langle k \rangle > 1$, a giant component made of $\propto N$ nodes emerges. The transition is continuous at the critical point. At the critical point, the size of the connected components obeys a power-law distribution.

An alternative and probably more intuitive way to show the emergence of the giant component is to adopt an dynamical viewpoint to build components by a branching process. We consider an analogue of infectious disease propagating along links and look at a node v_j that has been infected from its neighbour v_i. How many neighbours, other than v_i, can v_j infect in the next round? Because each link is an independent random process, the average number of newly infected nodes is equal to $q(N - 2) \approx \langle k \rangle$, in the limit $N \to \infty$. When $\langle k \rangle < 1$, the branching process terminates after a finite number of steps, and the components have a finite size. When $\langle k \rangle > 1$, the average number of new nodes grows exponentially and the branching process never ends with a positive probability. In practice, however, it must end because the network is finite.

Instances of the Erdős-Rényi random graph can be generated in different ways. One can perform a Bernoulli test of probability q on all pairs of nodes, which requires $O(N^2)$ operations. Alternatively, one can draw the degrees of N nodes from the Poisson distribution and build a network using the configuration model introduced in the next section, which respects the degree constraint. The second method is substantially faster for large networks.

The Erdős-Rényi random graph plays a fundamental role in network science. Its simple rules allow us to understand real-world phenomena, such as a small average distance L. However, it also produces unrealistic patterns, such as locally tree-like structure and the Poisson degree distribution. In fact, a majority of empirical networks has many triangles (i.e., not tree-like) and a long-tailed degree distribution.

Ex.V.1 : Reproduce numerically the results of Figure (14).

B. Configuration model

The configuration model is a generalisation of the Erdős-Rényi random graph to the case of an arbitrary but given degree of each node. 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 node v_i has degree k_i ($1 \leq i \leq N$). The degree sequence $\{k_i\}$ is 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 stubs (half-edges) at each node v_i such that its number is equal to k_i. Then, we randomly select pairs of stubs one by one to connect them as a link 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 in the middle. For example, if there remain three nodes 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 $k_j/2M$ because the number of stubs in the network is equal to $2M$. Because v_i owns k_i stubs, the expected number of links between v_i and v_j is given by

$$A^*_{ij} = \frac{k_i k_j}{2M}, \tag{121}$$

where A^*_{ij} represents the statistical average of the adjacency matrix.

When $\langle k^2 \rangle$ is finite, the average distance of the configuration model is given by

$$L = 1 + \frac{\log \frac{N}{\langle k \rangle}}{\log \frac{\langle k^2 \rangle - \langle k \rangle}{\langle k \rangle}} \tag{122}$$

for large N. For the power-law degree distribution $p(k) \propto k^{-\gamma}$, we obtain

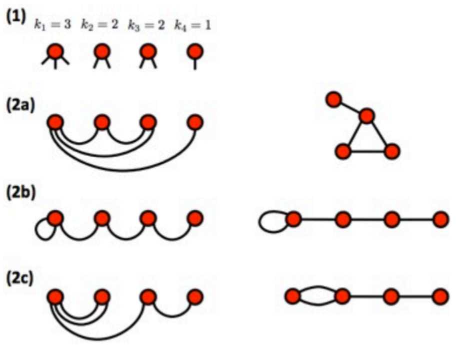

FIG. 15. Different realisations of the configuration model given a sequence of node degrees. Note that the network may present self-loops and multiple edges between pairs of nodes.

$$L \propto \begin{cases} \log \log N, & (2 < \gamma < 3), \\ \log N / \log \log N, & (\gamma = 3), \\ \log N, & (\gamma > 3), \end{cases} \tag{123}$$

. Only when $\gamma > 3$, $\langle k^2 \rangle$ is finite in the limit $N \to \infty$ such that Eq. (123) is consistent with Eq. (122). When $\gamma \leq 3$, L is very small such that the network is called ultra-small-world.

The clustering coefficient is given by

$$\begin{aligned} C &= \sum_{k'=1}^{\infty} \sum_{k''=1}^{\infty} \frac{k' p(k')}{\langle k \rangle} \frac{k'' p(k'')}{\langle k \rangle} \frac{(k'-1)(k''-1)}{\langle k \rangle N} \\ &= \frac{\left(\langle k^2 \rangle - \langle k \rangle \right)^2}{\langle k \rangle^3 N}. \end{aligned} \tag{124}$$

Clustering coefficient C is small unless the degree distribution is highly heterogeneous, in which case $\langle k^2 \rangle \gg \langle k \rangle^2$.

For the annealed adjacency matrix given by Eq. (121), the largest eigenvalue is evaluated as

$$\lambda_1 \approx \frac{\langle k^2 \rangle}{\langle k \rangle}. \tag{125}$$

Equation (125) diverges for networks with power-law degree distributions, so-called scale-free networks, with degree distribution $p(k) \propto k^{-\gamma}$, $\gamma \leq 3$. In this case, the largest eigenvalue diverges as $N \to \infty$.

Ex.V.2 : Generate randomized version of different empirical networks and verify that the number of self-loops and multiple edges becomes negligible when the system is sufficiently large.

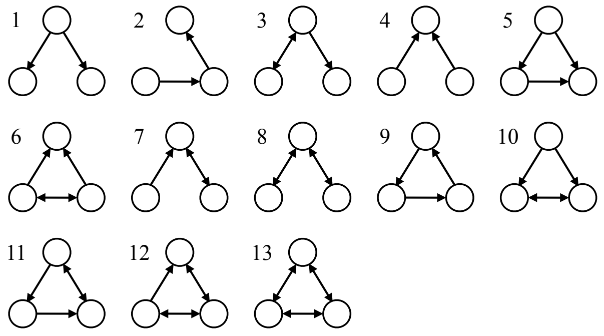

FIG. 16. Possible three-node motifs in static directed networks.

When analysing the properties of a real-world network, it is often advised to compare the observations to those made on an appropriate null model. The configuration model often plays this role, allowing to address the question: Does the sequence of node degrees alone allows to explain the observations, or does the system exhibit other important structural factors?

C. Network motifs

In a majority of empirical networks, triangles are abundant, which is the motivation of measuring the clustering coefficient. We can use the same argument to search for frequent small subgraphs as signatures of a given network. This is motif analysis, a computational method to enumerate small subgraphs (especially, three-node subgraphs) embedded in networks and assess whether a subgraph is significantly frequent. Significantly frequent subgraphs are called network motifs. The notion of statistical significance heavily relies on the notion of random graph defined above.

If we confine ourselves to weakly connected three-node subgraphs in directed networks, there are 13 candidate network motifs, as shown in Fig. 16. We should not determine relative frequency of these subgraphs simply by comparing the number of their appearance. For example, in sparse random networks, a subgraph containing fewer links, such as subgraph 1 in Fig. 16, would be more frequently found than a subgraph containing many links, such as subgraph 13. However, the difference in the frequency of subgraph 1 and that of subgraph 13 should be ascribed to sparsity of the network in this case, not to a particular tendency that this network prefers subgraph 1.

Therefore, we measure the frequency of each subgraph relative to that of a random network null model. While different null models do the job, the most frequently used null model is the directed variant of the configuration model, which is a random graph with the in- and out-degree of each node in a given network conserved. The implicit assumption then is that we are interested in over-represented subgraphs that cannot be explained by heterogeneity in the degree distribution (more precisely, the in- and out-degrees of each node in a given network). For any of the 13 three-node subgraphs, m, we denote by $C(m)$ the frequency of subgraph m in the given network and by $\tilde{C}(m)$ the frequency in a network generated from the configuration model. Because the configuration model is a random network model, the value of $\tilde{C}(m)$ is generally different every time we generate a network from the same configuration model. We define the Z score by

$$Z = \frac{C(m) - \langle \tilde{C}(m) \rangle}{\text{std}\left[\tilde{C}(m)\right]}, \tag{126}$$

where $\langle \tilde{C}(m) \rangle$ is the mean of $\tilde{C}(m)$ over the instances of networks generated by the configuration model, and std $\left[\tilde{C}(m)\right]$ is the standard deviation of $\tilde{C}(m)$. We calculate $\langle \tilde{C}(m) \rangle$ and std $\left[\tilde{C}(m)\right]$ on the basis of sufficiently many instances to lessen fluctuations in the estimates.

The Z score represents the normalised frequency of subgraph m relative to the configuration model. Calculating the Z score is essentially equivalent to calculating the p value in a statistical test. If the Z score is sufficiently large or small, subgraph m is overrepresented or underrepresented, respectively. Significantly overrepresented subgraphs are network motifs. Although the Z score is famous in the context of network motifs, significance of any quantity measured for a given network should be tested with the Z score whenever possible. Otherwise, we can be easily fooled by intuitively (but not necessarily statistically) large/small values of a measurement.

Network motifs can also be examined with larger subgraphs and undirected networks although there are exploding numbers of subgraphs to be searched and enumeration of each subgraph is computationally difficult for large subgraphs. For historical reasons, network motifs seem to be a main analysis tool for directed networks. Software mfinder is freely available for finding network motifs.

Ex.V.3: Take a directed empirical network and estimate the over/under-representation of its different directed triangles with respect to the configuration model.

Motif analysis is particularly popular in the study of biological systems, where over-represented motifs are understood as building blocks of the network, that can be combined to form more complex structures.

D. Growing network with preferential attachment

A broad range of networks grow in time in terms of both the number of nodes, N, and the number of links, 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. For example, there is a phenomenon called triadic closure, in which if there are links (v_1, v_2) and (v_2, v_3), then a new link (v_1, v_3) is likely to form, yielding a high clustering coefficient. Triadic closure is a mechanism that can be incorporated in growing network models.

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, 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 Pólya 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 m_0 nodes each of whose degree is at least one. A typical choice is the complete graph (i.e., a link exists between every pair of nodes) on m_0 nodes. Set a clock to $t = 0$.

2. Add a node with $m(\leq m_0)$ half-edges to the existing network. Suppose that the existing network has N' nodes with degrees k_i ($1 \leq i \leq N'$). The probability that each half-edge connects to v_i is specified by

$$\Pi(k_i) = \frac{k_i}{\sum_{j=1}^{N'} k_j} \quad (1 \leq i \leq N'). \tag{127}$$

Equation (127) indicates that a node receives a new link with the probability proportional to its degree, hence the name of preferential attachment. Equation (127) is applied under the constraint that we avoid multiple edges, although this constraint is non-essential. When $m \geq 2$, we have to decide on whether or not to update the relevant k_i values used in Eq. (127) when one of the m links has been added. However, this decision is again immaterial.

3. Add nodes one by one until we have N nodes according to step (2). Some of the first stages are schematically shown in Fig. 17.

As we show below, the BA model produces networks with a power-law degree distribution $p(k) \propto k^{-3}$. In early stages, nodes have similar values of k, which are

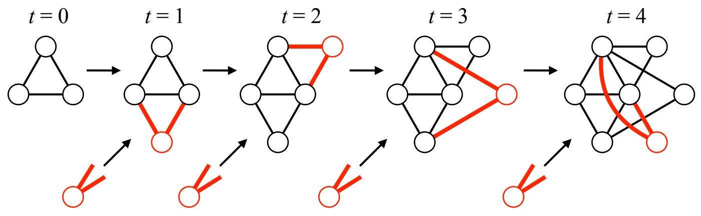

FIG. 17. First several stages of the BA model. The bold lines represent new links. We set $m_0 = 3$ and $m = 2$.

equal to or slightly larger than m. However, once $\{k_i\}$ 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 $p(k, t_i, t)$ the probability that a node v_i that has joined at time t_i has degree k at time t. The master equation for $p(k, t_i, t)$ is given by

$$p(k, t_i, t+1) = \frac{k-1}{2t} p(k-1, t_i, t) + \left(1 - \frac{k}{2t}\right) p(k, t_i, t) \tag{128}$$

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

$$p(k) = \lim_{t\to\infty} \frac{\sum_{t_i} p(k, t_i, t)}{t}. \tag{129}$$

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

The node that joins at time $t_i = t + 1$ has been absent at time t, such that $p(k, t+1, t) = 0$. By using this and summing Eq. (128) over t_i, we obtain

$$\sum_{t_i=1}^{t+1} p(k, t_i, t+1) = \frac{k-1}{2t} \sum_{t_i=1}^{t} p(k-1, t_i, t) + \left(1 - \frac{k}{2t}\right) \sum_{t_i=1}^{t} p(k, t_i, t). \tag{130}$$

By substituting $p(k) \approx \sum_{t_i=1}^{t} p(k, t_i, t)/t = \sum_{t_i=1}^{t+1} p(k, t_i, t+1)/(t+1)$ and $p(k-1) \approx \sum_{t_i=1}^{t} p(k-1, t_i, t)/t$ in Eq. (130), we obtain

$$(t+1)p(k) = \frac{k-1}{2t} tp(k-1) + \left(1 - \frac{k}{2t}\right) tp(k). \tag{131}$$

Equation (131) yields

$$p(k) = \frac{k-1}{k+2} p(k-1) \quad (k \geq m+1), \tag{132}$$

which yields

$$p(k) \propto \frac{1}{k(k+1)(k+2)} \propto k^{-3}. \tag{133}$$

Several other properties of the model have been derived analytically. By allowing self-loops and multiple edges to facilitate mathematical analysis, one obtains

$$L \propto \begin{cases} \log N, & (m = 1), \\ \log N / \log \log N, & (m \geq 2). \end{cases} \tag{134}$$

The network has a small average distance already with $m = 1$, and L is even smaller for $m \geq 2$. The clustering coefficient is given by

$$C \approx \frac{m-1}{8} \frac{(\log N)^2}{N}. \tag{135}$$

This equation implies that the BA model lacks the clustering property because $\lim_{N\to\infty} C = 0$. Various extensions of the BA model realise a non-vanishing C value as $N \to \infty$.

Ex. V.4: Generate numerically networks according to the BA model and verify the theoretical predictions for the degree distribution.

The BA model, as it has been exposed above, contains unrealistic ingredients, as a new node must have access to information about the whole network in order to decide which node 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 networks with a high density of cliques of different sizes. See Lambiotte, R., et al. "Structural Transitions in Densifying Networks." Physical review letters 117.21 (2016): 218301.

VI. COMMUNITY DETECTION

Many networks exhibit community structure. Community structure implies that the network is composed of groups of nodes such that the nodes are densely connected within the same group and relatively sparsely connected across different groups. According to a community detection algorithm, the social network of bottlenose dolphins shown in Fig. 18 has four communities indicated by different colours. There are many algorithms aiming to detect community structure in a given network in the absence of predefined labelling of nodes. 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. Alternative methods will be presented at a later stage, in the chapter dedicated to dynamics on networks.

A. Graph partitioning

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 minimized. 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. In this section, we consider the simplest instance of the problem, with $c = 2$, thus consisting in finding the best bipartition of the nodes such that the number of edges between the groups is minimized.

By definition, given a partition, the number of edges R running between the two groups of vertices, also called the *cut size*, is given by

$$R = \tfrac{1}{2} \sum_{\substack{i,j \text{ in} \\ \text{different} \\ \text{groups}}} A_{ij}, \tag{136}$$

which can be rewritten in more convenient form by defining the index verctor

$$s_i = \begin{cases} +1 & \text{if vertex } i \text{ belongs to group 1,} \\ -1 & \text{if vertex } i \text{ belongs to group 2.} \end{cases} \tag{137}$$

The latter satisfies the normalization condition $\mathbf{s}^T\mathbf{s} = n$. After using

$$\tfrac{1}{2}(1 - s_i s_j) = \begin{cases} 1 & \text{if } i \text{ and } j \text{ are in different groups,} \\ 0 & \text{if } i \text{ and } j \text{ are in the same group,} \end{cases} \tag{138}$$

we rewrite Eq. (136) as

$$R = \tfrac{1}{4} \sum_{ij} (1 - s_i s_j) A_{ij}. \tag{139}$$

and, after some algebra,

$$R = \tfrac{1}{4} \sum_{ij} s_i s_j (k_i \delta_{ij} - A_{ij}). \tag{140}$$

We can write this in matrix form as

$$R = \tfrac{1}{4} \mathbf{s}^T \mathbf{L} \mathbf{s}, \tag{141}$$

where the real symmetric matrix $\mathbf{L}$ with elements $L_{ij} = k_i \delta_{ij} - A_{ij}$ is the Laplacian matrix of the graph.

The graph partitioning problem is thus equivalent to choosing the vector $\mathbf{s}$ so as to minimize the cut size, Eq. (141). We rewrite the index vector as a linear combination of the normalized eigenvectors $\mathbf{v}_i$ of the Laplacian, $\mathbf{s} = \sum_{i=1}^n a_i \mathbf{v}_i$, where $a_i = \mathbf{v}_i^T \mathbf{s}$ and the normalization $\mathbf{s}^T \mathbf{s} = n$ implies that

$$\sum_{i=1}^{n} a_i^2 = n. \tag{142}$$

Then

$$R = \sum_i a_i \mathbf{v}_i^T \mathbf{L} \sum_j a_j \mathbf{v}_j = \sum_{ij} a_i a_j \lambda_j \delta_{ij} = \sum_i a_i^2 \lambda_i, \tag{143}$$

where λ_i is the eigenvalue of $\mathbf{L}$ corresponding to the eigenvector $\mathbf{v}_i$. By convention we assume that the eigenvalues are labeled in increasing order $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$. The task of minimizing R thus consists in choosing the nonnegative quantities a_i^2 so as to place as much as possible of the weight in the sum (143) in the terms corresponding to the lowest eigenvalues, and as little as possible in the terms corresponding to the highest, while respecting the normalization constraint (142).

We have shown before that the vector $(1, 1, 1, \ldots)$ is the dominant eigenvector of the Laplacian with eigenvalue zero, and that all eigenvalues of the Laplacian are nonnegative. Given these observations it is now straightforward to see how to minimize the cut size R. If we choose $\mathbf{s} = (1, 1, 1, \ldots)$, then all of the weight in the final sum in Eq. (143) is in the term corresponding to the lowest eigenvalue $\lambda_1 = 0$ and all other terms are zero, since $(1, 1, 1, \ldots)$ is an eigenvector and the eigenvectors are orthogonal. Thus this choice gives us $R = 0$, which is the smallest value it can take since it is by definition a nonnegative quantity.

Unfortunately, when we consider the physical interpretation of this solution, we see that it is trivial and uninteresting. Given the definition (137) of $\mathbf{s}$, the choice $\mathbf{s} = (1, 1, 1, \ldots)$ is equivalent to placing all the vertices in group 1 and none of them in group 2. Technically, this is a valid division of the network, but it is not a useful one. Of course the cut size is zero if we put all the vertices in one of the groups and none in the other, but such a trivial solution tells us nothing about how to solve our original problem.

We would like to forbid this trivial solution, so as to force the method to find a nontrivial one. To do so, we shift our attention to the other terms in the sum. If there were no further constraints on our choice of $\mathbf{s}$, apart from the normalization condition $\mathbf{s}^T \mathbf{s} = n$, our course would be clear: R would be minimized by choosing $\mathbf{s}$ proportional to the second eigenvector $\mathbf{v}_2$ of the Laplacian, also called the *Fiedler vector*. This choice places all of the weight in Eq. (143) in the term involving the second-smallest eigenvalue λ_2, also known as the *algebraic connectivity*. The other terms would automatically be zero, since the eigenvectors are orthogonal.

Unfortunately, there is an additional constraint on $\mathbf{s}$ imposed by the condition, Eq. (137), that its elements take the values ± 1, which means in most cases that $\mathbf{s}$ cannot be chosen parallel to $\mathbf{v}_2$. This makes the optimization problem much more difficult. Often, however, quite good approximate solutions can be obtained by choosing $\mathbf{s}$ to be as close to parallel with $\mathbf{v}_2$ as possible. This means maximizing the quantity

$$\left| \mathbf{v}_2^T \mathbf{s} \right| = \left| \sum_i v_i^{(2)} s_i \right| \leq \sum_i \left| v_i^{(2)} \right|, \tag{144}$$

where $v_i^{(2)}$ is the ith element of $\mathbf{v}_2$. Here the second relation follows via the triangle inequality, and becomes an equality only when all terms in the first sum are positive (or negative). In other words, the maximum of $|\mathbf{v}_2^T \mathbf{s}|$ is achieved when $v_i^{(2)} s_i \geq 0$ for all i, or equivalently when s_i has the same sign as $v_i^{(2)}$. Thus the maximum is obtained with the choice

$$s_i = \begin{cases} +1 & \text{if } v_i^{(2)} \geq 0, \\ -1 & \text{if } v_i^{(2)} < 0. \end{cases} \tag{145}$$

Even this choice however is often forbidden by the condition that the number of $+1$ and -1 elements of $\mathbf{s}$ be equal to the desired sizes n_1 and n_2 of the two groups, in which case the best solution is achieved by assigning vertices to one of the groups in order of the elements in the Fiedler vector, from most positive to most negative, until the groups have the required sizes. For groups of different sizes there are two distinct ways of doing this, one in which the smaller group corresponds to the most positive elements of the vector and one in which the larger group does. We can choose between them by calculating the cut size R for both cases and keeping the one that gives the better result.

This then is the spectral partitioning method in its simplest form. It is not guaranteed to minimize R, but, particularly in cases where λ_2 is well separated from the eigenvalues above it, it often does very well.

Ex.VI.1 : Implement numerically a spectral bipartitioning method, taking as an input a graph, and the desired size of the clusters.

B. Modularity

In the graph partitioning problem, we have implicitly have had to fix the number of groups but also their size. 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 underlying 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 advantage 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 set of nodes, denoted by CM. The underlying idea of modularity is to compare the number of links connecting nodes within CM with the expected number of links in an appropriate null model. Under the configuration model, the probability that nodes v_i and v_j are adjacent is given by $P_{ij} = k_i k_j / 2M$ (Eq. (121)). Other choices for the null model P_{ij} have also been considered. We quantify the contribution of CM to Q as

$$\sum_{\substack{i,j=1;\\ v_i,v_j \in \mathrm{CM}}}^{N} \left(A_{ij} - \frac{k_i k_j}{2M} \right). \tag{146}$$

Let us now consider a partition of the network into N_{CM} communities. The cth community ($c = 1, 2, \ldots, N_{\mathrm{CM}}$) is denoted by CM_c. Modularity is simply defined as a properly normalised sum of Eq. (146) over all communities, i.e.,

$$\begin{aligned} Q &= \frac{1}{2M} \sum_{c=1}^{N_{\mathrm{CM}}} \left[\sum_{\substack{i,j=1;\\ v_i,v_j \in \mathrm{CM}_c}}^{N} \left(A_{ij} - \frac{k_i k_j}{2M} \right) \right] \\ &= \sum_{c=1}^{N_{\mathrm{CM}}} \left[\frac{M_c}{M} - \left(\frac{\sum_{i=1; v_i \in \mathrm{CM}_c}^{N} k_i}{2M} \right)^2 \right], \end{aligned} \tag{147}$$

where M_c is the number of links connecting two nodes within community CM_c. According to Eq. (147), calculation of Q only requires the number of intra-community links and the sum of the degree of nodes within each community. We can also rewrite Q as

$$Q = \frac{1}{2M} \sum_{i,j=1}^{N} \left(A_{ij} - \frac{k_i k_j}{2M} \right) \delta(g_i, g_j), \tag{148}$$

where g_i is the community that the ith node belongs to, and $\delta(g_i, g_j) = 1$ if $g_i = g_j$ and $\delta(g_i, g_j) = 0$ otherwise.

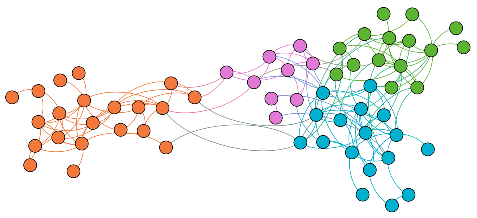
FIG. 18. A social network of bottlenose dolphins. Four communities detected by the Louvain algorithm implemented on gephi (http://www.gephi.org) are shown by different colours.

Maximising Q is expected to uncover the best partition into the best number of communities. However, the problem is not that simple, as we will discuss later. Modularity ranges in $[-0.5, 1]$. The trivial partition into one large community always yields

$$\begin{aligned} Q &= \frac{1}{2M} \sum_{i,j=1}^{N} \left(A_{ij} - \frac{k_i k_j}{2M} \right) \\ &= \frac{1}{2M} \left(2M - \frac{\sum_{i=1}^{N} k_i \sum_{j=1}^{N} k_j}{2M} \right) \\ &= \frac{1}{2M} \left(2M - \frac{2M \times 2M}{2M} \right) = 0, \end{aligned} \tag{149}$$

where we used the handshaking lemma.

Ex.VI.2 : Write a function that takes a graph and its partition as an input and returns its modularity.

C. Spectral optimization of modularity

Optimizatizing community is far from trivial, as it was proved to be NP-hard, and various approximate optimization 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 index vector $\mathbf{s}$ with elements as in Eq. (137), satisfying

$$\delta(g_i, g_j) = \tfrac{1}{2}(s_i s_j + 1). \tag{150}$$

Thus we can write modularity in the form

$$\begin{aligned} Q &= \frac{1}{4M} \sum_{ij} [A_{ij} - \frac{k_i k_j}{2M}](s_i s_j + 1) \\ &= \frac{1}{4M} \sum_{ij} [A_{ij} - \frac{k_i k_j}{2M}] s_i s_j, \end{aligned} \tag{151}$$

where we have used the handshaking lemma. This result can be conveniently rewritten in matrix form as

$$Q = \frac{1}{4M}\mathbf{s}^T\mathbf{B}\mathbf{s}, \tag{152}$$

where $\mathbf{B}$ is the real symmetric matrix having elements

$$B_{ij} = A_{ij} - \frac{k_i k_j}{2M}. \tag{153}$$

This matrix is called the *modularity matrix*, which satisfies the following properties. All rows (and columns) of the modularity matrix sum to zero, which implies that the vector $(1, 1, 1, \ldots)$ 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 (152) is the equivalent of Eq. (141) for the cut size and similar matrix methods can thus be applied to modularity optimization. By direct analogy, we write $\mathbf{s}$ as a linear combination of the normalized eigenvectors $\mathbf{u}_i$ of the modularity matrix, $\mathbf{s} = \sum_{i=1}^n a_i\mathbf{u}_i$ with $a_i = \mathbf{u}_i^T\mathbf{s}$, so that

$$Q = \frac{1}{4M}\sum_i a_i^2\beta_i, \tag{154}$$

where β_i is the eigenvalue of $\mathbf{B}$ corresponding to the eigenvector $\mathbf{u}_i$. We now assume that the eigenvalues are labeled in *decreasing* order $\beta_1 \geq \beta_2 \geq \ldots \geq \beta_n$. Modularity optimization 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, this would be a simple task if our choice of $\mathbf{s}$ were unconstrained (apart from normalization): 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 $s_i = \pm 1$, which means that $\mathbf{s}$ cannot normally be chosen parallel to $\mathbf{u}_1$. Again as before, however, 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

$$s_i = \begin{cases} +1 & \text{if } u_i^{(1)} \geq 0, \\ -1 & \text{if } u_i^{(1)} < 0. \end{cases} \tag{155}$$

This then is our first and simplest algorithm for community detection: we find the eigenvector corresponding to the most positive eigenvalue of the modularity matrix 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 recursively. 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 in which they are placed.

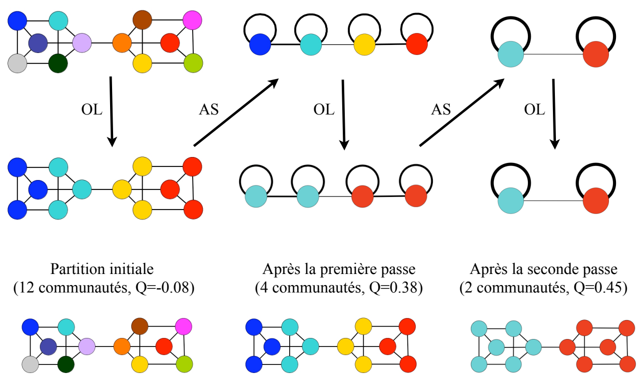

FIG. 19. Each pass of the algorithm is split into two phases. The first phase consists in a local optimization (LO), where each vertex can join one of its direct neighbors community. The second phase consists in a merging of the vertices (VM), i.e to the construction of a meta-graph whose vertices are the communities found at the end of the first phase. These passes are repeated until a modularity maximum is reached.

Ex.VI.3 : Implement numerically a spectral method to find the bipartition optimising modularity, taking as an input a graph. Test it on the Karate Club.

D. Louvain method

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. One of its limitations is computational, as its implementation requires the estimation of eigenvectors of the modularity matrix, which is prohibitively expensive for systems beyong 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 most 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 0 and $N-1$ has been randomly assigned. Algorithm 1 is a pseudo-code version of the method. It is then designed as follows:

- **First phase: local optimization.** The initial partition consists in placing each vertex into a separate community, this partition is therefore composed of N singleton communities. We then consider the first vertex, i.e. with index 0, and calculate the modularity variation obtained by removing 0 from its community and placing it in the community of one of its neighbors j. This variation is therefore calculated for each of the neighbors of 0 and the vertex 0 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 the vertex 0 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 node $N-1$, one returns to node 0, 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 $\mathcal{P}$ 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 link between two of these new vertices is given by the sum of the weights of the links 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 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 neighborhood. 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 recalls 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.

Algorithm 1 Pseudo-code of the community detection algorithm.

```
1: Community detection$G$ initial graph
2: repeat
3: Place each vertex of $G$ into a single community
4: Save the modularity of this decomposition
5: while there are moved vertices do
6: for all vertex $n$ of $G$ do
7: $c \leftarrow$ neighboring community maximizing the modularity increase
8: if $c$ results in a strictly positive increase then
9: move $n$ from its community to $c$
10: end if
11: end for
12: end while
13: if the modularity reached is higher than the initial modularity, then
14: $end \leftarrow false$
15: Display the partition found
16: Transform $G$ into the graph between communities
17: else
18: $end \leftarrow true$
19: end if
20: until $end$
```

Ex.VI.4 : The efficiency of the algorithm partly resides in the fact that the variation of modularity Δ_{ij} obtained by moving a vertex i from its community to the community of one of its neighbors j can be calculated with only local information. In practice, the variation of modularity is calculated by removing i from its community $\Delta_{remove;i}$ (this is only done once) then inserting it into the community of j $\Delta_{insert;ij}$ for each neighbor j of i. The variation is therefore: $\Delta_{ij} = \Delta_{remove;i} + \Delta_{insert;ij}$. Derive analytically $\Delta_{remove;i}$ when removing node i from its community C_i.

E. 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. Finally, although modularity allows us to com-

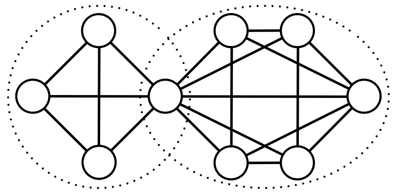

FIG. 20. Schematic of overlapping communities. Two communities are shown by dotted circles. One node belongs to both communities.

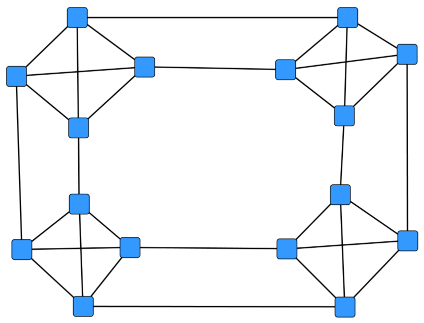

FIG. 21. Ring of 4 cliques.

pare 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 a 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.

Ex.VI.5 : Consider a ring of $N = 2K$ cliques, each composed of 4 nodes. Cliques are connected as in Figure 21. When does modularity favour a partition into K groups of 2 adjacent cliques over a partition into 2 K cliques. How would you interpret the result?

VII. DYNAMICS, TIME-SCALES AND COMMUNITIES

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 invariant subspaces in the dynamics that can be precisely described by graph partitions.

A. Notation

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 nodes (vertices) and m links (edges). 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 adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, where the weight of the link between node i and node j is given by A_{ij}. Note that $\mathbf{A} = \mathbf{A}^\top$ for an undirected graph. The weighted outdegrees (or strengths) of the nodes are given by the vector outdeg $= \mathbf{d} = \mathbf{A1}$, where $\mathbf{1}$ is the $n \times 1$ vector of ones. For a vector $\mathbf{x}$, we define diag($\mathbf{x}$) to be the matrix $\mathbf{X}$ with elements $X_{ii} = x_i$ and zero otherwise. We thus define the diagonal matrix of degrees as $\mathbf{D} = \text{diag}(\mathbf{d})$. The *combinatorial* graph Laplacian is defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$. It is symmetric positve semi-definite, with a simple zero eigenvalue when the graph is connected.

B. Consensus dynamics

Consensus has been one of the most popular and well-studied dynamics on networks. This is due to both its

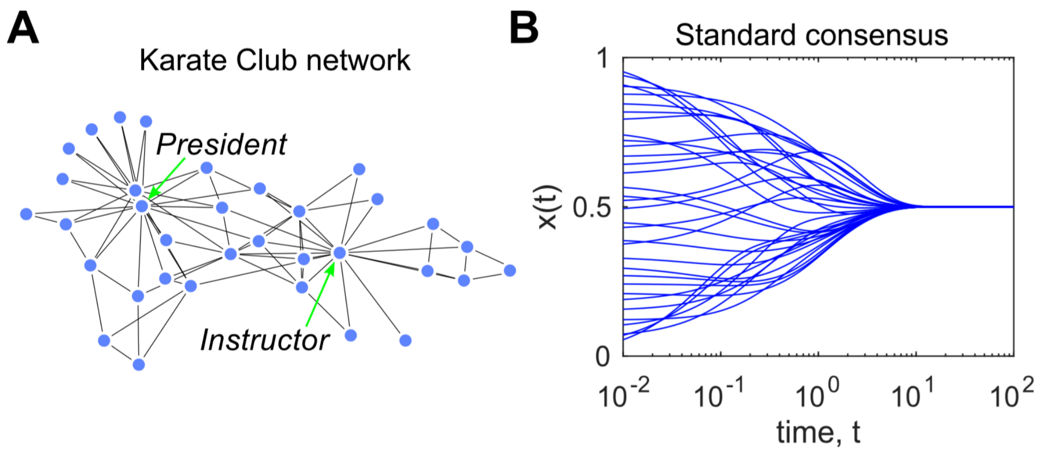

FIG. 22. **Illustration of a consensus dynamics on the Karate Club network. A** Karate Club network orginally analysed by Zachary. **B** Consensus dynamics on the Karate club network starting from a random initial condition. As discussed in the text, as time progresses the states of the individual nodes become more and more aligned, and eventually reach a consensus value, equal to the arithmetic average of the initial condition.

analytic tractability as well as its simplicity in approximating several fundamental behaviors. 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, consider a given connected network of n nodes and adjacency matrix A. Let us endow each node with a scalar state variable $x_i \in \mathbb{R}$. The (average) consensus dynamics on such a network is then defined as:

$$\dot{\mathbf{x}} = -\mathbf{L}\mathbf{x}, \qquad \text{(consensus dynamics)} \tag{156}$$

where L is the graph Laplacian. Note that in coordinate form this simply amounts to $\dot{x}_i = \sum_j A_{ij}(x_i - x_j)$, i.e., each node adjusts its state such that the difference to its neighbours is reduced. The name of these dynamics derives 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. Mathematically, this means that $x_i = x_*$ for all i as $t \to \infty$, where $x_* = \mathbf{1}^\top \mathbf{x_0}/n$ is given by 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. As we will show in the next chapter, this process is the dual of a random walk process taking place on a network.

Ex.VII.1 : Write a code to simulate consensus dynamics on a network, and verify that the dynamics asymptotically converges towards the state $x_* = \mathbf{1}^\top \mathbf{x_0}/n$.

C. 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:

$$\frac{dx}{dt} = f(x,y), \tag{157}$$

$$\epsilon^{-1}\frac{dy}{dt} = g(x,y), \tag{158}$$

where $\epsilon \ll 1$ 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

$$\frac{dx}{dt} = f(x,y), \tag{159}$$

$$\frac{dy}{d\tau} = g(x,y). \tag{160}$$

Whence, 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 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) behavior 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, e.g., that $x(t)$ eventually converges to some fixed point x_*. Since the behaviour of x is much faster than the time-evolution of y, we may simply assume that x has already converged. 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. However, it allows us again to focus on a simpler one-dimensional system, facilitating a simpler analysis.

To summarize, the separation of time-scales acts in such a way that it 'almost decouples' the system in two different regimes. For the fast system behavior, we may simply concentrate on x, whereas for the slow, long-term behavior we may simply focus on y and forget about the details of x.

D. Time-scale separation in networks with consensus dynamics

Let us now discuss how the above ideas can be translated into the context of networks on which a diffusion or consensus dynamics is acting. For simplicity we will describe the results here in the context of consensus, though translating these ideas to diffusion processes is straightforward.

For an initial condition $\mathbf{x}_0$, standard linear systems theory tells us that the solution to (156) is given by $\mathbf{x}(t) = \exp(-\mathbf{L}t)\mathbf{x}_0$, where $\exp(\cdot)$, denotes the matrix exponential. Writing the solution in this way disguises however the time-scales present in the evolution of $\mathbf{x}$ 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., $\mathbf{L}\mathbf{v}_i = \lambda_i \mathbf{v}_i$, 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 can now decompose the Laplacian as $\mathbf{L} = \sum_i \lambda_i \mathbf{v}_i \mathbf{v}_i^\top$, and accordingly write the solution $\mathbf{x}(t)$ in this spectral basis as $\mathbf{x}(t) = \sum_i \exp(-\lambda_i t)\mathbf{v}_i \mathbf{v}_i^\top \mathbf{x}_0$.

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 $\{0, \ldots, \lambda_k\}$, which are well separated from the remaining eigenvalues in the sense that $\lambda_k \ll \lambda_{k+1}$. Then after some time $\tau_0 \propto 1/\lambda_{k+1}$, the eigenmodes associated with eigenvalues $\{\lambda_{k+1}, \ldots, \lambda_n\}$ 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.

E. Example: A modular partitioned network structure induces time-scale separation

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 can be written as:

$$\mathbf{A} = \mathbf{A}_{\text{structure}} + \mathbf{A}_{\text{random}} = \begin{pmatrix} \mathbf{A}_1 & & & \\ & \mathbf{A}_2 & & \\ & & \ddots & \\ & & & \mathbf{A}_k \end{pmatrix} + \mathbf{A}_{\text{random}}.$$

Here $\mathbf{A}_{\text{random}}$ may be interpreted as a realization from an Erdős-Rényi random graph, with unstructured, sparse connectivity (a 'noise term'); A_i are the adjacency (sub-)matrices of the individual clusters which have higher connectivity inside.

How is the spectrum and the eigenvalues of the corresponding Laplacian $\mathbf{L} = \mathbf{L}_{\text{structure}} + \mathbf{L}_{\text{random}}$ affected by the structure present? Let us first consider the case where $\mathbf{L}_{\text{random}} = 0$, i.e., 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 $\mathbf{c}_i^{(j)} = 1$ if node i is in component j, and zero otherwise. To gain insight into the case where $\mathbf{L}_{\text{random}} \neq 0$, 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 $\mathcal{A}$ spanned by $\{\mathbf{c}^{(j)}\}$, and the subspace $\mathcal{A}'$ spanned by the corresponding eigenvectors of $\mathbf{L}$ associated with the smallest eigenvalues. On an intuitive level, the Davis-Kahan theorem states that if the noise level is not too high, then $\mathcal{A} \approx \mathcal{A}'$. 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 23A consists of 3 groups with 100 nodes each and is structured as outlined in our discussion above. As can be seen clearly in Figure 23B, the dynamics becomes effectively low dimensional after around $t = 0.05$ and can be well approximated by the dominant eigenmodes.

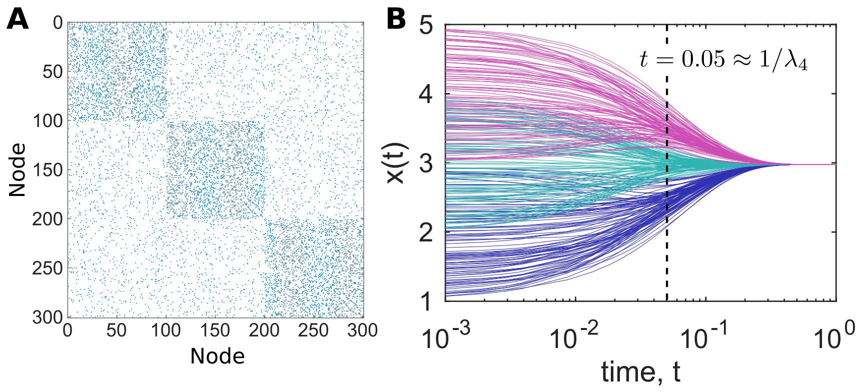

FIG. 23. **Illustration of a consensus dynamics on a structured network.** **A** Adjacency matrix (unweighted) of a structured network with 5 groups, as discussed in the text. **B** A consensus dynamics on this network displays a time-scale separation: until around $t = 0.05$, approximate consensus is reached within each group (groups indicated by color); then a consensus is reached between the groups. Note that for the shown network $\lambda_4 = 18$, in good agreement with our discussion above.

Finally, let us remark that most of the above discussion is not limited to the case where the structure of the network is effectively block-diagonal, but can be extended to the general case where the network consists of a low-rank structure plus a random 'noise' component. Note that this also includes networks that are structured according to a stochastic blockmodel, though if the network is very sparse the spectral properties of a realization of a stochastic blockmodel may not be concentrated around their expectation.

Ex.VII.2 : Write a code to reproduce the numerical results of Figure 23.

F. Dynamically invariant subspaces and external equitable partitions in networks

Let us now consider a somewhat different network structure that has important repercussions for a dynamical process acting on the network: the so-called external equitable partition (EEP).

EEPs extend the well known graph-theoretic notion of equitable partition (EP). An EP splits the graph into a set $\{\mathcal{C}_i\}$ of non-overlapping groups of nodes called cells. The number of connections to cell $\mathcal{C}_j$ from any node $v \in \mathcal{C}_i$ is only dependent on i, j. Stated differently, the nodes inside each cell of an EP have the same out-degree pattern with respect to every cell. For EEPs, this requirement is relaxed so that it needs to hold only for the number of connections between *different* cells $\mathcal{C}_i, \mathcal{C}_j$ $(i \neq j)$. An example a graph with an EEP is shown in Figure 24.

Algebraically, these definitions can be represented as follows. A partition of a graph with n nodes into c cells is encoded by the $n \times k$ indicator matrix $\mathbf{C}$, defined as $\mathbf{C}_{ij} = 1$ if node i is part of cell $\mathcal{C}_j$ and $\mathbf{C}_{ij} = 0$ otherwise. Hence the columns of $\mathbf{C}$ are indicator vectors $\mathbf{c}^{(i)}$ of the cells:

$$\mathbf{C} := [\mathbf{c}^{(1)}, \dots, \mathbf{c}^{(k)}]. \tag{161}$$

Given the Laplacian matrix L of a graph, we can write the definition of an EEP as follows:

$$\mathbf{L}\mathbf{C} = \mathbf{C}\mathbf{L}^{\pi}. \tag{162}$$

Here $\mathbf{L}^{\pi}$ is the $c \times c$ Laplacian of the quotient graph induced by H:

$$\mathbf{L}^{\pi} = (\mathbf{C}^{\top}\mathbf{C})^{-1}\mathbf{C}^{\top}\mathbf{L}\mathbf{C} = \mathbf{C}^{+}\mathbf{L}\mathbf{C}, \tag{163}$$

where the $k \times n$ matrix $\mathbf{C}^{+}$ is the (left) Moore-Penrose pseudoinverse of $\mathbf{C}$. Observe that multiplying a vector $\mathbf{x} \in \mathbb{R}^N$ by $\mathbf{C}^{\top}$ from the left sums up the components within each cell, and that $\mathbf{C}^{\top}\mathbf{C}$ is a diagonal matrix with the number of nodes per cell on the diagonal. Hence $\mathbf{C}^{+}$ may be interpreted as a *cell averaging operator.*

The quotient graph associated with an EEP is a coarse-grained version of the original graph, such that each cell of the partition becomes a new node and the weights between these new nodes are the out-degrees between the cells in the original graph (see Fig. 24a). Although the Laplacian of the original graph is symmetric, the quotient Laplacian will be asymmetric in general.

Using some straightforward algebraic manipulations it is moreover easy to show that:

$$\mathbf{C}^{+}\mathbf{L} = \mathbf{L}^{\pi}\mathbf{C}^{+}, \tag{164}$$

which summarises the relationship between the cell averaging operator $\mathbf{C}^{+}$ and the Laplacians of the original and quotient graphs.

The definition of the EEP (162) can be understood as an invariance of the partition encoded by $\mathbf{C}$ with respect to the Laplacian $\mathbf{L}$. Similarly, Eq. (164) shows that the cell averaging operator $\mathbf{C}^{+}$ exhibits a (distinct) invariance with respect to $\mathbf{L}$. In particular, Eq. (162) implies

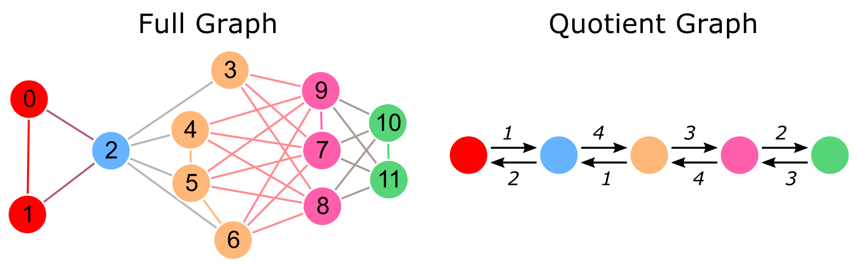

FIG. 24. **External equitable partition** A graph with $n = 12$ nodes (left) with an external equitable partition into five cells (indicated with colors) and its associated quotient graph according to the EEP (right).

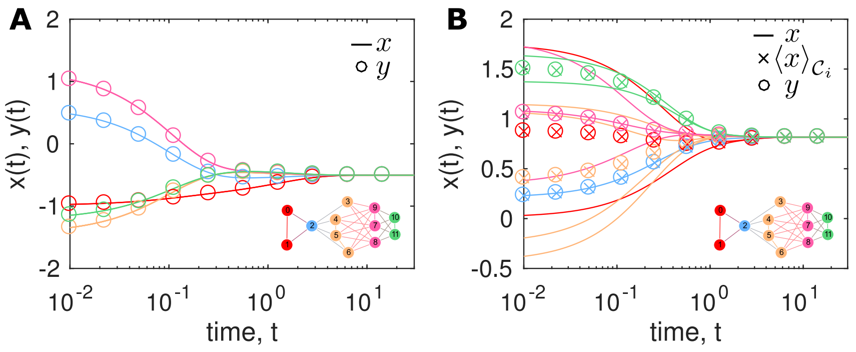

FIG. 25. **Dynamical implication of an external equitable partition** We consider again the graph shown in Figure 24. **A** The evolution of the consensus dynamics on the full graph (156) from an initial condition $\mathbf{x} = \mathbf{C}\mathbf{y}$ is shown with solid lines. The associated quotient dynamics (165) governing $\mathbf{y}$ is shown with circles. Once all states within each cell are equal (i.e., they are cluster-synchronized), the dynamics will remain cluster-synchronized and its dynamics will be described by the quotient dynamics for all times. **B** For consensus dynamics, the quotient graph dynamics (circles) also describes the cell-averaged dynamics (crosses) of the unsynchronized full graph dynamics (solid lines), as given by (166).

that the associated cell indicator matrix $\mathbf{C}$ spans an invariant subspace of $\mathbf{C}$, whence it follows that there exists a set of eigenvectors which is localised on the cells of the partition. Furthermore, the eigenvalues associated with the eigenvectors spanning the invariant subspace are shared with $\mathbf{L}^\pi$, the Laplacian of the quotient graph. If $\mathbf{C}$ has degenerate eigenvalues, an eigenbasis can be chosen such that it is localised on the cells of the partition.

The properties of the EEP (162)–(164) have noteworthy consequences for the dynamics dictated by $\mathbf{L}$, as illustrated by the case of linear consensus dynamics:

First, as shown in Fig. 24A, the EEP is consistent with an exact form of cluster consensus. In particular, if the initial state vector is given by $\mathbf{x} = \mathbf{C}\mathbf{y}$ for some arbitrary $\mathbf{y}$ (i.e., all the nodes within cell $\mathcal{C}_i$ have the same value y_i), the nodes inside the cells remain identical for all times and their dynamics is governed by the quotient graph:

$$\dot{\mathbf{x}} = \mathbf{C}\dot{\mathbf{y}} \quad \text{where} \quad \dot{\mathbf{y}} = -\mathbf{L}^\pi \mathbf{y}. \tag{165}$$

This follows directly from $\mathbf{L}\mathbf{C}\mathbf{y} = \mathbf{C}\mathbf{L}^\pi\mathbf{y}$.

Second, the dynamics of the cell-averaged states $\langle \mathbf{x} \rangle_{\mathcal{C}_i}$ is governed by the quotient graph:

$$\frac{d\langle \mathbf{x} \rangle_{\mathcal{C}_i}}{dt} = -\mathbf{L}^\pi \langle \mathbf{x} \rangle_{\mathcal{C}_i} \quad \text{where} \quad \langle \mathbf{x} \rangle_{\mathcal{C}_i} := \mathbf{C}^+\mathbf{x}, \tag{166}$$

which follows from $\mathbf{C}^+\mathbf{L}\mathbf{x} = \mathbf{L}^\pi\mathbf{C}^+\mathbf{x}$. Thus, the cell-averaged dynamics is governed by a lower dimensional linear model, with dimensionality equal to the number of cells in the EEP (see Fig. 24B).

Third, the results obtained for the autonomous dynamics with no inputs (156) can be equivalently rephrased for the system with a bounded input $\mathbf{u}(t)$:

$$\dot{\mathbf{x}} = -\mathbf{L}\mathbf{x} + \mathbf{u}(t). \tag{167}$$

In particular, similarly to (165), we also have cell invariance under inputs: if we apply an input consistent with the cells of an EEP (i.e., $\mathbf{u}(t) = \mathbf{C}\mathbf{v}(t), \mathbf{v}(t) \in \mathbb{R}^c$), the nodes inside each cell remain identical for all times.

We remark that, while we here have focussed on the implications of an EEP for consensus dynamics, invariant partitions like the EEP can also play a similar role for other linear and nonlinear dynamics.

Before closing this section, let us pause briefly to clarify the difference between the presence of an EEP in a network, and the presence of a time-scale separation in the system. Observe that in the context discussed here, both of these concepts can be related to the notion of an invariant subspace in the dynamics. Note that if we have an EEP there is a set of eigenvectors that are exactly constant on each cell in the graph. These eigenvectors may be associated to any eigenvalue of the graph, whether it is fast or slow. We may summarize this on an intuitive level by saying that for an EEP the 'shape' of the eigenvectors with respect to the cells is important, but the eigenvalues are irrelevant. Note that this in contrast to the notion of time-scale separation discussed above. There the defining criterion were the eigenvalues, or more precisely a gap between the eigenvalues, separating them into slow and fast eigenvalues. In our example in Figure 23 above, the associated eigenvectors were indeed approximately constant on each cell (each block of nodes) as well, but not exactly. This indicates that the notion of an EEP and that of a time-scale separation are not mutually exclusive. We may indeed have a EEP in which the set of eigenvectors corresponding to the cells are precisely the slowest eigenmodes, but this does not have to be the case. Conversely the eigenvectors corresponding to the slowest time-scales do not have to be exactly constant on every cell.

Ex.VII.3 : Write a code to reproduce the numerical results of Figure 24.

VIII. DYNAMICS I: RANDOM WALKS

Consensus dynamics has important applications, in connection to the fields of decentralized algorithms and synchronization. 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 settings, random walks can be seen as a dual process of the consensus dynamics described in the previous section.

A. Discrete-time random walks on networks

Let us consider a walker diffusing on an undirected network. At each step, the walker located on a node selects one link connected to the node at random and jumps to an adjacent node. This process is equivalent to a Markov chain (Section III H) and is described by the $N \times N$ transition matrix T (Eq. (75)). The probability that the walker visits the ith node after t steps, $p_i(t)$, is given in by Eq. (79), where $p(t) = (p_1(t) \ \cdots \ p_N(t))$.

The solution, Eq. (79), involves products of matrices and can be simplified with the use of a graph Fourier transform. The underlying idea is to decompose the signal in an adequate base of vectors, such that the matrix products take the form of algebraic products for amplitudes 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 $T_{ij} = A_{ij}/k_i$, representing the probability that the walker transits from the ith node to the jth node. The transition matrix, T, is in general asymmetric, except if the underlying graph is regular. Nonetheless, its spectral properties can be directly derived from those of the symmetric matrix

$$\tilde{A}_{ij} = \frac{A_{ij}}{\sqrt{k_i k_j}}, \tag{168}$$

whose properties are essentially equivalent to those of the normalised Laplacian (Eq. (104)). By applying the decomposition given by Eq. (69) to $\tilde{A}$, we obtain

$$\tilde{A}_{ij} = \sum_{\ell=1}^{N} \lambda_\ell u_\ell u_\ell^\top, \tag{169}$$

where λ_ℓ is the ℓth eigenvalue of $\tilde{A}$ and u_ℓ is the normalised eigenvector such that $\langle u_\ell, u_{\ell'} \rangle = \delta_{\ell\ell'}$.

Because $T_{ij} = \sqrt{k_j}\tilde{A}_{ij}/\sqrt{k_i}$, i.e., $T = D^{-1/2}\tilde{A}D^{1/2}$, where D is the $N \times N$ diagonal matrix whose (i, i) element is equal to k_i $(1 \le i \le N)$, the ℓth left and right eigenvectors of T are given by

$$u_\ell^{\mathrm{L}} = \left((u_\ell)_1\sqrt{k_1} \ \cdots \ (u_\ell)_N\sqrt{k_N}\right), \tag{170}$$

$$u_\ell^{\mathrm{R}} = \left((u_\ell)_1/\sqrt{k_1} \ \cdots \ (u_\ell)_N/\sqrt{k_N}\right)^\top, \tag{171}$$

respectively. Equation (170) is verified by

$$\left(u_\ell^\top D^{1/2}\right) T = \left(u_\ell^\top D^{1/2}\right)\left(D^{-1/2}\tilde{A}D^{1/2}\right) = \left(u_\ell^\top \tilde{A}\right) D^{1/2} = \lambda_\ell u_\ell^\top D^{1/2}, \tag{172}$$

which implies $u_\ell^{\mathrm{L}} = u_\ell^\top D^{1/2}$ with eigenvalue λ_ℓ. Equation (171) is verified by

$$T\left(D^{-1/2}u_\ell\right) = \left(D^{-1/2}\tilde{A}D^{1/2}\right)\left(D^{-1/2}u_\ell\right) = D^{-1/2}\left(\tilde{A}u_\ell\right) = \lambda_\ell D^{-1/2}u_\ell, \tag{173}$$

which implies $u_\ell^{\mathrm{R}} = D^{-1/2}u_\ell$ with eigenvalue λ_ℓ.

Using Eq. (71), we obtain

$$\begin{aligned} T^t &= \left(D^{-1/2}\tilde{A}D^{1/2}\right)^t = D^{-1/2}\tilde{A}^t D^{1/2} \\ &= D^{-1/2}\sum_{\ell=1}^{N}\lambda_\ell^t u_\ell u_\ell^\top D^{1/2} \\ &= \sum_{\ell=1}^{N}\lambda_\ell^t u_\ell^{\mathrm{R}} u_\ell^{\mathrm{L}}. \end{aligned} \tag{174}$$

Therefore,

$$p(t) = p(0)T^t = \sum_{\ell=1}^{N}\lambda_\ell^t u_\ell^{\mathrm{L}}\langle p(0), u_\ell^{\mathrm{R}}\rangle. \tag{175}$$

In Eq. (175), $a_\ell(0) \equiv \langle p(0), u_\ell^{\mathrm{R}}\rangle$ is the projection of the initial condition to the ℓth eigenmode. Equation (175) indicates that the state of the random walk after t steps is given by a linear combination of the eigenmodes as follows:

$$p_i(t) = \sum_{\ell=1}^{N} a_\ell(t)(u_\ell^{\mathrm{L}})_i, \tag{176}$$

where

$$a_\ell(t) = \lambda_\ell^t a_\ell(0). \tag{177}$$

By definition, the graph Fourier transform maps a vector defined on the nodes, i.e., $p(t)$, to a vector of amplitudes of the eigenvectors $(a_1(t)\ \cdots\ a_N(t))$. It is a generalisation of the standard Fourier transform to topologies different from lattices (e.g., one-dimensional chain, two-dimensional square grid), where the eigenvectors of the transition matrix play the role of e^{ikx}. Graph Fourier transforms can also be defined based on other matrices describing the graph, e.g., the adjacency matrix. The choice of matrix and its associated base of eigenvectors are motivated by their adequacy for the problem at hand.

As mentioned in Section III H, the eigenvalues λ_ℓ of the transition matrix are in the interval $[-1, 1]$. The mode with $\lambda_\ell = 1$ corresponds to the stationary density, and we thus write $u_\ell^{\mathrm{L}} = p^*$. The right eigenvector that corresponds to this mode is $u_\ell^{\mathrm{R}} \propto (1,\ \ldots\ ,1)^\top$. All modes for which $-1 < \lambda_\ell < 1$ decay to 0. The eigenvalue $\lambda_\ell = 1$ is the largest-magnitude eigenvalue, and the Perron–Frobenius theorem guarantees that all elements of u_ℓ^{L} and u_ℓ^{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. In directed networks, the eigenvalues satisfy $|\lambda_\ell| \le 1$. When $|\lambda_\ell| < 1$ holds for all but one eigenvalue, which is the case except for directed variants of multipartite graphs with an even number of components, the mode with $\lambda_\ell = 1$ corresponds to the stationary density. In this case, we obtain $u_\ell^{\mathrm{L}} = p^*$ and $u_\ell^{\mathrm{R}} \propto (1,\ \ldots\ ,1)^\top$. Again, the Perron–Frobenius theorem guarantees that all elements of u_ℓ^{L} are positive.

By letting $n \to \infty$ in Eq. (175), we obtain $p^* = u_{\max}^{\mathrm{L}}\langle p(0), u_{\max}^{\mathrm{R}}\rangle$, where the subscript "max" indicates the mode corresponding to the dominant eigenvalue (which is equal to 1). Because $u_{\max}^{\mathrm{R}} \propto (1,\ \ldots\ ,1)^\top$, it follows that $\langle p(0), u_{\max}^{\mathrm{R}}\rangle = 1$ regardless of the initial condition $p(0)$. This is consistent with the fact that $u_{\max}^{\mathrm{L}}$ gives the stationary density. By letting n be large but finite, we obtain

$$p(n) \approx u_{\max}^{\mathrm{L}}\langle p(0), u_{\max}^{\mathrm{R}}\rangle + \lambda_2^n u_2^{\mathrm{L}}\langle p(0), u_2^{\mathrm{R}}\rangle\,, \tag{178}$$

where λ_2 is the second-largest (in magnitude) eigenvalue of T. In deriving Eq. (178), we only kept two terms, because $|\lambda_\ell|^n \ll |\lambda_2|^n$ for all eigenvalues λ_ℓ with $\ell > 2$, assuming that $|\lambda_\ell| < |\lambda_2|$ (where $\ell \in \{3, \ldots, N\}$). Equation (178) indicates that 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 (i.e., a small-magnitude for λ_2) entails fast relaxation.

The "Cheeger inequality" gives useful bounds on λ_2. The "Cheeger constant", which is also called "conductance", is defined by

$$h = \min_S \left\{ \frac{(\text{number of edges that connect } S \text{ and } \overline{S})}{\min\{\mathrm{vol}(S), \mathrm{vol}(\overline{S})\}} \right\}, \tag{179}$$

where S is a set of nodes in a network, $\overline{S}$ is the complementary set of the nodes (i.e., $S \cap \overline{S} = \emptyset$ and $S \cup \overline{S}$ is the complete set of the N nodes), and $\mathrm{vol}(S) \equiv \sum_{i=1; v_i \in S}^{N} s_i$. In the minimization in Eq. (179), we seek a bipartition of a network such that the two parts are the most sparsely connected. (In other words, we want a minimum cut.) The denominator in the right-hand side of Eq. (179) prevents the selection of a very uneven bipartition, which would easily yield a small value for the numerator. The Cheeger inequality is

$$\frac{h^2}{2} < 1 - |\lambda_2| \le 2h\,, \tag{180}$$

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.

A fact related to the relaxation time is that the power method is a practical method to calculate the stationary density of an RW in a directed network. Suppose that we start with an arbitrary initial vector $p(0)$, excluding one that is orthogonal to p^*, and repeatedly left-multiply it by T. After many iterations, we obtain an accurate estimate of p^*. Because any $p(0)$ that is orthogonal to p^* includes a negative entry, one can start iterations with any probability vector $p(0)$. In practice, one may have to normalize $p(n)$ after each iteration (or after some number of iterations) to avoid the elements of $p(n)$ becoming too large or small.

B. 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 Section VIII A. In this section, we start by looking at discrete-time random walk on directed networks, which is a representative Markov chain (Section III H).

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

$$T_{ij} = \frac{A_{ij}}{k_i^{\text{out}}}. \tag{181}$$

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

The time evolution of the density of the random walk is driven by Eq. (77). The stationary density given by Eq. (80) essentially defines the PageRank. The PageRank states that node v_i is important if v_i receives many links, the links entering v_i emanate from important nodes, and a node v_j sending a directed link 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 $p^* = (p_1^* \ \cdots \ p_N^*)$, $p_i^* = \lim_{t\to\infty} p_i(t)$ $(1 \le i \le N)$ 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 (Section III H). 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

$$p_i(t+1) = \alpha \sum_{j=1}^{N} p_j(t) T_{ji} + (1-\alpha) u_i, \tag{182}$$

where the preference vector $(u_1 \ \cdots \ u_N)$, subject to the constraint $\sum_{i=1}^{N} u_i = 1$, determines the probability with which a walker teleports to the ith node when it does. The probability of teleportation is equal to $1-\alpha$. In the case of web browsing, teleportation is interpreted as a jump to a new webpage without following a hyper-link. In general, if $u_i \neq 0$ $(1 \le i \le N)$, any $0 \le \alpha < 1$ makes the altered random walk given by Eq. (182) ergodic such that it converges to a unique stationary state. Use of a small α value makes the convergence to the stationary state faster and numerically stable, but waters down the effect of the network structure encoded in matrix T. A rule of thumb is to set α close to unity, in order to minimise the effect of teleportation, but not too close. A common choice is $\alpha = 0.85$ and $u_i = 1/N$ $(1 \le i \le N)$.

The stationary state of Eq. (182) is formally given by

$$p_{i;\alpha}^* = (1-\alpha) \sum_{j=1}^{N} u_j \left[(I - \alpha T)^{-1} \right]_{ji}, \tag{183}$$

where the dependence of the stationary density on α has been made explicit. This solution can be Taylor expanded in terms of α to yield

$$p_{i;\alpha}^* = u_i + \sum_{\ell=1}^{\infty} \alpha^\ell \sum_{j=1}^{N} u_j \left(T_{ji}^\ell - T_{ji}^{\ell-1} \right). \tag{184}$$

This expression clearly shows the non-local nature of the PageRank because it is made of walks of all length ℓ. A large α value gives a high credit to long walks. As in the case of the Katz centrality, the stationary density may radically change when α is modified. This dependence is clear when rewriting Eq. (184) with $u_i = 1/N$ in the following form:

$$p_{i;\alpha}^* = \frac{1}{N} + \sum_{\ell=1}^{\infty} \frac{\alpha^\ell}{N} \sum_{j,j'=1}^{N} \left(\frac{k_{j'}^{\text{in}} - k_j^{\text{out}}}{k_{j'}^{\text{in}}} \right) T_{jj'} T_{j'i}^{\ell-1}. \tag{185}$$

The leading contribution for small α makes the PageRank uniform, thereby making all nodes equivalent. Differentiation emerges when α is increased. The contribution of each walk of length ℓ is proportional to $k_{j'}^{\text{in}} - k_j^{\text{out}}$. Note that each term of the summation vanishes when the network is regular, i.e., $k_i^{\text{in}} = k_i^{\text{out}} = M/N$ $(1 \le i \le N)$, which yields $p^*_{i;\alpha} = 1/N$ $(1 \le i \le N)$ regardless of the α value.

We have implicitly assumed that each node has at least one outgoing link, i.e., $k_i^{\text{out}} > 0$ $(1 \le i \le N)$. If this condition is violated, the transition probability given by Eq. (181) is ill-defined. Therefore, we usually force a teleportation step with probability unity (not with probability $1-\alpha$) when a walker arrives at a dangling node, i.e., a node without outgoing links. Mathematically, we set $T_{ji} = u_i$ $(1 \le i \le N)$ for dangling nodes v_j.

Because the PageRank is the eigenvector corresponding to the largest eigenvalue of a positive matrix T', whose (i,j) element is given by $T'_{ij} = \alpha T_{ij} + (1-\alpha)u_j$, we can efficiently compute it using the power method (Section III G). The power method converges rapidly if the spectral gap of T' is large.

Ex.VIII.1: Write your own code to calculate the Pagerank of a directed network. Test on an example the dependency of Pagerank on the teleportation coefficient.

C. Mean first-passage and recurrence times

The previous sections have focused on the evolution of the probability density of an ensemble of walkers, but the theory of random walks allows to answer many more properties, in particular on the behaviour of one single walker. When does a random walker starting from a certain source node arrive at a target node for the first time? The answer to this question is known as the "first-passage time" (or "first-hitting time") if the source and target nodes are different and is known as the "recurrence time" (or the "first-return time") when the source and target nodes are identical. Let m_{ij} (with $i \neq j$) denote the mean first-passage time (MFPT) from node v_i to node v_j. The mean recurrence time is m_{ii}. For directed networks, we assume strongly connected networks throughout this section to guarantee that $m_{ij} < \infty$ (for $i, j \in \{1, \dots, N\}$).

General networks: Let's first consider some general results. The following identity holds :

$$m_{ij} = 1 + \sum_{\ell=1;\ell\neq j}^{N} T_{i\ell} m_{\ell j} \,. \tag{186}$$

In its first step, a random walker moves from node v_i to node v_ℓ, which produces the 1 on the right-hand side of Eq. (186). If $\ell = j$, then the walk terminates at v_ℓ, resulting in a first-passage time of 1. Otherwise, we seek the first-passage from node v_ℓ (with $\ell \neq j$) to node v_j. This produces the second term on the right-hand side. Note that Eq. (186) is also valid when $i = j$.

In matrix notation, we write Eq. (186) as

$$M = J + T(M - M_{\text{dg}}) \,, \tag{187}$$

where $M = (m_{ij})$, all of the elements of the matrix J are equal to 1, and M_{dg} is the diagonal matrix whose diagonal elements are equal to m_{ii}. By left-multiplying Eq. (187) by p^* and using $p^* J = (1, \dots, 1)$ and $p^* T = p^*$, we obtain the mean recurrence time

$$m_{ii} = \frac{1}{p_i^*} \,. \tag{188}$$

Equation (188) is called "Kac's formula".

There are several different ways to evaluate the MFPT m_{ij} (with $i \neq j$), and it is insightful to discuss different approaches.

One method is simply to iterate Eq. (186).

A second method to calculate the MFPT, for a given j, is to rewrite Eq. (186) as

$$\overline{m}^{(j)} = 1 + \overline{T}^{(j)} \overline{m}^{(j)} \,, \tag{189}$$

where $\overline{m}^{(j)} = (m_{1j}, \dots, m_{j-1,j}, m_{j+1,j} \dots, m_{Nj})^\top$ and $1 = (1, \dots, 1)^\top$ are $(N-1)$-dimensional column vectors and $\overline{T}^{(j)}$ is the $(N-1)\times(N-1)$ submatrix of T that excludes the jth row and jth column. The formal solution of Eq. (189) is

$$\overline{m}^{(j)} = \left(\overline{L}^{(j)}\right)^{-1} \overline{D}^{(j)} 1 \,, \tag{190}$$

where $\overline{D}^{(j)}$ is the submatrix of D that excludes the jth row and jth column and $\overline{L}^{(j)} = \overline{D}^{(j)} - \overline{A}^{(j)}$, where $\overline{A}^{(j)}$ is the submatrix of A that excludes the jth row and jth column. The matrix $\overline{L}^{(j)}$ is sometimes called a "grounded Laplacian matrix" (although it is not a Laplacian matrix), and it is invertible because we assumed strongly connected networks. One can derive and solve Eq. (190) separately for each j.

A third method to examine the MFPT is to estimate m_{ij} using a mean-field approximation. Regardless of the source node v_i, the target node v_j is reached with an approximate probability of p_j^* in each time step. Therefore,

$$m_{ij} \approx \sum_{n=1}^{\infty} n p_j^* (1 - p_j^*)^{n-1} = \frac{1}{p_j^*} = m_{jj} \,. \tag{191}$$

Equation (191) is a rather coarse approximation, and m_{ij} can deviate considerably from $m_{jj} = 1/p_j^*$. More sophisticated mean-field approaches can likely do better, especially for networks with structures that are well-suited to the employed approximation.

There have been many studies of MFPTs for various network models using both analytical and numerical approaches. We will discuss some examples of undirected and unweighted networks. We focus mainly on the MFPT between different nodes, although it is of course also interesting to calculate recurrence times.

Regular networks: For a complete graph, m_{ij} (with $i \neq j$) is independent of i and j because of the symmetry of the network. Therefore, Eq. (186) reduces to

$$m_{ij} = \frac{1}{N-1} + \frac{N-2}{N-1}(1 + m_{ij}), \tag{192}$$

which yields $m_{ij} = N - 1$ for $i \neq j$. Kac's formula [see Eq. (188)] implies that $m_{ii} = N$.

For regular lattices $\mathbb{Z}^d$ of any dimension d, Eq. (188) implies that $m_{ii} \propto N$ because $p_i^* \propto k_i = 2d$ for any i. Define $m_{\bullet j}$ to be the MFPT averaged over all source nodes v_i ($i \neq j$). For $\mathbb{Z}^d$, it satisfies the scalings $m_{\bullet j} \propto N^2$ for $d = 1$, $m_{\bullet j} \propto N \ln N$ for $d = 2$, and $m_{\bullet j} \propto N$ for $d = 3$.

Erdős–Rényi (ER) random graphs: Consider an ER random graph $G(N, p)$, where p denotes the (independent) probability that each node pair has an edge. Assuming that the mean degree $\langle k \rangle$ is kept constant (i.e., $p = \langle k \rangle/(N-1) \propto 1/N$), we obtain $m_{ii} \propto N$ and $m_{ij} \propto N^{3/2}$ (with $i \neq j$) as $N \to \infty$ for the "giant component" (i.e., a largest connected component that scales linearly with the number N of network nodes as $N \to \infty$). Now suppose that we assume instead that $p > \ln N/N$, so that all nodes belong to a single component (in the $N \to \infty$ limit) and thus m_{ij} (for $i, j \in \{1, \ldots, N\}$) is well-defined. It then follows that m_{ij} averaged over all source and target nodes is equal to $N - 1$, independently of p. In other words, for a sufficiently dense ER random graph, the MFPT is the same as that for the complete graph. The MFPT is much longer for directed ER graphs than for undirected ones, because random walkers do not backtrack on directed networks.

Ex.VIII.2: By performing simulations of a random walker of a graph, verify numerically Kac's formula.

D. Application: Respondent-driven sampling

One often is interested in estimating a population mean of certain quantities, such as the fraction of infected individuals, the fraction of people who have a particular opinion, or demographics such as age. If a population is large, which is typical in the context of social surveys, it is impossible to record all individuals. In such situations, a common challenge is how to sample individuals in as unbiased manner as possible.

"Respondent-driven sampling" (RDS) is a popular sampling method that uses edge-tracing in a social network. In RDS, one starts from a seed individual (i.e., a seed node). The seed individual recruits his/her neighbors to a survey by passing a coupon to each of them. The successfully recruited individuals then participate in the survey and in turn pass coupons to their neighbors who have not yet participated. To try to promote participation, individuals who participate are rewarded financially. One takes a weighted mean of the samples to derive an estimate of the quantity of interest (e.g., mean age of a population).

It is necessary to take a weighted mean because the probability of being recruited depends on the position of a person in a network. The so-called "RDS II estimator" is an efficient and realistic estimator. Consider the case in which each respondent passes a single coupon to one of its uniformly randomly selected neighbors. One can then describe the recruitment process as a DTRW if one allows sampling with replacement for simplicity (i.e., if the same individual can be sampled more than once). Again for simplicity, let's also assume that the network is undirected and unweighted. The essential idea of the RDS II estimator is that one should discount the effect of a sampled node v_i by a factor of its degree k_i, because v_i is visited with probability $p_i^* \propto k_i$. Note that respondents have to report k_i to be able to calculate this estimator, although empirically it is difficult to accurately collect the k_i values of respondents.

We are interested in estimating the mean $\langle y \rangle$ of a quantity y_i assigned to node v_i. We denote the set of sampled nodes by S and the number of samples (i.e, the size of S) by N_S. The estimator $\langle \hat{y} \rangle$ of $\langle y \rangle$ is

$$\langle \hat{y} \rangle = \frac{1}{N_S} \sum_{v_i \in S} \frac{y_i}{N \hat{p}_i^*}, \tag{193}$$

where $\hat{p}_i^*$ is the estimate of the stationary density p_i^*. We set the discount factor on the right-hand side of Eq. (193) to be $N\hat{p}_i^*$, because it is normalized so that $\langle N\hat{p}_i^* \rangle = 1$. By assuming that we do not have access to the mean degree $\langle k \rangle$ of the entire network, we estimate it by calculating

$$\hat{p}_i^* = \frac{k_i}{N \langle \hat{k} \rangle}, \tag{194}$$

where $\langle \hat{k} \rangle$ is an estimate of $\langle k \rangle$. We use

$$\langle \hat{k} \rangle = \frac{\sum_{v_i \in S} \frac{k_i}{N p_i^*}}{\sum_{v_i \in S} \frac{1}{N p_i^*}} = \frac{N_S}{\sum_{v_i \in S} (k_i)^{-1}}. \tag{195}$$

Combining Eqs. (193), (194), and (195) yields

$$\langle \hat{y} \rangle = \frac{\sum_{v_i \in S} (k_i)^{-1} y_i}{\sum_{v_i \in S} (k_i)^{-1}}. \tag{196}$$

The estimated quantity y can be either continuous-valued or discrete-valued. Alternatively, one can estimate the proportion of nodes P_A that have a discrete type A (e.g., an infected state) by setting y_i to the indicator function (i.e., $y_i = 1$ when v_i is of type A and $y_i = 0$ otherwise). In this case, we obtain

$$\hat{P}_A = \frac{\sum_{v_i \in A \cap S} (k_i)^{-1}}{\sum_{v_i \in S} (k_i)^{-1}}. \tag{197}$$

Note that, even if one controls for the effect of p_i^* in this manner, the estimator $\langle y \rangle$ is statistically biased in practice. For example, the estimator is inaccurate when networks have community structure or have multiple connected components. Additionally, different techniques are required for directed networks, because Eq. (194) (or, more succinctly, $p_i^* \propto k_i$) does not hold for directed networks. Furthermore, actual sampling trajectories are non-backtracking, and one can incorporate this feature into RDS estimators.

E. Continous-Time Random Walks

Before proceeding to our discussion of other types of dynamical processes on networks, it is useful to pause and the differences and connections between update rules in discrete-time versus continuous-time dynamics. The results of this section have, so far, considered a discrete-time rule where an ensemble of walkers all move synchronously at discrete times. In contrast, the consensus dynamics of the previous section was implicitly implemented in a continuous-time setting, as can be seen from the differential equations describing the process. In this section, we will discuss more in detail ways to make a discrete-time random walk (DTRW) continuous. Note that the continuous-time consensus time can also be defined in discrete-time setting, in which case it is usually called De Groot model.

Continuous-time Random Walks (CTRWs) on networks have two main components: the statistics of a walker's trajectory in terms of the number of steps and the statistics of the times at which events take place. By combining these two components, one can specify the probability that a random walker visits a specified node at a specified time. For RWs on networks, the dynamics of a walker are affected not only by the statistical properties of temporal events, but also by the type of network unit in which a temporal process is defined. In the following, we distinguish between node-centric CTRWs and edge-centric CTRWs. For dynamical processes in general, there are often substantial differences between node-based dynamics and edge-based dynamics, so it is crucial to distinguish between these situations.

In a CTRW, a walker waits until the next move for a time τ, where τ is a random variable. For the sake of simplicity, we consider a scenario in which moves occur as independent Poisson processes. In other words, τ is distributed according to the exponential distribution with parameter λ. We can safely normalize λ to 1, because λ only sets the time scale. In a node-centric CTRW, a walker moves from node v_i when it becomes active, and it selects one of the out-neighbors, which we denote by v_j, as the destination with a probability proportional to A_{ij} [see Fig. 26(a)]. This assumption is the same as that for a DTRW.

The master equation for the node-centric CTRW on a network is

$$\frac{\mathrm{d}p(t)}{\mathrm{d}t} = p(t)(-I+T) = -p(t)D^{-1}L\,, \tag{198}$$

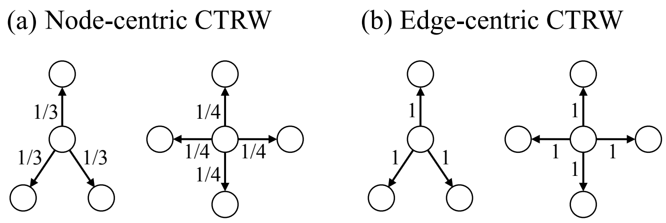

FIG. 26. Schematic of two types of continuous-time random walks (CTRWs) on networks: (a) a node-centric CTRW and (b) an edge-centric CTRW. In each case, a walker is visiting either a degree-3 node or a degree-4 node in a network, which we assume is unweighted for simplicity. We show the transition rates for each edge. In panel (a), the walker travels at a unit rate and moves to one of its out-neighbors with equal probability for each choice. Therefore, the transition rate for each edge is the reciprocal of the out-degree of the node that the walker is visiting. In panel (b), however, the transition rate on each edge is equal to 1. Therefore, on average, a walker visiting the node with out-degree 4 leaves the node earlier than a walker visiting the node with out-degree 3.

where

$$L \equiv D - A \tag{199}$$

is the ("combinatorial") "Laplacian matrix" of the network. The process is driven by the "random-walk normalized Laplacian"

$$L' \equiv D^{-1}L = I - T\,. \tag{200}$$

That is, $(L')_{ij} = \delta_{ij} - (A_{ij}/s_i^{\mathrm{out}})$. If we examine the node-centric CTRW in terms of the number n of moves, the trajectories are statistically the same as those of the DTRW in Eq. (77). Consistent with this observation, node-centric CTRWs are also called the "continuization" of the DTRW. In particular, the stationary density of the node-centric CTRW is the same as that of the DTRW. By setting the left-hand side of Eq. (198) to 0, we obtain $p^*(-I+T) = 0$, so that $p^* = p^*T$. If the network is undirected, $p_i^* = s_i / \sum_{\ell=1}^N s_\ell$. Node-centric CTRWs have been used in, for example, some empirical-data-driven metapopulation disease-spreading models. In those models, a network consists of subpopulations of individuals, and individuals move from one subpopulation to another through a mobility rule. The simplest mobility rule, which has been used widely, is that individuals move according to a Poissonian node-centric CTRW.

Another type of CTRW is an edge-centric CTRW, in which each edge (rather than a node) is activated independently according to a renewal process [see Fig. 26(b)]. By definition, once an edge is activated, it becomes available, and a random walker can use it to move to the associated adjacent node. This RW model has also been called the "fluid model".

When a Poisson process with a rate proportional to the edge weight is assigned independently to each edge, the master equation is

$$\frac{\mathrm{d}p(t)}{\mathrm{d}t} = p(t)(-D + A) = -p(t)L\,. \qquad (201)$$

The Poissonian edge-centric CTRW is associated with the unnormalized (i.e., combinatorial) Laplacian L. Equation (201) implies that the transition rate at node v_i is equal to s_i^{out}. A walker leaves a node with a large out-strength (such a node may be a network "hub") more quickly than a node with a small out-strength. This situation contrasts with the aforementioned node-centric CTRW, for which the transition rate of a walker is the same for all nodes.

The stationary density for Eq. (201) is

$$p^* L = 0\,. \qquad (202)$$

Equation (202) is equivalent to $p_i^* s_i^{\mathrm{out}} - \sum_{j=1}^{N} p_j^* A_{ji} = 0$ (for $i \in \{1, \ldots, N\}$), which indicates that the in-flow of the probability (i.e., $\sum_{j=1}^{N} p_j^* A_{ji}$) and the out-flow of the probability (i.e., $p_i^* s_i^{\mathrm{out}}$) are balanced at each node. Equation (202) also indicates that p^* is a left eigenvector of L with eigenvalue 0. In connected undirected networks, the 0 eigenvalue, which we denote by $\lambda_1 = 0$, is an isolated eigenvalue. Its associated eigenvector is

$$p^* = \frac{1}{N}(1\,, \ldots,\, 1)\,. \qquad (203)$$

Eq.201 shows that the edge-centric CTRW and the consensus dynamics are dual to each other. In the case of undirected networks, their dynamics is equivalent.

CTRWs on networks can also been used in situations when the inter-event times are not generated by a Poisson process, as we discussed here, but by renewal processes with general inter-event time distributions. In that case, the master equations become integro-differential and can be studied efficiently in the Laplace domain. See for instance: Generalized master equations for continuous-time random walks, VM Kenkre, et al., Journal of Statistical Physics 9 (1), 45-50, 1973 for classical lattices and Generalized master equations for non-Poisson dynamics on networks, T Hoffmann, et al., Physical Review E, 2012 on general networks.

IX. RANDOM WALKS TO REVEAL NETWORK STRUCTURE

As we have seen from the previous sections, particular network structure can have notable impact on a dynamics acting on a network. What about the converse? For a given dynamics, can we harness these effects (e.g. the time-scale separation of a dynamics), to infer something about dynamically important modular structure in the network? As we will see below, several network methods can be designed from a dynamical perspective and, in particular, based on random walk processes.

A. Markov stability

A deeper understanding of modularity, as well as rectification of some of its limitations, is gained from a random walk perspective. Modularity is related to Markov stability, a quality function representing the tendency for a random walker to remain within a community for a long time. More precisely, the Markov stability of a partition is defined as the probability that the walker is in the same community initially and after time t in the stationary state. Markov stability can be defined for different dynamical processes, each one giving rise to a different quality function and, in principle, to a different optimal partition of a network.

We focus on a node-centric continuous-time random walk, where the transition rate for the walker is set to unity and where the network is undirected for simplicity. Denote by p_i the probability that the random walker visits node v_i, omitting the time variable t. The master equation for the random walk process is given by

$$\frac{\mathrm{d}p}{\mathrm{d}t} = -pL^{'}, \qquad (204)$$

where $p = (p_1, \ldots, p_N)$ and L is the random walk normalised Laplacian matrix whose elements are given by

$$L_{ij}^{'} = \delta_{ij} - \frac{A_{ij}}{k_i}, \qquad (205)$$

as before. It should be noted that the probability with which a walker at the ith node moves to the jth node is given by $T_{ij} = A_{ij}/k_i$. The steady state is given by the left eigenvector of $L^{'}$ corresponding to the zero eigenvalue. Assuming that the network is connected, direct substitution verifies that the stationary density is given by

$$p_i^* = \frac{k_i}{\sum_{\ell=1}^{N} k_\ell} = \frac{k_i}{2M} \quad (1 \le i \le M), \qquad (206)$$

where the last equality holds true owing to the handshaking lemma (Eq. (89)).

To define Markov stability, denoted by $R(t)$, consider a pair of nodes v_i and v_j belonging to the same community. Equation (204) implies that, at stationarity, the joint probability that the walker visits v_i at time 0 and v_j at time t is given by $p_i^*(e^{-tL'})_{ij}$. As in the case of modularity, this quantity has to be compared with an appropriate null model. For Markov stability, the null model is given by the joint probability of finding a first walker on v_i at time 0 and a second independent walker on v_j at time t, i.e., $p_i^* p_j^*$. The null model is also regarded as the probability $p_i^*(e^{-tL'})_{ij}$ as $t \to \infty$, because the position of a walker at large times is independent of its initial position. By comparing the actual and independent cases, we define

$$R(t) = \sum_{i,j=1}^{N} \left[\left(p_i^* e^{-tL'} \right)_{ij} - p_i^* p_j^* \right] \delta(g_i, g_j). \tag{207}$$

Markov stability differs from modularity in several ways. First, it involves the exponential of the Laplacian, $e^{-tL'} = I - tL' + t^2 L'^2/2 + \cdots$ and thus combines paths of all lengths between two nodes. Second, Markov stability naturally incorporates a resolution parameter t. A larger value of t gives a more weight to longer paths in the exponential, corresponding to an exploration of the network at a larger time scale. The time thus acts as a resolution parameter enabling us to zoom in and out to uncover the multi-scale structure of the network. The optimal partition based on Markov stability is made of less, larger communities when t is increased. Third, Markov stability is a method derived from flows of probability. It is not a combinatorial method like modularity, in which finite elements (links) are counted. For this reason, Markov stability can detect the impact that certain types of network structure may have on dynamical processes. Finally, Markov stability has desirable mathematical properties including connections to spectral graph theory. It is exactly optimised by the bipartition given by the signs of the Fiedler vector, i.e., the second dominant eigenvector of the Laplacian, when t is sufficiently large.

In practice, estimating the exponential of the Laplacian for a large network can be computationally expensive. Therefore, it is often preferable to study a linearised version of stability, defined by its Taylor expansion for small t. Using $(e^{-tL'})_{ij} \approx \delta_{ij} - tL'_{ij}$, Eq. (205), $p_i^* = k_i/2M$ and $p_j^* = k_j/2M$, we reduce Eq. (207) to

$$R(t) = \frac{1}{2M} \sum_{i,j=1}^{N} \left[tA_{ij} + (1-t)\delta_{ij} k_i + \frac{k_i k_j}{2M} \right] \delta(g_i, g_j). \tag{208}$$

Because $\sum_{i,j=1}^{N} (1-t)\delta_{ij} k_i \delta(g_i, g_j) = \sum_{i=1}^{N} k_i$ is independent of the partitioning of the network, maximisation of $R(t)$ is equivalent to maximisation of

$$Q(\gamma) = \frac{1}{2M} \sum_{i,j=1}^{N} \left(A_{ij} - \gamma \frac{k_i k_j}{2M} \right) \delta(g_i, g_j), \tag{209}$$

where we introduced the structural resolution parameter $\gamma = 1/t$, more commonly used than t. We remove the condition that t is small.

With $\gamma = 1$, Eq. (209) coincides with Eq. (148), such that modularity can be seen as a simple, approximate instance of Markov stability. A large value of γ emphasises the penalty in classifying nodes in the same group such that it yields a large number of communities. A small value of γ yields a small number of communities. Resolution parameter γ allows us to tune the characteristic size of the communities in the optimal partition, i.e., not using the characteristic size imposed by modularity maximisation. However, it also raises the important problem of finding a γ value consistent with natural scales of the network.

Ex IX.1: Propose and justify a generalization of Markov stability Eq.(207) in the case of directed networks.

B. Infomap

In this section, we sketch an alternative community detection method based on random walks, called the Infomap. The method, originally proposed for non-overlapping community structure, works for directed and weighted networks.

Imagine a random walk on a given network. If the network has community structure, the random walker would wander within a community for a long time before crossing a bridge to a different community. A straightforward way to describe the trajectory of the random walk is to write down the visited nodes in an ordered list, e.g., v_1, v_4, v_1, v_7, v_3, ... The amount of information required to express the trajectory is estimated as follows. We code each node into a finite binary sequence, i.e., a code word, and concatenate the code words. For example, if v_1, v_3, v_4, and v_7 are coded into 000, 010, 011 and 110, the aforementioned trajectory is coded into $000011000110010\cdots$. For unique decoding, the code has to be prefix-free. In other words, a code word must not be a prefix (i.e., initial segment) of another code word. For example, if v_1 and v_2 are coded into 000 and 0001, respectively, the code is not prefix-free because 000 is an initial segment of 0001.

The Huffman code is a prefix-free code that encodes symbols separately and generally yields short binary sequences to represent trajectories of the random walk. It assigns a short code word to a frequently visited node and vice versa. The mean code word length per step of the random walk is given by $\sum_{i=1}^{N} p_i^* L(i)$, where p_i^* is the stationary density of the random walk at node v_i and $L(i)$ is the length of the code word for node v_i.

When the symbols (v_i in our case) appear independently, the Huffman code often yields a code length that is close to the theoretical lower bound obtained by the Shannon entropy, which is

$$H = -\sum_{i=1}^{N} p_i^* \log p_i^* \tag{210}$$

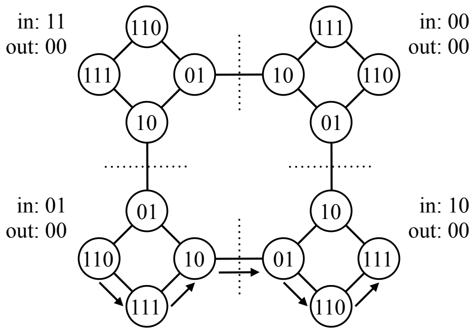

FIG. 27. Optimal partitioning according to Infomap and the resulting code words. This example is based on a demo applet available at Martin Rosvall's website http://www.mapequation.org/apps/MapDemo.html.

per step. However, the sequence of nodes is correlated in time because it is produced by the random walk. Then, an alternative coding scheme may lessen the mean code length. In particular, we can design a two-layered variant of the Huffman code to exploit the community structure of the network. Because there are less nodes in a community CM_i as compared to the entire network, we can express a trajectory within CM_i with a shorter, different Huffman code, which is local to CM_i. Based on this observation, we rebuild the Huffman code as follows.

1. When the walker enters community CM_i, a code word to represent this entry event is issued.

2. The walker wanders within CM_i. The trajectory of the walker during this period is encoded by concatenating the code words corresponding to the sequence of the visited nodes. The sequence of these code words is simply placed after the code word produced in the previous step (i.e., entry to CM_i). It should be noted that the intra-community code words make sense only within CM_i. A different community $\mathrm{CM}_{i'}$ ($i' \neq i$) may use the same code word as the one used within CM_i to represent a different node in $\mathrm{CM}_{i'}$.

3. The walker exits CM_i. This event is represented by a special code word, which is concatenated after the sequence of code words produced so far.

4. The exit from CM_i implies that the walker immediately enters a different community, CM_j. Therefore, a code word to notify that the walker has entered CM_j is issued. Then, the code words local to CM_j are used until the walker exits CM_j. We repeat this procedure.

Consider the example shown in Fig. 27. The optimal partitioning of this network obtained by Infomap is into four communities whose boundaries are shown by the dotted lines. The binary code word assigned to each node represents the local code within the corresponding community. Note that the code word uniquely determines a node within a community, but not across communities; the same code word is used in different communities. When the random walker enters or exits a community, the corresponding 'in' or 'out' code word is used, respectively. Therefore, the trajectory shown by the arrows in the figure is encoded into 0111011110001001110111. The first 01 indicates that the walk starts in the left-bottom community, and the 110 that follows indicates that the walk starts at the 110 node in this community. 0010 in the middle indicates that the walk exits this community (by the code word 00) and immediately enters the community to the right (by the code word 10).

In contrast to the original Huffman code, we have to invest $2N_{\mathrm{CM}}$ code words to mark the entry to and exit from a community. However, we can save the code length when the walker wanders in a community, which occupies a majority of steps. Overall, the mean code length is expected to be smaller with the two-layer code in the presence of community structure. In order to detect communities in practice, there is no need for devising the optimal code of a given partition. Infomap instead proceeds by optimising a quality function, called the map equation, which generalises Eq. (210). The resulting quality function provides a theoretical limit of how concisely we can specify a walk in the network using a given partition. The optimisation is then performed by a greedy algorithm similar to the one used for maximising modularity.

Ex IX.2: Read "Comparing clusterings - an information based distance, M Meila, 2017", and implement numerically a method to compare different partitions. Compare the partitions obtained by maximising modularity and the Map Equation on the network of your choice.

C. Similarity measures and Walktrap

When working with networks, many tasks can be simplified by defining a proper measure of distance, or similarity, between pairs of nodes. For instance, assume that labels are known for a fraction of the nodes of a graph, the problem of inferring unknown labels, i.e. the so-called node classification task, can be solved by using simple nearest neighbour techniques. Similarly, for link prediction, that is the task of predicting unknown links or links to appear in the future, a majority of approaches consist in finding the most similar nodes that are not yet connected. Similarly, for community detection, methods consist in performing a clustering of the similarity matrix, for instance by using k-means or hierarchical clustering. By definition, a similarity matrix S is an $n \times n$ matrix whose elements are positive real numbers, encoding the similarity between pairs of nodes. There exist many types of similarity matrices. The adjacency matrix itself is the simplest measure of similarity: two nodes are more similar if they are connected than if they are not, but it is a very coarse-grained measure, giving the same similarity to each pair of disconnected nodes, even if some may be only two edges apart and some much further away. In order to account for paths of different lengths, popular choices instead consider functions of the adjacency matrix, or of the Laplacian matrix.

In the case of community detection, an influential algorithm is Walktrap, where similarity is defined by means of DTRWs. Consider an undirected and unweighted network. The RW-based distance between two nodes, v_i and v_j is then defined as

$$r_{ij} = \sqrt{\sum_{\ell=1}^{N} \frac{(T^n_{i\ell} - T^n_{j\ell})^2}{k_\ell}}\,, \tag{211}$$

where n is the number of steps in a DTRW. The distance r_{ij} is small when a pair of random walkers — one starting from v_i and the other starting from v_j — visit each node with similar probabilities after n steps. The denominator k_ℓ discounts the fact that a walker visits v_ℓ with a probability proportional to k_ℓ at equilibrium. Note that n needs to be large enough for random walkers to be able to travel to any node. However, n should not be too large, because $\lim_{n\to\infty} T^n_{i\ell} = \lim_{n\to\infty} T^n_{j\ell} = p^*_\ell$ implies that r_{ij} is very close to 0 for all $i, j \in \{1, \ldots, N\}$ when n is large. We expect that a pair of nodes, v_i and v_j, that are separated by a small distance r_{ij} are likely to belong to the same community. Walktrap uses a standard agglomerative and hierarchical clustering algorithm on the distance matrix $r = (r_{ij})$. One starts from the partition composed of N single-node communities and joins a pair of communities (so-called "tentative communities") with the smallest distance, one pair at time, to produce a series of partitions until the entire network is in a single community. In the merging process, one measures the distance between two communities CM_c and $\mathrm{CM}_{c'}$ by the r_{ij} value, normalized in some way, between $v_i, v_j \in \mathrm{CM}_c \cup \mathrm{CM}_{c'}$. This agglomerative clustering algorithm is similar to a greedy algorithm to maximize modularity across partitionings with different numbers of communities. In Walktrap, one merges a pair of communities under the restriction that they can be merged only when they are adjacent to each other by at least one edge.

Another interesting approach of community detection based on similarities exploits the concept of mean first-passage time m_{ij} of a random walker and its symmetrization $m_{ij}+m_{ji}$ (the so-called "mean commute time"). The square root of the mean commute time has the desirable property of being a Euclidian distance between nodes. In this context, it is called the "Euclidian commute-time distance". It decreases when the number of paths between two nodes increases or when the length of any path between the two nodes decreases, and it can be derived from the pseudo-inverse of the combinatorial Laplacian matrix L.

D. Core–periphery structure

It is often insightful to decompose a network into one or more densely-connected cores along with sparsely-connected peripheral nodes. By definition, nodes in a core are heavily interconnected and also tend to be well-connected to peripheral nodes. By contrast, peripheral nodes are sparsely connected (or, ideally, not adjacent at all) to other peripheral nodes and tend to be adjacent predominantly to core nodes. This idea, whose intuition draws somewhat on the notion of pealing an onion (especially in the case of a single core), is also a mesoscale network structure, but it has a rather different character from community structure.

There is an RW-based algorithm to extract core–periphery structure from networks. The idea is that if a random walker is located at a peripheral node, it is very unlikely to visit another peripheral node in the next time step in a DTRW. One defines a "persistence probability" α_S for a set of nodes S by

$$\alpha_S = \frac{\sum_{i,j\in S} p^*_i T_{ij}}{\sum_{i\in S} p^*_i}\,, \tag{212}$$

where we recall that p^*_i is the stationary density at node v_i, and T_{ij} is the transition probability from v_i to v_j in a single move. Equation (212) is the steady-state probability that a DTRW starting from a node in S remains in S in the next time step. For an undirected network, we substitute $p^*_i = s_i / \sum_{\ell=1}^{N} s_\ell$ to reduce Eq. (212) to

$$\alpha_S = \frac{\sum_{i,j\in S} A_{ij}}{\sum_{i\in S} s_i}\,. \tag{213}$$

Ideally, one obtains $\alpha_S = 0$ for any set S of nodes that includes only peripheral nodes. This condition is trivially satisfied when S consists of a single node, and it becomes very difficult to satisfy as S becomes large. One possibility is to use the following greedy algorithm. Start from a node with the smallest total node strength $s^{\mathrm{in}}_i + s^{\mathrm{out}}_i$. If there are multiple such nodes, we select one of them uniformly at random. For undirected networks, this reduces to selecting a node with the minimum node strength. The set S is composed of a single node. One then adds one node to the set S so that adding this node yields the smallest value of α_S. Again, if there are multiple candidate nodes, we break the tie by selecting one of them uniformly at random. One continues this procedure and sequentially adds nodes to try to keep α_S small. One then assigns each node v_i a coreness value of α_i, which one sets as the value of α_S when v_i is added. Nodes with larger values of α_i are deeper into a network core. One also defines a network's "α-periphery" as the set of nodes that satisfy $\alpha_i \le \alpha$. Although the algorithm has randomness in it because of the tie-breakers, randomness has been shown to have negligible effects on the results for empirical networks.

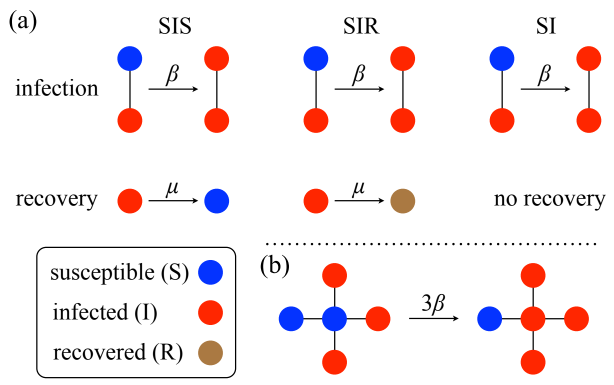

FIG. 28. (a) Infection and recovery rates in three epidemic process models. (b) Infection rate when a susceptible node is surrounded by one susceptible node and three infected nodes.

X. 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 considering two types of models on networks: meta-population models and spreading on contact networks.

A. Models of epidemic processes

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, i.e., 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 in Fig. 28(a).

The SIS model assumes two processes. When a susceptible node interacts with an infected node, the susceptible node contracts infection to transit to 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 $k_{\rm I}$ infected neighbours, the transition rate is equal to $k_{\rm I}\beta$ (see Fig. 28(b) for an example of $k_{\rm I} = 3$). An infected node recovers at rate μ irrespectively of the states of the neighbours. Once an infected node recovers, it transits 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, i.e., the complete graph, where each node 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 in the thermodynamic limit (i.e., $N \to \infty$) are given by

$$\frac{\mathrm{d}S(t)}{\mathrm{d}t} = -\beta I(t)S(t) + \mu I(t) \tag{214}$$

and

$$\frac{\mathrm{d}I(t)}{\mathrm{d}t} = \beta S(t)I(t) - \mu I(t). \tag{215}$$

The first terms in Eqs. (214) and (215) represent the fact that each of the $NS(t)$ susceptible nodes is infected at a total rate of $\beta I(t)$. Addition of Eqs. (214) and (215) yields

$$\frac{\mathrm{d}}{\mathrm{d}t}\left\{S(t) + I(t)\right\} = 0. \tag{216}$$

Therefore, $S(t) + I(t) = S(0) + I(0) = 1$ such that the total number of nodes is conserved.

In the equilibrium, setting the right-hand side of Eq. (215) to zero yields

$$I^*(\beta S^* - \mu) = 0, \tag{217}$$

where S^* and I^* are the fraction of susceptible and infected nodes in the equilibrium, respectively. If infection persists in the population in the equilibrium, by applying $I^* \neq 0$ and $S^* = 1 - I^*$ to Eq. (217), we obtain

$$I^* = 1 - \frac{\mu}{\beta}. \tag{218}$$

Equation (218) 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

$$\frac{\beta}{\mu} > 1. \tag{219}$$

We say that the epidemic threshold in terms of β/μ is equal to unity in the well-mixed population. 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

$$\frac{\mathrm{d}S(t)}{dt} = -\lambda I(t)S(t), \tag{220}$$

$$\frac{\mathrm{d}I(t)}{dt} = \lambda I(t)S(t) - \mu I(t), \tag{221}$$

$$\frac{dR(t)}{dt} = \mu I(t). \tag{222}$$

Summation of Eqs. (220), (221) and (222) confirms that the total number of nodes is conserved, i.e., $S(t) + I(t) + R(t) = 1$ for all t. When $\mathrm{d}I(t)/\mathrm{d}t > 0$ at $t = 0$, the number of infectious nodes first increases to a macroscopic (i.e., $O(N)$) number. In this case, we regard that an outbreak has occurred. Otherwise, initially infected nodes do not trigger secondary infections on a visible scale. The condition $\mathrm{d}I(t)/\mathrm{d}t|_{t=0} > 0$ gives the epidemic threshold, and the result coincides with that for the SIS model given by Eq. (219).

The third model, the SI model, is a simplified version of the SIR model. Infection events occur in the same manner as in the SIS and SIR models. In the SI model, once a node is infected, it will stay infected forever. Therefore, if infection is introduced to a connected static network, every node will be eventually infected. No notion of epidemic threshold exists for the SI model. Instead, one is interested in how fast infection spreads. The dynamics in the well-mixed population are governed by

$$\frac{\mathrm{d}I(t)}{\mathrm{d}t} = \lambda S(t)I(t), \tag{223}$$

where we omit the equation for the dynamics of $S(t)$. In an early stage of dynamics where only a tiny fraction of nodes is infected, Eq. (223) with $S(t) \approx 1$ yields

$$I(t) \propto e^{\lambda t}. \tag{224}$$

Although the SI model is unrealistic, it behaves similarly to the SIR model in the initial stage of the dynamics, where the number of recovered nodes in the SIR model can be safely neglected.

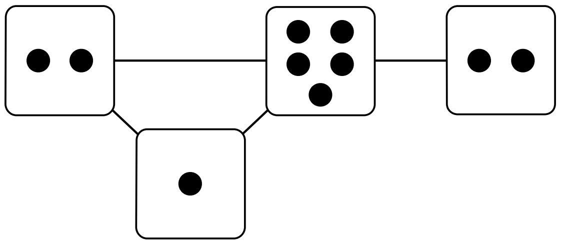

FIG. 29. A metapopulation model network with $\tilde{N} = 4$ subpopulations and $N = 10$ individuals.

B. SIS dynamics on metapopulation models

In this section, 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. In Fig. 29, a circle represents an individual, and a box containing individuals represents a subpopulation. A metapopulation model network consists of $\tilde{N}$ nodes, called subpopulations, 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 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; $\tilde{A}_{ij} = \tilde{A}_{ji}$ $(1 \le i, j \le \tilde{N})$. A link 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.

Metapopulation models are typically defined by two ingredients: a rule for the mobility of the individuals, and a rule for their interactions on the nodes. We **first** focus on the former aspect. Denote by N_i $(1 \le i \le \tilde{N})$ 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. A metapopulation model network on $\tilde{N} = 4$ subpopulations and $N = 10$ individuals is shown in Fig. 29. 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 $D\Delta t\tilde{A}_{ij}/\tilde{k}_i$, where

$$\tilde{k}_i = \sum_{j=1}^{\tilde{N}} \tilde{A}_{ij} \tag{225}$$

is the (weighted) degree of the ith subpopulation.

The number of individuals in each subpopulation, N_i, is approximated to be a continuous variable when N is large. The master equation for N_i is given by

$$\begin{aligned}\frac{\mathrm{d}N_i}{\mathrm{d}t} &= -DN_i + D\sum_{j=1}^{\tilde{N}} N_j \frac{\tilde{A}_{ji}}{\tilde{k}_j} \\ &= -D\sum_{j=1}^{\tilde{N}} N_j L^{'}_{ji},\end{aligned} \tag{226}$$

where $1 \le i \le \tilde{N}$ and $L^{'}$ denotes the random walk normalised Laplacian matrix, as usual. By setting the left-hand side of Eq. (226) to zero, we obtain the equilibrium density of individuals as

$$N_i^* = \frac{\tilde{k}_i N}{\langle \tilde{k}\rangle \tilde{N}}. \tag{227}$$

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. By definition, the master equation for the number of individuals in each subpopulation is given by Eq. (226) when a epidemic process does not take place. We denote by $N_{\mathrm{S},i}$ and $N_{\mathrm{I},i}$ 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

$$\frac{\mathrm{d}N_{\mathrm{S},i}}{\mathrm{d}t} = -\beta N_{\mathrm{S},i}N_{\mathrm{I},i} + \mu N_{\mathrm{I},i} - D_{\mathrm{S}}N_{\mathrm{S},i} + D_{\mathrm{S}}\sum_{j=1}^{\tilde{N}} \frac{\tilde{A}_{ji}}{\tilde{k}_j} N_{\mathrm{S},j}, \tag{228}$$

$$\frac{\mathrm{d}N_{\mathrm{I},i}}{\mathrm{d}t} = \beta N_{\mathrm{S},i}N_{\mathrm{I},i} - \mu N_{\mathrm{I},i} - D_{\mathrm{I}}N_{\mathrm{I},i} + D_{\mathrm{I}}\sum_{j=1}^{\tilde{N}} \frac{\tilde{A}_{ji}}{\tilde{k}_j} N_{\mathrm{I},j}, \tag{229}$$

where $\tilde{N}$ is the number of subpopulations such that $1 \le i \le \tilde{N}$, $\tilde{A}$ is the adjacency matrix of the network of the subpopulations, and $\tilde{k}_i = \sum_{j=1}^{\tilde{N}} \tilde{A}_{ij}$ is the degree of the ith subpopulation.

To determine the epidemic threshold, we consider the situation in which $N_{\mathrm{I},i}$ $(1 \le i \le N)$ is infinitesimally small. In this situation, Eq. (228) is essentially the same as the master equation without epidemic dynamics (Eq. (226)) because $\beta N_{\mathrm{S},i}N_{\mathrm{I},i}$ and $\mu N_{\mathrm{I},i}$ on the right-hand side of Eq. (228) are very small as compared to the other terms. As long as susceptible individuals move (i.e., $D_{\mathrm{S}} > 0$), the equilibrium for the susceptible individuals is given by

$$N_{\mathrm{S},i}^* = \frac{\tilde{k}_i N}{\langle \tilde{k}\rangle \tilde{N}}, \tag{230}$$

where N is the number of individuals in the entire population. Equation (230) is the same as Eq. (227). Substitution of Eq. (230) in Eq. (229) yields

$$\frac{\mathrm{d}N_{\mathrm{I},i}}{\mathrm{d}t} \approx \frac{\beta N}{\langle \tilde{k}\rangle \tilde{N}}\tilde{k}_i N_{\mathrm{I},i} - \mu N_{\mathrm{I},i} - D_{\mathrm{I}}N_{\mathrm{I},i} + D_{\mathrm{I}}\sum_{j=1}^{\tilde{N}} \frac{\tilde{A}_{ji}}{\tilde{k}_j} N_{\mathrm{I},j}. \tag{231}$$

Equation (231) admits the disease-free solution $N_{\mathrm{I},i}^* = 0$ $(1 \le i \le N)$. The destabilisation of the disease-free solution implies an endemic state, where infection can persist. We rewrite Eq. (231) in the vector form:

$$\frac{\mathrm{d}N_{\mathrm{I}}}{\mathrm{d}t} \approx BN_{\mathrm{I}}, \tag{232}$$

where $N_{\mathrm{I}} = (N_{\mathrm{I},1}, \ldots, N_{\mathrm{I},\tilde{\mathrm{N}}})^{\top}$ and B is the $\tilde{N} \times \tilde{N}$ matrix whose elements are given by

$$B_{ij} = \delta_{ij}\left(\frac{\beta N}{\langle \tilde{k}\rangle \tilde{N}}\tilde{k}_i - \mu - D_{\mathrm{I}}\right) + (1-\delta_{ij})D_{\mathrm{I}}\frac{\tilde{A}_{ji}}{\tilde{k}_j}. \tag{233}$$

The epidemic threshold is given when the largest eigenvalue of B becomes positive. In general, and in contrast to the case of mean-field populations, the epidemic threshold is not expressed only in terms of β/μ. To show so, let us look at some special cases. When $D_{\mathrm{I}} = \infty$, Eq. (231) implies

$$N_{\mathrm{I},i} = \frac{\tilde{k}_i N_{\mathrm{I}}}{\langle \tilde{k}\rangle \tilde{N}}, \tag{234}$$

where $N_{\mathrm{I}} \equiv \sum_{i=1}^{\tilde{N}} N_{\mathrm{I},i} \approx 0$ is the number of infected individuals in the entire population near the epidemic threshold. By substituting Eq. (234) in Eq. (231) and taking the summation over i, we obtain

$$\frac{\mathrm{d}N_{\mathrm{I}}}{\mathrm{d}t} = \left(\frac{\beta N\langle \tilde{k}^2\rangle}{\langle \tilde{k}\rangle^2 \tilde{N}} - \mu\right)N_{\mathrm{I}}, \tag{235}$$

where $\langle \tilde{k}^2\rangle \equiv \sum_{i=1}^{\tilde{N}} \tilde{k}_i^2/\tilde{N}$. Therefore, the condition for endemicity is given by

$$\frac{\beta}{\mu} > \frac{\langle \tilde{k}\rangle^2 \tilde{N}}{\langle \tilde{k}^2\rangle N}. \tag{236}$$

In other words, a metapopulation model with a broad degree distribution yields a small epidemic threshold because $\langle\tilde{k}^2\rangle \gg \langle\tilde{k}\rangle$.

When $D_\mathrm{I} = 0$, Eq. (231) represents $\tilde{N}$ decoupled dynamics. The epidemic threshold is determined by the subpopulation having the largest degree, denoted by $\tilde{k}_\mathrm{max}$. The condition for endemicity is given by

$$\frac{\beta}{\mu} > \frac{\langle\tilde{k}\rangle\tilde{N}}{\tilde{k}_\mathrm{max}N}. \tag{237}$$

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

Ex X.1: Write a Monte-Carlo code for the simulation of the SIS model with 1000 agents on a network of 100 nodes. The metapopulation network should be generated with the configuration model to tune the mean and the variance of the degree distribution. Verify numerically the theoretical predictions.

C. SI dynamics on contact networks

A second way to introduce networks into the modelling of disease spreading is to consider a virus propagating on a contact network of individuals. In contrast with the previous section, the nodes are now individuals, links represent contacts between them, and the spreading entity is the virus itself.

In this section, we consider such models for the simple SI dynamics. Each node i is defined, at each time t, by a variable $x_i(t)$, equal to 1 if the node is infected, and 0 otherwise. Keeping in mind that the model is stochastic, we are interested in the evolution of the expectation $\langle x_i(t)\rangle$ over a large number of realisations, which can be interpreted as the probability that node i is infected at time t. Its evolution is determined by the set of coupled differential equations:

$$\frac{d\langle x_i\rangle}{dt} = \lambda \sum_j A_{ij}\langle(1-x_i)x_j\rangle\ , \tag{238}$$

where A_{ij} is the adjacency matrix and the quantity $\langle(1-x_i)x_j\rangle$ is the probability (averaged over the ensemble of realisations) that node i is susceptible and node j is infected. Note that the set of equations is large but, worse, it is not closed: in order to solve the equations for $\langle x_i(t)\rangle$, one needs the evolution for $\langle(1-x_i)x_j\rangle$, but it will appear that an equation for the evolution of the latter quantity $\langle(1-x_i)x_j\rangle$ itself involves higher-order terms, leading to a hierarchy of more and more complicated equations. This is a standard problem appearing in a variety of problems, e.g. see the BBGKY hierarchy in statistical physics, that can be solved by truncating the hierarchy at some level. The simplest truncation is obtained by assuming that the states of neighbouring nodes are independent, that is

$$\langle(1-x_i)x_j\rangle = \langle 1-x_i\rangle\langle x_j\rangle = (1-\langle x_i\rangle)\langle x_j\rangle. \tag{239}$$

Note that more sophisticated moment closure approximations can be used.

Plugging (239) into (238), one obtains the system of equations

$$\frac{d\langle x_i\rangle}{dt} = \lambda(1-\langle x_i\rangle)\sum_j A_{ij}\langle x_j\rangle\,. \tag{240}$$

In order to get insight about the resulting dynamics, let us consider a scenario where a small fraction of nodes is initially infected, and assume that only linear terms are to be kept at the time of interest, so that

$$\frac{d\langle x_i\rangle}{dt} = \lambda\sum_j A_{ij}\langle x_j\rangle\,. \tag{241}$$

Performing a standard spectral decomposition of the adjacency matrix

$$A_{ij} = \sum_{\ell=1}^{N}\beta_\ell u_{\ell:i}u_{\ell:j}, \tag{242}$$

and keeping only the dominant eigenvector, u_1, one finds the dominant behaviour

$$x_i \approx e^{\lambda\beta_1 t}u_1. \tag{243}$$

The growth rate of the disease is thus determined by the infectious rate and by the dominant eigenvalue of the underlying contact network.

To get some intuition of the factors influencing β_1, we perform a mean-field approximation, by replacing the adjacency matrix by its annealed version, that is by replacing the adjacency matrix by the expected number of links between nodes in the corresponding configuration model (preserving the degrees of each nodes)

$$A'_{ij} = k_ik_j/2M,$$

see the section VI B on Modularity for instance. By doing so, we only keep information about the node degrees and neglect any other information about the network structure. One readily finds that its dominant eigenvector is proportional to the degrees of the nodes and that the corresponding eigenvalue is

$$\beta'_1 = \frac{\langle k^2\rangle}{\langle k\rangle}.$$

This result shows that the variance of the degree distribution plays an important role for spread of a disease on a network. An intuitive argument being that high degree nodes are important for two reasons, hence the square: because they can easily be infected (due to their large degree) and because they can also infect many other nodes. This observation, and similar ones for SIS and SIR, have had a strong impact in the early 2000s, especially in the community of researchers interested in scale-free networks where such a variance would diverge.

XI. WHAT'S NEXT?

This is a first course on network science and, even if we have covered several aspects of it, many aspects have been left on the side. Nonetheless, I hope that this introduction will have given you the keys, and the curiosity, to discover more advanced topics in the future.

Notably, let us note that all dynamical systems covered so far were linear dynamical systems and I can only encourage you to discover non-linear dynamical ones, for instance for synchronization. See for instance Arenas, A., Díaz-Guilera, A., Kurths, J., Moreno, Y. and Zhou, C. (2008). Synchronization in complex networks, *Physics Reports* **469**, pp. 93–153.

Other important topics, that have attracted much attention, are spatially-embedded networks (Barthélemy, M. (2011), Spatial networks, *Physics Reports* **499**, pp. 1–101.), that is networks where the underlying physical space affects the network organisation, and temporal networks (Holme, P. and Saramäki, J. (2012), Temporal networks, *Physics Reports* **519**, pp. 97–125; and A guide to temporal networks, N. Masuda and R. Lambiotte, World Scientific 2016), where the connections between the nodes are dynamical entities.

The connections between network science and statistics has also seen a stream of works recently, especially for the detection of significant structures in networks. See for instance the recent Parsimonious module inference in large networks, TP Peixoto, Physical review letters 110 (14), 148701.

This course remained mostly methodological and we have not focused in detail on the practical applications of network science. For more works in neuroscience, and in social/economic systems, see respectively: Networks, Crowds, and Markets: Reasoning About a Highly Connected World, David Easley and Jon Kleinberg; and Complex brain networks: graph theoretical analysis of structural and functional systems, E Bullmore, O Sporns, Nature Reviews Neuroscience 10 (3), 186-198 (2009).

Similarly, this course finds connections with many aspects in data mining, see for instance Akoglu, L., Tong, H. and Koutra, D. (2015). Graph based anomaly detection and description: A survey, *Data Mining and Knowledge Discovery* **29**, pp. 626–688.