1 Introduction

Over the last 20 years, networks and graphs have become a near-ubiquitous modelling framework for complex systems. By representing the entities of a system as nodes in a graph and encoding relationships between these nodes as edges, we can abstract systems from a variety of domains with the same mathematical language, including biological, social, and technical systems (Newman, 2018a). Network abstractions are often used with one of the two following perspectives in mind. First, graphs and networks provide a natural way to describe relational data (i.e., datasets corresponding to ‘interactions’ or correlations between pairs of entities). A prototypical example here is online social networks, in which we measure interactions between actors and can derive a network representation of the social system based on these measurements. We may then try to explain certain properties of the social system by modelling and analysing the network (e.g., by searching for interesting connectivity patterns between the nodes). Second, networks are often used to describe distributed dynamical systems, including prototypical examples such as power grids, traffic networks, or various other kinds of supply or distribution networks. The edges of the network are in this context not the primary object of our modelling. Rather, we are interested in understanding a dynamical process that takes place on this network. More specifically, we often aim to comprehend how the network structure shapes this dynamics (e.g., in terms of its long-term behaviour). In reality, of course, both these perspectives are simplifications in that for many real systems, there are typically uncertainty and dynamics associated to both node and edge variables which make up the network: think, for instance, of a rumour spreading on a social network, where both node variables (the infection state) and the network edges (who is in contact with whom) will be highly dynamic and uncertain. We may not know the exact status of each individual; moreover, edges will change dynamically, and their presence or absence may not be determined accurately.

No matter under what perspective we are interested in networks, it should be intuitively clear that networks with some kind of ‘modular structure’ may be of interest to us. For now, consider modular structure simply in terms of a network made of dense clusters that are loosely connected with each other. From the perspective of relational data, modular structure may be indicative of a hidden cause that binds a set of nodes together: this corresponds to the idea of homophily in social networks (McPherson, Smith-Lovin, & Cook, 2001), which can lead to the formation of communities of tightly knit actors. From the perspective of dynamics, it is often impractical to keep a full description of a dynamical process on a network when the number of dynamical units is too large. In many cases, it is unclear whether such finely detailed
data is necessary to understand the global phenomena of interest, as relevant observables can often be obtained by aggregating microscopic information into macroscopic information (i.e., aggregating information over many nodes). This kind of dimensionality reduction of the dynamics is particularly successful if there exist roughly homogeneously connected blocks of nodes (i.e., a modular network structure (Simon & Ando, 1961)).

As the title indicates, this Element will primarily adopt a dynamical perspective on network analysis. Accordingly, our core objective will be to explore the relations between modular structure and dynamics on networks; but we will also explain how certain aspects of the analysis of relational data can be interpreted from this lens. However, an exhaustive exposition of methods to characterise and uncover modules (also called blocks, clusters, or communities) in networks will not be the main focus of our exposition. We refer the interested reader to the extensive literature on this topic for more detailed treatments; see, for example, Doreian, Batagelj, and Ferligoj (2020); Fortunato and Hric (2016); Newman (2018a); Schaeffer (2007).

Network Dynamics and Network Structure

It is well-known that there exists a two-way relationship between dynamics on graphs and the underlying graph structure (Schaub et al., 2019b). On the one hand, the structure of a network affects dynamical processes taking place on it (Porter & Gleeson, 2016). In the simplest case of a linear dynamical system, this relationship derives from the spectral properties of a matrix encoding the graph, most often the adjacency matrix or the graph Laplacian. On the other hand, dynamics can help reveal salient features of a graph. This includes the identification of central nodes or the detection of modules in large-scale networks.

To illustrate this two-fold relation between network structure and network dynamics on an intuitive level, let us consider random walks on networks. Random walks are often used as a model for diffusion, and there is much research on the impact of network structure on different properties of random-walk dynamics (Masuda, Porter, & Lambiotte, 2017). In particular, degree heterogeneity, finite size effects and modular structure can all make diffusion processes on networks quantitatively and even qualitatively different from diffusion on regular or infinite lattices. At the same time, random walks are key to many algorithms that uncover various types of structural properties of networks. For example, the classical PageRank method for identifying important nodes may be interpreted in terms of a random walk (Gleich, 2015). Indeed, as we will discuss, several algorithms use trajectories of dynamical processes such as random walks to reveal mesoscale network patterns.
Our main ambition is to understand relationships between modular structure of a network, here highlighted in different node colours, and a dynamics taking place on it, here illustrated with the red trajectory on the network. The two complementary questions at the core of this Element are: (1) How does the modular structure of a network affect dynamics? (2) How can dynamics help us characterise and uncover the modular structure of a network?

Outline of This Element

In this Element, we try to provide a basic overview of the topic of modularity and dynamics on complex networks. Our exposition is structured as follows. In Section 2, we first discuss some background material in Network Science and then review classical notions of modular structure in networks in Section 3. In Sections 4 and 5, we discuss the interplay between dynamics and network structure in terms of timescale separation and symmetries, and how these aspects can be used to reduce the complexity of the description of network dynamics. In Section 6, we then explain how we can detect so-called assortative community structure, primarily based on the notion of timescale separation. Section 7 then discusses the definition and detection of more general (dynamical) block structure, leveraging ideas from linear systems theory and symmetry reduction. In Section 8, we conclude with a short discussion on avenues for future work and additional perspectives.

Why Are Networks Modular?

For many years, researchers have been fascinated by the prevalence of modularity in systems as different as the World Wide Web, foodwebs, and brain networks, raising the question: are there universal mechanisms driving the evolution of networks toward a modular architecture? Among
the many mechanisms that have been proposed (Meunier, Lambiotte, & Bullmore, 2010), the following profound idea of Herbert Simon (1962) stands out by its elegance. ‘Nearly-decomposable’ systems, as Simon calls them, allow faster adaptation or evolution of the system in response to changing environmental conditions. In Simon’s view, modules represent stable building blocks that ensure the robustness of a system evolving under changing or competitive selection criteria. To illustrate this idea, Simon wrote an intuitive parable about two watchmakers, called ‘Hora’ and ‘Tempus’ (Simon, 1962):

There once were two watchmakers, named Hora and Tempus, who manufactured very fine watches. Both of them were highly regarded, and the phones in their workshops rang frequently – new customers were constantly calling them. However, Hora prospered, while Tempus became poorer and poorer and finally lost his shop. What was the reason?

The watches the men made consisted of about 1,000 parts each. Tempus had so constructed his that if he had one partly assembled and had to put it down – to answer the phone say – it immediately fell to pieces and had to be reassembled from the elements. The better the customers liked his watches, the more they phoned him, the more difficult it became for him to find enough uninterrupted time to finish a watch.

The watches that Hora made were no less complex than those of Tempus. But he had designed them so that he could put together sub-assemblies of about ten elements each. Ten of these subassemblies, again, could be put together into a larger subassembly; and a system of ten of the latter subassemblies constituted the whole watch. Hence, when Hora had to put down a partly assembled watch in order to answer the phone, he lost only a small part of his work, and he assembled his watches in only a fraction of the man-hours it took Tempus.

This story illustrates in simple terms the potential evolutionary advantage that a modular system structure may have, and provides an argument for the ubiquity of modularity in a broad range of natural and social systems.\(^a\) In the following, we will not dwell on why there is modular structure in the network, but rather focus on the question: how does the modularity of a network affect its behaviour and, in particular, its dynamical properties?

\(^a\) One needs to be careful with such statements. Simon himself cautioned that many systems lack conclusive statistical evidence for being modular and may only be perceived as modular due to confirmation bias. However, the statement that many networks are modular has been validated on a large corpus of network datasets by now. See, for example, Fortunato (2010); Ghasemian, Hosseinmardi, and Clauset (2019); Leskovec et al. (2008).
Table 1 Notation

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n\in\mathbb{N})</td>
<td>number of nodes</td>
</tr>
<tr>
<td>(m\in\mathbb{R})</td>
<td>total weight of edges (number of edges for unweighted networks)</td>
</tr>
<tr>
<td>(C\in\mathbb{N})</td>
<td>number of modules / communities</td>
</tr>
<tr>
<td>(\mathcal{V} = {1, \ldots, n})</td>
<td>set of nodes / vertices</td>
</tr>
<tr>
<td>(i, j, \ell \in {1, \ldots, n})</td>
<td>indices for nodes</td>
</tr>
<tr>
<td>(\mathcal{P} = {A_1, \ldots, A_C})</td>
<td>partition of the nodes into communities (A_\alpha)</td>
</tr>
<tr>
<td>(A_\alpha)</td>
<td>set of nodes within the (\alpha)th community</td>
</tr>
<tr>
<td>(\alpha, \beta \in {1, \ldots, C})</td>
<td>indices for communities</td>
</tr>
<tr>
<td>(k_i \in \mathbb{R})</td>
<td>weighted degree (strength) of node (i)</td>
</tr>
<tr>
<td>(A \in \mathbb{R}^{n \times n})</td>
<td>weighted adjacency matrix of a network</td>
</tr>
<tr>
<td>(K(A) = \text{diag}(A) \in \mathbb{R}^{n \times n})</td>
<td>weighted degree matrix of a network</td>
</tr>
<tr>
<td>(L(A) = K - A)</td>
<td>combinatorial Laplacian matrix</td>
</tr>
<tr>
<td>(\mathcal{L}(A) = I - K^{-1/2}AK^{-1/2})</td>
<td>normalised Laplacian matrix</td>
</tr>
<tr>
<td>(L_{rw}(A) = I - K^{-1}A)</td>
<td>random-walk Laplacian matrix</td>
</tr>
<tr>
<td>(H \in {0, 1}^{n \times C})</td>
<td>partition indicator matrix with entries (H_{i\alpha} = 1), if node (i) is in the (\alpha)th community ((A_\alpha)), and (H_{i\alpha} = 0) otherwise</td>
</tr>
<tr>
<td>(h_\alpha \in {0, 1}^n)</td>
<td>Indicator vector of the (\alpha)th community (i.e., (\alpha)th column of (H))</td>
</tr>
<tr>
<td>(\gamma : {1, \ldots, n} \to {1, \ldots, n})</td>
<td>permutation function of node labels</td>
</tr>
<tr>
<td>(\Gamma)</td>
<td>permutation matrix associated to (\gamma)</td>
</tr>
<tr>
<td>(d(x, y))</td>
<td>distance function between (x) and (y)</td>
</tr>
<tr>
<td>(\kappa(x, y))</td>
<td>kernel function of (x) and (y)</td>
</tr>
</tbody>
</table>

Notation

We use the following general mathematical notations and conventions. We denote vectors by small letters in bold such as \(x, y\) and use \((\cdot)^\top\) to denote the transpose of a vector. Our convention is that all vectors are column vectors, and accordingly, \(x^\top\) is a row vector. We use \(1\) to denote the vector of all ones. Matrices are denoted by bold uppercase letters such as \(A, M\), where \(I\) is used to denote the identity matrix. We write \(\text{diag}(x)\) to denote the diagonal matrix whose diagonal entries are defined by the components of vector \(x\) and are 0 otherwise. Entries of vectors or matrices are non-bold with subscripts. For instance, vector \(x\) has entries \(x_1, \ldots, x_n\) and the matrix \(A\) has entries \(A_{ij}\). If there is ambiguity, we may alternatively use the notation \([v_i]_j\) to denote the \(j\)th entry of a vector \(v_i\) (or a matrix, accordingly). Finally, we use \(\mathbb{P}(\cdot)\) and \(\mathbb{E}[\cdot]\) for the probability and expectation of the statement inside the parentheses, respectively.
6 The Structure and Dynamics of Complex Networks

More specific notation regarding networks and associated objects is summarised in Table 1. These objects are explained in Section 2.

2 Background Material

In this section, we review some notions from algebraic and spectral graph theory as well as the theory of linear dynamical systems. These concepts will be essential for our discussions in the following sections.

2.1 Graph Theory

Networks provide a natural framework to represent systems composed of elements in interaction. At the core of a network representation is the inherent assumption that the system under investigation can be decomposed into nodes, representing the system elements, and edges, representing pairwise interactions between the system elements.

In the simplest setting, we assume that both the nodes and the edges of a network are all of the same type and their number is fixed. All of these assumptions can be relaxed, but we will be mostly concerned with undirected (and weighted) networks in this Element. Within this setup we can represent a network mathematically by a graph \( G(\mathcal{V}, \mathcal{E}) \), with a set of nodes \( \mathcal{V} \) of cardinality \( n := |\mathcal{V}| \), and a set of edges \( \mathcal{E} = \{i, j\} \mid i, j \in \mathcal{V} \). Without loss of generality, we will identify the node set \( \mathcal{V} \) with the set \( \{1, \ldots, n\} \). For a weighted graph, we endow the graph \( G \) with a weight function \( w_G : \mathcal{E} \to \mathbb{R}_+ \), which maps each edge \( \{i, j\} \) to a positive weight, \( w_G(\{i, j\}) = w_{ij} \).

More generally, we can consider directed graphs, meaning that node \( i \) may be adjacent (connected) to node \( j \) but not vice versa. This lack of symmetry leads to a number of mathematical complications that make directed networks and dynamical systems acting on directed networks far more difficult to analyse (cf. the box ‘The Case of Directed Networks’ in Section 2.3.3). We will thus focus on undirected networks, unless otherwise stated.

The edge set of a graph describes which nodes are adjacent (i.e., directly connected by an edge). Especially in the context of dynamical systems defined on graphs, we need to capture how a sequence of direct connections defines indirect connectivity between pairs of nodes, leading to the additional notions

\[1\] For simplicity, within this Element, we will concentrate on undirected graphs with positive edges weights and provide some additional discussion on directed networks. Relaxing the above modelling assumptions leads to various notions, including signed networks (Kunegis et al., 2010), multiplex networks (Kivela et al., 2014), temporal networks (Holme & Saramäki, 2019), and higher-order networks (Battiston et al., 2020; Lambiotte, Rosvall, & Scholtes, 2019; Schaub et al., 2021). These are active areas of research that we will mention further when relevant, and we invite the reader to consult the literature for further information on these topics.
of a walk, trail, path, and connectedness of a graph. A walk between a starting node $i$ and a terminal node $j$ is a sequence of edges such that there exists an associated node sequence $(i, \ell, \ldots, j)$, in which every subsequent pair of nodes is adjacent. A trail is a walk in which all edges are distinct, and a path is a trail in which additionally all nodes are distinct. A graph is connected if there is a path between any two nodes. When a graph is not connected, it is composed of several connected components.\(^2\)

There are different ways to describe a graph algebraically. One representation that will attract a lot of our attention is the so-called adjacency matrix. The adjacency matrix $A$ is an $n \times n$ matrix encoding the presence or absence of an edge between any pair of nodes in the graph. Its entries are

$$A_{ij} = \begin{cases} 1 & \text{if node } i \text{ is adjacent to node } j, \\ 0 & \text{otherwise}. \end{cases} \quad (2.1)$$

If the network is weighted, then $A_{ij} = w_{ij}$ if there is an edge between $i$ and $j$ and zero otherwise. Here $w_{ij}$ is the edge weight associated to edge $\{i, j\}$, as discussed earlier. Clearly, for undirected graphs we have $A_{ij} = A_{ji}$ (i.e., the adjacency matrix is symmetric ($A = A^T$)). Using the adjacency matrix, we can express the weighted degree of each node $i$ as $k_i = \sum_j A_{ij}$. The degree of a node is equal to the number of neighbours of a node for simple, unweighted graphs.

### 2.2 Random Graph Models

Many local or global properties of a network can potentially be of interest, whether in terms of their influence on a dynamics acting on a network or otherwise. For instance, let us consider the clustering coefficient $C$ defined as

$$C := \frac{\text{# triangles in graph}}{\left(\text{# number of connected triplets}\right)/3}. \quad (2.2)$$

which measures the relative abundance of triangles in a network. By construction, the value of the clustering coefficient $C$ lies between 0 and 1, and it is often considered as a measure of the cohesion inside a network.

Now suppose that we observe a value of the clustering coefficient of $C = 0.25$ in an empirical network. Should we conclude that this value is small or large? In order to answer this question and interpret our measurement in a meaningful way, we often require some suitable reference points for the measurement. Such reference points are often deduced from random graph models.

\(^2\) Note that a single node without any connection is a trivial connected component by definition.
A random graph model defines an ensemble of graphs (i.e., a set of possible graphs and a probability associated to each of those graphs). Random graph models are often defined by considering the edges in the graph as random variables, which obey certain probabilistic laws. One of the simplest random graph models is the Erdős–Rényi (ER) model, also called the Poisson or binomial random graph. The Erdős–Rényi random graph has two parameters: the number of nodes \( n \), and the probability \( q \) that a link exists between a pair of nodes. By definition, self-loops are excluded. Each pair of nodes (undirected edge) is then seen as an independent Bernoulli random variable that determines the presence or absence of a link. The combined realisations of all these \( \frac{n(n-1)}{2} \) Bernoulli random variables then determine one realisation of the random graph. Note that any network without multiple edges and self-loops can be generated by sampling from an ER model as long as \( 0 < q < 1 \). However, different realisations will be observed with different probabilities, depending on the value of \( q \). Exploiting the fact that each link exists independently with the same probability, several properties of the ER model can be calculated analytically, in particular the expected value of several network metrics as well as their variance.

However, the ER model is known to produce unrealistic edge patterns. In particular, it generates networks with a Poisson degree distribution, which is not typical for most observed networks in the real world.\(^3\) For this reason, other random graph models such as the configuration model are often considered to be a more appropriate baseline for real-world networks. The configuration model (Fosdick et al., 2018) is defined as a random graph model in which all possible configurations appear with the same probability under the constraint that each node \( i \) has a given degree \( k_i \) (\( 1 \leq i \leq n \)). Hence, the configuration model generates an ensemble of random graphs with a prescribed degree distribution that can either be taken from empirical data or chosen from a family of functions (e.g., a power-law distribution). The soft configuration model, or Chung-Lu model (F. F.Chung & Lu, 2002; Park & Newman, 2004), is defined analogously to the configuration model; but rather than fixing the exact degree sequence of the graph, only the expected degree sequence is prescribed. Similar to the ER model, several properties of the (soft) configuration model can be calculated analytically. Of interest for the following sections is the expected number of links between two nodes \( i \) and \( j \) in the soft-configuration model, which for \( k_i k_j < 2m \) is given by

---

\(^3\) Power-law or not (Broido & Clauset, 2019), many real-world networks tend to exhibit a degree distribution with a fat tail: the vast majority of nodes have only a small degree, and a small number of hubs have a large number of connections.
Modularity and Dynamics on Complex Networks

\[ \mathbb{E}[A_{ij}] = \frac{k_i k_j}{2m}. \quad (2.3) \]

Here \( \mathbb{E}[A_{ij}] \) represents the expected value of the adjacency matrix for the edge \( \{i, j\} \), which is simply the probability \( P(A_{ij} = 1) \) for an unweighted graph. As can be seen, this probability for an edge to exist between a pair of nodes is clearly not uniform, which was the case in the ER model. Note that here and in the following we are interested in the original soft configuration model proposed by Chung and Lu, as specified earlier, though a more general treatment can be given for graphs where the product \( k_i k_j \) can exceed \( 2m \) (Park & Newman, 2004).

Let us now return to the initial motivating example of this section (i.e., determining if a specific statistic like the clustering coefficient is small or large in a real-world network). A common practice is to consider how this measure would be statistically distributed for graphs drawn from a soft configuration model with an expected degree sequence matching the empirically analysed network. We can then determine if the empirical value is significantly different from that of the random graph model, for instance by calculating a \( Z \) score or a \( p \) value. For the specific case of the clustering coefficient, its expected value under the configuration model is given by M. E. J. Newman (2018a):

\[ C = \frac{\left( \mathbb{E}[k^2] - \mathbb{E}[k] \right)^2}{\mathbb{E}[k]^2 n}. \quad (2.4) \]

Note that this value is extremely small for large values of \( n \) unless the variance of the degree diverges. Thus, we would expect a very small clustering coefficient for large networks. This type of approach is popular for motif analysis, where the purpose is to uncover important motifs in a network, whose over- or under-representation may be associated to their function in the system (Milo et al., 2002). Let us emphasise here that this conclusion strongly depends on the choice of model, as the question of whether a particular statistic is significant can only be answered with respect to the chosen random model. Other models may have very different behaviour, and we thus need to exercise caution when declaring that some network property is significant or not.

### 2.3 Network Dynamical Systems and Linear Dynamics

#### 2.3.1 Linear Dynamics on Networks

In many situations, the nodes of a network are not static entities, but each node \( i \) carries a state \( x_i(t) \) that evolves in either continuous or discrete time. Consider, for instance, the formation of opinions in a social network, where each node \( i \) may update its opinion \( x_i(t) \) on a particular topic based on the interactions with
adjacent nodes or, alternatively, on the dynamics of a set of connected neurons in a part of the brain.

While each node has a dynamical state, it is typically assumed that the network is static when considering such network dynamical systems (i.e., the node set $\mathcal{V}$ and edge set $\mathcal{E}$ are constant over time). Although the network structure is thus not dynamic itself, its connectivity constrains how the node states $x_i(t)$ can influence each other. A crucial question in the study of network dynamical systems is therefore to characterise this influence of the network structure on the overall dynamics of the system. Vice versa, based on an observed network dynamics, we may also infer certain properties of the network.

A broad setup that is often considered in this context is the following set of autonomous, coupled ordinary differential equations,

$$\dot{x}_i = f_1(x_i) + \sum_{j=1, j \neq i}^{n} A_{ij} f_2(x_i, x_j), \quad \text{for all } i = 1, \ldots, n, \quad (2.5)$$

in conjunction with an initial condition $x(0) = x_0$. Here the function $f_1$ describes the intrinsic dynamics of the node (a form of self-coupling), and the function $f_2$ describes how states of two nodes interact with each other (e.g., nodes are assumed to have pairwise interactions, in agreement with a network representation). Note how the adjacency matrix in Eq. (2.5) ensures that nodes that are not connected do not influence each other directly. These types of dynamical models may appear in a variety of contexts such as synchronisation, decentralised consensus, and social dynamics (Bullo, 2019).

A simple, but important case is linear dynamics on a network, in which case the above dynamics can be written in the form

$$\dot{x} = F x \quad \text{with} \quad x(0) = x_0$$

for some matrix $F$, which we call the system matrix. We emphasise that given an undirected network with adjacency matrix $A$, only certain linear dynamics will be compatible with graph structures encoded by $A$ (see Section 2.3.2). Specifically, we will require that we can write $F = DA_G D^{-1}$ for some invertible diagonal matrix $D$ and an appropriately defined non-negative, symmetric matrix $A_G = A_G^T$ that has the same sparsity pattern as the adjacency matrix for all off-diagonal entries. In particular, for $i \neq j$ we require that $[A_G]_{ij} = F_{ij} = 0$ if $A_{ij} = 0$, and only nodes connected in the network interact directly.

Given such a system, linear systems theory tells us that the solution to the system just described is given by

$$x(t) = \exp(tF)x_0 = \left( \sum_{l=0}^{\infty} \frac{t^l F^l}{l!} \right) x_0.$$  

(2.7)