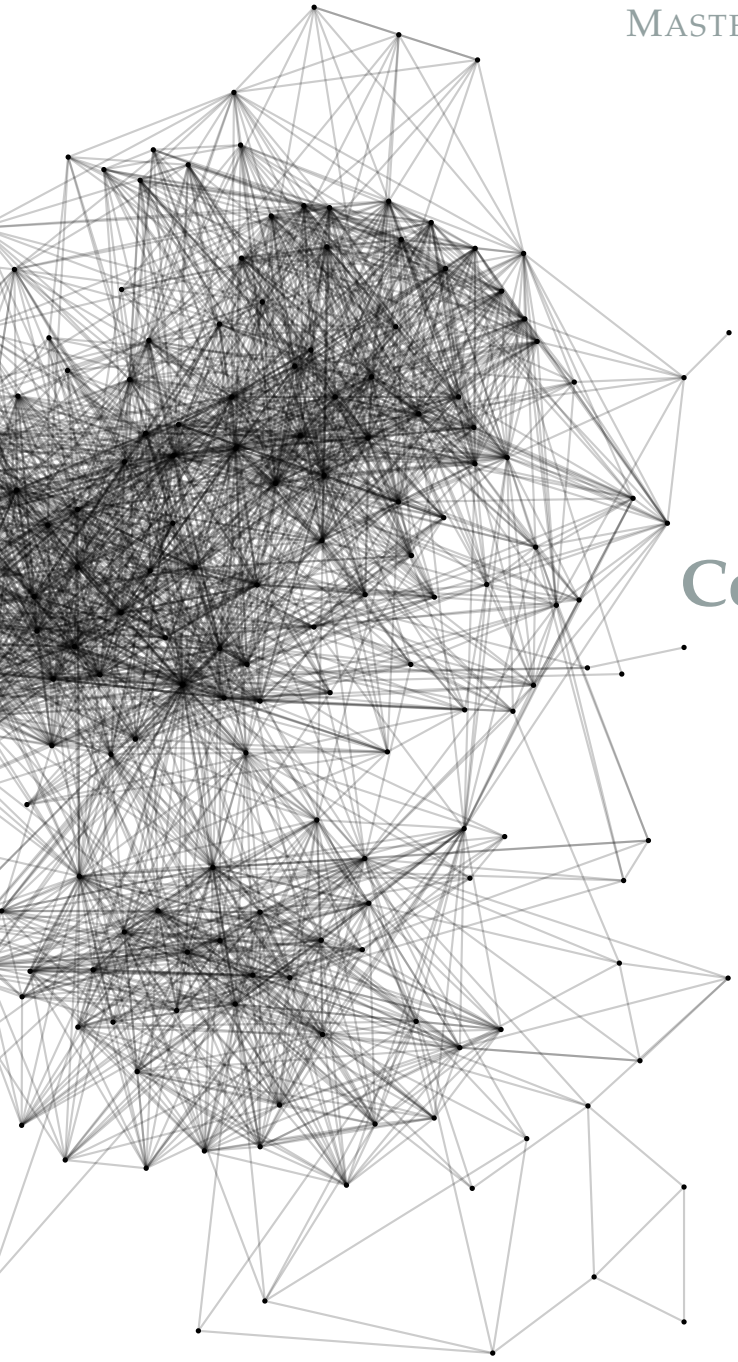
University of Fribourg

Master Thesis



Connectivity in networks

Author
Benoît Richard

Supervisors
Dr. Guiyuan SHI
Prof. Yi-Cheng Zhang

Department of Physics
Theoretical Interdisciplinary Physics Group

November 6, 2018

University of Fribourg

Abstract

Faculty of Science
Department of Physics

Master Thesis

Connectivity in networks

by Benoît RICHARD

Write the abstract

Contents

List of Symbols

Single layer networks

Symbol	Description	Math. definition
c	Expectation value for the degree	$\mathbb{E}[\deg v]$
C	A connected component	
$\deg v$	Degree of vertex v	
$\deg_1 v$	Degree of a vertex v that has been reached by following an edge	
$\mathbb{E}[\dots]$	Expectation value	
$g_0(z)$	Generating function for the degree of uniformly chosen nodes	$\sum_{k=0}^{\infty} p_k z^k$
$g_1(z)$	Generating function for the degree of nodes reached by following an edge	$\sum_{k=0}^{\infty} q_k z^k$
k_i	Degree of vertex i	
m	Number of edges in the network	
n	Number of nodes in the network	
$N(v)$	Neighborhood of a vertex v	
p_{ij}	Probability that vertices i and j are connected	
p_k	Probability that a random node has degree k	$P_0(\deg v = k)$
$P_0(\dots)$	Probability starting from a uniformly chosen node	
$P_1(\dots)$	Probability starting from a node reached by following an edge	
q_k	Probability that a node reached by following a edge has degree $k+1$	$P_1(\deg v = k+1)$
r_k	Probability that a random node of the GCC has degree k	$P_0(\deg v = k \vert v \in \mathrm{GCC})$
S	Fraction of the network which is part of the GCC in the large n limit	$P_0(v \in \mathrm{GCC})$
u	Probability that a node reached by following an edge is not part of the GCC	$P_1(v \notin \mathrm{GCC})$
v	Random variable representing a vertex chosen in a network, either uniformly or by following an edge depending of the context	
α	Exponent of a power law distribution	
$\zeta(\alpha)$	Riemann zeta function	

Multiplex networks

For multiplex networks, the convention we follows to indicate a value refer to layer i is to add an upper index (i) if the quantity has a lower index and to add a lower index i otherwise.

Symbol	Description	Math. definition
L	Number of layers of the multiplex network	
N	Number of parameters determining the degree distributions of a multiplex network	
$N_i(v)$	Neighborhood of a vertex v in layer i	
$F(\boldsymbol{\lambda}, \mathbf{z})$	Residual function	
$F_{\boldsymbol{\lambda}}(\mathbf{z})$	Residual function with fixed parameter vector $\boldsymbol{\lambda}$	
$g_0^{(i)}(z)$	Generating function for the degree of uniformly chosen nodes in layer i	$\sum_{k=0}^{\infty} p_k^{(i)} u_i^k$
$g_1^{(i)}(z)$	Generating function for the degree of nodes reached by following an edge in layer i	$\sum_{k=0}^{\infty} q_k^{(i)} u_i^k$
$J_{\boldsymbol{\lambda}}(\mathbf{z})$	Jacobi matrix of $F_{\boldsymbol{\lambda}}(\mathbf{z})$	
$p_k^{(i)}$	Probability that a uniformly chosen node has degree k in layer i	$P_0^{(i)}(\deg v = k)$
$P_0^{(i)}(\dots)$	Probability for an event in layer i starting from a uniformly chosen node	
$P_1^{(i)}(\dots)$	Probability for an event in layer i starting from a node reached by following an edge in layer i	
$q_k^{(i)}$	Probability that a node reached by following a edge has degree $k+1$	$P_1^{(i)}(\deg v = k+1)$
$\mathcal{R}$	Critical region	
u_i	Probability that a node reached by following an edge in layer i is not part of the GVC	$P_1^{(i)}(v \notin GVC)$
$\mathbf{u}$	Vector of all u_i	$(u_1, u_2, \dots, u_L)$
$\mathbf{u}_T$	Trivial solution for $\mathbf{u}$	$(1, 1, \dots, 1)$
$\mathbf{u}^\dagger$	Non trivial solution for $\mathbf{u}$	
λ_j	One of the N parameters determining the degree distributions of a multiplex network	
$\boldsymbol{\lambda}$	Parameter vector	$(\lambda_1, \lambda_2, \dots, \lambda_N)$
$\boldsymbol{\lambda}^c$	Parameter vector in the critical region	
$\psi(\boldsymbol{\lambda}, \mathbf{u})$	Fixpoint function for $\mathbf{u}$	
$\Psi(\Lambda, U)$	Interval extension of $\psi(\boldsymbol{\lambda}, \mathbf{u})$	

Todo list

Chapter 1

Single layer networks

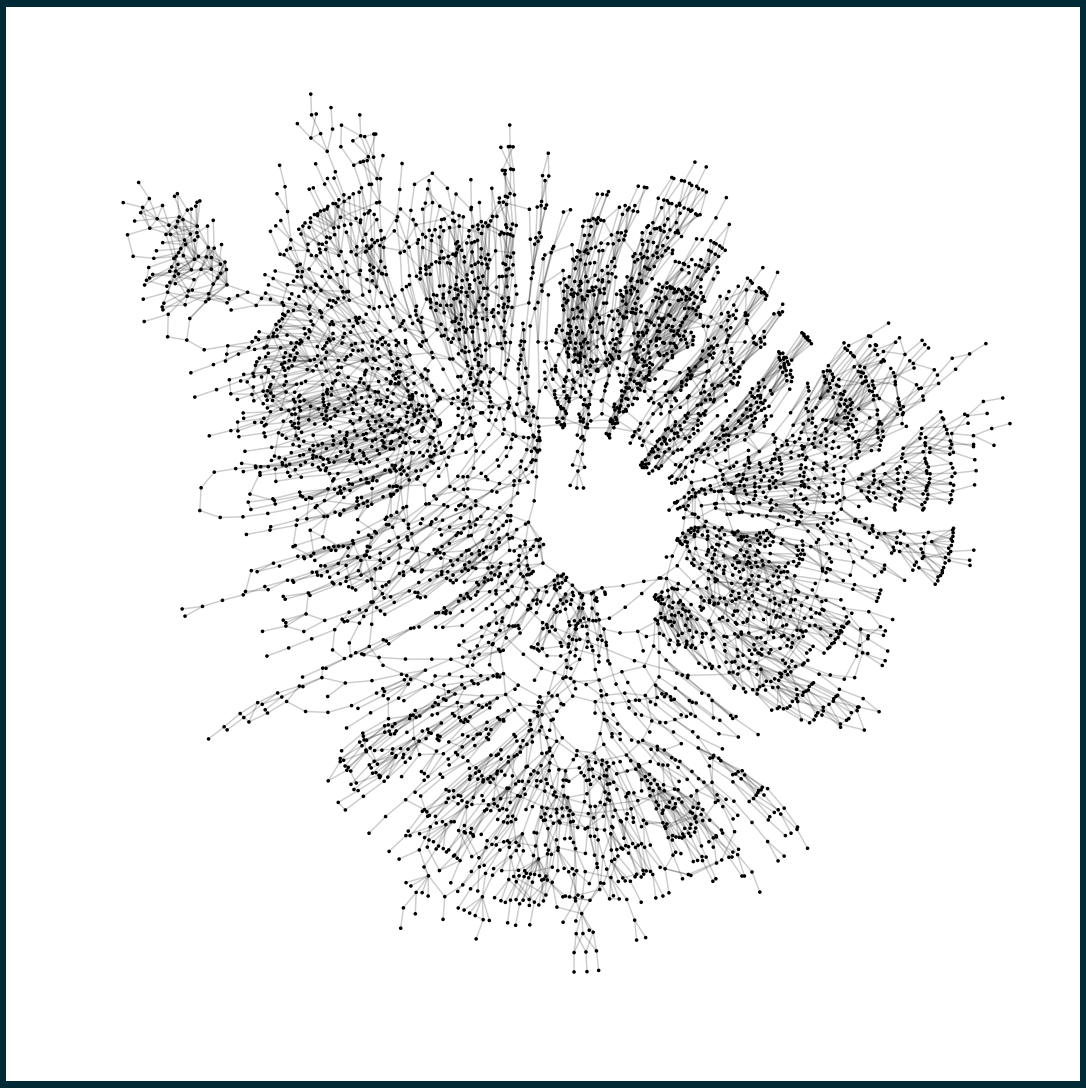
layer networks⟩

1.1 Introduction

Many systems in real world can be conceptually represented as a collection of objects being connected to each other in some way. Such representation is called a *network*. For example, a power grid can be represented as stations connected by power lines [1], as shown in fig. 1.1 for the power grid of western USA. The concept of network does not require the object or the links between them to be physical. As an example, we can represent collaboration as a network: two people are connected if they did collaborate, for example by coauthoring a paper [2], participate to the same board of direction [3] or having played in the same jazz band [4]. The latter being represented in fig. 1.2. This implies that a very different systems can be represented as networks, from cities connected by road [5, 6] (see fig. 1.3 for an exemple) to proteins connected if they interact [7, 8, 9] (see fig. 1.4 for a partial network of human proteins), including networks of emails user [10, 11], the internet [12] or networks of friendship [13]. Insight on fundamental properties of networks may therefore shed light on a very broad range of problems. To gain such insights, theoretical studies of general networks, such as the one presented in this thesis, are required.

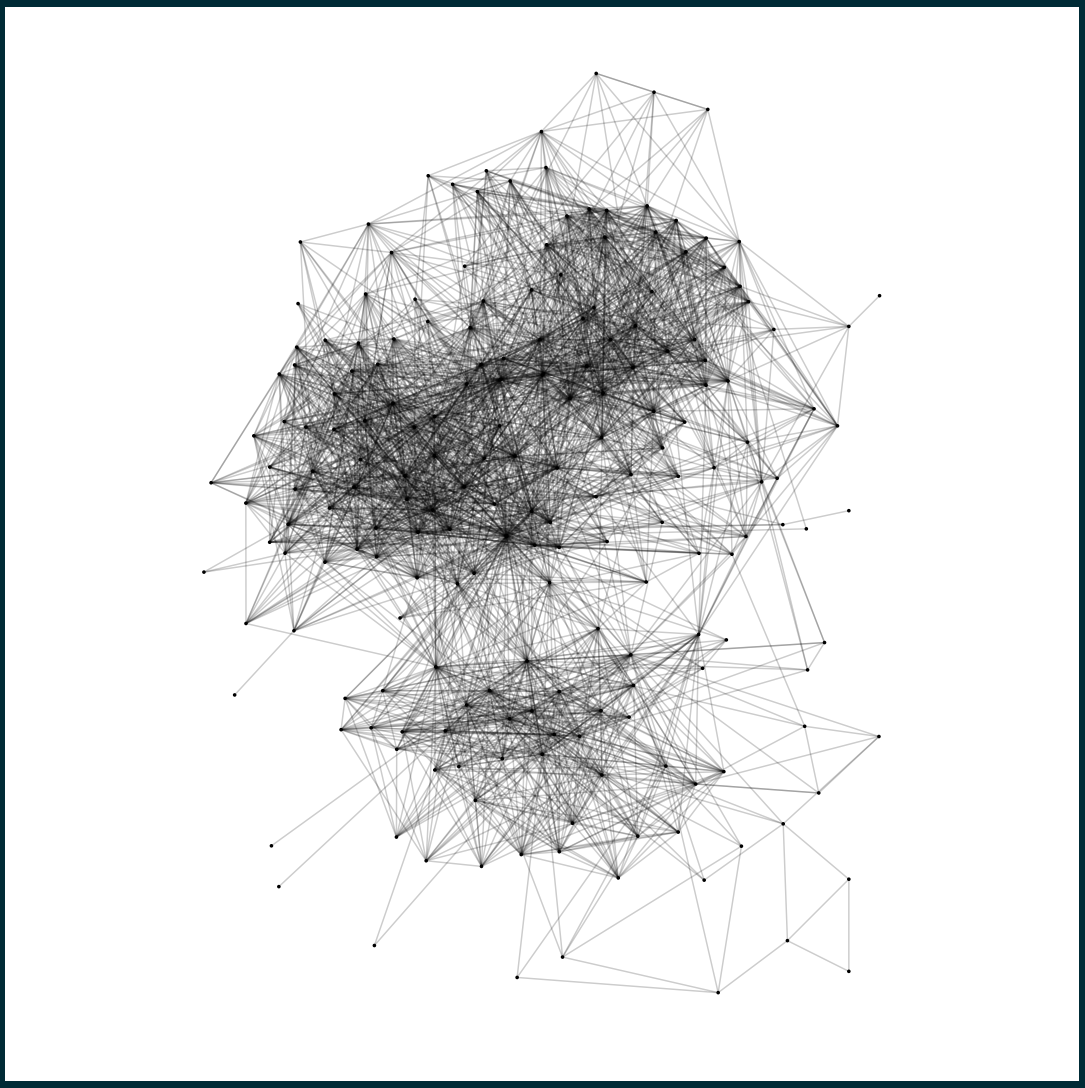
Mathematically, networks are represented as *graphs*. A graph is an object composed of a set V of *vertices* (also referred to as nodes) and a set of *edges* E. An edge is characterized by the fact that it connects two vertices together, which in mathematical terms translates to the fact that an edge can be written as a pair of vertices, or equivalently $E \subset \{(v_1, v_2) | v_1, v_2 \in V\}$. Many extensions of this model exist, for example edges may have a direction (*directed graph*), implying that $(v_1, v_2) \neq (v_2, v_1)$, or edges can carry a value (*weighted graph*) or multiple types of edges may exist, an extension called multiplex network that is the subject of Chapter 2.

In this first chapter we mainly follow the presentation given by Newman in Chapter 13 of [15] of standard unweighted and undirected networks, except for Sections 1.7.3, 1.10 and 1.11 that are addition absent of Newman's book. We further restrict our study to infinitely large networks as it allows for several convenient simplifications while still capturing important feature networks. We first present a general framework for the study of networks, the configuration model, and study several of its properties. Introducing the generating function mathematical tool, we then determine the low and high connectivity phases of networks, and their characteristics. We then apply the presented theory on three types of networks, Erdos-Renyi networks, scale-free networks and networks with exponential degree distribution. Finally, we present and study an algorithm generating fully connected networks.



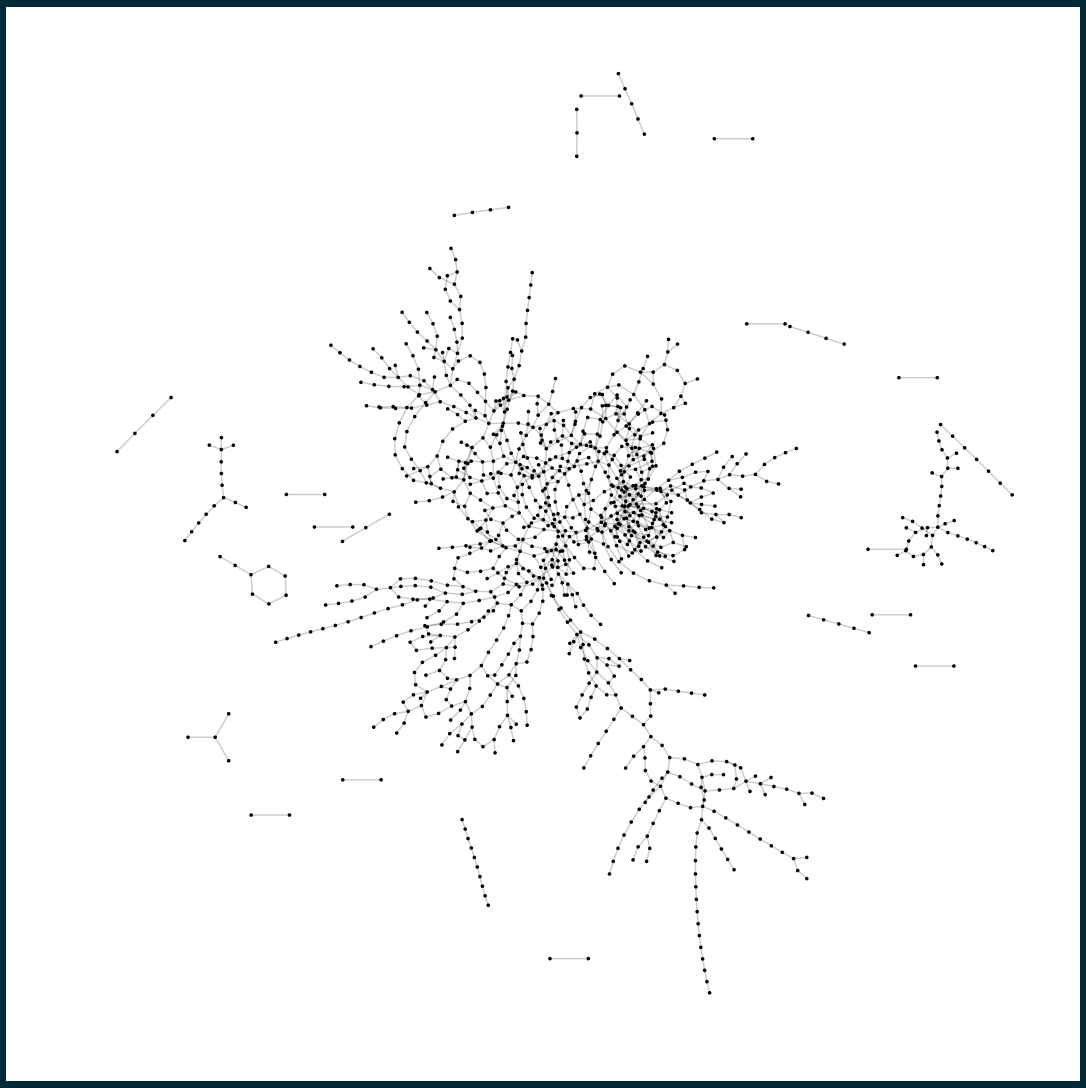
estern US powergrid⟩

FIGURE 1.1: Power grid network of the Western States of USA [1]. Nodes represent electrical stations (generator, transformator, substation) and edges represent power supply lines. Data retrieved from the Konect database [14] (Konect code UG).



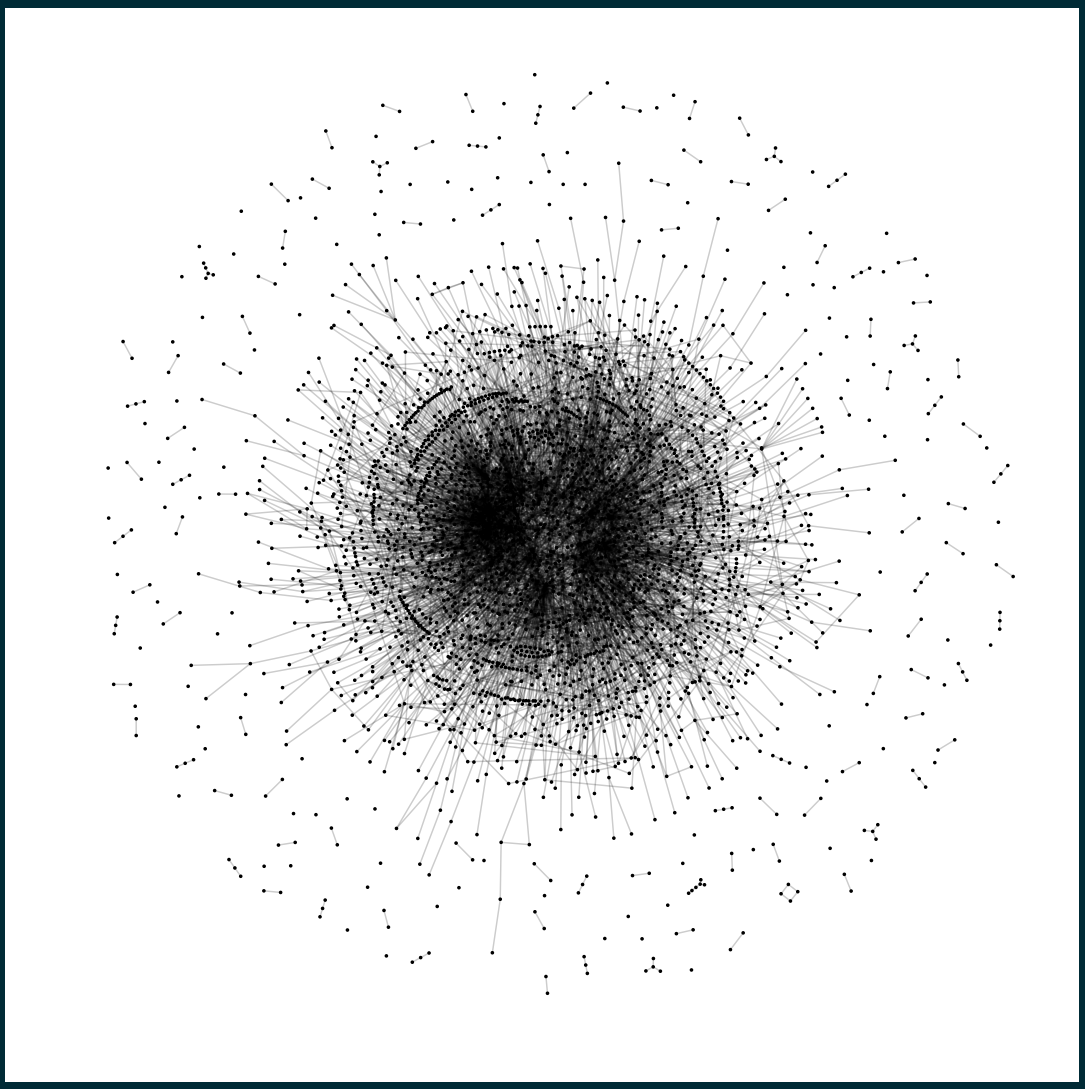
collaborations⟩

FIGURE 1.2: Collaboration network of jazz musicians as of 2003 [4]. Nodes represent musicians and edges represent the fact that the two musicians had played in the same band. Data retrieved from the Konect database [14] (Konect code JZ).



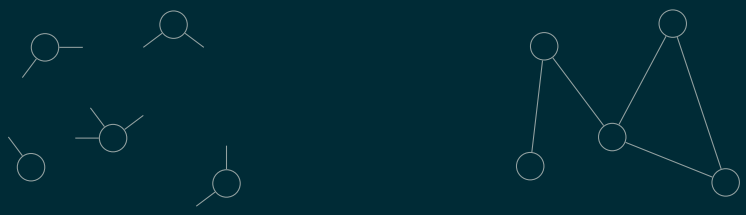
e: Network euroroad⟩

FIGURE 1.3: International E-road network, a network of road situated mainly in Europe [6]. Nodes represent cities and edges represent E-roads connecting them. Data retrieved from the Konect database [14] (Konect code ET).



human proteins⟩

FIGURE 1.4: Network of protein-protein interaction in humans [8]. Nodes represnt proteins and edges a binary interaction. Data retrieved from the Konect database [14] (Konect code MV).



(A) Nodes and theirs stubs (B) Stubs are randomly connected

FIGURE 1.5: Schematic representation of the configuration model.

1.2 Configuration model

Since we are interested in fundamental properties of networks, we need to abstract from the specificity of single networks. To do so we consider that networks are fully determined by their *degree distribution* $\{p_k\}$ where p_k is the probability for a node chosen randomly and uniformly to have degree k. Since knowing how a network can be constructed is useful both conceptually and to perform computation on properties of the network, we now present an algorithm called *configuration model* [15] that sample uniformly the space of all network with a given degree distribution.

Consider a network with n vertices with a given degree distribution $\{p_k\}$. If we cut every edges in two, every vertex keep a number of *stubs* (half-edges) equal to its degree. The resulting set of vertices and stubs is independent of the network structure, but common for all networks with the same degree distribution. The idea of this algorithm is thus to start from this state, a set of nodes with stub degree distribution $\{p_k\}$. Then each stub is connected to another chosen uniformly amongst the other stubs to form the edges of the network. By construction, the produced network has degree distribution $\{p_k\}$.

The fact that stubs are paired uniformly and independently is important in that it implies that the vertex reached by following a random edge does not depend, in probability, on the vertex at the starting end of the edge.

A computer algorithm able to simulate configuration model is rather straightforward to implement. First generate n random numbers $k_1, k_2, \ldots, k_n$ following the probability distribution p_k, one for each node. They are the respective degrees of the nodes. If the total degree $2m = \sum k_i$ is not even, add one node with degree 1, which has a negligible effect on the degree distribution but allows to have a even number of stubs and thus to connect all of them. Then build a list $\mathcal{S}_{\text{stubs}}$ of the stubs, where each stub is represented by the index of the node to which it is attached. As a consequence, index i is repeated k_i times in $\mathcal{S}_{\text{stubs}}$, and so for each i. This stub list is then to be shuffled uniformly[1]. Finally a node is create for each pairs of indices in $\mathcal{S}_{\text{stubs}}$, i.e. an edges is create between vertex s_i and s_{i+1} for $i = 1, 3, 5, \ldots, 2m-1$. This procedure ensures that the stubs are paired uniformly and thus that it is equivalent to the configuration model.

Finally since $\{p_k\}$ is a probability distribution, it is independent of the number of nodes n of the network, we can consider the limit for large n. In this thesis we only consider this limit as it allows several mathematical simplifications while still capturing the essential structure of a network. Moreover for sufficiently large networks, the difference between the large n limit and the actual network is small and can thus safely be neglected.

[1] Most programming languages provide this functionality.

1.3 Self-edges and multi-edges

Using the insight given by the algorithm, we can compute the probability that a node is connected to itself, thus making a so-called *self-edge*. First note that in the limit of large n, the number of vertices with degree k is equal to np_k and as a consequence the total number m of edges is equal to

$$m = \frac{1}{2}\sum_{k=0}^{\infty} nkp_k = \frac{n}{2}\mathbb{E}[\deg v]\,, \tag{1.1}$$

Total number of edg

with $\mathbb{E}[\ldots]$ denoting the expectation value

The probability that an edge connect node i with degree k_i to itself is equal to the number of way to connect both ends of the edge to this node $k_i(k_i-1)$, divided by the number of way an edge can be placed in the network $2m(2m-1)$. Multiplying this by the number of edges m gives the probability p_{ii} that node i is connected to itself

$$p_{ii} = \frac{k_i(k_i-1)}{2(2m-1)}, \tag{1.2}$$

{?}

The total number of self-edges is

$$\sum_{i=1}^{n} p_{ii} = \sum_{i=1}^{n} \frac{k_i^2 - k_i}{2(2m-1)} \tag{1.3}$$

{?}

$$= \frac{1}{2}\frac{\mathbb{E}\left[(\deg v)^2\right] - \mathbb{E}[\deg v]}{n\mathbb{E}[\deg v] - 1}. \tag{1.4}$$

{?}

Since we are considering a constant degree distribution p_k all expectations value remain constant when n grows, so the number of vertices having self edges goes to zeros as n becomes large and we can safely consider that the generated network has no self-edges at all.

Similarly, we find that the probability p_{ij} that two vertices i and j are connected is equal to

$$p_{ij} = m\frac{k_i k_j}{\binom{2m-2}{2}} = \frac{k_i k_j}{2m-1}. \tag{1.5}$$

{?}

The probability to have two or more edges between the vertices i and j is equal to the probability that i and j are connected and that they remain so after we remove one edge between them. The probability for them to be connected with one edge less is the same as p_{ij} but with one edge less in total and one stub less at both i and j, giving

$$\frac{(k_i-1)(k_j-1)}{2m-3}. \tag{1.6}$$

{?}

In consequence we find the probability to have at least two edges between i and j to be

$$p_{ij}\frac{(k_i-1)(k_j-1)}{2m-3} = \frac{k_i k_j (k_j-1)(k_i-1)}{2(2m-1)(2m-3)}. \tag{1.7}$$

{?}

Summing over all vertex pairs and dividing by two to avoid double counting the pairs, we find that the total expected number of multi-edges is

$$\frac{\sum_{ij} k_i k_j (k_j - 1)(k_i - 1)}{2(2m-1)(2m-3)} \approx \frac{1}{8m^2} \sum_i k_i(k_i - 1) \sum_j k_j(k_j - 1) \tag{1.8}$$

$$= \frac{1}{2} \left[\frac{\mathbb{E}\left[(\deg v)^2\right] - \mathbb{E}[\deg v]}{\mathbb{E}[\deg v]} \right]^2 , . \tag{1.9}$$

The approximation holds since in the limit of large n we have $2m-3 \approx 2m-3 \approx 2m$ as m scale proportionally to n. The number of multi-edges is thus asymptotically constant and we can therefore consider that the generated networks has no multi-edges, since the fraction of affected nodes goes to zero in the limit of large n.

A network having this two properties, absence of self-edges and of multi-edges, is said to be a *simple graph*. Since for n large enough all networks in the context of the configuration model have approximately these two properties, we always consider that the networks are simple graphs in the remaining of this thesis.

1.4 Uniformity of network space sampling

In the previous section we demonstrated that the configuration model asymptotically produces simple graphs in the limit of large n. This is necessary to prove the claims we make in Section 1.2 that the configuration uniformly sample the space of networks with a given degree distribution. To prove that claim is equivalent to prove that each different simple network appears with the same probability.

A *matching* of the stubs is a possible way to connect $2m$ distinguishable stubs. By construction the pair of stubs that are bound are chosen uniformly and thus each matching appears with the same probability than any other. However, in practice the stubs of a single vertex are not distinguishable, nor are the vertices of the same degree, so several matching leads to the same network. There are $k!$ way of arranging the stubs of a vertex of degree k, and $(np_k)!$ way of arranging the np_k nodes of degree k (in the limit of large n) resulting in a total of

$$\prod_{i=1}^{n} k_i! \prod_{k=0}^{\infty} (np_k)! \tag{1.10}$$

matchings corresponding to the same network, where k_i is the degree of vertex i[2]. This quantity only depends on the degree distribution, thus all networks with the same degree distribution have the same number of different matchings. Since each matching appears with the same probability, we can conclude that each network with a given degree distribution appears with the same probability in the configuration model.

In a sense the configuration model is therefore optimal for the point of view we adopt in this thesis. Indeed we consider a network fully determined by its degree distribution and all network with a common degree distribution are equal with regard to the configuration model as it sample them uniformly. We could in fact present the configuration model as a consequence of the requirement that we want a

[2] This value differs from the one given by Newman in [15] as he does not take in account the indiscernibility of vertices with the same degree, but this does not change the conclusion.

sampling method introducing no unnecessary constrain on the produced networks. See Section 15.2 of [15] or Section 2 of [16] for discussions of that point of view.

1.5 Excess degree distribution

As we will see below, while we consider that a network is fully determined by its degree distribution, considering vertices reached by following an edge gives valuable insights on the network structure. We call such vertex a *first neighbor* vertex and we denote $P_1(\dots)$ the probability associated with a first neighbor, while we denote $P_0(\dots)$ the probability associated with uniformly chosen vertices[3]. We can define the *excess degree distribution* $\{q_k\}$ as

$$q_k = P_1(\deg v = k+1), \qquad \forall k \in \mathbb{N}. \tag{1.11}$$ {?}

The probability q_k correspond to a first neighbor having degree $k+1$, or equivalently to the probability to have k edges other than the on used to reach the node in the first place, hence the name excess degree distribution.

The excess degree distribution can be computed explicitly by noting that a stub has the same probability to be connected to any if the other $2m-1$ stubs, thus the probability that this stub is connected to a given node of degree k is $k/(2m-1)$. Multiplying by the total number of node of degree k, np_k in the large n limit,gives the probability that a given node is attached to a node of degree k as

$$\frac{k}{2m-1} np_k = \frac{kp_k}{\mathbb{E}[\deg v]}. \tag{1.12}$$ {?}

Since q_k is the probability that a first neighbor has degree $k+1$, we can conclude

$$q_k = \frac{(k+1)p_{k+1}}{\mathbb{E}[\deg v]}. \tag{1.13}$$ qk as function of p

1.6 Generating functions

ting functions⟩? A powerful way of representing a discrete probability law (or any sequence of number) is the *generating function* of the distribution [17]. For a degree distribution $\{p_k\}$ it is defined as the function

$$g_0(z) = \sum_{k=0}^{\infty} p_k z^k. \tag{1.14}$$ Definition of g0

The fact that p_k is a probability distribution implies that

$$\sum_{k=0}^{\infty} p_k = 1, \tag{1.15}$$?Normalization of pk

hence the infinite sum in eq. (1.14) always converges for $|z| \leq 1$, which allows to numerically evaluate $g_0(z)$ in that range.

[3] In principle $P_j(\dots)$ could be defined, corresponding to the probability associated with vertices reached after following j edges.

Observe that the derivative with respect to z of $g_0(z)$ is

$$g_0'(z) = \sum_{k=0}^{\infty} p_k k z^{k-1}. \tag{1.16}$$ Derivative of g

Comparing with the definition of the expectation value we find

$$\mathbb{E}[\deg v] = \sum_{k=0}^{\infty} p_k k = g_0'(1). \tag{1.17}$$ Expectation val

Therefore, the generating function is sufficient to know the expectation value of a distribution. This result can be generalized: if we look at the second derivative of the generating function, by differentiating eq. (1.16), we find

$$g_0''(1) = \sum_{k=0}^{\infty} p_k k(k-1) = \sum_{k=0}^{\infty} p_k k^2 - \sum_{k=0}^{\infty} p_k k \tag{1.18}$$ {?}

$$= \mathbb{E}\left[(\deg v)^2\right] - \mathbb{E}[\deg v]\,. \tag{1.19}$$ Second derivati

Thus we can write

$$\mathbb{E}\left[(\deg v)^2\right] = g_0''(1) + g_0'(1) = \frac{\partial}{\partial z}\left(z g_0'(z)\right)\Big|_{z=1}, \tag{1.20}$$ Second moment a

which means that the second moment is fully determined by the generating as well. In fact such formula exists for all moments, but we do not present it here as in this thesis we use at most the first two moments.

All these properties do not depend on the degree distribution p_k to which the generating function is associated and apply to any discrete probability distribution. One other important distribution for network is the excess degree distribution $\{q_k\}$, we define its generating function g_1 as

$$g_1(z) = \sum_{k=0}^{\infty} q_k z^k. \tag{1.21}$$ Definition of g

Inserting eq. (1.13) in this definition, we get

$$g_1(z) = \frac{1}{\mathbb{E}[\deg v]} \sum_{k=0}^{\infty} (k+1) p_{k+1} z^k = \frac{g_0'(z)}{g_0'(1)}, \tag{1.22}$$ g1 as a functio

where we used eq. (1.17) and eq. (1.69) to obtain the last equality.

Similar to eq. (1.17) we find the expected value of the excess degree distribution as

$$g_1'(1) = \frac{g_0''(1)}{g_0'(1)} = \frac{\mathbb{E}\left[(\deg v)^2\right]}{\mathbb{E}[\deg v]} - 1, \tag{1.23}$$ {?}

where we inserted the expression in eq. (1.19) for the second derivative of $g_0(z)$. Recall that the excess degree distribution count one edge less than the degree of a vertex v, we can find the mean degree of first neighbors as

$$\mathbb{E}[\deg_1 v] = g_1'(1) + 1 = \frac{\mathbb{E}\left[(\deg v)^2\right]}{\mathbb{E}[\deg v]}, \tag{1.24}$$ Average degree

where we use $\deg_1 v$ to denotes the degree of a first neighbor. By introducing the variance of the degree distribution

$$\mathrm{var}[\deg v] = \mathbb{E}\left[(\deg v)^2\right] - \left(\mathbb{E}[\deg v]\right)^2 \tag{1.25}$$ {?}

$$\tag{1.26}$$ {?}

and replacing the second moment in eq. (1.24) we find a new expression for the average degree of first neighbors

$$\mathbb{E}[\deg_1 v] = \mathbb{E}[\deg v] + \frac{\mathrm{var}[\deg v]}{\mathbb{E}[\deg v]}. \tag{1.27}$$ {?}

Using this form, it is manifest that the average degree and the average degree of first neighbor differ. Precisely they difference depend on the variance of the degree distribution, which has one immediate striking consequence: since both the average degree and its variance are non negative, we have in general

$$\mathbb{E}[\deg_1 v] \geq \mathbb{E}[\deg v]\,. \tag{1.28}$$

Friends of friends

In other words, as Feld put it in the name of his original paper on the subject, this explains "why your friends have more friends than you do" [18]. Note that this feature is universal. We present it here in the context of the configuration model, but inserting the degree distribution of any network in the calculations presented leads to same result.

1.7 Giant connected component

ected component

1.7.1 Size of the GCC

An interesting property of a network is the presence and size of *connected components*. A set of nodes is said to be connected if there is a path formed of successive edges from any of its node to any other. All networks can be divided in connected components such that all nodes are element of exactly one component, as it is exemplified in fig. 1.6. The connectedness of network is crucial in many real world realisations of networks. In particular any logistic network, such as power grid networks, rail road network or the internet network, is functional only if it is able to transfer goods or services (electricity, passengers or informations) from any node to any other.

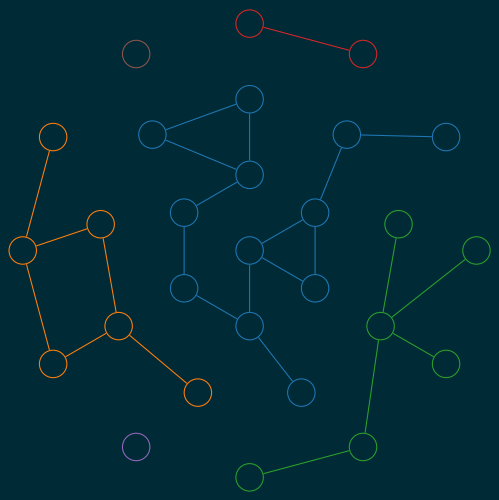
Insight on the property of networks can be found by studying the case where the fraction of the network occupied by a connected component does not vanish in the large n limit. Such component is called a *giant connected component* (GCC).

To get informations about the GCC we first compute the fraction occupied by an arbitrary connected component C of the network. We can write the probability S that a randomly chosen vertex is part of C as

$$S = 1 - P_0(w \notin C\ \forall w \in N(v)) \tag{1.29}$$ {?}

$$= 1 - \sum_{k=0}^{\infty} P_0(w \notin C\ \forall w \in N(v) | \deg v = k) P_0(\deg v = k). \tag{1.30}$$ {?}

The probability $P_0(w \notin C\ \forall w \in N(v)|\deg v = k)$ is the probability that no neighbors of a node with degree k are part of the component C. This in turn is the probability that by following the k edges going out of vertex w we we find each time



onnected components⟩

FIGURE 1.6: Scheme of a graph with six connected components, each drawn with a different color.

a node which is not part of C. However, as mentioned at the end of Section 1.2, by construction of the configuration model, the vertex at the end of an edge does not depend on the vertex at the other end. Therefore, we can consider the k edges attached to w to be independent, which allow us to write

$$P_0(w \notin C \; \forall w \in N(v) | \deg v = k) = [P_1(w \notin C)]^k = u^k. \tag{1.31}$$ Probability tha

In the last equality we introduce the probability u that a node reached by following an edge is not part of C,

$$u = P_1(w \notin C). \tag{1.32}$$?Definition of u

Noting that $P_0(\deg v = k) = p_k$, we finally find

$$S = 1 - \sum_{k=0}^{\infty} u^k p_k \tag{1.33}$$ {?}

$$= 1 - g_0(u). \tag{1.34}$$ {?}

We now have a compact expression for S in terms of u and the generating function of the degree distribution g_0. To determine u we observe that if a vertex v is not part of C, none of its neighbors is either. By definition of u this was already known for the node at the other end of the edge from where v was reached. Thus there are only $\deg v - 1$ to take in account when computing the probability v is not part of C. Using this observation we can express u as

$$u = P_1(w \notin C \; \forall w \in N(v)) \tag{1.35}$$ {?}

$$= \sum_{k=1}^{\infty} P_1(w \notin C \; \forall w \in N(v) | \deg v = k-1) P_1(\deg v = k-1) \tag{1.36}$$ {?}

$$= \sum_{k=0}^{\infty} u^k q_k \tag{1.37}$$ {?}

$$= g_1(u), \tag{1.38}$$ {?}

where we use eq. (1.31) again together with the fact that by definition of the excess degree distribution $q_k = P_1(\deg v = k+1)$.

We end up with two equations to describe the size of a connected component

$$S = 1 - g_0(u) \tag{1.39}$$ Single layer S fina

$$u = g_1(u). \tag{1.40}$$ Single layer u fina

If we can solve the second one we immediately get the size of the connected component C. These results were first derivated in an alternative form by Molloy and Reed in [19] and then presented in the framework of generating functions and extended by Newman *et al.* in [20][4]

However eq. (1.40) only gives u implicitly and its form strongly depends on the degree distribution, therefore no general analytical solutions can be given. From its graphical representation, shown in fig. 1.7, we can see that it has at most two solutions. First the trivial solution $u = 1$ is always present, as by the definition (1.21) of $g_1(z)$, we have $g_1(1) = 1$, implying $S = 0$. The components described by this regime are not giant connected components, as their relative size S vanish in the large n limit. We do not discuss it further here, but the full component size distribution can be analyzed [15, 21].

Then in some cases, another solution exists with $u < 1$ and $S > 0$. This solution correspond to the GCC. Moreover, if a GCC exists it is unique. To see that consider a network with two GCC with degree distribution such that $p_k \neq 0$ for some $k \geq 2$[5]. The probability P_c that a node v of degree k connects both GCC is strictly smaller than one. Indeed if P_c was one, it would imply that all nodes are part of the GCC and thus that the GCC is unique.

Finally, in the limit of large n, the number of vertices of degree k is np_k and hence the probability that no degree k vertex connect both GCC is

$$(1 - P_c)^{np_k}. \tag{1.41}$$ {?}

Since we assumed $p_k > 0$ and since $P_c < 1$, this probability goes to zero in the limit of large n. Therefore we can conclude that the GCC, if it exists, is unique.

The appearance of the GCC, can be seen as a phase transition from a low connectivity phase to a highly connected one. In this context it is natural to search the critical point $\mathcal{R}$ of this transition. To find it observe that, as can be seen on fig. 1.7, the $g_1(z)$ curve must "go below" the identity curve at $z = 1$ to create a non trivial solution. This requirement means that the slope of $g_1(z)$ at $z = 1$ must be greater than the slope of the identity, which is 1. In term of the derivative of $g_1(z)$ the condition is thus $g_1'(1) > 1$ to have a non trivial solution to eq. (1.40). Therefore the critical parameter $\mathcal{R}$ for which a GCC appears is defined by the boundary condition

$$g_1'(1) = 1. \tag{1.42}$$ Boundary condition

[4] The results and derivations that first appeared in [20] are fully reproduced, with additions and clarifications, by Newman in Chapter 13 of [15]. As such we prefer to cite the latter when no historical motivation dictate otherwise.

[5] If $p_k = 0$ for all $k \geq 2$, all nodes have degree 0 or 1 and thus the biggest component is at most of absolute size 2, making clear that no GCC can exist.

We can rewrite this condition in term of statistical properties of the network by using eq. (1.22), which yields

$$g_1'(1) = \left[\frac{\partial}{\partial z}\frac{g_0'(z)}{g_0'(1)}\right]_{z=1} \tag{1.43}$$

$$= \frac{g_0''(1)}{g_0'(1)} \tag{1.44}$$

$$= \frac{1}{\mathbb{E}[\deg v]}\left(\mathbb{E}\left[(\deg v)^2\right] - \mathbb{E}[\deg v]\right) \tag{1.45}$$

For the last equality we used the relation between the first and second moment of the degree distribution and the generating function, $g_0(z)$ given in eq. (1.17) and (1.19). Inserting in eq. (1.42) and rearranging we find

$$\mathbb{E}\left[(\deg v)^2\right] - 2\mathbb{E}[\deg v] = 0. \tag{1.46}$$

This condition for the critical point of the phase transition between the low and high connectivity phases was first proved in the general case by Molloy and Reed in [19]. In this paper, they also proved that in the high connectivity phase, where non trivial solution to eq. (1.40) exist, a GCC must actually be present and that it must be unique. For a proof of this using the generating function formalism see Section 13.6 of [15].

Finally eq. (1.40) can be solved numerically by noticing that the solution u is a fixpoint of the function $g_1(z)$. Since all coefficients in eq. (1.21) are non negative, $g_1(z)$ and all its derivative are positive for $z > 0$. As a consequence starting from $z_0 = 0$, the sequence z_k defined by the iteration

$$z_{k+1} = g_1(z_k) \tag{1.47}$$

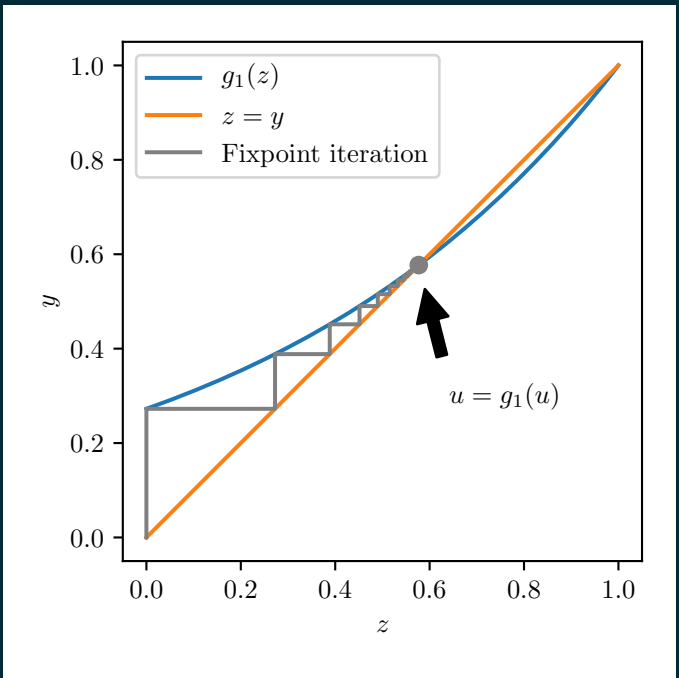
always converges toward the smallest solution of eq. (1.40).

1.7.2 Algorithm to find the connected components

Since we are working in the limit of large n, the networks we are generating and analysing have large n as well. Therefore the algorithm we use must be designed with some care to avoid consuming to much computing time, which would make them impractical to use. This motivate us to present the algorithm we use here.

To find all connected components of a network we proceed as follow

1. Add all nodes to the set $\mathcal{S}_{\text{unprocessed}}$ of unprocessed nodes.
2. Remove one node from $\mathcal{S}_{\text{unprocessed}}$ and add it to the set $\mathcal{S}_{\text{queued}}$ of queued nodes.
3. Start a new component C.
4. Remove one node from $\mathcal{S}_{\text{queued}}$ nodes and name it v.
5. Add v to the current component C.
6. Add all unprocessed neighbors of v to $\mathcal{S}_{\text{queued}}$.
7. If $\mathcal{S}_{\text{queued}}$ is not empty go to 4, else store the component C and continue.
8. If $\mathcal{S}_{\text{unprocessed}}$ is not empty go to 2, else terminate.



(u) graphically⟩

FIGURE 1.7: Graphical representation of eq. (1.40) for an Erdos-Renyi network with $c = 1.3$. Gray solid line represent the successive steps of the fixpoint iteration.

This algorithm goes through each node exactly once and is thus of complexity $\mathcal{O}(n)$ which is the optimal complexity since all nodes must be associated to a component.

However, to ensure that the algorithm is fast we must be to efficiently find all neighbors of a node. To do that we represent the network as an *adjacency list*: each node is given an index i and the adjacency list A_i contains all the neighbors of i. The whole network is thus represented as a list of adjacency list $A = (A_1, \dots, A_n)$.[6]

Other representation of networks exist, which are more convenient and efficient for some purposes. However we do not use them in this thesis and we stick to the adjacency list representation.

1.7.3 Degree distribution in the GCC

tion in the GCC⟩

Per Bayes theorem we have for two random events A and B

$$P(A|B) = P(B|A)\frac{P(A)}{P(B)}. \tag{1.48}$$

?Bayes theorem?

We can apply it to compute the probability r_k that a vertex in the GCC has degree k

$$r_k = P\left(\deg v = k | v \in GCC\right) \tag{1.49}$$

{?}

$$= P(v \in GCC | \deg v = k)\frac{P(\deg v = k)}{P(v \in GCC)} \tag{1.50}$$

{?}

$$= \frac{p_k}{S}\left(1 - P(v \notin GCC | \deg v = k)\right) = \frac{p_k}{S}(1 - u^k). \tag{1.51}$$

Degree distribution

[6] For better performance during the creation of the network, our implementation goes a step further and actually request sorted adjacency lists.

To get the final result we used eq. (1.31). This result was previously presented more generally and following a very different path by Bauer and Bernard in [16].

Therefore we see that considering a vertex in the GCC biases the probability that it has degree k by a factor $(1-u^k)/S$ as compared to choosing a vertex uniformly in the network. Since both u and S are smaller than 1, the net effect is to lower the proportion of low degree vertices in the GCC and thus to increase the proportion of high degree vertices. This result can be understood intuitively since in the configuration model the stubs are connected independently. Therefore each edge of a node increases the probability that this node is connected to the GCC, making high degree node over represented in the GCC.

1.8 Erdos-Renyi networks

Erdos-Renyi networks refer to networks build using the first model of random graphs studied in the literature. This model was probably first introduced in 1951 by Solomonoff and Rapoport [22] and latter popularized (and probably independantly rediscovered) by Gilbert [23] and Erdos and Renyi [24], the latter giving it their name as they further studied it in the subsequent years [25, 26]. The model grows a network as follow: for each pair of nodes i and j, an edge is added with fixed probability p.

To find the degree distribution in such network, first notice that the expected degree, usually denoted c for Erdos-Renyi network, is equal to the number of other vertices multiplied by the probability to be connected to each of them, i.e.

$$c = \mathbb{E}[\deg v] = (n-1)p. \tag{1.52}$$ {?}

We generally use c as the parameter defining an Erdos-Renyi network, rather than p, since it makes sense to keep c constant when n becomes large, rather than p.

The probability for a node to have degree k is

$$p_k = \binom{n-1}{k} p^k (1-p)^{n-1-k}, \qquad \forall k \in \mathbb{N}. \tag{1.53}$$?Poisson degree

We recognize a binomial degree distribution for $n-1$ trials with success probability p. In the limit of large n we can approximate such distribution by a Poisson distribution with parameter $c = (n-1)p$

$$p_k \approx \frac{c^k}{k!} e^{-c}. \tag{1.54}$$ pk for Erdos-Re

The parameter c is the expected degree in the network, it is proportional to $n-1$ rather than n because we only tries to bind each vertex with each other, and not with itself, making a total of $n-1$ trials.

Inserting the degree distribution in the definition of the generating function (1.14), we recognize Taylor series representing the exponential function and thus we get

$$g_0(z) = e^{-c} \sum_{k=0}^{\infty} \frac{z^k c^k}{k!} = e^{-c} e^{cz} = e^{c(z-1)}. \tag{1.55}$$ g0 for ER netwo

Taking the derivative and inserting in eq. (1.21) yields the generating function for the excess degree distribution

$$g_1(z) = e^{c(z-1)}, \tag{1.56}$$ g1 for ER netwo

which appears to be equal to $g_0(z)$. This can be used to determine the critical mean degree $\mathcal{R}$ for which a GCC appears by using eq. (1.42), namely

$$1 = g_1'(1) = \mathcal{R}. \tag{1.57}$$ {?}

To conclude this general discussion of Erdos-Renyi network, we observe that to simulate such network we do not need to generate Poisson distributed network. Indeed, if we pick $2m$ node uniformly, a node is picked k time with a probability following the Binomial distribution

$$\binom{2m}{k}\left(\frac{1}{n}\right)^k\left(1-\frac{1}{n}\right)^{2m-k}, \tag{1.58}$$ {?}

which has expectation value $2m/n$. According to eq. (1.1) this is equal to c and hence approximating by a Poisson distribution gives

$$\frac{c^k}{k!}e^{-c}. \tag{1.59}$$ {?}

This expression is the same as eq. (1.54) and thus picking vertex randomly make them appear with the same distribution as the degree distribution. Therefore to generate the list of stubs $\mathcal{S}_{\text{stubs}}$ presented in Section 1.2, we can simply uniformly choose $2m$ vertex indices between 1 and n and simulate the network from there.

Using this algorithm, we simulate Erdos-Renyi networks for a range of mean degree c and plot the result on fig. 1.8(A) together with the numerical solution of eq. (1.39) and an outline of the position of the critical point $\mathcal{R} = 1$. As expected, simulations run with low number of nodes drift significantly from the large n theoretical prediction, while simulation with large n closely match it.

1.9 Scale-free networks

-free networks)? The concept of scale-free network was first introduced in 1999 by Barabási and Albert in [27] to describe networks poorly approximated by the Erdos-Renyi model. In this paper they introduce a new network model, the *preferential attachment model* in order to account for the fact that in many networks, several nodes tend to have much more neighbors that what the Erdos-Renyi model predicts, an effect sometimes referred as the degree distribution having a *heavy-tail*. The preferential attachment model builds a network by iteratively adding vertices to it. Vertices are labelled such that vertex v_i is added at step i. When a vertex is added, the probability to connect the new vertex and a vertex already present with degree d is

$$\Pi(d) = \frac{d}{\sum_{i=1}^{k-1}\deg v_i}. \tag{1.60}$$ Preferential attach

The rationale behind this formula is that in many system being prevalent tends to give an advantage to becoming even more prevalent. For example, a scientific paper which has been cited many times is likely to be cited often, because it is well-known. Moreover, we see that in average, the model add

$$\sum_{i=1}^{\infty}\Pi(\deg v_i) = 1 \tag{1.61}$$ {?}

edge each time a vertex is added. Using eq. (1.1) we have that the average degree is

$$\mathbb{E}[\deg v] = 2\frac{m}{n}, \tag{1.62}$$ {?}

with m the total number of edges and n the total number of vertices. Adding as much edges as vertices is therefore useful to keep the average degree (and in this precise case also the degree distribution [27]) approximatively constant during the process.

An important property of networks grown using the preferential attachment model is that their degree distribution follows a power law with exponent α

$$p_k = \frac{k^{-\alpha}}{\zeta(\alpha)}, \qquad \forall k \in \mathbb{N}^*, \tag{1.63}$$?Power law degre

with $p_0 = 0$ and where $\zeta(\alpha)$ is the Riemann zeta function defined as

$$\zeta(\alpha) = \sum_{k=1}^{\infty} k^{-\alpha}. \tag{1.64}$$ Definition Riem

At first exemplified with an actor collaboration network, the US power grid and the world wide web in [27], degree distributions exhibiting power law tails have since been found in various types of networks including microbial networks [28], metabolic networks [29] or citation networks [30, 31]. A recent analysis of available network data [32] even found that about 11% of the 927 data analysed display "strong statistical evidence" to have power law tails with exponent between 2 and 3, which make power law an important property of real networks, albeit not universal. In this paper, "strong statistical evidence" is defined as degree distribution such that the tail of the distribution has at least 50 nodes and for at least 50% of the network

1. The p-value of the goodness of fit measure used must be greater than 0.1,
2. The power law is favored over all the alternative laws tested (exponential, log-normal and Weibull distributions).

Due to the prevalence of power law tailed networks, many property of power law networks has been studied [33, 34, 35, 36, 37, 38], as a first step to understanding the underlying property of heavy-tail degree distributions.

However, despite having wide application, the power law distribution is mathematically more challenging than the previous example as its generating function can not be represented in term of elementary function. The best we can do is introducing the *polylogarithm* $\mathrm{Li}_\alpha(z)$

$$\mathrm{Li}_\alpha(z) = \sum_{k=1}^{\infty} k^{-\alpha} z^k. \tag{1.65}$$ Definition of p

The polylogarithm is a generalization of the Riemann zeta function, as can be seen by the fact that for $z = 1$ we have

$$\mathrm{Li}_\alpha(1) = \sum_{k=1}^{\infty} k^{-\alpha} = \zeta(\alpha). \tag{1.66}$$ Polylogarithm o

With that notation, the generating function $g_0(z)$ for scale-free networks can be written

$$g_0(z) = \sum_{k=1}^{\infty} \frac{k^{-\alpha}}{\zeta(\alpha)} z^k = \frac{\mathrm{Li}_\alpha(z)}{\zeta(\alpha)}. \tag{1.67}$$ g0 for scale-free n

While no simple expression exist for the polylogarithm, its formal definition (1.65) is sufficient to compute its derivative

$$\frac{\partial}{\partial z}\mathrm{Li}_\alpha(z) = \sum_{k=1}^{\infty} k^{-\alpha+1} z^{k-1} = \frac{1}{z}\mathrm{Li}_{\alpha-1}(z)\,. \tag{1.68}$$?Derivative of the p

We therefore find

$$g_0'(z) = \frac{\mathrm{Li}_{\alpha-1}(z)}{z\zeta(\alpha)}. \tag{1.69}$$ Derivative of g0 fo

Using eq. (1.17) and(1.66), we find the mean degree as

$$\mathbb{E}[\deg v] = g_0'(1) = \frac{\zeta(\alpha-1)}{\zeta(\alpha)}. \tag{1.70}$$ Mean degree in scal

Inserting these two last equations in eq. (1.22) we get the generating function for the excess degree distribution.

$$g_1(z) = \frac{\mathrm{Li}_{\alpha-1}(z)}{z\zeta(\alpha-1)}. \tag{1.71}$$ g1 for scale-free n

Since we are not considering the analytic continuation of the zeta function, but only its real sum description given in eq. (1.64), $\zeta(\alpha)$ diverges for $\alpha < 1$. We can therefore distinguish three cases. First for $\alpha \leq 2$, eq. (1.70) diverges and the mean degree always goes to infinity as the network grows. The name scale-free comes from that fact, as there is no *typical scale* for the degrees and greater degrees can always happen with probability too high to be ignored.

In the case $2 < \alpha \leq 3$, the mean degree is finite, but its second moment is not. It can be seen by calculating explicitly using using eq. (1.20),

$$\mathbb{E}\left[(\deg v)^2\right] = \frac{\partial}{\partial z}\left(z g_0'(z)\right)\Big|_{z=1} = \frac{\partial}{\partial z}\left(\frac{\mathrm{Li}_{\alpha-1}(z)}{\zeta(\alpha)}\right)\Big|_{z=1} \tag{1.72}$$ {?}

$$= \frac{\mathrm{Li}_{\alpha-2}(1)}{\zeta(\alpha)} = \frac{\zeta(\alpha-2)}{\zeta(\alpha)}. \tag{1.73}$$ {?}

The divergence of the second moment come from the fact that the factor $\zeta(\alpha-2)$ diverges for $\alpha \leq 3$. In consequence, the variance of the degree distribution diverges, which means the degree are very widely distributed. Moreover, since in this regime the mean degree is finite, the fact that the second moment diverges implies that the first neighbor degree distribution (1.24) diverges as well. This make scale-free networks a rather extreme example of the inequality (1.28) stating that the average degree of first neighbor is greater than the average degree of random vertices.

Finally for $\alpha > 3$, the two first moment are finite and only higher moments diverge, which does not implies any peculiar behavior.

Before presenting the results concerning the scale-free networks, we must address some technical details. First, while the polylogarithm $\mathrm{Li}_\alpha(z)$ is easy to manipulate analytically, computing its numerical values precisely is challenging. Hopefully, some well established libraries implement it, in this thesis we use the implementation provided by the `mpmath` Python package [39]. This does not solves all problems however, since the excess degree distribution (1.71) depends on $\mathrm{Li}_\alpha(z)/z$ that can not be directly computed using that form for $z = 0$, and may display significant numerical error for z close to zero. To solve that issue, we approxmate this quantity using the few first terms of the polylogarithm for small z, precisely

$$\frac{\mathrm{Li}_\alpha(z)}{z} \approx 1 + \frac{z}{2^\alpha} + \frac{z^2}{3^\alpha}, \qquad \text{for } z < \varepsilon, \tag{1.74}$$ Small z polylog

for some branching value ε. In this thesis we us $\varepsilon = 10^{-8}$ since the difference between the true value of $\mathrm{Li}_\alpha(z)/z$ and its approximation (1.74) is

$$z^3 \sum_{k=0}^{\infty} \frac{z^k}{(k+4)^\alpha} < z^3 \sum_{k=0}^{\infty} z^k = \frac{z^3}{1-z} < \frac{\varepsilon^3}{1-\varepsilon} \approx 10^{-24}, \tag{1.75}$$ {?}

which is numerically zero relative to 1.0 for standard double precision floating point number. Hence, this approximation introduces no error greater that the numerical inaccuracy.

Then in order to apply the configuration model, power law distributed numbers need to be generated. It appears such generator is rather uncommon, so we implemented our own based on the algorithm presented in Appendix D of [40].

All technical issues being now out of the way, we can now solve eq. (1.39) for scale-free network and simulate scale-free network. Results for a range of exponents α are shown in fig. 1.8. Note that as opposed to what we did for Erdos-Renyi and geometric networks, we do not use the mean degree of the network as parameter here, as its relation (1.70) to α is not simple to invert.

1.10 Exponential networks

xponential networks⟩ In the previous section, we described how the concept of preferential attachment intuitively make sense: the advantage gained by a node helps it gain even more advantage. However, the model is still rather arbitrary, for example the form of the preference $\Pi(k)$ given in eq. (1.60) could in principle be changed to any function and this change leads to degree distribution different than power law [41].

Also the requirement that the added node always receives the new edges can also be relaxed, and this change is sufficient to move the degree distribution from a power law to an discrete exponential law [42] $p_k \propto \lambda^k$ where λ is a parameter strictly smaller than 1. Interestingly an other model has been found to end up with the same degree distribution [43].

For convenience, we prefer to write the degree distribution in the form of a geometric law with parameter $p < 1$

$$p_k = (1-p)^{k-1} p. \tag{1.76}$$ Geometric degre

This is equivalent to the exponential form $p_k \propto \lambda^k$ only if degree 0 nodes are forbidden. If not the degree distribution $\tilde{p}_k = p_{k+1}$ can be defined, that is similar in

most aspect. We prefer the version given in eq. (1.76) as it yields slightly simpler expression and result for $\tilde{p}_k$ can be found by applying the exact same computations.

The generating function for the degree distribution is given by

$$g_0(z) = \sum_{k=1}^{\infty} p_k z^k = \sum_{k=1}^{\infty} p(1-p)^{k-1} z^k \tag{1.77}$$ {?}

$$= \frac{p}{1-p}\left[\sum_{k=0}^{\infty} ((1-p)z)^k - 1\right] \tag{1.78}$$ {?}

$$= \frac{p}{1-p}\left[\frac{1}{1-(1-p)z} - 1\right] \tag{1.79}$$ {?}

$$= \frac{pz}{1-(1-p)z}. \tag{1.80}$$ g0 for exponential

We use the formula for geometric series to get rid of the infinite sum. The derivative of $g_0(z)$ is

$$g_0'(z) = \frac{p}{[1-(1-p)z]^2}. \tag{1.81}$$ {?}

Using eq. (1.17) we can then find the mean degree as

$$c = g_0'(1) = \frac{1}{p}, \tag{1.82}$$ {?}

that can be used as an alternative parameter to determine the degree distribution. The next step is to insert these results in eq. (1.22) to find $g_1(z)$ as

$$g_1(z) = \frac{g_0'(z)}{g_0'(1)} = \frac{p^2}{[1-(1-p)z]^2}. \tag{1.83}$$ g1 for exponential

Finally by taking the derivative of $g_1(z)$ and putting it in eq. (1.42), we find the boundary condition

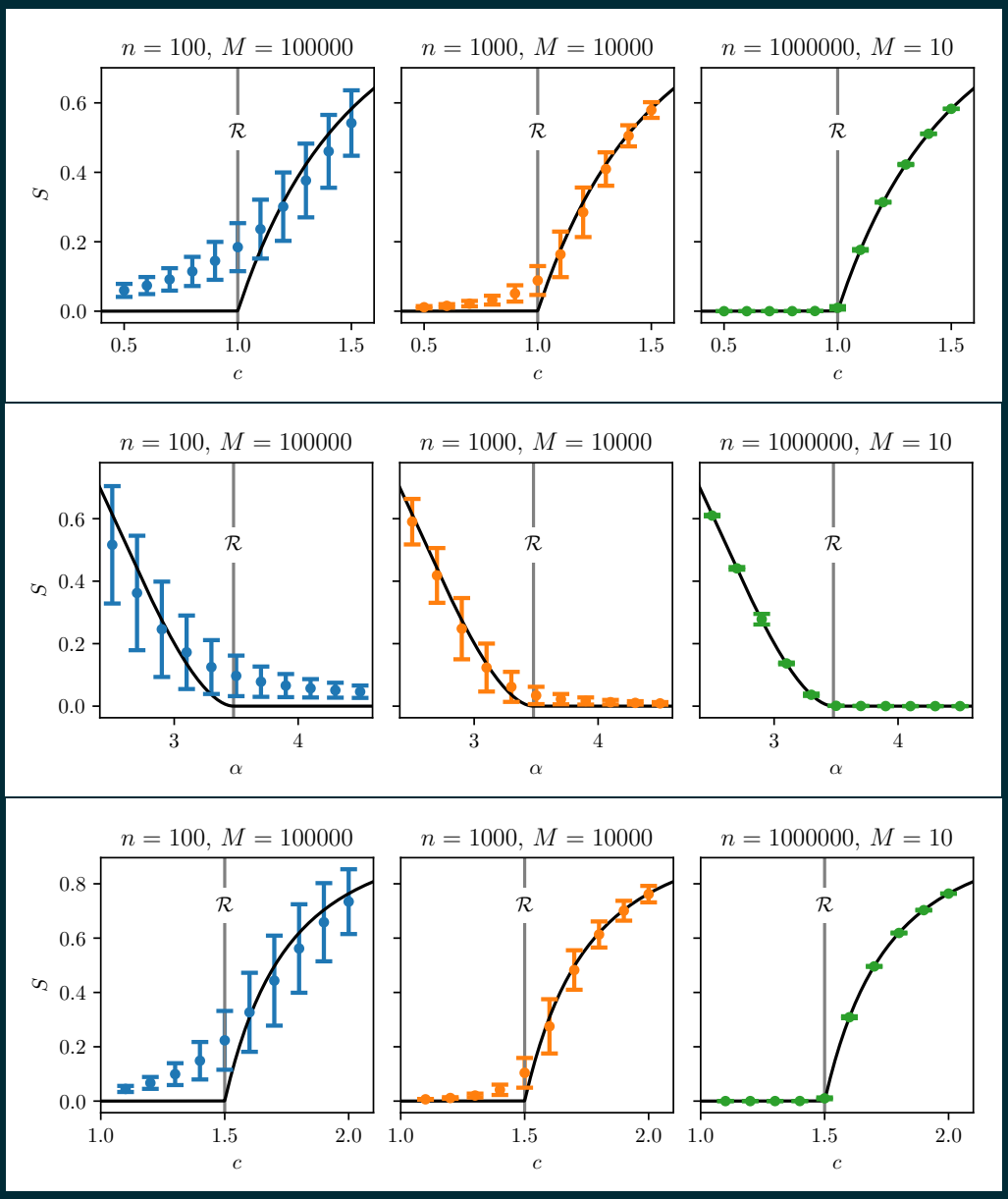
$$1 = g_1'(1) = \left.\frac{2p^3\left(\frac{1}{p}-1\right)}{[1-(1-p)z]^3}\right|_{z=1} \tag{1.84}$$ {?}

$$= 2\left(\frac{1}{p}-1\right), \tag{1.85}$$ {?}

that we can rewrite as

$$c = \frac{1}{p} = \frac{3}{2}. \tag{1.86}$$ {?}

The size S of the GCC for geometric networks computed using eq. (1.40) as well as result of simulations are shown in fig. 1.8(B). To generate number following a geometric degree distribution, we use the utility provided by the `Distributions.jl` Julia package.



layer S simulation⟩

FIGURE 1.8: Numerical solution for S of eq. (1.39) and (1.40) (solid black line), together with results on simulated networks. The critical point $\mathcal{R}$ computed using eq. (1.42) is shown as a vertical gray line. Results of simulations are average over M independently generated networks for various values of the number of nodes n in the network. (A) Poisson degree distribution.(B) Power law degree distribution. (C) Geometric degree distribution.

1.11 Generating connected networks

nected networks⟩

1.11.1 Algorithm

The knowledge of the degree distribution in the GCC can be used generate a connected component of a given degree distribution r_k as it has been previously presented in [44]. To do so, we first determine a degree distribution p_k fulfilling eq. (1.51) for some target degree distribution r_k. Then we generate a network with degree distribution p_k using the configuration model. Finally we take its GCC as our connected network. By construction the vertices in the GCC will have degree distribution r_k. Determining the factors p_k is not immediate however since u is an unknown which is itself a function of p_k. We propose an algorithm to determine it numerically.

First we isolate p_k from eq. (1.51) to get

$$p_k = S\pi_k(u), \quad \text{with} \quad \pi_k(z) = \frac{r_k}{1 - z^k} \tag{1.87}$$ {?}

Inserting this in the expression (1.40) for u, we get

$$u = \frac{\sum_{k=1}^{\infty} k\pi_k(u)u^{k-1}}{\sum_{k=1}^{\infty} k\pi_k(u)}. \tag{1.88}$$ Fixpoint equation f

Therefore u is a fixpoint of the function

$$\mu(z) = \frac{\sum_{k=1}^{\infty} k\pi_k(z)z^{k-1}}{\sum_{k=1}^{\infty} k\pi_k(z)}, \tag{1.89}$$ Defition of mu

which is fully determined by the GCC degree distribution r_k. Note that for $r_1 = 0$, we have the fixpoint $u = 0$ and $p_k = r_k$ for all k. This is consistent with the fact that small component of a network produced with the configuration model have a probability 0 to have loop [15]. Indeed if $p_1 = 0$ all components must have loops, therefore the probability to have small components is 0 as well.

On the other hand $r_1 > 0$ implies $u > 0$. To approximate its value we define the sequence $u_{j+1} = \mu(u_j)$, with $u_0 = r_1$. This sequence will converge toward u for large j. A proof of this statement is given in Appendix A.1.

In practice we can not deal evaluate infinite sums numerically, thus we need to choose a cutoff index K for the sums such that

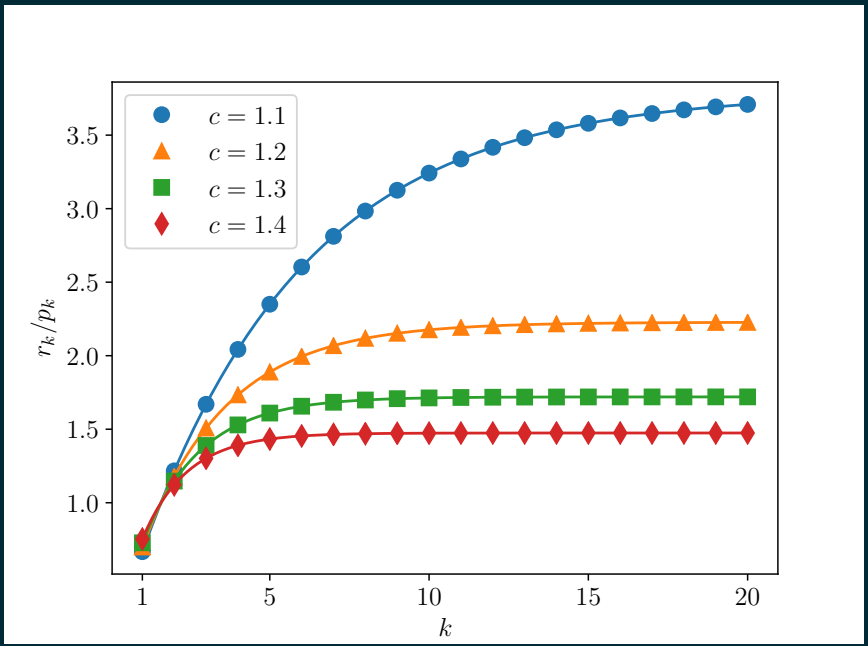
$$\sum_{k=K+1}^{\infty} k\pi_k(u) \ll 1. \tag{1.90}$$ {?}

For scale-free network with exponent smaller than 2 for example, this sum always diverges and thus this method is not applicable.

Once u is approximated, we can compute the first K probabilities p_k, which is sufficient to sample random numbers between 1 and K with relative probability p_k. If K is chosen such that $r_k << 1$ for $k > K$, the degree distribution in the GCC closely approximate the distribution r_k.

1.11.2 Erdos-Renyi reconstruction

In order to test the algorithm presented, we choose the target connected degree distribution r_k to be the degree distribution of the GCC of an Erdos-Renyi network.



enyi reconstruction)

FIGURE 1.9: Bias factor r_k/p_k for r_k being the degree distribution of the GCC of an Erdos-Renyi network with various mean degree c and cutoff constant $K = 10000$. The p_k have been determined using the algorithm presented in the text. Solid line is the expected value $(1 - u^k)/S$ for the bias factor.

It is then expected that the reconstructed p_k closely approximate a Poisson degree distribution.

The probability p_k to have degree k in an Erdos-Renyi network is given in eq. (1.54). Using eq. (1.40) and (1.39) to find u and S yield everything we need to be able to determine the GCC degree distribution r_k from eq. (1.51). We can therefore use the algorithm on these r_k.

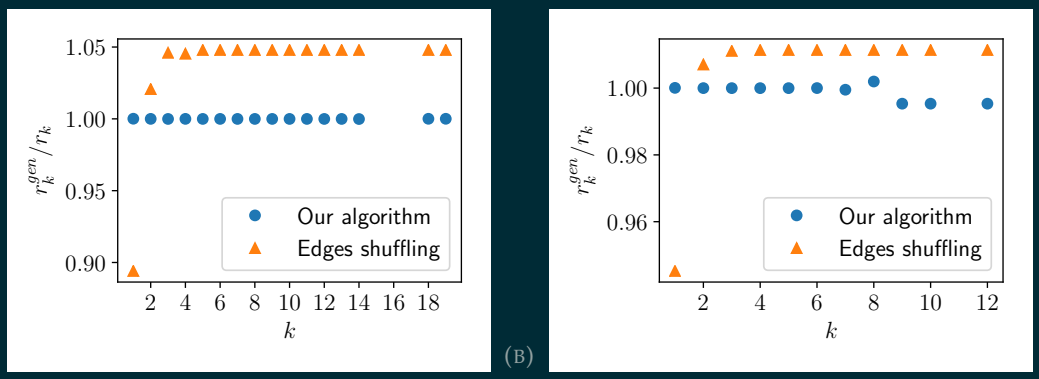
When computing S for the original Poisson distribution, we should however be cautious, as the reconstructed p_0 will always be 0. The expected resulting degree distribution, correctly normalized, is therefore

$$p_k = \frac{c^k}{k!}\frac{1}{e^c - 1}. \tag{1.91}$$ {?}

The expected bias ratio r_k/p_k is shown for various mean degree c and a cutoff constant $K = 10000$ in fig. 1.9 together with the same value computed from the algorithm presented above. As it can be seen, the agreement is very good. During the computations it has been observed that the closer the mean degree is to the critical value $c = 1$, the slower the fixpoint iteration converges.

1.11.3 Real world networks

As an example of use of the algorithm presented, we apply it to real networks. We choose two by design connected network from the Konect network database [14], the powergrid of the western states of the United States [1] and the road network of the state of California [5] (the codes of the networks in the Konect database are respectively UG and RO). The power grid network is represented in fig. 1.1, while



: Real examples⟩

FIGURE 1.10: Plot of the ratio of the generated connected degree distribution r_k^{gen} to the target degree distribution r_k taken from a real network for our algorithm and the edges shuffling method. Missing values correspond to degree with probability 0 to appear. (A) Western US power-grid network. (B) California road network.

the Californian road network is not shown due to his high number of vertices ($n \approx 2 \times 10^6$).

The connected degree distribution r_k is taken to be the empirical degree distribution of the real network considered. As a consequence, the cutoff constant K is the maximal degree appearing in the network. Then, to sample the resulting degree distribution p_k, we simply take the number of vertex d_k of degree k to be the closest integer to np_k, where n is the total number of nodes. In the examples presented we use $n = 10^7$.

We compare the degree distribution r_k^{gen} of the GCC of the newly generated network with the target distribution r_k by looking at the ratio r_k^{gen}/r_k. Resulting ratios are shown in fig. 1.10, together with the results obtained by taking the GCC of the reshuffled network.

End of chapter 1 (for ref in todo list

Chapter 2

Multiplex networks

tiplex networks⟩

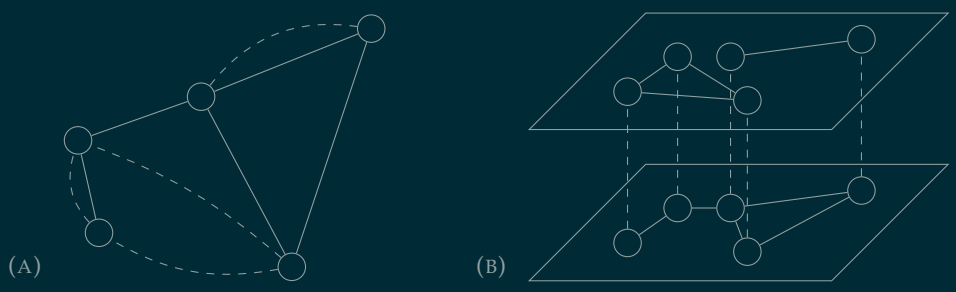
2.1 Introduction

The concept of network can be extended by allowing different types of edges between the nodes of a network. Such generalized networks are called *multiplex networks* or *multi layers networks*. Among the real world example of such multiplex we find the various kind of connections between cities: transport connections (road, railroads, airlines), information connections (phone lines, internet connections) or supply connections (electricity, water).

However, such network can be more conveniently represented in a slightly different but equivalent way. We consider each type of edge to generate its own network. If we started with L different type of edges, we therefore get L networks, all sharing the same set of nodes. Conceptually it is similar to stack L networks on top of each other, each network forming a *layer* of the composed multi layers network (hence the name). An illustration of both representation of multiplex networks is shown in fig. 2.1 for a two layers network.

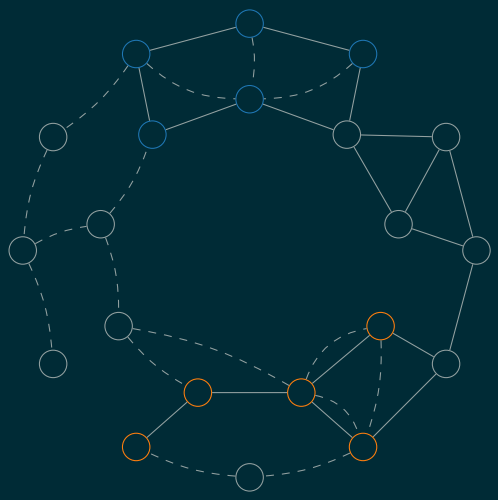
This representation also has the advantage of being readily generalizable to interdependent networks: rather than having nodes depend on exactly one node in each other layers, it may depends on nodes in other layer with some probability. This generalization allows to model any type of interdependent networks, which are ubiquitous in the real world, examples including interdepent infrastructure networks [45], such as power-grids depending on their control network [46] and multiple networks of species interacting [47]. However, despite the strong modelling power of such interdependent networks, we restrict ourselves to the study of multiplex networks in this thesis.

Thinking in term of layers also allows to reuse all quantities define in the context of single layer networks, such as the generating functions. We the extended quantity



tiplex networks⟩

FIGURE 2.1: Possible representations of a two layers network. (A) A network with two kinds of edges, straight solid lines and curved dashed ones. (B) Two networks in separate layers with corresponding nodes.



ected is not viable⟩

FIGURE 2.2: Intersection of connected components of two layers of a multiplex networks. Edges in each layer are represented respectively with solid straight lines and curved dashed lines. The intersection is composed of the colored nodes, however there is no path in the intersection connecting the blue and orange nodes. Therefore the intersection results in two distinct viable clusters.

the same way as the original one, adding a subscript i to its symbol to denote the fact that it applies to layer i, except if the symbol already has a subscript. In this case we add a superscript (i).

Finally, we only consider that each layer of a multiplex network is independant to each other. This is most likely not true for real networks, but we restrict ourself to this case for simplicity.

2.2 Giant viable cluster

ant Viable Cluster⟩?

2.2.1 Size of the GVC

ection: Size of GVC⟩

For single layer network, we are interested in the GCC as it is common for a system to be operational only if it is connected to the whole corresponding network. In the context of multiplex networks, the concept of connected component is extended by introducing *viable clusters*. A viable cluster is defined as a subset of the multiplex network such that in every layer there is a path between any pair of vertices in that subset. Note that the path between two vertices must be totally included in the viable cluster, therefore the intersection of connected components is not in general a viable clusters as can be seen in fig. 2.2. This property is the rationale behind the definition of viable cluster as it is the minimal requirement to be sure that the resulting cluster are connected.

Similar to the connected components in the single layer case, a viable cluster may occupy a non zero fraction of the network in the large n limit. Such viable cluster is called the *giant viable cluster* (GVC). Its existence and size are the main subjects of the discussion of this Chapter.

To be able to determine the size of the GVC, first let $g_0^{(i)}$ and $g_1^{(i)}$ be the generating functions of respectively the degree and the excess degree distribution in layer i. Moreover define u_i as the probability that a vertex reached after following an edge in layer i is not part of the giant viable cluster.

Then if we pick a vertex v at random the probability S that it is part a viable cluster V can be written as

$$S = P_0\left(\bigcap_{i=1}^{L} \exists w \in N_i(v)\ w \in V\right), \tag{2.1}$$

that is vertex v must have a neighbor in V in each layer. By requiring that the layers are independent from one others, we can rewrite S as a product

$$\begin{aligned}
S &= \prod_{i=1}^{L} P_0^{(i)}\left(\exists w \in N_i(v)\ w \in V\right) && (2.2)\\
&= \prod_{i=1}^{L}\left[1 - P_0^{(i)}\left(w \notin V\ \forall w \in N_i(v)\right)\right] && (2.3)\\
&= \prod_{i=1}^{L}\left[1 - \sum_{k=0}^{\infty} P_0^{(i)}\left(w \notin V\ \forall w \in N_i(v) | \deg v = k\right) p_k^{(i)}\right] && (2.4)\\
&= \prod_{i=1}^{L}\left[1 - \sum_{k=0}^{\infty} P_1^{(i)}\left(v \notin V | \deg v = k\right) p_k^{(i)}\right] && (2.5)\\
&= \prod_{i=1}^{L}\left[1 - \sum_{k=0}^{\infty} u_i^k p_k^{(i)}\right] && (2.6)\\
&= \prod_{i=1}^{L}\left[1 - g_0^{(i)}(u_i)\right]. && (2.7)
\end{aligned}$$

Multiplex GCC size

Here the reasoning is similar to the one leading to size of a connected component (1.39) in the single layer case presented in Section 1.7. We can also find u_j by a similar reasoning. First note that $1 - u_j$ is the probability that a vertex reached by following an edge in layer j is in V. Which, due to the independence of the layers, can as before be written in the form

$$\begin{aligned}
1 - u_j &= P_1^{(j)}\left(\bigcap_{i=1}^{L} \exists w \in N_i(v)\ w \in V\right) && (2.8)\\
&= \prod_{i=1}^{L} P_1^{(j)}\left(\exists w \in N_i(v)\ w \in V\right). && (2.9)
\end{aligned}$$

Moreover since the layers are independent, the fact that we reached v by following an edge in layer j to reach vertex v is irrelevant in all other layers. However in layer j this means that the degree distribution follows the distribution $q_k^{(j)}$ rather than $p_k^{(j)}$. Putting this together we get

$$\begin{aligned}
1 - u_j &= \left[1 - \sum_{k=0}^{\infty} u_j^k q_k^{(j)}\right] \prod_{\substack{i=1\\ i\neq j}}^{L}\left[1 - \sum_{k=0}^{\infty} u_i^k p_k^{(i)}\right] && (2.10)\\
&= \left[1 - g_1^{(j)}(u_j)\right] \prod_{\substack{i=1\\ i\neq j}}^{L}\left[1 - g_0^{(i)}(u_i)\right]. && (2.11)
\end{aligned}$$

Multiplex u final

This equation has a trivial solution $u_j = 1$ for all layers, since by definition of the generating function (1.14) we have

$$g_0^{(j)}(1) = g_1^{(j)}(1) = 1. \tag{2.12}$$ {?}

This also implies that the size of component V is 0. Hence to have a GVC it is necessary to find a non trivial solution to eq. (2.11).

The size of the GVC as given by eq. (2.7) and (2.7) was first given in that form by Son *et al.* in [48]. In this paper Son *et al.* also treat a generalization in the form of the interdependent networks mentioned in the introduction of this chapter.

To rephrase this result in a mathematically more tractable way, we introduces some more notation. First consider that the degree distributions of the network's layers are determine by a finite parameter vector $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_N)$, where N is the number of parameters and does not need to equal the number of layers N. A typical parameter could be the mean degree of an Erdos Renyi graph for example.

Then, as in the single layer case, it is useful to express the u_j as fixpoint of some function. In addition to providing a simple way of numerically solving eq. (2.11), it can provide valuable information on the presence or absence of solutions as presented in Section 2.4.

To make the fixpoint nature of the u_j manifest, we define the functions

$$\psi_j(\boldsymbol{\lambda}, \mathbf{z}) = 1 - \left[1 - g_1^{(j)}(z_j)\right] \prod_{\substack{i=1 \\ i \neq j}}^{L} \left[1 - g_0^{(i)}(z_i)\right], \tag{2.13}$$ Definition of p

where $\mathbf{z}$ is the L dimensional vector

$$\mathbf{z} = (z_1, z_2, \ldots, z_L). \tag{2.14}$$ {?}

While we do not write it explicitly to avoid overcharging the notation, it is understood that the generating functions $g_0^{(i)}$ depend on the parameter vector $\boldsymbol{\lambda}$.

Introducing a L dimensional vector $\mathbf{u}$ and a L dimensional function ψ as respectively

$$\mathbf{u} = (u_1, u_2, \ldots, u_L) \tag{2.15}$$ {?}

$$\psi(\mathbf{z}) = (\psi_1(\mathbf{z}), \psi_2(\mathbf{z}), \ldots, \psi_L(\mathbf{z})), \tag{2.16}$$ {?}

we can compactly rewrite eq. (2.11) as

$$\mathbf{u} = \psi(\boldsymbol{\lambda}, \mathbf{u}), \tag{2.17}$$ uvec as a fixpo

which explicitly show that $\mathbf{u}$ is a fixpoint of the function ψ.

Finally, with this notation, the trivial solution to eq. (2.11) is written as

$$\mathbf{u}_T = (1, 1, \ldots, 1). \tag{2.18}$$?Definition of m

We have now gathered nearly as much informations about the GVC than we have about the GCC from the study we do of it in Section 1.7. However, while we have determined that the number of solutions to eq. (2.17) indicates if the networks is in a low or high connectivity phase, finding the boundary between the two is not as straightforward as it was in the single layer case. We return to that question in details

in Section 2.3, after we discuss one parameter multiplex networks, that allows easier visualization, and an algorithm to find the GVC.

2.2.2 Single parameter multiplex networks

meter networks)? Graphical representation of solutions of multi dimensional equations such as eq. (2.11) is challenging. To avoid this problem we introduce single parameter networks. We define them as multiplex networks in which each layer has the same degree distribution. By letting the degree distribution depend on only one parameter λ, the dimension of the problem reduces to one, as in the single layer case, while still providing important inside in the difference brought by allowing multiple layers.

Mathematically, imposing constant degree distribution across layers implies that the generating functions are constant as well,

$$g_0^{(i)}(z) = g_0(z), \tag{2.19}$$ {?}

$$g_1^{(i)}(z) = g_1(z), \qquad \forall i = 1, \dots, L. \tag{2.20}$$ {?}

Moreover by definition u_j is the probability that, in the large n limit, the probability that a vertex reached by following an edge is not part of the GVC. Having the same degree distribution in each layer, thus implies that the $u = u_j$ is the same for each layer.

Plugging these two observations into eq. (2.11) yields the single one dimensional equation for u

$$u = 1 - [1 - g_1(u_j)] \prod_{\substack{i=1 \\ i \neq j}}^{L} [1 - g_0(u_i)] \tag{2.21}$$ {?}

$$= 1 - [1 - g_1(u)] \left[1 - g_0(u)\right]^{L-1}. \tag{2.22}$$ Single parameter u

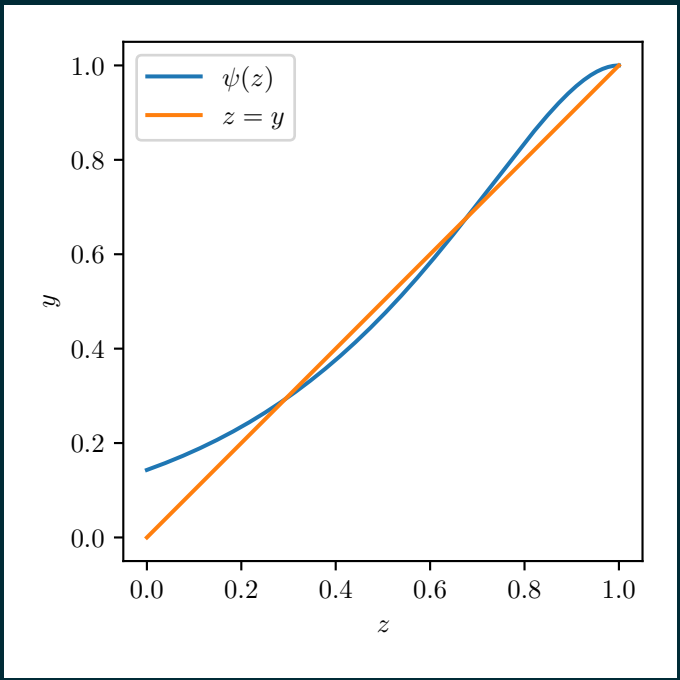
Similarly using eq. (2.7) we find for S,

$$S = [1 - g_0(u)]^L. \tag{2.23}$$ Single parameter u

Figure 2.3 shows a graphical representation of eq. (2.23) for a two layer single parameter Erdos Renyi network. The main difference from the one layer case is that $\psi(z)$ has an inflexion point, as opposed to $g_1(z)$ in eq. (1.40) that never does. As a consequence, eq. (2.23) can have more than two solutions and the condition (1.42) does not hold. Indeed, while a new solution still appears when $\psi(z)$ is tangent to the identity line, this tangency point no longer appears at $z = 1$. This also indicates that the phase transition may be discontinuous for multiplex networks, a fact that we numerically verify for several examples in Section 2.4.5.

2.2.3 Algorithm to find the viable clusters

Similar to what we do in the single layer case, we would like to simulate and manipulate large networks. Therefore, algorithms must be carefully designed to be able to complete in a reasonable amount of time. The algorithm to find the GVC is significantly more complicated that the one presented in Section 1.7.2 because the paths we must consider to determine the GVC are dependent on the GVC itself: a path is valid if it is fully enclosed in the GVC, but a vertex is part of the GVC only if a path exist to all other vertices in the GVC.



ly for single param⟩

FIGURE 2.3: Graphical representation of eq. (1.40) for multiplex network of two layers, both of which are Erdos Renyi graph with mean degree $c = 2.6$.

The main algorithm, adapted from [49], goes as follow:

1. Add all nodes the set $\mathcal{S}_{\text{unprocessed}}$ of unprocessed nodes.
2. Create a new viable connected component guess $\mathcal{S}_{\text{guess}}$ as an independent copy of $\mathcal{S}_{\text{unprocessed}}$.
3. Remove a node from $\mathcal{S}_{\text{unprocessed}}$ and name it v.
4. For each layer, exclude from $\mathcal{S}_{\text{guess}}$ all nodes that are not part of the same connected component as v in the subgraph spanned by the current $\mathcal{S}_{\text{guess}}$.
5. If the previous step modified $\mathcal{S}_{\text{guess}}$ go back to it, else add $\mathcal{S}_{\text{guess}}$ to the list of viable components and remove all nodes in it from $\mathcal{S}_{\text{unprocessed}}$.
6. If $\mathcal{S}_{\text{unprocessed}}$ is empty terminate, else go to 2.

When the algorithm terminate, it has found all viable components of the multiplex network. To find the connected component in step 4 we use the procedure presented in Section 1.7.2.

However, in this state, the algorithm may be crippling slow. To understand why observe that the most expensive viable component to determine tend to the smallest. Indeed for small components, step 4 may have to be repeated a large number of times until it has excluded most of the other vertices in the network. Moreover to be close to the critical point, where all layers are highly connected but no GVC emerges, is an aggravating factor. Not only the high connectedness of the layers makes step 4 longer to compute, but also the GCC intersection may be large and only composed of very small components that must individually be found among a large GCC intersection. For example for a two-layer multiplex network with each layer being an Erdos-Renyi network with $c = 2.3$ and $n = 10000$, we found that in average

(over 10 runs) the intersection of the GCC occupied $74.3(6)\%$ of the network, but no GVC exists and every of the viable component is composed of exactly one node. This makes the determination of the GVC prohibitively costly for large networks.[1]

Since we are only interested in the size of the GVC, we can mitigate this problem by excluding all single layer connected components that are too small to be considered a GVC on their own. Only valid contender are the GCC of each layers, so we can immediately restrict our search to the intersections of the layers' GCC.

We then determine all connected components in the subgraph spanned by the vertices in the GCCs intersection. All components which occupy a fraction of the network smaller than a fixed level ε are discarded. In this thesis we use $\varepsilon = 0.1\%$. The process is then repeated on the remaining vertices until no new vertices are excluded. Formally this preprocessing algorithm reads

1. Compute the GCC in every layer of the network. Put each vertex that is in every GCC in $\mathcal{S}_{\text{initial}}$.
2. For every layer determine all connected components of the subgraph spanned by the vertices in $\mathcal{S}_{\text{initial}}$.
3. Remove from $\mathcal{S}_{\text{initial}}$ all nodes that are part of a connected component occupying a fraction of the network smaller than ε in any layer.
4. If the previous step modified $\mathcal{S}_{\text{initial}}$ go to 2, else terminate.

The only change that must be done to the main algorithm is to modify step 1 such that only the vertices in $\mathcal{S}_{\text{initial}}$ are added to $\mathcal{S}_{\text{unprocessed}}$.

2.3 Critical region

for multilayer⟩ As mentioned at the end of Section 2.2.1, we would like to be able to determine under which conditions a multiplex network is in his high connectivity phase. To do this we want to find the critical region $\mathcal{R}$ in the parameter space between the low and high connectivity phase.

Before we can proceed to determining the critical region, we need to introduce some new concepts. First we define the residual functions

$$f_j(\boldsymbol{\lambda}, \mathbf{z}) = 1 - z_j - \left[1 - g_1^{(j)}(z_j)\right] \prod_{\substack{i=1 \\ i \neq j}}^{L} \left[1 - g_0^{(i)}(z_i)\right] \tag{2.24}$$?Definition fj?

$$= \psi_j(\boldsymbol{\lambda}, \mathbf{z}) - z_j. \tag{2.25}$$ {?}

The functions $\psi_j(\boldsymbol{\lambda}, \mathbf{z})$ are the ones introduced in eq. (2.13). To get a compact representation of our problem, we gather all functions f_j in the high dimensional function

$$f : \mathbb{R}^N \times \mathcal{I}^L \to \mathbb{R}^L \tag{2.26}$$ {?}

$$(\boldsymbol{\lambda}, \mathbf{z}) \mapsto F(\boldsymbol{\lambda}, \mathbf{z}) = (f_1(\boldsymbol{\lambda}, \mathbf{z}), f_2(\boldsymbol{\lambda}, \mathbf{z}), \ldots, f_L(\boldsymbol{\lambda}, \mathbf{z})) \tag{2.27}$$?Definition F?

$$= \psi(\boldsymbol{\lambda}, \mathbf{z}) - \mathbf{z}, \tag{2.28}$$ {?}

where $\mathcal{I} = [0, 1]$.

[1] Baxter *et al.* do not mention the problem in the presentation of the algorithm on which we base our own [49], presumably because they do not present result of simulations in their paper.

Then, since the functions $g_0^{(i)}$ and $g_1^{(i)}$ are analytic with respect to the z_i, the function

$$f_{\boldsymbol{\lambda}} : \mathcal{I}^L \to \mathbb{R}^L \tag{2.29}$$
$$\mathbf{z} \mapsto f(\boldsymbol{\lambda}, \mathbf{z}), \tag{2.30}$$

is continuously differentiable for all parameter vectors $\boldsymbol{\lambda}$. Therefore we can define Jacobi matrix $J_{\boldsymbol{\lambda}}(\mathbf{z})$ of $F_{\boldsymbol{\lambda}}$ as having coefficients

$$[J_{\boldsymbol{\lambda}}(\mathbf{z})]_{ij} = \frac{\partial f_i(\boldsymbol{\lambda}, \mathbf{z})}{\partial z_j} = \frac{\partial \psi_i(\boldsymbol{\lambda}, \mathbf{z})}{\partial z_j} - \delta_{ij}. \tag{2.31}$$

Now that we have introduced the quantities we will need, we can go back to the matter of solving eq. (2.11). In term of the function F, this is equivalent for a given $\boldsymbol{\lambda}$ to finding $\mathbf{u} \in \mathcal{I}^L$ that is a zero of F, i.e.

$$f(\boldsymbol{\lambda}, \mathbf{u}) = 0. \tag{2.32}$$

Since this equation can be solve for all $\boldsymbol{\lambda}$, it is tempting to try to express the $\mathbf{u}$ solving it has a function of $\boldsymbol{\lambda}$, say $h(\boldsymbol{\lambda})$. However, eq. (2.11) is too complicated to be invertible, so we only know $h(\boldsymbol{\lambda})$ through the implicit equation

$$f(\boldsymbol{\lambda}, h(\boldsymbol{\lambda})) = 0, \quad \forall \boldsymbol{\lambda} \in U. \tag{2.33}$$

Now assume that f (and not only $f_{\boldsymbol{\lambda}}$) is continuously differentiable and that we are interested in $h(\boldsymbol{\lambda})$ such that it takes the value $\mathbf{u}^*$ for some $\boldsymbol{\lambda}^*$, $h(\boldsymbol{\lambda}^*) = \mathbf{u}^*$. The second condition is necessary to determine h uniquely since eq. (2.11) may have multiple solutions, as shown in fig. 2.3.

If these conditions are fulfilled, the implicit function theorem tell us that in small neighbourhood U around $\boldsymbol{\lambda}^*$ we can find an uniquely defined continuous function $h(\boldsymbol{\lambda})$ that solve eq. (2.33) if the Jacobi matrix $J_{\boldsymbol{\lambda}^*}(\mathbf{u}^*)$ is not singular, that is if its determinant is not zero.

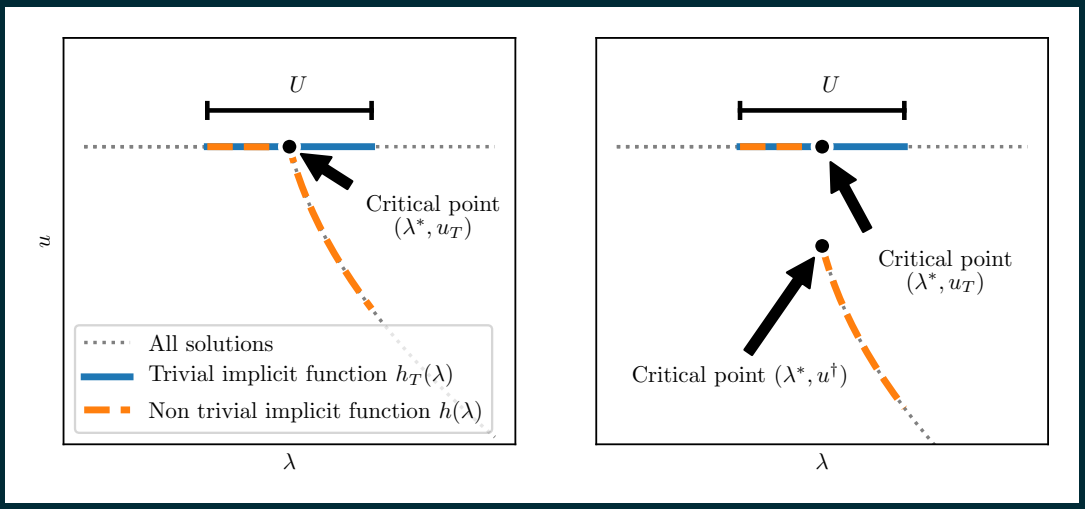
The result in which we are interested here comes from the contrapositive of this statement, namely that if for all neighbourhoods U of $\boldsymbol{\lambda}^*$ we can not find a uniquely defined continuous function h, then the determinant of the Jacobi matrix $J_{\boldsymbol{\lambda}}(\mathbf{u})$ must be zero, namely

$$\det\left[J_{\boldsymbol{\lambda}^*}(\mathbf{u}^*)\right] = 0, \tag{2.34}$$

We now prove that such situations arise if $\boldsymbol{\lambda}^*$ is a critical point $\boldsymbol{\lambda}^c$ of the phase transition between the absence and existence of a GVC, and therefore that eq. (2.34) is a sufficient condition to find the critical region of such phase transition. This condition has previously been stated, without proof in [49].

First consider a continuous phase transition. As shown in fig. 2.4, on one side of the phase transition occurring at $\boldsymbol{\lambda}^c$ one solution exists, while on the other at least two do. Since the phase transition is continuous, for any U open containing $\boldsymbol{\lambda}^c$ we can define two distinct functions on U that fulfil eq. (2.33), the trivial $h_T(\boldsymbol{\lambda}) = \mathbf{u}_T$ and another function h corresponding to the non trivial solutions, with $h(\boldsymbol{\lambda}^c) = h_T(\boldsymbol{\lambda}^c) = \mathbf{u}_T$. So the function h of the implicit function theorem is not uniquely defined, regardless of the neighborhood U chosen, and thus $\det\left[J_{\boldsymbol{\lambda}}(\mathbf{u}_T)\right] = 0$.

On the other hand, let consider a discontinuous phase transition at $\boldsymbol{\lambda}^c$. For any neighbourhood U of $\boldsymbol{\lambda}^c$ there are two sequences $(\boldsymbol{\lambda}_n, \mathbf{u}_n)$ and $(\boldsymbol{\eta}_m, \mathbf{v}_m)$ with $\boldsymbol{\lambda}_n, \boldsymbol{\eta}_m \in$



ase transitions⟩

FIGURE 2.4: Left panel: Scheme of a continuous phase transition. Right panel: Scheme of a discontinuous phase transition.

U for all $m, n \in \mathbb{N}$ such that

$$\lim_{n\to\infty} (\boldsymbol{\lambda}_n, \mathbf{u}_n) = (\boldsymbol{\lambda}^c, \mathbf{u}^\dagger) \quad \text{with } \mathbf{u}^\dagger \neq \mathbf{u}_T \tag{2.35}$$

$$\lim_{m\to\infty} (\boldsymbol{\eta}_m, \mathbf{v}_m) = (\boldsymbol{\lambda}^c, \mathbf{u}_T) \tag{2.36}$$

$$f(\boldsymbol{\lambda}_n, \mathbf{u}_n) = 0 \quad \forall n \tag{2.37}$$

$$f(\boldsymbol{\eta}_m, \mathbf{v}_m) = 0 \quad \forall m. \tag{2.38}$$

If we assume that an unique continuous function h solving eq. (2.33) exists, we would have

$$h(\boldsymbol{\lambda}_n) = \mathbf{u}_n \quad \forall n \tag{2.39}$$

$$h(\boldsymbol{\eta}_m) = \mathbf{v}_m \quad \forall m. \tag{2.40}$$

The continuity of h would furthermore imply

$$h(\boldsymbol{\lambda}^c) = \lim_{n\to\infty} h(\boldsymbol{\lambda}_n) = \lim_{n\to\infty} \mathbf{u}_n = \mathbf{u}^\dagger, \tag{2.41}$$

but also

$$h(\boldsymbol{\lambda}^c) = \lim_{m\to\infty} h(\boldsymbol{\eta}_m) = \lim_{m\to\infty} \mathbf{v}_m = \mathbf{u}_T. \tag{2.42}$$

Since $\mathbf{u}_T \neq \mathbf{u}^\dagger$ as shown inf fig. 2.4, this gives raise to the contradiction $h(\boldsymbol{\lambda}^c) \neq h(\boldsymbol{\lambda}^c)$. Therefore our assumption must be false and no continuous function h can be defined to solve eq. (2.33). So finally, we have $\det\left[J(\boldsymbol{\lambda}^c, \mathbf{u})\right] = 0$, $\mathbf{u}$ being either $\mathbf{u}_T$ or $\mathbf{u}^\dagger$.

This conclude the proof, as it demonstrate that fulfilling eq. (2.34) is always a necessary condition for the existence of a phase transition, regardless of the fact that it is or not continuous. Let stress here that in eq. (2.34), the $\boldsymbol{\lambda}^*$ and $\mathbf{u}^*$ must also solve eq. (2.17). Therefore to find the critical region $\mathcal{R}$, we must find a all couple $(\mathbf{u}, \boldsymbol{\lambda})$

that solve the system

$$\begin{cases} \mathbf{u} = \psi(\boldsymbol{\lambda}, \mathbf{u}) \\ \det\left[J_{\boldsymbol{\lambda}}(\mathbf{u})\right] = 0. \end{cases} \tag{2.43}$$

Boundary condit

Note that in the sequel we assume that in the high connectivity phase a GVC actually exists, but the theory presented in this chapter does not prove it, and to our knowledge no proof of this fact has been published.

Before going further, we briefly discuss the single layer case $L = 1$ in this framework. In that case the Jacobi matrix J reduces to the scalar quantity

$$J_{\boldsymbol{\lambda}}(u) = \frac{\partial}{\partial z}\left(g_1(z) - z\right)|_{z=u} = g_1'(u) - 1. \tag{2.44}$$

{?}

Therefore the condition for the boundary $\det\left[J_{\boldsymbol{\lambda}}(u)\right] = 0$ becomes

$$g_1'(u) = 1. \tag{2.45}$$

{?}

As expected this condition is the same as the one given in eq. (1.42) when discussing single layer networks.

2.4 Interval arithmetic

nterval arithemetic⟩

2.4.1 Motivation

In order to verify that eq. (2.34) indeed gives the critical region $\mathcal{R}$ for a multiplex network, we need to introduce an independent numerical method to approximate it. This comes down to find the number of solutions of eq. (2.11) in the whole parameter space and then draw the boundary of the regions with a constant number of solution. However this leads to two problems.

First, standard algorithms to find zeros of functions are not guaranteed to find all zeros of a function. These algorithms usually find one solution at a time, the one being found depending on the initial guess solution provided by the user, no global information about the solution space being computed during the process. This causes immediate problem for our purpose, since missing a solution of eq. (2.11) may change our estimation of the boundary region by leading to the conclusion that some regions of the parameter space are part of trivial domain while they are not.

Secondly, the standard algorithms can only deal with one point of the parameter space at a time. This forces us to restrict our search on a discrete set of parameters. If this set is ill chosen, we may miss important features of the critical region. This second point is rather minor compared to the first. We indeed expect the critical region to be smooth and therefore restricting our analysis on any lattice should reasonably approximate it. We still mention this caveat however, since it can be solved using the same framework needed to solve the first issue.

Moreover, in Section 1 we observed that for multiplex networks the phase transition displayed by eq. (2.11) seems to be discontinuous. However, to prove this fact we need to find some region between the trivial and non trivial solutions that can be proved to be exempt of any solution. As outlined above, standard zero finding algorithm are not suitable for such purpose. Others methods exist to prove the continuity of discontinuity of a phase transition, including using graphical solution of the kind presented in Fig. 2.3 [48] or analytical arguments [49]. However, the former

only apply to single parameter problem and we were unable to find any analytical argument determining the order of the phase transition, hence the use of numerical methods.

To solve all these problems, we therefore introduce the field of *interval arithmetic* and methods thereof that allows to rigorously guarantee presence or absence of solutions of equations in whole region of space.

2.4.2 Theoretical foundation

oundation of IA⟩

First of all an *interval* Z is defined a set of the form

$$Z = [a, b] = \{x \in \mathbb{R} | a \leq x \leq b\}. \tag{2.46}$$ {?}

The set of all intervals is denoted as $\mathbb{IR}$. The N-dimensional equivalent of an interval is an *interval box* $\mathbf{Z}$, defined as the Cartesian product of N intervals,

$$\mathbf{Z} = Z_1 \times Z_2 \times \cdots \times Z_N, \quad Z_k \in \mathbb{IR} \quad \forall k = 1, \ldots, L \tag{2.47}$$ {?}

The set of all N-dimensional interval boxes is denoted $\mathbb{IR}^N$.

Given a function $\phi : \mathbb{R}^L \to \mathbb{R}^L$ we define a new interval valued function Φ such that

$$\Phi : \mathbb{IR}^L \to \mathbb{IR}^L \qquad \text{and} \qquad \mathbf{z} \in \mathbf{Z} \quad \Rightarrow \quad \phi(\mathbf{z}) \in \Phi(\mathbf{Z}). \tag{2.48}$$ Definition interval

A function Φ with this property is called an *interval extension* of ϕ. In other words $\Phi(\mathbf{Z})$ is guaranteed to contains the image of $\mathbf{Z}$ under ϕ. Note that an interval extension is not unique, and we often would like to use the *tightest* possible extension, meaning that we want $\Phi(\mathbf{Z})$ to approximate $\phi(\mathbf{Z})$ (i.e. the image of the set $\mathbf{Z}$ by ϕ) as closely as possible. A perfect estimation is however not possible in general since the image of an interval box is not always itself an interval box.

Note that as oppose to ϕ, its interval extension Φ is exactly representable numerically. This can be done by requiring the that the implementation of Φ includes numerical inaccuracy in such a way that the eq. (2.48) holds for the finite precision numerical result $\Phi(\mathbf{Z})$. Hopefully good implementations, that respect this condition and are reasonably tight, can be found in existing libraries[2]. In this thesis we use the implementation provided by the Julia packages `IntervalArithmetic.jl` [50] and `IntervalRootFinding.jl` [51].

The properties of the interval extension allows to solve equations in a guaranteed way. To see that first consider an equation expressed as the fixpoint problem

$$\phi(\mathbf{z}) = \mathbf{z}, \tag{2.49}$$ Interval arithmetic

and Φ an interval extension of ϕ. There are many ways of defining ϕ to make an arbitrary equation equivalent to a fixpoint problem. For interval arithmetic the most common are Newton and Krawczyck operators [52, 53]. As we use it below in Section 2.4.5, we introduce the latter in Appendix C. However we will also use the more natural fixpoint displayed by eq. (2.17).

Now that we have expressed a problem in term of a fixpoint, let the interval box $\mathbf{Z}$ contain a solution $\mathbf{z}^*$ of eq. (2.49). By the definition 2.48 of an interval extension

[2] This implementation are however not absolute. See Appendix B for detials about the limitations.

and the fact that $\mathbf{z}^*$ is a fixpoint of ϕ, we have

$$\mathbf{z}^* = \phi(\mathbf{z}^*) \in \Phi(\mathbf{Z}) \tag{2.50}$$ {?}

Since $\mathbf{z}^* \in \mathbf{Z}$ we can even say

$$\mathbf{z}^* \in \mathbf{Z} \cap \Phi(\mathbf{Z}). \tag{2.51}$$ Generic solutio

Therefore in general the relation between $\mathbf{Z}$ and $\Phi(\mathbf{Z})$ can give information about the presence of solution in $\mathbf{Z}$. First if $\mathbf{Z} \cap \Phi(\mathbf{Z}) = \varnothing$, there can not be any solution of eq. (2.49) in $\mathbf{Z}$.

On the other hand, if $X \subseteq \Phi(\mathbf{Z})$, we can use the fact that both $\mathbf{Z}$ and $\Phi(\mathbf{Z})$ are interval boxes and thus are compact convex subspaces of $\mathcal{R}^N$. Brouwer fixpoint theorem can be used, which states that under such conditions, $\Phi(\mathbf{Z})$ must have a fixpoint. As a consequence, $\mathbf{Z}$ must contain at least one solution. It has also been proven that for Newton and Krawczyck operators if $\mathbf{Z}$ is a subset of the interior of $\Phi(\mathbf{Z})$, then the solution in $\mathbf{Z}$ is unique [52, 53]. We do not discuss the uniqueness further in this thesis though, as it is a technical point not necessary for our purpose.

Moreover, in the case where $\mathbf{Z}$ is neither outside of or included in $\Phi(\mathbf{Z})$, we can not conclude anything about the number of solutions of eq. (2.49) in $\mathbf{Z}$. However, in this case $\mathbf{Z}$ could be bisected in the hope to be able to get more conclusive result on finer interval box.

To conclude this section, consider an interval box $\mathbf{Z}_0$ and the sequence $\mathbf{Z}_k$ of interval boxes recursively defined as

$$\mathbf{Z}_{k+1} = \mathbf{Z}_k \cap \Phi(\mathbf{Z}_k), \quad \forall k \in \mathbb{N}. \tag{2.52}$$ Generic refinem

Equation (2.51) tells that all zeros present in the initial interval box $\mathbf{Z}_0$ are also present in $\mathbf{Z}_k$ for all k. By definition $\mathbf{Z}_{k+1} \subseteq \mathbf{Z}_k$, so iterating in the sequence for large k can only yield a closer enclosure of the solutions in $\mathbf{Z}_0$. In fact, if $\mathbf{Z}_0$ contains one isolated solution $\mathbf{z}^*$ in its interior and $\mathbf{z}^*$ is an attractive fixpoint of ϕ, then $\mathbf{Z}_k$ converges to $\{\mathbf{z}^*\}$ for large k. For Newton and Krawczyk operator it can actually be proven that it converges under weak assumptions [52, 53].

This allows us to present a generic branch and bound algorithm to determine in principal, all fixpoints of a function ϕ in a starting interval box $\mathbf{Z}_0$, given an interval extension Φ of ϕ:

1. Define the set $\mathcal{S}_{\text{working}}$ and add $\mathbf{Z}_0$ to it.
2. If $\mathcal{S}_{\text{working}}$ is empty go to 8, otherwise remove an interval box from $\mathcal{S}_{\text{working}}$ and name it $\mathbf{Z}$.
3. If the diameter of $\mathbf{Z}$ is smaller than some tolerance δ add $\mathbf{Z}$ to the set $\mathcal{S}_{\text{unkown}}$ of interval box for which the algorithm is inconclusive and go to 2.
4. Compute the contracted interval box $\mathbf{C} = \Phi(\mathbf{Z})$.
5. If $\mathbf{C} \subset \mathbf{Z}$, add $\mathbf{Z}$ to the set $\mathcal{S}_{\text{exist}}$ of region in which a solution exists[3] and go to 2.
6. If $\mathbf{C} \cap \mathbf{Z} = \varnothing$, discard $\mathbf{Z}$ and go to 2.

[3] For Newton and Krawczyk operator, checking if $\mathbf{C}$ is in the interior of $\mathbf{Z}$ proves that the solution is unique. When using Krawczyk operator latter, this step is modified in that sense.

7. Bisect the interval box $\mathbf{C} \cap \mathbf{Z}$ along its largest edge and add both halves to $\mathcal{S}_{\text{working}}$[4]. Go to 2.

8. Use the refinement iteration (2.52) until convergence is achieve on all elements of $\mathcal{S}_{\text{exist}}$.

Such algorithm is present in the package `IntervalRootFinding.jl` [51] which implementation we use latter with the Krawczyk operator[5].

conclusion

2.4.3 Parametrized interval extension

erval extension⟩

As mentioned in the motivation of this section, interval arithmetic does not only give guaranteed results, it also allows to work with regions of a parameter space. To see that, consider a function

$$\phi(\boldsymbol{\lambda}, \mathbf{z}) : \mathbb{R}^N \times \mathbb{R}^L \to \mathbb{R}^L, \tag{2.53}$$ {?}

for which we search a fixpoint

$$\phi(\boldsymbol{\lambda}, \mathbf{z}) = \mathbf{z}. \tag{2.54}$$ {?}

Such parametrized problem, parametrized in the sense $\boldsymbol{\lambda}$ can be seen as the parameters of the problem, have been previously discuss in the literature in the context of interval arithmetic, see for example [54].

To find fixpoint of ϕ, let first introduce $\Phi(\boldsymbol{\Lambda}, \mathbf{Z})$ an interval extension of $\phi(\boldsymbol{\lambda}, \mathbf{z})$. By definition of the interval extension (2.48), we have

$$\phi(\boldsymbol{\lambda}, \mathbf{z}) \in \Phi(\boldsymbol{\Lambda}, \mathbf{Z}), \qquad \forall \boldsymbol{\lambda} \in \boldsymbol{\Lambda}, \forall \mathbf{z} \in \mathbf{Z}. \tag{2.55}$$ {?}

The crucial observation here is that, as a consequence, $\Phi(\boldsymbol{\Lambda}, \mathbf{Z})$ is an interval extension of the function

$$\phi(\boldsymbol{\lambda}, \cdot) : \mathbb{R}^L \to \mathbb{R}^L \tag{2.56}$$ {?}

$$\mathbf{z} \mapsto \phi(\boldsymbol{\lambda}, \mathbf{z}) \tag{2.57}$$ {?}

for every $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$. This means that all reasoning using interval extension presented in the previous chapter can be applied at once for all value of $\boldsymbol{\lambda}$ in $\boldsymbol{\Lambda}$.

In particular this may allow to prove that a function has no solution for any $\boldsymbol{\lambda}$ in some region of the parameter space, or on the opposite that it has a unique solution for every $\boldsymbol{\lambda}$ of this region.

Too much should not be expected when working with large $\boldsymbol{\Lambda}$ region. Indeed, in this case the interval extension may be far too broad to be able to conclude anything, making it of little use. However, as already mentioned in the previous section, in such case it is possible to bisect $\boldsymbol{\Lambda}$ until it is small enough to provide conclusive results.

[4] Some care should be taken here in the case where $\mathbf{C}$ end up being undefined or unbounded (this may happen with Newton and Krawczyk operator if the Jacobian is zero in $\mathbf{Z}$). See the implementation in the source code of [51] for details.

[5] From a technical point of view, it should be noted that the version we use is the master branch in its state of the 31th of October 2018, with the inclusion of the pull request #97. Also, while reviewed by and based on the previous work of the community, both Krawczyk operator and the generic branch and bound algorithm were written by the author, and should thus meet the same scepticism as the rest of this thesis.

2.4.4 Algorithm to approximate the critical region

We would like to approximate the critical region $\mathcal{R}$ by an union of interval boxes, in the sense that we want a set of interval boxes $\mathbf{\Lambda}_k$ such that

$$\mathcal{R} \subseteq \bigcup_k \mathbf{\Lambda}_k. \tag{2.58}$$ {?}

Obviously, we would this union to be the tightest possible around $\mathcal{R}$. However, we can not directly find such an union, the best we can do is to exclude interval boxes where the critical region can not be, either because we disprove the existence of any non trivial solution to eq. (2.17) on the whole parameter interval, or because we prove the existence of a non trivial solution for any parameter in the parameter interval. Technically, there is a third possibility using the system (2.43), which tell us we can exclude regions where the determinant of the Jacobian $J_{\boldsymbol{\lambda}}$ can not be zero. However this last method would only give a numerical solution to our previous result, here we would like an independent approximation of the critical region to verify that our result is correct.

To verify these criteria, we apply a logic similar to what is presented in Sections 2.4.2 and 2.4.3 to the function ψ defined by eq. (2.13). Since we are interested in its fixpoints (as displayed by eq. (2.17)), this is fully compliant with the theory presented. Let $\Psi(\mathbf{\Lambda}, \mathbf{Z})$ be an interval extension of $\psi(\boldsymbol{\lambda}, \mathbf{z})$. Given a parameter interval box $\mathbf{\Lambda}$ we want to distinguish three different cases:

1. A non trivial solution exists for all λ in $\mathbf{\Lambda}$, and therefore $\mathbf{\Lambda}$ is fully enclosed in the high connectivity phase.

2. All interval boxes $\mathbf{Z}$ which are not guaranteed to be empty satisfy $\underline{Z_k} > 1 - \varepsilon$ for all $k = 1, \ldots, L$, where $\underline{Z_k}$ is the lower bound of the component k of $\mathbf{Z}$ and ε is a small number, chosen to be $\varepsilon = 10^{-2}$ in the examples presented below. This is needed to identify region where only the trivial solution $\mathbf{u}_T$ exists, as the algorithm is not able in general to conclude on the uniqueness of solution on the boundary of the search region. In this case we thus conclude that $\mathbf{\Lambda}$ is included in the trivial region.

3. If none of the above is satisfied, we can not conclude and we bisect Λ until we can either conclude or its diameter falls under a threshold δ. For the examples presented we used $\delta = 5 \times 10^{-}3$.

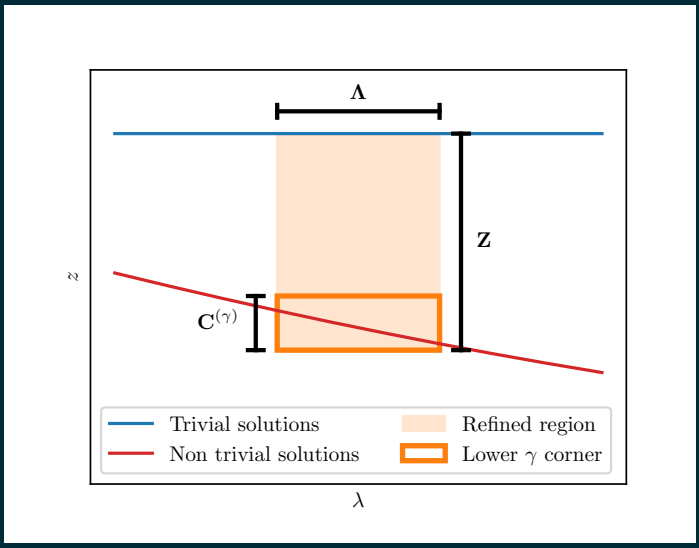
The root finding algorithm presented at the end of Section 2.4.2 would work for that purpose, but give us more informations that what we need, we therefore use an algorithm better fitting our needs. Before proceeding to the presentation of the algorithm, we make several observations. First, we adapt the refinement iteration (2.52) so it becomes

$$\mathbf{Z}_{k+1} = \Psi(\mathbf{\Lambda}, \mathbf{Z}_k). \tag{2.59}$$ Fixpoint refine

The intersection with $\mathbf{Z}_k$ is removed as it only gives minor improvement.

Then, to conclude on the non triviality of a region, we only need to find one non trivial solution. A good place to search for it is in the lower corner of a refinement of the unit interval box $\mathcal{I}^L$, as shown in fig. 2.5. Formally we define the lower γ corner $\mathbf{C}^{(\gamma)}$ of $\mathbf{Z}$ as

$$C_k^{(\gamma)} = \left[\underline{Z_k}, (1-\gamma) + \gamma \underline{Z_k}\right], \tag{2.60}$$ {?}



e: Lower corner⟩

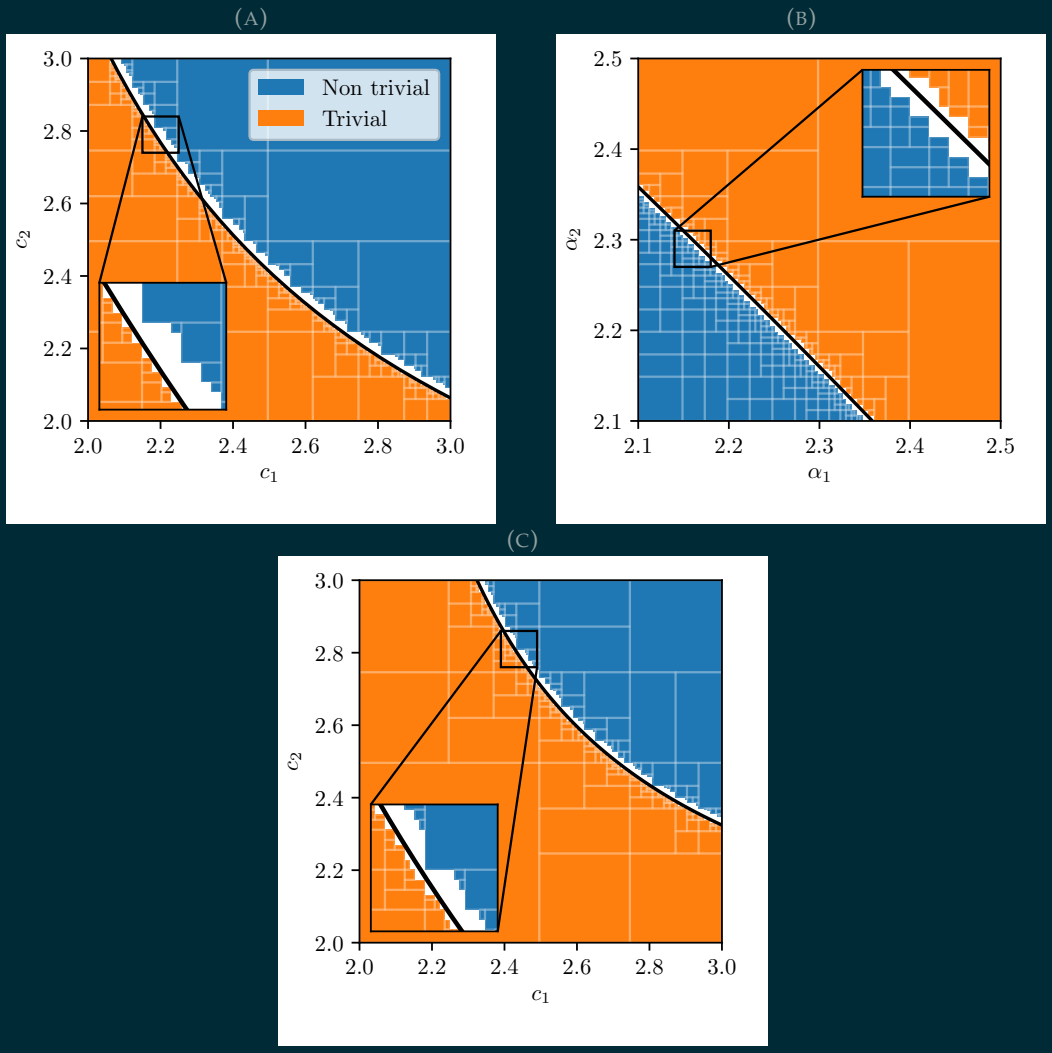
FIGURE 2.5: Example of a lower γ corner of the refinement $\mathbf{Z}$ of the the unit interval over a parameter box $\mathbf{\Lambda}$ in one dimension. It is expected that the lower bound of $\mathbf{Z}$ is close to the non trivial solutions, and hence that the lower γ corner often contains them.

for each component $C_k^{(\gamma)}$ of $\mathbf{C}^{(\gamma)}$. We empirically found that $\gamma = 0.75$ gives good results in our example, being wide enough to be able to map it into itself under $\Psi(\mathbf{\Lambda}, \cdot)$ and thus proving the existence of a solution.

We now present the algorithm we use, which goes as follow

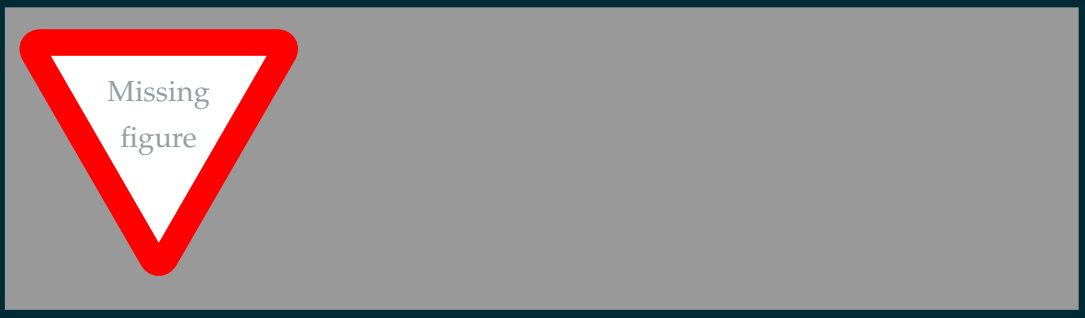
1. Add the global interval box $\mathbf{\Lambda}_0$ to the set $\mathcal{S}_{\text{working}}$ of regions to be processed.
2. If $\mathcal{S}_{\text{working}}$ is empty, terminate.
3. Remove an interval box $\mathcal{S}_{\text{working}}$ and name it $\mathbf{\Lambda}$.
4. If the diameter of $\mathbf{\Lambda}$ is smaller than the tolerance δ, add it to $\mathcal{S}_{\text{unkown}}$ and go to 2.
5. Apply the refinement iteration (2.59) until convergence to the interval box $\mathbf{Z}$, starting with $\mathbf{Z}_0 = \mathcal{I}^L$.
6. If $\mathbf{Z}$ is approximatively the trivial solution, in the sense presented above, add $\mathbf{\Lambda}$ to the set $\mathcal{S}_{\text{trivial}}$ of trivial regions and go to 2.
7. Compute the γ upper corner $\mathbf{Z}_c$ of the box $\mathbf{Z}$. Apply the refinement iteration (2.59) to $\mathbf{Z}_c$ until either convergence or until $\mathbf{Z}_{k+1} \subseteq \mathbf{Z}_k$. In the latter case if $\mathbf{Z}_{k+1}$ does not contain $\mathbf{u}_T$, add $\mathbf{\Lambda}$ to the set $\mathcal{S}_{\text{nontrivial}}$ of non trivial regions and go to 2.
8. Bisect $\mathbf{\Lambda}$ and add both halves to $\mathcal{S}_{\text{working}}$ Go to 2.

This algorithm outputs the three regions described above: the high and low connectivity regions, and the region for which the algorithm is inconclusive. We apply this algorithm to networks composed of two independent layers with the same degree distribution. Results for Erdos-Renyi, scale-free and exponential layers are shown in fig. 2.6. The regions are represented together with numerical solutions of the system (2.43), computed using the `NLsolve.jl` [55] package. As can be seen on the figure, both methods agree very well, confirming the validity of the boundary condition (2.34).



egions and boundary⟩

FIGURE 2.6: Phase diagram for a multiplex network composed of two layers with the same degree distribution. In the blue region a non trivial solution for $\mathbf{u}$ has been found, in the orange region only the trivial solution $\mathbf{u}_T$ exists and in the uncolored region the algorithm was unable to conclude in favor of either case. The solid black line is the numerical solution to eq. (2.11) and (2.34). (A) Erdos-Renyi layers with mean degrees c_1 and c_2. (B) Scale-free layers with power law exponents α_1 and α_2. (C) Layers with exponential distribution with mean degrees c_1 and c_2.



ntinuity check⟩?

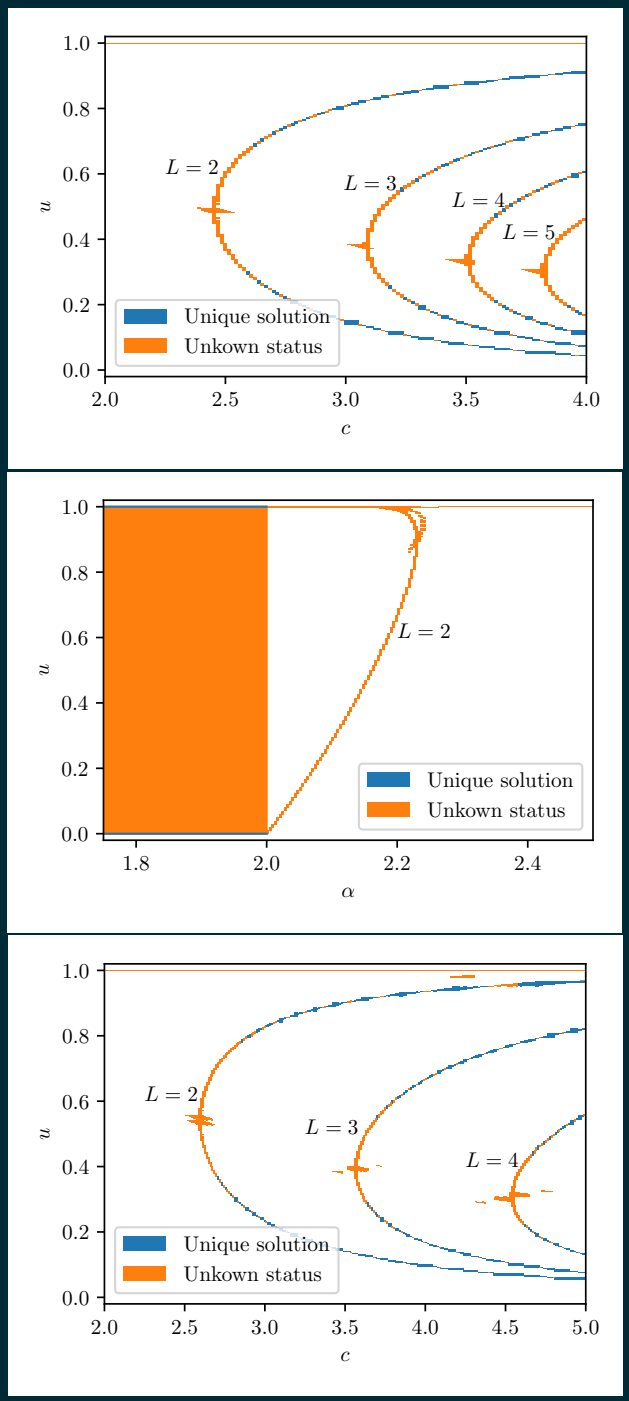
FIGURE 2.7: Discontinuity check

2.4.5 Discontinuity of the phase transition

hase transition⟩ A discontinuous phase transition implies a "gap" in between the trivial solution and the non trivial ones. Mathematically it is equivalent to the following

1. There is a region $\mathcal{L}$ of the parameter space and a vector $\hat{\mathbf{z}} \in \mathcal{I}^L$, $\hat{\mathbf{z}} \neq \mathbf{u}_T$, such that for every parameter $\boldsymbol{\lambda} \in \mathcal{L}$, all non trivial solutions $\mathbf{u}$ of eq. (2.23) have $\hat{z}_k > u_k$, for all $k = 1, \ldots, L$.

2. $\mathcal{L}$ covers the critical region $\mathcal{R}$, i.e. $\mathcal{R} \subseteq \mathcal{L}$.



er single parameter)?

FIGURE 2.8: Solution of eq. (2.23) for various number of layers L. Exactly one solution exists for each c in blue boxes, while uncolored part of the plot are guaranteed to contain no solutions. No conclusion can be made about existence of solution in the orange regions. These plots are sufficient to prove the discontinuity of the phase transition for Erdos-Renyi and exponential networks, but inconclusive for scale-free networks. (A) Erdos-Renyi networks with mean degree c. (B) Scale-free networks with power law exponent α. The region with $\alpha \leq 2$ has been added manually according to the explanation given in the text. No non empty and non trivial region have been found for $\alpha > 2$ and $L \geq 3$. (C) Exponential networks with mean degree c.

Appendix A

Fixpoint iteration for connected networks generation

A.1 Convergence of the fixpoint iteration

int convergence⟩ First notice that the case $r_1 = 0$ is trivial, as described in the main text. We will therefore assume in this Appendix that $r_1 > 0$, immediately giving $\mu(0) = r_1 > 0$. Second, not that (1.89) tells us that $z < 1$ implies $\mu(z) < 1$. From there we separate two cases:

If $u = 1$ is the unique solution of eq. (1.88) then $\mu(z)$ must be continuous for $z \in [0, 1]$ and $\mu'(1) < 1$, making $u = 1$ and attractive fixpoint. On the other hand if there is another solution u^* to eq. (1.88), it is the unique solution with $0 \leq u^* < 1$ since $\mu(z)$ is an increasing function of z, as it is demonstrated in Appendix A.2. Moreover, since $\mu(0) > 0$ we have $\mu'(u^*) < 1$, which makes it an attracting fixpoint and makes $u = 1$ a repulsive one.

We can thus conclude that the fixpoint iteration proposed always converges and converges to the degenerate case $u = 1$ only if it is the unique possibility.

A.2 Monotonicity of $\mu(z)$

x: Monotonicity⟩ To prove that $\mu(z)$ is an increasing function, we compute its derivative with respect to z, which yields

$$\mu'(z) = \left[\sum_{k=1}^{\infty} k\pi_k(z)\right]^{-2} (s_1(z) + s_2(z)) \tag{A.1}$$ {?}

$$s_1(z) = \sum_{j,k} kj\pi'_k(z)\pi_j(z)\left(z^{k-1} - z^{j-1}\right) \tag{A.2}$$ {?}

$$s_2(z) = \sum_{j,k} k(k-1)j\pi_k(z)\pi_j(z)z^{k-2}. \tag{A.3}$$ {?}

The sum $s_1(z)$ can be rewritten as

$$s_1(z) = \sum_{j>k} kj\left(\pi'_k(z)\pi_j(z) - \pi'_j(z)\pi_k(z)\right)\left(z^{k-1} - z^{j-1}\right) \tag{A.4}$$ {?}

$$= \sum_{j>k} \frac{kr_k}{1-z^k}\frac{jr_j}{1-z^j}\frac{z^k - z^j}{z^2}\left(\frac{k}{z^{-k}-1} - \frac{j}{z^{-j}-1}\right) \tag{A.5}$$ {?}

$$= \sum_{j>k} kj\pi_k(z)\pi_j(z)\frac{z^k - z^j}{z^2}\left(\frac{k}{z^{-k}-1} - \frac{j}{z^{-j}-1}\right). \tag{A.6}$$ {?}

Using the fact that the function

$$f_z(\lambda) = \frac{\lambda}{z^{-\lambda} - 1} \tag{A.7}$$ {?}

is a decreasing function of λ we can see that for $z \in [0, 1)$ and $j > k$ we have

$$z^k - z^j \geq 0 \tag{A.8}$$ {?}

$$\frac{k}{z^{-k} - 1} - \frac{j}{z^{-j} - 1} \geq 0, \tag{A.9}$$ {?}

and thus $s_1(z) \geq 0$. Moreover each terms in $s_2(z)$ is non-negative, so we have $s_2(z) \geq 0$. We can therefore conclude that $\mu'(z) \geq 0$ and thus that $\mu(z)$ is an increasing function of z.

Appendix B

Limitations of numerical interval arithmetic

rval arithmetic⟩

B.1 Dependency problem

Consider the interval $X = [-2, 3]$. The interval extension of the square root gives in that case $X^2 = [0, 9]$, as can easily be verified. However we have

$$X \cdot X = [-2, 3] \cdot [-2, 3] = [-6, 9] \neq X^2. \tag{B.1}$$ {?}

The reason for that is that replacing the (numerical) intervals does not contains any information about possible dependencies between intervals. As such, in the second equality, the interval extension of the multiplication has no way to know it should always take the same value in each intervals it is given. Therefore for example the value $-2 \cdot 3 = -6$ is a valid possible result, hence the discrepancy with the result of X^2. We present the problem with a peculiar example, but it is general for any function where a given interval appears multiple times.

Despite being very counter intuitive, this problem does not make any of further operations invalid in the sense of interval arithmetic, since the result is a correct interval extension. It is a very bad one though, that can make subsequent algorithm fail, for example if a positive result is expected when squaring a number.

In our case it has unwanted side effect that the interval extension of the generating functions may return intervals containing values outside the region $[0, 1]$, which is not possible. The Poisson degree and excess degree distributions (1.55) and (1.56) does not suffer from the problem as the parameter z and c only appear once in the formula, while power law generating functions (1.67) and (1.71) does not have interval extension in term of simple functions, a problem addressed in the next section of this Appendix. As such only the exponential degree distributions (1.80) and (1.83) cause a problem we need to address in the context presented in this Section.

For exponential networks, $g_0(z)$ can be rewritten as follow

$$g_0(z) = \frac{pz}{1-(1-p)z} \tag{B.2}$$ {?}

$$= \frac{pz}{1-z+pz} \tag{B.3}$$ {?}

$$= \left[\frac{1}{pz} - \frac{1}{p} + 1\right]^{-1} \tag{B.4}$$ {?}

$$= \left[\frac{1}{p}\left(\frac{1}{z} - 1\right) + 1\right]^{-1}. \tag{B.5}$$ {?}

In the last expression each parameter appears exactly once, and it is thus usable directly for interval arithmetic. Applying the same idea to $g_1(z)$, though, yields

$$g_1(z) = \left[\frac{1}{p}(1-z) + z\right]^{-2}, \tag{B.6}$$ {?}

which can not be rewritten, to our knowledge, in a way that avoid the repetition of both parameters. However, by defining

$$a(z) = \frac{1}{p}(1-z), \tag{B.7}$$ {?}

we have that $a(z)$ decreases with z. Using am interval extension $A(Z)$ of $a(z)$ we define

$$b_1 = \overline{A(Z)} + \underline{Z} \tag{B.8}$$ {?}

$$b_2 = \underline{A(Z)} + \overline{Z} \tag{B.9}$$ {?}

$$B(Z) = [\min\{b_1, b_2\}, \max\{b_1, b_2\}], \tag{B.10}$$ {?}

where $\underline{X}$ and $\overline{X}$ represent respectively the lower and higher bound of an interval X. Finally we have that

$$[B(Z)]^{-2} \tag{B.11}$$ {?}

is a tight interval extension of $g_1(z)$, which we use in our implementation.

As a concluding remark for this Section, we emphasise the fact this problem may disappear in the future by symbolically analysing the operations and building suitable interval extension using the gathered information. To our knowledge, this has however not been implemented yet in any interval analysis package available.

B.2 Special functions

Functions like the Riemann zeta $\zeta(\alpha)$ or the polylogarithm $\mathrm{Li}_\alpha(z)$ can not be computed numerically using only simple functions for which interval extensions are available. Even worse, standard implementations usually use various different approximation depending on the values of the argument, resulting in branching with which standard interval extensions can not always deal[1].

This general problem is however solvable in our case for the power law generating functions (1.67) and (1.71). Indeed, generating functions are always increasing with respect to z and power law degree distribution are also decreasing with respect to the exponent α. To see the latter, note that a smaller exponent α leads to higher degrees nodes, and thus smaller α implies bigger components, which means bigger S in eq. (1.39) and smaller u in eq. (1.40), which implies the fact that both $g_0(z)$ and $g_1(z)$ are decreasing with respect to α.

Monotonicity immediately gives an interval extension, for example for an increasing function $f(x)$, the interval valued function

$$F(X) = \left[\lfloor f(\underline{X})\rfloor, \lceil f(\overline{X})\rceil\right] \tag{B.12}$$ {?}

[1] For example the interval extension of the comparison operator $<$ should, formally, return [True, False] for the comparison $[0.5, 2] < 1$. Such case is not well handled by `if` statements, in any programming language.

is an interval extension of $f(x)$, $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ representing respectively rounding down and up to the nearest exactly representable floating point number.

We implemented this method for the generating function of a scale-free network, adding clamping results to the interval $[0, 1]$ to avoid nonsensical results, and thus we have a suitable interval extension of both generating functions.

Appendix C

Krawczyk operator

awczyk operator⟩

C.1 Mean value extension

Before we can introduce Krawczyk operator and prove it has the properties we wish, we need to present a peculiar form of interval extension, the mean value extension. For this consider a differentiable function $a : \mathbb{R}^M \to \mathbb{R}$. Then by the mean value theorem, there is a vector ξ lying on a straight path going from $\mathbf{x}$ to $\mathbf{y}$[1],

$$a(\mathbf{x}) - a(\mathbf{y}) = \nabla a(\boldsymbol{\xi}) \cdot (\mathbf{x} - \mathbf{y}). \tag{C.1}$$

Mean value theorem

We note $A'(\mathbf{X})$ and interval extension of $\nabla a(\mathbf{x})$. Interval extension of function including derivatives can be computed either by explicitely knowing the derivatives or by automatic differentiation [56]. The interval extension of the dot product $\cdot$ can be found by composing the interval extensions of the arithmetic operations, which allows to define the interval valued function

$$A_{mv}(\mathbf{X}) = a(m(\mathbf{X})) + A'(\mathbf{X}) \cdot (\mathbf{X} - m(\mathbf{X})), \tag{C.2}$$

Mean value extensio

with $m(\mathbf{X})$ being the center of interval box $\mathbf{X}$. Using the definition of interval extension (2.48) and the mean value theorem (C.1), we see that

$$a(\mathbf{x}) \in A_{mv}(\mathbf{X}), \qquad \forall \mathbf{x} \in \mathbf{X}, \tag{C.3}$$

{?}

and thus $A_{mv}(\mathbf{X})$ is itself an interval extension of $a(\mathbf{x})$, called the mean value extension of $a(\mathbf{x})$. This proves useful in the next section, but is not in general the preferred way of extending a function as it requires computing its derivative.

C.2 Fixpoint in Krawczyk operator

Let consider a differentiable function $f : \mathbb{R}^M \to \mathbb{R}^M$ define over an interval box $\mathbf{X}$ with interval extension F. We denote the Jacobian of f as f' and the interval extension of f' as F'. Assume that $f'(m(\mathbf{X})$ is not singular and hence invertible, which allows us to define Y as its inverse. All these elements are needed to be able to define the Krawczyk operator of f

$$K(\mathbf{X}) = m(\mathbf{X}) - Yf(m(\mathbf{X})) + \left[I - YF'(\mathbf{X})\right](X - m(X)), \tag{C.4}$$

Krawczyk operator

where I is the identity matrix. Vector of intervals (interval boxes) and matrices of intervals follows the standard multiplication rules, where each component is an interval and arithmetic operations are replaced by on of their interval extension.

[1] Formally we can say that there is a real $t \in [0, 1]$ such that $\boldsymbol{\xi} = (1 - t)\mathbf{x} + t\mathbf{y}$ fulfils eq. (C.1)

Krawczyk operator interest us because it fits in the framework presented in Section 2.4.2: it is the interval extension of a function $k(\mathbf{x})$ that has a fixpoint for solutions of the equation

$$f(\mathbf{x}) = 0. \tag{C.5}$$ Krawczyk genera

To prove this, we first write $k(x)$ explicitly as

$$k(\mathbf{x}) = \mathbf{x} - Y f(\mathbf{x}). \tag{C.6}$$ {?}

Since Y is not singular, $k(\mathbf{x}^*) = \mathbf{x}^*$ if and only if $f(\mathbf{x}^*) = 0$. Moreover, for $k_i(\mathbf{x})$ the i-th component $k(\mathbf{x})$, $[f'(\mathbf{x})]_i$ the i-th column of $f'(\mathbf{x})$ and $e_i = (0, \dots, 1, \dots, 0)$ where the one is the i-th component, we have

$$\nabla k_i(\mathbf{x}) = e_i - Y \left[f'(\mathbf{x})\right]_i. \tag{C.7}$$ {?}

Applying eq. (C.2) and using $F_i(\mathbf{X})$ and $\mathbf{X}_i$ to denote the i-th component of respectively $F(\mathbf{X})$ and X, we find the mean value extension $K_i(\mathbf{X})$ of $k_i(\mathbf{x})$,

$$K_i(\mathbf{X}) = m(\mathbf{X}_i) - Y f_i(m(\mathbf{X})) - \left[e_i - Y \left[F'(\mathbf{X})\right]_i\right] \cdot (\mathbf{X} - m(\mathbf{X})). \tag{C.8}$$ {?}

Putting every component together, we end up on precisely the Krawczyk operator as given in eq. (C.4), thus proving that it is an interval extension of $k(\mathbf{x})$. Therefore, following the reasoning presented in Section 2.4.2, we have that $K(\mathbf{X}) \subseteq \mathbf{X}$ implies the existence of a solution in $K(\mathbf{X})$ and that $K(\mathbf{X}) \cap \mathbf{X} = \varnothing$ implies the absence of solution in $\mathbf{X}$.

Interestingly, while the Krawczyk operator was first introduced by Krawczyk in 1969 in [57], he only proved in this paper that it refines an interval $\mathbf{X}$ around a solution x^* of eq. (C.5). This is only in 1977 in [58] that Moore made the observation and proved that Krawczyk operator can be used to determine the existence or absence of solution in a interval.

Note that in principle any matrix could be used in place of Y in the definition of the Krawczyk operator, as we do not use the fact it is the inverse Jacobian in our demonstration. Using an approximation of the inverse of $f'(\mathbf{x}^*)$ is however useful as it ensure that the iteration (2.52) converges towards the solution and that it does so faster than linearly [57].

Finally, note that this dependence on the inverse of $f'(\mathbf{x})$ is in general a liability. Indeed, if it is zero for a solution $\mathbf{x}^*$, the algorithm described in Section 2.4.2 is not able to prove its existence. Indeed, if X contains $\mathbf{x}^*$ such that $f'(\mathbf{x}^*) = 0$, then $Yx = \mathbb{IR}^M$ for any vector or interval box $\mathbf{x}$. As a consequence we have $K(\mathbf{X}) = \mathbb{IR}^M$, which is surely not included in $\mathbf{X}$.

In this regard, Krawczyk method is however more reliable. Its alternative, the Newton method, indeed requires that the Jacobian $f'(\mathbf{x})$ is regular everywhere in the interval box $\mathbf{X}$ considered [52, 53], while the Krawczyk only need in principle that it is the case for the solution.[2]

[2] In fact by using extended interval arithmetic, the problem can be avoided altogether with Newton method [59], but we decided not use this generalization here for simplicity.

Bibliography

[1] D. J. Watts and S. H. Strogatz. “Collective dynamics of ‘small-world’networks”. In: *nature* 393.6684 (1998), p. 440.

[2] J. W. Grossman and P. D. F. Ion. “On a portion of the well-known collaboration graph”. In: *Congressus Numerantium* (1995), pp. 129–132.

[3] G. F. Davis and H. R. Greve. “Corporate elite networks and governance changes in the 1980s”. In: *American journal of sociology* 103.1 (1997), pp. 1–37.

[4] P. M. Gleiser and L. Danon. “Community structure in jazz”. In: *Advances in complex systems* 6.04 (2003), pp. 565–573.

[5] J. Leskovec et al. “Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters”. In: *ArXiv e-prints* (Oct. 2008). arXiv: 0810.1355 [cs.DS].

[6] L. Šubelj and M. Bajec. “Robust network community detection using balanced propagation”. In: *The European Physical Journal B* 81.3 (2011), pp. 353–362.

[7] R. M. Ewing et al. “Large-scale mapping of human protein–protein interactions by mass spectrometry”. In: *Molecular systems biology* 3.1 (2007), p. 89.

[8] J.-F. Rual et al. “Towards a proteome-scale map of the human protein–protein interaction network”. In: *Nature* 437.7062 (2005), p. 1173.

[9] U. Stelzl et al. “A human protein-protein interaction network: a resource for annotating the proteome”. In: *Cell* 122.6 (2005), pp. 957–968.

[10] R. Guimera et al. “Self-similar community structure in a network of human interactions”. In: *Physical review E* 68.6 (2003), p. 065103.

[11] M. E. Newman, S. Forrest, and J. Balthrop. “Email networks and the spread of computer viruses”. In: *Physical Review E* 66.3 (2002), p. 035101.

[12] R. Albert, H. Jeong, and A.-L. Barabási. “Internet: Diameter of the world-wide web”. In: *nature* 401.6749 (1999), p. 130.

[13] C. C. Foster, A. Rapoport, and C. J. Orwant. “A study of a large sociogram II. Elimination of free parameters”. In: *Behavioral science* 8.1 (1963), pp. 56–65.

[14] J. Kunegis. “Konect: the Koblenz network collection”. In: *Proceedings of the 22nd International Conference on World Wide Web*. ACM. 2013, pp. 1343–1350.

[15] M. E. J. Newman. *Networks: an introduction*. Oxford University Press, Oxford, 2010.

[16] M. Bauer and D. Bernard. “Maximal entropy random networks with given degree distribution”. In: *arXiv preprint cond-mat/0206150* (2002).

[17] H. S. Wilf. *generatingfunctionology*. AK Peters/CRC Press, 2005.

[18] S. L. Feld. “Why your friends have more friends than you do”. In: *American Journal of Sociology* 96.6 (1991), pp. 1464–1477.

[19] M. Molloy and B. Reed. “A critical point for random graphs with a given degree sequence”. In: *Random structures & algorithms* 6.2-3 (1995), pp. 161–180.

[20] M. E. J. Newman, S. H. Strogatz, and D. J. Watts. "Random graphs with arbitrary degree distributions and their applications". In: *Physical review E* 64.2 (2001), p. 026118.

[21] I. Kryven. "General expression for the component size distribution in infinite configuration networks". In: *Physical Review E* 95.5 (2017), p. 052303.

[22] R. Solomonoff and A. Rapoport. "Connectivity of random nets". In: *The bulletin of mathematical biophysics* 13.2 (1951), pp. 107–117.

[23] E. N. Gilbert. "Random graphs". In: *The Annals of Mathematical Statistics* 30.4 (1959), pp. 1141–1144.

[24] P. Erdos and A. Rényi. "On random graphs I". In: *Publicationes Mathematicae (Debrecen)* 6 (1959), pp. 290–297.

[25] P. Erdos and A. Rényi. "On the evolution of random graphs". In: *Publ. Math. Inst. Hung. Acad. Sci* 5.1 (1960), pp. 17–60.

[26] P. Erdos and A. Rényi. "On the strength of connectedness of a random graph". In: *Acta Mathematica Hungarica* 12.1-2 (1961), pp. 261–267.

[27] A.-L. Barabási and R. Albert. "Emergence of scaling in random networks". In: *science* 286.5439 (1999), pp. 509–512.

[28] M. T. Agler et al. "Microbial hub taxa link host and abiotic factors to plant microbiome variation". In: *PLoS biology* 14.1 (2016), e1002352.

[29] H. Jeong et al. "The large-scale organization of metabolic networks". In: *Nature* 407.6804 (2000), p. 651.

[30] M. E. Newman. "The structure of scientific collaboration networks". In: *Proceedings of the national academy of sciences* 98.2 (2001), pp. 404–409.

[31] H. Jeong, Z. Néda, and A. L. Barabási. "Measuring preferential attachment in evolving networks". In: *EPL (Europhysics Letters)* 61.4 (2003), p. 567. URL: http://stacks.iop.org/0295-5075/61/i=4/a=567.

[32] A. D. Broido and A. Clauset. "Scale-free networks are rare". In: *arXiv preprint arXiv:1801.03400* (2018).

[33] W. Aiello, F. Chung, and L. Lu. "A random graph model for massive graphs". In: *Proceedings of the thirty-second annual ACM symposium on Theory of computing*. Acm. 2000, pp. 171–180.

[34] R. Albert, H. Jeong, and A.-L. Barabási. "Error and attack tolerance of complex networks". In: *nature* 406.6794 (2000), p. 378.

[35] K.-I. Goh, B. Kahng, and D. Kim. "Universal behavior of load distribution in scale-free networks". In: *Physical Review Letters* 87.27 (2001), p. 278701.

[36] G. Ichinose and H. Sayama. "Invasion of cooperation in scale-free networks: Accumulated versus average payoffs". In: *Artificial life* 23.1 (2017), pp. 25–33.

[37] R. Pastor-Satorras et al. "Epidemic processes in complex networks". In: *Rev. Mod. Phys.* 87 (3 Aug. 2015), pp. 925–979. DOI: 10.1103/RevModPhys.87.925. URL: https://link.aps.org/doi/10.1103/RevModPhys.87.925.

[38] M. P. Stumpf, C. Wiuf, and R. M. May. "Subnets of scale-free networks are not scale-free: sampling properties of networks". In: *Proceedings of the National Academy of Sciences* 102.12 (2005), pp. 4221–4224.

[39] F. Johansson et al. *mpmath: a Python library for arbitrary-precision floating-point arithmetic (version 1.0.0)*. Sept. 2017. URL: http://mpmath.org/.

[40] A. Clauset, C. R. Shalizi, and M. E. Newman. "Power-law distributions in empirical data". In: *SIAM review* 51.4 (2009), pp. 661–703.

[41] P. L. Krapivsky, S. Redner, and F. Leyvraz. "Connectivity of growing random networks". In: *Physical review letters* 85.21 (2000), p. 4629.

[42] D. S. Callaway et al. "Are randomly grown graphs really random?" In: *Physical Review E* 64.4 (2001), p. 041902.

[43] L. Jing-Zhou and T. Yi-Fa. "An exponential distribution network". In: *Chinese Physics* 14.4 (2005), p. 643.

[44] P. Bialas and A. K. Oleś. "Correlations in connected random graphs". In: *Physical Review E* 77.3 (2008), p. 036124.

[45] S. M. Rinaldi, J. P. Peerenboom, and T. K. Kelly. "Identifying, understanding, and analyzing critical infrastructure interdependencies". In: *IEEE Control Systems* 21.6 (2001), pp. 11–25.

[46] S. V. Buldyrev et al. "Catastrophic cascade of failures in interdependent networks". In: *Nature* 464.7291 (2010), p. 1025.

[47] M. J. Pocock, D. M. Evans, and J. Memmott. "The robustness and restoration of a network of ecological networks". In: *Science* 335.6071 (2012), pp. 973–977.

[48] S.-W. Son et al. "Percolation theory on interdependent networks based on epidemic spreading". In: *EPL (Europhysics Letters)* 97.1 (2012), p. 16006.

[49] G. J. Baxter et al. "Avalanche collapse of interdependent networks". In: *Physical review letters* 109.24 (2012), p. 248701.

[50] L. Benet, D. P. Sanders, et al. *IntervalArithmetic.jl (version 0.15.0)*. Sept. 2018. URL: https://github.com/JuliaIntervals/IntervalArithmetic.jl/.

[51] L. Benet, D. P. Sanders, et al. *IntervalRootFinding.jl (version 0.4.0)*. Sept. 2018. URL: https://github.com/JuliaIntervals/IntervalRootFinding.jl/.

[52] R. E. Moore, R. B. Kearfott, and M. J. Cloud. *Introduction to interval analysis*. Vol. 110. Siam, 2009.

[53] W. Tucker. *Validated numerics: a short introduction to rigorous computations*. Princeton University Press, 2011.

[54] A. Neumaier. "The enclosure of solutions of parameter-dependent systems of equations". In: *Reliability in Computing*. Elsevier, 1988, pp. 269–286.

[55] P. K. Mogensen et al. *NLsolve.jl (version 3.0.0)*. Nov. 2018. URL: https://github.com/JuliaNLSolvers/NLsolve.jl.

[56] J. A. Fike and J. J. Alonso. "Automatic differentiation through the use of hyper-dual numbers for second derivatives". In: *Recent Advances in Algorithmic Differentiation*. Springer, 2012, pp. 163–173.

[57] R. Krawczyk. "Newton-algorithmen zur bestimmung von nullstellen mit fehlerschranken". In: *Computing* 4.3 (1969), pp. 187–201.

[58] R. E. Moore. "A test for existence of solutions to nonlinear systems". In: *SIAM Journal on Numerical Analysis* 14.4 (1977), pp. 611–615.

[59] E. Hansen. "A globally convergent interval method for computing and bounding real roots". In: *BIT Numerical Mathematics* 18.4 (1978), pp. 415–424.