UNIVERSITY OF FRIBOURG

MASTER PROJECT

Giant connected component in networks

Author:
Benoît RICHARD

Supervisors:
Dr. Guiyuan SHI
Prof. Yi-Cheng ZHANG

Theoretical Interdisciplinary Physics Group
Department of Physics

September 18, 2018

University of Fribourg

Abstract

Faculty of Science
Department of Physics

Master Project

Giant connected component in networks

by Benoît RICHARD

Write the abstract

Contents

List of Symbols

Symbol	**Description**	**Math. definition**
c	Expectation value for the degree	$\mathbb{E}\left[\deg v\right]$
$\deg v$	Degree of vertex v	
$\mathbb{E}\left[\ldots\right]$	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	
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	$P_1(\deg v = k)$
S	Fraction of the network which is part of the GCC in the large n limit	$P_0(v \in GCC)$
u	Probability that a node reached by following an edge from is not part of the GCC	$P_1(v \notin GCC)$
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	
1		
$g_0^{(i)}(z)$	Generating function for the degree of uniformly chosen nodes in layer i	
$g_1^{(i)}(z)$	Generating function for the degree of nodes reached by following an edge in layer i	
$p_k^{(i)}$	Probability that a uniformly chosen node has degree k in layer i	

Todo list

Chapter 1

Introduction

1.1 Networks

Many systems in real world can be conceptually represented as objects being connected to others. Such representation is called a network. For example, a road network can be represented as a set of crossings that are connected by direct roads. However, the concept of network does not require the object or the links between them to be physical. We can represent friendship relations as a network: two people are connected if they consider being friends.

Mathematically, networks are represented as *graphs*. A graph is an object composed of a set V of nodes (also referred to as vertices) and a set of edges E. An edge is characterized by the fact that it connects two nodes together, which in mathematical terms translate to the fact that an edge can be written as a pair of nodes or equivalently $E \subset \{(v_1, v_2) | v_1, v_2 \in V\}$. To fit the numerous systems many extension of this simple model can be considered, for example edges may have a direction (*directed graph*), meaning that $(v_1, v_2) \neq (v_2, v_1)$, edges or vertices can also have carry a value (*weighted graph*) or multiple edges between two vertices may be allowed.

1.2 Attribution

Find a good place to put this. Acknowledgement section at the end ?

Make sure the right sections are referenced

Large parts of this thesis are not original, in particular sections 2.4 through 2.8 are directly inspired by the presentation done in [1].

Chapter 2

Single layer networks

layer networks⟩?

2.1 Configuration model

guration model⟩? Single layer networks correspond to the classic picture of a network, in opposition to multiplex networks (also called multi-layer networks) which are a generalization of the concept of network discussed in Chapter 3. Since we are interested in fundamental properties of networks, we need to abstract from the specificity of one network. 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. This point of view is called the *configuration model* [?]. 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 that sample uniformly the space of all network with a given degree distribution.

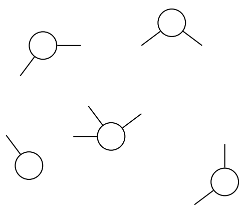
Missing ref

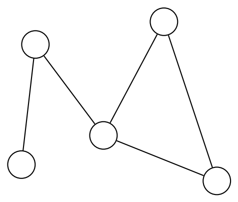
Consider a network with n vertices with a given degree distribution $\{p_k\}$. If we cut every edges in two, every vertex keep a number of *stubs* (half edges) equal to its degree. The resulting set of vertices and stubs is independent of the network structure, but common for all networks with the same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

Add stuff about the network space being uniformly sampled

2.2 Self-edges and multi-edges

Network distribution and multi- and self-edges distribution must be proof read and probably clarified


(A) Nodes and theirs stubs


(B) Stubs are randomly connected

guration model⟩?

FIGURE 2.1: Schematic representation of the configuration model.

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*. In a network with m edges, there are $2m$ stubs. The probability p_{ii} that a stub of vertex i connect to another stub of the same vertex is thus

$$p_{ii} = m\frac{\binom{k_i}{2}}{\binom{2m}{2}} = \frac{k_i(k_i-1)}{2(2m-1)}, \qquad (2.1)\ \{?\}$$

where k_i is the degree of vertex i. 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

$$m = \frac{1}{2}\sum_{k=0}^{\infty} nkp_k = \frac{n}{2}\mathbb{E}\left[\deg v\right], \qquad (2.2)\ \{?\}$$

with $\mathbb{E}\left[\ldots\right]$ denoting the expectation value. The average number of self-edges is therefore asymptotically constant as n becomes large, so the fraction of vertices having self edges goes to zeros as n grows and we can safely consider 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

$$p_{ij} = m\frac{k_i k_j}{\binom{2m-2}{2}} = \frac{k_i k_j}{2m-1}. \qquad (2.3)\ \{?\}$$

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

$$\frac{(k_i-1)(k_j-1)}{2m-3}. \qquad (2.4)\ \{?\}$$

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

$$p_{ij}\frac{(k_i-1)(k_j-1)}{2m-3}, \qquad (2.5)\ \{?\}$$

giving the average number of multi-edges

$$\frac{\sum_{ij} k_i k_j (k_j-1)(k_i-1)}{2(2m-1)(2m-3)} \approx \frac{1}{8m^2}\sum_i k_i(k_i-1)\sum_j k_j(k_j-1) \qquad (2.6)\ \{?\}$$

$$= \frac{1}{2}\left[\frac{\mathbb{E}\left[(\deg v)^2\right] - \mathbb{E}\left[\deg v\right]}{\mathbb{E}\left[\deg v\right]}\right]^2. \qquad (2.7)\ \{?\}$$

The approximation arise as in the limit of large n we have $2m-3 \approx 2m-3 \approx 2m$ as m scale proportionally to n. As in the case of self-edges, the number of multi-edges is asymptotically constant and we can therefore consider that the generated networks has no multi-edges.

Since $\{p_k\}$ is a probability distribution, it is independent of the number of nodes n of the network. Therefore for any degree distribution, we can consider the limit for large n, which we do as it allows several mathematical simplifications of the problem as outlined below. For sufficiently large networks, the difference between the large n limit and the actual network is small and can thus safely be neglected.

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.

2.3 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[1]. We can define the *excess degree distribution* $\{q_k\}$ as

$$q_k = P_1(\deg v = k+1), \qquad \forall k \in \mathbb{N}. \tag{2.8}$$ {?}

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

$$\frac{k}{2m-1} np_k = \frac{kp_k}{\mathbb{E}\left[\deg v\right]}. \tag{2.9}$$ {?}

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

$$q_k = \frac{(k+1)p_{k+1}}{\mathbb{E}\left[\deg v\right]}. \tag{2.10}$$ qk as function of p

2.4 Generating functions

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

$$g_0(z) = \sum_{k=0}^{\infty} p_k z^k. \tag{2.11}$$ Definition of g0

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

In a similar way we can define the generating function g_1 of the excess degree distribution $\{q_k\}$ as

$$g_1(z) = \sum_{k=0}^{\infty} q_k z^k. \tag{2.12}$$

Definition of g

troduce equation
r all moments

Noting that

$$\mathbb{E}[\deg v] = g_0'(1) \tag{2.13}$$

Expectation val

where the prime denotes derivation with respect to z and using eq. (2.10), we can rewrite $g_1(z)$ as

$$g_1(z) = \frac{1}{\mathbb{E}[\deg v]} \sum_{k=0}^{\infty} (k+1) p_{k+1} z^k = \frac{g_0'(z)}{g_0'(1)}. \tag{2.14}$$

g1 as a functio

2.5 Erdos-Renyi networks

real networks ?

In this thesis we will focus on three types of networks Erdos-Renyi networks, scale-free networks and geometric networks.

An Erdos-Renyi networks is characterized by the fact that it can be grown as follow: for each pair of nodes i and j, add an edge with 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.

$$c = \mathbb{E}[\deg v] = (n-1)p. \tag{2.15}$$

{?}

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

The probability for a node to have degree k is

$$p_k = \binom{n-1}{k} p^k (1-p)^{n-1-k}, \qquad \forall k \in \mathbb{N}. \tag{2.16}$$

?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 $c = (n-1)p$

$$p_k \approx \frac{c^k}{k!} e^{-c}. \tag{2.17}$$

{?}

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 (2.11), we recognize Taylor series representing the exponential function and thus we get

$$g_0(z) = e^{-c} \sum_{k=0}^{\infty} \frac{z^k c^k}{k!} = e^{-c} e^{cz} = e^{c(z-1)}. \tag{2.18}$$

?g0 for ER netwo

Taking the derivative and inserting in eq. (2.12) yields the generating function for the excess degree distribution

$$g_1(z) = e^{c(z-1)}, \tag{2.19}$$ {?}

which appears to be equal to $g_0(z)$.

2.6 Geometric networks

Geometric networks have a geometric degree distribution

Ask Guiyuan which version to use and if there is a some motivation in using this geom. dist.

2.7 Scale-free networks

The degree distribution of a so-called scale-free network follows a power law with exponent α

$$p_k = \frac{k^{-\alpha}}{\zeta(\alpha)}, \qquad \forall k \in \mathbb{N}^*, \tag{2.20}$$?Power law degree di

where $\zeta(\alpha)$ is the Riemann zeta function and $p_0 = 0$. This kind of networks is interesting as many real networks exhibits power law tail in their degree distribution . However, power law distribution are mathematically more complicated than the two previous examples as their generating function can not be represented in term of elementary function. The best we can do is introducing the *polylogarithm* $\mathrm{Li}_\alpha(z)$

Add examples

$$\mathrm{Li}_\alpha(z) = \sum_{k=1}^{\infty} k^{-\alpha} z^k. \tag{2.21}$$ Definition of polyl

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

$$\mathrm{Li}_\alpha(1) = \sum_{k=1}^{\infty} k^{-\alpha} = \zeta(\alpha). \tag{2.22}$$ Polylogarithm of 1

Credit the guy the polylog code comes from. Maybe say a bit more about polylog/reimplement polylog using intergation

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

$$g_0(z) = \sum_{k=1}^{\infty} \frac{k^{-\alpha}}{\zeta(\alpha)} z^k = \frac{\mathrm{Li}_\alpha(z)}{\zeta(\alpha)}. \tag{2.23}$$?Generating function

While no simple expression exist for the polylogarithm, its formal definition (2.21) is sufficient to compute its derivative

$$\frac{\partial}{\partial z}\mathrm{Li}_\alpha(z) = \sum_{k=1}^{\infty} k^{-\alpha+1} z^{k-1} = \frac{1}{z}\mathrm{Li}_{\alpha-1}(z)\,. \tag{2.24}$$?Derivative of the p

We therefore find

$$g_0'(z) = \frac{\mathrm{Li}_{\alpha-1}(z)}{z\zeta(\alpha)} \tag{2.25}$$?Derivative of g

that we can insert in eq. (2.14) to get

$$g_1(z) = \frac{\mathrm{Li}_{\alpha-1}(z)}{z\zeta(\alpha-1)}. \tag{2.26}$$ {?}

We used eq. (2.22) to perform the simplification $g_0'(1) = \zeta(\alpha-1)/\zeta(\alpha)$. Thanks to eq. (2.13), we know that $g_0'(1) = \mathbb{E}[\deg v]$. Thus we have

$$\mathbb{E}[\deg v] = \frac{\zeta(\alpha-1)}{\zeta(\alpha)}. \tag{2.27}$$ Mean degree in

Since we are not considering the analytic continuation of the zeta function, but only its real sum description given in eq. (2.22), $\zeta(\alpha)$ diverges for $\alpha < 1$. We can therefore distinguish three cases. First for $\alpha \leq 2$, eq. (2.27) 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 variance diverges. . Finally if $\alpha > 3$, both the mean degree and its variance are finite and not much happen .

y more about that
d do the calcula-
on

orse explanation
er, do better !

connected component⟩

2.8 Giant connected component

2.8.1 Size of the GCC

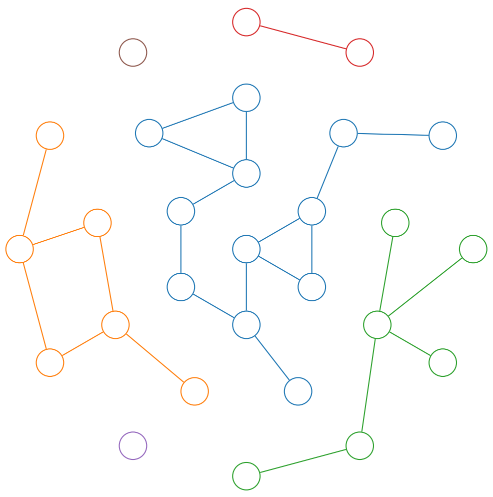
An interesting property of a network is the presence and size of *connected component*. A set of nodes is said to be connected if there is a path from any of its node to any other. All networks can be divided in connected components such that all nodes are element of exactly one component, as is exemplified in fig. 2.2. 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 forming its biggest component does not vanish in the large n limit. This component is called the *giant connected component* (GCC). The first question to answer about it is: what will be the size of the giant connected component ?

To determine this we first define u the probability that a node reached by following an edge is not part of the GCC. We can therefore write the probability S that a randomly chosen vertex is part of the GCC as

$$\begin{aligned} S &= 1 - P_0(w \notin GCC\ \forall w \in N(v)) && (2.28) \\ &= 1 - \sum_{k=0}^{\infty} P_0(w \notin GCC\ \forall w \in N(v) | \deg v = k) P_0(\deg v = k). && (2.29) \end{aligned}$$ {?} {?}

The probability $P_0(w \notin GCC\ \forall w \in N(v) | \deg v = k)$ is the probability that no neighbors of a node with degree k are part of the GCC. This in turn is the probability



cted components⟩

FIGURE 2.2: Scheme of a graph with six connected components, each drawn with a different color.

that by following k independent edges[2] we find each time a node which is not part of the GCC. This observation allow us to write

$$P_0(w \notin GCC\ \forall w \in N(v)|\deg v = k) = [P_1(w \notin GCC)]^k = u^k. \tag{2.30}$$ Probability that a

In the last equality we introduce the probability u for a first neighbor not to be part of the GCC,

$$u = P_1(w \notin GCC). \tag{2.31}$$?Definition of u?

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

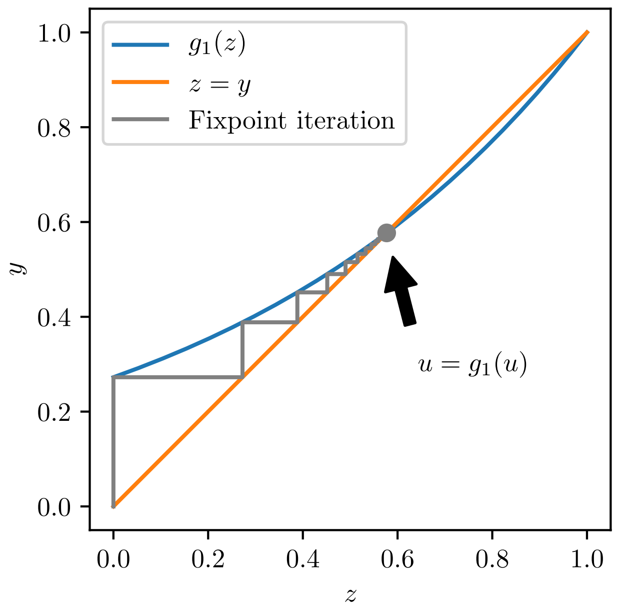
$$S = 1 - \sum_{k=0}^{\infty} u^k p_k \tag{2.32}$$ {?}
$$= 1 - g_0(u). \tag{2.33}$$ {?}

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 is not part of the GCC, none of its neighbors is either. Thus we can write

$$u = P_1(w \notin GCC\ \forall w \in N(v)) \tag{2.34}$$ {?}
$$= \sum_{k=0}^{\infty} P_1(w \notin GCC\ \forall w \in N(v)|\deg v = k) P_1(\deg v = k) \tag{2.35}$$ {?}
$$= \sum_{k=0}^{\infty} u^k q_k \tag{2.36}$$ {?}
$$= g_1(u), \tag{2.37}$$ {?}

where we use eq. (2.30) again together with the fact that by definition of the excess degree distribution $q_k = P_1(\deg v = k)$.

[2]Edges are independant by construction of the configuration model.



= g1(u) graphically⟩

FIGURE 2.3: Graphical representation of eq. (2.39) for an Erdos-Renyi network with $c = 1.3$. Gray solid line represent the successive steps of the fixpoint iteration.

We end up with two equations to describe the GCC size

$$S = 1 - g_0(u) \tag{2.38}$$

Single layer S

$$u = g_1(u). \tag{2.39}$$

Single layer u

If we can solve the second one we immediately get the GCC size. However eq. (2.38) only gives u implicitly and its form strongly depends on the degree distribution, therefore no general analytical solutions can be given. A trivial solution is however always present for $u = 1$ as by its definition (2.12), we have $g_1(1) = 1$. This implies $S = 0$ and thus the absence of a GCC. A necessary condition for the presence of a GCC is thus the existence of a non trivial solution to eq. (2.39)[3].

This equation can be solved numerically by noticing that the solution u is a fixpoint of the function $g_1(z)$, as can be seen in fig. 2.3. Since all coefficient in eq. (2.12) 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 $z_{k+1} = g_1(z_k)$ always converges toward the smallest solution of eq. (2.39).

2.8.2 Algorithm to find the connected components

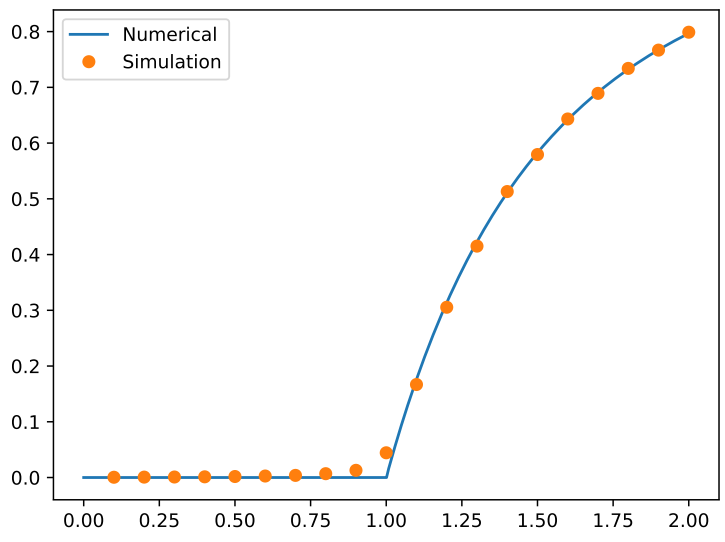
Since we are working in the limit of large n, the networks we are generating and analysing have large n as well. Therefore the algorithm we use must be designed with some care to avoid consuming to much computing time, which would make them impractical to use. This motivate us to present the algorithm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

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

1. Add all nodes to the list of `unprocessed` nodes.

[3]This is even a sufficient condition, see [1] for a proof of this.

(A)



(B)

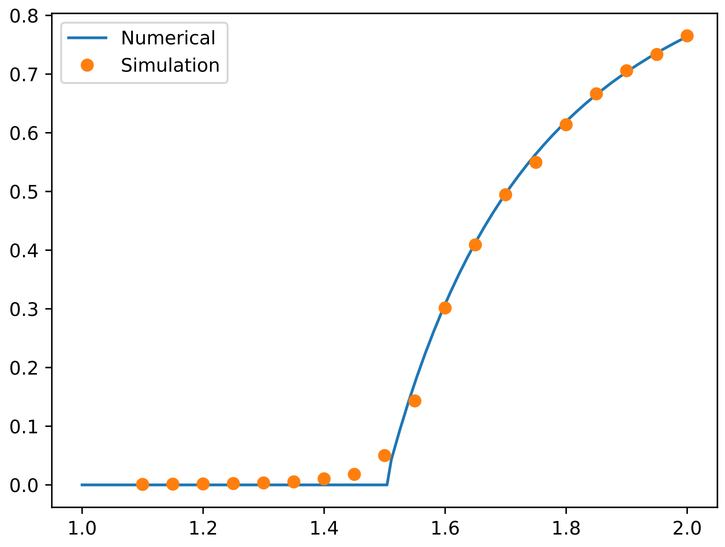


FIGURE 2.4: Numerical solution of eqs. (2.38) and (2.39), together with results on simulated networks. Results of simulations are average over 10 runs and the network size was set to 10^4 nodes. Simulations ran for several seconds in total, indicating that much larger network should be easily usable. (A) Poisson degree distribution. (B) Geometric degree distribution.

2. Remove one node from the list of `unprocessed` nodes and add it to the list of `queued` node.
3. Start a new component `C`.
4. Remove one node from the list of `queued` nodes and name it `v`.
5. Add `v` to the current component `C`.
6. Add all `unprocessed` neighbors of `v` to the list of `queued` nodes.
7. If there is any `queued` node go to 4, else store the component `C` and continue.
8. If there is any `unprocessed` node go to 2, else terminate.

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 $A = (A_1, \dots, A_n)$.[4]

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.

2.8.3 Simulations and stuff

Write subsection and find better title

2.8.4 Degree distribution in the GCC

ibution in the GCC⟩? Per Bayes theorem we have for two random events A and B

$$P(A|B) = P(B|A)\frac{P(A)}{P(B)}. \tag{2.40}$$

?Bayes theorem?

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

$$\begin{aligned} r_k &= P\left(deg(v) = k | v \in GCC\right) && (2.41)\ \{?\} \\ &= P(v \in GCC | deg(v) = k)\frac{P(deg(v) = k)}{P(v \in GCC)} && (2.42)\ \{?\} \\ &= \frac{p_k}{S}(1 - u^k). && (2.43) \end{aligned}$$

Degree distribu

Explain more the calculation

This result was previously presented more generally and following a very different path in [2]. In the context of the configuration model we choose the probabilities p_k which determine u and S through equations (2.38) and (2.39).

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

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

proportion of low degree vertices in the GCC and thus to increase the proportion of high degree vertices.

2.9 Generating connected networks

ected networks⟩?

2.9.1 Algorithm

Algorithm previously in [3]

The knowledge of the degree distribution in the GCC can be used generate a connected component of a given degree distribution r_k as follow: we first determine a degree distribution p_k fulfilling eq. (2.43) 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. (2.43) to get

$$p_k = S\pi_k(u), \quad \text{with} \quad \pi_k(z) = \frac{r_k}{1 - z^k} \tag{2.44}$$

{?}

Inserting this in the expression (2.39) for u, we get

$$u = \frac{\sum_{k=1}^{\infty} k\pi_k(u)u^{k-1}}{\sum_{k=1}^{\infty} k\pi_k(u)}. \tag{2.45}$$

Fixpoint equation fo

Therefore u is a fixpoint of the function

$$\mu(z) = \frac{\sum_{k=1}^{\infty} k\pi_k(z)z^{k-1}}{\sum_{k=1}^{\infty} k\pi_k(z)}, \tag{2.46}$$

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 [1]. 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.

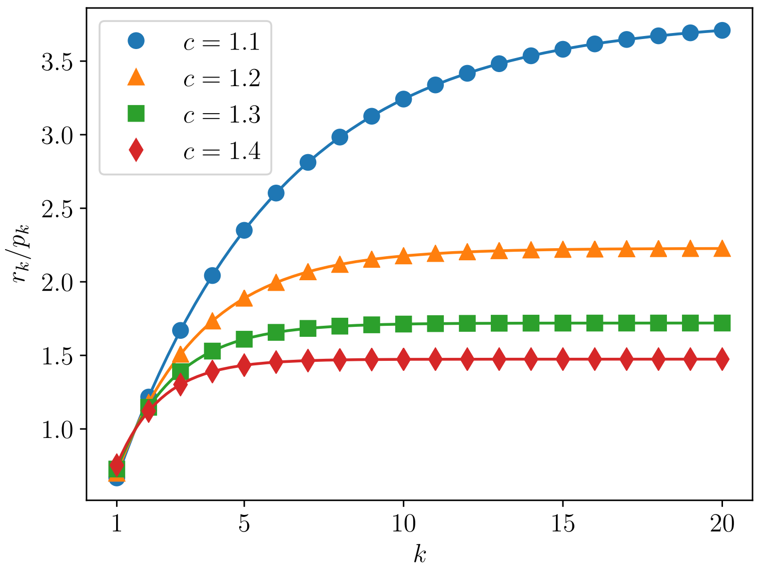
In practice we can not deal evaluate infinite sums numerically, thus we need to choose a cutoff index K for the sums such that

$$\sum_{k=K+1}^{\infty} k\pi_k(u) \ll 1. \tag{2.47}$$

{?}

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.



enyi reconstruction⟩

FIGURE 2.5: 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.

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

The probability to have degree k in an Erdos-Renyi network is

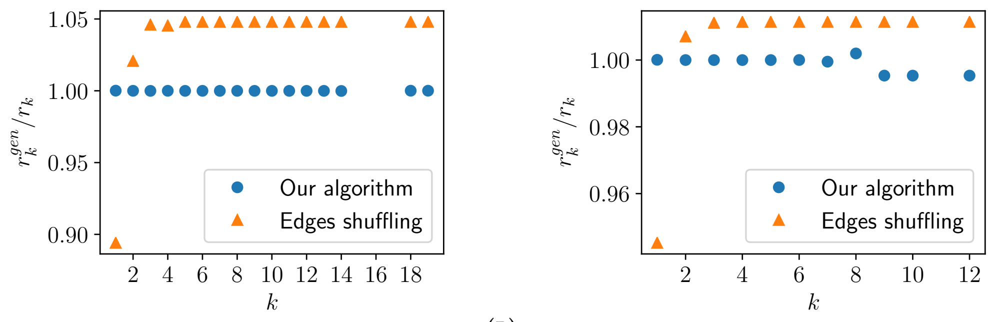
$$p_k = \frac{c^k}{k!} e^{-c}, \tag{2.48}$$ {?}

where c is the mean degree in the network. Using eq. (2.39) and (2.38) to find u and S yield everything we need to be able to determine the GCC degree distribution r_k from eq. (2.43). 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 result, correctly normalized, is therefore

$$p_k = \frac{c^k}{k!} \frac{1}{e^c - 1}. \tag{2.49}$$ {?}

The expected bias ratio r_k/p_k is shown for various mean degree c and a cutoff constant $K = 10000$ in fig. 2.5 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.



(A) (B)

: Real examples⟩

FIGURE 2.6: 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.

2.9.3 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 [4], the powergrid of the western states of the United States and the road network of the state of California[5].

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. 2.6, together with the results obtained by taking the GCC of the reshuffled network.

[5]The codes of the networks in the Konect database are respectively `UG` and `RO`.

Chapter 3

Multiplex network

tiplex networks⟩

3.1 Giant viable cluster

Consider a multiplex network with L layers. Let $g_0^{(i)}$ and $g_1^{(i)}$ be the generating functions of respectively the degree and the excess degree 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 of the giant viable cluster can be written as

$$S = P_0\left(\bigcap_{i=1}^{L} \exists w \in N_i(v)\ w \in GVC\right). \tag{3.1}$$ {?}

By requiring that the layers are independent from one others, we can rewrite S as a product

$$\begin{aligned} S &= \prod_{i=1}^{L} P_0\left(\exists w \in N_i(v)\ w \in GVC\right) && (3.2)\ \{?\} \\ &= \prod_{i=1}^{L} \left[1 - P\left(w \notin GVC\ \forall w \in N_i(v)\right)\right] && (3.3)\ \{?\} \\ &= \prod_{i=1}^{L} \left[1 - \sum_{k=0}^{\infty} P\left((w \notin GVC\ \forall w \in N_i(v) | deg(v) = k\right) p_k^{(i)}\right] && (3.4)\ \{?\} \\ &= \prod_{i=1}^{L} \left[1 - \sum_{k=0}^{\infty} u_i^k p_k^{(i)}\right] && (3.5)\ \{?\} \\ &= \prod_{i=1}^{L} \left[1 - g_0^{(i)}(u_i)\right]. && (3.6) \end{aligned}$$

Multiplex GCC size

We can 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 the giant viable cluster. Which as before can be written in the form

$$\begin{aligned} 1 - u_j &= P_1^{(j)}\left(\bigcap_{i=1}^{L} \exists w \in N_i(v)\ w \in GVC\right) && (3.7)\ \{?\} \\ &= \prod_{i=1}^{L} P_1^{(j)}\left(\exists w \in N_i(v)\ w \in GVC\right). && (3.8)\ \{?\} \end{aligned}$$

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

$$1 - u_j = \left[1 - \sum_{k=0}^{\infty} u_j^k q_k^{(j)}\right] \prod_{\substack{i=1\\i\neq j}}^{L} \left[1 - \sum_{k=0}^{\infty} u_i^k p_k^{(i)}\right] \tag{3.9}$$ {?}

$$= \left[1 - g_1^{(j)}(u_j)\right] \prod_{\substack{i=1\\i\neq j}}^{L} \left[1 - g_0^{(i)}(u_i)\right]. \tag{3.10}$$ Multiplex u fin

3.2 Algorithm to find the GVC

Write. Make sure to introduce small components problem

3.3 Boundary condition

This section is copy pasted from the corresponding draft paper

3.3.1 General case

Up to now, we have considered the multiplex generated to be determined via the degree distributions of each of its layer. However a degree distribution has an infinite number of degrees of freedom, therefore it is more practical to let the degree distributions depend on a finite set of parameters $\lambda_1, \lambda_2, \ldots, \lambda_N$ and express the behaviour of the network in term of them. Note that the number of parameters N does not need to match the number of layers L.

In order to make our main statement about the critical region for a multiplex network, we first need to introduce several quantities. First, let introduce

$$\mathbf{u} = (u_1, u_2, \ldots, u_L) \tag{3.11}$$ {?}

$$\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_N) \tag{3.12}$$ {?}

$$f_j(\boldsymbol{\lambda}, \mathbf{u}) = 1 - u_j - \left[1 - g_1^{(j)}(u_j)\right] \prod_{\substack{i=1\\i\neq j}}^{L} \left[1 - g_0^{(i)}(u_i)\right] \tag{3.13}$$?Definition fj?

and the function

$$F : \mathbb{R}^N \times \mathcal{I}^L \to \mathbb{R}^L \tag{3.14}$$ {?}

$$(\boldsymbol{\lambda}, \mathbf{u}) \mapsto F(\boldsymbol{\lambda}, \mathbf{u}) = (f_1(\boldsymbol{\lambda}, \mathbf{u}), f_2(\boldsymbol{\lambda}, \mathbf{u}), \ldots, f_L(\boldsymbol{\lambda}, \mathbf{u})), \tag{3.15}$$?Definition F?

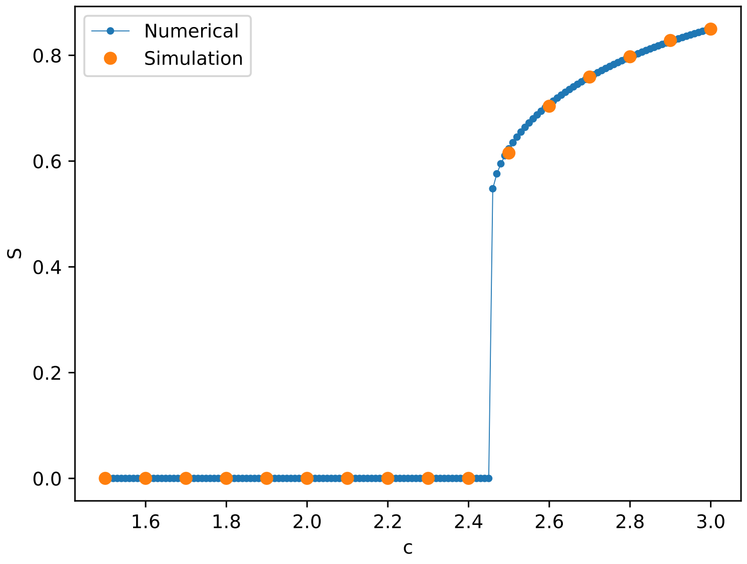
where $\mathcal{I} = [0, 1]$. The variables $\mathbf{u}$ in which we are interested are always in $\mathcal{I}^L$, since u_i represents a probability for all i.

Since the functions $g_0^{(i)}$ and $g_1^{(i)}$ are analytic with respect to the u_i, the function

$$F_{\boldsymbol{\lambda}} : \mathcal{I}^L \to \mathbb{R}^L \tag{3.16}$$ {?}

$$\mathbf{u} \mapsto F_{\boldsymbol{\lambda}}(\mathbf{u}) = F(\boldsymbol{\lambda}, \mathbf{u}), \tag{3.17}$$ {?}

(A)



(B)

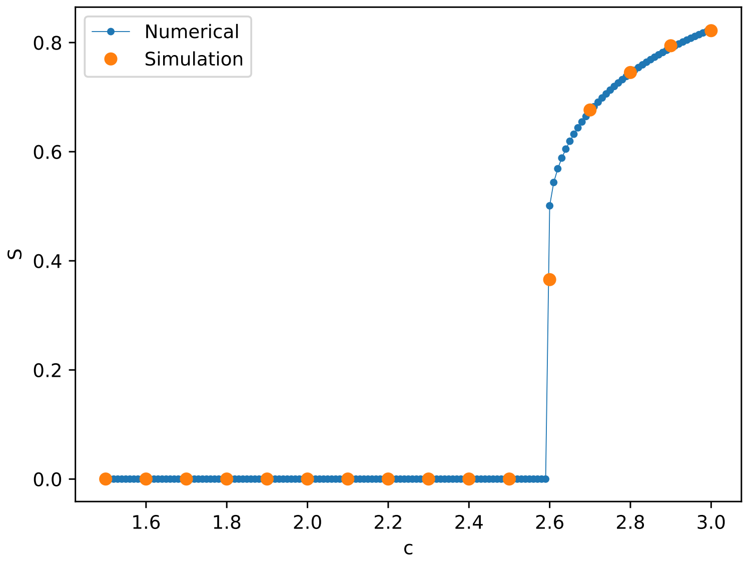


FIGURE 3.1: Numerical solution of eqs. (3.6) and (3.10), together with results on simulated networks for multiplex networks composed of two layers with the same distribution and mean number of edge c. Results of simulations are average over 10 runs and the network size was set to 10^4 nodes. Simulations ran for several seconds in total, indicating that much larger network should be easily usable. (A) Poisson degree distribution. (B) Geometric degree distribution.

is continuously differentiable for all parameters $\boldsymbol{\lambda}$. Therefore we can define Jacobi matrix $J(\boldsymbol{\lambda}, \mathbf{u})$ of $F_{\boldsymbol{\lambda}}$ as having coefficients

$$[J(\boldsymbol{\lambda}, \mathbf{u})]_{ij} = \frac{\partial f_i(\boldsymbol{\lambda}, \mathbf{u})}{\partial u_j}. \tag{3.18}$$

{?}

With the help of the notation introduced, we can now express solving eq. (3.10) as being equivalent to finding $\mathbf{u}^* \in \mathcal{I}^L$ such that

$$F(\boldsymbol{\lambda}, \mathbf{u}^*) = 0. \tag{3.19}$$

Implicit equati

If this equation only admits the trivial solution $\mathbf{u}^* = \mathbf{u}_T$, the parameter $\boldsymbol{\lambda}$ corresponds to a state without GVC. On the other if multiple solutions $\mathbf{u}^*$ exist, a GVC must exist as well. To determine the boundary between these two regions (i.e. the critical region), we use the implicit function theorem.

First, we assume that F (and not only $F_{\boldsymbol{\lambda}}$) is continuously differentiable and that we know a solution $\mathbf{u}^*$ of (3.19) for some parameter vector $\boldsymbol{\lambda}^*$. With that assumption the implicit function theorem can be state as follow:

If $\det[J(\boldsymbol{\lambda}^*, \mathbf{u}^*)] \neq 0$ then there is an open neighbourhood $U \subset \mathbb{R}^L$ of $\boldsymbol{\lambda}^*$ such that there is an unique continuously differentiable function $h : U \to \mathcal{I}^L$ with

$$h(\boldsymbol{\lambda}^*) = \mathbf{u}^* \tag{3.20}$$

{?}

$$F(\boldsymbol{\lambda}, h(\boldsymbol{\lambda})) = 0, \quad \forall \boldsymbol{\lambda} \in U. \tag{3.21}$$

Implicit soluti

The result in which we are interested here comes from the contrapositive of this statement, namely that if for all neighbourhoods U 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,

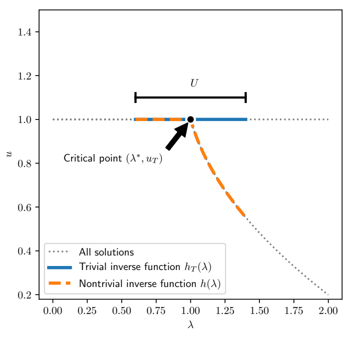
$$\det[J(\boldsymbol{\lambda}^*, \mathbf{u}^*)] = 0. \tag{3.22}$$

Boundary condit

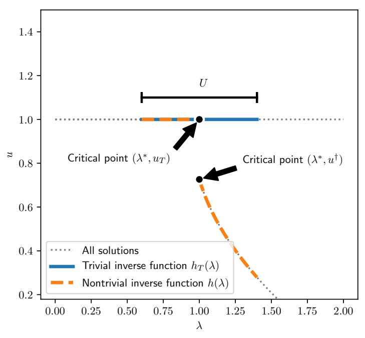
This condition has previously been outlined, without proof in [5]. We now prove that such situations arise if $\boldsymbol{\lambda}^*$ is a critical point of the phase transition between the absence and existence of a GVC, and therefore that eq. (3.22) is a sufficient condition to find the critical region of such phase transition.

First notice that in the context of multiplex network a phase transition appears between the trivial solution $\mathbf{u}_T = (1, 1, \ldots, 1)$ (where $S = 0$) and non trivial solutions ($S > 0$). However, the trivial solution $\mathbf{u}_T$ solves eq. (3.10) for any generating function and thus for any parameter vector $\boldsymbol{\lambda}$. For a continuous phase transition this immediately gives us $\mathbf{u}^* = \mathbf{u}_T$. Moreover, on one side of the phase transition occurring at $\boldsymbol{\lambda}^*$ one solution exists, while on the other at least two do. Therefore for any U open containing $\boldsymbol{\lambda}^*$ we can define two distinct functions on U that fulfil eq. (3.21), the trivial $h_T(\boldsymbol{\lambda}) = \mathbf{u}_T$ and another function h corresponding to the non trivial solutions, with $h(\boldsymbol{\lambda}^*) = h_T(\boldsymbol{\lambda}^*) = \mathbf{u}_T$. So the function h of the implicit function theorem is not uniquely defined and thus $\det[J(\boldsymbol{\lambda}^*, \mathbf{u}_T)] = 0$.

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



(A)



(B)

se transitions)?

FIGURE 3.2: (a) Scheme of a continuous phase transition. (b) Scheme of a discontinuous phase transition.

U such that

$$\lim_{n\to\infty} (\boldsymbol{\lambda}_n, \mathbf{u}_n) = (\boldsymbol{\lambda}^*, \mathbf{u}^\dagger) \quad \text{avec } \mathbf{u}^\dagger \neq \mathbf{u}_T \tag{3.23}$$ {?}

$$\lim_{m\to\infty} (\boldsymbol{\eta}_m, \mathbf{v}_m) = (\boldsymbol{\lambda}^*, \mathbf{u}_T) \tag{3.24}$$ {?}

$$F(\boldsymbol{\lambda}_n, \mathbf{u}_n) = 0 \quad \forall n \tag{3.25}$$ {?}

$$F(\boldsymbol{\eta}_m, \mathbf{v}_m) = 0 \quad \forall m. \tag{3.26}$$ {?}

If we assume that an unique continuous function h solving eq. (3.21) exists, we would have

$$h(\boldsymbol{\lambda}_n) = \mathbf{u}_n \quad \forall n \tag{3.27}$$ {?}

$$h(\boldsymbol{\eta}_m) = \mathbf{v}_m \quad \forall m. \tag{3.28}$$ {?}

The continuity of h would furthermore imply

$$h(\boldsymbol{\lambda}^*) = \lim_{n\to\infty} h(\boldsymbol{\lambda}_n) = \lim_{n\to\infty} \mathbf{u}_n = \mathbf{u}^\dagger, \tag{3.29}$$ {?}

but also

$$h(\boldsymbol{\lambda}^*) = \lim_{m\to\infty} h(\boldsymbol{\eta}_m) = \lim_{m\to\infty} \mathbf{v}_m = \mathbf{u}_T. \tag{3.30}$$ {?}

Since $\mathbf{u}_T \neq \mathbf{u}^\dagger$, this gives raise to the contradiction $h(\boldsymbol{\lambda}^*) \neq h(\boldsymbol{\lambda}^*)$. Therefore our assumption must be false and no continuous function h can be defined to solve eq. (3.21). So finally, we have $\det[J(\boldsymbol{\lambda}^*, \mathbf{u}^*)] = 0$, $\mathbf{u}^*$ being either $\mathbf{u}_T$ or $\mathbf{u}^\dagger$.

3.3.2 One dimensional case

Introduce 1D variables more clearly

If $L = N = 1$, the problem is the classical problem of a one layer network which degree distribution is determined by a single parameter λ. In that case the Jacobi matrix J reduces to the scalar quantity

$$J(\lambda, u) = \frac{\partial}{\partial u}(g_1(u) - u) = \frac{\partial g_1(u)}{\partial u} - 1. \tag{3.31}$$ {?}

Therefore the condition for the boundary $\det J(\lambda, u) = 0$ becomes

$$\frac{\partial g_1(u)}{\partial u} = 1. \tag{3.32}$$ {?}

sing ref

This condition was already introduced and verified previously by[?].

3.4 Interval estimation of the critical region

3.4.1 Theoretical foundation

Let $C \subset \mathcal{R}^L$ be the set of all parameters $\boldsymbol{\lambda}$ corresponding to a critical point. This set correspond to the parameters that solve simultaneously eq. (3.10) and eq. (3.22) for some $\mathbf{u} \in \mathcal{I}^L$. In this section we present an alternative and independent method to estimate C in order to verify that eq. (3.10) and (3.22) yield the expected result.

sing ref

To do so we introduce concepts from *interval arithmetic*[?]. First of all an interval I is defined a set of the form

$$I = [a, b] = \{x \in \mathbb{R} | a \leq x \leq b\}. \tag{3.33}$$ {?}

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

$$B = I_1 \times I_2 \times \cdots \times I_N, \quad I_k \in \mathbb{IR} \quad \forall k = 1, \ldots, N \tag{3.34}$$ {?}

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

sing ref

Given a function $\phi : \mathbb{R}^M \to \mathbb{R}^N$ it is possible[?] to determine a new interval valued function $\Phi : \mathbb{IR}^M \to \mathbb{IR}^N$ such that

$$x \in B \quad \Rightarrow \quad \phi(x) \in \Phi(B). \tag{3.35}$$ Definition inte

A function Φ with this property is called an interval extension of ϕ. To determine were the critical region is, we would like, in a sense, to solve eq. (3.10) and (3.22). Several general schemes exist to solve equations in a guaranteed way using intervals[?], but here we use a simpler algorithm inspired by them and more suited to our present needs.

sing ref

We define

$$\psi(\boldsymbol{\lambda}, \mathbf{u}) = F(\boldsymbol{\lambda}, \mathbf{u}) + \mathbf{u}, \tag{3.36}$$ {?}

with components

$$\psi_j(\boldsymbol{\lambda}, \mathbf{u}) = 1 - \left[1 - g_1^{(j)}(u_j)\right] \prod_{\substack{i=1 \\ i \neq j}}^{L} \left[1 - g_0^{(i)}(u_i)\right]. \tag{3.37}$$ {?}

Also we define Ψ as an interval extension of ψ.

Now, let $\mathbf{u}^* \in U_0$ be a solution of (3.19) for some $\boldsymbol{\lambda} \in \Lambda$. By the definition of ψ and eq. (3.35),

$$F(\boldsymbol{\lambda}^*, \mathbf{u}^*) = 0 \quad \Rightarrow \quad \mathbf{u}^* = \psi(\boldsymbol{\lambda}^*, \mathbf{u}^*) \in \Psi(\Lambda, U_0). \tag{3.38}$$ {?}

Therefore if we apply $\Psi(\Lambda, \cdot)$ to an interval box U_0 containing a solution, the resulting interval box contains the solution as well. We know that all solutions $\mathbf{u}^*$ are contained in $\mathcal{I}$ by definition of $\mathbf{u}$ and thus by iterating the previous argument from $U_0 = \mathcal{I}$, all solutions are elements of the interval boxes $U_k(\Lambda)$ recursively defined by

$$U_{k+1}(\Lambda) = \Psi(\Lambda, U_k(\Lambda)). \tag{3.39}$$

Recursion relation

Therefore if we find k such that $U_k = \{\mathbf{u}_T\}$, we know that the system for any $\boldsymbol{\lambda} \in \Lambda$ only admits the trivial solution.

In praxis however, the sequence U_k never converges to exactly the set $\{\mathbf{u}_T\}$, we therefore consider the criterion to be met if

$$U_k \subset [1 - \varepsilon, 1]^L, \tag{3.40}$$

Criterion for trivi

for some small tolerance ε.

Furthermore it is possible in some cases to guarantee the presence of non trivial solutions, allowing to conclude that a GVC emerges. Indeed, if we can find some interval boxes Λ and U such that

$$\Psi(\Lambda, U) \subset U \quad \text{and} \quad \mathbf{u}_T \notin U, \tag{3.41}$$

Criterion for non t

then by definition of the interval extension (eq. (3.35)), we have

$$\psi(\boldsymbol{\lambda}, U) \subset U, \quad \forall \boldsymbol{\lambda} \in \Lambda. \tag{3.42}$$

{?}

Since U is closed and simply connected, the fixpoint theorem [?] applies, implying that for each $\boldsymbol{\lambda} \in \Lambda$ there must be at least one $\mathbf{u}^*$ in U such that $\mathbf{u}^*$ is a fixpoint, or in other words such that $\mathbf{u}^* = \psi(\boldsymbol{\lambda}, \mathbf{u}^*)$. Therefore eq. (3.41) is a sufficient condition to prove the existence of at least one solution. Moreover since we imposed $\mathbf{u}_T \notin U$, the solution present can not be the trivial one.

Find back which on exactly

Missing ref

3.4.2 Algorithm

Equations (3.40) and (3.41) give guaranteed criteria for respectively the absence and presence of a non trivial solution in the region Λ considered. This is sufficient to propose an algorithm to estimate the critical region C.

1. Choose a starting parameter region Λ_0 and store it in the set S_{work} of regions yet to be processed.
2. If S_{work} is empty terminate, otherwise retrieve the next parameter region from S_{work}, and call it Λ.
3. If the radius of Λ is smaller than some tolerance δ, store it in the set S_{unkown} of regions for which the algorithm is unable to conclude using the tolerance δ.
4. Compute $U_k(\Lambda)$ for k big, using eq. (3.39).
5. If $U_k(\Lambda)$ fulfil eq. (3.40), store Λ in the set $S_{trivial}$ of trivial regions and go to 2.
6. Take a subset V of $U_k(\Lambda)$ such that $\mathbf{u}_T \notin V$.
7. If V fulfil eq. (3.41), store Λ in the set S_{GVC} of non trivial regions and go to 2.
8. Bisect Λ in two sub regions and add both to S_{work}. Go to 2.

By construction, when the algorithm terminates, we have for the critical region C

$$C \subset \bigcup_{U \in S_{unkown}} U, \tag{3.43}$$ {?}

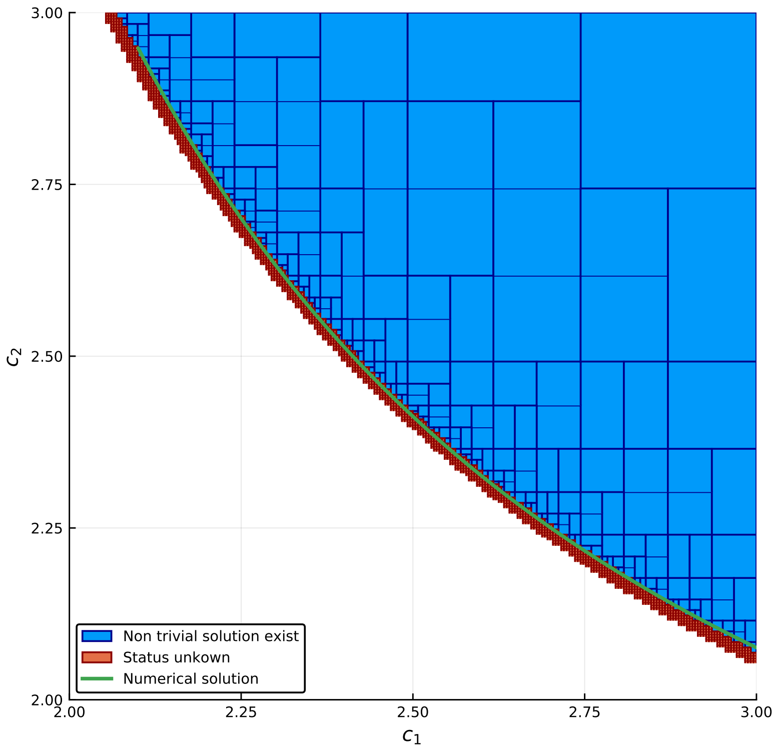
thus providing an estimation of C. For the estimation of C to be faitfull, it is crucial for the interval extension Ψ to be the tightest possible. To get good result in that regard, we used the interval arithmetic implementation of the `IntervalArithmetic.jl` Julia package[?].

sing ref

3.5 Results

We apply the two methods presented to different cases of multiplex network in order to test it. The algorithm using interval arithmetic is compared to the numerical solution of the system composed by eq. (3.10) and (3.22), computed using the `NLsolve.jl` Julia package.

Choose what multiplex networks should be used and with what parameters and actually produce the results.



s and boundary⟩?

Make the plot more readable and less ugly

Find why the two methods seems drift away one from the other far from the center.

FIGURE 3.3: Phase diagram for a multiplex network composed of two Erdos-Renyi layer with mean degree c_1 and c_2. In the blue region a non trivial solution for $\mathbf{u}$ has been found, in the uncolored region only the trivial solution $\mathbf{u}_T$ exists and in the red region the algorithm was unable to conclude in favor of either case. The solid green line is the numerical solution to eq. (3.10) and (3.22).

Appendix A

Fixpoint iteration to generate connected networks

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 (2.46) tells us that $z < 1$ implies $\mu(z) < 1$. From there we separate two cases:

If $u = 1$ is the unique solution of eq. (2.45) 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. (2.45), 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.

Bibliography

[1] M. Newman. *Networks: an introduction*. 2010.

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

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

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

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