Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecture with Computer Exercises:
Modelling and Simulating Social Systems with MATLAB

Project Report

Interdependent Evolution of Non-Spectral Opinions and Social Networks

Fabian Russmann & Stefan Rustler
Group Name: Social State Physicists

Zurich, Switzerland
December 2012

Agreement for free-download

We hereby agree to make our source code for this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Fabian Russmann

Stefan Rustler

Contents

1 Abstract

The question whether our social network shapes the opinion we hold or whether our opinions determine who is part of our network, was central to this work. A model that simulates the interdependent evolution of non-spectral opinions and social networks was implemented by including both mechanisms of one's social network shaping one's opinion and vice-versa. We were able to show that by merely altering the reconnection probability Φ that determines which of the two mechanisms is favored, a phase transition of the opinion distribution can be observed at the critical point of $\Phi_c = 0.32 \pm 0.05$. In one phase a giant cluster of equal opinion forms, whereas in the other there are only several smaller clusters. Attempts were made to describe this critical behavior in terms of solid state physics. We could show that different network sizes have the same critical point. Further a response function, the average time to reach consensus in the network, that diverges at the critical point could be identified. Finally, results were compared to hand-picked empirical data to give an idea of the applicability of our results.

2 Introduction

In this work, we investigated the dynamics of opinion and network formation. Both of these topics have received significant attention in the social sciences. Recently, however, other disciplines have also dealt with such problems, often applying models and methods of analysis from the physical sciences to social problems. Such studies have made quite some advances in the study of complex networks, (a good summary being given by [1]), but have also dealt with the problem of network dynamics and opinion formation, for example in voter models [3; 8]

The process of opinion formation is arguably one of the most complex but also one of the most fundamental in sociology, with wide-ranging implications in politics, economics and social psychology. For example, questions such as "How do the landscape of political parties and election results evolve?", "Why and how did different religions emerge in the course of history?", "How do traders decide which stocks to buy or sell?" or "Why do people prefer to buy certain brands over others?" could all be described as problems of understanding opinion formation. Though these are admittedly very difficult challenges, an understanding of these questions could potentially allow us to influence (for the better or worse) the processes of opinion and belief formation.

In this work, we want to study an opinion and network formation process using a quantitative model. Of course, as any model, it will only provide a stark simplification and idealization of the problem. However, in contrast to a lot of existing studies on opinion formation and the dynamics of social networks, this work is unique in the sense that it proposes a model combining these two approaches: We want to study the mechanisms of people forming opinions while connected with other people. Hence we not only study how opinions evolve in a social setting, but we also allow the network itself to be adaptable to the opinions existing on it, making two interdependent forces of network evolution and opinion formation measurable. In a sense, we want to address the question:

Does our social network shape the opinion we hold or does our opinion determine who is part of our network?

The motivation for this project involves two different angles on very fundamental dynamics of our society. First of all, the social sciences try to understand the ways in which humans become who they are under the influence of their environment. How do people form their opinions, values, and beliefs and how do their friends and acquaintances play a role in this? Secondly, we are interested in the way our networks of friends and social ties form in the first place. How and why do we choose to be friends with certain people and not with others? How do networks of people in a society form? From an intuitive point of view (considering that we are all social beings) most people would argue that the two aspects are interdependent or even that they are two extremes of the same process: Our social environment certainly shapes what we believe and which opinions we hold, while in turn our own values and opinions influence whom we choose to connect with and make part of our social network. On the basis of this rather vague but plausible assumption, our project is an attempt to disentangle and study the effects of these two mechanisms by the means simulations on networks with opinion attributes.

An example of a system that one would expect to be subject to such behavior is religious affiliation within a society. It could be argued that the social surroundings have an effect on (or, as an extreme, completely determine) which religion a person chooses to belong to. However, one's religion also influences to whom we connect with socially, for example through the community in a church (the opposing extreme would be that a person's social ties are entirely composed of members of the same religion).

A different example, as mentioned above, is brand loyalty. Especially in cases where the quality of different products on the market is arguably very similar, social processes strongly influence one's decision in favor of a certain brand. Recent examples of this are smartphone brands, operating systems or browsers for computers but also more traditional examples such as car brands. Interestingly, brand loyalty in these cases has often been compared to religious mindsets.

In this context, it is also evident that the notion of ”opinion” in this work is seen as a more generalized one: An opinion could be a religious belief, affiliation to a political party, preference for a certain brand etc. Also the definition of nodes and edges in a graph can be seen as much more general than just an individual in a network of personal acquaintances. In the example of religion for example, a node could be a family and ”adapting an opinion” could mean a member of the family marrying someone of different religion, thereby changing the family's religious affiliation on the time scale of generations.

Another concrete restriction due to the definition of our model (see section 3.1) is that it only describes mutually exclusive, ”non-spectral” opinions, i.e. an individual can hold one and only one opinion which is independent of all other opinions (i.e. there are no similar opinions). The above examples can be seen as roughly falling into these categories. An interpretation in terms of political views is difficult, however, since political opinions are multi-dimensional (a person can have several opinions on different aspects of political life) and of spectral nature (some views or parties are similar to each other).

3 The Model

3.1 Initial Setup

In this work the network of opinion holders will be modeled by means of a graph with N nodes and M edges connecting the nodes. Thus ”graph” in this context is a collection of edges and nodes and not to be confused with graphs of functions. The entire model is based on the work of P. Holme and M. E. J. Newman [5].

Each node represents an opinion holder, e.g. a single person, indexed by i, to whom a certain opinion g_i is randomly assigned. The number of all possible opinions G is only limited by the number of nodes N existing in the network to a certain factor $\gamma = \frac{N}{G}$, where $\gamma > 1$. In this sense, it is not possible for every single individual to hold a unique opinion. In this model we are externally setting γ, i.e. the initial average number of nodes with the same opinion, as opposed to the number of existing opinions.

It is important to point out the nature of opinions this model assumes. Though different opinions will be denoted by a set of subsequent integers given by G, this is not to be mistaken with a spectrum of opinions with two extremes. That is no opinion is more similar to another than yet another. Each opinion is equally likely to be adopted by an opinion holder. This will have important implications on the type of empirical data the results of this model can be compared to, as already alluded to in the introduction.

The edges that connect two nodes represent a bridge of interaction of two nodes, i.e. direct interaction can only occur between two connected nodes. In general an individual edge can be directed, which means that information can only be carried from one node to another along its direction. In our model each connection will be undirected, i.e. two nodes can influence *each other*. In contrast to [5], we have forbidden self-edges and multi-edges to occur, as this is typical for acquaintance networks. In this sense a node as such is already connected to itself and can also not connect with certain nodes more strongly than with others via more edges. The number of edges leaving a node i is given by the degree k_i. The average degree of the graph is therefore given by $k_{avg} = \frac{2M}{N}$. By externally setting N and k_{avg}, edges are distributed in a random and uniform fashion: each pair of nodes is equally likely to be connected by an edge. This random graph generation was first proposed by Paul Erdos and Alfred Renyi [4]. There are many other types of graphs but due its simplicity and wide applicability, this work focusses on random graphs of different sizes.

Even though we are using the interpretation of nodes and edges as people with opinions and social connections for reasons of intuition, it is important to not restrict the notion of connected and interacting nodes to such a specific context. The concept of nodes holding and influencing each other's opinion is much broader as mentioned in the introduction.

3.2 Time Evolution Algorithm

The above-mentioned setup of nodes, edges and opinions is now subject to time-evolution. Specifically, nodes are allowed to interact such that a *co*evolution of edges and opinions occurs:

In two basic update steps, we will enable each node to follow one of the two mechanisms, reconnecting to a node of equal opinion *or* adapting his/her opinion to the neighborhood. A tunable probability Φ of choosing either one of them will be included in the model. In this work this probability will be referred to as *reconnection* *probability*. In more detail, at each time step the following will be done:

1. Pick a random node i with opinion g_i.
2. If $k_i = 0$, start over at step 1.
3. With probability Φ select at random one of the nodes j that i is connected to.
   a. If $g_i = g_j$, start over at step 1.
   b. Otherwise, reconnect to a randomly chosen j' of same opinion, i.e. $g_{j'} = g_i$.
4. Otherwise, with probability $1 - \Phi$ randomly select one of the neighboring vertices j and change g_i to g_j.

Here the third step corresponds to the mechanism in the network of an individual reconnecting to like-minded individuals, whereas the fourth step to that of an individual adapting his/her opinion to his/her neighborhood.

Steps 1. to 4. are iteratively performed until a convergent or equilibrium state is achieved. The attentive reader might have noticed in the steps above that step 3a could result in an infinite loop as soon as every node is only connected to nodes of equal opinion. This is exactly the definition of the convergent state and will be referred to as the *consensus state*. We will be mainly concerned with how this convergent state looks like. Separated clusters of different opinions will have formed, whose size distribution in dependence of several external parameters, namely Φ, N and k, this work is mainly concerned with.

Intuitively, one can picture two extreme scenarios: one in which there is one prominent opinion, and one in which there is no such opinion but several much less prominent ones. In other words, an emergent phenomenon of different manifestation is to be expected as macroscopic outcome from the microscopic steps described above.

4 Theory of Critical Phenomena

Several analogies can be drawn from the system of interacting nodes to systems in solid state physics. Therefore some formulas and concepts of this field will be applied in this work, all of which will be derived in this section and translated to our case. The works of [10] and [6] served as main source for this. The derivations in this section are also motivated by an attempt to understand the formulas stated in [5] without further derivation. It has to be stressed that some analogies arise from assumptions made that would in fact need further justifications and investigations which go beyond the scope of this work.

As indicated in the previous section, two extreme scenarios are possible in the opinion network. Speaking in physics terms, two phases that can transition into each other are expected to occur. An individual phase is characterized by several internal parameters, most importantly the so-called *order parameter*. In this sense one can also speak of an *ordered* and *unordered* state. If at some point this parameter changes drastically upon slight variation of another external parameter, one can speak of a *phase transition*. This point of drastic change is called the *critical point*. This sensitive response or criticality of the system typically manifests itself in a divergence of response function. One can also speak of a divergence of the *coherence length*, meaning that the effect of a slight change at some point in the system can influence the whole system.

A simple example is the boiling of water. Upon slight variation of the (external) temperature near the boiling point, a drastic change of (internal) density is observed: the unordered liquid state transitions into an ordered gaseous state. In this work the external variable will be the reconnection probability Φ, whereas the internal order parameter will be the relative size of the biggest opinion cluster, denoted by S. The critical point at which the phase transition occurs will be denoted by Φ_c. Further, the two phases occuring will be referred to as the *unordered* and *ordered phase*, the latter of which being marked by a high value of the order parameter S.

The behavior of a system near its critical point is characterized by a single quantity, the dimensionless *critical exponent*. These exponents are at the heart of the theory of critical phenomena. According to this theory, any physical quantity that is dependent on a specific external parameter follows a power law close to the critical value of this external parameter. In statistical physics, for example, all quantities that depend on temperature would behave the same way close to the critical temperature. Therefore, there is exactly one critical exponent per external parameter. How close the system is to the critical point is generally expressed by a *reduced* quantity, e.g. the reduced temperature $\tau = \frac{T-T_c}{T_c}$. In our case we will define the reduced reconnection probability likewise as

$$\varphi = \frac{\Phi - \Phi_c}{\Phi_c}. \tag{1}$$

The dependent variable is the relative maximal cluster size $S(\varphi)$. Its behavior near the critical point Φ_c is described by

$$S(\varphi) \sim |\varphi|^{\alpha}. \tag{2}$$

The tilde is read as "behaves as" for $\varphi \to 0$. The absolute value is based on the general observation that the direction from which the critical point is approached, does not matter. In this work we do not test this equivalence. The critical exponent is denoted by α. It turns out that seemingly very different systems have the same critical exponent and therefore behave the same way near the critical point. According to the hypothesis of universality, they are said to belong to the same universality class uniquely defined by one critical exponent for each external parameter. In our case we do only vary one external parameter, i.e. the reconnection probability, for which we can determine the critical exponent and check which universality class it belongs to. The universal critical exponent itself is only dependent on the following:

1. Dimensionality d of the system
2. Range of interaction
3. Spin dimensionality (irrelevant in our case)

Translating all of these dependencies to our case is rather difficult. The dimensionality of a graph that carries no spatial information and in which each site can interact with every other site, unlike a regular crystal lattice for example, cannot be pictured easily if at all. The case is simpler with the range of interaction. The concepts presented in this section are based on an ideal system of infinite size (thermodynamic limit). If we limit the size of our system, i.e. we set N to a certain finite value, the range of interaction that would usually diverge (response functions diverge) at the critical point is limited by N. It is therefore to be expected that for different network sizes different critical exponents will be obtained. This will manifest in lines of different slopes intersecting in the critical point in a log-log plot.

This issue solved with the so-called scaling law that enables us to correct for different system sizes and extract one single critical exponent α according to (2) that is independent of the size of the system. In the following we will derive this relation and state how the critical exponent can be determined.

In the context of statistical physics all functions of quantities that undergo a phase transition, are some derivative of the free energy F of the system that has to be minimized in order form a low energy ground state, i.e. a new phase. In our case the ground state is the consensus state, which is of low energy in a sense that no disagreeing nodes are connected. We further assume that the maximal cluster size is some derivative of F. It can be proven that the free energy must be a generalized homogeneous function that follows

$$F(\lambda^x \varphi) = \lambda F(\varphi), \tag{3}$$

where λ is some real scalar that we can set to N^d, where d is the dimensionality of the system. Here x denotes some exponent of which the critical exponent will be dependent. Further we need to consider the free energy *density* that is independent of the system size. We get

$$f(N^x \varphi) = N^d f(\varphi), \tag{4}$$

which can be derived with respect to its only internal parameter φ to get

$$N^{x-d} \cdot \frac{\partial f(N^x \varphi)}{\partial N^x \varphi} = \frac{\partial f(\varphi)}{\partial \varphi}. \tag{5}$$

In the next step we are assuming that $\frac{\partial f(\varphi)}{\partial \varphi} = S(\varphi)$. This step is by no means trivial and needs further justification. However, it leads to equation (1) in the work of [5] that was used for determining the critical exponent and will be employed in this work as well. We can thus write

$$S(\varphi) = N^{x-d} S(N^x \varphi) \tag{6}$$

or to make it more comparable to [5],

$$N^a S(\varphi) = S(N^b \varphi). \tag{7}$$

This equation will be used to extract the values of a and b from curves of $S(\varphi)$ of different N. In order to get the critical exponent, (7) has to be put into the form of (2). In order to do so, it can be chosen that $N = \varphi^{-1/b}$ such that $S(1) = const.$ The power law behaviour of $S(\varphi)$ is therefore given as

$$S(\varphi) \sim |\varphi|^{\frac{a}{b}} =: |\varphi|^{\alpha}. \tag{8}$$

5 Implementation

The model described in section 3 was implemented and simulated with MATLAB® (Version R2012b). This section will outline the structure of our implementation and data organization by using pseudocode.

At the heart of our implementation lies the representation of a network of N nodes and an opinion distribution using matrices and vectors, respectively. An NxN matrix A_{ij} with entries 1 or 0 was used to represent existing or non-existing connections between nodes i and j. The entries could take binary values only as multi-edges were not considered. Since only undirected nodes were allowed, the matrix was perfectly symmetric, i.e. a connection between i and j was written as both the (i, j) and the (j, i) entry being equal to 1. Self-edges were also disallowed, so the matrix diagonal was 0 by definition. Hence, all valuable information is carried in the upper triangle of the matrix.

Different opinions were simply labeled as subsequent integers into a vector of length N with the indices of the vector representing the nodes.

5.1 Main Script

Upon implementing the model, a functional programming approach was taken in which logically closed subroutines (e.g. the generation of a random network or the time evolution steps of a graph) were written as closed, independently callable functions. All functions and commands necessary to carry out the simulation were called from a "main script" file, which was the script executed to start the simulation. The structure of this main script is outlined in Algorithm 1.

```
Input: $k_{avg}$, $\gamma$, $N_{range}$,$\Phi_{range}$, $runs$
foreach $N$ in $N_{range}$do
    generate random graph;
    generate random opinion distribution;
    both are kept constant for one N;
    foreach $\Phi$ in $\Phi_{range}$ do
        foreach run in runs do
            execute simulation;
            sum up results;
        end
        average over results;
        save results;
    end
end
```

Algorithm 1: Structure of main script

First the input parameters, which are the average degree k_{avg}, the average number of nodes per opinion γ, and parameter ranges for different system sizes N and Φ that are defined in the beginning. Then the different system sizes are looped over whereas one random graph and one random opinion distribution are generated for each system size and kept for that iteration to ensure consistency. With this initial network, the range of Φ is iterated over. For each Φ, the time evolution is executed a specified number of times in order to be able to calculate average values of the results. The results, above all the cluster size distribution and the order parameter, are then saved. Saving usually included only the average values since saving for each individual run would have created unnecessary large amounts of data, although this is useful in the beginning to ensure the simulation is operating correctly.

5.2 Random Graph Generation

Generating a random graph is a seemingly simple task, even with the restrictions of our model (no self- and multi-edges allowed) being taken into account. Since we needed the number of edges in the network to be exactly as specified, the *uniform* $G(N, M)$-variant was chosen over the $G(N, p)$-variant of Erdős-Rényi algorithm [4]. In the latter variant k_{avg} would be subject to small fluctuations. As we were going to investigate critical phenomena that are potentially very sensitive to changed in the external parameters, we wanted to make sure to exclude such fluctuations and wrote our own uniform random graph generator as shown in algorithm 2.

```
FUNCTION;
Input: $N$, $M$
Output: $A_{ij}$
initialize empty matrix;
initialize counter;
while counter $\leq$ Mdo
    choose random i;
    choose random j;
    if (i == j) or (i, j already connected) then
        continue;
    end
    else
        set $A_{ij}$ and $A_{ji}$ to 1;
        increment counter;
    end
end
```

Algorithm 2: Generation of a random graph

The pseudocode is fairly simple and straightforward. A randomly chosen pair i, j is connected symmetrically (in both directions) unless $i = j$ (self-edges forbidden) or i and j are already connected. Using a counter and a while-loop, we made sure that exactly M edges are placed randomly.

The generation of an initial opinion vector is in fact very simple. For each entry of the opinion vector, one of G possible opinions is selected at random.

5.3 Simulation Script

The central feature of our model is the time evolution of the network and its opinion structure. Though the steps of updating the network have already been described in section 3.2, some of the exact details of the process are actually not trivial to implement in Matlab. Thus, for reasons of reproducibility, the time evolution of the simulation is presented here in pseudocode 3 and the important details are clarified in words.

The outermost loop represents the iteration over time steps. This needs to be a while loop since we want to keep updating until a consensus state has been reached, which occurs after some previously unknown convergence time t.

It is important to note that at each time step, we *first* perform a check as to whether or not the consensus state has already been reached. This is because we continue the loop (i.e. pick a new (i, j) pair) each time i and j by chance already have the same opinion. If we were to check for the convergence after choosing i and j, we would eventually get stuck in an infinite loop. The decision of not letting i and j interact if they are already of the same opinion is by no means a trivial one and stands in contrast to [5], as discussed in section 3.2.

On the basis of performance optimization, the check for a consensus state is formulated in a negative way, i.e. we break the convergence check if any pair of connected nodes is *not* of the same opinion: We do not need to waste time and iterate through all other pairs of neighbors if even a single one is not of the same opinion. One such pair is enough to conclude that no consensus has been reached yet.

Another non-trivial technicality is to require a pair of potential new connection (i, j') to not be already connected. Otherwise, i.e. if they are already connected, we would overwrite this existing connection whilst deleting the old one. This would decrease the total number of edges M in the network, which needs to be constant.

```
FUNCTION;
Input: $A_{ij}$, opinion vector, $N$, $\Phi$
Output: $A_{ij}$, opinion vector, convergence time $t$
initialize convergent state binary abort ; initialize $t$;
while convergent state not reached do
    /* Check if convergent state has been reached:        */;
    foreach node $x$ in $N$ do
        foreach node $y$ connected to $x$ do
            if opinion($x$) != opinion($y$) then
                no convergent state reached;
                break out of convergence check loops, continue at time evolution;
            end
            else
                set abort binary to indicate that convergence has been reached;
            end
        end
    end
    /* Perform time evolution:                     */;
    increment $t$;
    pick random node $i$;
    if $i$ is at all connected to any other node then
        pick random node $j$ that is connected to $i$;
        if opinion($i$) == opinion($j$) then
            continue, i.e. pick new $i, j$ pair and start over;
        end
        if probability $\Phi$ then
            pick random node $j'$ out of nodes with same opinion as $i$;
            if $i$ and $j'$ NOT already connected then
                delete connection between $(i, j)$;
                establish new connection between $(i, j')$;
            end
        end
        else probability $1 - \Phi$
            Set opinion($i$) = opinion($j$);
        end
    end
end
```

Algorithm 3: Simulation Function: Time Evolution of Network and Opinion Structure

5.4 Optimization and Data Storage

Several means of increasing computation and storing efficiency have been realized. Pre-defined MATLAB functions have been employed wherever possible. To save memory, sparse adjacency matrices have been used instead of full ones. As already mentioned in the previous subsection, we optimized our simulation function such that unnecessary iterations are broken off early, e.g. when checking for a convergent state or upon having chosen two nodes of already equal opinion.

Our main script is in principle parallelizable as the two input parameters that we varied (N and Φ) are producing independent data points in the parameter space. Though we did not use the parallelization toolbox, we usually ran multiple MATLAB sessions at the same time to take advantage of multi-core processors. We also prepared a version of our simulation that is executable on ETH's Brutus cluster (see appendix), but were not able to take full advantage of it due to missing licenses of the frequently used bioinformatics toolbox.

As already mentioned above, functions have been logically separated into initial generation, time evolution, and data saving. Plotting and analysis were performed after the simulations had finished. For this, the entire workspace after the simulation was saved to a file. The file was named with a string containing the parameter combinations that were used in order to keep an overview. This string was also used by the different plotter scripts to open and analyze data for a desired combination of input variables. When creating some plots, e.g. the (rescaled) order parameter vs. φ or the average convergence time, the entire data was loaded and plotted by "walking through" the data output directory. Though this process could have been improved, we did not spend too much time on optimizing scripts that take only? the order of seconds to run. The codes of data plotting are not further explained with pseudo codes but included with comments in the appendix.

6 Results and Discussion

In this section, the results of the performed simulations are presented and discussed. It is divided into subsections labelled by the focus of the particular aspects that were analyzed.

6.1 Cluster Size Distribution

In this subsection we investigated the effect of different reconnection probabilities on different convergent states. As stated before, this state is made up of several decoupled clusters all of whose nodes hold the same opinion. As we are dealing with non-spectral opinions, we are not interested in which particular opinions are held by clusters but how their size is distributed in the consensus state. As already alluded to in section 3, we observed a phase transition between two fundamentally different opinion cluster distributions. The two extreme scenarios or phases are the following.

The first scenario results from a low reconnection probability, i.e. nodes are more likely to adapt to their neighborhood. The initial opinions compete against each other over time, some of them dying out others being dominant in the end. The resulting consensus state is therefore marked by the occurrence of one giant cluster among several smaller ones (Fig. 1).

The second scenario results from a high reconnection probability, i.e. nodes are more likely to keep their opinion and reconnect to like-minded nodes. Therefore the time evolution does not majorly change the initial distribution of opinions but the connections of their holders. This results in a consensus state in which there is no giant cluster but only several smaller ones (Fig. 3).

A more quantitative description as well as how the two scenarios transition into one another is presented and discussed in the following. In this subsection we only vary the reconnection probability Φ as external parameter, while keeping all others constant. In the specific results presented here this is $N = 400$, $k = 4$, $\gamma = 10$. Moreover, the data points were obtained after averaging over 200 runs. The data is presented in a normalized histogramm of cluster size s. We can therefore also speak of a probability distribution $P(s)$. Since the biggest possible cluster is limited by the size of the network, we will have $s \in [1, N]$.

In the first plot of Fig. 1, one clearly sees power law distribution for small cluster size in the approximate range of $[1, 20]$ as well as a peak near the maximal cluster size of $s = N$. We therefore refer to this scenario as the *ordered phase.* It is quite remarkable that in the relatively wide range, approximately being $[40, 300]$, no clusters are to be found. For low values of s the data points agree well with a fitting line, whereas for higher values random fluctuation around the fit occur. This is simply

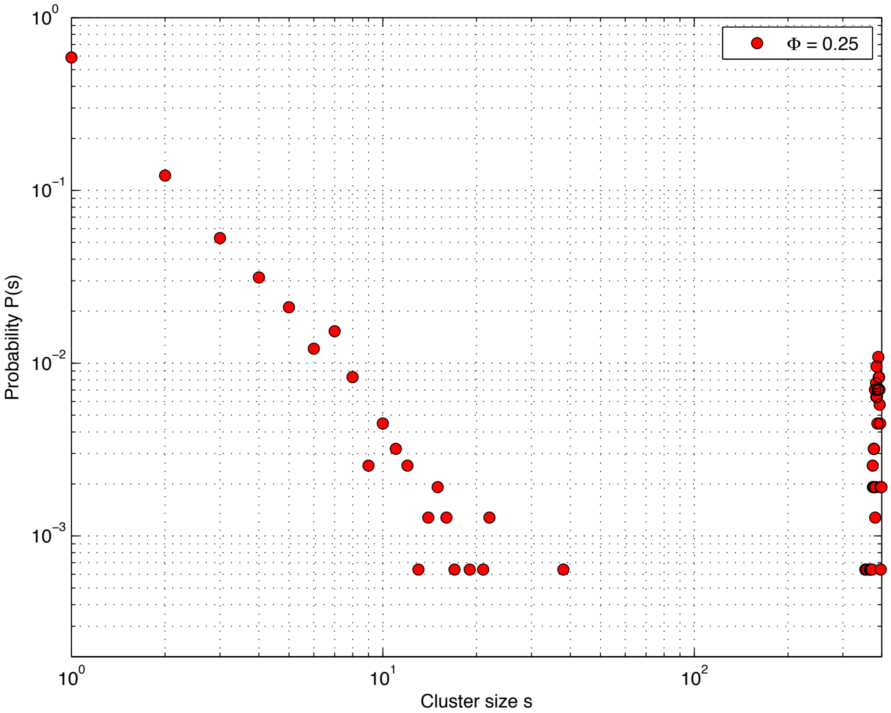

Figure 1: Cluster size distribution for $N = 400$ at $\Phi < \Phi_c$.

due to the fact that the averaging works better for smaller cluster sizes because a certain value for a small cluster size is more likely to be obtained than for a bigger one. This fact is most pronounced in the region of the giant cluster. There the data points seem to be on top of each other, creating a sharp peak. In fact, each one of them sits on another value of s that has been realized only a few times in all of the runs and thus is not averaged over many times.

Before turning our attention to the transition point, we discuss the third plot of Fig. 3. Here the situation is very different as there is no giant cluster. In fact, the maximal cluster is smaller than 30, which is less than 10% of the giant cluster size for small Φ described above. Instead, the consensus state is marked by many small clusters. We therefore refer to this scenario as the *unordered phase*. What strikes even more is the very smooth alignment of data points. The maximum of this distribution is approximately at $s = 10$, which is exactly the value that was set for γ,

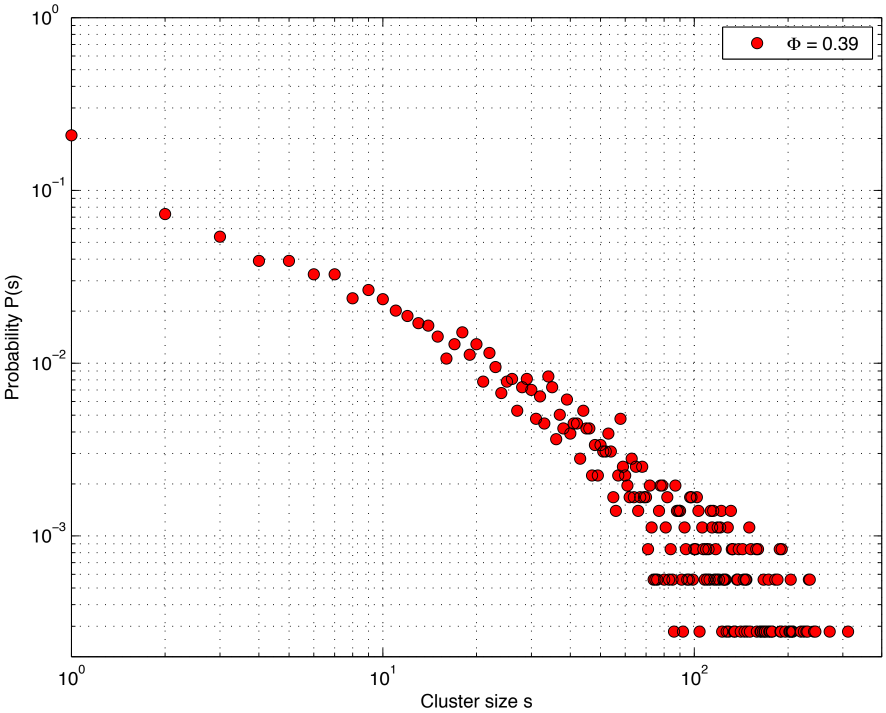

Figure 2: Cluster size distribution for $N = 400$ at $\Phi \approx \Phi_c$.

i.e. the average number of nodes holding the same opinion. Upon analyzing a wide array of distributions for different system sizes (not shown here), it can be observed that the maximum of the unordered distribution is always at approximately $s = \gamma$. This is certainly no coincidence and can be explained as follows. For Φ close to unity only very few opinion changes are allowed. Consensus is achieved mostly entirely through reconnection. As γ is set for the initial setup of the opinions, which will - due to the high value of Φ - barely change, the most likely cluster size will be exactly γ. This fact is shown for a specific value of Φ and N but is independent of N and Φ as long as the latter is above Φ_c. Since the cluster size s is given by the (integer) count of nodes in one cluster, the distribution can be considered as Poisson-like. This is also stated in the work of [5].

In our case, however, the resemblence is certainly not perfect. Most notably, the tail at the lower end of the distribution performs an increase to another local maximum at $s = 1$. Clusters of small size remain to be very likely to occur. The reason for this can be stated as follows: Small initial clusters in which members hold different opinions will reach local consensus very fast. The only chance for them to connect to other clusters to form bigger clusters, is that another node of the same opinion decides to reconnect to this small cluster. As for high γ this is of course rather unlikely, smaller clusters remain ”forever alone” in their local consensus state. This is most pronounced for lone nodes. They are in consensus by themselves and most unlikely to be connected to.

Now the question that remains to be answered, is *how* and *when* the transition between the two phases occurs. These can be answered when looking at the plot of 2. A rather straight line that spans almost over the whole range of s can be seen. The approximate value of Φ for which this occurs is $\Phi_c = 0.39 \pm 0.05$, where the error has been estimated from visual judgement of the cluster size distributions. This is only a first guess and will be investigated in the following subsections.

The curve combines aspects of both phases: there is a continuous distribution as well as a relatively big cluster, resulting in a power law distribution of clusters. This is an indicator for the continuous nature of the transition and is corroborated when considering the full range of Φ for the cluster size distribution. There it can be seen that for increasing Φ the giant cluster shrinks and smears out, thus closing the gap to the small clusters and forming the smooth straight line at the transition point. When further increasing Φ this straight line bends to form the Poisson-like distribution.

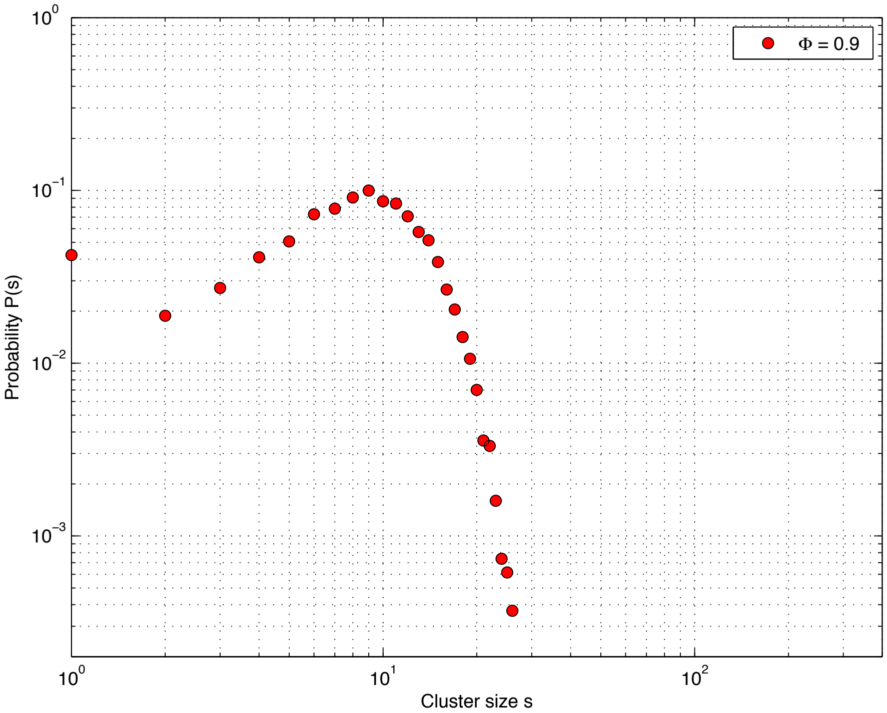

Figure 3: Cluster size distribution for $N = 400$ at $\Phi > \Phi_c$.

In the following subsection 6.2 the observation of a continuous phase transition will be further investigated and supported by further arguments.

6.2 Critical Point and Rescaling

A more exact way of finding the critical point Φ_c is realized when altering the network size N. As stated in section 4 different values for N limit the range of interaction, i.e. the divergence of the coherence length, in the system. Therefore different critical exponents and hence different slopes around a point of intersection are observed. Smaller networks should be characterized by a less dramatic transition, i.e the slope at the critical value should be smaller than those of bigger networks. Qualitatively speaking, this is because in smaller networks it is more likely that a consensus state is achieved in the ordered phase without a very big giant cluster having formed. As a consequence the order parameter S (which is introduced in this section) will on average be lower for smaller networks in the ordered phase. On the other hand, in the unordered phase the order parameter of smaller networks will on average be higher. This is because in smaller networks an equally often occurring reconnection is more likely to result in a bigger cluster. In short, closer to the thermodynamic limit of $N \to \infty$ the transition will be more ideal, i.e. more dramatic.

For these reasons and the intuition that the critical point should not depend on the system size (the boiling point of water is the same for 1ml and 10 ml), the point of intersection of the different curves for different N can be interpreted as the wanted critical point Φ_c. In Fig. 4 these curves are presented. In this figure each circle denotes a data point that was obtained from a cluster size distribution of the type presented in the previous subsection. From those distributions the size of the biggest cluster was divided by the network size, which yields the order parameter $S = s_{max}/N$. A sufficient amount of curves of different values of N have been chosen such that the critical point is clearly visible. In general greater networks result in smoother curves. However, the computation exceeded reasonable limits, especially in the critical region (also see next subsection). Moreover, the difference of the curves becomes less and less visible. Hence, values for $N > 400$ were not included.

The individual curves in the plot resemble those of a continuous phase transition. This is in agreement with the qualitative observation of the cluster size distribution in the previous subsection. A high order parameter marks the ordered phase, which is separated from the unordered phase (marked by a low order parameter) by the critical point. The critical point itself is found to be at $\Phi_c = 0.32 \pm 0.02$, where the error was estimated from the Φ-steps taken and the relative overlap of data points near $\Phi = 0.32$. This value does, however, not exactly agree with the value obtained in the previous subsection. In Fig. 2 the respective critical value is at $\Phi = 0.39 \pm 0.05$. There is, in fact, an overlap at $\Phi = 0.34$. But the following considerations will demonstrate that by applying the scaling laws introduced in section 4 pretty much any value of Φ can be turned into a critical point. We will show this for the example of $\Phi = 0.39$, also to reproduce the method used to quantitatively obtained the critical value in the work of [5].

In Fig. 5 a rescaled order parameter, $N^a S$, was plotted against a rescaled reduced reconnection probability, $N^b \varphi$, according to (7). In this form a can be used to stretch or compress the curves in vertical direction depending on N, while b can be used to do this in horizontal direction. Following the procedure outlined in the work of [5], first a is tuned such that a neat point of intersection does occur at all and at $\varphi = 0$. Without adapting b (keeping it zero) yet, the resulting plot is shown for $a = 0.4$ in Fig. 5.

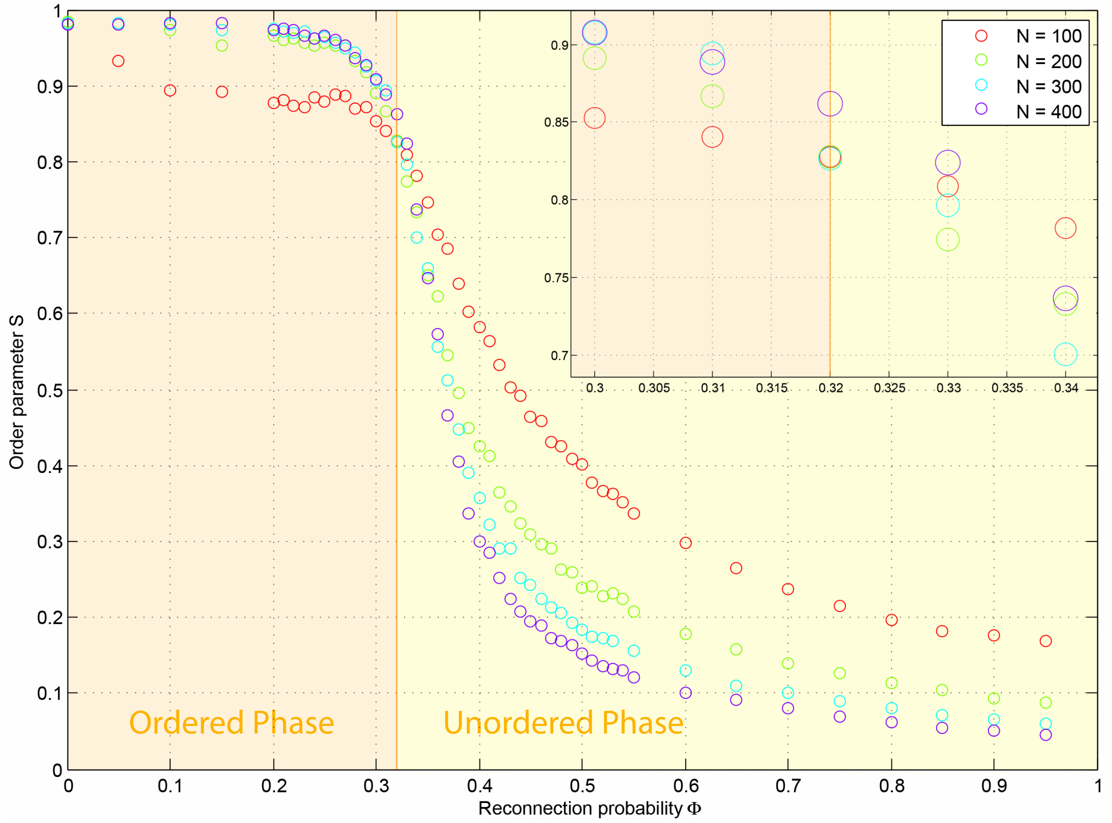

Figure 4: Order parameter versus reconnection probability for different network sizes.

In the second step b can be tuned such that the different $S(\varphi)$-curves collapse into one single curve near the transition point, indicating the independence of the system size N. This was already pointed out in section 4 and is expressed in the fact that we get a single critical exponent α according to the (8). This collapse of curves in the critical region was obtained for a value of $b = 0.8$. The plot is presented in 6.

Having determined a and b, the critical exponent α could be calculated according to (8). It turns out to be $\alpha = 0.5$. This value is not presented with errors as this is a very crude if not arbitrary estimate. A comparison and assignment to universality classes of phase transitions would therefore not make any sense. Again, the calculation of α was mainly intended to reproduce the method of [5] and to show that the calculation of a critical exponent is in theory possible. The main difference in our work is that we obtained a neat point of intersection without having to tune a to a non-zero value. The only puzzling consequence is that this would correspond

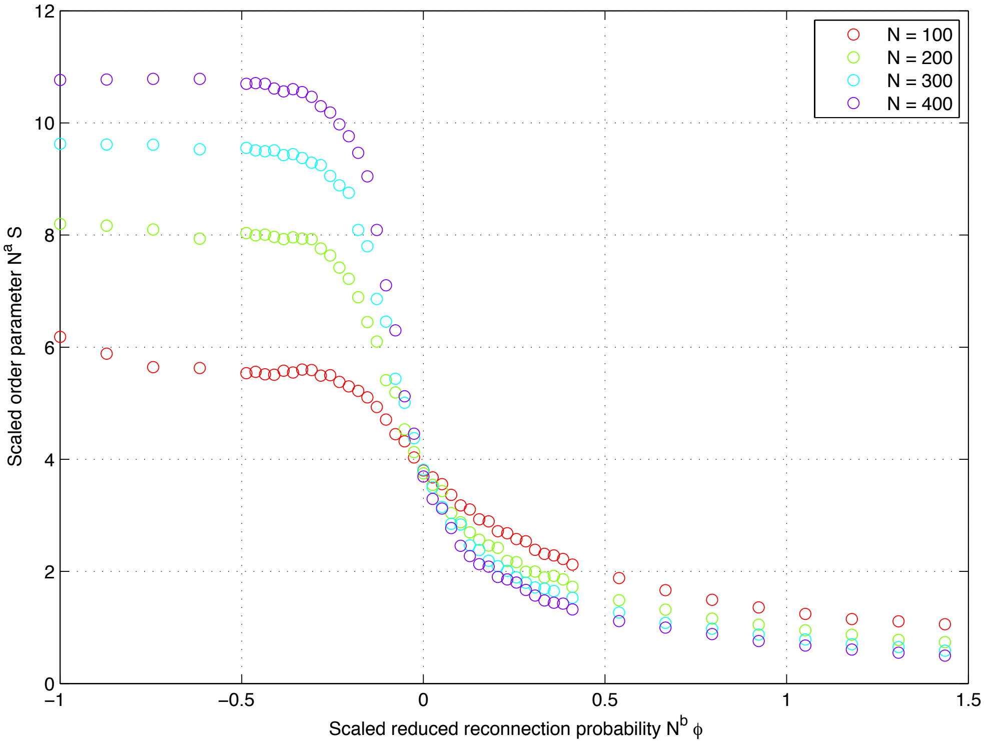

Figure 5: Order parameter versus reconnection probability for different network sizes.

to a critical exponent of zero according to (2), which clearly cannot be the case.

The demonstrative example above was based on the indication of Fig. 2 that $\Phi_c = 0.39$. The fact that we do have a neat point of intersection of the $S(\varphi)$-curves and thus a relatively clear critical point of $\Phi_c = 0.32$ is, however, stronger evidence. Additionally, the following subsection, which analyzes the convergence time of the system, provides even more evidence in favor of $\Phi_c = 0.32$.

6.3 Convergence Time

The focus of our analysis was on the cluster size distribution and the arising phase transition. However, another interesting quantity is the relative average convergence

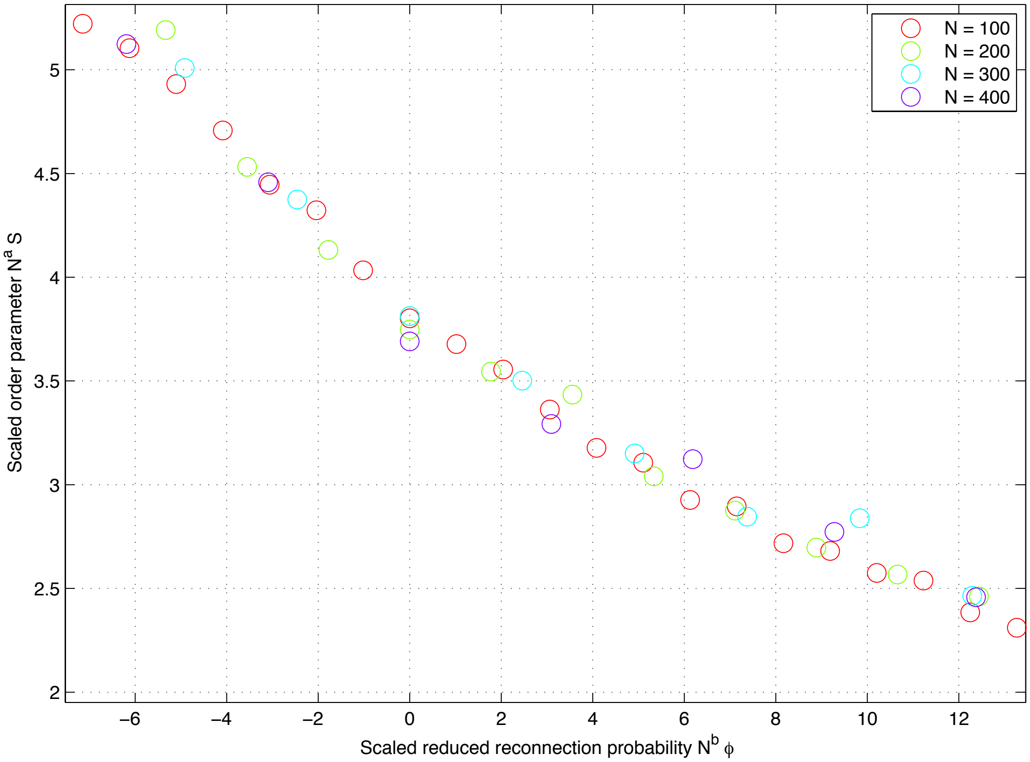

Figure 6: Zoom into critical region of rescaled order parameter versus rescaled reduced reconnection probability. Here $a = 0.4$ and $b = 0.8$ such that the curves for different N collapse into a single curve.

time τ_{avg}. This is the number of iterations relative to the number of nodes N that the system takes (on average) to reach the consensus state. We initially recorded this quantity for each parameter combination and each run that we performed to optimize our simulation and parallelize the scripts most efficiently. However, upon plotting τ_{avg} in dependence of the reconnection probability Φ, we noticed that it is also conceptually relevant. Fig. 7 shows a plot of τ_{avg}/N vs. Φ for different system sizes N.

It is clearly visible that the there is a value of $\Phi \approx 0.32$ at which the convergence time reaches a sharp peak which is approximately the same for all N. We can also

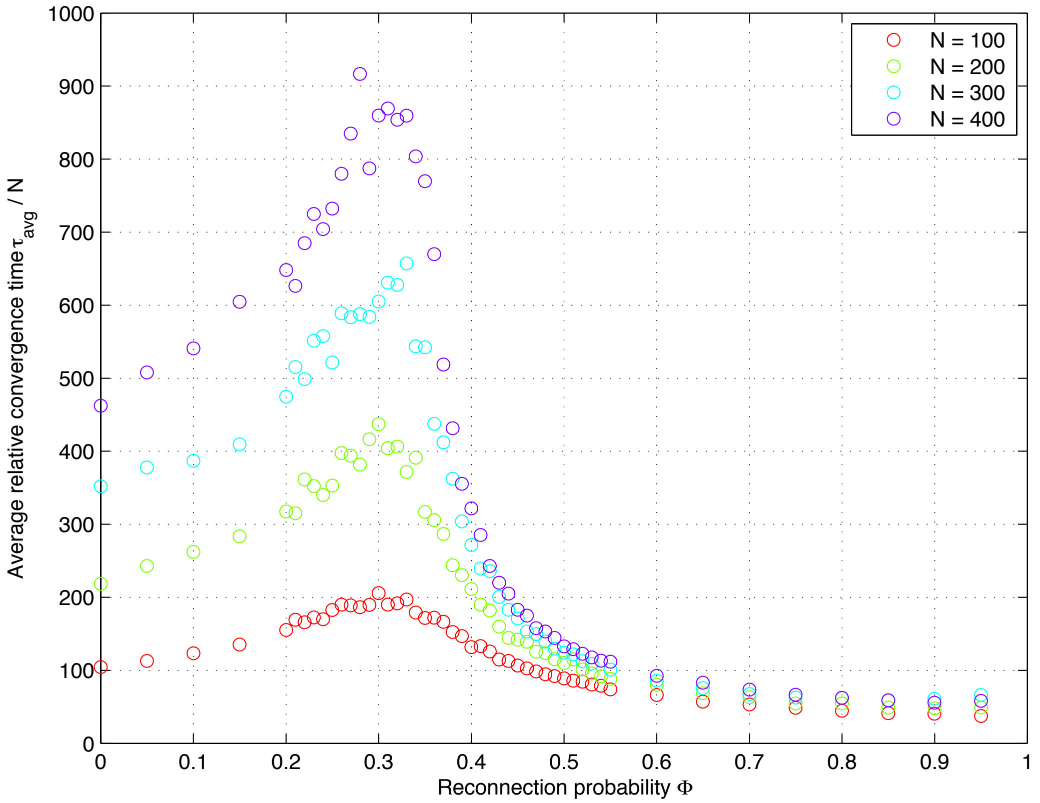

Figure 7: Average time per node to reach consensus as a function of Φ

observe that the peak convergence time and the time for low values of Φ is significantly larger the greater the system size, while for $\Phi \to 1$, we need roughly the same number of sweeps to reach consensus. More importantly, the peak seems to sharpen with increasing system size.

The sharpening of the peak and the shape of the curves as such bear a remarkable resemblance to divergent response functions or susceptibilities at the critical point as they appear in solid state physics. For physical systems, in fact, a divergent response function is part of the very definition of a phase transition. A divergence in such cases typically occurs in the (generally n-th order) derivative of some thermodynamic potential. For example, the magnetic susceptibility, which is proportional to the second derivative of the Helmholtz free energy with respect to magnetic field, diverges (i.e. becomes infinite in theory) at the critical temperature of a paramagnet-ferromagnet transition. This divergence, however, only occurs for infinitely large systems. For finite systems, the infinite peak broadens as the systems size is decreased.

Precisely this behavior can be seen (at least qualitatively) in our case, which reinforces our previous conclusion that the process of opinion and network formation in our model indeed obeys a phase transition. From this picture, we would obtain a critical probability $\Phi_c \approx 0.32 \pm 0.05$ which is not necessarily in agreement with the transition value determined in the previous section. However, as was already established, determining the critical point is very difficult with our given accuracy.

Additionally, it is tempting to try to identify the convergence time of the network as some sort of response function or susceptibility, although it turned out to be difficult to find a straightforward physics analogy in this case. If we assume, as we did in section 4, that the reconnection probability Φ can be seen analogous to a temperature, it is very hard to point out a quantity in our system that corresponds to an external field. Also interpretations of τ_{avg} as a heat capacity or entropy are not very intuitive. Despite the difficulties to find an analogy to a response function, we remain confident that the peaks of the convergence time at the same critical value can hardly be a coincidence. Further investigations of this would certainly be interesting (see section 7).

From a purely handwaving and intuitive point of view, however, it does make sense that the convergence time increases sharply at the (presumably) critical Φ_c. After all, this is the point at which the two processes of opinion change and reconnection are in tough competition: What would otherwise be nearly a consensus state in the case of either small or large Φ is likely to be broken and pushed towards another phase by the respective other process, stretching the time until a consensus is reached.

6.4 Comparison with Empirical Data

As what was proposed to be an optional task, we also set out to compare the results, i.e. the opinion size distribution, with actual data on opinions of a certain aspect, e.g. religious view. One could then think about by what factors our Φ is determined in reality and how to interpret it, if at all possible. Ideally, one could try to find data on the distribution of opinions, fit a function to it and based on the best match to the distribution of opinion cluster, determine the Φ that this real-world distribution situation corresponds to: An indicator for whether the opinion structure in question is formed rather by reconnection in the network or by adoption of opinions.

As already expected, getting "good" data that could be easily compared to theoretical predictions was not easy. Hence one possible simplification we thought of was instead of comparing entire distribution, it could also be an option to simply compare the size of the largest group of an opinion with our order parameter, which is a measure of exactly that. That way, only a single scalar value would have to be compared instead of a whole distribution function.

Due to the nature of the data, it also proved to be difficult to even generate cluster size histograms of the type in section 6.1 since in most real world cases just a handful of data points (opinions) even exist by the nature of the problem. For example, there is only a handful of mobile web browsers on the market. It is thus difficult to count the number of browsers of the same size since the count will always be one unless we significantly increase the binning interval. In such cases, we at least provided some qualitative observations and arguments.

As already argued in the introduction, our model is only applicable to generalized opinion structures that are mutually exclusive and "non-spectral", i.e. opinions exist independently without similarities.

6.4.1 Religious Views

Since the analysis of the distribution of religious groups was the initial conceptual motivation of this project, we naturally tried to find sources to compare this type of data first. Getting reliable and finely-resolved statistics on religious adherence, however, is not an easy task. Also the definition of what the term "religion" entails varies from source to source. Especially when separating confessions into separate religions or aggregating them as one data point dramatically affects the distribution of sizes, which is exactly what we are trying to analyze and compare.

One already existing analysis is that of the total size distribution of religious communities globally, which happens to roughly follow a power law [5] [9]. Fig. 8 (which is directly taken from [9]) shows this behavior. The paper cites the website ”adherents.com” as the source of its data.

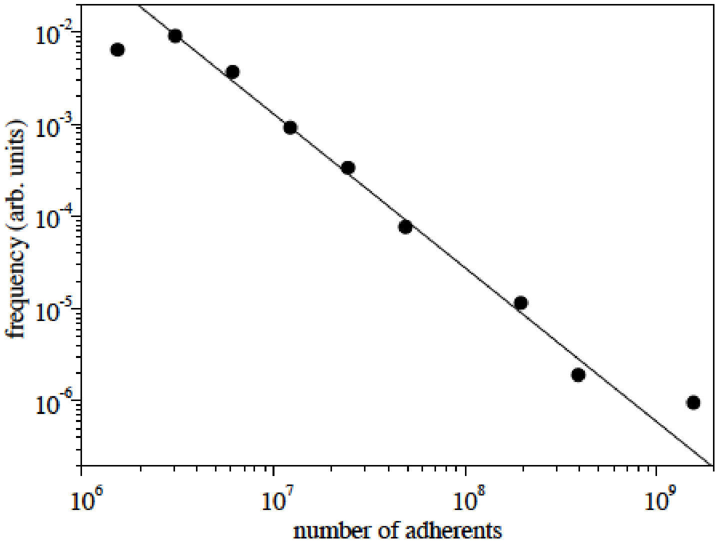

Figure 8: Size distribution of religious groups worldwide as found in [9]

In our model, we found a power law behavior of the cluster size distribution around the critical point! Comparing this to the empirical statistics would lead to the conclusion that religion is a critical phenomenon of opinion formation, of course under the assumption that our model is indeed applicable to reality. This would mean that the formation of religious views is not dominated by either of the two processes.

As already mentioned, finding further data for good quantitative analysis was rather difficult. However, without having to look at further data we can take the comparison to a qualitative argument: We know from experience that there are societies in which the distribution looks different and where one religion is completely dominating, for example in countries such as the USA or Saudi-Arabia. In such cases, the corresponding phase of our model would be the ordered one generated by a small reconnection probability Φ, which means that individuals tend to adapt rather than reconnect.

Additionally, one needs to note that applying our model of opinion *formation* to religion brings about some theoretical objections, e.g. that religion is an opinion that is rarely chosen freely (much less randomly) by an individual at some initial point. Thus the religious distribution within a region or country is unlikely to evolve significantly at all due to individual-level social processes. Thus we also thought of an alternative interpretation of ”nodes” as families adhering to a particular religion, as already briefly mentioned in the introduction. In this interpretation, one could think of the process of opinion adoption as one where a family member changes his/her religion due to marrying a member of a family of different faith that he is connected with, thereby changing the religion of his/her entire family in the long term (on the time scale of a generation). On the other hand, a reconnection would correspond to a family secluding itself from families of other religions. Keeping this in mind, one could think of the reconnection probability Φ as a kind of *intolerance indicator* towards other religions: A low Φ would persist in a society in which intra-faith marriages are tolerated or even encouraged, while a high Φ would mean the opposite.

Looking at the results of our simulations, one could continue to say that a high intolerance indicator leads in the long term to a society in which one religion dominates strongly (ordered phase) while in a tolerant society, religions follow a much ”softer” distribution (unordered or critical phase). While this may at first glance satisfy our intuition, one needs to be extremely careful with (potentially judgemental) conclusions such as ”societies with a dominating religion are intolerant” since the validity of this interpretation is essentially speculation and very hard to prove. In fact, it is just as plausible to say that in the course of history borders of states were likely to be defined according to giant clusters of a religious view. Pakistan and India are prime examples for this.

6.4.2 Mobile Web Browser Market Share

A second possible interpretation of the generalized notion of ”opinion” that our model could be applied to, is consumer preference or brand loyalty. In this subsection, the example of mobile web browser usage is analyzed. Under the (of course idealized and by no means obvious) assumption that all internet browsers for mobile devices are in fact objectively of equal quality, we can interpret an individual’s choice in favor of one particular browser as an opinion. Since the individuals in a network of mobile browser users can be assumed to be socially interacting, we could try to compare such a scenario to our proposed model.

In doing so, we once again faced the problem of obtaining proper data that would be displayable in a histogram of cluster sizes. In essence, there are too few browsers to generate a meaningful histogram. Instead, we used the market shares of each browser as a proxy for the number of people who prefer one particular brand of mobile phone used to surf the web and make some qualitative arguments based on that.

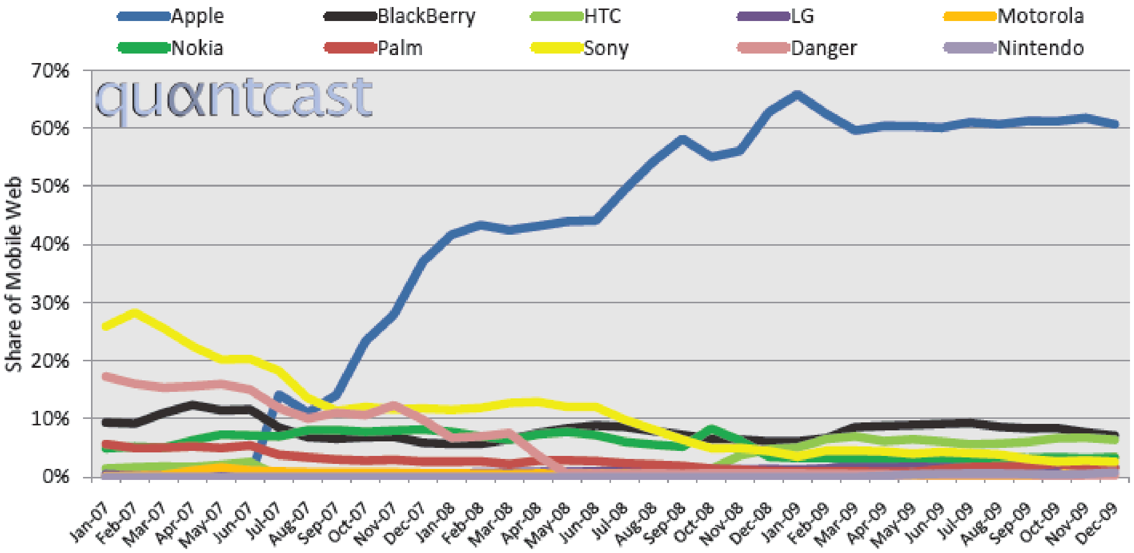

Figure 9: Time Development of Worldwide Market Shares for Mobile Web Browsers

Fig. 9 shows the development of these market shares between 2007 and 2009, taken from a report by Quantcast [7]. One can see that the distribution changes quite drastically with the introduction of Apple's iPhone: From what looks initially like a power law or Poisson distribution, the market arrives in a state in which one brand is very dominant. Unfortunately we were unable to find the raw data that was the basis of this plot, so it is not possible to establish what mathematical dependence is really present. Simply in terms of the largest market share, however, we might establish a connection to our order parameter: By the end of 2009, the largest competitor had a significant portion of the market share (ordered phase, small Φ) while in mid-2007, it was much more evenly distributed (unordered phase, large Φ). If this analogy can be made, we could interpret Φ as a kind of "brand disloyalty indicator" or even a quantity characterizing social influence: In the state of the market in 2009 (low Φ), our corresponding dynamics in the model would assert that individuals are much more likely to adopt their friends' opinion (choice of phone brand), leading to a dominant cluster. Once again, though, this backward conclusion can only be stated as an assumption with the data available to us.

As a side note, it should be said that comparing different states of the market with different phases in our model can be problematic as such since our model only deals with *equilibrium consensus states*, but this real world example is of course a continuously evolving and thus non-equilibrium system! By applying our model to it, we are essentially assuming some kind of equivalent to an adiabatic principle in thermodynamics: This would imply that at each time that we are measuring market shares, the underlying Φ has changed, causing the system to end up in a different convergent state. This assumption is of course a crude one that is hard to defend for real systems.

7 Summary and Outlook

7.1 Summary

The initial motivation for this project was to investigate the interdependent evolution of opinions and networks by disentangling the combined process of one's social network shaping one's opinion and vice-versa. The central research question was whether our social surroundings determine which opinion we hold or whether our opinion determines whom we make part of our network and how these extremes influence the formation of networks and opinion structures.

In this work we were able to show that this general combined process can be modelled by setting a single parameter, the reconnection probability that gives the likelihood of an opinion holder to reconnect to a like-minded neighborhood versus adapting its opinion to its neighborhood. In other words this parameter determines the ratio with which each of two basic processes are realized. In different contexts the reconnection probability can be interpreted as a different real-life quantity. For example, an intolerance indicator influences the distribution of religious views in a country or region. In the context of consumer behavior a brand disloyalty indicator determines market shares of different brands of the same type of product. As the concept of a non-spectral opinions in social networks is very broad, many more examples are possible.

This work was in major parts an adaption of Holme's and Newman's work ([5]). However, two essential parts were implemented differently. Firstly, while they were included in [5] for reasons of simplicity, self- and multi-edges were forbidden in our model. Secondly, the explicitly does not alter local consensus states or "happy nodes". With this extended setup the most important results could be reproduced.

First of all and most remarkably, we found that a continuous phase transition occurs when gradually altering the reconnection probability. It is certainly non-trivial to have a giant cluster of equal opinion that at some point, around the critical value, suddenly is on equal standing with all the other opinions in the network. This trend was found to be increasingly pronounced for greater network sizes. The critical value for $k_{avg} = 4$ and $\gamma = 10$, which can be seen as measures of connection density and opinion diversity, respectively, was found to be $\Phi_c = 0.32 \pm 0.05$. As a consequence of this critical region around this value, a network that is in a state of one dominating opinion, be it religion or brand, can in theory quite dramatically change to state in which this dominating opinion vanishes to a multitude of more or less equal opinions. What "dramatically" means in reality, depends on the context and can by no means be inferred from our model, which does not carry any information of (real) time or space.

Further, the analogies to statistical physics that could be drawn both qualitatively and quantitatively, are on the one hand hard to miss but should on the other hand be taken with caution. While it was possible to reproduce typical behaviour in thermodynamic systems, some predictions of the theory of critical phenomena could not be reproduced. The most notable analogy is the fact that for different network sizes, the phase transition curves of $S(\varphi)$ all intersect in the critical point. Furthermore, we could find a variable, the convergence time for reaching consensus, that diverges at this very critical point. Yet it is difficult to establish an analogy to divergent response functions in physics like the magnetic susceptibility.

Critical behavior in various systems is characterized and categorized by critical exponents which were attempted to be determined for the system in this work. It turned out that by strictly following the method suggested in [5], a wide range of values of Φ can be tuned to become the critical value. The tuning parameters a and

b can be used to calculate a critical exponent, which in our exemplary calculation turned out to be $\alpha = 0.5$. In our case any of this tuning was not even necessary for obtaining a neat critical point, i.e. the trivial case of $a = b = 0$ is just as well justifiable by this method. However, the puzzling consequence is that this would correspond to a critical value of zero, which is not in accordance to our qualitative findings and physical models.

7.2 Outlook

In future works there are vast possibilities to further investigate and extend the model in order to enable more analogies to statistical physics and its proven tools as well as to better account for reality.

The values of k and γ have been kept constant throughout this work. It would be of interest to investigate how a changing connectivity density and opinion diversity affects the critical reconnection probability and general behavior of the network. The work of [5] suggests that Φ_c increases with k but is constant with γ. But do those values need to be time-invariant at all? Networks could become more dense and diverse over time.

Speaking in physics terms, further state variables could enhance the applicability of physical models to this system. For example, a phase diagram could be established. It would be most expedient for this purpose to add to our ”temperature” of the system, i.e. the reconnection probability, a ”magnetic field”. A most intuitive way to realize this is to introduce a finite number of nodes that hold a constant and equal opinion, penetrating the system like an externally applied field. In the work of [2] this was realized by including ”informed agents” (nodes with constant opinion) in the network. However, this work was missing a temperature equivalent.

Another potentially far-reaching extension that can be added to the model is the spectrality of opinions. In this work only non-spectral opinions were considered. Yet there are many types of opinions that are much better described by *spectral* opinions, i.e. opinion that are on a spectrum or continuum between extreme views. While religious views could on a larger time scale be argued to be of non-spectral type, it is certainly more intuitive to consider them as spectral opinions. It does make sense that certain religions are more similar to certain others. In order to account for this in our model, connecting or adapting to nodes of *similar* as opposed to equal opinion has to be allowed and made dependent on the degree of similarity of opinions. In this sense it is also reasonable to initially distribute opinions in a way that favors opinions of central over extreme parts of the spectrum.

References

[1] R. Albert, A.-L. Barabasi. *Statistical mechanics of complex networks.* Rev. Mod. Phys, 74:4798, 2002.

[2] S. Brugger, C. Schwirzer. *Opinion formation by "employed agents" in social networks.* Project Report: Modelling and Simulating Social Systems with MATLAB, ETH Zurich, 2011.

[3] C. Castellano, D. Vilone, and A. Vespignani. *Incomplete ordering of the voter model on small-world networks.* Europhys. Lett., 63:153158, 2003.

[4] P. Erdős, A. Rényi. *On the evolution of random graphs.* Publications of the Mathematical Institute of the Hungarian Academy of Sciences 5: 1761, 1960.

[5] P. Holme, M. E. J. Newman. *Nonequilibrium phase transition in the coevolution of networks and opinions.* Physical Review E, vol. 74, Issue 5, id. 056108, 11/2006.

[6] W. Nolting. *Grundkurs Theoretische Physik 6: Statistische Physik.* Springer Berlin Heidelberg. 5. Aufl, 2004.

[7] Quantcast, *Quantcast Review of 2009 Mobile Web Trends.* "https://www.quantcast.com/inside-quantcast/2010/01/quantcast-review-of-2009-mobile-web-trends-mobile-web-share-up-110-in-north-america-and-148-globally" Accessed December 13, 2012.

[8] V. Sood, S. Redner. *Voter model on heterogeneous graphs.* Phys. Rev. Lett., 94:178701, 2005.

[9] D. H. Zanette,S. C. Manrubia. *Vertical transmission of culture and the distribution of family names.* Physica A, 295:18, 2001.

[10] A. Zheludev. *Advanced solid state physics.* Lecture, ETH Zurich, Fall 2012.

List of Figures

8 Appendix

8.1 Source Code

8.1.1 Main Script

```
%%% This is the main script from which all functions are called.
%Functions that are called are ”uniform_random_graph(N,M)”,
%”simulation2(AA_sp, gg, N, phi)” and ”clustersize_distr(A_sp)”.
%This script saves a matlab-file with all
%relevant data.

clear;

%% INITIAL PARAMETERS
N_input = [400];    %Network sizes(number of nodes)
phi_input = [0.0:0.05:0.20 0.21:0.01:0.54 0.55:0.05:0.95]; %Phi range
ii = 200;           %Number of iterations used for the averaging loop
k_avg_set = 4;      %Wanted average degree
gamma = 10;         %Wanted average number of people per opinion

for NN = N_input    %Iterate over different network sizes (number of nodes)

    %Calculate other parameters based on initial parameters
    N=NN;               %Set current NN-value to N for further calculations
    M = k_avg_set*N/2;  %Wanted umber of edges
    G = N/gamma;        %Number of opinions


    %% GENERATING OPINION GRAPH

    %Generate bare graph first
    A_sp = uniform_random_graph(N,M);   %Generate random graph as sparse adjacency matrix
    k_avg = sum(sum(A_sp))/N;           %Compute actual avg degree of generated graph to compare with set/wanted degree
    AA_sp=A_sp;                         %Store A_sp into AA_sp that will be unaffected by the averaging-loop, retaining the initial matrix

    %Generate opinion vector
    g = randsample(G,N,true);           %Nx1 vector filled with rand ints from 1 to G with replacement=true
    gg=g;                               %Store g into gg that will be unaffected by the averaging-loop


%% AVERAGING LOOP FOR GRAPH SIMULATION
%Let's the simulation run several times starting from the SAME random
```

```
%network given by AA_sp and gg.

    for phi=phi_input  %Iterate for different reconnection probabilities

        %Write strings with relevant data for documentation
        str=['N',num2str(N),'k',num2str(k_avg_set),'gamma',num2str(gamma),'Phi',num2str(phi),'Runs',num2str(ii)]; %Shorter string for the file name.

        %Prepare simulation
        s_sum=zeros(1,N);       %Initialize sum for calculating the average of s (cluster size distribution vector)
        s=cell(ii,1);           %Create cell with one matrix stored into cell element for each run. Needed for averaging.
        op=0;                   %Initialize order parameter for averaging.
        tt=zeros(ii,1);         %Create vector that contains convergence time for each run.

        %Run simulation
        for i=1:ii
            %Some status info printed to screen while simulation is running
            status = ['Run␣', num2str(i),'␣of␣', num2str(ii), '␣at␣phi␣=␣', num2str(phi), '␣and␣N␣=␣', num2str(N)]

            [A_sp, g,t] = simulation(AA_sp, gg, N, phi); %Return connections (A_adj), opinions (g) and convergence time (t) of the consensus state, always starting from the initial AA-adj and gg!
            ss=clustersize_distr(A_sp);             %Generate s-vector from adjacency matrix. For this ONLY connections, not specific opinions, matter! ss is intermediate variable because s is occupied by s-cell.
            s{i}=horzcat(ss,zeros(1,N-length(ss))); %Concatenate zeros to s-vector to ensure that s is of dimension N. This is crucial for being able to add them up and then averaging them! Adding zeros simply means that there are no bigger clusters than the biggest one. The possibly biggest cluster is of size N.
            op=op+find(s{i}, 1, 'last' )/N;         %Calculate order parameter from each s{i}-value by storing the index, i.e. s-value, of the "1" "'last'" cell that is not zero.
            s_sum=s_sum+s{i};                       %Do the summation
            tt(i)=t;                                %Fill vector of convergence time needed for mean and standard deviation calculations
            %loglog(s{i},'o');hold all;grid on;     %Leave this command to be able to plot s for each run such that the s_avg can be compared to it.
        end

        char 'Simulation␣done!'


        %% PLOT AND SAVE RESULTS

        op=op/ii;                    %Calculate the average order parameter value
        s_avg=s_sum/ii;              %Calculate the average cluster size distribution vector
        save(['Data/',str,'.mat']);  %Save all relevant variables into a matlab file.

    end
end
```

8.1.2 Random Graph Generator

```
function [A_sp] = uniform_random_graph(N,M)
% Generates a random graph of NxN nodes and M connections

A_sp = sparse(zeros(N));    %Empty sparse NxN matrix

M_counter = 1;              %Counter going through ints

while(M_counter <= M)
    %Loop to set random connections
    %It does not stop until M connections have been set, hence the use of a
    %counter in a while loop:

    %Pick random nodes
    i = randi(N);
    j = randi(N);

    %If diagonal entry or connection already exists --> do nothing an
    %continue loop at next iteration, do not increment counter
    if i == j || A_sp(i,j) == 1
        continue

    %Else set connection from i to j (both entries)
    else
        A_sp(i,j) = 1;
        A_sp(j,i) = 1;

        %Increment counter to indicate that a connection has been set.
        M_counter = M_counter + 1;

    end
end
end
```

8.1.3 Simulation

```
function [A_sp,g,t] = simulation(A_sp, g, N, phi)
%SIMULATION Executes the simulation steps in a given network
%   Takes an adjencency matrix A_sp and an opinion vector g and also the
%   parameters N and phi. It applies iterative rules, letting opinions and
%   the network itself evolve over time. When consensus state is reached,
%   a network and opinion vector is returned.

t = 0;

abort = false; %Boolean to stop the simulation loop. Will be set to true when convergent state is reached

while(abort == false)


%% CHECKING FOR CONSENSUS STATE:

    for i = 1:N        %Loop through all nodes
    %Strategy: As soon as one pair of neighboring nodes have different
    %opinions, break the consensus-check loop and continue updating.

        unequal = false;
        %By "default" assume that there are no different opinions between
        %two neighbors. We will check all connections and set unequal to
        %true if differing opinions exist.

        for neighbor = find(A_sp(i,:));    %Loop through Vector of neighbors of current i
            %(entries that are non-zero mean that there is a neighbor)

            if g(i) ~= g(neighbor) %Compare g(i) with one of its neighbors.
                %If the opinions are DIFFERENT break both loops and don't
                %change the abort variable since at least one neighbor has a
                %different opinion and we will continue the big while loop

                unequal = true;
                %Using this variable, we denote an unequal event and break
                %INNER loop
                break

            end

        end

        if unequal == true
            %If different opinions of neighbors were detected, also break
            %OUTER loop
            break
        end


    end

    %Now we've gone through all nodes and are able to check for consensus:

    if unequal == false
    %If no different opinions of neighbors were ever detected and 'unequal'
    %is still false, set 'abort' to true such that while loop will break on
    %next iteration:
        abort = true;
    end


%% PERFORM TIME EVOLUTION

    t = t + 1;

    i = randi(N);           %Pick a random node i out of N nodes (Returning a random 1x1 matrix with entry form 1:N)
    if sum(A_sp(i,:)) ~= 0  %Calculate (double-)degree of i-th node. If not zero (i.e. if its connected to SOMEBODY), do following step:

        i_cluster = find(A_sp(i,:));                %Creates a vector of nodes that are connected to i (Find nonzero cells in i-th row and return a col-vec with their indices)
        j = i_cluster(randi(length(i_cluster)));    %Choose a random node j that is connected to i to let i and j interact

        if g(i)==g(j)
            continue        %If opinions are the same jump to next step in while-loop, i.e. choose a new i-j pair. This leaves local consensus states untouched!
        end

        %At this point we have randomly chosen a node i and one of its
        %neighbours j. Now let them interact:

        if rand<phi         %RECONNECT WITH PORBABILITY PHI

            g_idx = find(g == g(i));            %Find nodes that have the same opinion as i and store their INDEX in a col-vector
            i2 = g_idx(randi(length(g_idx)));   %Choose a random element from index vector g_idx assign the corresponding value to i2 (= choosing random node with same opinion), which is also an index of g (a node)

            %Reconnect i with i2
            if A_sp(i,i2) ~= 1 && i ~= i2    %Reconnect only if i and i2 are
```

```
                not already connected AND if i is not the same i2!
                %This way a true reconnection to another node is ensured
                %(excluding self-edges!) Otherwise, skip step and do
                % nothing such that k stays constant.

                %Store reconnection into adjacency matrix
                A_sp(i,j) = 0;      %Delete old connection
                A_sp(j,i) = 0;
                A_sp(i,i2) = 1;     %Add new connection
                A_sp(i2,i) = 1;
            end

        else      %ADJUST OPINION WITH PROBABILITY 1-PHI

            g(i) = g(j);  %Set opinion of i to opinion of neightbor j

        end

    end


end
t   %Write convergence time to mark the end of a run
end
```

8.1.4 Cluster Size Distribution

```
function [s] = clustersize_distr(A_sp)
%CLUSTERSIZE_DISTR returns an unnormalized histogram vector that contains the frequency with
%which a cluster of certain size occurs in the network.

[~,nodes] = graphconncomp(A_sp);
%This function returns the number (replaced by ~ because it is not used in
%the function) of clusters in the network defined by A_sp and nodes, which
%is a N-dimensional vector telling which cluster each node belongs to:
%Indices are nodes, values are the cluster labels that each node belongs to

n = hist(nodes,max(nodes));
%Each value of 'nodes' uniquely corresponds to one cluster in the network.
%n is the histogram that plots the count versus the cluster label. The
%clustersize_distr_backup_old.m function falsely took the opinion-value as
%opposed to the cluster-value as the indicator to which cluster it belongs
%to. This did not account for the fact that it is possible to have SEVERAL
%clusters of the SAME opinion.

s=hist(n,max(n));
%Generate histogram of cluster size distribution (see Fig. 2 in paper)
end
```

8.1.5 Order Parameter Plotter

```
%%%This is a simple a data-plotter of data generated by main_script.m

N_range = [100 200 300 400];    %Choose curves to be compared to each other
col=hsv(length(N_range));       %Creates as many colors as there are curves to be plotted
j=1;                            %Manually create integer counter for the Phi-loop
legcell=cell(1,length(N_range));%Create string cell for legend in graph

filelisting = dir(fullfile('Data', '*.mat'));   %Create list of .mat files in directory

for N2read = N_range            %Choose curves to be compared

    op_local = [];              %Initialize local op-vector
    phi_local = [];             %Initialize local phi-vector

    %This loop extracts the order parameter op from each data set and puts
    %it into op_local. It also extracts and builds phi correspondingly.

    for file_idx = 1:length(filelisting)    %Create index running up until the number of files in the directory

        load(['Data/' filelisting(file_idx).name]); %Load data of file at this index
        if N == N2read                              %If N is the N we are reading right now, add phi and op to local vectors
            phi_local = [phi_local phi];            %Concatenate current value of phi to previous ones.
            op_local = [op_local op];               %Concatenate current value of order parameter to previous ones.
        end
    end

    %%Plot curves into graph that is held open.
    a=0.4; b=0.8;                           %Tune a and b values rescaling the plot.
    plot(phi_local, (N2read^(a))*op_local, 'o','color',col(j,:)) %Plot in which the normal phi is plotted.
    plot(N2read^(b)*(phi_local-0.39)/0.39, (N2read^(a))*op_local, 'o','color',col(j,:)) %Plot in which the REDUCED phi is plotted. Here PHI_crit=0.32.
    legcell{j}=['N␣=␣',num2str(N2read)];%Create cell for data range (e.g. N_range or k_range)
    j=j+1;                                  %Manually increment loop counter. This is just for getting different colors in the plot.
    hold on;
end
```

```
hold off;

%%Add labels and legend to graph
%xlabel('Reconnection probability \Phi');
%ylabel('Order parameter S');
xlabel('Scaled reduced reconnection probability N^b \phi');
ylabel('Scaled order parameter N^a S');
legend(legcell);
grid on;
```

8.1.6 Convergence Time Plotter

```
%%This is a simple a data-plotter of data generated by main_script.m

N_range = [100 200 300 400];    %Choose curves to be compared to one another
col=hsv(length(N_range));       %Creates as many colors as there are curves to be plotted
j=1;                            %Manually create integer counter for the Phi-loop
legcell=cell(1,length(N_range));%Create string cell for legend in graph

filelisting = dir(fullfile('Data', '*.mat'));   %Create list of .mat files in directory

for N2read = N_range            %Choose curves to be compared

    tau = [];               %Initialize
    phi_local = [];

    %This loop extracts the order parameter op from each data set and puts
    %it into op_local. Also extracts phi correspondingly

    for file_idx = 1:length(filelisting)    %Create index running up until the number of files in the directory

        load(['Data/' filelisting(file_idx).name]);        %Load data of file at this index
        if N == N2read                                     %If N is the N we are reading right now, add phi and op to local vectors
            phi_local = [phi_local phi];                   %Concatenate current value of phi to previous ones.
            tau = [tau (mean(tt))/N];                      %Concatenate the value of the size of the biggest cluster for a given phi.
        end
    end

    %Plot curves into graph that is held open.
    plot(phi_local, tau, 'o','color',col(j,:))
    legcell{j}=['N = ',num2str(N2read)];    %Create cell for data range (e.g.
         N_range or k_range)
    j=j+1;                                  %Manually increment loop counter.
         This is just for getting different colors in the plot.
    hold on;
end
hold off;

%%Add labels and legend to graph
xlabel('Reconnection␣probability␣\Phi');
ylabel('Average␣relative␣convergence␣time␣\tau_a_v_g␣/␣N');
legend(legcell);
grid on;
```