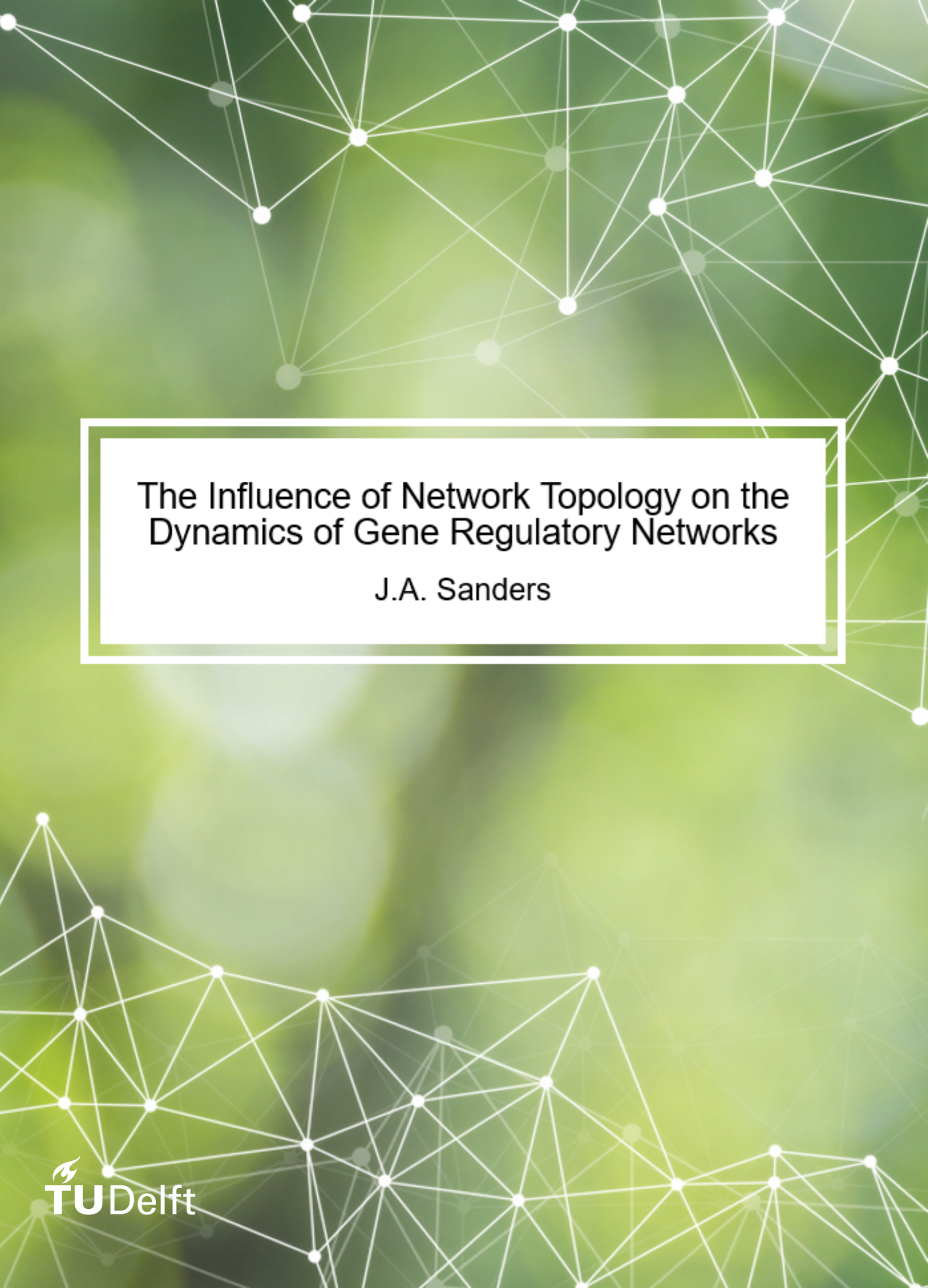


The Influence of Network Topology on the Dynamics of Gene Regulatory Networks

J.A. Sanders

TUDelft

The Influence of Network Topology on the Dynamics of Gene Regulatory Networks

Julian Sanders
4675045

Delft University of Applied Technology
BSc Applied Physics & Applied Mathematics

05 July 2020
Coordinators: Dr. Dubbeldam & Dr. Idema

Contents

Abstract

Over the last decades, studies have shown that the gene regulation of a wide range of organisms can be described with networks [Jeong H., Tombor B., Albert R., Ottval Z.N., Barabási A.L., 2000] [Jeong H., Mason S.P., Barabási A.L., Oltvai Z.N., 2001]. The interactions between the mRNA strands and proteins form links in the network, while these molecules form the nodes of the network. Numerical models for the dynamics of such a network, through solving a minimally nonlinear stochastic differential equation show nontrivial dynamics. This behavior is caused by non linearity introduced by the positivity condition, this is due to the fact that molecular concentrations cannot cannot be negative. Whether these dynamics are stable, oscillatory or chaotic seems to depend on the average connectivity of the network. There appears to be a region of networks that can transition from chaotic behavior to stable behavior. For a simple differential equation this would not be a region, but just one value. This region, referred to as the "Edge-of-Chaos" region, is therefore rather interesting.

In this report we analyse the Lyapunov spectrum in order to quantify these dynamics. This is a way to measure the stability of the solution of out differential equation. We look at how different parameters in the minimally nonlinear differential equation and the generation of the network affect these dynamics. The investigated parameters include different edge weights, different noise levels, different equilibrium values and different network types. After studying the dynamics of a single network, we study the dynamics of populations of networks using an evolutionary algorithm and a co-evolutionary algorithm.

We found that an edge of chaos region exists universally, that is for all types of networks we looked at, for all feasible noise levels, for all equilibrium values and for all network sizes. Whether or not a network lies in this region is determined by how much self interaction its nodes have compared to the strength of the interactions between the nodes. Additionally, the size of the network determines how well this "Edge-of-Chaos" plateau forms. Larger networks in this region behave more stable than smaller ones.

We also came to the conclusion that both the evolutionary and co-evolutionary algorithm, using the Kullback-Leibler divergence in accordance with a probability distribution defined by a Hamiltonian to compare the fitness of individuals, do not yield the expected results. That is, they do not show any correlation between the connectivity of a network and the development of the population over time.

1 Introduction

Network theory has become quite a broad field of study. Its applications range from solving problems in logistics to modelling the spread of an epidemic. In this report, we have applied network theory to gene regulation. The interactions between different mRNA segments and the proteins that they encode can be modeled as a network. Here the mRNA and the proteins are represented by nodes, and the reactions between them as the edges. We have looked at the relationship between the structure of a network, and the dynamics of the model. We have done this both for a single network, and for a population of networks that have interactions between them.

More specifically, we will look at at a region of network structures called the "Edge-of Chaos region", or the stability plateau. This is a range of network topologies for which the network exhibits stable, oscillatory behavior. This means that it is right on the edge of chaotic, divergent behavior and asymptotically stable behavior. This allows the system to transition between stable and chaotic dynamics, something which allows it to adapt to changes in the environment [Hanel R., Pöchacker M., Thurner S., 2010].

The main question that is answered in this report is: **"How do the stochastic dynamics of gene regulatory networks with different topologies compare, regarding a single network and an interacting population of networks?"** Sub questions that come into play when answering this question are: "What parameters play a role in the dynamics of a single gene regulatory network?", "How does the connectivity of a single network affect its stability behavior?", "How does the additive noise affect the stability behavior of a single network?", "Does the type of random network have an effect on the shape of the Lyapunov plateau?", and "Do populations of networks under selective pressure for adaptiveness converge to a Lyapunov plateau?"

The thesis is built up as follows: The chapter "Theory" consists of three parts. First, we explain network theory and it's connection to biology. Second, we give a comprehensive explanation of the model used to simulate the dynamics of one gene regulatory network, and make several predictions about the dynamics. Finally we give an explanation of the relative information model used to study the dynamics of an interacting population of networks. In the chapter "Computational Method" we discuss the numerical methods used to obtain the dynamics of a single network from the associated minimally nonlinear stochastic differential equation. We also discuss the co-evolutionary algorithm used to model the dynamics of a population of networks. In the chapter "Results & Discussion" we show the resulting dynamics for different network structures and parameters, both for the single network model and the population model. Here we also discuss these results. In the chapter "Conclusion" we answer the research question and its sub questions, and sum up our findings.

This report was written as a bachelor's thesis project for the degree of Applied Mathematics and Applied Physics at Delft University of Technology.

2 Theory

2.1 Networks

2.1.1 Connection to Biology

Networks are a ubiquitous way of describing interactions that take place in natural processes. Examples include food chains, taxonomy and social networks. These are all networks on the scale of the individual or species. In this report we are interested in networks on the molecular scale; gene regulatory networks.

Most processes in a cell are mediated by proteins. These molecules regulate transport through the various membranes, the conversion of glucose to energy, and the use of this energy for various other processes and reactions. These proteins, are in turn translated from mRNA. Molecules that are transcribed copies of sections of DNA, containing the information to make these proteins. Though this is a very complex system, with many variables, we can model this as a network of interactions between the different mRNA strands and proteins. A network with the concentrations of the mRNA molecules and proteins as nodes, and the reactions between them as edges [Nelson, 2007].

In this model we are making a few assumptions. Primarily, we are assuming that the molecules are distributed homogeneously. So we are representing the cell as a volume with no barriers, concentration gradients or vesicles. Secondly, we are assuming that there are sufficient nucleotides and amino acids to make all the mRNA strands and proteins. Additionally, we are assuming that the concentrations are low, so the increase in one concentration does not affect another through volumetric means, just through the interactions describes in the network.

2.1.2 Types of Networks

Network Theory is the study of graphs, when the graph is used to represent some relationship between objects. These relations can either be symmetric or asymmetric, depending on whether the relationship is mutual or just one way. This corresponds to undirected graphs and directed graphs of course.

Definition 1. *A **graph** $G = (V, E)$ is a pair of sets, where V is the set of vertices, and E is a set of two-sets representing the edges, connections between the vertices.*
[Aardal K. , van Iersel L., Janssen R., 2019]

Definition 2. *A **directed graph** $G = (V, E)$ is a pair of sets, where V is the set of vertices, and E is a set of pairs representing the edges, connections between the vertices.*
[Aardal K. , van Iersel L., Janssen R., 2019]

During this research we will be looking at random graphs, or networks that have the connections between their nodes generated according to particular rules. There is of course a whole spectrum of different "rules" to generate random networks, but I will mention the most relevant ones here.

Definition 3. *The $G_{n,p}$ graph, also known as a **Erdös-Rényi graph** or a binomial graph, generates a random graph according to a Bernoulli distribution. Given the parameter n, the number of nodes, and parameter p, the probability that between any two nodes there is a connection. It*

is possible to generate these in $\mathcal{O}(n+m)$ time, where $m = pn(n-1)/2$ is the expected number of nodes. This graph can be turned into a directed graph simply by randomly giving each edge an orientation. [Hagberg A., Schult D., Swart P., 2019]

Definition 4. *A **scale-free graph** is a graph whose degree distribution follows a power law for large values of k. Here, the probability that any node has k connections, $P(k)$ is given by a power-law distribution with parameter γ: $P(k) = k^{-\gamma}$. [Hagberg A., Schult D., Swart P., 2019].*

2.1.3 The Structure of Gene Regulatory Networks in Nature

It has been shown using data from the WIT database for Intermediate metabolism and bioenergetics that the gene regulatory networks of various organisms, including ones from the Archea, Prokaryote and Eukaryote domains have Scale-Free structures with the parameter $\gamma = 2.2 \pm 0.1$ [Jeong H., Tombor B., Albert R., Ottval Z.N., Barabási A.L., 2000]. The species for which this structure has been researched include, *M. tuberculosis*, *E. coli*, *H. influenzae*, *S. cerevisiae*, and *O. sativa*. In layman's terms, these species names are Tuberculosis, *E. coli*, the flu, Brewer's Yeast and Rice. Of course, there are differences in the networks. Their sizes are different, and their nodes correspond to different proteins and mRNA strands. What matters is that the structure of the networks seem to be analogous, which is quite interesting. This raises the question: why this structure?

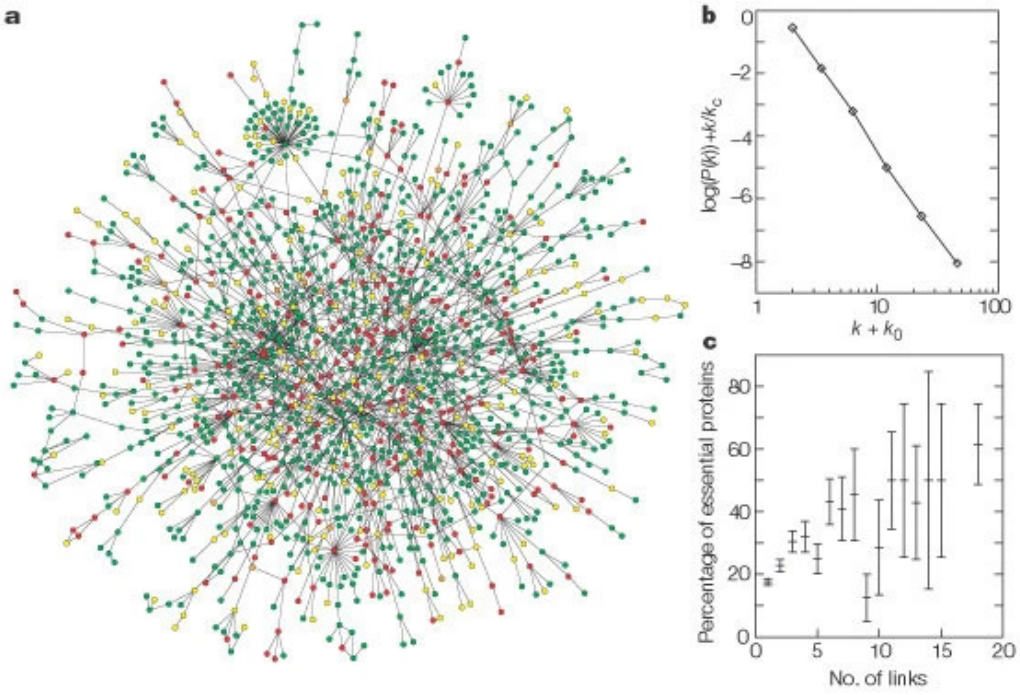


Figure 1: ***a****: The network structure of the protein-protein interactions of S. cerevisiae, Brewer's Yeast. Only the largest cluster is shown, which contains 78% of all the proteins.* ***b****: The probability distribution $P(k)$ of a node having k connections, this clearly follows the power law distribution, since the data lies on a straight line.* ***c****: The fraction of proteins with exactly k links versus their connectivity. [Jeong H., Mason S.P., Barabási A.L., Oltvai Z.N., 2001]*

2.1.4 Adjacency Matrices

The mathematical description of a graph, $G = (V, E)$ is nice enough for proofs, but we would like to model the interactions between the nodes. In this case, it is more convenient to write the structure of our graph in terms of an adjacency matrix. The indices of the matrix correspond to the nodes. If $a_{i,j}$ is nonzero, then there is an edge between node i and j of the corresponding weight. An unweighted graph will just have values of 0 and 1 as the elements of the matrix. For an undirected graph this matrix will be symmetric, since $a_{i,j} = a_{j,i}$.

In our case we will be looking at directed graphs with weights. Additionally, the nodes will have negative feedback on themselves. This can be represented as each node having an edge that connects back to the node, forming a loop. The weight of this edge is negative, $-D$. All the other edges will have weights drawn from an normal distribution with mean zero and standard deviation σ_A, a $\mathcal{N}(0, \sigma_A)$ distribution.

As an example, see figure 2. If we take the weights of the mRNA interactions to be normally distributed and denote them as $n_{i,j}$, and denote the self interaction of as D, the network gives rise to the matrix in equation (1). An example realization of this matrix with $D = 0.05$ and $n_{i,j} = \mathcal{N}(0, 0.01)$ is given in equation (2)

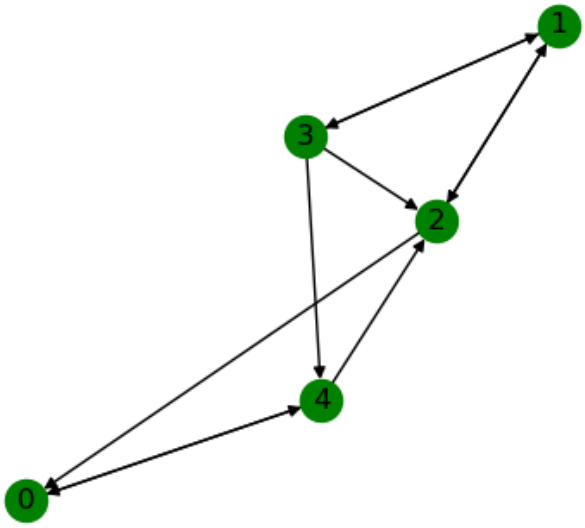


Figure 2: A directed Erdös-Rènyi graph random graph with 5 nodes and a connectivity parameter 2.

$$A = \begin{bmatrix} -D & 0 & 0 & 0 & n_{0,4} \\ 0. & -D & n_{1,2} & n_{1,3} & 0 \\ n_{2,0} & n_{2,1} & -D & 0 & 0 \\ 0 & n_{3,1} & n_{3,2} & -D & n_{3,4} \\ n_{4,0} & 0 & n_{4,2} & 0 & -D \end{bmatrix} \tag{1}$$

$$A = \begin{bmatrix} -0.05 & 0 & 0 & 0 & 0.053 \\ 0. & -0.05 & -0.003 & 0.099 & 0 \\ 0.171 & 0.041 & -0.05 & 0 & 0 \\ 0 & -0.01 & -0.086 & -0.05 & 0.151 \\ -0.069 & 0 & 0.027 & 0 & -0.05 \end{bmatrix} \tag{2}$$

2.1.5 Eigenvalues of the Adjacency Matrix

The eigenvalues of the adjacency matrix come into play when analyzing the stability behavior of a network. Everyone who has ever taken a Linear Algebra or Quantum Mechanics course knows that you find eigenvalues with the definition $A\mathbf{v} = \lambda\mathbf{v}$. However, this requires you know the values of the matrix. What we would like to do is predict the eigenvalues given that our random matrix is generated with certain parameters.

Since our random graph has an adjacency matrix which follows the distribution of a random matrix, we can use results from random matrix theory. The most relevant of which is Girko's

Circular Law.

Theorem 1: Girko's Circular Law. *Let $(A_i)_{i=1}^{\infty}$ be a sequence of independent and identically distributed random $n \times n$ matrices whose elements are distributed according to any random variable with mean 0 and variance 1. Let $\lambda_{i,1}, \lambda_{i,2}, ..., \lambda_{i,n}$ denote the eigenvalues of $\frac{1}{\sqrt{n}} A_i$. Then the empirical spectral distribution is defined as:*

$$\mu_{\frac{1}{\sqrt{n}} A_i}(X) = \frac{1}{n} \sum_{j=1}^{n} \mathbb{1}\{\lambda_{i,j} \in X\} \qquad for \qquad X \in \mathcal{B}(\mathbb{C}) \tag{3}$$

Here $\mathbb{1}\{\lambda_{i,j} \in X\}$ is an indicator function, meaning that is is 1 in when $\lambda_{i,j} \in X$ and 0 otherwise. The sequence of measures, $\mu_{\frac{1}{\sqrt{n}} A_i}$, will almost surely converge in distribution to the uniform measure on the unit disc. [Liu, 2001][Girko, 1984]

To put this into perspective, it essentially states that the eigenvalues of a random matrix generated according to the given criteria are distributed uniformly on a disk in $\mathbb{C}$ centered at the origin and with radius $\sqrt{n}$. At least, this eigenvalue distribution converges to this uniform distribution for many independent identically distributed matrices. This means that in the limit, the real components of the eigenvalues are distributed according to the Wigner semicircle distribution. For a single matrix this means that the Wigner semicircle distribution is a good estimate for the distribution of the real components of the eigenvalue, but by no means spot on. This explains the fact why some of the eigenvalues lie just outside the unit circle in figure 3.

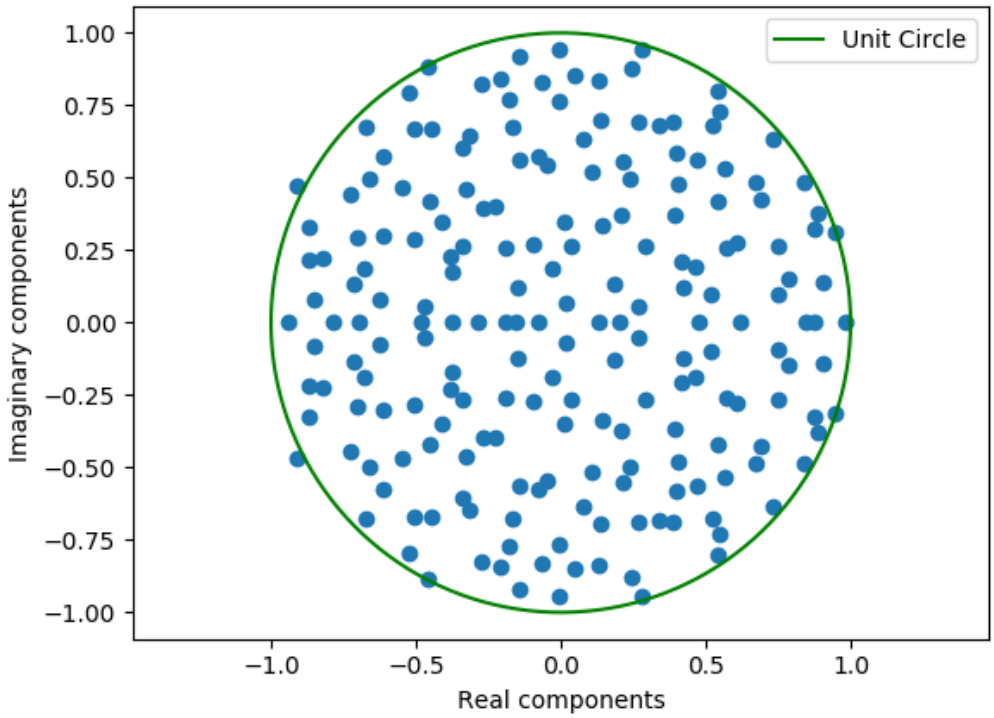


Figure 3: The eigenvalue spectrum of a random 100 by 100 matrix, whose elements are realizations of a normal distribution with mean 0 and standard deviation 0.1. These values were chosen such that the spectrum would converge to the unit circle. This particular matrix has 16 real eigenvalues, and 42 conjugate eigenvalue pairs. The expected number of real eigenvalues for a 100 by 100 random matrix is about 12.5, according to equation (5)*..*

Definition 5. *A random variable X is distributed according to the* ***Wigner Semicircle Distribution*** *on the interval $[-R, R]$ if its probability density function is equal to:*

$$f_X(x) = \frac{2}{\pi R^2}\sqrt{R^2 - x^2} \qquad for \qquad x \in [-R, R] \tag{4}$$

[Weisstein, 2020]

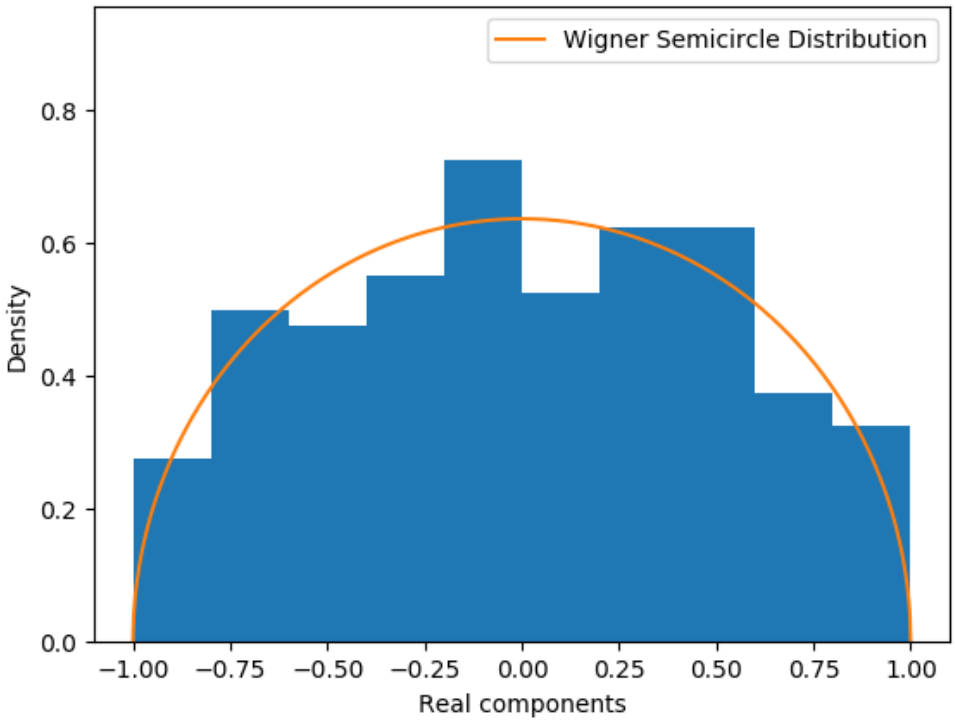


Figure 4: A histogram of the real components of the eigenvalue spectrum seen in figure 3, with the Wigner Semicircle Distribution plotted. This particular matrix has 16 real eigenvalues, and 42 conjugate eigenvalue pairs, so there are 58 realizations of the Wigner semicircle distribution.

Since our adjacency matrices are real, we must take into consideration that out eigenvalues either come in complex conjugate pairs or as strictly real singlets. So if the matrix has n eigenvalues, of which m are strictly real, then the set of real components of all the eigenvalues will contain $m + \frac{n-m}{2}$ realizations of the Wigner Semicircle Distribution. The expected number of real values of an n by n matrix can be calculated with equation (5) [Edelman A., Kostlan E., Shub M., 1994]. This function is plotted in figure 5.

$$m = \mathbb{E}(\text{number of real eigenvalues}) = \sqrt{\pi}\frac{\Gamma(\frac{n+1}{2})}{\Gamma(\frac{n}{2})} \tag{5}$$

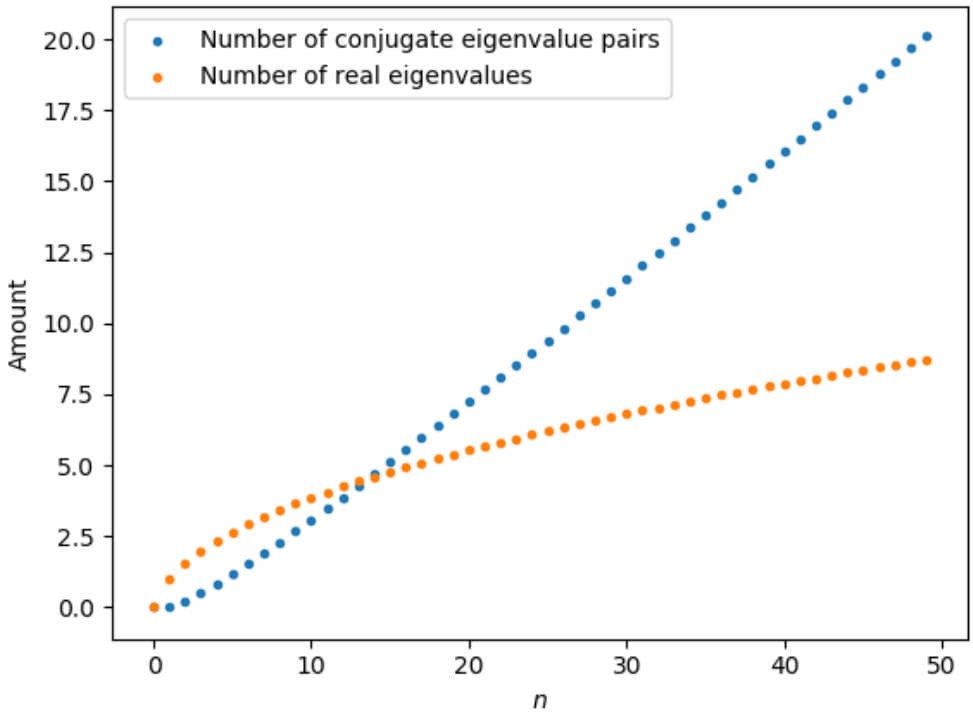


Figure 5: The number of strictly real eigenvalues and conjugate eigenvalue pairs of an n by n random matrix, see also equation 5.

In the section "Lyapunov Exponents" we see that the largest real component of the eigenvalue spectrum is important in predicting the dynamics of the network. Therefore, we would like be able to estimate what this value will be for different networks. Since the real components are $m + \frac{n-m}{2}$ values drawn from the Wigner semicircle distribution, the largest real component is the largest value.

Assume that $X_1, X_2, ..., X_N$ are $m + \frac{n-m}{2}$ random variables distributed according to the Wigner Semicircle Distribution. These represent the different real components of the eigenvalue spectrum of a random n by n matrix. We can order these in increasing order as $X_{(1)}, X_{(2)}, ..., X_{(N)}$. So $X_{(1)}$ is the smallest value, and $X_{(N)}$ is the largest value. We are interested in the distribution of the largest value. We can obtain this distribution from the Wigner Semicircle distribution, see equation (6) [Bijma, 1997].

$$f_{X_{(N)}}(x) = NF(x)^{N-1}f(x) \tag{6}$$

Here $F(x)$ is the distribution function of the Wigner semicircle distribution, and $f(x)$ the probability density function. We are interested in where this largest real component will end up, so we would like to know the expectation of $X_{(N)}$. See equation (7)

$$\mathbb{E}[X_{(N)}] = \int x f_{X_{(N)}}(x) dx \tag{7}$$

$$= \frac{2N}{\pi^N} R \int_{-1}^{1} u(u\sqrt{1-u^2} + \arcsin(u))^{N-1} \sqrt{1-u^2} du$$

$$= C(N)R \tag{8}$$

Computing this integral is not possible using analytical methods that are within the scope of a bachelor in mathematics. Luckily, in this day and age we have access to computers. Therefore, we do not have to compute this integral analytically, but we can compute it numerically. After applying the midpoint method to integrate this equation for various different N we got the following plot, see figure 6. The issue is, this numeric integral cannot be calculated for $N \geq 621$. This is due to the fact that the integrand assumes very large values, and we are using double precision floats as datatypes in Python. But, we can extrapolate what this integral will be based on data for smaller N. In figure 7, the log-log scale plot of the relationship between $1 - C_N$ and N shows that closely follows a power law, equation (9). The data points are nearly on a straight line, with a slight offset for smaller N. The orange line is a least squares fit through all the data points. This line is not representative for larger N, as it lies above the data. This is a shame, since those are the coefficient for which the terms in the integral become too large for the computer. Since we want a formula that estimates $C(N)$ for large N, we need a fit that is more representative for larger N. This is where the second fit comes in, the orange line. This is a fit for $N \geq 300$. This results in the coefficients $a = 1.278, b = 0.668$.

$$1 - C(N) = aN^b \tag{9}$$

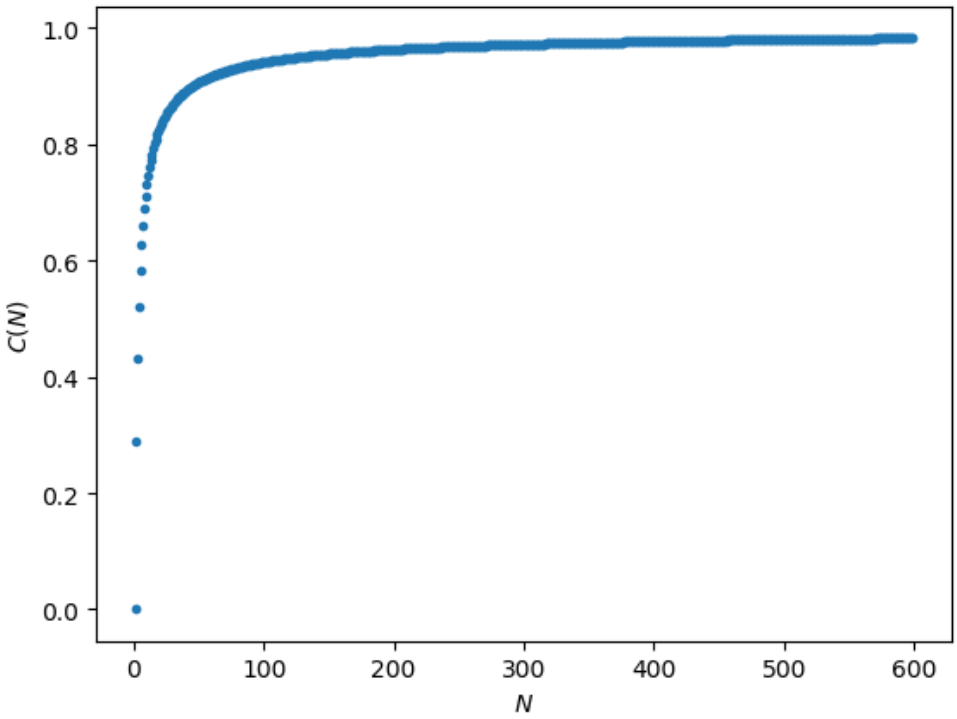


Figure 6: The coefficient, $C(N)$, for finding the largest real component in the eigenvalue spectrum of an matrix with N different real eigenvalue components.

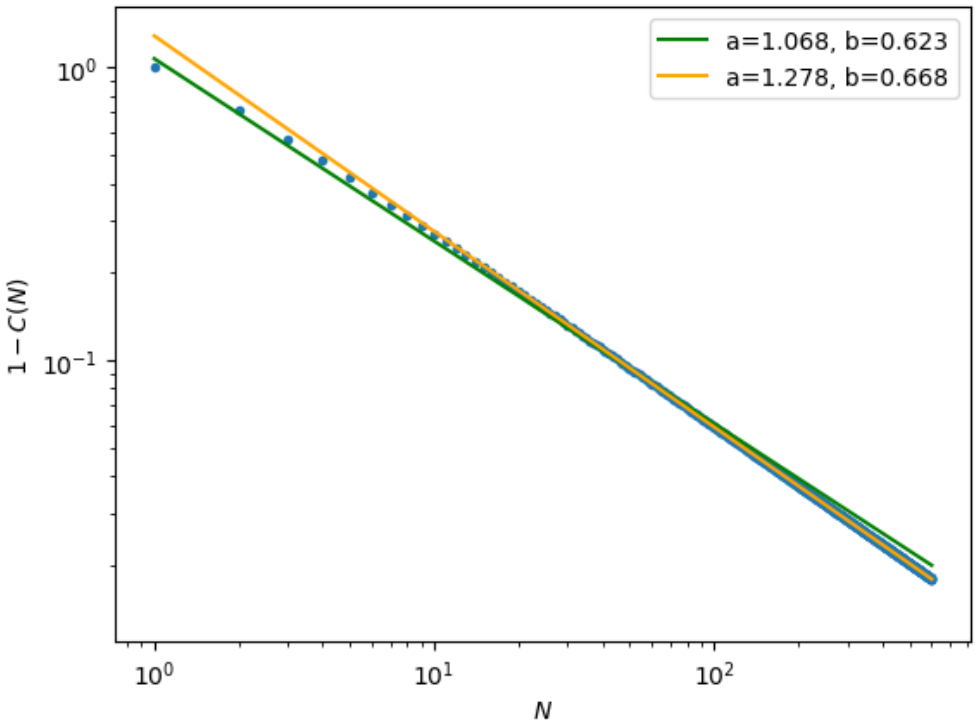


Figure 7: The coefficient, $C(N)$, for finding the largest real component in the eigenvalue spectrum of an matrix with N different real eigenvalue components. This is plotted as $1 - C(N)$ on a double logarithmic scale, to make the power law relationship clear. The orange and green lines are fit's through all the data points and just the second half of the data points, respectively.

In conclusion, using Girko's circle theorem we obtain the following result for the largest real component of the eigenvalues of a random matrix. Given an n by n random matrix with terms that are distributed according to a random variable with mean zero and variance σ^2, this matrix is expected to have m strictly real eigenvalues, see equation (5). The other $n-m$ eigenvalues will be conjugate pairs. The expectation value of the largest real component of all the eigenvalues is expected to be $C(N)\sigma\sqrt{n}$. Here $N = m + \frac{n-m}{2}$ and $C(N)$ is given by equation (7), which can be numerically integrated for $N < 621$, or approximated with a power law for $N \geq 621$, see equation 9.

2.2 The Dynamics of One Network

2.2.1 The Minimally Nonlinear Stochastic Differential Equation

It has been shown that gene regulatory networks have a scale-free structure. See section 2.1.3. We would like to model the mRNA and protein interactions, in order to analyze their development over time.

Gene expression can be modelled according to the network of interaction. The nodes in this network correspond to segments of DNA. How much these gene segments are expressed

is determined by the concentration of their corresponding mRNA molecule in the cell. These mRNA pieces are translated into proteins, which cause the function of this gene to be executed. All mRNA segments have negative feedback on their own expression. If the concentration of one type of mRNA strand becomes too high, the cell will transcribe the corresponding segment of DNA less. Additionally, some of these mRNA molecules influence the production of other mRNA segments. Some proteins will also be needed to transcribe other segments of DNA. Therefore, through the protein as a mediator the concentration of one mRNA strand can have an effect one how much another mRNA strand is produced. These reactions are represented by the edges in the network. We are effectively hiding the nodes that represent the protein concentrations in our network. These interactions are included in the interactions between the mRNA nodes.

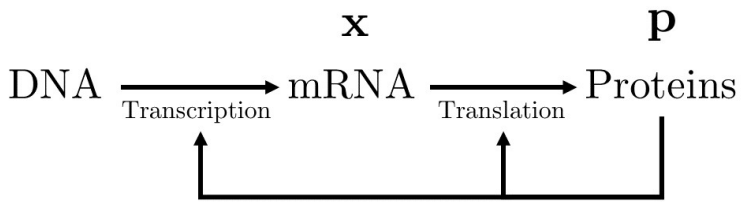


Figure 8: The central dogma of molecular biology. Usually the arrows from the proteins to transcription and translation are not included. Here this interaction is relevant. There are proteins needed to execute transcription and translation.

This network of interactions can be summarized into a differential equation. A minimally nonlinear stochastic differential equation to be specific. This can be shown as follows:

Assume that we have a vector $\mathbf{y} = (\mathbf{x}, \mathbf{p})$. Here $\mathbf{x}$ is a vector representing the mRNA concentrations, and $\mathbf{p}$ is a vector containing the protein concentrations. Since each mRNA concentration can influence others, this system behaves according to the equation:

$$\frac{d}{dt}\mathrm{x}_i = F_i(\mathbf{x}, \mathbf{p}) \tag{10}$$

In this equation, $\mathbf{F}$ is a function describing the interaction between the mRNA and the proteins. Assuming that there exists an equilibrium $\mathbf{x}_0$, we can linearize our equation around this equilibrium. This yields a differential equation which can be used to approximate the solution for small deviations from the equilibrium. The interations between the proteins and the mRNA are represented with coefficients $c_{\alpha,j}$:

$$\delta p_\alpha = \sum_j c_{\alpha,j}\delta\mathrm{x}_j \tag{11}$$

From a first degree Taylor expansion of $\mathbf{F}$, we obtain the following linear relation between our protein concentrations and mRNA concentrations:

$$\frac{d}{dt}\delta\mathrm{x}_i = \sum_j \Big[\frac{\partial F_i}{\partial \mathrm{x}_j}\Big|_{x_0} + \frac{\partial F_i}{\partial p_\alpha}\Big|_{p_0} c_{\alpha,j}\Big]\delta\mathrm{x}_i \tag{12}$$

Since this differential equation is linear and homogeneous, it can be written as an ordinary differential equation with a matrix $\bar{A}$:

$$\frac{d}{dt}\delta\mathrm{x}_i = \sum_j \bar{A}_{i,j}\delta\mathrm{x}_j \tag{13}$$

However, this matrix can fluctuate over time. This is caused by the fact that since we are dealing with concentrations, our quantities are intrinsically stochastic. Therefore the strength of the interactions is subject to fluctuations. These fluctuations can be included in the equation by writing the matrix as: $\bar{A}_{i,j}(t) = A_{i,j} + \eta_{i,j}(t)$, with $\eta_{i,j}$ white noise. By separating the additive terms from the multiplicative terms, $\xi_{i,j}$ and μ_i respectively,the equation then becomes equation (14). Since we are discussing concentrations, we are talking about nonnegative quantities. This can be represented by adding the following condition (15) our differential equation:

$$\frac{d}{dt}\delta \mathrm{x}_i = \sum_j A_{i,j}\delta \mathrm{x}_j + \sum_j \xi_{i,j}\delta \mathrm{x}_j + \mu_i \tag{14}$$

$$\mathrm{x}_i \geq 0 \tag{15}$$

2.2.2 Lyapunov Exponents

We now have an equation that describes the dynamics of the network. The solution represents the development of the mRNA concentrations over time. What we would like to do now is to have a way to quantify this behavior. That is, quantify how much the concentrations diverge from the equilibrium or converge to the equilibrium.

An ideal quantifier for this is the Lyapunov exponent, λ. This characterizes the rate of separation of two infinitesimally close trajectories. We know that this separation will be exponential for an ordinary differential equation, so one without noise and the nonzero condition (16).

$$\frac{d}{dt}\delta \mathrm{x}_i = \sum_j A_{i,j}\delta \mathrm{x}_j \tag{16}$$

Differential equation (16) will have a solution of the form:

$$\delta \mathbf{x}(t) = C\exp(At) = c_1\mathbf{v}_1 e^{\lambda_1 t} + ... + c_n\mathbf{v}_n e^{\lambda_n t} \tag{17}$$

Where λ_i and $\mathbf{v}_i$ are the eigenvalues and eigenvectors of matrix A, and c_i are constants. In the limit for $t \to \infty$ the eigenvalue with the largest real component dominates the behavior of the solution. Therefore it can be approximated as (18), where $\lambda = \max\{\mathfrak{Re}(\lambda_i) : i = 1, \cdots, n\}$.

$$||\delta \mathbf{x}(t)|| \approx Ce^{\lambda t}||\delta \mathbf{x}^1(0)|| \tag{18}$$

Therefore, we can find this Lyapunov exponent with equation (19). Here $\mathbf{x^1}(t)$ and $\mathbf{x^2}(t)$ are two solutions with two different initial conditions with a small separation between them.

$$\lambda = \lim_{t\to\infty} \frac{1}{t} \ln\Big[\frac{||\mathbf{x^1}(t) - \mathbf{x^2}(t)||}{||\mathbf{x^1}(0) - \mathbf{x^2}(0)||}\Big] \tag{19}$$

To use this equation to estimate the Lyapunov exponent using a numerical solution, we have to settle for a time that is not infinite. That would rather inconvenient to calculate, to say the least. We simply have to settle for N time steps. Of course, larger values of N will yield better estimates.

$$\lambda = \frac{1}{N\Delta t} \ln\Big[\frac{||\mathbf{x^1}_N - \mathbf{x^2}_N||}{||\mathbf{x^1}_0 - \mathbf{x^2}_0||}\Big] \tag{20}$$

But why would you go through all that trouble when it's just the largest real component of all your eigenvalues? Well, we want to find the Lyapunov exponent of solutions to equation (14), so the model with noise and with the non-negativity condition. This is no longer linear, so its dynamics can't be predicted that easily.

2.2.3 The Three Types of Behavior

When the noise and the positivity condition are included, the solution to equation (14) no longer follows equation (17). We can still classify the behavior of the solution as convergent, oscillatory and divergent. In this section we will look at the meaning of these behaviors for the cell. The three types can be seen in figure 9.

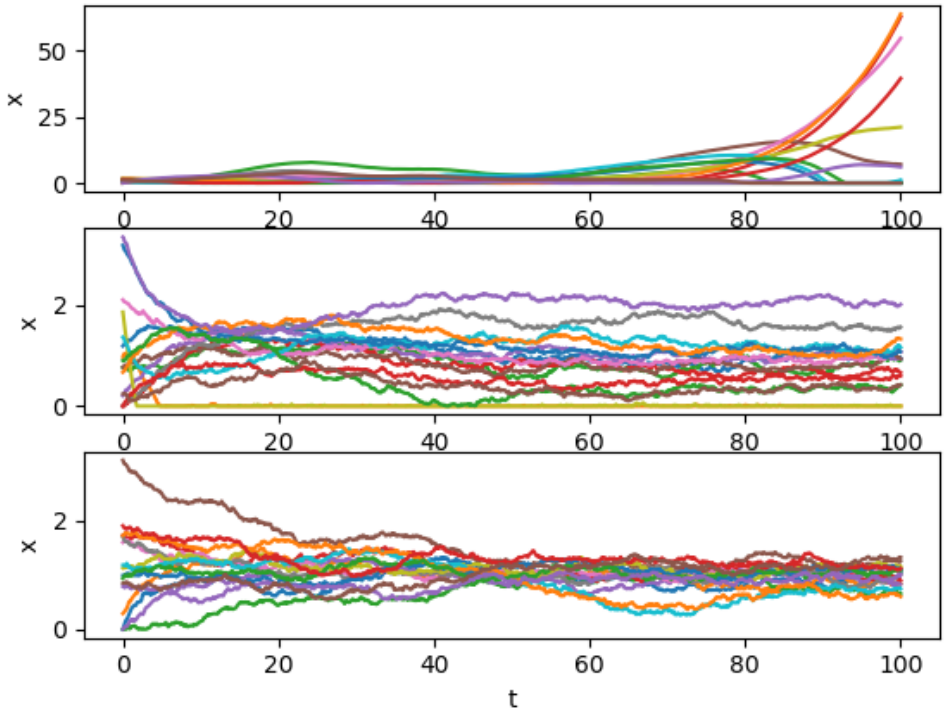


Figure 9: The three types of behavior of the mRNA concentrations, divergent, oscillatory and stable, respectively.

When the solution is convergent, this means that regardless of the initial condition, the concentrations will always converge to the equilibrium. At first one would think that this is desirable, but this does not leave room to adapt to changes in the environment. The internal concentrations will always converge to the same value.

When the solution is divergent, the concentrations will become very large. This is undesirable. If several proteins or mRNA strands are expressed too much in a cell, and their concentration grows exponentially this is clearly due to a defect. This inhibits the cell's regular functions. An example of this would be when a virus implants its genetic information into a cell. This causes the cell to start producing more and more copies of the mRNA molecule and the proteins required to make up the capsid, the shell of the virus [Nelson, 2007].

The desirable behavior is oscillatory, when the concentrations oscilate around their equilibrium value. This way, the concentrations are stable, they do not go too far off, but the cell can also adapt.

2.2.4 Lyapunov with Weights

One way to correct for the noise, is not to just use the final values of the tracks to calculate the Lyapunov exponent. It would be better to use the entire length of the solution. A possible solution could be taking an average of equation (19) at every possible point of the track. However, the behavior of the largest real component is still dominant for larger values. We would like to adjust the weights of this average to represent this, so the estimate for the Lyapunov exponent will be more accurate. An equation that uses a weighted average to calculate the Lyapunov exponent is given in equation 21. The weights need to become larger for the points further in the numerical solution. A linear increase in the weight seems a decent choice, this way the $\frac{1}{i}$ factor in equation 21 gets eliminated. An average with weights that increase linearly for larger times has been implemented in equation (23). In this equation N stands for the number of time steps that the numerical solutions $\mathbf{x}^1$ and $\mathbf{x}^2$ have been calculated for.

$$\lambda = \sum_{i=1}^{N} \rho_i \frac{1}{i\Delta t} \ln\Big(\frac{||\mathbf{x}_i^1 - \mathbf{x}_i^2||}{||\mathbf{x}_0^1 - \mathbf{x}_0^2||}\Big) \tag{21}$$

$$\rho_i = \frac{2i}{N(N+1)} \tag{22}$$

$$\lambda = \frac{2}{N(N+1)\Delta t} \ln\Big(\prod_{i=1}^{N} \frac{||\mathbf{x}_i^1 - \mathbf{x}_i^2||}{||\mathbf{x}_0^1 - \mathbf{x}_0^2||}\Big) \tag{23}$$

2.2.5 The Effect of Noise

By looking at equation (19) one would think that it is sufficient to simulate two tracks for quite some time steps, and then fill in the values of the numeric solution in the equation. There is an issue with this, however. For the differential equation without the noise and the positivity condition it works, since the infinitesimal separation will decay exponentially. Now imagine adding noise to this solution. If the behavior of your solution is asymptotically stable, your tracks will converge to the equilibrium values for x_i. But with the added noise, when they reach this equilibrium they will constantly be 'kicked' off this equilibrium trajectory. So no matter for how many time steps you run the numeric simulations, both tracks will bounce around the equilibrium with due to the noise. This messes up equation (19) since the behavior in the time limit is now no longer dominated by λ, the largest real eigenvalue, but by the noise.

In order to correct for this, we need to estimate the effect of the noise. We will do this for the case where the additive noise, μ_i, is uncorrelated white noise with mean σ and the multiplicative noise, $\xi_{i,j}$ is zero. We are not considering the non-negativity condition for simplicity.

$$\frac{d}{dt}\delta \mathrm{x}_i = \sum_j A_{i,j}\delta \mathrm{x}_j + \mu_i \tag{24}$$

The solution to this differential equation is given in equation (25).

$$\mathbf{x}(t) = \int_0^t e^{A(t-\tau)}\boldsymbol{\mu}(\tau)d\tau + \mathbf{x_0}e^{A(t)} \tag{25}$$

Since $\boldsymbol{\mu}$ is wide-sense stationary, it is easy to see that the expectation value of $\mathbf{x}$ will be:

$$\mathbb{E}[\mathbf{x}(t)] = \int_0^t e^{A(t-\tau)}\boldsymbol{\mu}(\tau)d\tau + \mathbf{x_0}e^{A(t)} \tag{26}$$

We are more interested in its variance however, as the deviation from the expectation is what makes it difficult to find the Lyapunov exponent in the first place. Filling in the definition, equation (27), we get the result (55). If we assume that our matrix A is diagonalizable, and has eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ all with multiplicity one, and corresponding normalized eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$, we can write out the exponent in terms of the eigenvalues and eigenvectors. This gives equation (29). A more comprehensive derivation is given in appendix A.1.

$$\mathrm{Var}[\mathrm{x}_i(t)] = \Big| \int_0^t e^{A(t-\tau)} \mu_i(\tau) d\tau \Big|_i^2 \tag{27}$$

$$\mathrm{Var}[\mathrm{x}_i(t)] = \sigma^2 \int_0^t \sum_j \left| [e^{A(t-\tau)}]_{i,j} \right|^2 d\tau \tag{28}$$

$$\mathrm{Var}[\mathrm{x}_i(t)] = \sigma^2 \sum_j \frac{1}{2\mathfrak{Re}(\lambda_j)} [||\mathbf{v}_j||]_i^2 (e^{2\mathfrak{Re}(\lambda_j)t} - 1) \tag{29}$$

From equation (29) we can conclude that for eigenvalues with strictly negative real components, the variance in the noise level will converge to a constant value, $\sigma^2 \sum_j \frac{1}{2\mathfrak{Re}(\lambda_j)} [||\mathbf{v}_j||]_i^2$. Does this make sense intuitively? Well, the noise level causes the track to be offset from the expected value, this happens at a rate σ. However, in the case of eigenvalues with negative real components, the decaying nature of the differential equation will cause the track to go back to the equilibrium. Therefore it makes sense that these two opposing 'forces' will cause the noise level to assume a constant value over time. For strictly positive real values, the noise will cause exponential divergence.

To show that these predictions are indeed correct, we looked at the noise levels of equation (24). We can compute the numerical solution of this stochastic differential equation several times, and compare them. We chose the initial value of $\delta\mathbf{x}$ to be 0. So the tracks start at the equilibrium value. The solution without noise would just be a constant line at the equilibrium value, but due to the noise the tracks deviate from this value. For the sake of simplicity, we chose A as a 2 by 2 matrix, see expression (30).

$$A = \begin{bmatrix} -0.1 & 0.1 \\ 0.1 & -0.5 \end{bmatrix} \tag{30}$$

As the noise level we chose $\sigma = 0.1$. The time step of this numeric simulation is $\Delta t = 0.01$, so the numeric solution is calculated for 1000 steps, until $t = 10$. The δx_1 and δx_2 tracks of 10 different numerical solutions are given in figure 10. The bold blue lines are the value we predicted for the standard deviation, so the square root of equation (29). Note that it does indeed appear as if 68.2 % of the time the tracks are within the predicted bounds, as one would expect with a standard deviation. Of course, we would like to show this with a more mathematically sound, quantifiable method. Therefore, instead of computing 10 tracks, we can compute 1000 tracks. A plot of the empirical standard deviation of these 1000 tracks with respect to the equilibrium value is given in figure 11. From this it becomes clear that our prediction is in accordance with the actual deviation due to the noise.

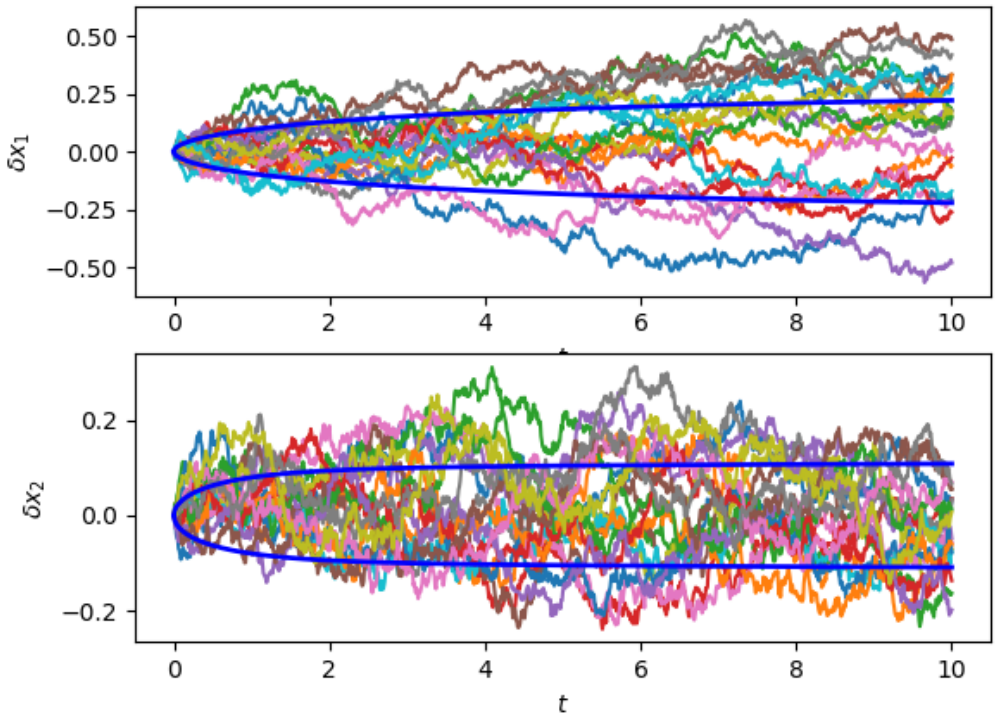


Figure 10: The $\delta\mathbf{x}_1$ and $\delta\mathbf{x}_2$ components of 10 numerical solutions of equation (24)*, the stochastic differential equation with additive noise. The noise level is $\sigma = 0.1$. The time step is $\Delta t = 0.01$. The bold blue lines represent the predicted deviation from the equilibrium due to the noise. This is the square root of the predicted variance, see equation* (29)*.*

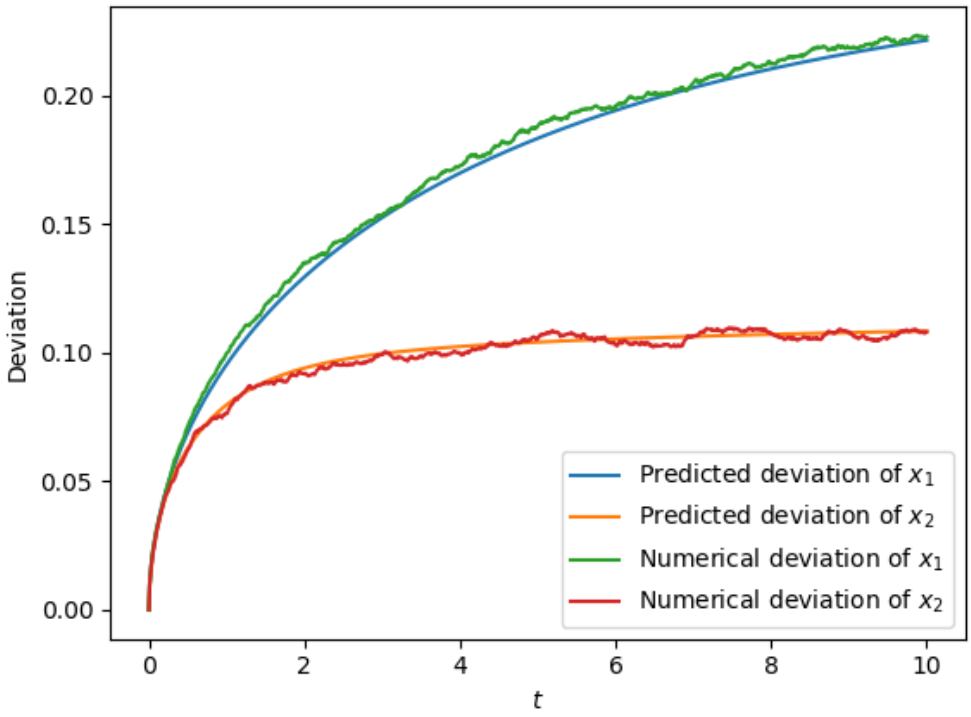


Figure 11: The empirical standard deviation of 1000 numerical solution of equation (24)*, the stochastic differential equation with additive noise. This is compared to the theoretical prediction of this standard deviation, the square root of the variance, equation* (29)*. The noise level is* $\sigma = 0.1$*. The time step is* $\Delta t = 0.01$*.*

2.2.6 The Standard Deviation Method

Before we have just considered two tracks when calculating the Lyapunov exponent, since it is defined according to their change in separation. However, one can expand this concept to multiple tracks. Say that you find l numerical solutions for a set of l initial values, $\{\mathbf{x}_0^1, \ldots, \mathbf{x}_0^l\}$. A good measure for the mutual distance of this set of solutions is their empirical standard deviation, $S_{\mathbf{x}_N}$. Using this as the distance between tracks, the formula to estimate the Lyapunov exponent becomes equation (31).

$$\lambda = \frac{1}{N\Delta t} \ln\left(\frac{S_{\mathbf{x}_N}}{S_{\mathbf{x}_0}}\right) \tag{31}$$

Here $S_{\mathbf{x}_i}$ is the standard deviation operator, applied to all the tracks in $\{\mathbf{x}_0^1, \ldots, \mathbf{x}_0^l\}$ at time i, see equation (32)

$$S_{\mathbf{x}_N} = \frac{1}{\sqrt{l-1}} \sqrt{\sum_{j=1}^{l} (\mathbf{x}_N^j - \bar{\mathbf{x}}_N)^2} \tag{32}$$

2.2.7 The Stability Plateau

We are interested in how the dynamics of one of these metabolic networks differ for different degrees of connectivity. Since the dynamics of a network can be evaluated using a Lyapunov exponent, we can look at this parameter for networks with different connectivity parameters, k. Based on the fact that the Lyapunov exponent can be seen as the largest real component of the eigenvalues, and the Girko Circle Theorem we can make a few predictions.

Let's discuss the equation without the positivity condition and noise first, equation (16). Here the Lyapunov exponent is equal to the largest real component of the eigenvalues, as there are no other dynamics at play than the interaction of the nodes. From Girko's circle theorem, we can estimate what this largest real component will be. If we apply this theorem to our case, where we have a matrix with a diagonal values $-D$ and the off-diagonal values that occur on average kn times in the matrix, and if they occur their value is normally distributed with mean 0 and standard deviation σ_A. Combined, each off-diagonal element of the matrix is essentially a random variable with mean 0 and variance $\frac{k}{n}\sigma_A^2$. This means the eigenvalues converge to the uniform distribution on a circle in $\mathbb{C}$ with center $-D$ and radius $\sigma_A\sqrt{k}$. Therefore, a decent approximation for the real part of the largest possible eigenvalue is equation (33).

$$\lambda_{max} \approx -D + \sigma_A\sqrt{k} \tag{33}$$

We can improve this estimation by considering the fact that the largest eigenvalue will not be located on the boundary given by Girko's circle theorem, but slightly below it. We must correct for this with a coefficient that depends on the number of independent real parts of the eigenvalues, N. See equation (34). This coefficient was obtained in the section '2.1.4: Eigenvalues of the Adjacency Matrix'.

$$\lambda_{max} = -D + C(N)\sigma_A\sqrt{k} \tag{34}$$

The dynamics of this differential equation, (16), get more complicated if we introduce the bound that the concentrations cannot go below zero. Now the concentrations can hit the boundary, causing the node to essentially turn off. A concentration zero will have no influence on the other concentrations. This hitting zero is not a problem for cases where the Lyapunov exponent is smaller than zero, here the solutions just go towards the equilibrium value. However, for Lyapunov exponents greater than zero the tracks will start to diverge from the equilibrium value. Therefore, on average, half the concentrations will hit this zero boundary, and half will not. Therefore half the nodes of the network will eventually have concentrations that are equal to zero. Thus half the nodes will turn off. This, in turn, leads to half the vertices losing their functionality, which means that in the matrix, half of the off diagonal terms are not participating, and would not affect the behavior of the network if they actually would have a different value. They are turned off, so essentially zero. This turned off network corresponds to a matrix with just half the off-diagonal values as the fully operational network. The off-diagonal terms are still drawn from a normal distribution with mean 0 and standard deviation σ_A. But now only $\frac{1}{2}kn$ terms are participating. Thus, this in essentially a random variable with mean 0 and variance $\frac{k}{n}\sigma_A^2$. From this we can conclude that in the case with the boundary, the Lyapunov exponent will be as in equation (35)

$$\lambda_{max} = -D + C(N)\frac{1}{2}\sigma_a\sqrt{k} \tag{35}$$

So what happens in the region of k values when the prediction for the full network, equation (34), says the Lyapunov exponent should be greater than zero, but the prediction for the network that is half off, equation (35) says it is still lower than zero? Well, in this case some tracks will

diverge, causing some to hit the boundary. This will cause nodes to turn off. However, since not all tracks will diverge, the number of nodes that are expected to turn off is not half the total number of nodes. This behavior, where some tracks hit the boundary, some diverge, and some converge will cause the Lyapunov exponent to be zero. This means that the total behavior is oscillatory, not diverging or converging. What is interesting about this is that the oscillatory behavior is possible for an entire range of network connectivities, not just for one point as one would have without the nonzero boundary condition. See figure 12

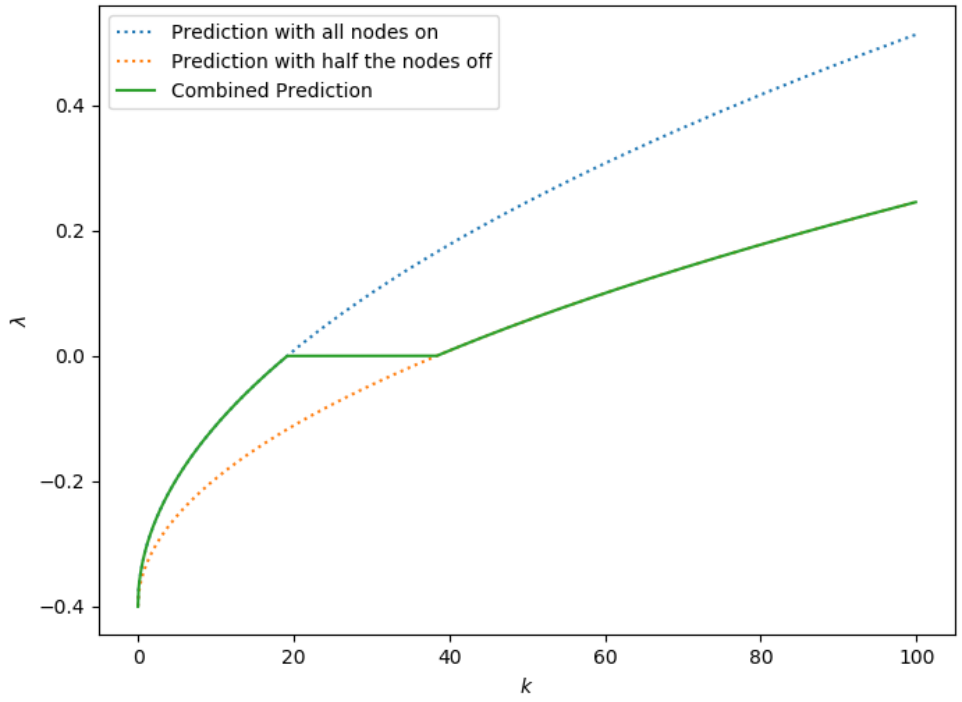


Figure 12: The predicted relationship between the connectivity of the network and the Lyapunov exponent of the corresponding stochastic differential equation. This plot is for a 100 by 100 network with a node self interaction coefficient of $D = 0.4$ *and normally distributed edge weights with standard deviation* $\sigma_A = 0.1$ *and mean* 0*.*

2.3 The Dynamics of a Population of Networks

2.3.1 Information in a Network

As we saw in the previous section, nodes in a network can either be on or off. This must suggest a way to discretize our model, so we do not have to deal with the values of the nodes, but just with their state. This way, the internal state of an entire network can be summed up with a Boolean string, or a string of ones and zero's. For a network with n nodes, there are 2^n possible internal configurations.

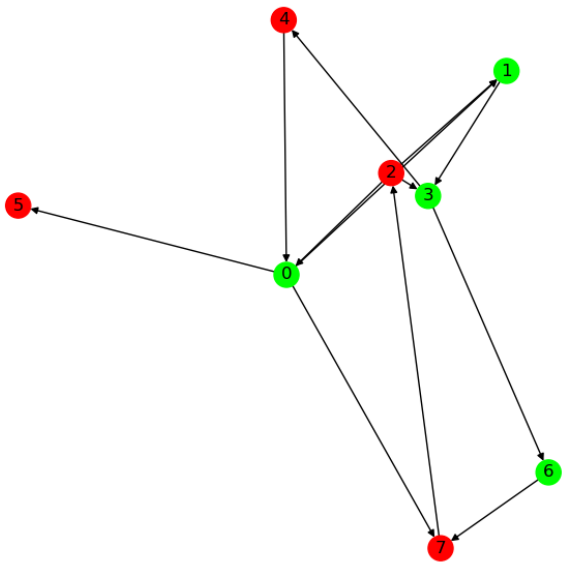


Figure 13: An 8 node network with 4 nodes turned off (red) and 4 nodes on (green). This corresponds to to the discretization $\mathbf{s} = (1, 1, 0, 1, 0, 0, 1, 0)$.

2.3.2 Adapting to an Environment

In this model we are assuming that our networks have some internal parameter, β that they are using to adapt to their environment, which has a parameter α. In order to determine what the optimal internal parameter is, we need to have a way to measure how well it represents the environment. We will be using tools from statistical physics to do this.

We can define a Hamiltonian, which gives the energy for a certain node on-off configuration, $\mathbf{s}$, given an internal parameter β. This can be done using the topology of the network as in equation (36). In this equation, A is the adjacency matrix of the network, $a_{i,j}$ are the elements of this matrix.

$$H^A_{int}(\mathbf{s}|\beta) = -\beta \frac{1}{N} \sum_{i,j=0}^{N} a_{i,j} s_i s_j \tag{36}$$

This Hamiltonian can be used to compute the probability of that the network assumes a certain node configuration given the fact that it has an internal parameter β. See equation (37).

$$P_{int}(\mathbf{s}|\beta) = \frac{\exp(-H_{int}(\mathbf{s}|\boldsymbol{\beta}))}{Z_{int}(\beta)} \tag{37}$$

Here Z_{int} is the partition function, which normalized the probability over all 2^N possible states of the network.

$$Z_{int}(\beta) = \sum_{\mathbf{s}} \exp(-H_{int}(\mathbf{s}|\beta)) \tag{38}$$

Our individual networks will change their internal state in order to represent the environment. To this end, we need to define the Hamiltonian with respect to this environmental parameter, α. This is done according to the Ising model. The Hamiltonian of the Ising model can be reduces to equation (39), where all constant terms have been dropped. The probability distribution and the corresponding partition function can be defined as before, see equation (40) and (41).

$$H^A_{ext}(\mathbf{s}|\alpha) = -\alpha \frac{1}{N} \sum_{i,j>i}^{N} s_i s_j \tag{39}$$

$$P_{ext}(\mathbf{s}|\alpha) = \frac{\exp(-H_{ext}(\mathbf{s}|\alpha))}{Z_{ext}(\alpha)} \tag{40}$$

$$Z_{ext}(\alpha) = \sum_{\mathbf{s}} \exp(-H_{ext}(\mathbf{s}|\alpha)) \tag{41}$$

In order to be able to compare these internal and external probability distributions with each other, or to compare the distribution function of one network to that of another, we need a measure. This measure must quantify how much information one probability distribution contains relative to another distribution. A suitable candidate for this is the Kullback-Leibler divergence D. This is related to the Shannon-entropy from statistical physics. The Kullback-Leibler divergence quantifies the loss of information when distribution Q is used to approximate another distribution P [Kullback S., Leibler R. A., 1951].

$$D(P(\cdot)|Q(\cdot)) = \sum_{\mathbf{s}} P(\mathbf{s}) \ln \left(\frac{P(\mathbf{s})}{Q(\mathbf{s})} \right) \tag{42}$$

We now have a tool that can be used to compare the internal parameter of a network with the environment, or with that of another network. When comparing one network, with internal parameter β to the environment, α, we need to know how well the internal representation can represent the outside conditions. Therefore, we would like to minimize the distance, or information loss between the two associated distributions. Therefore we would like to find $\text{argmin}_\beta D(\alpha|\beta)$, see equation (43).

$$D(\alpha|\beta) = \sum_{\mathbf{s}} P_{ext}(\mathbf{s}|\alpha) \ln \left(\frac{P_{ext}(\mathbf{s}|\alpha)}{P_{int}(\mathbf{s}|\beta)} \right) \tag{43}$$

We can compare the distributions of two networks in the same way, see equation (44).

$$D(\beta_i|\beta_j) = \sum_{\mathbf{s}} P_{int}(\mathbf{s}|\beta_i) \ln \left(\frac{P_{int}(\mathbf{s}|\beta_i)}{P_{int}(\mathbf{s}|\beta_j)} \right) \tag{44}$$

3 Computational Method

3.1 Dynamics of a Single Network

3.1.1 Numerical Method

In order to evaluate the dynamics of a network, we must have a way to model it's behavior. Previously we have seen that this can be described with a stochastic minimally nonlinear equation, equation (14). However, the bounds and the noise in this equation make it far from trivial to solve analytically. Luckily, we can apply numerical methods to obtain an estimate that is fairly close to the actual solution. This is where time discretization methods such as Euler-Maruyama, Modified Euler, The trapezoid method, the Heun method and the 4th order Runge-Kutta method come into play. The numerical solutions are obtained with the following method, see equation (46).

$$\mathbf{w_0} = \delta \mathbf{x}_0 \tag{45}$$

$$\mathbf{w}_{n+1} = \mathbf{M}(\mathbf{w}_n, \Delta t) + \boldsymbol{\mu}_n \sqrt{\Delta t} \tag{46}$$

Here $\mathbf{w}_n$ is the numerical solution approximating $\mathbf{x}(t_0 + n\Delta t)$. $\mathbf{M}(\mathbf{w}_n, \Delta t)$ is a function that that depends on the chosen method, see table 1. Since our Stochastic differential equation is time-independent, this function only takes the previous value for $\mathbf{w}$ and the time discretization size, Δt, as input values. $\boldsymbol{\mu}_n$ is the noise term, this is a vector of which each element is drawn from a normal distribution with mean 0 and a standard deviation equal to the desired noise level. The factor $\sqrt{\Delta t}$ works to correct the magnitude of the noise to the time discretization. The noise is a Wiener process, so it scales with the time interval. If this factor would not be included the noise would be too large when using smaller time steps to solve the equation, as the noise term is added each step.

Table 1: The numerical methods used to solve the stochastic differential equation. Their order corresponds to the order of the global error due to the use of a numeric method instead of an analytic method. The stability condition states whether the numerical solution will be stable when applied to a stiff differential equation with eigenvalue λ.

Method	$\mathbf{M}(\mathbf{w}_n, \Delta t)$	**Order**	**Stability condition**
Euler Maruyama (Forward-Euler)	$\mathbf{M}(\mathbf{w}_n, \Delta t) = \mathbf{w}_n + A\mathbf{w}_n \Delta t$	$\mathcal{O}(\Delta t)$	$\Delta t \leq \frac{-2}{\lambda}$
Heun's	$\mathbf{k} = \mathbf{w}_n + \Delta t A \mathbf{w}_n$ $\mathbf{M}(\mathbf{w}_n, \Delta t) = \mathbf{w}_n + \frac{1}{2}\Delta t(A\mathbf{w}_n + A\mathbf{k})$	$\mathcal{O}(\Delta t^2)$	$\Delta t \leq \frac{-2}{\lambda}$
Runge-Kutta	$\mathbf{k}_1 = \Delta t A \mathbf{w}_n$ $\mathbf{k}_2 = \Delta t A(\mathbf{w}_n + \mathbf{k}_1/2)$ $\mathbf{k}_3 = \Delta t A(\mathbf{w}_n + \mathbf{k}_2/2)$ $\mathbf{k}_4 = \Delta t A(\mathbf{w}_n + \mathbf{k}_3)$ $\mathbf{M}(\mathbf{w}_n, \Delta t) = \mathbf{w}_n + \frac{1}{6}(\mathbf{k_1} + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)$	$\mathcal{O}(\Delta t^4)$	$\Delta t \leq \frac{-2.8}{\lambda}$

It is possible that a differential equation that has a stable analytical solution has an unstable numerical solution for a given time step. The stability condition of the different numerical methods gives all the possible timesteps for which the numerical solution will be stable, when applied to the differential equation $\frac{d}{dt}\mathrm{x} = \lambda \mathrm{x}$. Here the parameter λ is negative, so the analytical

solution is stable. This applies only to real values of λ. The stability condition becomes a stability region when looking at complex values of λ. This condition can be used to roughly find a time step that yields a stable numerical solution, given that our eigenvalues are likely t have real components greater than $-D - \sigma_A$ according to Girko's circle theorem. Therefore, $\Delta t \leq \frac{2}{D+\sigma_A}$ gives stable numerical solutions for all methods when the analytical solution is also stable.

So we have the choice of three methods, but what method is preferable? Normally, when finding the solution of an ordinary differential equation one would use the method with the smallest oder of the error. This would be the 4-th order Runge-Kutta method. This allows one to achieve a higher accuracy with a larger timestep, thus limiting the number of function evaluations in your algorithm while still achieving an accurate solution. However, we are not solving an ODE but an SDE. The noise introduced has a more pronounced effect than the error of the numerical method. Also, we want to choose out time step small to achieve a higher time resulotion, and more steps where the noise is applied. Therefore, since the Forward-Euler method has the least function evaluations per step, this method is the one that is preferable in our case. Because we are using it to solve an SDE it is also referred to as the Euler-Maruyama method.

3.1.2 The Positivity Condition

The previously described method works to solve an stochastic differential equation. Our problem requires solving a minimally nonlinear stochastic differential equation. That means we must also include the positivity condition from equation (14). This is not too complicated. Since we are discussing concentrations, this boundary absorbs any solutions coming into it. If any component of the solution approaches it, it will stay there until either the noise or the dynamics of the equation make it go up again. It is not reflected. Implementing this in the numerical method is quite simple. every step we check if one of the components of our solution is below zero, and if it is we make it equal to zero. However, since our numerical solution $\mathbf{w}$ corresponds to $\delta\mathbf{x}$, the difference between the solution and the equilibrium value, $\mathbf{x}_0$, it is allowed to have negative values. It just is not allowed to go below $-\mathbf{x}_0$. Therefore, this is what we check for every step.

$$\begin{aligned} &\text{If } \mathrm{w}_{0,i} < -\mathrm{x}_{0,i}\text{:} \\ &\qquad \mathrm{w}_{0,i} := -\mathrm{x}_{0,i} \end{aligned}$$

3.2 Dynamics of an Interacting Population

3.2.1 The Evolutionary Algorithm

The first algorithm we used to study the dynamics of a population of networks is the evolutionary algorithm. Here the individual networks, which have different structures and different target connectivities have to adapt to an environmental cue. The algorithm works by generating a different environmental cue at each iteration, and having the networks that adapt poorly be removed from the population, while the others reproduce and mutate. The algorithm can be summarized as follows [Hidalgo J., 2014]:

Initialization:

1. Define a population of networks with different connectivities

Iteration:

1. Choose an environmental parameter, α, from a certain distribution, $\rho(\alpha)$.
2. For all the networks, minimize $D(\alpha|\beta)$ by finding an optimal β. See formula (43).
3. One of the individuals is removed with a probability proportional to it's value of $D(\alpha|\beta)$. The probability for the i-th individual to be removed is:

$$P_{kill}(i) = \frac{D(\alpha|\beta_i)}{\sum_j D(\alpha|\beta_j)} \tag{47}$$

4. Replace the killed individual with one that is the mutated offspring of one of the remaining networks, drawn uniformly. This means that the offspring's target connectivity will be $k_{\text{offspring}} = k_{\text{parent}} + \nu$. ν is normally distributed with mean 0 and variance σ_ν^2.

3.2.2 The Co-Evolutionary Algorithm

The second algorithm used to study the dynamics of a population of networks is the co-evolutionary algorithm. Here the individuals do not adapt to the environment, but to each other. We have implemented this for a population consisting of individuals with the same network structure, with different networks but the same target connectivity and for a population with different target connectivities. At each iteration a sub population is selected, of which one individual is removed and replaced with the offspring of another. They are selected according to Kullbach-Leibler divergence between them and another. The co-evolutionary algorithm goes as follows [Hidalgo J., 2014]:

Initialization:

1. Define a population of networks, each with a parameter β according to a certain distribution $\rho(\beta)$

Iteration:

1. Select a sub population of size K randomly from the total population.
2. Find one individual in this sub population to remove and replace, with probability:

$$P_{kill}(i) = \frac{\sum_{j=1}^{K} D(\beta_j|\beta_i)}{\sum_{l=1}^{K}\sum_{m=1}^{K} D(\beta_m|\beta_l)} \tag{48}$$

3. For the individual that is selected to be removed, keep the network but change its parameter to the mutated version of another in the sub population, according to $\beta_{\text{offspring}} = \beta_{\text{parent}} + \nu$. ν is normally distributed with mean 0 and variance σ_ν^2.

4 Results

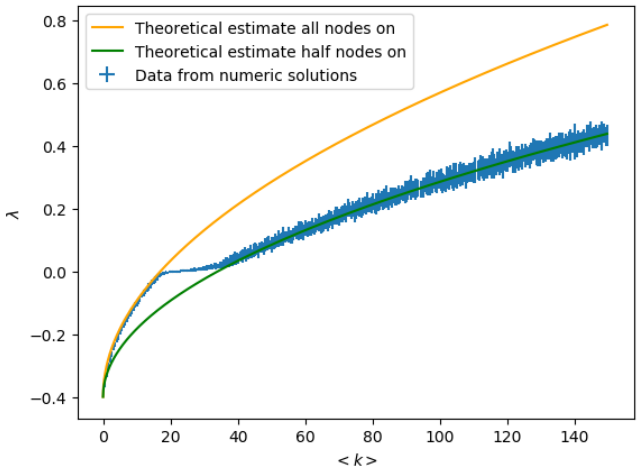
4.1 Parameter Choices

There are a lot of different parameter variations possible for out networks. To simplify our work, we chose the optimal value for some parameters, and used this value throughout the results. For the one network section we chose the parameters for the weights, D and σ_A to be 0.4 and 0.1. According to our theoretical prediction, these values make the Lyapunov plateau form at about $16 < k < 32$, but this changes depending on the size of the network. This is ideal for the network sizes we are using here. For the section about a population of networks we used $D = 0.2$ and $\sigma_A = 0.1$. This is because we are limited to smaller networks for these calculation, so we need the plateau to be between $k = 4$ and $k = 8$. For most of our calculations, we used directed Erdös-Rényi networks. This is because Erdös-Rényi networks are generated with an polynomial time algorithm, speeding the calcculations up significantly. Additionally, it is easier to target a certain connectivity with an Erdös-Rényi network than a Scale-Free network. All the numerical solutions were calculated with the Euler Maruyama method for 5000 steps with $\Delta t = 0.1$. This time step is sufficiently small to yield stable numerical solutions when the analytical solution is also stable, see table 1.

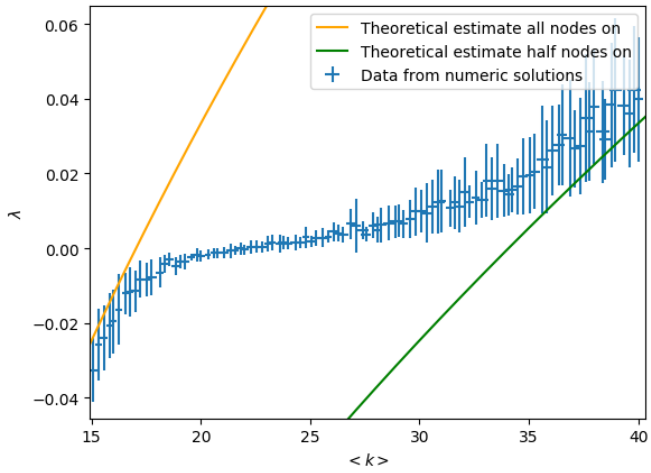
4.2 The Dynamics of One Network

4.2.1 The Lyapunov Exponent Versus the Connectivity

To investigate the relationship between the average connectivity of a node in the network, $\langle k \rangle$ and the Lyapunov exponent we executed numeric simulations for Erdös-Rényi networks with 500 nodes, generated with different target values of k. For each target value 25 different networks were generated. Each of these networks had the numerical solution of its corresponding minimally nonlinear stochastic differential equation calculated 25 times, with different initial conditions, drawn from a normal distribution with mean 1000 and standard deviation 50. No noise was included in calculating the solution. Each solution was calculated for 5000 steps with $\Delta t = 0.1$. These numerical solutions were used to calculate the Lyapunov exponent according to equation (31). For each target connectivity the average and standard deviation in the connectivities of the network, and the Lyapunov exponents of all the numerical solution of the different networks were calculated. The target connectivities range from 0 to 150, with 0.5 step sizes. In the plateau region, 15 to 40, the step size was chosen to be 0.25. Plotting this data gives figure 14. The predicted behavior is given in figure 12.



(a) The plot plot for $0 < \langle k \rangle < 150$



(b) The plot for $15 < \langle k \rangle < 40$

Figure 14: The relationship between $\langle k \rangle$*, the connectivity of a network, and* λ*, its Lyapunov exponent. This was generated using Erdös-Rényi graphs with 500 nodes. For each target connectivity 25 networks were generated, the numerical solution was calculated 25 times per network, for different initial values. The error bars in both directions represent the standard deviation in the data belonging to the same target connectivity. The curves are the theoretical estimates for this relationship for a fully active network and a half turned off network, equation* (34) *and* (35)*. See figure 30 in the appendix for more details.*

In figure 15 we plotted the number of nodes that are on at the end of the numerical solution versus the average connectivity of the network, this is for the same networks and numerical solutions as figure 14. One can see that the number of nodes that are switched on decreases approximately linearly with $\langle k \rangle$ in the range $17 < \langle k \rangle < 35$. This corroborates our analysis leading tho the theoretical predictions of how the Lyapunov exponent depends on the connectivity of the network.

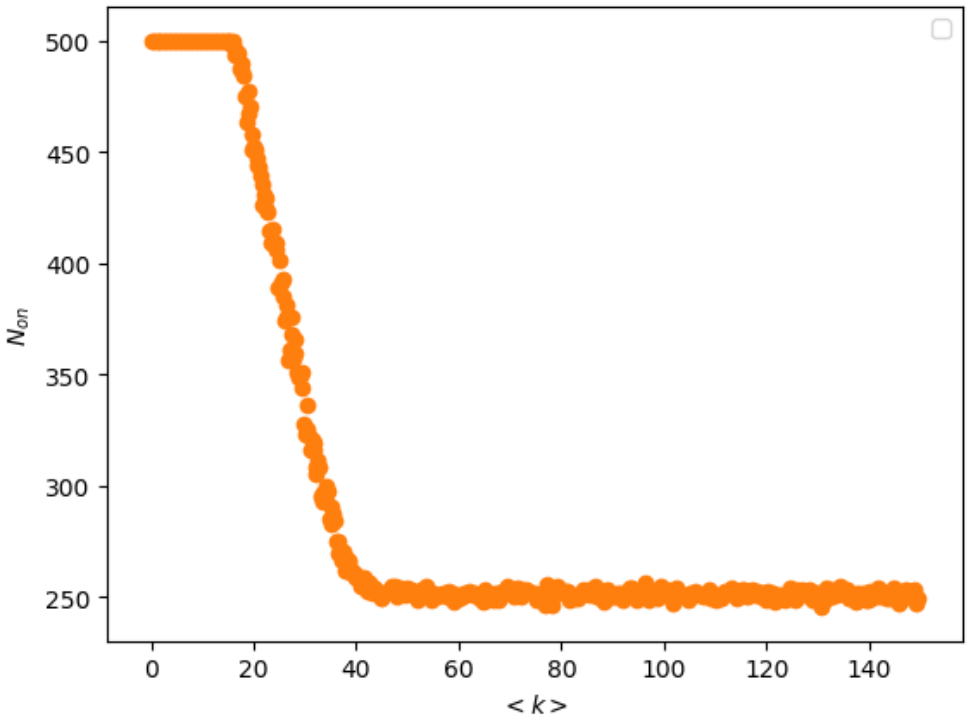


Figure 15: The amount of nodes that are on, so have nonzero concentrations, at the end of the numerical simulation. These simulations are the same as in figure 14.

4.2.2 The Dynamics of Different Types of Networks

Another interesting aspect of our problem is whether the different ways to generate networks will have an effect on the dynamics. We have compared the dynamics of Erdös-Rényi networks and Scale-Free networks. The Scale-Free networks were generated using the Holme and Kim algorithm [Holme P., Kim B. J., 2002]. The power law parameter is $\gamma = 2.2$. Here we looked at networks with 500 nodes, and with a range of target connectivities. For each target connectivity 20 networks of both types were generated, the numerical solution was calculated 10 times per network, for different initial values. The initial values were drawn from a normal distribution with mean 1000 and standard deviation 50. The mean and standard deviation of the Lyapunov exponent and average connectivity of the group of simulations belonging to the same target connectivity were calculated. This data is plotted in 16. We also looked at the relationship between the connectivity and the nodes that are on in figure 17.

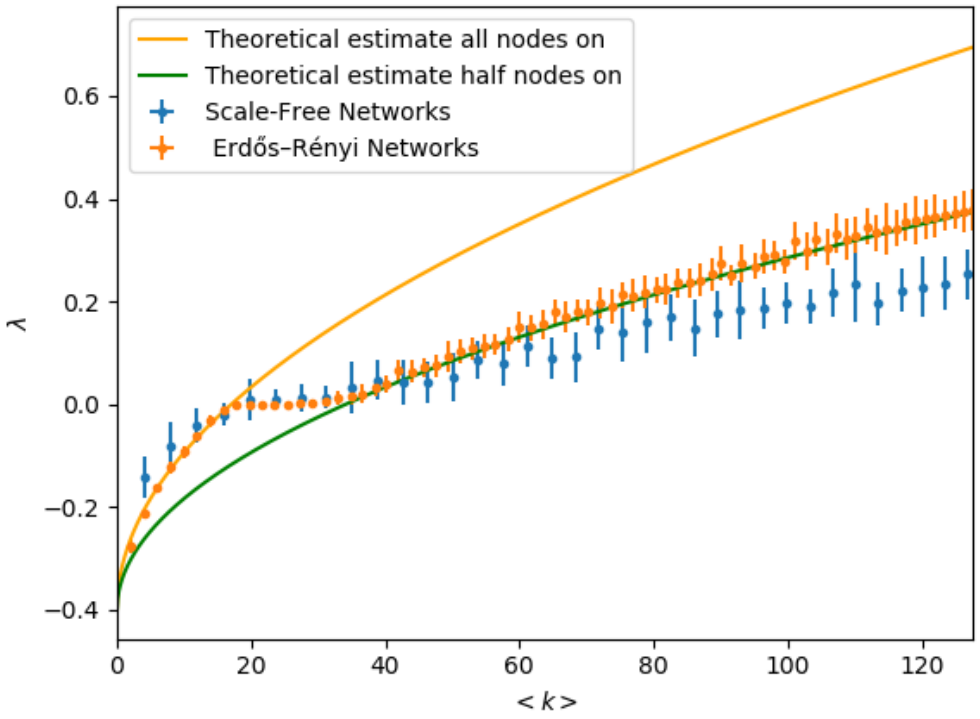


Figure 16: The relationship between $\langle k \rangle$, the connectivity of a network, and λ, its Lyapunov exponent. This was calculated for Scale-Free graphs and an Erdös-Rényi graphs, both with 500 nodes. The Scale-Free graph was generated with a parameter $\gamma = 2.2$. For each target connectivity 20 networks were generated, the numerical solution was calculated 10 times per network, for different initial values. The error bars in both directions represent the standard deviation in the data belonging to the same target connectivity. The curves are the theoretical estimates for this relationship for a fully active network and a half turned off network, equation (34) *and* (35)*. See figure 30 in the appendix for more detail.*

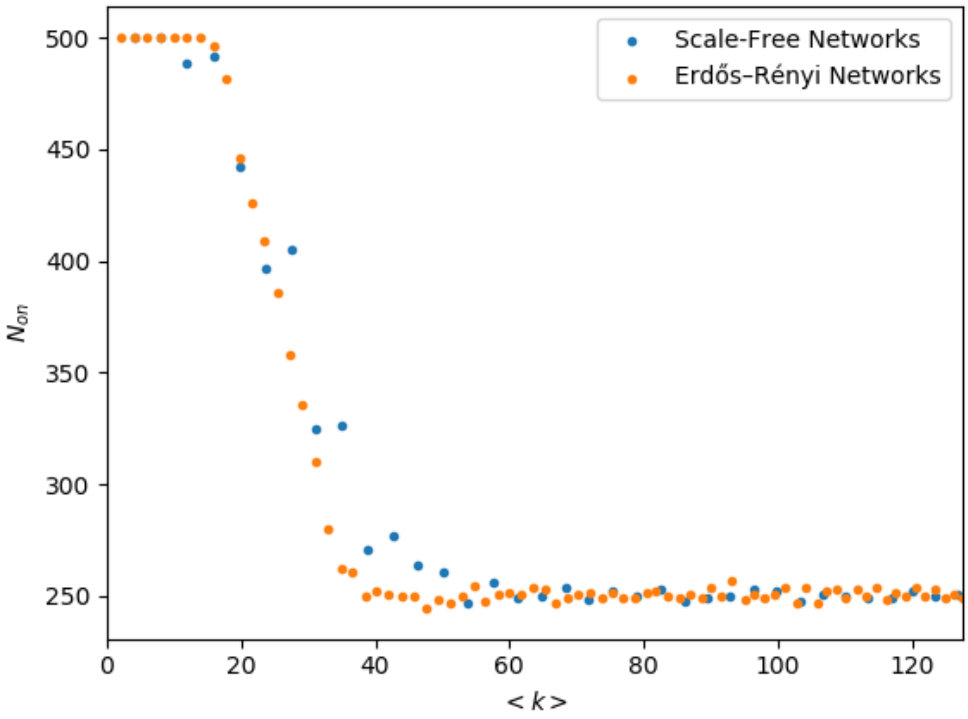
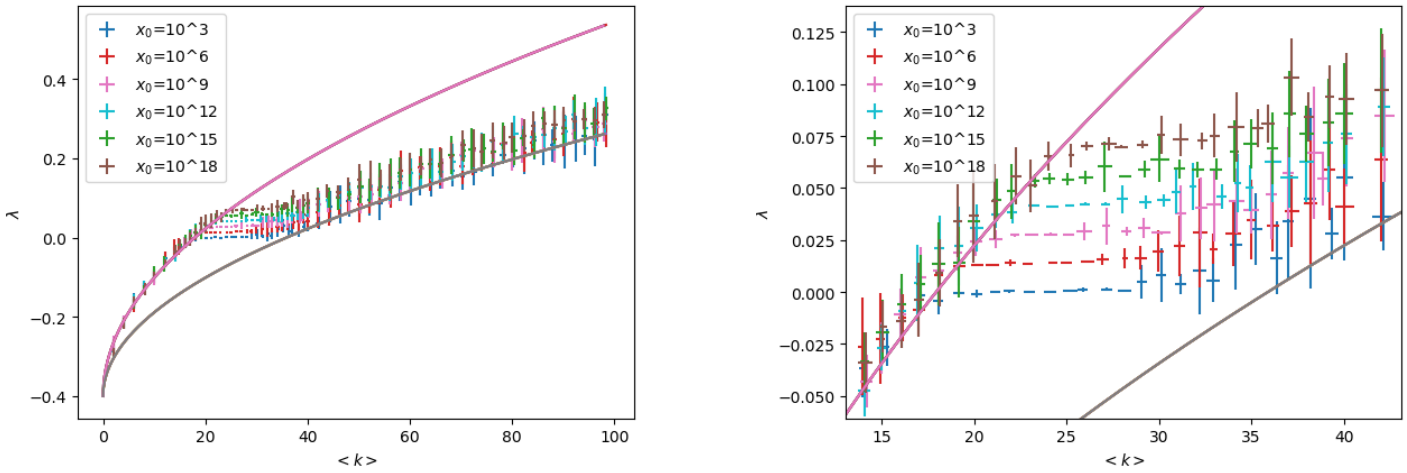


Figure 17: The number of nodes that are on at the end of the simulations, N_{on}, versus the connectivity of the network. The plot shows the result for both Erdös-Rényi networks and Scale-Free networks.

4.2.3 The Dynamics for Different Equilibrium Values

So far, we have chosen an equilibrium value that matches the mean of the initial perturbation. However, we would of course like to know the effect of different equilibrium values on the Lyapunov exponent. To this end, we generated figures analogous to figure 14, but with different values for $\mathbf{x}_0$. We calculated numerical solutions for 10 networks per target connectivity, and calculated the numerical solutions for each network 10 times. This was done for 200 node Erdös-Rényi networks with $D = 0.4$ and $\sigma_A = 0.1$. The initial perturbation for the tracks was normally distributed with mean 1000 and standard deviation 50.



(a) The plot for $0 < \langle k \rangle < 100$

(b) The plot for $14 < \langle k \rangle < 42$

Figure 18: The relationship between the Lyapunov exponent, λ*, and the average connectivity,* $\langle k \rangle$*, of an Erdös-Rényi graph with* 200 *nodes for different equilibrium values* x_0 *with respect to an initial perturbation from the equilibrium of* $\delta x_0 = 10^3$*. The parameters of the weights of the edges are* $D = 0.4$ *and* $\sigma_A = 0.1$*. Each point represents 10 networks which were generated differently, but with the same target connectivity. For each Erdös-Rényi network the numerical solution was calculated 10 times, with different initial conditions. These initial conditions are normally distributed with mean 1000 and standard deviation 50. The error bars are the standard deviations in their respective values. The curves are the theoretical estimates for this relationship for a fully active network and a half turned off network, equation* (34) *and* (35)*.*

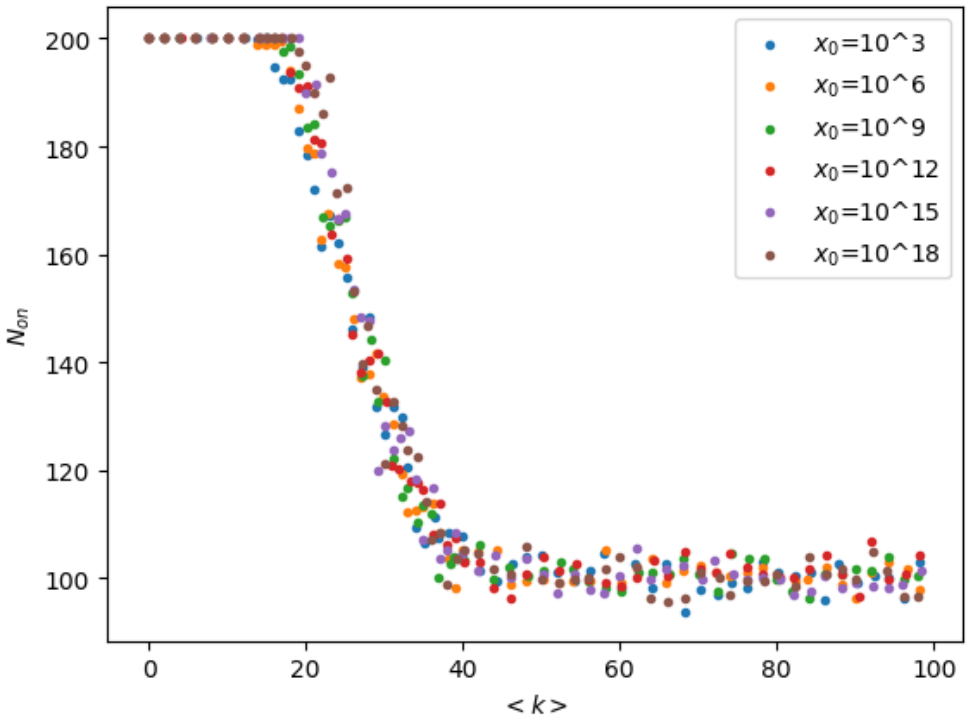


Figure 19: The relationship between the number of nodes that are on, N_{on}, so have nonzero values, and the average connectivity of the network, $\langle k \rangle$. The different plots are for different for different equilibrium values, x_0, with respect to an initial perturbation from the equilibrium of $\delta x_0 = 10^3$. This data was generated using Erdös-Rényi graphs with 200 nodes. The parameters of the weights of the edges are $D = 0.4$ and $\sigma_A = 0.1$. Each point represents 10 networks which were generated differently, but with the same target connectivity. For each network the numerical solution was calculated 10 times, with different realizations of the noise.

4.2.4 The Dynamics for Different Noise Levels

Since we are studying a stochastic differential equation, it is of course very interesting what effect the strength of of this noise has on the dynamics. To this end, we looked at the relationship between their connectivity and their Lyapunov exponent, but with different values for σ_μ. We calculated numerical solutions for 10 networks per target connectivity, and calculated the numerical solution for each network 10 times, with different initial perturbations and noise realizations. This was done for 200 node Erdös-Rényi networks with $D = 0.4$ and $\sigma_A = 0.1$. The initial perturbations for the tracks are normally distributed with mean 1000 and standard deviation 50.

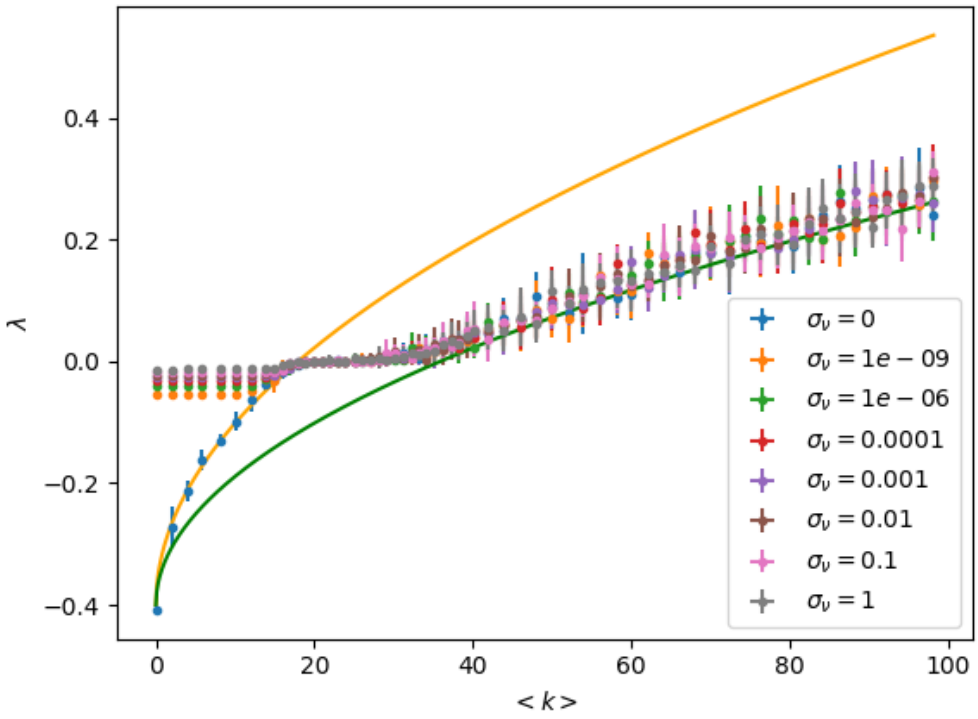


Figure 20: The relationship between the Lyapunov exponent, λ and the average connectivity of an Erdös-Rényi network, $\langle k \rangle$. This relationship is plotted for different noise levels. Each point with the corresponding error bars are the average and standard deviation of 10 networks. Each network has been simulated 10 times with different initial conditions and noise levels. The initial perturbations are normally distributed with mean 1000 and standard deviation 50. The noise is also drawn from a normal distribution, with mean 0 and standard deviation σ_μ.

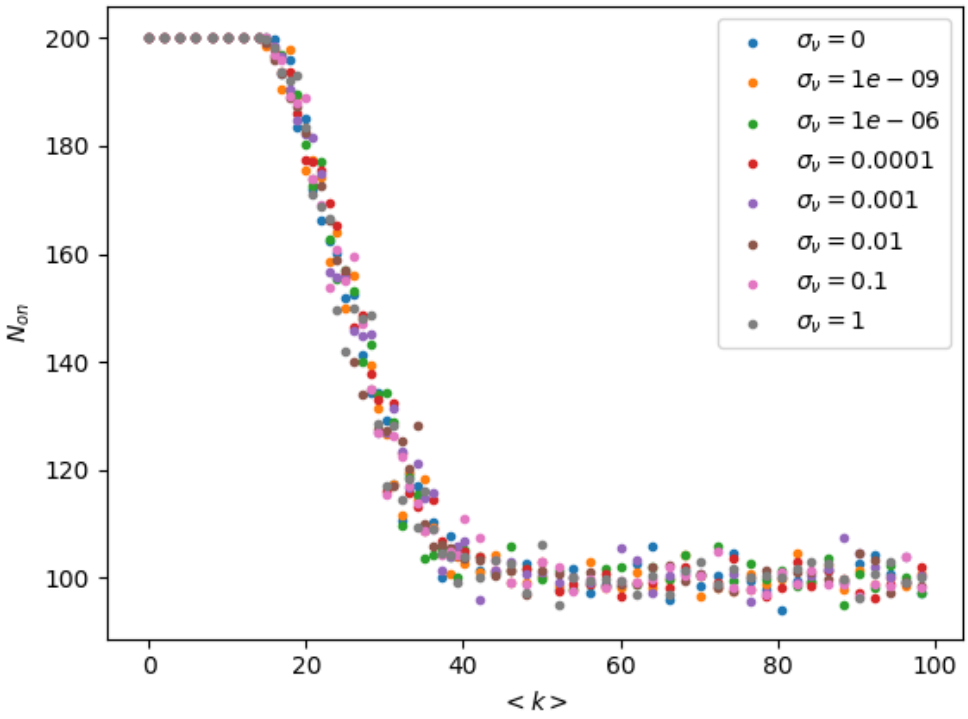
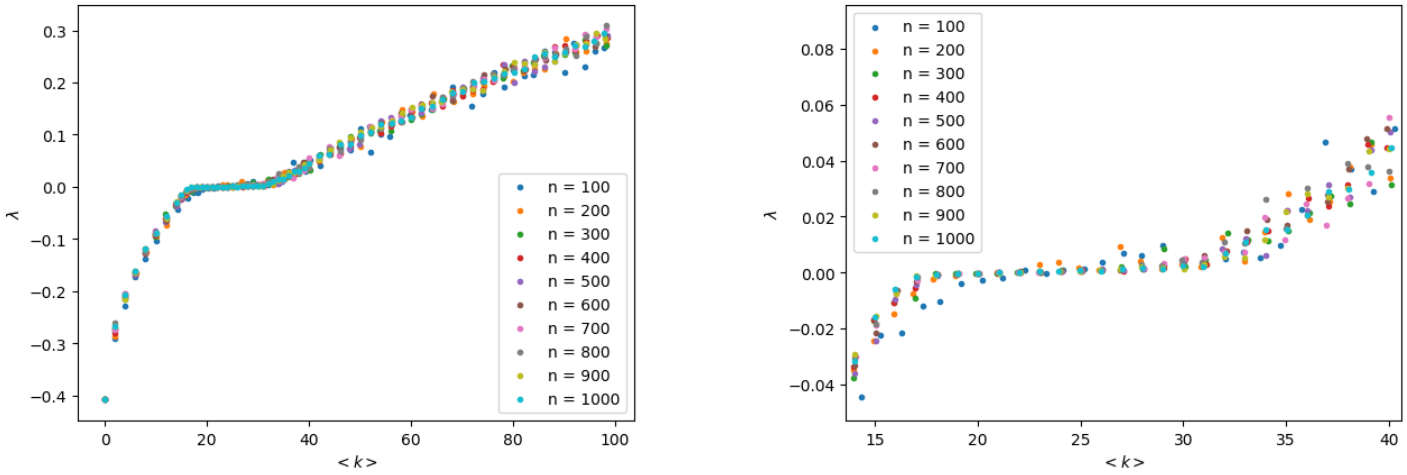


Figure 21: The average amount of nodes that are on at the end of the simulation, N_{on}, relative to the average connectivity of a network, $\langle k \rangle$. This corresponds to the data in figure 20.

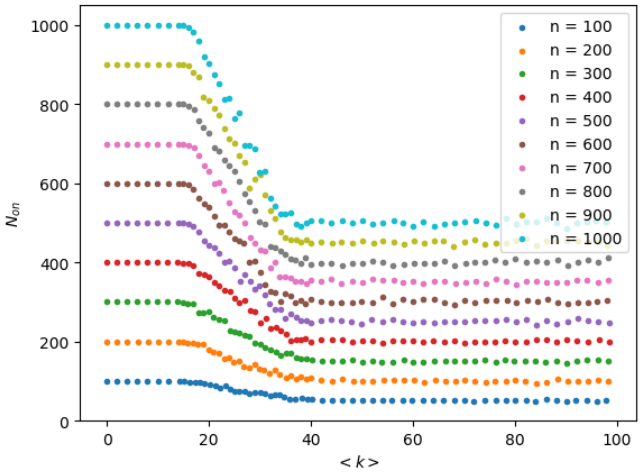
4.2.5 The Dynamics for Different Network Sizes

In the theory we have seen that the behavior of a network depends on it's size. The number of nodes determines the number of eigenvalues, and this affects the Lyapunov exponent, decreasing it by coefficient $C(N)$. We want to test empirically if this is truly the case, and whether the size of the network has other effects as well. In order to investigate this fact, we looked at the $\langle k \rangle$-Lyapunov relationships for Erdös-Rényi networks with varying sizes. The parameters for the weights in the network are $D = 0.4$ and $\sigma_A = 0.1$. For each target connectivity 10 networks were generated, and for each network the numerical solution was calculated for 10 different initial perturbations. The initial perturbations were drawn from a normal distribution with mean 1000 and standard deviation 50. The equilibrium, x_0 is located at 1000. The numerical solutions were calculated for 5000 steps with time step $\Delta t = 0.1$.

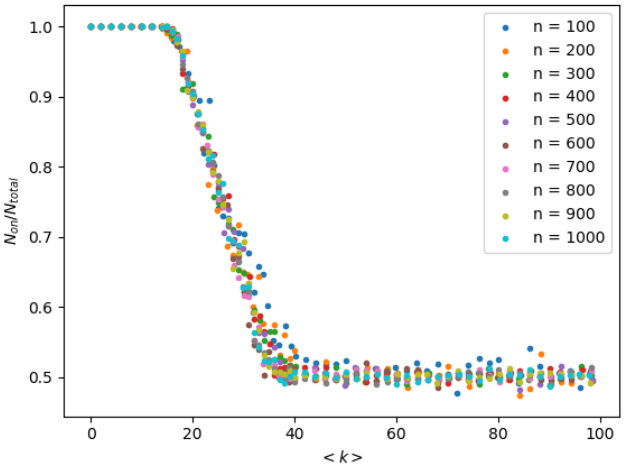


(a) The plot for $0 < \langle k \rangle < 100$ *(b) The plot for* $14 < \langle k \rangle < 40$

Figure 22: The relationship between the average connectivity of the network, $\langle k \rangle$*, and the Lyapunov exponent of Erdös-Rényi graphs of different sizes. The parameters of the weights of the edges are* $D = 0.4$ *and* $\sigma_A = 0.1$*. Each point represents 10 networks which were generated differently, but with the same target connectivity. For each network the numerical solution was calculated 10 times, with different realizations of the noise. The error bars are the standard deviation in their respective values.*



(a) The plots for N_{on}



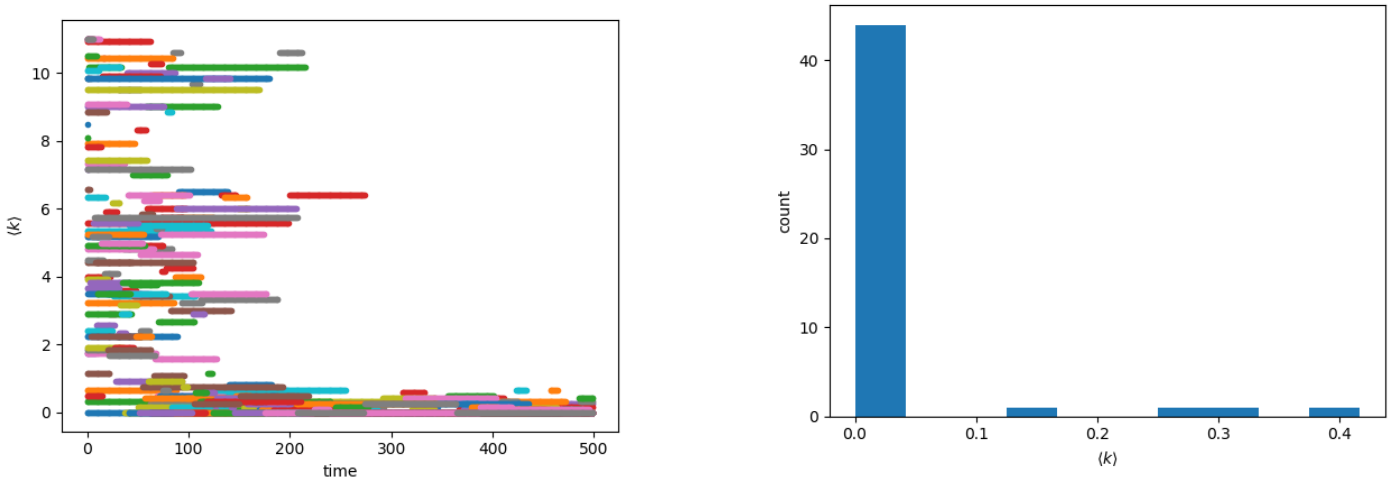
(b) The plots for N_{on} normalized for the network size.

Figure 23: The average amount of nodes that are on at the end of the simulation, N_{on}, relative to the average connectivity of a network, $\langle k \rangle$. The plots show are for different network sizes. This corresponds to the data in figure 22.

4.3 The Dynamics of an Interacting Population

4.3.1 The Evolutionary Algorithm

The results for an evolutionary algorithm for a population of 48 networks are shown in figure 24. The networks used were 12 node networks with parameters $D = 0.15$ and σ_A. This means that the stability plateau exists at about $2.5 < \langle k \rangle < 9$. The target connectivities of the initial population are $k = 0, 0.25, 0.5, 0.75, \ldots, 11.75$. The selection and mutation was carried out 500 times, with $\sigma_\nu = 0.1$. The environmental parameter is chosen to be $\alpha = 1$. Note that the connectivities in the population converge to zero.

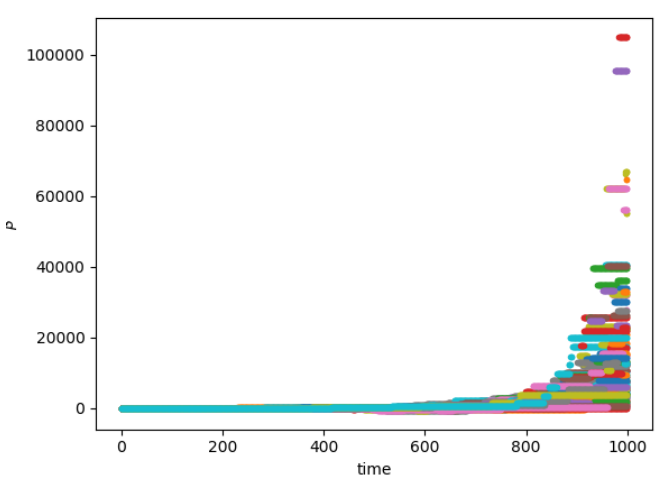


(a) A plot showing the lifetime of each individual *(b) A plot showing the distribution in $\langle k \rangle$ at the end of the simulation*

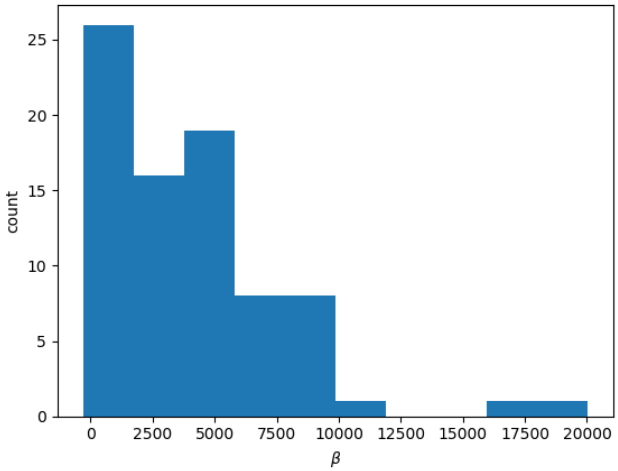
Figure 24: The results of the evolutionary algorithm for a population of 48 networks consisting of 12 nodes each. The networks used were 12 node networks with parameters $D = 0.15$ and σ_A. The initial connectivities were chosen to be uniformly spread from 0 to 12. The mutation in k was carried out with $sigma_\nu = 0.1$. The colors represent different individuals.

4.3.2 The Co-Evolutionary Algorithm

When applying the Co-Evolutionary algorithm, we looked at three types of populations. First, we looked at at population consisting of just one network, figure 25. Then we looked at a population of different networks, but with the same target connectivity, figure 26. Finally we looked at a population networks with different connectivities, figure 27.

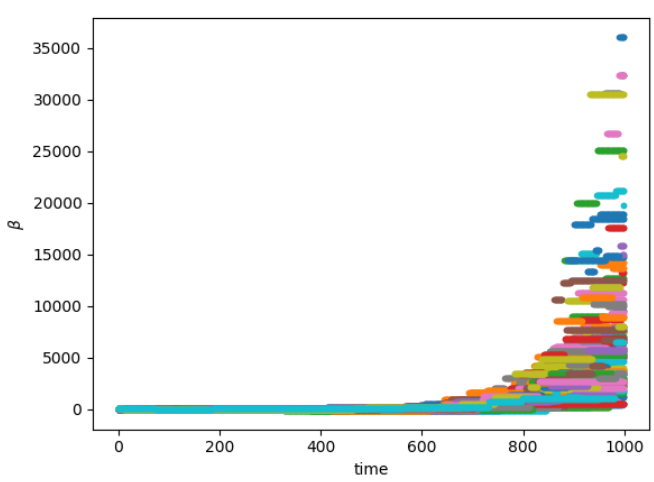


(a) A plot showing the lifetime of each individual

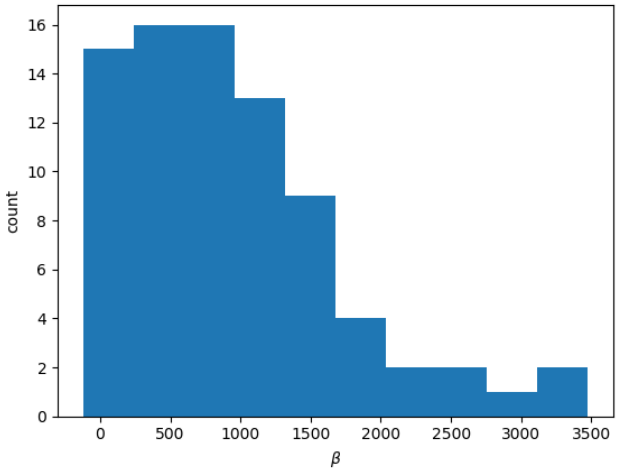


(b) A plot showing the distribution in β at the end of the simulation

Figure 25: The results of the co-evolutionary algorithm for a population for which all individuals have the same network structure, with $k = 4$. The population consists of 80 networks, each with 8 nodes and the same topology. The algorithm was run for 1000 steps, with subpopulation size $K = 2$, mutation parameter $\sigma_\mu = 0.1$, $D = 0.2$ and $\sigma_A = 0.1$. Initially the values of β were distributed uniformly between -10 and 10. The colors in figure (a) stand for the different individuals.

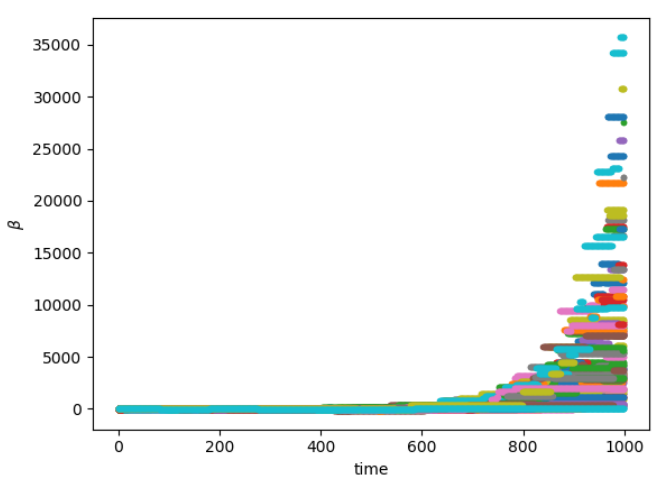


(a) A plot showing the lifetime of each individual

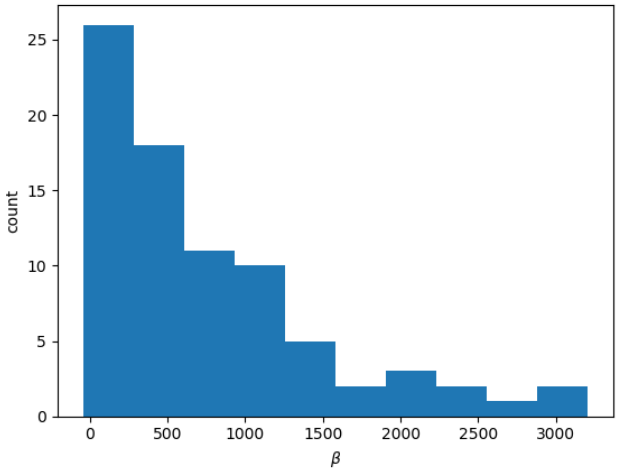


(b) A plot showing the distribution in β at the end of the simulation

Figure 26: The results of the co-evolutionary algorithm for a population for which all individuals have the same target connectivity, $k = 4$. The population consists of 80 networks, each with 8 nodes. The algorithm was run for 1000 steps, with subpopulation size $K = 2$, mutation parameter $\sigma_\mu = 0.1$, $D = 0.2$ and $\sigma_A = 0.1$. Initially the values of β were distributed uniformly between -10 and 10. The colors in figure (a) stand for the different individuals.



(a) A plot showing the lifetime of each individual



(b) A plot showing the distribution in β at the end of the simulation

Figure 27: The results of the co-evolutionary algorithm for a population which consists of netorks with connectivities in the range 0 to 20. The population consists of 80 networks, each with 8 nodes and the same topology. The algorithm was run for 1000 steps, with subpopulation size $K = 2$, mutation parameter $\sigma_\mu = 0.1$, $D = 0.2$ and $\sigma_A = 0.1$. Initially the values of β were distributed uniformly between -10 and 10. The colors in figure (a) stand for the different individuals.

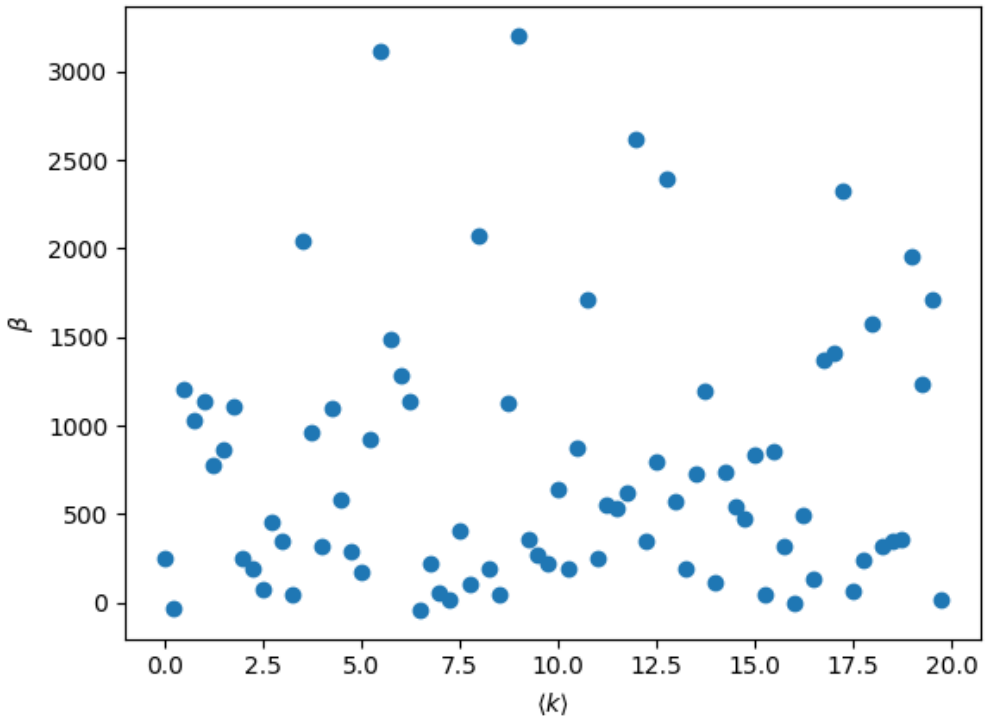


Figure 28: The correlation between the connectivity, $\rangle k\langle$ of a network and the value β of the corresponding network after 5000 selection steps. Note that there appears to be no correlation.

5 Discussion

5.1 The Dynamics of One Network

5.1.1 Numerical Methods

While our implementation of numerical methods to solve the minimally nonlinear stochastic differential equation (14) is efficient and accurate, it does have two major drawbacks compared to using analytical methods, such as the Fokker-Planck equations, to solve this differential equation. Primarily, numerical methods are limited to finite time; you cannot let your script compute what the value near infinity will be since you do not have infinite time to wait for your program to finish running. If you find the solution analytically, you are set. You will know the solution for all times, past the starting time of course. But why is it necessary to know the solution near infinite time? This is because the Lyapunov exponent is defined with a limit to infinity, see equation (19). When we are using finite simulations, this introduces an error, which decreases exponentially the longer your numerical solution is. The second limitation of using numerical methods is that computers do not work with all values in $\mathbb{R}$. In our case, we executed the calculations using the "float" datatype. Therefore, we are limited to numbers x with $-2^{1024} < x < 2^{1024}$. This is not a big issue in most cases, unless you are dealing with strong divergence in your solution, which is possible for our differential equations. What is more important is that this float datatype has a maximal resolution. Values can only be represented with their 15 most significant digits. More exceeds the resolution of the float datatype. This can cause issues when looking at solutions that converge strongly paired with noise. Normally, even with strong convergence one can store the offset from the equilibrium as a float. With noise however, since this value is small compared to the offset due to the noise it will become lost. Therefore, even if one subtracts the noise, this small value cannot be recovered.

5.1.2 The Lyapunov Exponent Versus the Connectivity

The resulting plot, seen in figure 14, is in accordance with our prediction. Primarily, we do see the plateau forming behavior, created in the switch between the configuration with all the nodes on, and the configuration with half the nodes on. Before the plateau, the data lies nicely along the prediction for the fully active network. Afterwards, it lies centered on the predicted curve for the network with half the nodes on. It makes the switch between these two regions when the Lyapunov exponent is zero. We can indeed see that this plateau is caused by an increasing amount of nodes that are off, see figure 15.

However, it does not create a perfectly flat plateau. The curve transitions smoothly between the upper prediction curve, to the plateau, to the lower prediction curve. This means that the Lyapunov exponents are slightly below zero for the first part of the plateau, and slightly above zero for the latter. What could possibly cause this? Well, this data is the average of 25 networks per cross. If we are looking at the early part of the plateau, where the points are lower than predicted, this means some networks have more nodes turned off than they should have. This is to be expected, in this early region of the plateau there are simply more possible states of the network where it has more nodes turned off than it should have than states where it has more nodes turned on than is should have. After all, it can never exceed having 500 nodes turned on. This could also explain why the latter part of the plateau is above the prediction. It is more likely that a network has less nodes turned off than is should have. A possible cause of this could be the fact that we are limited to finite times in our calculation for the Lyapunov exponents. Therefore, some nodes might still turn off given more time, but they have not yet within the numerical solution.

It is clear that the errorbars for λ are larger in the region where the data follows the prediction for a network with half the nodes on, than for the region where the network follows the prediction for the region with all the nodes on. One can find quite a simple explanation for this. In the region where all the nodes are on, truly all the nodes are on. This holds until one gets close to the plateau, near $k = 17$. It is unlikely that one node will turn of, as the asymptotically stable behavior of the numerical solutions of the minimally nonlinear stochastic differential equation does not permit the tracks to hit the non negativity condition boundary. One needs divergence to hit this. The spread in the data in this region is caused by the fact that different networks will have different eigenvalues. On the other hand, in the region where the network is only half turned on, one does not expect the number of nodes that are on to be consistently half of the total. A difference in the connections of the network and their weights could cause the network to have a little more or a little less nodes turned on. This will of course impact the Lyapunov exponent, causing a larger spread for different networks. These two facts are also observable in figure 15. In the fully on region, $0 < k < 17$, the data consistently says that all 500 nodes are on. However, in the half on region, $k > 45$, the points are more spread around the value of 250.

An important observation is that the error bars in the λ direction become relatively small at the plateau. This is more clearly seen in figure 14b. This is quite remarkable, it means that the networks that are in this plateau, will converge more readily to one type of dynamics; the oscillatory behavior when $\lambda = 0$.

5.1.3 The Dynamics of Different Types of Networks

Our prediction was that Erdös-Rényi networks and Scale-Free networks would have the same dynamics when it comes to the formation of the stability plateau. That is what makes this result in figure 16 so interesting. The plateau does indeed form for both networks. What is odd, is the fact that the data for the Scale-Free network does not follow the predicted curves for networks with all their nodes on, and for networks with half their nodes on. For small values of $\langle k \rangle$, the data appears to lie slightly above the prediction. This can however be accredited to the large errors, as these do come close to the predicted value. The strange behaviors occurs for large values of $\langle k \rangle$, here the data lies about 2 times the error below the predicted behavior. This must mean that the eigenvalues of a Scale-Free network are lower than than predicted with the Girko-circle theorem. A possible reason for this is that the adjacency matrix of a weighted Scale-Free graph is not a random matrix. Some type of correlation between the elements of this matrix must exist. If we look at the structure of a Scale-Free network, we can see that a few nodes have a lot of connections, while most nodes have less than the average amount of connections. This is due to the fact that the amount of edges per node follows a power law distribution. This explains why the terms in the matrix are correlated. If one node has a lot of outgoing edges, a lot of elements in the corresponding row are nonzero. Thus, there exists a correlation between one element in a matrix being nonzero, and the other elements in that row also being nonzero. This could explain why the data does not follow the behavior predicted with Girko's Circle theorem, as this assumes that the adjacency matrix is a random matrix. As is the case for Erdös-Rényi networks.

5.1.4 The Dynamics for Different Equilibrium Values

Figure 18, the relationship between the Lyapunov exponent and the average connectivity for networks with different equilibrium values, shows that the plateau forms for all the equilibrium values, even if they are orders of magnitude bigger than the initial perturbation. What is strange is that when the equilibrium value is larger than the initial perturbation, the plateau does not form at $\lambda = 0$, but at values greater than zero. How much greater than zero seems to

correlate logarithmically with the value of the equilibrium; the plateau lies 0.015 higher when the equilibrium value becomes 1000 times larger. This indicated that this effect has something to do with the logarithm in the estimator for the Lyapunov exponent, see equation (31). Moreover, for $\lambda > 40$ the data lies above the predicted curve by about the same amount. It makes sense that these values would be higher for larger equilibrium values, when the equilibrium value becomes so high that the tracks can never reach zero, and are thus stopped by the non negativity condition, the behavior would simply follow the predicted curve for networks with all nodes on. This could explain why our plateau lies above the predicted value, not all the nodes that will turn off have had time to turn off yet. When we look at figure 19, we see that the data corresponding to the largest equilibrium value, the red points, seems to start to decrease for larger values of $\langle k \rangle$ than the other end of the spectrum, the blue points. This is only a slight difference, but it could explain the light difference in plateau height for the different equilibrium values. If the equilibrium value is large, it takes longer for the tracks of the numerical solution to reach the non negativity condition, if they are headed that way. Therefore, not all the nodes will be off at the end of the numeric solution that would be off if this solution would be calculated any further. To see if this is really the case, one would have to experiment with how the simulation time affects the height of the plateau.

5.1.5 The Dynamics for Different Noise Levels

Adding noise to the simulations has one major drawback: The Lyapunov exponents become hard to calculate for asymptotically stable solutions. This can clearly be seen in figure 20. For small $\langle k \rangle$ values the Lyapunov exponent becomes nearly constant. This is no surprise. In the Theory chapter we say that the noise causes a constant deviation in the tracks if the real components of the eigenvalues are negative. In our case, this means that the standard deviation in the tracks at the end of the numerical solution will be a constant that depends on the noise level. Therefore, the Lyapunov exponent is a constant until the eigenvalues become large enough to dominate the noise. From that point on the data follows the curve in the predicted way, like in the case without noise. The stability plateau still forms. Figure 21, shows no unexpected behavior. No matter the noise level, the nodes turn off in the same way.

5.1.6 The Dynamics for Different Network Sizes

The data in figure 22 agrees with our predictions. The larges the network, the higher the Lyapunov-connectivity curve. This becomes apparent when comparing the 1000 node network data with the 100 node network data. The 1000 node network data is on average above the other data points, while the 100 node data is generally below the other data points. What we do see, but did not predict is that the plateau is "flatter" for networks with more nodes. What we mean by that is that the data in the plateau region lies more closely along $\lambda = 0$ for larger networks. Smaller networks deviate more from the plateau, the data is lower for smaller values of $\langle k \rangle$, and higher for larger values. This can be accredited to the effect mentioned in section 5.1.2, where since there are more possible node configurations that lean one way, the network will sometimes have a few nodes more or less turned off than it should. Now, why is this effect more prominent for smaller networks? Well, in a smaller network having one node more of less turned off has a bigger impact on the overall dynamics than in a large network.

5.2 Dynamics of an Interacting Population

5.2.1 The Evolutionary Algorithm

From the data in figure 24 it becomes clear that under our fitness function, the population converges to low connectivity values. $\langle k \rangle$ approaches zero quite quickly. This is not in accordance with our predictions. We thought the connectivities of the networks in the populaion would converge to the Lyapunov plateau. The reason for this must be that a the hamiltonian corresponding to a network with just self interaction between nodes generates a disctribution fuction that is closer to the distribution function of the Ising model. A possible reason for this development is the fact that by eliminating the connections between the nodes, one is eliminating terms from the hamiltonian, (36). Therefore, the different states of $\mathbf{s}$ will have less of an impact on it value. Due to the fact that there is a bigger difference in the values of the hamiltonian for the different network states, $Z_{int}(\beta)$, which is a sum over the negative exponent of these values will become smaller, see equation (38). Sometimes, a link term will exponent to be larger, sometimes smaller. However, due to the nature of the exponential functon, the larger values will be more pronounced. This adds up to make $Z_{int}(\beta)$ larger when adding more edges, and smaller when there are few edges. Thus, $P_{int}(\mathbf{s}|\beta$ becomes larger for networks with low connectivities. Therefore, the Kullback-Leibler divergence becomes smaller, making the individual more fit compared to the rest. Therefore, the hamiltonian we used must not be suitable to unify population dynamics with the stability dynamics of one network.

5.2.2 The Co-Evolutionary Algortihm

What becomes immediately obvious from figures 25, 26 and 27 is that the values of β do not converge, they diverge. Moreover, they do this at different rates for different random seeds. One thing is clear, even though the initial distribution of β is uniform from -10 to 10, the negative values become eliminated quite quickly. This divergence in β is also not as expected, we expected it to converge somewhere, and for the connectivity of the network to influence this convergence.

There must be a reason why β diverges to infinity. Our speculation is that beta becoming larger amplifies the value of the hamiltonian. Since the hamiltonian is negative, it will become more negative for larger β. This causes $Z_{int}(\beta)$ to become smaller, this happens faster for $Z_{int}(\beta)$ than for $\exp(-H_{int}(\mathbf{s}|\beta))$. This in turn causes $P_{int}(\mathbf{s}|\beta)$ to become larger, see equation (37). Therefore, the relative Kullback-Leibler divergence of a network with a large β will small.

Due to the fact that β diverger regardless of the network topology, there is no correlation between $\langle k \rangle$ and β for the population with different connectivities, as can be seen in figure 28.

5.3 Further Research

To anyone interested in researching this topic, we have the following suggestions:

1. Use analytical methods such as the Fokker-Planck equations to solve the minimally nonlinear differential equation. This eliminates the need for numerical methods, making the results more precise. Moreover, the Fokker-Planck equation gives the distribution function of the solution, not just one realization of this random variable. This is a more robust way of looking at the dynamics.

2. Investigate the effect number of time steps in the numerical solution has on the shape of the Lyapunov curve near the plateau.

3. In nature, pink noise, also called $1/f$ noise, tends to be more common than white noise. It would be interesting to see how this affects the dynamics.

4. Look at the population dynamics with a different fitness function for the evolutionary and co-evolutionary algorithm, one based on the eigenvalue spectrum of the network,

5. Use an evolutionary algorithm to determine the node states of a population of networks. Use this as the parameter that is minimized, instead of β. This makes more sense, as this is what a network used to adapt to it's environment. How well it can do this is what determines how fit the individual is.

6 Conclusion

In this work, we created a simplified model of gene regulation in a cell with networks and a minimally nonlinear stochastic differential equation to model the interactions in this network. We also implemented an evolutionary algorithm and a co-evolutionary algorithm to model the behavior of a population of networks. We predicted that due to the fact that concentrations cannot become negative, an "edge-of-chaos" region occurs, a range of possible network topologies where the networks have stable, yet adaptive behavior. We wanted to see if the population dynamics would also indicate that this region is favorable.

To evaluate the stability of the networks, we used the Lyapunov exponent. This parameter quantifies how stable the development two numerical solutions is relative to their initial values. It approximates the largest real component of the eigenvalues of the adjacency matrix corresponding to a network. This largest real component is dominant over the other eigenvalues when it comes to the stability of the network.

The research question of this project was: **"How do the stochastic dynamics of gene regulatory networks with different topologies compare, regarding a single network and an interacting population of networks?"** The sub questions that come into play when answering this question are: "What parameters play a role in the dynamics of a single gene regulatory network?", "How does the connectivity of a single network affect its stability behavior?", "How does the additive noise affect the stability behavior of a single network?", "Does the type of random network have an effect on the shape of the Lyapunov plateau?", and "Do populations of networks under selective pressure for adaptiveness converge to a Lyapunov plateau?".

To answer the first sub question, the parameters that affect the stability behavior of a network are the number of nodes, n, its average connectivity, k, the self interaction parameter, D, the standard deviation in the weights of the edges, σ_A, and the noise level, σ_μ, the equilibrium value of the concentrations, x_0, and finally the method chosen to generate the network. These parameters affect the behavior in the following ways.

The behavior of network dynamics for networks with different connectivities is as predicted. There is a region of possible connectivities where the networks exhibit stable behavior, referred to as the "Edge-of-chaos" plateau. Moreover, the data points in the plateau have smaller error bars than elsewhere. This suggests that networks that are in the plateau have a tendency to converge to stable, adaptive behavior.

We saw that the size affects the formation of the plateau, larger networks have a flatter plateau region. Additionally, when the equilibrium value becomes much larger than the perturbation, the plateau moves upwards.

When the noise level is less than the deviation in the initial perturbation, the plateau still forms. When it becomes larger nodes start to turn off due to the chaotic nature of a system with that much noise. Beforehand we thought that noise was necessary to observe the plateau behavior, to keep the system from settling in saddle points and small equilibria. However, it has become clear that the noise is not required for the formation of the plateau.

To answer the the third question, the type of network does have an impact on the stability behavior, but not on the formation of the plateau. Erdös-Rényi graphs perfectly follow our predictions, but Scale-Free graphs have lower Lyapunov exponents than expected.

Considering the population dynamics, both the evolutionary and co-evolutionary algorithm had unexpected outcomes. That is, they do not show any correlation between the connectivity of a network and the development of the population over time. This is likely due to the fitness function we used, the Kullback-Leibler divergence in accordance with a probability distribution defined by a Hamiltonian. It is suggested to use a different fitness function in follow up re-

search. For example, one that is related to the eigenvalue spectrum of the network. It is also recommended not to optimize for β in the evolutionary algorithm, but for the network state, $\mathbf{s}$.

7 Acknowledgements

First and foremost, I would like to thank Dr. Johan Dubbeldam for his guidance during this project. I am grateful for the countless hours he spent with me over Skype explaining concepts and discussing ideas. It was an excellent opportunity to get a taste of what doing research is like. Secondly, I would like to thank Dr. Timon Idema for providing feedback and ideas from the perspective of a biologist and a physicist, and for letting me be part of his wonderful research group. Naturally, I want to thank the members of the Idema group for their suggestions during out group meetings, but also for the good times we had outside the meetings. I would also like to thank Prof. Dr. Frank Redig and Dr. Ir. Lidewij Laan for being part of my thesis defense committee. Finally, I want to thank my girlfriend Pien for always being there for me, and for designing the cover of this report.

References

[Aardal K. , van Iersel L., Janssen R., 2019] Aardal K. , van Iersel L., Janssen R. (2019). *Optimization*. Delft University of Technology, Delft.

[Bijma, 1997] Bijma, F. (1997). *An Introduction to Mathematical Statistics*. Amsterdam University Press, Nieuwe Prinsengracht, Amsterdam, 1 edition.

[Edelman A., Kostlan E., Shub M., 1994] Edelman A., Kostlan E., Shub M. (1994). How many eigenvalues of a random matrix are real? *Journal of the American Mathematical Society*, 7:247–267.

[Girko, 1984] Girko, V. (1984). The circular law. *Teoriya Veroyatnostei i ee Primeneniya*, 29:669–679.

[Hagberg A., Schult D., Swart P., 2019] Hagberg A., Schult D., Swart P. (2019). Networkx reference. `https://networkx.github.io/documentation/stable/reference/index.html`. Consulted on 19-06-2020.

[Hanel R., Pöchacker M., Thurner S., 2010] Hanel R., Pöchacker M., Thurner S. (2010). Living on the edge of chaos: minimally nonlinear models of genetic regulatory dynamics. *Philosophical Transactions of the Royal Society*, 368:5583–5596.

[Hidalgo J., 2014] Hidalgo J., e. a. (2014). Information-based fitness and the emergence of criticality in living systems. *PNAS*, 111:10095–10100.

[Holme P., Kim B. J., 2002] Holme P., Kim B. J. (2002). Growing scale-free networks with tunable clustering. *Physical Review*, 65.

[Jeong H., Mason S.P., Barabási A.L., Oltvai Z.N., 2001] Jeong H., Mason S.P., Barabási A.L., Oltvai Z.N. (2001). Lethality and centrality in protein networks. *Nature*, 411:41–42.

[Jeong H., Tombor B., Albert R., Ottval Z.N., Barabási A.L., 2000] Jeong H., Tombor B., Albert R., Ottval Z.N., Barabási A.L. (2000). The large-scale organization of metabolic networks. *Nature*, 407:652–654.

[Kullback S., Leibler R. A., 1951] Kullback S., Leibler R. A. (1951). On information ans sufficiency. *The Annals of Mathematical Statistics*, 22:79–86.

[Liu, 2001] Liu, Y.-K. (2001). *Statistical Behavior of the Eigenvalues of Random Matrices*. Princeton University.

[Nelson, 2007] Nelson, P. (2007). *Biological Physics*. W.H.Freeman & Co Ltd, New York NY, 2 edition.

[Weisstein, 2020] Weisstein, E. W. (2020). Wigner's semicircle law. From MathWorld–A Wolfram Web Resource: `https://mathworld.wolfram.com/WignersSemicircleLaw.html`. Consulted on 27-05-2020.

Appendix

A.1 Noise Estimation

In order to correct for this, we would like to estimate the effect of the noise. We will do this for the case where the additive noise, μ_i, is wide-sense stationary white noise with mean σ and the multiplicative noise, ξ_i, j is zero. We are not considering the non-negativity condition for simplicity.

$$\frac{d}{dt}\delta x_i = \sum_j A_{i,j}\delta x_j + \mu_i \tag{49}$$

The solution to this differential equation is given in equation (50).

$$\mathbf{x}(t) = \int_0^t e^{A(t-\tau)}\boldsymbol{\mu}(\tau)d\tau + \mathbf{x_0}e^{A(t)} \tag{50}$$

Since $\boldsymbol{\mu}$ is wide-sense stationary, it is easy to see that the expectation value of $\mathbf{x}$ will be:

$$\mathbb{E}[\mathbf{x}(t)] = \int_0^t e^{A(t-\tau)}\boldsymbol{\mu}(\tau)d\tau + \mathbf{x_0}e^{A(t)} \tag{51}$$

We are more interested in its variance however, as the deviation from the expectation is what makes it difficult to find the Lyapunov exponent in the first place. Filling in the definition, equation (27), we get the result (55). If we assume that our matrix A is diagonalizable, and has eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$ all with multiplicity one, and corresponding normalized eigenvalues $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$.

$$\mathrm{Var}[\mathbf{x_i}(t)] = \Big| \int_0^t e^{A(t-\tau)}\boldsymbol{\mu}(\tau)d\tau \Big|_i^2 \tag{52}$$

$$\mathrm{Var}[\mathbf{x_i}(t)] = \int_0^t \int_0^t \sum_j \sum_k [e^{A(t-\tau)}]_{i,j}[e^{A(t-\tau')}]^*_{i,k}\mu_j(\tau){\mu_k}^*(\tau')d\tau d\tau' \tag{53}$$

$$\mathrm{Var}[\mathbf{x_i}(t)] = \int_0^t \int_0^t \sum_j \sum_k [e^{A(t-\tau)}]_{i,j}[e^{A(t-\tau')}]^*_{i,k}\sigma^2\delta(\tau-\tau')\delta_{j,k}d\tau d\tau' \tag{54}$$

$$\mathrm{Var}[\mathbf{x_i}(t)] = \sigma^2 \int_0^t \sum_j \Big|[e^{A(t-\tau)}]_{i,j}\Big|^2 d\tau \tag{55}$$

Using the fact that $\mathbb{E}(\mu_j(\tau){\mu_k}^*(\tau)) = \sigma^2\delta(\tau-\tau')\delta_{j,k}$

We can substitute the formula for calculating the exponent from the eigenvalues and eigenvectors:

$$e^{A(t-\tau)} = \begin{bmatrix} \mathbf{v}_1 e^{\lambda_1(t-\tau)} & \mathbf{v}_2 e^{\lambda_2(t-\tau)} & \cdots & \mathbf{v}_n e^{\lambda_n(t-\tau)} \end{bmatrix} \tag{56}$$

$$[e^{A(t-\tau)}]_{i,j} = [\mathbf{v}_j]_i e^{\lambda_j(t-\tau)} \tag{57}$$

$$\mathrm{Var}(\mathbf{x_i}(t)) = \sigma^2 \int_0^t \sum_j [\mathbf{v}_j]_i [\mathbf{v}_j]_i^* e^{\lambda_j (t-\tau)} e^{\lambda_j^* (t-\tau)} d\tau \tag{58}$$

$$\mathrm{Var}(\mathbf{x_i}(t)) = \sigma^2 \sum_j \frac{1}{2\mathfrak{Re}(\lambda_j)} [||\mathbf{v}_j||]_i^2 (e^{2\mathfrak{Re}(\lambda_j)t} - 1) \tag{59}$$

We now have a function that predicts the variance in the solution of our stochastic differential equation due to additive noise. This is function (55) in general, and function (59)

B.2 Figures

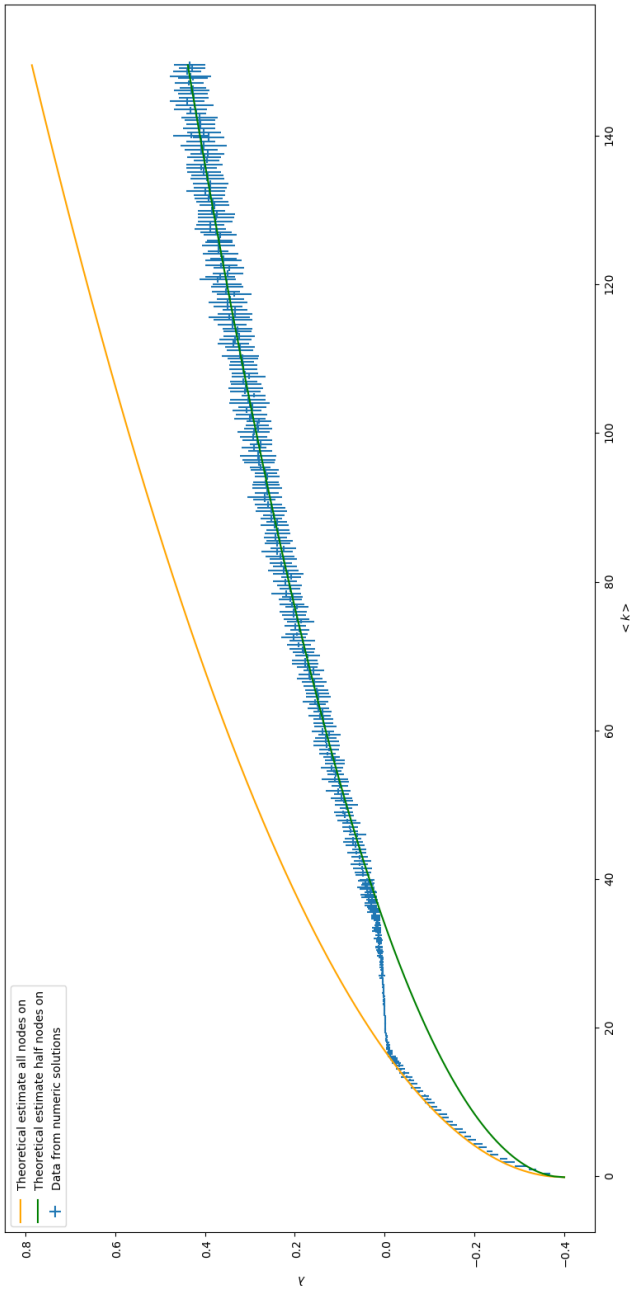

Theoretical estimate all nodes on
Theoretical estimate half nodes on
Data from numeric solutions
λ
< k >
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0
20
40
60
80
100
120
140

Figure 29

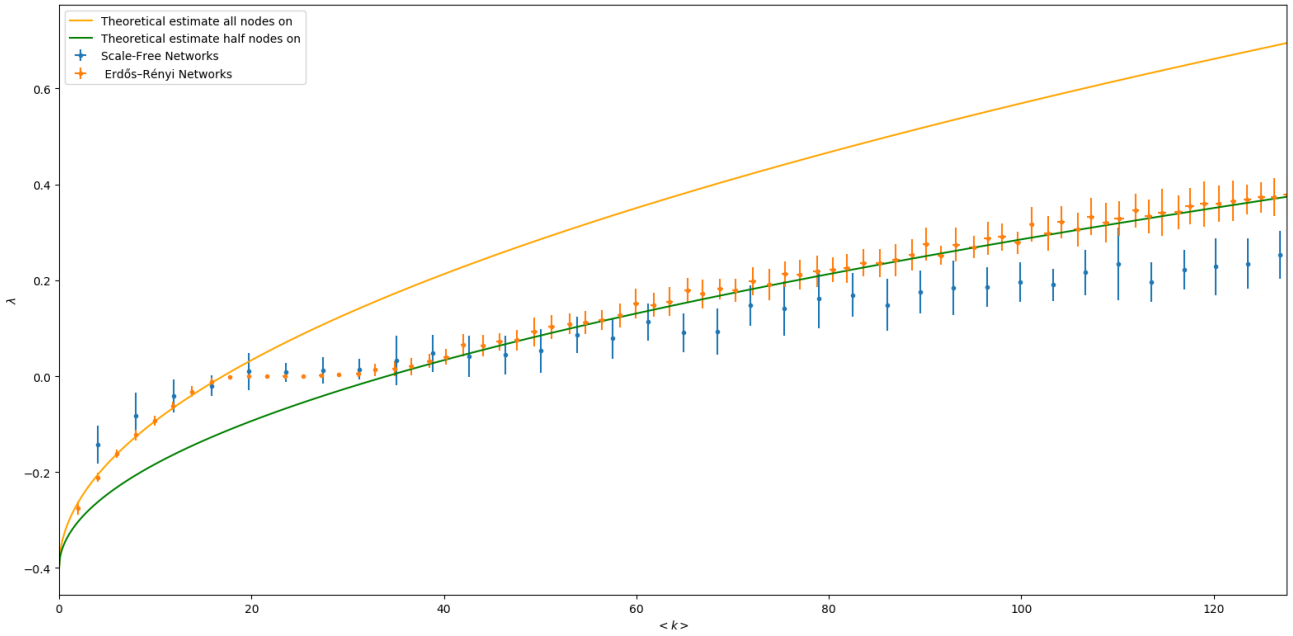
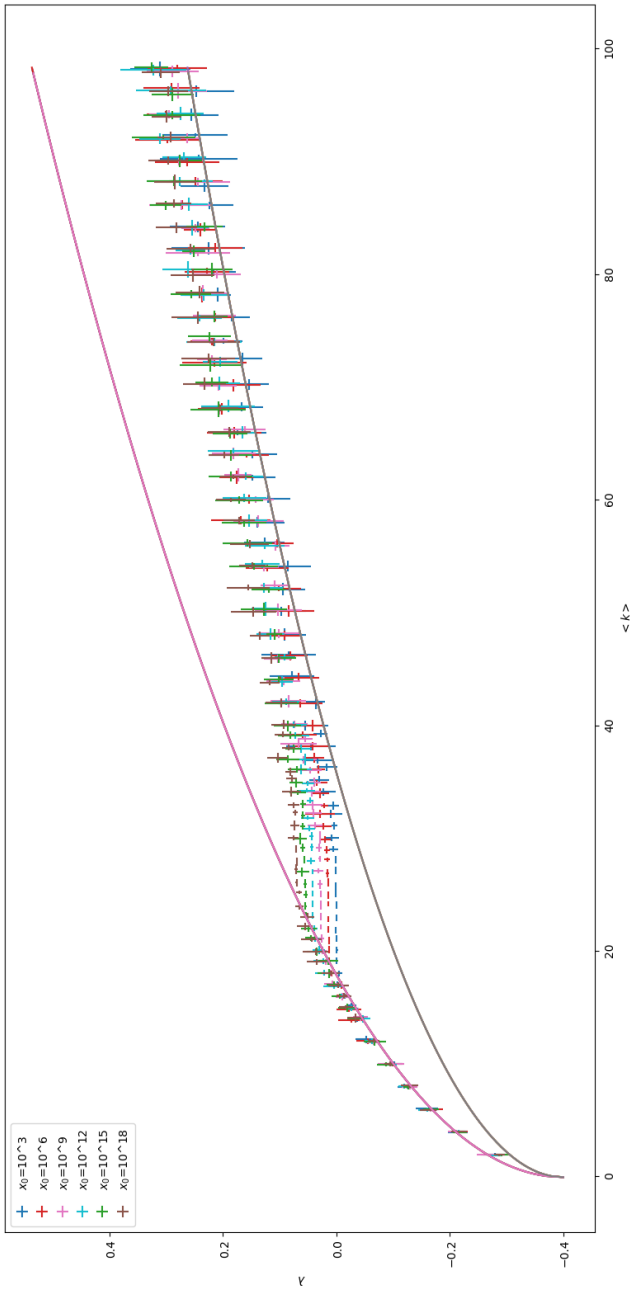

Theoretical estimate all nodes on
Theoretical estimate half nodes on
Scale-Free Networks
Erdős–Rényi Networks
λ
< k >
0.6
0.4
0.2
0.0
−0.2
−0.4
0
20
40
60
80
100
120

Figure 30


x₀=10^3
x₀=10^6
x₀=10^9
x₀=10^12
x₀=10^15
x₀=10^18
λ
0.4
0.2
0.0
−0.2
−0.4
0
20
40
60
80
100
< k >
Figure 31

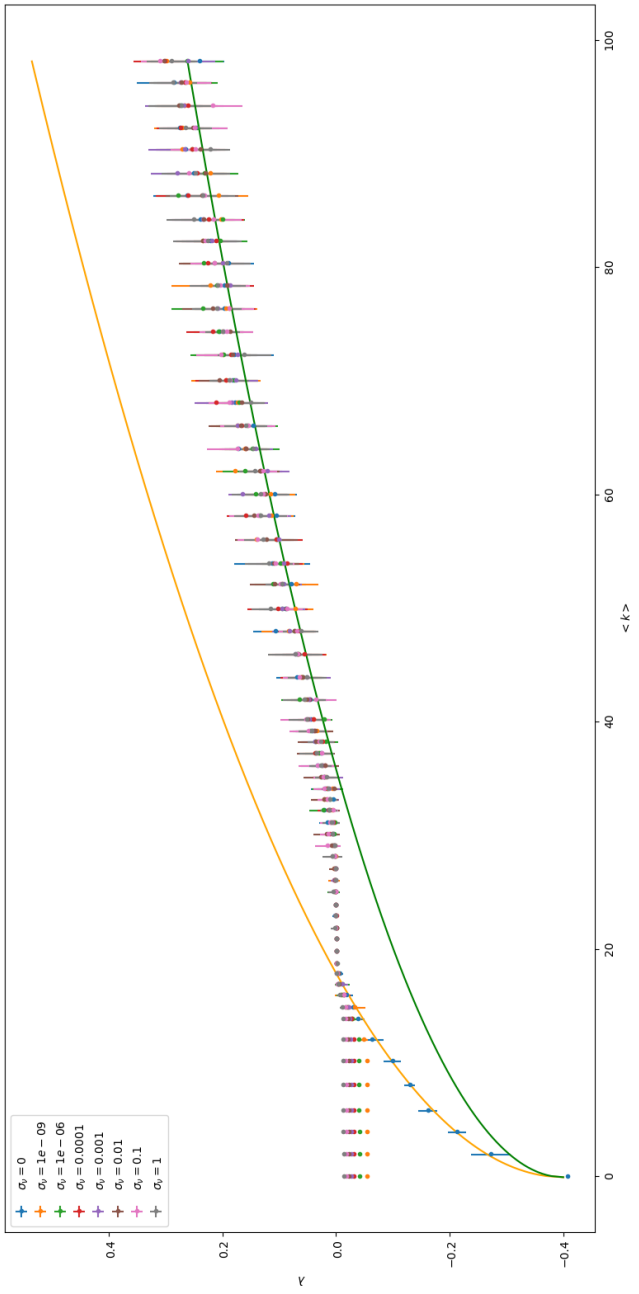

$\sigma_v = 0$
$\sigma_v = 1e-09$
$\sigma_v = 1e-06$
$\sigma_v = 0.0001$
$\sigma_v = 0.001$
$\sigma_v = 0.01$
$\sigma_v = 0.1$
$\sigma_v = 1$
λ
0.4
0.2
0.0
−0.2
−0.4
0
20
40
60
80
100
< k >

Figure 32

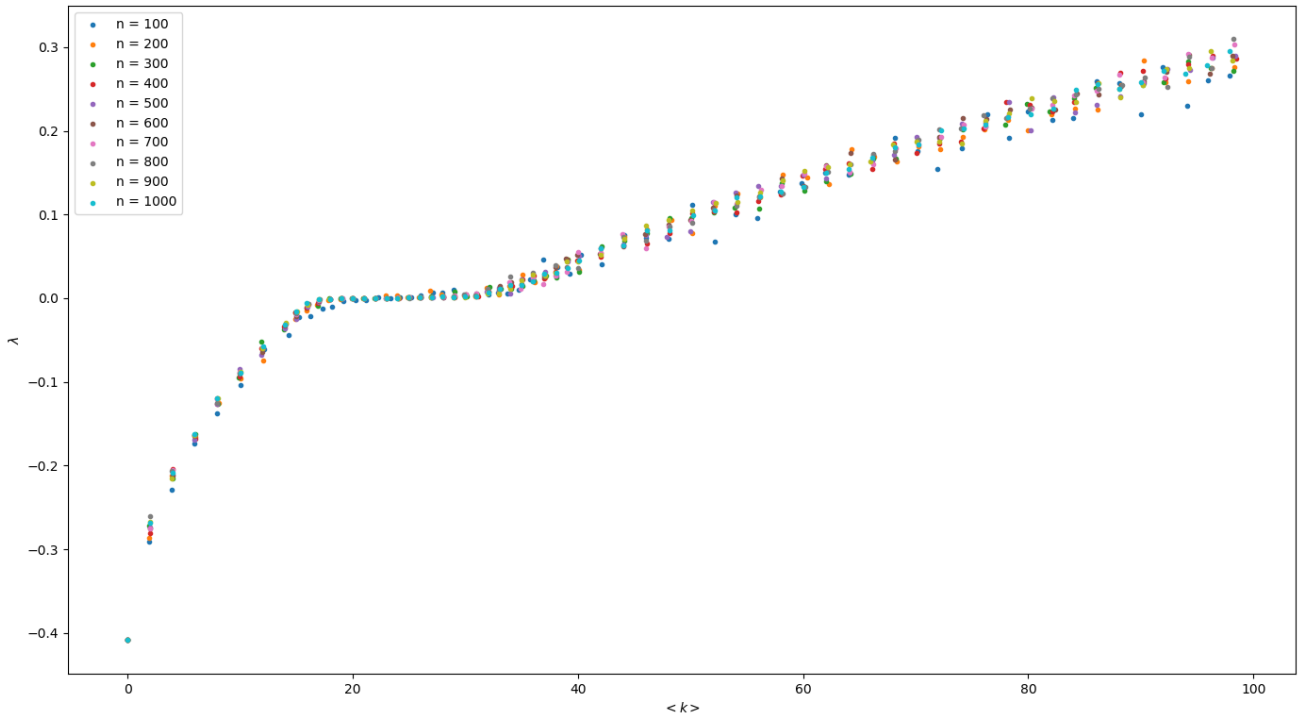

n = 100
n = 200
n = 300
n = 400
n = 500
n = 600
n = 700
n = 800
n = 900
n = 1000
λ
< k >
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.4
0
20
40
60
80
100

Figure 33