



Utrecht University

A thesis for the bachelor Mathematics and Applications

Spectral Theory for Sparse Random Networks

Veerle van den Hurk

Supervised by Dr. Ivan Kryven

November 15, 2020

Preface

Network theory caught my attention during the bachelor courses related to complex systems. What appealed to me the most about this subject is that networks have many applications in a lot of different disciplines. Furthermore, the fact that network theory is quite a new field in science, which gained a lot of attention in the last few decades, is also very interesting to me. Therefore, I am very glad that my supervisor Dr. Ivan Kryven presented the particular subject of this thesis to me. For this and all his help and enthusiasm during the past few months, I would like to thank him.

Contents

1	Introduction	3
2	Generating random networks	4
2.1	Configuration model	4
2.1.1	Properties	5
2.1.2	Semicircle law	6
3	Spectral density of locally treelike networks: an approximation	8
3.1	General spectral density	8
3.2	Specification for locally treelike networks	9
3.3	Approximation for the configuration model	11
3.3.1	Comparison between the message passing method and the approximation	13
3.3.2	Analytic solutions for $h(z)$	13
3.3.3	Edges of the spectrum	14
4	Results	17
5	Giant component	19
5.1	Network with nodes of degree a	19
5.1.1	Connection to the spectrum	19
5.2	Network with nodes of two different degrees	20
5.2.1	Network with nodes of degree 1 and b	20
5.2.2	Connection to the spectrum	20
5.3	Spikes	20
5.4	Spectrum of the giant component	21
6	Conclusion and discussion	23
	References	24
A	Python code	25

1 Introduction

We can think of a *network* as a set of points with lines connecting them. These points are called *nodes*, the lines are called *links*. Or, in graph terminology, vertices and edges. An example of a network is the network of Facebook users. Here, the nodes represent the users. If two users are friends, the nodes corresponding to those users will be connected by a link. For other basic network terminology, we would like to refer to *Chapter 6: Mathematics of Networks* of Newmans book *Networks* [5].

Most real-world networks, such as the World Wide Web, are represented by large sparse random graphs. Other examples of real-world networks are social networks, as our example of Facebook from the previous paragraph, and genetic networks. By sparse, we mean that most elements of the adjacency matrix are zero. Or in other words, only a small fraction of the possible links in the network is present.

We would like to analyse the set of eigenvalues of the adjacency matrix of such sparse graphs. This set of eigenvalues is called the spectrum. From this spectrum, we can derive important information about the structure of the network. We can, for instance, derive bounds on the eigenvalues to make statements about the stability of the network ([5], p. 698). Another example is that the spectrum can be used to calculate the eigenvector centrality, which is a measure of the importance of a node in the network ([5], p. 159).

For non-sparse large random graphs, it is known that the spectrum of the adjacency matrix follows the Wigner semicircle law [8]. This means that, under some conditions, the distribution of the eigenvalues is close to a semicircle in the limit of large network size. This result has many applications in physics [9]. In section 2.1.2, we will say more about this semicircle law and its conditions.

However, when we have a sparse random graph, just as those that represent real-world networks, the eigenvalues are not distributed according to the semicircle law [4]. It is interesting to see what the distribution of the eigenvalues will look like in this case. Analyzing this is our main goal for this thesis.

The outline of this thesis is as follows. In Chapter 2, the idea of random networks and how to generate them using the configuration model is explained. Furthermore, some useful properties of this model are described. We also derive for which kind of degree distributions the accompanying spectrum has a semicircular shape. Next, in Chapter 3, we derive an approximation of the spectral density for locally treelike networks generated by the configuration model. We also use this approximation to derive bounds on the edges of the spectrum. In Chapter 4, we present the results of this approximation of the spectral density and the derived bounds with the help of different network examples. Then, in Chapter 5, we take a closer look at whether some simple networks generated by the configuration model have a giant component, and what the effect of having a giant component is on the spectrum. We also discuss the spikes we see in certain spectra and look at the spectrum of the giant component alone. Finally, in Chapter 6, we arrive at our conclusions.

2 Generating random networks

When we want to create an artificial network with known properties, we use a network model. However, usually not all properties of a network are known. The network thus has some unknown properties, and therefore is random. To generate these kind of networks, we use random network models.

A *random network model* is an ensemble of networks and it is defined as a probability distribution over the possible networks. When we examine the properties of such a model, we look at the whole network ensemble and its average properties. When the size of a generated network increases, its properties are closer to the average properties of the network ensemble.

An example of a random network model is the $G(n, p)$ model ([5], Chapter 11). This model generates networks with n nodes, where each distinct pair of two nodes is connected by a link with probability p . $G(n, p)$ is one of the most well known models for generating ensembles of random graphs because its properties can easily be determined. Other names that are often used to refer to this model are 'Erdos-Renyi model' or 'Poisson random graph'. Although the $G(n, p)$ model is a widely studied random network model, it is not a very representative model for real-world networks. For example, one of the disadvantages of this model is that it generates networks that have a Poisson degree distribution in the limit of large network size, whereas real-world networks can have very different degree distributions. Since we want to analyse properties of real-world networks, we cannot use this random network model. Instead, we will apply the configuration model to generate random network ensembles.

2.1 Configuration model

Because of the reason stated in the previous paragraph, we introduce the following random network model in this section: the configuration model ([5], Chapter 12). The networks in this ensemble are still very simple, which makes it possible to determine certain properties. A very important advantage of this model, is that the degree distribution is not determined. We can thus choose for the network to have any degree distribution we want.

To generate a network with the configuration model, we need a given degree sequence. The *degree sequence* is a sequence of numbers that represent the degrees of each of the n nodes in the network. This sequence fixes the number of links in the network, since the sum of the degrees equals twice the number of links. With the help of an example shown in figure 1 below, we can show how we generate a network from a determined degree sequence.

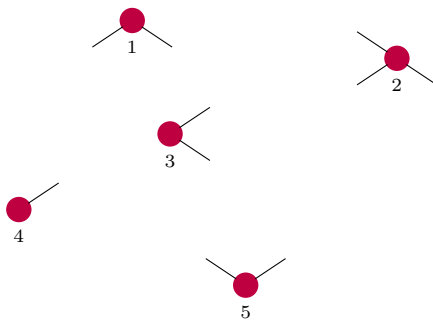


Figure 1: Nodes with half edges corresponding to the degree sequence $[2, 3, 2, 1, 2]$

Let the degree sequence $[2, 3, 2, 1, 2]$ be given. We now give all of the five nodes a number of so called 'half edges' corresponding to its degree. In Figure 1, the nodes are labeled. Next, we choose a pair of half edges uniformly at random and connect them: a link is made. We continue choosing pairs of half edges uniformly at random until all of the half edges are connected.. Now, we have a random network generated by the configuration model.

Note that the degrees in the sequence have to add up to an even number. If this is not the case, we have a single half edge left at the end of the generating process. We then cannot connect this half edge to another unused one. Consequently, the degrees of the nodes do not correspond to the given degree sequence. Another remark we can make is that a network generated by this model can contain self-loops and multi-links. *Self-loops* are links between two half edges that belong to the same node. *Multi-links* are created when we add a link between two nodes that are already connected by a link. However, self-loops

and multi-links tend to disappear in the limit of large network size.

Within this model, every half edge in the network is equally likely to form a link with every other half edge. The configuration model thus generates every possible network corresponding to the given degree sequence with the same probability.

However, there are cases when we only have a given degree distribution instead of a degree sequence. The *degree distribution* is a probability distribution on the non-negative integers k . So p_k gives the probability that a node, which is randomly chosen, has degree k . Also, p_k gives us the fraction of nodes that have degree k . ([5], p. 314)

In this case, we can still use the configuration model, but we first have to generate a degree sequence at random for the n nodes from the given distribution p_k . Subsequently, we can proceed with generating a network as described above. Once again, the differences between a generated network from a given degree sequence and one from a given degree distribution vanish in the limit of large network size.

2.1.1 Properties

In this section, we discuss some of the properties of the configuration model ([5], Chapter 12). These properties will be important in the following chapter(s) of this thesis.

Locally treelike The configuration model generates networks that are locally treelike. *Locally treelike* means that, if you start at a node and look at the set of nodes at a certain distance from this starting node, this set will take the form of a tree (with a probability that tends to 1 if $n \rightarrow \infty$). A *tree* is an undirected network which has no loops, but is connected. By *connected* we mean that the network consists of one part. Figure 2 below shows an example of a tree.

Since a tree has no loops, certain properties are simple to calculate. For example, there exists only one possible path between any two nodes. A *path* is a route from node to node in the network that does not intersect itself. If there were two possible paths, we could walk the first path from node a to node b , and then take the other one to go back to node a , which then completes a loop. Then, by definition, our network is not a tree. ([5], p. 121-122)

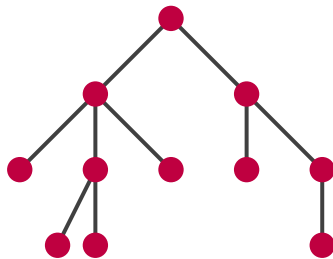


Figure 2: An example of a tree

Excess degree As described before, we can choose the network generated by the configuration model to have a certain degree distribution p_k . What turns out to be interesting as well is the degree distribution of a node which is reached by following a link. So, if we pick a link uniformly at random, what is the probability that the node at the end of that link has degree k ?

We know that the sum of the degrees k_i in our degree sequence equals twice the number of edges m , $\sum_{i=1}^n k_i = 2m$. Therefore, if we pick a link uniformly at random, the end of that link is connected to one of the $2m - 1$ other half edges. Since we work with the configuration model, every pair of half edges is equally likely to be connected. So our chosen link has to be connected to one of the k out of the $2m - 1$ half edges in order to end at a particular node of degree k . Such is the case with probability $\frac{k}{2m-1}$.

In our network, we have np_k nodes of degree k . Therefore, the probability to end at **any** node with degree k by following a link is

$$\frac{k}{2m-1} np_k. \quad (1)$$

In this thesis, we work with large networks. Meaning we can drop the -1 in the denominator of equation (1). Note that the average degree $\langle k \rangle$ equals $\frac{2m}{n}$, so equation (1) becomes

$$\frac{k}{\langle k \rangle} p_k. \quad (2)$$

Later on in this thesis, we will look at the number of links connected to the node we arrive at, without taking the link we followed into consideration. This number of links is called the *excess degree* and it equals the degree k of that node minus 1. From equation (2) we now find that the excess degree distribution q_k is given by

$$q_k = \frac{k+1}{\langle k \rangle} p_{k+1}. \quad (3)$$

The probability of the node you arrive at by following a link having an excess degree of k thus depends on the probability of that node having a total degree of $k+1$.

Giant component A *giant component* is a component (connected part) in the network which size grows for growing n . For random networks, the probability of having two giant components goes to zero in the limit of large network size. ([5], p. 306-307)

Whether a network generated by the configuration model has a giant component, depends on the following fraction

$$\frac{\text{average number of second neighbours}}{\text{average number of first neighbours}}.$$

By *first neighbours of node i* we mean the nodes at distance 1 from node i , the *second neighbours* are defined in a similar way.

Note that the average number of first neighbours just equals the average degree of the network $\langle k \rangle$. The average number of second neighbours is $\langle k \rangle \times \text{average excess degree}$. We now conclude that the fraction stated above just equals the average excess degree. The network generated by the configuration model thus has a giant component if and only if

$$\text{average excess degree} > 1. \quad (4)$$

Note that, if a network has no giant component, it is completely made up of small components. Because the configuration model generates networks that take the form of a tree in every local neighbourhood when $n \rightarrow \infty$, these small components must be trees.

2.1.2 Semicircle law

In the introduction, it was stated that non-sparse random graphs have a spectrum that follows the semicircle law of Wigner [8]. In the previous sections, we learned how to generate networks with a specified degree distribution by using the configuration model. It is interesting to see for which degree distributions we get a generated network that has eigenvalues distributed close to a semicircle.

First, we need to state the conditions of the semicircle law ([7], p. 159-161). The spectral distribution of Wigner matrices converges almost surely to the Wigner semicircular distribution if the following holds

- The upper triangular elements w_{ij} where $j > i$ of the matrix are identically and independent distributed complex random variables. They also need to have a mean of zero and a unit variance.
- The diagonal elements w_{ii} are identically and independent distributed real random variables. They are independent of the upper triangular elements w_{ij} where $j > i$. These diagonal elements need to have a bounded mean and variance.

We would like to analyse the spectral distribution of the adjacency matrix of a network generated by the configuration model. Important to note is that our adjacency matrix is a real symmetric matrix if our network is undirected. Wigner matrices include real symmetric matrices.

Thus, for the spectrum of an undirected network to follow the semicircle law, the upper triangular elements (and because of symmetry, also the lower triangular elements) of its adjacency matrix have to be independent and identically distributed random variables. For the configuration model however, we defined the degrees k_j ($j = 1, \dots, n$) of the nodes to be independent and identically distributed random variables, not the elements a_{ij} ($i, j = 1, \dots, n, i \neq j$) of the adjacency matrix. Nevertheless, we know that the j th column (and row, in case of an undirected network) sums up to the degree of the j th node. This means that $\sum_{i=1}^n a_{ij} = k_j$. Since the configuration model generates networks where self-loops tend to disappear in the limit of large network size, the diagonal elements of the adjacency matrix tend to be all zero. This is also what we expect from the treelike structure of the configuration model. Note that this

means that the conditions of the second statement mentioned above hold, and that $\sum_{\substack{i=1 \\ i \neq j}}^n a_{ij} = k_j$. We now see that for the adjacency matrix of the network generated by the configuration model to follow the semicircle law, we need the random variables k_j to have a degree distribution that can be obtained by summing $n - 1$ independent and identically distributed random variables a_{ij} , where $i \neq j$.

We now know for what kind of degree distributions the configuration model can generate networks for which the semicircle law applies. One particular distribution that immediately comes to mind is the binomial degree distribution. A random variable k_j that follows this degree distribution can be obtained by summing $n - 1$ independent and Bernoulli distributed random variables a_{ij} , where $i \neq j$. To clarify this, we know that a random variable X is Bernoulli distributed if $P(X = 1) = p = 1 - P(X = 0)$. For the X_1, \dots, X_{n-1} independent and Bernoulli distributed random variables, it follows that

$$\sum_{k=1}^{n-1} X_k = \binom{n-1}{k} p^k (1-p)^{n-1-k},$$

where the right hand side defines the binomial distribution with parameters $n - 1$ and p . Unfortunately, the Bernoulli distributed random variables do not have the necessary condition of a zero mean as stated above. However, as we see in figure 3 below, the spectrum of a network with a binomial degree distribution generated by the configuration model does have a semicircular shape.

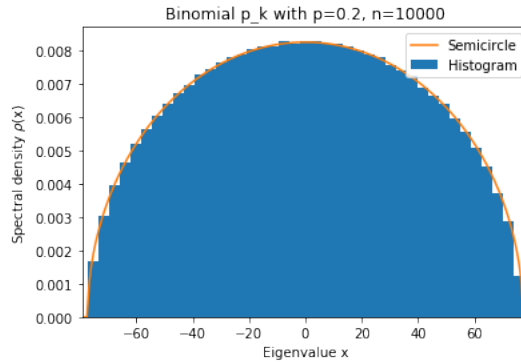


Figure 3: Spectrum of a network with a binomial degree distribution

It turns out that networks with a binomial degree distribution generated by the configuration model do follow the semicircle law, despite the Bernoulli random variables in the adjacency matrix not having a zero mean. This result was presented in Theorem 2.9 of a paper by Erdos, Knowles, Yau and Yin [3], in which they show that Erdos-Renyi random networks follow the semicircle law. Note that we stated above that these kind of networks generated by the $G(n, p)$ model have a Poisson degree distribution in the limit of large network size. However, the exact degree distribution of these networks is the binomial distribution and the elements of its adjacency matrices are independent (except for symmetry) and Bernoulli distributed random variables.

3 Spectral density of locally treelike networks: an approximation

In this section, we eventually derive an approximation to calculate the spectral density of a locally treelike network. To derive this approximation, we start with rewriting the spectral density function. With this we use the properties of a locally treelike network, to then approximate the derived equations for the case of the configuration model. This method is stated in the paper *Spectra of random networks with arbitrary degrees* by Newman, Zhang and Nadakuditi [6], which we follow in this chapter.

Assume that we have a locally treelike network which is undirected and unweighted. This network is made up of n nodes, where $n \rightarrow \infty$. Because we have an undirected network, the adjacency matrix of this network is symmetric. Its elements are also real, namely zeros and ones.

3.1 General spectral density

The spectral density function describes the distribution of the eigenvalues of the network's adjacency matrix. Note that, since our adjacency matrix is real and symmetric, all eigenvalues are real. The spectral density of an eigenvalue x is given by

$$\rho(x) = \frac{1}{n} \sum_{i=1}^n \delta(x - \lambda_i). \quad (5)$$

This is the general formula for the spectral density, it therefore also holds for networks which are not locally treelike. In this equation, $\lambda_1, \dots, \lambda_n$ are the n eigenvalues of the adjacency matrix of the network. Furthermore, $\delta(x)$ describes the Dirac delta function. The Dirac delta function is a generalized function which is zero at all points except for $x = 0$. At this point the function is undefined, whereas the measure over the whole domain is 1 by definition ([1], p. 918-919).

The delta function can be written as the limit of a Cauchy distribution, which gives us

$$\delta(x) = \frac{1}{\pi} \lim_{\epsilon \rightarrow 0^+} \frac{\epsilon}{x^2 + \epsilon^2}. \quad (6)$$

The probability density function of a Cauchy distribution also looks like a single peak centered at $x = 0$. The parameter ϵ describes the width of this peak ([1], p. 447). Now, we can express this as

$$\delta(x) = \frac{1}{\pi} \lim_{\epsilon \rightarrow 0^+} \operatorname{Im} \frac{1}{x - i\epsilon} = -\frac{1}{\pi} \lim_{\epsilon \rightarrow 0^+} \operatorname{Im} \frac{1}{x + i\epsilon}. \quad (7)$$

Next, we substitute (7) into (5) and find the spectral density to be

$$\rho(x) = -\frac{1}{n\pi} \lim_{\epsilon \rightarrow 0^+} \operatorname{Im} \sum_{i=1}^n \frac{1}{x - \lambda_i + i\epsilon} \quad (8)$$

For simplicity, we define $z = x + i\epsilon$ and obtain

$$\rho(z) = -\frac{1}{n\pi} \sum_{i=1}^n \frac{1}{z - \lambda_i}. \quad (9)$$

Note that this function gives the spectral density $\rho(x)$ of the network if we take the limiting value of the imaginary part when ϵ goes to zero from above.

In terms of matrix theory, equation (9) equals

$$\rho(z) = -\frac{1}{n\pi} \operatorname{Tr}(z\mathbf{I} - \mathbf{A})^{-1}, \quad (10)$$

where \mathbf{A} is the $n \times n$ adjacency matrix of the network and \mathbf{I} is the $n \times n$ identity matrix. We can now expand the matrix inverse $(z\mathbf{I} - \mathbf{A})^{-1} = \frac{1}{z}(\mathbf{I} - \frac{\mathbf{A}}{z})^{-1}$ as a Neumann series. From substituting $\frac{1}{z}(\mathbf{I} - \frac{\mathbf{A}}{z})^{-1} = \frac{1}{z} \sum_{k=0}^{\infty} \frac{\mathbf{A}^k}{z^k}$ in equation (10), we get

$$\rho(z) = -\frac{1}{n\pi z} \sum_{k=0}^{\infty} \frac{\operatorname{Tr} \mathbf{A}^k}{z^k}, \quad (11)$$

where we used that we take the trace of the Neumann series term by term. Unfortunately, this sum does not converge for all z . Therefore, we derive the radius of convergence for z to see when the sum converges. We see that

$$\lim_{k \rightarrow \infty} \left| \frac{\text{Tr} \mathbf{A}^{k+1}}{z^{k+1}} \frac{z^k}{\text{Tr} \mathbf{A}^k} \right| = \lim_{k \rightarrow \infty} \left| \frac{\text{Tr} \mathbf{A}^{k+1}}{z \text{Tr} \mathbf{A}^k} \right|$$

From $\lambda_1 > \frac{\text{Tr} \mathbf{A}^{k+1}}{\text{Tr} \mathbf{A}^k}$, we conclude that the sum certainly converges for $\frac{1}{|z|} < \frac{1}{|\lambda_1|}$. Here, we assumed that λ_1 is the largest absolute eigenvalue. For smaller values of z , the spectral density can be computed from equation (10).

We now take a closer look at the part $\text{Tr} \mathbf{A}^k$. First, we have to define a few things. A *walk* is any route along the edges of the network, from node to node. A *closed walk* is a walk that starts and ends in the same node, it therefore is a loop. Lastly, the *length of a walk* is the number of edges passed by taking the route.

The values of the elements b_{ij} of the matrix \mathbf{A}^b are equal to the number of walks of length b between node i and node j . Therefore, the diagonal element b_{ii} of this matrix gives us the number of closed walks of length b that start and thus end in node i . If we then sum the diagonal terms, we obtain the total number of closed walks of length b in the network. Consequently, $\text{Tr} \mathbf{A}^k$ tells us how many closed walks of length k there are in total in the network. ([5], p 131-132)

3.2 Specification for locally treelike networks

Up till now, we manipulated the general equation (5) for the spectral density of a network. We derived that we need to know the number of closed walks of length k , for all values of k in the network, to compute the spectral density. In the beginning, we assumed that we have a locally treelike network. We now can apply the properties of such a network to determine the number of closed walks, and with that the spectral density. To determine the number of closed walks in the network, we are going to derive a message passing equation. The concept behind this idea is that we split our large network into smaller pieces, so the calculations become more manageable. Recently, Newman and Cantwell derived the spectral density for sparse symmetric matrices of networks with loops in a similar way [2].

In section 2.1.1, we concluded that locally treelike networks only have one possible path between any pair of nodes in the network. The existence of another possible path creates a loop, then by definition we do not have a locally treelike network anymore. Because of this property, the only way to create a closed walk in this network is to follow a path starting at node a and ending at node b , and follow this same path backwards starting at node b to node a . A closed walk thus consists of an even number of links, since each link is traversed twice.

We first want to calculate n_{2r}^{uv} , which we define to be the number of closed walks of length $2r$ starting by traversing the link from u to v . We thus assume that the first step in our closed walk is traversing the link from u to v . Since we then end our closed walk by going from v to u , this link is traversed twice (once in each direction). The other links in our closed walk may be traversed more often, but still an even number of times.

Because we cross the link from u to v twice, the length of the remaining closed walk from v to v has to be $2r - 2$. Note that this walk can consist of only one single excursion of length $2r - 2$ from v to v , or m smaller excursions of length $2r_i$, where $\sum_{i=1}^m 2r_i = 2r - 2$. When we have multiple smaller excursions, we thus visit the node v at the end of every excursion. At this node, the next excursion also starts. The length of an excursion $2r_i$ is always even, because we cross every edge twice. We now can varyate between lengths of single excursions $2r_i$, and the total number of excursions m . But we need to remember that the combination of m excursions, each with a certain length, only contributes to the number n_{2r}^{uv} when $\sum_{i=1}^m 2r_i = 2r - 2$. This is why we have to include a Kronecker delta function into our function for n_{2r}^{uv} , where

$$\delta(2r - 2, \sum_{i=1}^m 2r_i) = \begin{cases} 1 & \text{if } 2r - 2 = \sum_{i=1}^m 2r_i \\ 0 & \text{else} \end{cases}$$

Of course, the number of closed walks of length $2r_i$ when we start at node v can also be described by $n_{2r_i}^{vw_i}$. Here, we assume that the first link that is crossed is from node v to node w_i . Obviously, if we want to know n_{2r}^{uv} , we need to include all possibilities for node w_i and multiply them. Therefore we define

N_v to be the set of nodes that are connected to v by a link. Combined with the fact that all of the m excursions start by traversing a link from v to one of its neighbours, and the total number n_{2r}^{uv} is the product of the number of distinct excursions, we can now derive the following formula

$$n_{2r}^{uv} = \sum_{m=1}^{\infty} \left[\sum_{\substack{w_1 \in N_v \\ w_1 \neq u}} \cdots \sum_{\substack{w_m \in N_v \\ w_m \neq u}} \right] \left[\sum_{r_1=1}^{\infty} \cdots \sum_{r_m=1}^{\infty} \right] \prod_{i=1}^m n_{2r_i}^{vw_i} \times \delta(2r - 2, \sum_{i=1}^m 2r_i). \quad (12)$$

We now have a system of equations which we want to solve for n_{2r}^{uv} . To make this a bit easier, we define

$$h^{uv}(z) = \sum_{r=1}^{\infty} \frac{n_{2r}^{uv}}{z^{2r}}. \quad (13)$$

Equation (11) shows why this is helpful.

Next, we substitute equation (12) into equation (13) and directly sum over r . We obtain

$$h^{uv}(z) = \frac{1}{z^2} \sum_{m=1}^{\infty} \left[\sum_{\substack{w_1 \in N_v \\ w_1 \neq u}} \cdots \sum_{\substack{w_m \in N_v \\ w_m \neq u}} \right] \prod_{i=1}^m \sum_{r_i=1}^{\infty} \frac{n_{2r_i}^{vw_i}}{z^{2r_i}}. \quad (14)$$

We can write the last part in terms of equation (13),

$$h^{uv}(z) = \frac{1}{z^2} \sum_{m=1}^{\infty} \prod_{i=1}^m \sum_{\substack{w_i \in N_v \\ w_i \neq u}} h^{vw_i}(z) = \frac{1}{z^2} \sum_{m=1}^{\infty} \left[\sum_{\substack{w \in N_v \\ w \neq u}} h^{vw}(z) \right]^m. \quad (15)$$

Since we know that $\sum_{k=1}^{\infty} a^k = \frac{1}{1-a}$ when $|a| < 1$, we can rewrite equation (15) and get

$$h^{uv}(z) = \frac{1}{z^2} \frac{1}{1 - \sum_{\substack{w \in N_v \\ w \neq u}} h^{vw}(z)}. \quad (16)$$

We now know how to calculate the number of closed walks of length $2r$ when starting in node u . The first step in the closed walks is set to be traversing the link from node u to node v . For computing the spectral density from equation (11), we need to know this number n_{2r}^{uv} , but now without the condition of the first traversed link. We want to know the number of closed walks of length $2r$ when starting and ending in node u : n_{2r}^u . The derivation of the equation for n_{2r}^u is similar to the derivation of the equation for n_{2r}^{uv} . Therefore, we will not explain this in detail.

We obtain

$$n_{2r}^u = \sum_{m=1}^{\infty} \left[\sum_{v_1 \in N_u} \cdots \sum_{v_m \in N_u} \right] \left[\sum_{r_1=1}^{\infty} \cdots \sum_{r_m=1}^{\infty} \right] \prod_{i=1}^m n_{2r_i}^{uv_i} \times \delta(2r, \sum_{i=1}^m 2r_i). \quad (17)$$

To solve this system for the numbers n_{2r}^u , we define

$$g^u(z) = \sum_{r=1}^{\infty} \frac{n_{2r}^u}{z^{2r}}. \quad (18)$$

We proceed the same way as above, so we substitute (17) into (18), sum over r and rewrite the equation. We obtain

$$g^u(z) = \sum_{m=1}^{\infty} \left[\sum_{v_1 \in N_u} \cdots \sum_{v_m \in N_u} \right] \prod_{i=1}^m \sum_{r_i=1}^{\infty} \frac{n_{2r_i}^{uv_i}}{z^{2r_i}} \quad (19)$$

$$= \sum_{m=1}^{\infty} \prod_{i=1}^m \sum_{v_i \in N_u} h^{uv_i}(z) \quad (20)$$

$$= \sum_{m=1}^{\infty} \left[\sum_{v \in N_u} h^{uv}(z) \right]^m. \quad (21)$$

By rewriting this series, we get

$$g^u(z) = \frac{1}{1 - \sum_{v \in N_u} h^{uv}(z)}. \quad (22)$$

We now derived a way to calculate the number of closed walks of length $2r$ that start and end in node u . Note that we already divided this by z^{2r} and summed this for every possible r (look at equation (18)). We can now calculate the spectral density by summing (22) over every possible node u and substituting this into equation (11), this gives us

$$\rho(z) = -\frac{1}{n\pi z} \sum_{u=1}^n g^u(z) = -\frac{1}{n\pi z} \sum_{u=1}^n \frac{1}{1 - \sum_{v \in N_u} h^{uv}(z)}. \quad (23)$$

We now conclude that we can calculate the spectral density of z by first computing the value of $h^{uv}(z)$ from equation (16), then substituting this into equation (23). Remember that we can calculate the spectral density of x by taking the limiting value of the imaginary part when ϵ goes to zero from above.

The equation (16) can be solved numerically for $h^{uv}(z)$ for every value of z . However, we derived this equation as a tool for the next section, in which we will approximate the spectral density for the specific case of the configuration model. Therefore, solving equation (16) directly is not our main focus. For that reason, we will not discuss this in more detail here.

3.3 Approximation for the configuration model

In this section, we derive an approximation for the case of the configuration model. We can recall from section 2.1.1 that the configuration model generates locally treelike networks. Therefore, we can use the formulas derived in the previous section to make an approximation.

A scatter plot of the values of $h^{uv}(z)$ for a given z shows that these values are divided into different, non-overlapping point clouds. Each point cloud contains the values of $h^{uv}(z)$ belonging to the nodes v of a certain degree. We therefore can approximate the values of $h^{uv}(z)$ by taking the value of the mean of the point cloud corresponding to the degree of node v . This makes the value $h^{uv}(z)$ a function that only depends on the degree of node v . In Chapter 4, we will see if this approximation is accurate. First, we derive the corresponding equations for the spectral density with this approximation.

We start by rewriting equation (16) as

$$z^2 h^{uv}(z) - z^2 h^{uv}(z) \sum_{\substack{w \in N_v \\ w \neq u}} h^{vw}(z) = 1, \quad (24)$$

or

$$z^2 h^{uv}(z) = 1 + z^2 h^{uv}(z) \sum_{\substack{w \in N_v \\ w \neq u}} h^{vw}(z). \quad (25)$$

Let us define $h_k(z)$ as the mean value of a point cloud. Here, k is the degree of node v for the points $h^{uv}(z)$ in the point cloud. If we now substitute our approximation $h_k(z)$ for $h^{uv}(z)$, we have to make some changes in our formula. Note that the u disappears, but we still have the condition $w \neq u$. Therefore, we have to sum over all possible $u \in N_v$. However, this gives us a sum of the values $\sum_{\substack{w \in N_v \\ w \neq u}} h^{vw}(z)$, instead of only one single value $\sum_{\substack{w \in N_v \\ w \neq u}} h^{vw}(z)$. Thus, we have to average over all the edges between u and v . As stated before, v has degree k , so we have $nk p_k$ of those edges. Furthermore, we have to compute these values when the degree of v equals k , for all possible values of k . We obtain for equation (25)

$$z^2 h_k(z) = 1 + z^2 h_k(z) \frac{1}{nk p_k} \sum_{v: k_v=k} \sum_{u \in N_v} \sum_{\substack{w \in N_v \\ w \neq u}} h^{vw}(z) \quad (26)$$

$$= 1 + z^2 h_k(z) \frac{k-1}{nk p_k} \sum_{v: k_v=k} \sum_{w \in N_v} h^{vw}(z). \quad (27)$$

The $k-1$ in equation (27) comes from combining the sum over $u \in N_v$ and the condition $w \neq u$. Another property of the configuration model is that the degrees of neighbouring nodes are uncorrelated.

This means that the degrees of v and w are uncorrelated. Therefore, the average of $h^{vw}(z)$ when summing over $w \in N_v$ does not depend on k . For $n \rightarrow \infty$ this average just equals the average of $h^{vw}(z)$ in the network as a whole. We define this average in the network as a whole to be $h(z)$. We can thus substitute $h(z)$ for $\frac{1}{nkp_k} \sum_{v:k_v=k} \sum_{w \in N_v} h^{vw}(z)$. Equation (27) then becomes

$$z^2 h_k(z) = 1 + z^2 h_k(z)(k-1)h(z). \quad (28)$$

Or, equivalently,

$$h_k(z) = \frac{1}{z^2} \frac{1}{1 - (k-1)h(z)}. \quad (29)$$

Now notice that the average of $h_k(z)$ (the mean of the point cloud corresponding to nodes v of degree k) over all k equals the average in the network as a whole, $h(z)$. We have to make a remark here. The degree k of node v does not follow the degree distribution p_k of the network. This is because our node v is per definition reached by following the link from u to v . The degree k thus follows the excess degree distribution as defined in section 2.1.1. Note, however, that our node v has a degree k in total, so its excess degree is $k-1$. The excess degree distribution of node v thus becomes,

$$q_{k-1} = \frac{kp_k}{\langle k \rangle}. \quad (30)$$

We conclude that the average message $h(z)$ is given by the following

$$h(z) = \sum_{k=1}^{\infty} q_{k-1} h_k(z). \quad (31)$$

When we substitute equation (29) into this, we find

$$h(z) = \frac{1}{z^2} \sum_{k=1}^{\infty} \frac{q_{k-1}}{1 - (k-1)h(z)}. \quad (32)$$

By shifting the index from k to $k+1$ we obtain

$$h(z) = \frac{1}{z^2} \sum_{k=0}^{\infty} \frac{q_k}{1 - kh(z)}. \quad (33)$$

To compute the spectral density, we now make a similar approximation based on the degree of nodes u for $g^u(z)$ in equation (22). We approximate $g^u(z)$ by $g_k(z)$. Here, $g_k(z)$ is the mean of $g^u(z)$ over all nodes u with degree k . If we substitute this into equation (22), u again disappears, but the condition $v \in N_u$ stays. We correct this by summing over all nodes u that have degree k , and averaging over all nodes of degree k . There are np_k nodes of degree k in the network. Equation (22) thus becomes

$$g_k(z) - g_k(z) \frac{1}{np_k} \sum_{u:k_u=k} \sum_{v \in N_u} h^{uv}(z) = 1. \quad (34)$$

Note that we can replace the last part $\frac{1}{np_k} \sum_{u:k_u=k} \sum_{v \in N_u} h^{uv}(z)$ with $kh(z)$, because we have a large network where $n \rightarrow \infty$. To get a better understanding of this, we can look back at how we defined $h(z)$ before.

We thus obtain

$$g_k(z) - g_k(z)kh(z) = 1 \quad (35)$$

Or, similarly,

$$g_k(z) = \frac{1}{1 - kh(z)}. \quad (36)$$

We can now derive the formula for the spectral density by substituting this into equation (23), we get

$$\rho(z) = -\frac{1}{n\pi z} \sum_{u=1}^n g^u(z) \quad (37)$$

$$= -\frac{1}{n\pi z} \sum_{k=0}^{\infty} np_k g_k(z) \quad (38)$$

Note that in (38), we started the sum at $k = 0$ instead of $k = 1$, as we do in equation (31). We make this change because k now represents the degree of node u instead of the degree of node v , as in equation (31). Since node v is reached by following an edge, its degree is always at least one. For node u , this is not the case, so the sum in (38) starts at $k = 0$.

If we now substitute (36) into (38), we find

$$\rho(z) = -\frac{1}{\pi z} \sum_{k=0}^{\infty} \frac{p_k}{1 - kh(z)} \quad (39)$$

We now conclude that we can calculate the approximation of the spectral density of z by solving equation (33) and substituting this into equation (39). We recall that we can calculate the spectral density of x by taking the limiting value of the imaginary part when ϵ goes to zero from above. Note that we can solve equation (33) for every given z by iteration, with a given sufficient starting point.

In this method we approximated several values by their averages. Keep in mind that we work with a network where n is large, and that the averages become more accurate when the size of the network grows.

3.3.1 Comparison between the message passing method and the approximation

In summary, the message passing method for calculating the spectral density from section 3.2 consists of equations (16) and (23), whereas the approximation consists of equations (33) and (39). What strikes immediately when comparing the equations is that the approximation only depends on the degree distribution, whereas the message passing method depends on the size and structure of the network.

Another interesting difference between the two methods is that, to find the spectral density for a given z in (23), we need to find solutions for $h^{uv}(z)$ for multiple values of u and v . We therefore have to iterate multiple equations. Since for every u , we need to iterate an equation for each neighbour of u , this gives a total of $\sum_{i=1}^n k_i = 2m$ equations. The number of equations that we have to iterate thus equals twice the number of edges in the network. However, to find the spectral density for a given z from equation (39), we only need to iterate one equation. Since we have a network where $n \rightarrow \infty$, iterating equation (16) directly will be very difficult. We are not even certain if this converges.

The last remark we make in this section is that, because of the more complicated message passing method, computing the spectral density of z from equations (16) and (23) will need more running time than computing it from equations (33) and (39).

3.3.2 Analytic solutions for $h(z)$

As discussed, we can solve equation (33) by iteration. However, for some specific simple degree distributions, we can derive the solution for $h(z)$ analytically.

We first look at the solutions for a network with nodes of degree a . This means that $p_a = 1$ and the average degree is a . For this example, equation (33) becomes

$$h(z) = \frac{1}{z^2} \frac{1}{1 - (a-1)h(z)}. \quad (40)$$

If we rewrite this, we obtain

$$0 = \frac{1}{z^2} - h(z) + (a-1)h(z)^2. \quad (41)$$

We can solve this quadratic equation and find

$$h(z) = \frac{1 \pm \sqrt{1 - \frac{4(a-1)}{z^2}}}{2(a-1)}. \quad (42)$$

If we substitute this into equation (39), we get

$$\rho(z) = \frac{1}{\pi z (ah(z) - 1)} \quad (43)$$

To obtain the approximation for the spectral density for x , we need to take the limiting value of the imaginary part when ϵ goes to zero from above of (43). We obtain this approximation to be

$$\rho(x) = \frac{a}{2\pi} \frac{\sqrt{4(a-1) - x^2}}{a^2 - x^2}. \quad (44)$$

Important to note is that this approximation is in fact the exact spectral density for x . Since all nodes have the same degree, the scatter plot consists of one single point cloud. The neighbourhood of every node is exactly the same, so $h^{uv}(z)$ has the same value for all possible combinations of u and v . The point cloud thus consists of only one single point. Hence, we approximate the mean of the point cloud (and thus the average of $h^{uv}(z)$ in the network as a whole) by the exact value of every message $h^{uv}(z)$.

The second network we are going to look at is the one where we have nodes of two different degrees, a and b . We find equation (33) to be

$$h(z) = \frac{1}{z^2(ap_a + bp_b)} \left[\frac{ap_a}{1 - (a-1)h(z)} + \frac{bp_b}{1 - (b-1)h(z)} \right]. \quad (45)$$

To solve this equation for $h(z)$, we need to solve a cubic equation. This will give us a complicated solution. We therefore look at the case where we have nodes of degrees $a = 1$ and b arbitrary, where $a \neq b$ and $ab \neq 0$. In this case, equation (33) will be

$$h(z) = \frac{1}{z^2(p_a + bp_b)} \left[p_a + \frac{bp_b}{1 - (b-1)h(z)} \right]. \quad (46)$$

If we now substitute $p_a = 1 - p_b$, we obtain

$$h(z) = \frac{1}{z^2(1 + (b-1)p_b)} \left[1 - p_b + \frac{bp_b}{1 - (b-1)h(z)} \right]. \quad (47)$$

Rewriting this equation gives us a quadratic equation, which can be solved. However, it gives us very complicated solutions. We therefore do not proceed with this case.

Note that for networks other than the one where we only have nodes of one degree, analytic solutions for $h(z)$ are complicated to derive.

3.3.3 Edges of the spectrum

What is interesting about the spectra of the network's adjacency matrix is the position of the edges of these spectra. As already stated in the introduction, these spectrum bounds can be very useful to, for example, make statements about the stability of the network ([5], p. 698).

We take a look at equation (33) to gain more understanding about the edges of the spectrum. From this equation and a given z , we find one or multiple solutions for $h(z)$. These solutions can be either real or complex. Note that real solutions for $h(z)$ give us a spectral density of zero, since we take the imaginary part of equation (39). We thus need to have at least one complex valued solution to obtain a nonzero spectral density. In particular, we are interested in when a complex solution first appears. Then, we go from a zero spectral density to a nonzero spectral density. This gives us exactly the edges of the spectrum, in which we are interested.

To see when a complex solution of equation (33) first occurs, we take a look at figures 4, 5 and 6. Here, we plotted the right and left hand side of

$$hz^2 = \sum_{k=0}^{\infty} \frac{q_k}{1 - kh} \quad (48)$$

for different values of z and an arbitrary network. Note that the right hand side has poles at values of h where $h = \frac{1}{k}$. In this situation, k are all the possible nonzero values of the excess degree in the network. We see that the left hand side is just a linear equation which goes through the origin and has a slope of z^2 .

We saw in section 3.3.2 that equation (33) is a polynomial of degree $m + 1$. Here, m is the number of distinct nonzero values of the excess degree. Therefore, equation (48) gives us $m + 1$ solutions for $h(z)$. Note that the number of distinct nonzero values of the excess degree m corresponds to the number of poles.

As stated earlier, we are looking for a z for which a complex solution first appears. If we now take a look at figure 4, we notice that a large z gives us $m + 1$ real solutions. For this specific network in the figure, we have four poles. The blue line intersects with the red line five times, so there are five real solutions. Since all $m + 1$ solutions are real, this value of z gives us a spectral density of zero. Next, we look at figure 5. We conclude that if the blue line is tangent to the leftmost part of the red lines, we have m real solutions. For this particular network with four poles, we also have four real solutions. So in this case

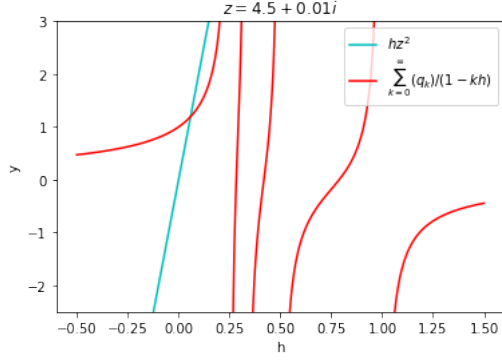


Figure 4: Real solutions for $h(z)$ when $z = 4.5 + 0.01i$

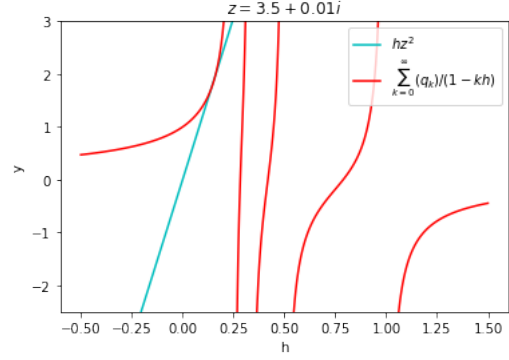


Figure 5: Real solutions for $h(z)$ when $z = 3.5 + 0.01i$

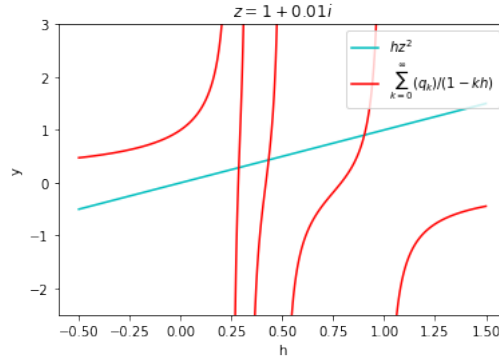


Figure 6: Real solutions for $h(z)$ when $z = 1 + 0.01i$

where we have m real solutions, we also have one complex solution. We are interested in the value of z for which a complex solution first appears. We conclude that this happens when the blue line is tangent to the leftmost part of the red lines. Lastly, we look at figure 6. Here z is relatively small. We obtain $m - 1$ real solutions. Therefore, we have two remaining complex solutions. These give us a nonzero spectral density.

It is interesting to see if we can say something about the z for which the left hand side is of (48) is tangent to the leftmost part of the right hand side of (48). For this z , we are at the edges of the spectrum. Note that for this value of z , the tangent point lays above the point of intersection of the right hand side and the y -axis. This point of intersection is at $y = \sum_{k=0}^{\infty} q_k = 1$. We find that for the desired value of z holds

$$z^2 > \frac{1}{h}. \quad (49)$$

Furthermore, the tangent point is located left from the first pole. This pole is at $\frac{1}{K}$, where K is the largest value of the excess degree. So for the tangent point holds that $h < \frac{1}{K}$. We find our interested z to satisfy

$$z^2 > \frac{1}{\frac{1}{K}} = K. \quad (50)$$

The edges of the spectrum are thus at the locations where z satisfies $z \geq \sqrt{K}$ (upper edge) and $z \leq -\sqrt{K}$ (lower edge). Note that these bounds imply that networks with unbounded degree distributions have a spectrum without edges, because z diverges as K diverges.

Next, we are going to try to derive better bounds on the edges of the spectrum. We therefore start with computing a lower bound on the right hand side of (48), where $h < \frac{1}{K}$ (so for the leftmost part).

We obtain

$$\sum_{k=0}^{\infty} \frac{q_k}{1-kh} = \frac{q_K}{1-Kh} + \sum_{k \neq K} \frac{q_k}{1-kh} \quad (51)$$

$$\geq \frac{q_K}{1-Kh} + \sum_{k \neq K} q_k \quad (52)$$

$$= \frac{q_K}{1-Kh} + 1 - q_K. \quad (53)$$

There is a tangent point if this lower bound and the left hand side of (48) have exactly one intersection point. So when

$$hz^2 = \frac{q_K}{1-Kh} + 1 - q_K \quad (54)$$

has one solution for h . We can rewrite this equation as the quadratic equation

$$Kz^2h^2 - (-q_KK + K + z^2)h + 1 = 0. \quad (55)$$

This gives us one solution for h if the discriminant equals zero, so if

$$(-q_KK + K + z^2)^2 - 4Kz^2 = 0. \quad (56)$$

This gives us the quadratic equation

$$z^4 - 2(q_K + 1)Kz^2 + ((1 - q_K)K)^2 = 0, \quad (57)$$

which we can solve for z^2 . This solution gives us a better lower bound on the edge of the eigenvalue spectrum. We find this quadratic equation to have the solution

$$z^2 = (\sqrt{q_K} + 1)^2 K. \quad (58)$$

We now conclude that this lower bound on the right hand side of (48) gives us a lower bound of z on the edge of the spectrum

$$z \geq (\sqrt{q_K} + 1)\sqrt{K}. \quad (59)$$

We now derive an upper bound on the edge of the eigenvalue spectrum. We do this by using the value of h at the intersection between the lower bound (53) and the left hand side of (48). We already started to derive this above. We found that there is one solution for h if $z^2 = (\sqrt{q_K} + 1)^2 K$. The value of h then becomes

$$h = \frac{K(1 - q_K) + z^2}{2Kz^2} \quad (60)$$

$$= \frac{1}{(\sqrt{q_K} + 1)K}. \quad (61)$$

We use this value of h as this value is quite close to the true value of h for the intersection at the tangent point, but smaller because of the lower bound. We also once again assume that we are in the region $h < \frac{1}{K}$, because as concluded before, that is where the tangent point is located. If we rewrite (48) as

$$z^2 = \frac{1}{h} \sum_{k=0}^{\infty} \frac{q_k}{1-kh} \quad (62)$$

and use the derived value of h , we obtain

$$z^2 \leq (\sqrt{q_K} + 1)K \sum_{k=0}^{\infty} \frac{q_k}{1 - \frac{k}{(\sqrt{q_K} + 1)K}}. \quad (63)$$

We now have derived an improved lower bound (59) and upper bound (63) on the edge of the spectrum using the approximation. We will look at some example networks to see if these bounds are correct in the next chapter.

4 Results

In this chapter we look at the spectra of different networks generated by the configuration model and analyse the performance of our approximation derived in the previous chapter. To generate the figures below, we produced a code in Python which computes values for the equations derived in the previous chapter and uses them to generate a plot of the spectral density. This script can be found in Appendix A. In all of the plots, we used $\epsilon = 0.01$. In the generated figures, the histogram plotted in blue shows the distribution of the eigenvalues calculated directly from the generated network with $n = 10000$ nodes. We plotted the spectral density derived from equation (9) in green. The red line shows the approximation of the spectral density obtained from combining equations (33) and (39). For completeness, we also plotted the semicircle distribution in orange.

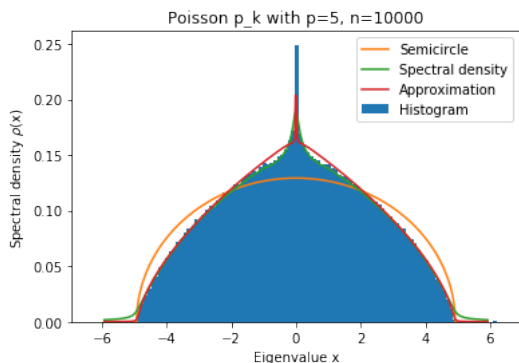


Figure 7: Poisson degree distribution with mean 5

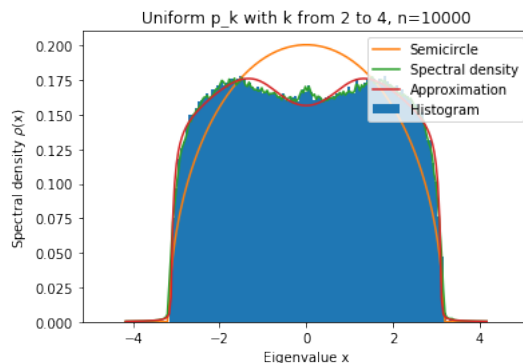


Figure 8: Nodes of degree 2, 3, 4, with equal probability

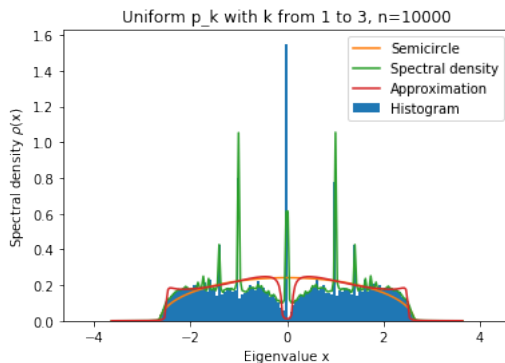


Figure 9: Nodes of degree 1, 2, 3, with equal probability

In figure 7, we see the spectrum of a network with a Poisson degree distribution with mean 5 generated by the configuration model. Note that the Poisson distribution is infinitely divisible, so the elements of the adjacency matrix of the network are independent and identically distributed random variables. Therefore, we may expect this spectrum to be distributed according to the semicircular law of Wigner. However, the Poisson degree distribution has a small mean of five. Therefore, this network is sparse and, as stated in the introduction, its spectrum thus does not follow the semicircle law. This is in line with what we see in figure 7. The orange line of the semicircle distribution differs a lot from the eigenvalue spectrum.

We see that the approximation based on degrees does a good job on approximating the distribution of the eigenvalues in this case. The edges are derived correctly, but closer to zero the approximation differs a bit from the directly calculated spectrum. However, the approximation does capture the peak at $x = 0$. Important to note is that the Poisson degree distribution is not an appropriate distribution if we want to describe real-world networks, as already stated in Chapter 2.

Next, we look at figure 8. Here, we have a network with nodes of degree 2, 3 and 4, all present with the same probability. This network has an average degree of 3, so it is very sparse. The approximation again looks quite good. It includes the right edges of the spectrum. But, in contrast to what we saw before, at

$x = 0$ the approximation does seem to differ from the directly calculated spectrum.

Lastly, we look at figure 9, which includes a network with nodes of degree 1, 2 and 3. Once again, with equal probability. This network is extremely sparse, its average degree is only 2. The spectrum appears to be full of spikes. We directly see that the approximation does not capture these spikes. In the next chapter, we will pay more attention to this. Furthermore, as seen in the previous figure 8, at $x = 0$ the approximation deviates a lot from the directly calculated spectrum. However, the edges of the spectrum are still adequately covered by the approximation.

In section 3.3.3, we derived bounds (59) and (63) on the edges of the spectrum. We now computed these bounds for the specific networks in figures 8 and 9. For the network in figure 8, we found that the edge of the spectrum falls in the interval $2.887 \leq z \leq 3.118$. In the figure, we observe that this narrow interval gives good bounds on the position of the edge. However, the upper bound (63) seems a little too small, since it is located a bit left from the edge of the true spectrum. For the network in figure 9, we found the edge to be in the interval $2.414 \leq z \leq 2.510$. We conclude the same as we did for the previous network. Namely that the small interval seems correct but the upper bound is located left from the true edge.

5 Giant component

As discussed in section 2.1.1, the configuration model generates networks with a giant component if and only if

$$\text{average excess degree} > 1. \quad (64)$$

It is fascinating to see what the effect of having a giant component is on the spectrum. To observe this, we first derive conditions on networks with a specific degree distribution on having a giant component.

5.1 Network with nodes of degree a

When we have a network that only consists of nodes of degree a , the average excess degree is

$$\sum_{k=0}^{\infty} kq_k = \frac{(a-1)ap_a}{ap_a} = a-1. \quad (65)$$

We conclude that this network has a giant component if and only if

$$a > 2. \quad (66)$$

5.1.1 Connection to the spectrum

In the previous section, we concluded that when a changes from 2 to 3, we go from not having a giant component to having a giant component. It is interesting to see what happens to the spectrum when a changes from 2 to 3.

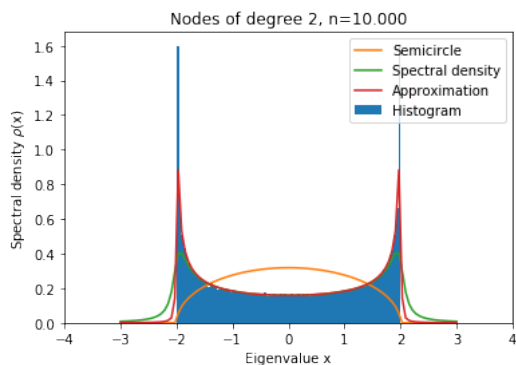


Figure 10: Spectrum for $a = 2$

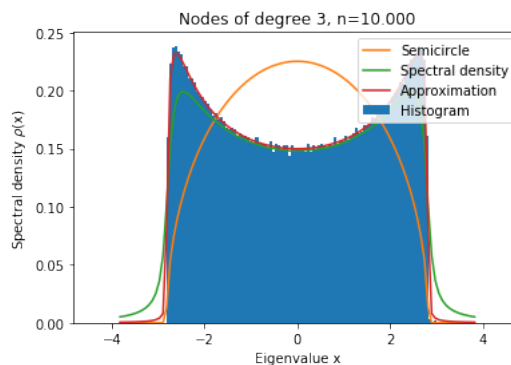


Figure 11: Spectrum for $a = 3$

In figure 10, we see the spectrum of $a = 2$. In figure 11, we see the spectrum of $a = 3$. What strikes us the most is that in the first case where $a = 2$, eigenvalues at the bounds of the spectrum have a very high density. Note that the scaling of the axes is different in both figures. We can explain this high density by using equation (44) from section 3.3.2. For $a = 2$, this equation equals

$$\rho(x) = \frac{1}{\pi\sqrt{4-x^2}}. \quad (67)$$

For $a = 3$, (44) equals

$$\rho(x) = \frac{3}{2\pi} \frac{\sqrt{8-x^2}}{9-x^2}. \quad (68)$$

We conclude that the shape of the graph of equation (67) looks like an upward opening parabola, while (68) has a trough in the middle and a peak on each side of this through. However, in contrast to an upward opening parabola, the value of (68) decreases to zero at the edges of the domain. For values of $a > 3$ (so when there is a giant component), we see that the spectrum keeps having this second shape, similar to figure 11.

5.2 Network with nodes of two different degrees

For this second network, we have nodes of two different degrees a and b (thus $a \neq b$ and $ab \neq 0$). The probability of a node to have degree a is p_a and the probability to have degree b is p_b . Note that $p_a + p_b = 1$. The average excess degree of this network is

$$\sum_{k=0}^{\infty} kq_k = \frac{(a-1)ap_a + (b-1)bp_b}{ap_a + bp_b}. \quad (69)$$

Note that we can now substitute $p_a = 1 - p_b$. We conclude that this network has a giant component if and only if

$$\frac{(a^2 - a)(1 - p_b) + (b^2 - b)p_b}{a(1 - p_b) + bp_b} > 1. \quad (70)$$

5.2.1 Network with nodes of degree 1 and b

If we fix $a = 1$ and leave b arbitrary, we conclude from (70) that this network has a giant component if and only if

$$\frac{(b^2 - b)p_b}{1 + (b - 1)p_b} > 1. \quad (71)$$

Or, equivalently,

$$p_b > \frac{1}{b^2 - 2b + 1} = \frac{1}{(b - 1)^2}. \quad (72)$$

Note that for $b = 2$, equation (72) equals $p_b > 1$. Since p_b is a probability, this can never happen. So a network with two different degrees $a = 1$ and $b = 2$ cannot have a giant component. For $b > 2$, a giant component can occur, depending on the value of p_b .

5.2.2 Connection to the spectrum

In the previous section we concluded that, when $a = 1$, it depends on b and p_b whether the network has a giant component. We now take a look at a network with nodes of degrees $a = 1$ and $b = 3$. We want to see if something interesting happens to the spectrum if we vary p_b . If we take a look at figures 12 and 13, where we plotted this network with the values $p_b = \frac{1}{5}$ and $p_b = \frac{1}{2}$, we can conclude from equation (72) that when $p_b = \frac{1}{5}$, the network does not have a giant component whereas for $p_b = \frac{1}{2}$ the network does have one.

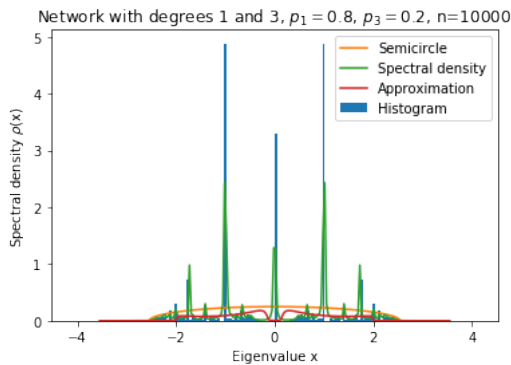


Figure 12: Network without a giant component

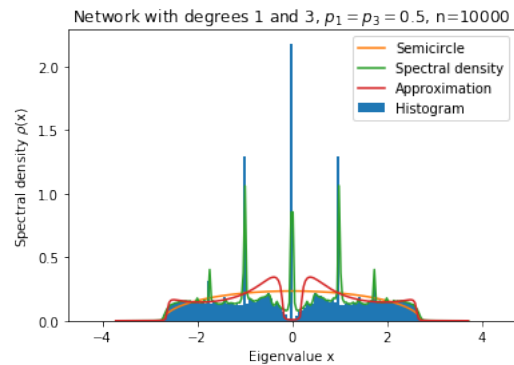


Figure 13: Network with a giant component

We observe that the right plot is more dense, whereas the left plot has higher spikes. Note that the scaling of the y-axis is different in both figures. In figure 12, the spikes around $x = \pm 1$ are higher than the spike at $x = 0$, while in figure 13 the opposite holds. In both cases, the network is extremely sparse. As we have already seen in the previous chapter, our approximation does not work well on these networks.

5.3 Spikes

As we already saw in earlier examples, the spectrum appears to have a lot of spikes when the network is extremely sparse. We also saw that our approximation failed to capture these spikes.

We can now ask oneself where these spikes come from. To analyse this problem, we look at the adjacency matrix of the undirected network. If we have a network that consists of multiple components, this adjacency matrix \mathbf{A} is a block diagonal matrix. The adjacency matrices \mathbf{A}_i of the separate components are the square blocks on the diagonal of the adjacency matrix \mathbf{A} of the whole network. The rest of the adjacency matrix \mathbf{A} consist of zeros. Since

$$\det(\mathbf{A} - \lambda\mathbf{I}) = \det(\mathbf{A}_1 - \lambda\mathbf{I})\det(\mathbf{A}_2 - \lambda\mathbf{I}) \cdots \det(\mathbf{A}_n - \lambda\mathbf{I}),$$

the eigenvalues of \mathbf{A} are just the combination of the eigenvalues of the diagonal elements \mathbf{A}_i . This explains where the peaks at $x = \pm 1$ come from, namely from the separate components of size 2. To see this, we look at the block diagonal element $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, which is the adjacency matrix of a component of size 2. We see that the eigenvalues of this matrix are $x = 1$ and $x = -1$. Remember from the previous section that the example network without a giant component did have higher peaks at $x = \pm 1$ than the network with a giant component. This thus means that the network without a giant component has more separate components of size 2. We conclude that the spikes in spectra of extremely sparse networks occur because of the existence of multiple small components.

To see why our approximation fails to capture these spikes, we take a look at equation (13). We concluded that these spikes are due to the existence of various small components in the network. We expect the following reasoning to explain why the approximation does not capture the spikes. The value of h^{uv} of a node v in a small component is smaller than the value for a node v in the giant component. In a small component, we have way less possible closed walks n_{2r}^{uv} of longer lengths than in a giant component. Our approximation however does not distinguish between nodes v in a small component and in a giant component. We approximate h^{uv} by the mean of the point cloud to which node v belongs, based on its degree. We expect a node v of a certain degree in a small component to have a value of h^{uv} , which is less then the value of h^{uv} for node v of that degree in a giant component. With that, if we have a giant component, more nodes belong to the giant component than to a small component. Therefore, there will be more high values of h^{uv} belonging to a certain degree of node v . The approximation by the mean of the point cloud thus is not a good one for nodes v in a small component. To see if this reasoning is true, we would like to plot the values of h^{uv} for a given z in the complex plane. We expect the point clouds in this plot to have multiple outliers for an extremely sparse network, corresponding to the nodes in small components.

5.4 Spectrum of the giant component

From the previous section, we concluded that the spikes in the spectrum are due to the existence of numerous small components. We therefore expect that, if we look at the spectrum of the giant component alone, these spikes are not present. To see if this assumption is correct, we look at figures 14 and 15 below.

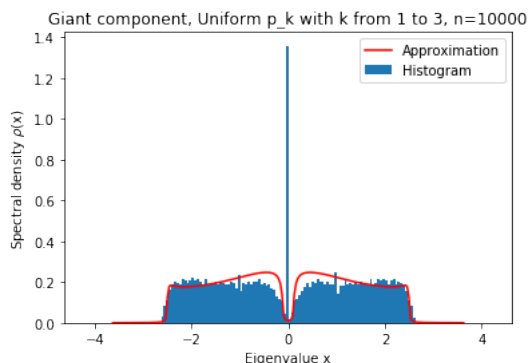


Figure 14: Spectrum of the giant component

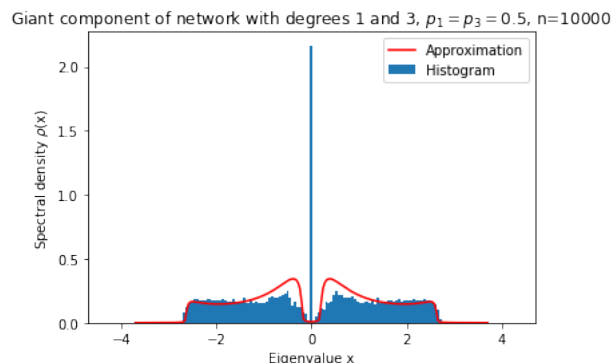


Figure 15: Spectrum of the giant component

In figure 14 we see the spectrum of the giant component of a network generated by the configuration model with nodes of degree 1, 2 and 3 (each with equal probability). Note that this is the same degree distribution as the network we generated and plotted the spectrum for in figure 9. We immediately notice that the peaks disappeared when we compare figures 9 and 14, except for the peak at $x = 0$. The rest of

the spectrum looks the same.

In figure 15 we plotted the spectrum of the giant component for a network generated by the configuration model with nodes of degree 1 and 3 (with equal probability). Note that network has the same degree distribution as the network we analysed in section 5.2.2. If we compare figures 13 and 15, we notice that the spikes indeed have disappeared, once again except the spike at $x = 0$. Furthermore, the rest of the spectrum appears to be quite the same in both figures.

We thus conclude that the peak at $x = 0$ does not come from the existence of a small component, since it also appears in the spectrum of the giant component alone. Note that the density of the eigenvalue $x = 0$ is the same in figures 13 and 15. So the existence of small components really has nothing to do with the peak at $x = 0$. The peaks at $x = 0$ in figures 9 and 14 do differ in height a bit. This can be explained by the fact that the plots are generated for two different networks, which have the same degree distribution and are generated by the configuration model.

In both figures 14 and 15, we plotted the approximation of the spectrum of the whole network in red. We can see that this approximation does not capture the spike at $x = 0$. In the previous section, we discussed why the approximation fails to capture spikes that occur in extremely sparse networks due to the presence of small components. However, as concluded, this spike at $x = 0$ is not related to the existence of a small component. Thus, at this point, we have not yet found an explanation to where this spike at $x = 0$ comes from, and to why our approximation fails to capture it.

6 Conclusion and discussion

As stated in our introduction, our goal of this thesis was to analyse the spectra of sparse random networks. We did this by deriving a degree based approximation for the spectral density of networks generated by the configuration model. In section 3.3, this approximation, which applies in the limit of large network size, can be found. In Chapter 4 we looked at some examples of network spectra on which we applied the approximation. We conclude that the approximation works well on the edges of the spectrum. However, when the network becomes extremely sparse, the approximation differs from the directly calculated network spectrum. It especially shows deviations from the true spectrum around $x = 0$. We also looked at bounds of the edge of the spectrum, which we derived in section 3.3.3. We conclude that these bounds give a narrow interval, which captures the edge of the spectrum well. Nonetheless, the upper bound seems too small in all considered examples.

In Chapter 5, we stated that the approximation does not distinguish between nodes in small components and in a giant component, which is why spikes in the spectrum are not well captured by the approximation. To see if this reasoning is true, we would like to look at a plot of the values of h^{uv} in an extremely sparse network for a given z . This will be interesting to look at in further research.

These spikes occur due to the existence of numerous small components in extremely sparse networks. We thus understand that a better approximation, in which we make this difference between nodes in small components and those in a giant component, will be possible. It will be interesting to see in future projects whether such an approximation gives the desired results.

Another important conclusion we derived in Chapter 5 is that the peaks located at $x = 0$, that occur in extremely sparse networks, are not related to the existence of small components. Interesting further research will be to analyse where these spikes come from and why they are not captured by our approximation.

References

- [1] R. A. Adams and C. Essex. *Calculus: A Complete Course*. Pearson, ninth edition, 2018.
- [2] George T. Cantwell and M. E. J. Newman. Message passing on networks with loops. *Proceedings of the National Academy of Sciences*, 116(47):23398–23403, 2019.
- [3] László Erdős, Antti Knowles, Horng-Tzer Yau, and Jun Yin. Spectral statistics of erdős–rényi graphs i: Local semicircle law. *The Annals of Probability*, 41(3B):2279–2375, 2013.
- [4] Illés J. Farkas, Imre Derényi, Albert-László Barabási, and Tamás Vicsek. Spectra of “real-world” graphs: Beyond the semicircle law. *Physical Review E*, 64(2), 2001.
- [5] M. E. J. Newman. *Networks*. Oxford University Press, Oxford, second edition, 2018.
- [6] M. E. J. Newman, Xiao Zhang, and Raj Rao Nadakuditi. Spectra of random networks with arbitrary degrees. *Physical Review E*, 99(4), 2019.
- [7] T Tao. *Topics in random matrix theory*. American Mathematical Society, 2012. <https://terrytao.files.wordpress.com/2011/02/matrix-book.pdf> (last accessed on 30-10-2020).
- [8] Eugene P. Wigner. On the distribution of the roots of certain symmetric matrices. *Annals of Mathematics*, 67(2):325–327, 1958.
- [9] Eugene P. Wigner. Random matrices in physics. *SIAM Review*, 9(1):1–23, 1967.

Logo on title page: <https://www.uu.nl/organisatie/huisstijl/downloads/logo> (last accessed on 14-11-2020)

A Python code

```
import math
2 import numpy as np
import matplotlib.pyplot as plt
4 from scipy.stats import bernoulli
from scipy.stats import poisson
6 from scipy.stats import randint
from scipy.stats import binom
8 import networkx as nx
from scipy.optimize import fsolve
10 from scipy.optimize import fminbound
from scipy.optimize import newton
12 import cmath
import sympy
14
16 # Number of nodes
n = 100
18
19 # Degree distribution parameters
distr = np.random.choice
20 p = 0.001
low = 1
22 high = 3
lijst = [5,10]
24
25 # Degree sequence (draw a sequence from the specified degree distribution)
26 # and other important settings per distribution
28
29 if distr == randint:
    deg_seq = [distr.rvs(low, high+1) for i in range(0,n)]
30     while sum(deg_seq) % 2 == 1:
        deg_seq = [distr.rvs(low, high+1) for i in range(0,n)]
32     c=distr.mean(low, high+1)
    tekst = str('Giant component, ') + str('Uniform p_k ') + str('with k from ')
34         + str(low) + str(' to ') + str(high) + str(', n=') + str(n)
    def pmf(k):
36         return distr.pmf(k, low, high+1)
38
39 elif distr == binom:
    deg_seq = [distr.rvs(n-1, p) for i in range(0,n)]
40     while sum(deg_seq) % 2 == 1:
        deg_seq = [distr.rvs(n-1, p) for i in range(0,n)]
42     c=distr.mean(n-1,p)
    tekst = str('Binomial p_k ') + str('with p=') + str(p) + str(', n=') + str(n)
44     def pmf(k):
        return distr.pmf(k, n-1, p)
46
47 elif distr == np.random.choice:
    deg_seq = [distr(lijst) for i in range(0,n)]
48     while sum(deg_seq) % 2 == 1:
        deg_seq = [distr(lijst) for i in range(0,n)]
50     t = [(1/len(lijst))*k for k in lijst]
    c = sum(t)
52     tekst = str('Giant component of network with degrees ') + str('1 and 3,')
        + str(' $p_1=p_3=0.5$') + str(', n=') + str(n)
54     def pmf(k):
        if k in lijst:
56             return 1/len(lijst)
        else:
58             return 0
60
61 else:
    deg_seq = [distr.rvs(p) for i in range(0,n)]
62     while sum(deg_seq) % 2 == 1:
        deg_seq = [distr.rvs(p) for i in range(0,n)]
64     c = distr.mean(p)
    tekst = str('Poisson p_k ') + str('with p=') + str(p) + str(', n=') + str(n)
66     def pmf(k):
        return distr.pmf(k, p)
68
69 print("Sum of the degrees is " + str(sum(deg_seq)))
70 print('c = ' + str(c))
72 print(tekst)
```

```

74 # Generate random matrix A (Configuration model)
G = nx.configuration_model(deg_seq, create_using=None, seed=None)
76
77 #remove self loops and parallel edges
78 G= nx.Graph(G)
G.remove_edges_from(G.selfloop_edges())
80
81 # Adjacency matrix
82 A = nx.adjacency_matrix(G, nodelist=None, weight=None)
84
85 # Adjacency matrix in numpy
Ad = nx.to_numpy_matrix(G)
86
87
88 # Find eigenvalues of A
L= nx.adjacency_spectrum(G, weight=None)
90
91 #Giant component
92 GC= G.subgraph(max(nx.connected_components(G), key=len))
GCA=nx.adjacency_matrix(GC, nodelist=None, weight=None)
94 GCL=nx.adjacency_spectrum(GC, weight=None)
96
97 # Semicircle
R =np.min(L)
98 print("Smallest eigenvalue is " +str(R))
print("Largest eigenvalue is " +str(np.max(L)))
100
101 def semicircle(x):
102     return (2/(np.pi*R**2))*np.sqrt(R**2-x**2)
104
105 # Spectral density
def spectral_density(x,n,L):
106     e=0.01
z=np.complex(x,e)
108     som = [1/(z-L[i]) for i in range(0, len(L))]
return -(1/(n*np.pi))*np.sum(som)
110
111 def plotsd(b, n, L):
112     u =[]
for i in b:
114         u.append(np.imag(spectral_density(i,n,L)))
return u
116
117 '''Functions for degree-based approximation'''
118
119 # Excess degree distribution
120 def excess(k):
return (((k+1)*pmf(k+1))/c)
122
123 # Equation 33 -h, so we have to find the roots of this function
124 def eq19h(h,z):
125     som = [(excess(k)/(1-k*h)) for k in range(0,n)]
# k can be at most n-1
128     return (((1/((z)**2))*np.sum(som))-h)
130
131 # Equation 39
solutionhplus = [0]
132 solutionh = [0]
def rho(x):
134     e= 0.01
z = np.complex(x,e)
136     if x>0:
hi = newton(eq19h, solutionhplus[-1], fprime=None, args=(z,))
138         solutionhplus.append(hi)
else:
140         hi = newton(eq19h, solutionh[-1], fprime=None, args=(z,))
solutionh.append(hi)
142     d = [(pmf(k)/(1-k*hi)) for k in range(0,n)]
a = (-(1/(np.pi*z))*np.sum(d))
144     #print(x, hi, np.imag(a))
return a
146

```

```

148 def plot(b):
149     l = []
150     for i in b:
151         l.append(np.imag(rho(i)))
152     return l
153
154 # Make a histogram of the spectral density
155 v = np.linspace(R-1,-R+1,200)
156 plt.hist(L, bins=200, density=True, label = 'Histogram') #plot histogram
157 plt.plot(v, semicircle(v), label = 'Semicircle') #plot semicircle
158 plt.plot(v, plotsd(v,n,L), 'g', label = 'Spectral density') #plot spectral density
159 plt.plot(v, plot(v), 'r', label = 'Approximation') #plot approximation
160 plt.xlim(R-2, -R+2)
161 plt.xlabel('Eigenvalue x')
162 plt.ylabel('Spectral density ' r'$\rho(x)$')
163 plt.title(tekst)
164 plt.legend(loc='upper right')
165 plt.show()
166
167 # Plot for deriving bounds on the edges of the spectrum
168 def links(h,x):
169     z=np.complex(x, 0.01)
170     return h*z**2
171
172 def rechts(h):
173     som = [excess(k)/(1-k*h) for k in range(0,n-1)]
174     b=sum(som)
175     return b
176
177 x=np.linspace(-0.5,1.5,100)
178 v=np.linspace(-0.5, 1/4-0.001, 100)
179 vv=np.linspace(1/4+0.001, 1/3-0.001, 100)
180 vvv=np.linspace(1/3+0.001, 1/2-0.001, 100)
181 vvvv=np.linspace(1/2+0.001, 1-0.001, 100)
182 vvvvv=np.linspace(1+0.001, 1.5, 100)
183 plt.plot(x,links(x,3.5),'c', label='$hz^2$')
184 plt.plot(v,rechts(v), 'r', label='$\sum_{k=0}^{\infty}(q.k)/(1-kh)$')
185 plt.plot(vv,rechts(vv), 'r')
186 plt.plot(vvv,rechts(vvv), 'r')
187 plt.plot(vvvv,rechts(vvvv), 'r')
188 plt.plot(vvvvv,rechts(vvvvv), 'r')
189 plt.xlabel('h')
190 plt.ylabel('y')
191 plt.ylim(-2.5, 3)
192 plt.legend(loc='upper right')
193 plt.title('$z=3.5+0.01i$')
194 plt.show()

```