

Anomaly Detection in Networks by their Laplacians' Divergence Measures

Uri Heinemann

*Advisor:
Prof. Naftali Tishby*

School of Computer Science and Engineering
The Hebrew University of Jerusalem

January 30, 2010

Abstract

Our goal in this work is to detect anomalies in networks. As networks change over time, we would like to acquire the ability to detect the time-points in which the network behaves in an anomalous way. For example, let us look at the e-mail network. The email addresses are represented by vertices, while the edges' weights are functions of the frequency of correspondence between any two addresses. The network changes over time - people start (or stop) writing to people they had previously written to. In the setting of normal activity, a new correspondence will start between people who share a common friend or are a part of the same social group. If an anomaly in the network happens, it will be characterized by changing of the social groups, i.e, some new correspondence will start between people from different social groups whereas old correspondences will stop. The object of this work is to identify such moments.

For this purpose, we define two divergence measures: the Trace and the Determinant measures. These two measures take into account only the "smaller" eigenvectors of the two Laplacians (of the networks' adjacency matrices). By projecting the subspaces (spanned by the eigenvectors) on each other we create a matrix M , the eigenvalues of which may be considered as the cosine of the angle between the two subspaces. The two measures evaluate the size of this matrix, and through this find the divergence between the two networks. After having obtained such divergence measures, we present a simple algorithm that when given a certain sensitivity threshold, detects the anomalous time-points with that sensitivity.

Contents

1	Introduction	3
2	Notation	6
3	Problem definition and the divergence measures	8
3.1	Problem definition	8
3.1.1	Pre-Processing	8
3.1.2	Problem definition 1	8
3.1.3	Problem definition 2	9
3.2	The proposed divergence measures	9
4	Why Laplacian	11
4.1	Laplacian properties	11
4.2	Matrix Laplacian vs Continuous Laplacian	14
4.3	Laplacian and electricity	15
4.4	Random walks and the Laplacian	16
4.5	Laplacian and spectral graph theory	18
5	The neglect of the eigenvalues and some eigenvectors	20
5.1	Spectral clustering	20
5.2	Why neglect the eigenvalues	20
5.3	The eigenvector of an adjacency matrix and the Fourier transform	21
5.4	Eigenvectors and diffusion	22
5.5	Diffusion maps	24
6	From projection of subspace to our divergence measures	26
6.1	The CS decomposition	26
6.2	Reasoning the divergence measures	28
6.2.1	The Determinant measure	28
6.2.2	The Trace measure	28
7	A Bound for the Trace measure	29
7.1	$AX - BX = Y$	29
7.2	A weak version of Davis-Kahan theory	30
7.3	Error bound for the Trace measure	31
8	Graph dynamics as a multivariate normal distribution	34
8.1	Node activity by Gibbs measure	34
8.2	Node activity by incident matrix	36
8.3	Kullback-Leibler divergence and maximum likelihood	39

8.4	From Kullback-Leibler divergence to our divergence measure	41
9	Our Algorithm	43
9.1	The basic method	43
9.2	The initialization and detect methods	43
9.3	getDistance method	45
10	Methods	47
10.1	The network model	47
10.1.1	Barábsi model	47
10.1.2	The small-world model	48
10.1.3	The high-cluster model	48
10.2	Sampling from graphs	48
10.3	Anomalies	49
11	Results	52
11.1	Our Divergence measures and algorithm	52
12	Conclusions & Future Work	60

Chapter 1

Introduction

Many real-world systems may be described as networks. From the cell, which may be described as chemicals connected by chemical reaction [1],[2], to global economy, a network of states connected by commerce [3]; from the brain - a network of neurons connected by axons [4], to an e-mail network where each e-mail address may be represented by a node and the links may represent the correspondence between any two e-mail addresses [5]; from the well-known social network where each person (node) is connected to its acquaintance [6], to the WWW, a network of sites connected by links from site to site [7].

Most of these networks change over time. If you examine the network at different points in time, you will not see the same connections, and sometimes - not even the same nodes. The reactions taking place inside a cell in the different stages of the cell cycle or after a mutation, are different, and hence the network will be different. The friends one has in youth are not the same as in adulthood, and their composition will probably not look the same if one starts suffering from depression, for instance; therefore the social network will be different at various times.

When trying to monitor a network, one will want to have a tool which raises an alarm each time the network behaves in an anomalous way. In all the examples above, we gave a normal and an abnormal change of the network; we will want the tool to have the ability to disregard the normal change of the networks while detecting the anomalous one. This work will try to provide such a tool. In order to do so, we will want to examine the networks and discern whether there are some features of networks which we can take advantage of for our goal.

In recent years, research of complex networks has been intensive [8]. Three features have been found in real-life networks - the small-world phenomenon, scale-free degree distribution and clustering coefficient. Small-world phenomenon indicates that the average number of links necessary to be made in order to pass from one node to another (path length) is small compared to the network size (logarithmic) [9]. A degree of a node is the number of neighbors it has. When examining the prevalence of each number of neighbors, it can be seen that the degree distributes like power-law, which is a scale-free distribution [9]. The feature which is most significant to our work is the third, the clustering coefficient.

A node's cluster coefficient is defined as the ratio between the number of links that the node's neighbors have between them and the maximum of such links $\frac{d_i(d_i-1)}{2}$ - each neighbor has a link to all other neighbors but the links are not directed, hence we divide by two. It was found that in most real-world networks, the average cluster coefficient is higher than what was expected in a random graph [9], [10]. This characteristic implies that networks tend to cluster - a network can be divided into groups which are (relatively) well-connected within each group but sparsely connected between them.

Community structure was indeed observed by Grivan et al [11]; in this paper, the authors suggest an algorithm to find the different communities within a network, and indeed find such in the collaboration network and in the food web. Others have found the community structure in other networks, such as the protein interaction network [12].

In order to define the problem more accurately, we will consider the network as a graph - a simple, well-defined mathematic model that describes the relation between entities. It is the union of two sets: the vertices (nodes) - each vertex representing a different entity, and the edges (links) - each representing the existence of a relation between two entities. This description may be extended to more complex models, for example by giving direction to the edges - whether the connection is directed, like a one-way road that connects two intersections; or by allowing an edge to connect more than two vertices. In this work we will consider the weighted graph, a graph with a different strength for each edge. So a graph (network) may represent the different strengths of a connection, such as the frequency of correspondence between the two e-mail addresses in the e-mail network.

This notation allows us to define the changes which may take place between graphs representing the same system at different time-points. The change may take place at three different levels: The first and most drastic change, the addition or deletion of a vertex - an entity either no longer existing or only coming into existence. The second, changing the labels of the vertex - we do not know which vertex in the current graph is the given vertex in the preceding graph. And the third, the change in the weight of some edges - thereby changing the character of the relation; note that changing the weight of an edge to zero or from it is equivalent to removing or adding an edge, respectively.

In this work we will not deal with changes of the first and second kind, adding and removing a vertex or change of labels. When monitoring a system, we consider it a basic requirement to maintain consistency in the labels of the entities between different time samples, a demand usually easy to fulfill. As for the first kind of change, the addition or removal of a vertex is an event which in most cases takes place very rarely, and only on a long time scale.

After defining the changes that are of interest to us, several questions arise: Do all the changes in the graph effect the graph in the same way? Can the present graph be considered as a continuous development of the former graph, or is it a different graph altogether - in other words, is this graph anomalous in respect to the other graph? When does one consider two graphs to be far away? All these questions may be formalized to: what is the divergence measure between two graphs?

The most intuitive graph distance is the edit distance [13]. The edit distance is defined as the minimum changes required to be made on one graph in order for it to be identical to a second graph, a certain weight given to each of the three changes listed above. Since in this work we are only dealing with the third kind of change, the edit distance is reduced to the sum of the differences between the weights of all the edges.

A note must be made: In the literature, the distance between graphs, or graph similarity, is usually used in order to find the best matching between the two graphs' nodes. In other words, the main effort is towards finding the correct mapping (permutation) of the nodes of one graph to the other, a problem known as the graph matching. Various methods have been suggested for estimating the solution, such as the ones in [14],[15], but since we assume that the labeling of the vertices is known, we find most of these methods inadequate to our goal. We therefore return to the question of what the correct divergence measure between graphs is.

The first step in answering the question is to consider what is important to us in a graph, and what is less important; or, what are the features which are of interest to us in graphs? We will not presume to answer these questions fully, but for even a partial answer we need to examine the graph again and establish how it models the systems and what information it gives us regarding them.

We described the graph as a model for relation between entities. At a simple glance, one may mistakenly think that if two vertices (entities) are not connected by an edge, the graph holds no relation between these vertices. This is not true; there is, of course, no direct connection (otherwise an edge would have existed), but an entity can effect another by a common neighbor. So, not only the immediate neighborhood is important to us, but rather a more distant influence. If examined from the point of view of the edge, the importance of an edge is not only its weight or the vertices it connects, but rather more general features, such as its being the only edge connecting two groups of vertices.

This brings us to the third feature of complex networks - the cluster coefficient. If a graph is divided into groups, a change in the weights of edges with both end-points in the same group will be considered normal, while change in weight of edges that connect vertices in two groups will be considered more anomalous.

We have presented the intuitions leading us to this work; the rest of the work will be arranged as follows: First, some notations and definitions will be presented, to establish common terminology. Then we will define the problem and present our divergence measure. The next two sections will explain the choices made in the divergence measure - why we choose the Laplacian as the form to represent the graph, and why we take into account only the smallest eigenvectors. Chapter 7 will describe the relation between our divergence measure and the difference matrix. Chapter 8 will try to give a probabilistic interpretation of our divergence measure. In the last section, we will present our results.

Chapter 2

Notation

We will define some notations that will be used throughout this work. Capital letters A will denote matrices while lower-case letters a will denote vectors; A_j is the j column of the matrix A and $A_{i,j}$ is the value in the i row and j column of the matrix. If not stated otherwise, all our matrices are $N \times N$ symmetric matrices - $A \in \mathfrak{R}^{N \times N}$ so that $A_{i,j} = A_{j,i} \forall i, j \in \{1, \dots, N\}$. x^T will denote the transpose of x , x being a matrix or a vector. The eigenvalue matrix Λ is a diagonal matrix with eigenvalues on the diagonal. The eigenvalues are arranged in increasing order, so that $\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_N$ and $\Lambda_{i,i} = \lambda_i$. The set of all eigenvalues of a matrix A will be denoted as $\sigma(A)$. The eigenvectors will be denoted as Φ and will be arranged in the same order as their corresponding eigenvalues. We employ the above conventions in order to ease our terminology, and will refer to the eigenvectors corresponding to the smaller eigenvalues as the smallest or first eigenvectors.

We will add two further notations: $A_{[i\dots j]}$ will denote all the vectors of A starting from i till the j vector, so $A_{[i\dots j]} = (A_i, A_{i+1}, \dots, A_j)$, e.g. $A_{[i\dots j]} \in \mathfrak{R}^{N \times (j-i)+1}$; and $A_{[i\dots k, j\dots l]}$ will denote the matrix A after removal of the first $i - 1$ rows and first $j - 1$ columns, and removal of the last $N - k$ rows and the last $N - l$ columns, so $A_{[i\dots k, j\dots l]} \in \mathfrak{R}^{k-i+1 \times l-j+1}$.

Let $G = (V, E)$ be an undirected graph with vertices $V = \{1\dots N\}$ and an edge $E \subseteq (V \times V)$, the edge $e = (v_i, v_j)$ denoted as $e_{i,j}$. In this work we are working with time-dependent graphs, and we will denote the graph at time t as $G(t) = (V(t), E(t))$. As noted above, we assume that the vertices are not removed, added or relabeled during the monitoring process, and hence $V(t) = V(l)$ and $v(t)_i = v(l)_i$ for all t, l, i .

We will define W , a weight function on the edges:

Definition 1. *The weight function of the graph (V, E) at time t is defined:*

$$W: E \times \mathfrak{R}^+ \rightarrow \mathfrak{R}^+$$

■

Note, that since the graph is undirected, $e_{i,j} = e_{j,i}$ is the same edge. Hence, $W(e_{i,j}) = W(e_{j,i})$ for all i, j .

The adjacency matrix W will be defined as:

Definition 2. *Given a graph $G(t)$ and a weight function W , the adjacency matrix $W(t)$ will be:*

$$W_{i,j}(t) = \begin{cases} W(e_{i,j}, t) & e_{i,j} \in E \\ 0 & \text{otherwise} \end{cases} \quad (2.0.1)$$

where $i, j \in 1, 2, \dots, N$, $N = |V|$ - the number of vertices. ■

We define a matrix W for each time-point.
The degree matrix D will be defined:

Definition 3. Let W be the adjacency matrix, then:

$$D_{i,j} = \begin{cases} \sum_{k=1} W_{i,k} & i = j \\ 0 & i \neq j \end{cases}$$

■

The degree of the i vertex will be defined as $d_i = D_{i,i}$, where D is the degree matrix. If a matrix, graph or vector is time-dependent, it will denote $X(t)$ as the matrix, graph or vector at time t .

L will denote the Laplacian matrix, defined as:

Definition 4. Let W be the adjacency matrix; its Laplacian L is:

$$L = D - W$$

where D is as Definition 3.

Of note, two additional matrices are called Laplacian, and are defined as follows:

Definition 5. Let W be the adjacency matrix; its Normalized Laplacian L is:

$$L_{sym} = I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}}$$

where D is as in Definition 3.

Definition 6. Let W be the adjacency matrix; its Probability Laplacian L is:

$$L_{prob} = I - D^{-1} W$$

where D is as in Definition 3.

Note that this definition dictates the meaning of the entry $W_{i,j}$ as necessarily from i to j .

Chapter 3

Problem definition and the divergence measures

3.1 Problem definition

Network anomaly detection is the effort of detecting time-points in which the network behaves not as we expect it to behave. This somewhat amorphous description of our goal needs a precise definition. This section will attempt to define the problem and present the core of the suggested solution.

3.1.1 Pre-Processing

Let us start from the most basic concept. When monitoring a network, what is the data available for our use, or in other words - what is the observable data? This data may be either the activity of vertices (nodes), or the activity of the edges (links). In some cases, vertex activity may be the only activity one can see. In the cell, for example, one cannot expect to observe the reaction rate, but rather only the different quantities of the chemicals - the change in vertices' activity. In other networks, one can observe only edge activity, the usage of a specific edge. For example, in the e-mail network, one can monitor the frequency of e-mail correspondence between any two e-mail addresses.

From each time-unit, a weighted graph will be created, based on the observation. In the cases where the edges are examined, one may simply assign a weight to each edge, the weight being a certain function of the activity observed for the edge. If the vertices' activity is observed, one may create the graph by assigning each edge with a certain function of the covariance between the activity of the end-point vertices. So for each time-point we have a graph $G(t)$, and a weight function W as defined by Definition 1. The adjacency matrix $W(t)$ is defined by Definition 2 with W as its weight function.

3.1.2 Problem definition 1

We can now define a stochastic process $\{G_i\}$, with a certain probability mass function:

$$Pr \{(G(1), G(2), \dots, G(T)) = (W(1), W(2), \dots, W(T))\} = P(W(1), W(2), \dots, W(T))$$

where $W(t)$ is $N \times N$ symmetric matrix at time t , such that $W_{i,j}(t) \geq 0, W_{i,i}(t) = 0$ for all $i \neq j$ and for all t .

The probability P has two distinct states:

$$P(W(T)|W(1), W(2), \dots, W(T-1)) = \begin{cases} P_{typical}(W(T)|W(1), W(2), \dots, W(T-1)) & 1 - p_{err} \\ P_{abnormal}(W(T)|W(1), W(2), \dots, W(T-1)) & p_{err} \end{cases}$$

Some observation has to be made before defining the problem. Describing an event as normal or typical, is saying in other words that this event is similar in some way to the previous events. If we want to express such a notion in terms of probability, we will expect a normal event to have high correlation with previous events. So in the P_{typical} probability, there should be high dependence between the current state and the previous states, whereas in the P_{abnormal} one, we will expect less dependence between them. This will also imply that p_{err} must be small.

Keeping this observation in mind, the problem now can be defined as follows:

Given $W(1), W(2), \dots, W(T-1)$, was $W(T)$ generated from P_{typical} or rather from P_{abnormal} distributions?

We thus define the problem as a hypothesis test between two hypotheses, and according to the Neyman-Pearson lemma, the likelihood-ratio test is the most powerful test of size α . In Chapter 8 we will see how from this definition we conclude our divergence measure.

3.1.3 Problem definition 2

Another possible definition of our problem exists. We may consider $G(t)$, the resulted graph at time t , as a point in \mathfrak{R}^{N^2} .

The problem may now be defined as the following question: Is $G(t)$ too far from the center of previous normal points to be considered normal?

This definition of the problem instructs us to search for the correct way to measure divergence between entities - what the correct divergence function is. After obtaining the divergence measure, we can set a threshold - according to the desired ratio between type I and type II errors - that when crossed will enable us to declare the sample as anomalous.

3.2 The proposed divergence measures

We will describe the proposed divergence measures; the justifications for these will be elaborated in the following chapters. Our divergence measures are function $\Delta : G \times G \rightarrow \mathfrak{R}$. When the divergence is small, the graphs are similar to each other, and when it is large- the graphs are unlike. We will now describe our divergence measure.

We will represent the graphs as Laplacian, as was defined in Definition 4; the reason for this choice of representation will be the topic of Chapter 4. Then we will decompose the Laplacian into its eigenvector and eigenvalue $L = \Phi^T \Lambda \Phi$. We will define variables k and δ .

Definition 7. Let Λ be the eigenvalue matrix of L , with k and δ of this L will be defined as:

$$\delta = \max_i (\Lambda_{i+1,i+1} - \Lambda_{i,i})$$

$$k = \arg \max_i (\Lambda_{i+1,i+1} - \Lambda_{i,i})$$

■

The k was defined as to maximize the spectral gap - δ . From the Laplacian, we consider only the first k eigenvectors (corresponding to the smallest k eigenvalues) - $\Phi_{[1,\dots,k]}$; in Chapter 5 we will try to explain why we neglect all the other eigenvectors and disregard the eigenvalues.

Given two graphs, we denote the first as G and the second as \tilde{G} ; all variable of the graph will be denoted in the same way. We project the eigenvectors of one graph on the other, to define the M matrix.

Definition 8. Let G and \tilde{G} be two graphs. Let Φ and $\tilde{\Phi}$ be their corresponding eigenvector matrices. Let k and \tilde{k} be as defined by Definition 7. Then the matrix M is defined as:

$$M = \begin{cases} \Phi_{[1\dots k]}^T \tilde{\Phi}_{[1\dots \tilde{k}]} & k \geq \tilde{k} \\ \tilde{\Phi}_{[1\dots \tilde{k}]}^T \Phi_{[1\dots k]} & \text{else} \end{cases}$$

■

We will try to explain the meaning of projecting the two subspaces on each other, in Chapter 6. We now define two different measures of matrix M .

Definition 9. Let G and \tilde{G} be two graphs. Let M be defined as Definition 8. The determinant divergence is:

$$\Delta(G, \tilde{G}) = 1 - \sqrt{\det(M^T M)}$$

■

Intuition for this measure will be given in Chapter 6.

Definition 10. Let G and \tilde{G} be two graphs. Let M be defined as Definition 8. Let $\hat{k} = \min(k, \tilde{k})$, where k and \tilde{k} are as defined in Definition 7. Then the Trace divergence is:

$$\Delta(G, \tilde{G}) = \hat{k} - \text{Trace}(M^T M)$$

■

We will try to justify this measure in Chapter 6; we will show that this measure is bounded from above in Chapter 7. This measure will also be deduced from the attempt to solve the problem through the probabilistic definition in Chapter 8.

Chapter 4

Why Laplacian

Graphs describe relations between entities, or in other words - a linear transformation of the vertex; hence, a graph may be described by a matrix. Therefore, graphs may be considered operators of functions on vertices. In many cases, the vertices are essentially junctions, so one fundamental feature which we require of any transformation is that the amount of input entering a vertex and the amount exiting - are equal. The Kirchhoff current law is an example of such a demand in an electricity network, see 4.3. This demand is expressed on the transformation G in the following way: We will denote an input to the vertex as l_j ; note that this input originates from outside the network/graph. Moreover, note that this demand in itself dictates that the net input will be zero. For every function $f : V \rightarrow \mathfrak{R}$ on the vertex j , the operator must fulfill:

$$l_j + \sum_{i=1}^N W_{i,j} f(v_i) = \sum_{i=1}^N W_{j,i} f(v_j)$$

Alternatively:

$$l_j = \sum_{i=1}^N W_{j,i} f(v_j) - \sum_{i=1}^N W_{i,j} f(v_i) \quad (4.0.1)$$

$$= f(v_j) \sum_{i=1}^N W_{j,i} - \sum_{i=1}^N W_{i,j} f(v_i) \quad (4.0.2)$$

$$= d_j f(v_j) - \sum_{i=1}^N W_{i,j} f(v_i) \quad (4.0.3)$$

And in a matrix notation form, we get:

$$f(v)^T L = l \quad (4.0.4)$$

If the matrix is symmetric we get

$$L f(v) = l$$

where L is the Laplacian, as previously defined in Definition 4. Note that since self edges are not allowed we lose no information by using the Laplacian. In the next section we will examine additional important properties of the Laplacian, following the work of [16].

4.1 Laplacian properties

In this section we will show the following properties of L :

Proposition 1. *The matrix L satisfies the following properties:*

1. L is symmetric.
2. $\forall f \in \mathfrak{R}^N \quad f^T L f = \frac{1}{2} \sum_{i,j} W_{i,j} (f_i - f_j)^2$
3. L is a positive-semidefinite matrix.
4. All eigenvalues of L are greater or equal to 0.
5. The number of eigenvalues of L that equal to 0 is the number of connected components in the graph.

Property 1 is due to the symmetry of W and D .

Property 2 claims that the quadratic form of L equals to the square of the difference between the value of two vertices, weighted by their connectivity in the graph. We will prove this as follows:

Proof.

$$\frac{1}{2} \sum_{i,j} W_{i,j} (f_i - f_j)^2 = \frac{1}{2} \sum_{i,j} W_{i,j} (f_i^2 - 2f_i f_j + f_j^2) \quad (4.1.1)$$

$$= \sum_i d_i f_i^2 - \sum_{i,j} W_{i,j} f_i f_j \quad (4.1.2)$$

$$= f^T D f - f^T W f \quad (4.1.3)$$

$$= f^T (D - W) f \quad (4.1.4)$$

$$= f^T L f \quad (4.1.5)$$

□

Property 3 claims that L is positive-semidefinite, defined as $\forall x \in \mathfrak{R}^N \quad x^T L x \geq 0$. Therefore, this property follows directly from Property 2 and Definition 1.

Property 4 is derived directly from L being positive-semidefinite.

Property 5 may be proved as follows:

Let k denote the number of components in the graph, G_l denote the l component of graph G , and $n_l = |G_l|$ denote the size of the l component. It is clear that $\sum_{l=1}^k n_l = N$. The i vertex is represented by column and row i of the adjacency matrix W , storing the weight of the edges starting and ending at the i vertex. Without loss of generality, the vertices are aggregated into their components, so the first n_1 indices stand for all the vertices belonging to component G_1 , $i_{n_1+1} \dots i_{n_1+n_2}$ stand for the second component vertices, etc. So generalizing, the vertices of the l component are the $i_{\sum_{t=1}^{l-1} n_t+1} \dots i_{\sum_{t=1}^l n_t}$ indices. Note that in the specified order of the vertices, the adjacency matrix W has blocks on its diagonal and zeroes elsewhere.

We will define a matrix $I \in \mathfrak{R}^{N \times k}$:

$$I_{i,l} = \begin{cases} 1 & i \in G_l \\ 0 & \text{otherwise} \end{cases}$$

I is the indicator matrix - if $I_{i,j} = 1$, it implies that vertex $i \in G_j$.

Having defined the above, we are ready to prove our claim.

We will first show that the matrix has at least k eigenvalues which equal to 0, by showing that the columns of the I matrix are the corresponding eigenvectors:

$$I_l^T L I_l = \frac{1}{2} \sum_{i,j} W_{i,j} (I_{i,l} - I_{j,l})^2 = 0$$

The first equality is Property 2. In regard to the second equality - if $i, j \in G_l$ then $I_{i,l} = I_{j,l}$, and otherwise $W_{i,j} = 0$; therefore, this equality holds. We found that the columns of the matrix are eigenvectors with eigenvalue 0, and are independent since $I_l^T I_m = 0$ for all $l \neq m$, so the columns are perpendicular to each other.

Now we will prove by contradiction that there are no more than k zero-valued eigenvalues: Assume that there are indeed more than k zero-valued eigenvalues. Hence, there is an eigenvector

$$v \notin \text{span}\{I_1 \dots I_k\}, v \neq \vec{0}$$

such that:

$$Lv = 0 \tag{4.1.6}$$

Let i be a coordinate belonging to the l component, such that $v_i \geq v_j$ for all $j \in G_l$. According to 4.1.6:

$$0 = \sum_{j=1}^N L_{i,j} v_j = \sum_{j \in G_l} L_{i,j} v_j = d_i v_i - \sum_{j \in G_l} W_{i,j} v_j$$

We will change sides and divide by d_i , to get:

$$v_i = \sum_{j \in G_l} \frac{W_{i,j}}{\sum_{t \in G_l} W_{i,t}} v_j$$

From here we see that v_i is a weighted mean of all v_j , j being a neighbor of i . However, having defined above that v_i is greater than or equals v_j for all $j \in G_l$, this means that all such v_j are necessarily equal to v_i . Due to the connectivity of G_l , we can repeat this method until all $i, j \in G_l$ must fulfill $v_i = v_j$. As l was arbitrarily chosen, the claim holds for all the components. Hence $v_i \in \text{span}\{I_1 \dots I_k\}$ in contradiction to the definition of v .

Note that our definition also applies to the one component graph, when $I = \vec{1}$ (the all-one vector), so we have at least one zero eigenvalue.

We will now prove Proposition 1 for the two other Laplacian definitions, Normalized Laplacian defined by Definition 5 and Probability Laplacian defined by Definition 6.

We will prove it first for the L_{sym} : Property 1 is due to the symmetry of the identity matrix, and of W . Multiplying a matrix on both sides with the same diagonal matrix preserves its symmetry, so $D^{\frac{1}{2}} W D^{\frac{1}{2}}$ is also symmetrical. Property 2 takes the following form for this definition:

$$\forall f \in \mathfrak{R}^N \quad f^T L_{sym} f = \frac{1}{2} \sum_{i,j} W_{i,j} \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2 \tag{4.1.7}$$

Proof.

$$\frac{1}{2} \sum_{i,j} W_{i,j} \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2 = \frac{1}{2} \sum_{i,j} W_{i,j} \left(\frac{f_i^2}{d_i} - 2 \frac{f_i}{\sqrt{d_i}} \frac{f_j}{\sqrt{d_j}} + \frac{f_j^2}{d_j} \right) \tag{4.1.8}$$

$$= \sum_i f_i^2 - \sum_{i,j} W_{i,j} \frac{f_i}{\sqrt{d_i}} \frac{f_j}{\sqrt{d_j}} \tag{4.1.9}$$

$$= f^T f - f^T D^{-\frac{1}{2}} W D^{-\frac{1}{2}} f \tag{4.1.10}$$

$$= f^T \left(I - D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \right) f \tag{4.1.11}$$

$$= f^T L_{sym} f \tag{4.1.12}$$

□

From here Property 3, Property 4 and Property 5 take exactly the same steps of proving.

L_{prob} is not symmetric, so the first two properties do not apply to this definition. We will show that L_{sym} and L_{prob} share the same eigenvalues, and from this we can conclude the validity of the last three properties.

Let ϕ be an eigenvector with a corresponding eigenvalue λ of L_{sym} . Accordingly:

$$\lambda\phi = L_{sym}\phi = \left(I - D^{-\frac{1}{2}}WD^{-\frac{1}{2}}\right)\phi = D^{-\frac{1}{2}}LD^{-\frac{1}{2}}\phi$$

Multiplying both sides from the left with $D^{-\frac{1}{2}}$:

$$\lambda D^{-\frac{1}{2}}\phi = D^{-1}LD^{-\frac{1}{2}}\phi$$

Substituting $\hat{\phi} = D^{-\frac{1}{2}}\phi$, we get:

$$\lambda\hat{\phi} = D^{-1}L\hat{\phi} = (I - D^{-1}W)\hat{\phi} = L_{prob}\hat{\phi}$$

So $\hat{\phi}$ is an eigenvector of L_{prob} with eigenvalue λ , and hence both definitions share the same eigenvalues.

There is a certain relation between the adjacency matrix W 's eigenvalues and eigenvectors and the eigenvalues and eigenvectors of its Laplacian L . One such relation is given below:

Proposition 2. *Let \hat{W} be an adjacency matrix as before. We will define $W = D^{-1}\hat{W}$. W and its Laplacian L share the same eigenvectors. If λ is an eigenvalue of W , then $1 - \lambda$ is an eigenvalue of L - the Laplacian of W .*

Proof. Let ϕ be an eigenvector of W and λ its corresponding eigenvalue:

$$L\phi = (D - W)\phi \tag{4.1.13}$$

$$= (I - W)\phi \tag{4.1.14}$$

$$= \phi - W\phi \tag{4.1.15}$$

$$= \phi - \lambda\phi \tag{4.1.16}$$

$$= (1 - \lambda)\phi \tag{4.1.17}$$

□

where 4.1.14 is due to the definition of W .

A note must be made: We called L Laplacian even though the W is not symmetric. The justification for this is the L_{prob} Laplacian, which suffers from the same problem but has the last three properties of Proposition 1.

4.2 Matrix Laplacian vs Continuous Laplacian

When asked what a Laplacian is, most scientists will not think of our definition, but rather of the continuous Laplace operator denoted by Δ . Let us present the relation between the two definitions:

A possible definition of the Laplace continuous operator is $\Delta f = \nabla^2 f = \sum_{i=1}^M \frac{\partial^2 f}{\partial x_i^2}$, where $f : \mathfrak{R}^M \rightarrow \mathfrak{R}$ is a twice-differentiable function, and X_i are the Cartesian coordinates.

When trying to demonstrate the relation between a continuous operator, such as Δ , and a discrete operator, such as L , approximation of the \mathfrak{R}^M space is necessary (note that L is an operator, $f : V \rightarrow \mathfrak{R}$ - a function of the graph's vertices, and $L(f) = f^T Lf$). We will approximate the \mathfrak{R}^M space as a lattice

of M dimensions, and will set the function on the edges to be $W(e) = \frac{1}{h^2}$, $\forall e \in E$, $W(e) = 0 \forall e \notin E$. f is a function on the vertices of the lattice, and as h tends to 0, f will tend to \tilde{f} , $\tilde{f} : \mathbb{R}^M \rightarrow \mathbb{R}$.

$$\lim_{h \rightarrow 0} \lim_{N \rightarrow \infty} L(f) = \lim_{h \rightarrow 0} \lim_{N \rightarrow \infty} f^t L f \quad (4.2.1)$$

$$= \lim_{h \rightarrow 0} \lim_{N \rightarrow \infty} \sum_{i=1}^N \sum_{j=1}^N W(e_{i,j}) (f_i - f_j)^2 \quad (4.2.2)$$

$$= \lim_{h \rightarrow 0} \sum_{e_{i,j} \in E} \frac{(f_i - f_j)^2}{h^2} \quad (4.2.3)$$

$$\approx \lim_{h \rightarrow 0} \sum_{i=1}^M \frac{\left(\tilde{f}(a_1, a_2, \dots, a_i + h, a_{i+1}, \dots, a_M) - \tilde{f}(a_1, a_2, \dots, a_i, a_{i+1}, \dots, a_M) \right)^2}{h^2} \quad (4.2.4)$$

$$= \lim_{h \rightarrow 0} \sum_{i=1}^M \frac{\partial^2 \tilde{f}}{\partial x_i^2} \quad (4.2.5)$$

when 4.2.2 is according to Property 2, and 4.2.4 is due to the regularity of G and the characteristics of f . Note, a formal proof of this relation has been attempted in several studies, for example in [17].

4.3 Laplacian and electricity

In the last section we referred to the Laplacian as an operator on a function of the vertices. One desirable feature of such an operator is expressed as the Kirchhoff current law - the sum of current entering a node is equal to the sum of current leaving the node. This law expresses the simple notion that a node does not create a flow de-novo, rather it only distributes the flow to its neighbors. In order to see how this notion is compelled by the Laplacian, we will show how this property is expressed in an electricity network.

Let us define the matrix W to describe the electricity network, and denote $R_{i,j}$ as the resistance between the i and j nodes, such that $W_{i,j} = \frac{1}{R_{i,j}}$ if $R_{i,j} < \infty$, and otherwise $W_{i,j} = 0$ (note that $W_{i,i} = 0$ for all i). The Laplacian will be defined as usual $D - W$. For every vertex i the Kirchhoff current law can now be written as:

$$\sum_{j=1}^N W_{i,j} v_j = \sum_{j=1}^N W_{j,i} v_j + I_i$$

where I_i is an outside-added current to the i node.

Because of the symmetry of W , we may write:

$$I_i = \sum_{j=1}^N W_{i,j} v_j - \sum_{j=1}^N W_{j,i} v_j = d_i v_i - \sum_{j=1}^N W_{i,j} v_j = [Lv]_i$$

So the Laplacian matrix is formed by enforcing the Kirchhoff current law on the function of the nodes. Note that the I_i 's are constrained $\sum_{i=1}^N I_i = 0$, which can easily be justified by our notation:

$$I = Lv$$

We will multiply both sides with the $\vec{1}$ all-one vector:

$$\sum_{i=1}^N I_i = \vec{1}^T LV = \vec{1}^T \Phi \Lambda \Phi^T V = \left[N \sqrt{\frac{1}{N}}, 0, \dots, 0 \right] \Lambda \Phi^T V = \vec{0} \Phi^T V = 0$$

where Φ is the eigenvector matrix and Λ is the eigenvalue matrix.

If we wish to assess the amount of energy for a specific state, we get the following equation:

$$E(v) = v^T I = v^T L v \quad (4.3.1)$$

Note that this equals to $\sum_{i,j=0}^N \frac{1}{R_{i,j}} (v_i - v_j)$ by Property 2, which can now be interpreted as the electrical potential between all nodes scaled by the conductivity.

Let us examine another relation to electricity, using a network similar to the one presented at [18]. The network consists of N capacitors, connected at one end to the ground and at the other - to each other through the resistor network G (defined as before). We will assume that the capacity of all capacitors is 1. The network can now be expressed using the equation:

$$\dot{v}_i = \sum_j W_{i,j} v_j - \sum_j W_{j,i} v_i = \sum_j W_{i,j} v_j - d_i v_i$$

and in a matrix notation:

$$-L v = \dot{v}$$

with the solution:

$$v(t) = e^{-Lt} v(0) \quad (4.3.2)$$

The equation describes the potential of the capacitors at time t , given $v(0)$ as their potential at time 0.

4.4 Random walks and the Laplacian

A random walk on a graph is a Markov process of moving from vertex to vertex in each time-unit, according to the graph structure. Let P_t be the probability vector, such that $P(x_t = i)$ is the probability to be at vertex i at time t . Let W be as previously defined. Given that at time t one's location is at vertex i , the probability to be at time $t + 1$ at vertex j is $P(x_{t+1} = j | x_t = i) = W_{i,j}/d_i$, where d_i is as previously defined. Hence in matrix notation, it may be written $P_{t+1} = P_t^T D^{-1} W$, and by induction $P_t = P_0^T (D^{-1} W)^t$.

Following [18], we will define a different random walk - a lazy walk. This walk differs from a regular walk by the option of staying in the same vertex with probability of $1 - \alpha$ (note that this variation may be interpreted as a self edge with a constant weight to all the vertices). The probability to move from vertex i to vertex j is $P(x_{t+1} = j | x_t = i) = \frac{\alpha W_{i,j}}{d_i}$, when $0 \leq \alpha \leq 1$. The probability to be in vertex j in time $t + 1$ is therefore:

$$\begin{aligned} P_{t+1}(j) &= P_t(j) - \sum_i \frac{\alpha W_{j,i}}{d_j} P_t(j) + \sum_i \frac{\alpha W_{i,j}}{d_i} P_t(i) \\ &= P_t(j) + \alpha \left(\sum_i \frac{W_{i,j}}{d_i} P_t(i) - \frac{W_{j,i}}{d_j} P_t(j) \right) \end{aligned}$$

We will change sides to get:

$$P_{t+1}(j) - P_t(j) = \alpha \left(\sum_i \frac{W_{i,j}}{d_i} P_t(i) - \frac{W_{j,i}}{d_j} P_t(j) \right)$$

Moving to continuous time, by adding δt steps to t :

$$\frac{P_{t+\delta t}(j) - P_t(j)}{\delta t} = \frac{\alpha}{\delta t} \left(\sum_i \frac{W_{i,j}}{d_i} P_t(i) - \frac{W_{j,i}}{d_j} P_t(j) \right)$$

In the limit where $\delta \rightarrow 0$, and replacing $\frac{\alpha}{\delta t} = \beta$, we get:

$$\dot{P}(j) = -\beta \left(P_t(j) - \sum_i \frac{W_{i,j}}{d_i} P_t(i) \right)$$

and in matrix notation:

$$\dot{P}_t = -\beta P_t^T D^{-1} L$$

with the solution:

$$P_t = P_0^T e^{-\beta D^{-1} L t}$$

thereby reaching an equation similar to equation 4.3.2. Note that here we just reached the L_{prob} definition (see Definition 6).

We will show another approach by defining a more intuitive random walk - a quick walk. As in the lazy walk, we will remove some "stiffness" from the walk: in each time we will allow the walk to take more than one step. More formally, the vector probability $P_t = P_0^T \left(e^{-1} \sum_{k=0}^{\infty} \frac{\tilde{W}^k}{k!} \right)^t$, where e is the base of the natural logarithm, $\tilde{W} = D^{-1} W$ and P_0 is the initial probability. Let us first make sure that we defined a conditional distribution - we need to make sure every row is equal to 1.

For the l row:

$$\sum_{j=1}^N e^{-1} \sum_{k=0}^{\infty} \frac{[\tilde{W}^k]_{l,j}}{k!} = e^{-1} \sum_{k=0}^{\infty} \frac{\sum_{j=1}^N [\tilde{W}^k]_{l,j}}{k!} \quad (4.4.1)$$

$$= e^{-1} \sum_{k=0}^{\infty} \frac{1}{k!} \quad (4.4.2)$$

$$= e^{-1} e \quad (4.4.3)$$

$$= 1 \quad (4.4.4)$$

where 4.4.2 is due to definition of \tilde{W} and to the fact that \tilde{W}^k is a conditional distribution.

We will show the relation between this walk and the Laplacian of \tilde{W} . The same note as in Proposition 2 must be made.

$$P_t = P_0^T \left(e^{-1} \sum_{k=0}^{\infty} \frac{\tilde{W}^k}{k!} \right)^t \quad (4.4.5)$$

$$= P_0^T \left(e^{-1} e^{\tilde{W}} \right)^t \quad (4.4.6)$$

$$= P_0^T \left(\Phi^T e^{-1 + \Lambda_{\tilde{W}}} \Phi \right)^t \quad (4.4.7)$$

$$= P_0^T \left(\Phi^T e^{-\Lambda_L} \Phi \right)^t \quad (4.4.8)$$

$$= P_0^T e^{-L t} \quad (4.4.9)$$

where 4.4.6 is due to the definition of matrix exponent, and 4.4.8 is due to Proposition 2.

Again we reach the same equation as 4.3.2.

This equation has a well-known importance in classical physics. It is known as the Heat equation, which is the Diffusion equation for a constant diffusion coefficient. So, this last-delineated random walk describes diffusion as the process of an infinite number of walkers beginning such a random walk.

4.5 Laplacian and spectral graph theory

Spectral graph theory is the study presenting the connection between linear algebra and graph theory. In [19], the author showed numerous connections between the graph and the eigenvalues of its adjacency matrix, or more correctly, the eigenvalues of the normalized Laplacian. We will present one of the basic connections - the bound of the Cheeger constant of a graph and the second eigenvalue of the Laplacian.

The Cheeger constant was presented on an unweighted graph; we will define an extension of the constant for a weighted graph. Let $S \subset V$, we will define the edge boundary weight to be

$$\partial S = \sum_{i \in S} \sum_{j \notin S} W(e_{i,j})$$

where W as defined by Definition 1. The volume of a set of vertices will be defined as

$$\text{vol}(S) = \sum_{i \in S} d_i$$

where d_i as defined by Definition 3. So the Cheeger constant may be defined as

$$h_G = \min_S \frac{\partial S}{\min(\text{vol}(S), \text{vol}(\bar{S}))}$$

We will show only one side of the Cheeger inequality.

Theorem 1. *Let h_G , defined as above.*

Let λ_i be the eigenvalues of the normalized Laplacian as defined by Definition 5. Then

$$2h_G \geq \lambda_2$$

We will now extend the Cheeger constant to apply not only to the case of two sets of vertices, but to many.

Let the edge boundary weight be defined as above. We can now define the constant as:

$$\tilde{h}_{G,k} = \min_{\{S_1, S_2, \dots, S_k\}} \sum_{i=1}^k \frac{\partial S_i}{\text{vol}(S_i)}$$

This definition is now exactly the Ncut problem as defined by [20].

We will now state the bound.

Theorem 2. *Let $\tilde{h}_{G,k}$ be defined as above.*

Let λ_i be the eigenvalues of the normalized Laplacian as defined by Definition 5. Then:

$$2\tilde{h}_{G,k} \geq \sum_{i=1}^k \lambda_i$$

Note that this definition applies also to the two-set case, since the first eigenvalue is always zero.

Proof. We will define the matrix H as follows:

$$H_{i,j} = \begin{cases} \frac{1}{\sqrt{\text{vol}(S_i)}} & j \in S_i \\ 0 & \text{otherwise} \end{cases}$$

$$\tilde{h}_{G,k} = \min_{\{S_1, S_2, \dots, S_k\}} \sum_{i=1}^k \frac{\partial S_i}{\text{vol}(S_i)} \quad (4.5.1)$$

$$= \min_{\{S_1, S_2, \dots, S_k\}} \sum_{i=1}^k \sum_{l \in S_i} \sum_{m \notin S_i} \frac{W(e_{l,m})}{\text{vol}(S_i)} \quad (4.5.2)$$

$$= \min_{\{S_1, S_2, \dots, S_k\}} \sum_{i=1}^k \frac{1}{2} \sum_{l,m=1}^N W_{l,m} (H_{i,l} - H_{i,m})^2 \quad (4.5.3)$$

$$= \min_{\{S_1, S_2, \dots, S_k\}} \sum_{i=1}^k \frac{1}{2} H_i^T L H_i \quad (4.5.4)$$

$$= \min_{\{S_1, S_2, \dots, S_k\}} \frac{1}{2} \text{Trace}(H^T L H) \quad (4.5.5)$$

It is easy to see that $H_i^T H_j = 0$ for all $i \neq j$ so H_i are perpendicular to each other. Hence, demanding only that the H_i be independent from each other is a relaxation of the definition of H_i , and moreover minimizing H is exactly like minimizing $U = D^{\frac{1}{2}} H$. Note that $H^T D H = I$, and we get

$$\tilde{h}_{G,k} = \min_{\{S_1, S_2, \dots, S_k\}} \frac{1}{2} \text{Trace}(H^T L H) \quad (4.5.6)$$

$$\geq \min_{\{H | H_i^T H_j = 0 \forall i \neq j\}} \frac{1}{2} \text{Trace}(H^T L H) \quad (4.5.7)$$

$$= \min_{\{U | U^T U = I\}} \frac{1}{2} \text{Trace}(U^T D^{-\frac{1}{2}} L D^{-\frac{1}{2}} U) \quad (4.5.8)$$

$$= \min_{\{U | U^T U = I\}} \frac{1}{2} \text{Trace}(U^T L_{\text{sym}} U) \quad (4.5.9)$$

$$= \frac{1}{2} \sum_{i=1}^k \lambda_i \quad (4.5.10)$$

□

Note that the matrix U that minimizes the Eq. (4.5.9) is $U = \Phi_{[1 \dots k]}$ - the first eigenvectors of the normalized Laplacian. Hence there is a close connection between these eigenvectors and the optimal division of the vertices into k separate and distinct sets. This brings us to the next section, of why we use only these vectors in our divergence measure.

Chapter 5

The neglect of the eigenvalues and some eigenvectors

Decomposing a matrix to its eigenvector and eigenvalue matrices is a basic method for separating two aspects of the matrix. The eigenvectors describe the coordinate system dictated by the matrix, while the eigenvalues show the amount by which the transformation scales each vector in the direction of the corresponding eigenvector.

In our divergence measure we only take into account the first eigenvectors. In this section we will try to explain this choice.

5.1 Spectral clustering

Spectral clustering is a well-known method for clustering data. This method is based on taking the first eigenvectors of the representing matrix, and from these vectors derive the different clusters by simple methods.

There are several intuitions for spectral clustering.

One, it is regarded as the solution for the min cut problem. An example is [20], which proves that the first eigenvectors minimize a relaxation of the normalized cut, the grouping of vertices so that the sum of weights of edges between groups is minimal relative to the groups' inner edges' weights.

Another intuition is found in [21], which is derived from the perturbation theory. In the optimal case, the groups will be unconnected, so the first eigenvectors will be indicator vectors- having zero when the vertex is not in the group. The real case is regarded like the ideal case, where the edges between the different groups are regarded as noise. (We use similar intuition in Chapter 7).

The success of the method in real-life applications proves that the first eigenvectors indeed hold the information regarding how to divide the graph into groups. This is a global, important feature of the graph. Hence, we believe that any change in these eigenvectors is important, and that even the smallest changes are critical. On the other hand, these eigenvectors are robust against normal fluctuations of the graph, such as change inside clusters.

5.2 Why neglect the eigenvalues

One may ask what information should be gathered from the eigenvalues. As mentioned in Definition 8, the choice of which eigenvectors will be counted as the "first", as well as how many eigenvectors will be chosen, is the only information we gather from the eigenvalues. At first glance, it may therefore seem that not taking the values of the eigenvalues into account is actually to lose vital information. We will try to explain why we do not take other information of the eigenvalues into account, and demonstrate

why information from the eigenvalues may be misleading - or, more accurately, why the information we gather is the only information worth taking.

Our goal in this work is to detect anomalies, and as was stressed in the introduction, not all changes should have the same weight in our divergence function. The edit distance simply sums the changes between the graphs, thereby giving an equal weight to all changes. [22] showed that the eigenvalue change is in direct correlation to the edit distance. So we can conclude that changes that should have little weight, if any, on the divergence measure are effecting eigenvalues, and the eigenvalues are essentially effected by information that is irrelevant to us. Thus, the eigenvalues, being sensitive to all changes and not only to relevant ones, are limited as a feature for detecting anomalies.

We have stressed the point that the eigenvalues contain irrelevant information. Moreover, the other way around is also true, and the eigenvalues are not sensitive enough when dealing with small changes. This can be seen from the bound [23]

$$\min_{\pi} \max_i (\tilde{\lambda}_{\pi(i)} - \lambda_j) \leq \|A - \tilde{A}\|$$

which is true when both A and \tilde{A} are symmetric, as in our case; meaning that the eigenvalues cannot be too far from each other if the perturbation is small. So in conclusion, the eigenvalues are on the one hand effected by irrelevant information, and on the other hand - not sensitive enough for information that might be relevant. This renders them of very limited use for our purpose.

5.3 The eigenvector of an adjacency matrix and the Fourier transform

The Fourier transform is widely used in the signal processing and image processing communities. The transformation of a function into the frequency domain allows removal of unwanted frequencies. The low-pass filter removes the highest frequencies, leaving the lowest ones, which describe the more fundamental properties of the function.

When trying to compare two graphs, a reasonable approach will be to compare the fundamental properties of these graphs. Extracting those properties from the graph may be achieved by an approach similar to the one using the low-pass filter - the Fourier transform. One way to get the discrete Fourier transform is as the eigenvector of the adjacency matrix of the one-dimension lattice. Getting the exact DFT will be only in the infinite lattice, so to avoid the boundary problem, the margins will be connected, with a result of a torus. More precisely, given $G(V, E)$, $V = \{v_0, v_1 \dots, v_{M-1}\}$ and $E = (v_i, v_{i+1}) | 1 \leq i \leq M - 2 \cup (v_{M-1}, v_1)$, it is easy to see that the adjacency matrix of this graph is a circulant matrix. If we examine the eigenvector of a circulant matrix A with eigenvector ϕ [24]:

$$\Phi^T A \Phi = \Lambda$$

In a circulant matrix, $A_{k,l} = A_{1,(l-k) \bmod M} = a_{(l-k) \bmod M}$. Therefore, if we look at coordinate k of the m eigenvector $\Phi_{k,m}$, we will get:

$$\begin{aligned} \Lambda_{m,m} \Phi_{k,m} &= [A \Phi_m]_k \\ &= \sum_{l=0}^{M-1} A_{k,l} \Phi_{l,m} \\ &= \sum_{l=0}^{k-1} a_{l-k+M} \Phi_{l,m} + \sum_{l=k}^{M-1} a_{l-k} \Phi_{l,m} \\ &= \sum_{l=M-k}^{M-1} a_l \Phi_{l+k-M,m} + \sum_{l=0}^{M-k-1} a_l \Phi_{k+l,m} \end{aligned}$$

We will guess a solution to the equation - $\Phi_{k,m} = e^{\frac{-2\pi mik}{M}}$, where $i^2 = -1$ (the imaginary i). We will now find Λ :

$$\Lambda_{m,m} e^{\frac{-2\pi mik}{M}} = \sum_{l=M-k}^{M-1} a_l e^{\frac{-2\pi mi(l+k-M)}{M}} + \sum_{l=0}^{M-k-1} a_l e^{\frac{-2\pi mi(k+l)}{M}}$$

Dividing both sides by $e^{\frac{-2\pi mik}{M}}$:

$$\begin{aligned} \Lambda_{m,m} &= e^{-2\pi mi} \sum_{l=M-k}^{M-1} a_l e^{\frac{-2\pi mil}{M}} + \sum_{l=0}^{M-k-1} a_l e^{\frac{-2\pi mil}{M}} \\ &= \sum_{l=0}^{M-1} a_l e^{\frac{-2\pi mil}{M}} \end{aligned}$$

We will verify our solution:

$$\begin{aligned} [A\Phi_m]_k &= \sum_{l=M-k}^{M-1} a_l e^{\frac{-2\pi mi(l+k-M)}{M}} + \sum_{l=0}^{M-k-1} a_l e^{\frac{-2\pi mi(k+l)}{M}} \\ &= e^{\frac{-2\pi mik}{M}} \sum_{l=0}^{M-1} a_l e^{\frac{-2\pi mil}{M}} \\ &= \Phi_{k,m} \Lambda_{m,m} \end{aligned}$$

From the above one may see that the eigenvectors are the various vectors of the DFT. The same may be applied to higher dimensions, by ordering the vertices such that a $n + 1$ -dimension torus is created.

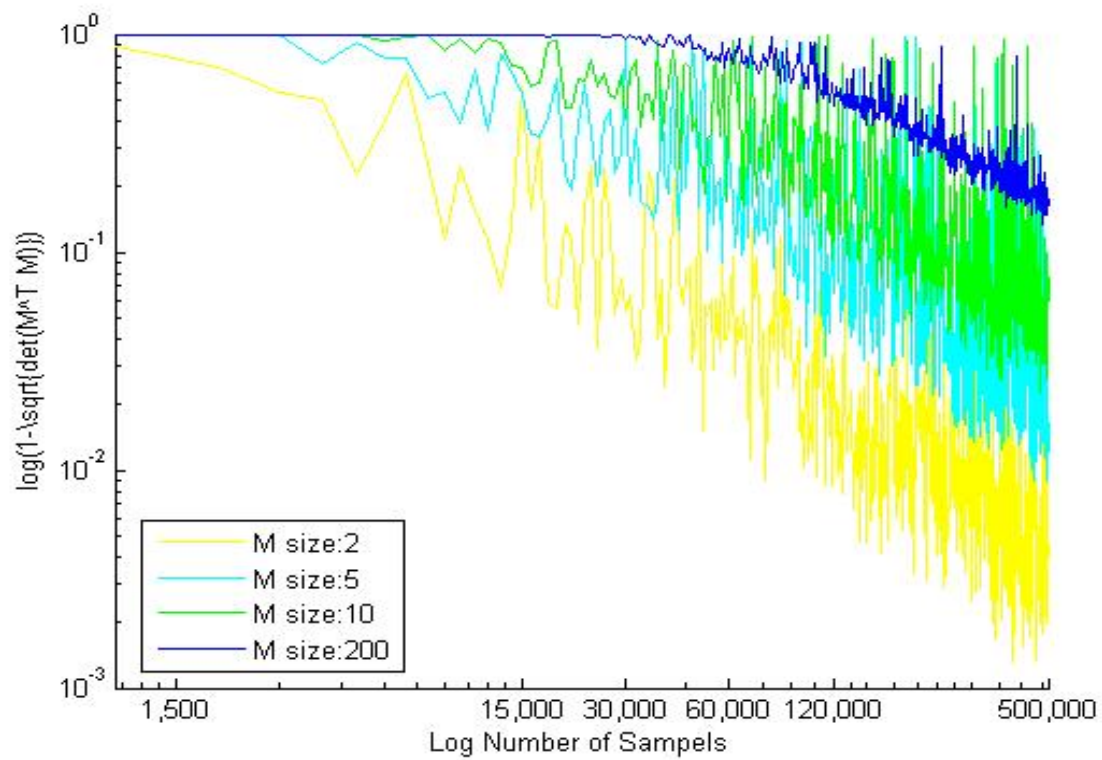
We have seen that the low-pass filter is actually choosing the lowest eigenvectors of a very ordered graph - a lattice. In the same spirit, taking the lowest eigenvectors of any graph may give the same desired properties of the low-pass filter.

Another reason to prefer the low frequencies is due to the following fact: The NyquistShannon sampling theorem [25] states that when trying to calculate the sampling rate necessary for perfect reconstruction of a function, the sample frequency has to be more than twice the highest frequency of the sampled function. This drives us to the conclusion that low frequencies of the function are better estimated at a low-frequency sampling rate. So using only the low frequency, or low eigenvector, as our divergence measure, allows us to avoid using under-sampled functions, and hence, inaccurate features of the graph. Figure 5.1 demonstrates this principle; we can see that the smaller eigenvectors converge to their values before the higher eigenvectors do. Yet another point can be concluded from this figure: When comparing the two sub-figures, it can clearly be seen that in sub-figure 5.1(b) the small eigenvectors converge a great deal faster than in sub-figure 5.1(a). This is due to the underlying structure of the high-cluster graph (sub-figure 5.1(b)) and its lack in the random graph. From this point of view, the high-cluster graph is a function with dominant low frequency, while the random graph is a function lacking this property.

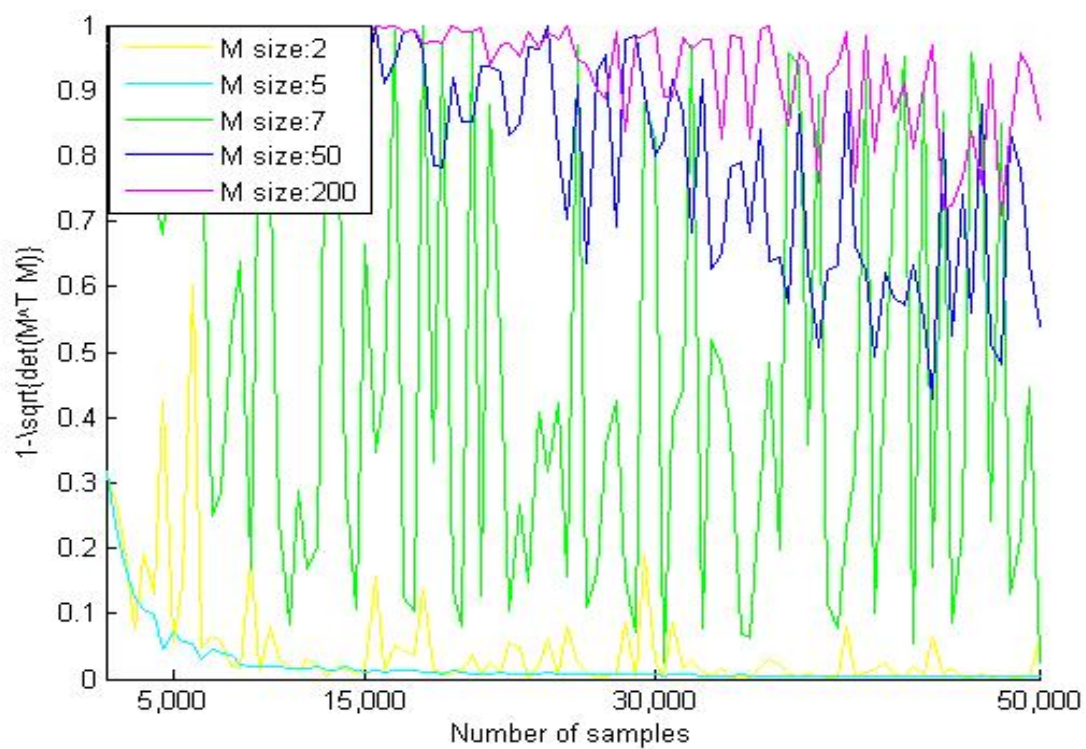
5.4 Eigenvectors and diffusion

In the previous section, we have seen the Heat equation 4.3.2. We have shown that it may be regarded as a random walk, and is known to describe the diffusion of heat. Activity of a graph may also be regarded as such a process. If a vertex is active, and not as a result of an influence of another vertex, the activity is seen as the beginning of a diffusion process - an outside input to the graph.

Nadel , Coifman, Lafon and others have demonstrated the close relation between the normalized Laplacian and the diffusion operator (e.g [26], [27], [17], [28], [29]) . In our context, their main theory



(a) Random Graph



(b) High Cluster Graph

Figure 5.1: Convergence of the eigenvectors

states that as the number of vertices tends to infinity while the mean degree remains constant, the random walk on the graph converges to the diffusion process. Therefore the finite case may be regarded as an approximation of the infinite case, and hence known facts about the diffusion process may be applied on the random walk.

Let us look at the Eq. (4.3.2) again:

$$e^{-Lt}P(0) = \sum_{i=1}^N e^{-\lambda_i t} \Phi_i^T \Phi_i P(0)$$

It is a known fact that as $t \rightarrow \infty$, the diffusion reaches a steady state, and in this state each of the connected components has the same distribution inside it. This observation may easily be deduced if we remember Property 5, the only zeroed eigenvectors are the ones that are uniform inside each component.

In a connected graph which can be easily divided into clusters, a similar behavior may be observed. If enough time has elapsed, the concentration in all vertices of each of the components will be almost the same, while it will take longer to reach the state where all vertices in the graph have the same concentration - the steady state. Moreover, this equation implies that whenever there is a gap in the spectrum (regardless of whether this gap is due the graph being well-clustered), the spectrum can be divided into two parts: The first group is expected to have influence only in short time-periods - the group that has the larger eigenvalues; and the second governs the diffusion process - the eigenvectors corresponding to the smaller eigenvalues.

Tishby et al [30] use this feature for clustering. They use the transition matrix after t steps - which is similar to the diffusion process - hence using only the small eigenvectors, since the higher eigenvectors were lost by the walking process. So in principle, they present an algorithm similar to the spectral clustering algorithm. By this they give an additional intuition to the spectral clustering algorithm, and some understanding as to how many eigenvectors should be used.

An interpretation of this observation can be seen to agree with the conclusions of the previous subsection. The eigenvectors that almost exclusively have an effect on the graph in the long run, or in the terminology of last subsection - the low frequencies, are the eigenvectors corresponding to the smallest eigenvalues.

5.5 Diffusion maps

In this chapter we have tried to explain why we use only the first eigenvectors. By doing this we have given several interpretations to eigenvectors themselves. The first interpretation is that these eigenvectors take the role of the low frequencies in the Fourier transform. We argue that the activity of the graph can be seen as a diffusion process. In this context, the first eigenvectors are concentration states of the system in which the diffusion process lingers, before it reaches its steady state (given that the graph is connected). These eigenvectors are still sensitive to the initial condition at a certain level of resolution.

In this sub-section we present another interpretation of these eigenvectors. In the work of Coifman, Nadel et al. they present a new distance measure - diffusion distance.

$$D_t^2(v_{i_1}, v_{i_2}) = \sum_{j=1}^N ([\tilde{W}^t]_{i_1,j} - [\tilde{W}^t]_{i_2,j})^2 \frac{1}{\Phi_{1,j}}$$

where $\tilde{W} = D^{-1}W$.

This distance measure compares two vertices by comparing their paths' distributions. More specifically, it compares the probability of all paths of length t that start from vertex v_{i_1} , with the probabilities of paths that start at vertex v_{i_2} . The idea behind this comparison is, that after t steps, the initial condition (we start from vertex v_i) is to a certain extent forgotten. Hence, two vertices that share the same

surroundings will have roughly the same probabilities for all paths. while two vertices not sharing the same neighborhood, such as vertices in different clusters, will have a great difference between their probabilities.

After defining the distance measure, and explaining its reasoning, we are ready to show its relation to the first eigenvectors. If we decompose $\tilde{W} = \Psi^T \Lambda \Phi$, the distance can be written as:

$$D_t^2(v_{i_1}, v_{i_2}) = \sum_{j=1}^N \lambda_j^{2t} (\Psi_{i_1,j} - \Psi_{i_2,j})^2$$

If we now define a mapping from the vertices to R^N

$$F_t(v_i) = [\lambda_1^t \Psi_{1,i}, \lambda_2^t \Psi_{2,i}, \dots, \lambda_N^t \Psi_{N,i}]$$

So in the map space, the Euclidian distance is equal to the diffusion distance in the original space.

In this context, the eigenvectors are the mapping of the vertices into the Euclidian space. A few notes: First, the Laplacian is not used here at all, but remembering Proposition 2 we can conclude that both share the same eigenvectors. In this mapping we scale the eigenvectors by their corresponding eigenvalues. Remembering that we take only the first eigenvectors - which correspond to the near 1 eigenvalues of the matrix, we can neglect this scaling and lose only little information about the mapping; for detail see [31].

Chapter 6

From projection of subspace to our divergence measures

The matrix which is the sole parameter to our divergence measures is the matrix M . According to Definition 8, the matrix is the result of projection of two subspaces - one for each graph. The question we will try to answer in this section is why projecting the subspace, and what is the meaning of the chosen functions which evaluate this projection.

We will start by trying to explain why to project the subspace, by quoting a theory [32] (a more relaxed version of this theory can be found at [23], [33]).

6.1 The CS decomposition

Theorem 3. (*The CS Decomposition*)

Let W be an $M \times M$ unitary matrix, partitioned as

$$W = \begin{matrix} & \begin{matrix} k & M-k \end{matrix} \\ \begin{matrix} l \\ M-l \end{matrix} & \begin{pmatrix} W_{1,1} & W_{1,2} \\ W_{2,1} & W_{2,2} \end{pmatrix} \end{matrix}$$

where $k \leq l \leq M$ and $2l \leq M$.

Then there are unitary matrices $U = \text{diag}(U_{1,1}, U_{2,2})$ and $V = \text{diag}(V_{1,1}, V_{2,2})$, where $U_{1,1}$ is a $l \times l$ unitary matrix and $V_{1,1}$ is a $k \times k$ unitary matrix, such that:

$$U^T W V = \begin{matrix} & \begin{matrix} k & l-k & k & M-l-k \end{matrix} \\ \begin{matrix} k \\ l-k \\ k \\ M-l-k \end{matrix} & \begin{pmatrix} C & 0 & -S & 0 \\ 0 & I & 0 & 0 \\ S & 0 & C & 0 \\ 0 & 0 & 0 & I \end{pmatrix} \end{matrix} \quad (6.1.1)$$

where C and S are nonnegative diagonal matrices, with diagonal entries $0 \leq c_1 \leq c_2 \leq \dots \leq c_k \leq 1$ and $0 \leq s_1 \leq s_2 \leq \dots \leq s_k \leq 1$, respectively, and

$$S^2 + C^2 = I \quad (6.1.2)$$

The proof uses the QR-decomposition for the different parts of the matrix, and the fact that the product of unitary matrices is a unitary matrix.

Note that Eq. (6.1.2) implies that $c_i^2 = 1 - s_i^2$ for all $1 \leq i \leq k$.

A simple result of this theorem is the following one:

Theorem 4. Let $X = (X_1 X_2)$, $Y = (Y_1 Y_2)$ be two orthonormal bases of \mathfrak{R}^M , such that $X_1 = [X_1 \dots X_l]$ and $Y_1 = [Y_1 \dots Y_k]$ and $2l < M$, $k \leq l$. Then there exist unitary matrices $U = \text{diag}(U_1, U_2)$ and $V = \text{diag}(V_1, V_2)$, where U_1 is a $l \times l$ unitary matrix and V_1 is a $k \times k$ unitary matrix, such that:

$$\begin{matrix} & & & & k & l-k & k & M-l-k \\ & & & & C & 0 & -S & 0 \\ l & & k & M-k & 0 & I & 0 & 0 \\ & & & & S & 0 & C & 0 \\ M-l & & & & 0 & 0 & 0 & I \end{matrix} \begin{pmatrix} U_1^T X_1^T Y_1 V_1 & U_1^T X_1^T Y_2 V_2 \\ U_2^T X_2^T Y_1 V_1 & U_2^T X_2^T Y_2 V_2 \end{pmatrix} = \begin{matrix} k & l-k \\ l-k & k \\ k & M-l-k \\ M-l-k & k \end{matrix} \begin{pmatrix} C & 0 & -S & 0 \\ 0 & I & 0 & 0 \\ S & 0 & C & 0 \\ 0 & 0 & 0 & I \end{pmatrix} \quad (6.1.3)$$

such that:

$$C^2 + S^2 = I \quad (6.1.4)$$

Note:

$$\begin{pmatrix} C \\ 0 \end{pmatrix} = U_1^T X_1^T Y_1 V_1 \quad (6.1.5)$$

$$\begin{pmatrix} S \\ 0 \end{pmatrix} = U_2^T X_2^T Y_1 V_1 \quad (6.1.6)$$

and

$$\begin{pmatrix} 0 & -S & 0 \\ I & 0 & 0 \end{pmatrix} = U_1^T X_1^T Y_2 V_2 \quad (6.1.7)$$

This theorem can have another interpretation: Let X_1, Y_1 be two orthonormal bases of l, k -dimensional subspaces Ξ, Ψ of \mathfrak{R}^N , respectively. Projecting these bases on each other, gives us Eq. (6.1.5), Eq. (6.1.6) and Eq. (6.1.7), where in this context $X_2 = X_1^\perp$ - a certain perpendicular base for the complement subspace of Ξ , and the same for $Y_2 = Y_1^\perp$. If we notice the fact that $S^2 + C^2 = 1$, then the following definition is expected:

Definition 11. The angle operator Θ between subspaces Ξ and Φ

$$\Theta(\Xi, \Psi) = \arcsin S$$

■

Note that S and Θ are diagonal matrices. The values on the diagonal of Θ are called canonical angles. An important feature of the angles is that they are independent from the choice of the orthogonal base. To prove this feature, let $X_1 \hat{X}_1$ be two orthogonal bases of Ξ and let Y_1 be the same for Φ . Since the choice of orthogonal base of the complement of Ξ is independent of the choice of the base of Ξ , and from Eq. (6.1.6) - one can conclude the independence of S from the choice of the base of Ξ . If we look at Eq. (6.1.2) we conclude the independence of C from the base choice. The same argument can be applied to Y using the same argument with Eq. (6.1.7).

This definition agrees with our usual definition for angle between vectors (subspace of one dimension). If we force $|x| = 1$ and $|y| = 1$ - as we force our base to be normalized - it is reduced to:

$$\cos \theta = x^T y$$

This is exactly our definition, since $\cos \Theta = C$, which is a diagonal matrix with the eigenvalue of $X^T Y$ on its diagonal. The canonical angles have another definition and it can be proved that the two definitions are the same, for more details see [33].

We are now in position to understand the matrix M . The non-zero singular values of the M matrix are the canonical angles between the subspaces spanned by the first eigenvectors of each graph. Measuring similarity between subspaces through these angles is the most intuitive and natural technique, the same as measuring the distance between lines with angles. From here it is easy to deduct that the way to measure the size of M is by similarity-invariant functions. This brings us to the next topic - the reasoning of the divergence measures.

6.2 Reasoning the divergence measures

6.2.1 The Determinant measure

The first measure has a geometric intuition. In Definition 11 we have seen that the non-zero singular value of M may be regarded as the $\cos\theta$, where θ denotes the angles between the subspaces. The natural way to proceed is to find the volume of the parallelotope created by the cosine of the angles. Finding the volume of a k -dimension parallelotope is achieved by calculating the Gramian matrix of M ; the Gramian matrix is defined as the inner product between all vectors of a matrix. Calculating the volume is done by taking the square-root of the Gramian matrix's determinant, which results in $\sqrt{\det(M^T M)}$.

In the work of [34], they define an angle between two subspaces of the Euclidian N -dimension space. The angle is defined as:

Definition 12. Let $A \in \mathfrak{R}^{N \times q}$ and $B \in \mathfrak{R}^{N \times p}$ be two bases subspaces of the Euclidean space \mathfrak{R}^N . Let $M_{i,j} = \langle A_i, B_j \rangle$, so $M = A^T B$. The angle φ between the subspace is defined as:

$$\cos\varphi = \frac{\sqrt{\det(M M^T)}}{\sqrt{\Gamma_A} \sqrt{\Gamma_B}}$$

where Γ_X stands for the Gram's determinant of X - $\Gamma_X = \det(X^T X)$

■

Our base is orthonormal, hence the Gram's determinant of the base equals 1. So, our divergence measure in this definition is reduced to $1 - \cos\varphi$, one minus the cosine of the angles between the subspaces.

Since the maximum value of the cosine function is one, the maximum value of the determinant is also one. The determinant reaches such a value when the values of all the cosines are one. This will happen only when all the angles between the subspaces are zero, meaning that one subspace is contained in the other. In a divergence measure, we wish to compute how far the objects are from being equal. This is the reason to subtract 1 by the square-root of $M^T M$'s determinant, which results in Definition 9.

6.2.2 The Trace measure

We concluded so far that the singular value of M represents the angles between the subspaces. One of the most natural ways to proceed is to measure M by some similar-invariant norm. The well-known Frobenius norm is such a measure; we will use its square. It is a well-known fact that the Frobenius norm of a matrix fulfills the following equation:

$$\|A\|_{fro} = \sqrt{\text{Trace}(A^T A)} = \sqrt{\sum_{i=1}^N \lambda_i^2}$$

where λ_i is the i singular value. In the case of equality between the graphs, all angles equal zero. Since the singular values of M are the cosines of the angles, the cosines will all be 1 and their sum will equal k - the minimum between the two dimensions of the subspace.

Since we want a divergence measure, we conclude the Trace measure defined by Definition 10.

Chapter 7

A Bound for the Trace measure

The bound described below arises from the matrix perturbation theory; here we present the proof of a relaxed version of the *Davis-Kahan Sin* Θ theory [35].

When a matrix is subject to perturbation, its spectral decomposition changes in two separate levels: both in its eigenvalues and in its eigenvectors. The theorem of Davis-Kahan bounds the amount of change in the eigenvectors in relation to the size of the perturbation and the way the eigenvalues of the matrix are scattered. Our interest is only with a weaker version of this theorem, the change in a group of eigenvectors and not in all the eigenvectors.

The main reason for stating the proof here is that it explains the theorem and what it means. The proof here is a combination of the proof of this theorem from [23], [33] and [36].

7.1 $AX - BX = Y$

We will present another theory before presenting the simpler version of *Davis-Kahan Sin* Θ theory [35].

Theorem 5. *Let A and B be normal matrices, and let $\sigma(A) \subset (\alpha, \beta)$ and $\sigma(B) \subset \mathfrak{R} \setminus (\alpha - \delta, \beta + \delta)$, $\delta > 0$. Then the solution X of the equation $AX - XB = Y$ satisfies the inequality*

$$\|X\| \leq \frac{1}{\delta} \|Y\| \quad (7.1.1)$$

where $\| \cdot \|$ is any unitary-invariant norm.

Proof: If A and B are diagonal with $\sigma(B) = [\lambda_1, \lambda_2, \dots, \lambda_N]$ and $\sigma(A) = [\sigma_1, \sigma_2, \dots, \sigma_M]$, then the equation is reduced to:

$$\sigma_i X_{i,j} - \lambda_j X_{i,j} = Y_{i,j}$$

So

$$X_{i,j} = \frac{Y_{i,j}}{\sigma_i - \lambda_j}$$

and the inequality holds.

If A or B are not diagonal, from their normality they have spectral decompositions $A = V\Lambda V^T$ and $B = U\Sigma U^T$, respectively. We can now write the equation:

$$Y = AX - XB = V\Lambda V^T X - XU\Sigma U^T$$

Multiplying from the left with V^T and from the right with U , we get:

$$\underbrace{V^T Y U}_{\hat{Y}} = \underbrace{\Lambda}_{\text{diagonal matrix}} \underbrace{V^T X U}_{\hat{X}} - \underbrace{V^T X U}_{\hat{X}} \underbrace{\Sigma}_{\text{diagonal matrix}}$$

We are back in the diagonal case:

$$\hat{X}_{i,j} = \frac{\hat{Y}_{i,j}}{\lambda_i - \sigma_j}$$

From the definition of δ :

$$\|\hat{X}\| \leq \frac{1}{\delta} \|\hat{Y}\|$$

However $\|\hat{X}\| = \|X\|$ since U, V are unitary, and the norm is unitary invariant. The same argument applies to \hat{Y} . Hence we arrive at the inequality. ■

Note that A and B can change roles, such that $R = BX - XA$.

7.2 A weak version of Davis-Kahan theory

We are now ready to state the main theory:

Theorem 6. *Let A, H be two symmetric matrices, and let $\tilde{A} = A + H$. Let $X = (X_1 X_2)$ be unitary matrices such that:*

$$\begin{pmatrix} X_1^T \\ X_2^T \end{pmatrix} A (X_1 X_2) = \begin{pmatrix} L_1 & 0 \\ 0 & L_2 \end{pmatrix}$$

in the same way $F = (F_1 F_2)$

$$\begin{pmatrix} F_1^T \\ F_2^T \end{pmatrix} \tilde{A} (F_1 F_2) = \begin{pmatrix} M_1 & 0 \\ 0 & M_2 \end{pmatrix}$$

If we assume that:

$$\sigma(M_1) \subset (\alpha, \beta)$$

and

$$\sigma(L_2) \subset \Re \setminus (\alpha - \delta, \beta + \delta)$$

when $\delta > 0$,

then for any unitary-invariant norm $\|\cdot\|$

$$\|\sin(\Theta(\Xi, \Psi))\| \leq \frac{\|H\|}{\delta} \tag{7.2.1}$$

where Ξ is the subspace spanned by the column of X_1 , and Ψ the same for F_1 .

Proof: We will define R as:

$$R = AF_1 - F_1 M_1$$

and hence

$$\begin{aligned} X_2^T R &= X_2^T AF_1 - X_2^T F_1 M_1 \\ &= X_2^T (X_1 L_1 X_1^T + X_2 L_2 X_2^T) F_1 - X_2^T F_1 M_1 \\ &= L_2 \underbrace{X_2^T F_1}_{\hat{X}} - \underbrace{X_2^T F_1}_{\hat{X}} M_1 \end{aligned}$$

According to Theorem 5:

$$\|X_2^T F_1\| \leq \frac{\|X_2^T R\|}{\delta}$$

We will regard each side separately, starting with the left side.

Because the norm is unitary invariant, and U_2 and V_1 in Theorem 4 are unitary, we can write:

$$\|X_2^T F_1\| = \|U_2^T X_2^T F_1 V_1\|$$

According to note 6.1.6, we may apply Definition 11.

Therefore, the left side is equal to:

$$\|\sin(\Theta(\Xi, \Psi))\|$$

The right side follows:

$$\begin{aligned} R &= AF_1 - F_1 M_1 \\ &= AF_1 - F_1 (F_1^T \tilde{A} F_1) \\ &= AF_1 - \tilde{A} F_1 \\ &= AF_1 - (A + H) F_1 \\ &= H F_1 \end{aligned}$$

so we get

$$\frac{\|X_2^T R\|}{\delta} = \frac{\|X_2^T H F_1\|}{\delta}$$

X and F are unitary, so the singular values of X_2 and F_1 are on the unit circle. Noting one character of the invariant norm:

$$\|AB\| \leq \|A\| \|B\|_2$$

$$\|AB\| \leq \|A\|_2 \|B\|$$

we may conclude that

$$\frac{\|X_2^T H F_1\|}{\delta} \leq \frac{\|H\|}{\delta}$$

Combining the two sides, we conclude:

$$\|\sin(\Theta(\Xi, \Psi))\| \leq \frac{\|H\|}{\delta}$$

■

7.3 Error bound for the Trace measure

We can now draw the bound, in form of the change between the graphs. One assumption must be made on the difference: it must be symmetric. This assumption is almost inherent, since we are dealing with an undirected graph and hence the change must also be symmetric.

Since the previous theorem is symmetric (A can take the role of $A + H$ and vice versa), we may regard the matrix with the smallest k as A . For ease of notation alone, we will assume that $\lambda_k \leq \lambda_{\bar{k}}$; otherwise the roles may be switched. We are dealing with undirected graphs, and hence all the relevant matrices are normal.

Theorem 7. *Let G, \tilde{G} be two graphs.*

Let L, \tilde{L} be the corresponding Laplacians, as was defined by Definition 4.

Let M be defined as in Definition 8.

Then

$$\sqrt{\Delta_{Trace}(G, \tilde{G})} \leq \frac{\|L - \tilde{L}\|_{fro}}{\delta}$$

where

$$\delta = \tilde{\lambda}_{\tilde{k}+1} - \lambda_k$$

Proof: We will set $E_1 = \Phi_{[1\dots k]}$ and $F_1 = \tilde{\Phi}_{[1\dots \tilde{k}]}$.

$$\sqrt{\Delta(G, \tilde{G})} = \sqrt{k - Trace(M^T M)} \quad (7.3.1)$$

$$= \sqrt{k - Trace\left(\left(E_1^T F_1\right)^T E_1^T F_1\right)} \quad (7.3.2)$$

$$= \sqrt{k - Trace\left(F_1^T E_1 E_1^T F_1\right)} \quad (7.3.3)$$

$$= \sqrt{k - Trace\left(V_1^T F_1^T E_1 U_1 U_1^T E_1^T F_1 V_1\right)} \quad (7.3.4)$$

$$= * \quad (7.3.5)$$

U_1, V_1 are the unitary matrices as in Theorem 4.

The last equality holds due to the similarity-invariant of the $Trace$ and to $Trace(AB) = Trace(BA)$.

$$* = \sqrt{k - Trace\left(\left(U_1^T E_1^T F_1 V_1\right)^T U_1^T E_1^T F_1 V_1\right)} \quad (7.3.6)$$

$$= \sqrt{k - \sum_{i=1}^k c_i^2} \quad (7.3.7)$$

$$= \sqrt{\sum_{i=1}^k 1 - c_i^2} \quad (7.3.8)$$

$$= \sqrt{\sum_{i=1}^k s_i^2} \quad (7.3.9)$$

$$= \|\sin(\Theta(\Xi, \Psi))\|_{fro} \quad (7.3.10)$$

$$= ** \quad (7.3.11)$$

All the equalities are direct derivations of Theorem 4.

$$** \leq \frac{\|H F_1\|_{fro}}{\delta} \quad (7.3.12)$$

$$\leq \frac{\|H\|_{fro}}{\delta} \quad (7.3.13)$$

$$= \frac{\|L - \tilde{L}\|_{fro}}{\delta} \quad (7.3.14)$$

The inequalities are due to Theorem 6 and its proof.

■

We concluded that this divergence is bound from above by the Frobenius norm of the distance between the two Laplacians divided by the gap between the subspaces. In this definition we combined

two different characteristics: one - the actual change in the weights of the graph, and the other - the gap, a characteristic that signifies how easily this graph is separated into k groups. The first characteristic takes into account each change, punishing more severely for bigger differences in values, as well as for a change of a certain vertex in respect to the others (due to taking the difference between the Laplacians instead of between the adjacency matrices). The second characteristic is divided by, in order to insure that if the graph is easily separated into groups, the eigenvectors corresponding to the small eigenvalues will not be effected (as long as the change does not add weight to the edges connecting the different groups). This can be better seen from 7.3.12, where the change is projected into the subspace spanned by these eigenvectors.

Chapter 8

Graph dynamics as a multivariate normal distribution

In this chapter we will show how the Trace divergence measure can be derived if the problem is defined by Definition 3.1.2. We had formulated the problem as a likelihood-ratio test - is the likelihood that a certain sample had come from an abnormal distribution significantly higher than the likelihood that it had come from the typical distribution. Note that since we cannot, and should not, assume anything on the abnormal distribution, any distribution may be considered as the abnormal one. For this purpose we will have to assume a specific distribution of the data.

As was previously pointed out, when examining a network at a given moment, two observations are possible: vertex activity vs. edge activity. We will assume two different distributions of vertex activity. In one, we will set an energy function on the graph and use Gibbs measure. In the second, we will assume that we observe edge activity, and from it deduce distribution of vertex activity.

8.1 Node activity by Gibbs measure

We will first develop a probability measure for a node-activity instance, assuming a certain energy function. For this we will assume that the structure of the network, the adjacency matrix of the graph, is given. The activity of the vertices will be denoted as v .

Motivated by Eq. (4.3.1), we will define an energy function to be $E(v) = \frac{1}{2} \sum_{i,j} W_{i,j} (v_i - v_j)^2$ - the weighted sum of difference between the activity of any two nodes. As in electricity, we will want the difference between vertices to contribute according to the connectivity between the vertices. When two vertices have a high correlation, e.g high connectivity, the activity in the node should be similar; so we will observe low energy when there is high correlation between the difference in activity of two vertices and their connectivity.

We will make an addition to the energy function, by measuring not the total activity but the extent by which it diverges from a certain basal activity. Now the Energy will be :

$$E(v) = \frac{1}{2} \sum_{i,j} W_{i,j} ((v_i - \bar{v}_i) - (v_j - \bar{v}_j))^2$$

where \bar{v} is the basal activity.

After defining the energy of the system, we can define the Gibbs measure $P(V = v) = \frac{1}{Z(\beta)} e^{-\beta E(v)}$.

In our case, the Gibbs measure is reduced to $P(V = v) = \left(\frac{\beta^N \det L}{(2\pi)^N} \right)^{\frac{1}{2}} e^{-\beta(v-\bar{v})^T L(v-\bar{v})}$, and thus appears as a well-defined distribution. Now we can see that the Laplacian in this context is to be thought of as the inverse of the covariance matrix of a multivariate normal distribution.

If we recall Property 5, we will notice a problem - some dimensions have an infinity variance. So the covariance matrix cannot be defined, and even in our notation is wrong since the determinant of a singular matrix is 0 - and so we do not get a probability measure. We will denote the number of zero-value eigenvalues, the number of the dimensions of infinity variance, as k .

The solution to this problem is to restrict the sample space, such that it will be perpendicular to the null space of the Laplacian. This will, of course, require a change in the normalization part of the distribution.

We will define the sample space to be $\Omega = \{v \in \mathfrak{R}^N | v \notin \text{span}\{\phi_1, \dots, \phi_k\}\}$. Note that $\bar{v} \in \Omega$ must be fulfilled as well. A direct outcome is that our sample space has now a dimension of $N - k$, so our distribution is $P(V = v) = \left(\frac{\beta^{N-k} \hat{\det} L}{(2\pi)^{N-k}}\right)^{\frac{1}{2}} e^{-\beta(v-\bar{v})^T L(v-\bar{v})}$. The $\hat{\det}$ here is a restricted determinant - using the equation $\det L = \prod_{i=1}^N \lambda_i$ we will restrict it to the non-zero eigenvalues, remembering that the eigenvalues are ordered in non-decreasing order.

We will define

$$\hat{\det} L = \prod_{i=k+1}^N \lambda_i \quad (8.1.1)$$

Let us look again at

$$\begin{aligned} L &= \Phi \Lambda \Phi^T \\ &= \sum_{i=1}^N \lambda_i \Phi_i \Phi_i^T \\ &= \sum_{i=k+1}^N \lambda_i \Phi_i \Phi_i^T \\ &= \Phi_{[k+1 \dots N]} \Lambda_{[k+1 \dots N, k+1 \dots N]} \Phi_{[k+1 \dots N]}^T \end{aligned}$$

We will define

$$Q = \Phi_{[k+1 \dots N]} \Lambda_{[k+1 \dots N, k+1 \dots N]}^{-\frac{1}{2}}$$

First, let us verify that we have a distribution:

$$\int_{v \in \Omega} P(V = v) dv = \int_{v \in \Omega} \left(\frac{\beta^{N-k} \hat{\det} L}{\pi^{N-k}}\right)^{\frac{1}{2}} e^{-\beta(v-\bar{v})^T L(v-\bar{v})} dv$$

We will change variable $v - \bar{v} = Qy$; note that $y \in \mathfrak{R}^{N-k}$ and that $\det Q = (\hat{\det} L)^{-\frac{1}{2}}$.

$$\begin{aligned} &= \left(\frac{\beta}{\pi}\right)^{\frac{N-k}{2}} \int_{-\infty}^{\infty} e^{-\beta y^T y} dy \\ &= \left(\frac{\beta}{\pi}\right)^{\frac{N-k}{2}} \int_{-\infty}^{\infty} e^{-\beta \sum_{i=1}^{N-k} y_i^2} dy \\ &= \left(\frac{\beta}{\pi}\right)^{\frac{N-k}{2}} \int_{-\infty}^{\infty} \prod_{i=1}^{N-k} e^{-\beta y_i^2} dy \\ &= \left(\frac{\beta}{\pi}\right)^{\frac{N-k}{2}} \int_{-\infty}^{\infty} e^{-\beta y_1^2} dy_1 * \int_{-\infty}^{\infty} e^{-\beta y_2^2} dy_2 * \dots * \int_{-\infty}^{\infty} e^{-\beta y_{N-k}^2} dy_{N-k} \end{aligned}$$

Remember that $\int_{-\infty}^{\infty} e^{-\beta y^2} dy = \sqrt{\frac{\pi}{\beta}}$

$$= \left(\frac{\beta}{\pi}\right)^{\frac{N-k}{2}} \prod_{i=1}^{N-k} \sqrt{\frac{\pi}{\beta}} = 1$$

We have shown that this is a well-defined distribution.

We will now present another node distribution.

8.2 Node activity by incident matrix

When observing edge activity, the simplest data that can be collected is the activity of the connection that an edge represents, for each edge in every time-unit. For example, if we examine the commerce network, where each state is a vertex, the edges' activity will be the total trade (in currency) between two states during a specific time-unit.

When gathering the data in such a way, we have a vector in which each entry signifies a different edge. The data in this form is only partial; we lose the simple but vital information of which edges share the same vertex.

Instead of this representation, we will hold the data in a matrix in which rows signify the edges, and columns - the vertices. Each edge will arbitrarily be set a direction; it will become clear later that this selection has no effect on our calculations. For each edge, a function of the activity and direction will be held in the columns that are its end-points of this edge. This matrix is a variation of the incidence matrix [37].

More formally: Let W be the activity function as defined by Definition 1. We will set a certain order of the edges $\sigma : E \rightarrow \{1, \dots, M\}$, such that if $e_{i,j} \neq e_{k,l}$ then $\sigma(e_{i,j}) \neq \sigma(e_{k,l})$. Note that $e_{i,j} = e_{j,i}$, so there is only one such edge. Let S be the function that sets the direction of the vertex $S : E \times V \rightarrow \{-1, 0, 1\}$. For each edge $e_{i,j}$, the function S assigns the role of source to one of its end-points, so $S(e_{i,j}, v_i) = 1$; to the other end-point vertex, it assign the role of destination vertex, so $S(e_{i,j}, v_j) = -1$; for any other vertex $S(e_{i,j}, v_k) = 0$.

We are now ready to define our matrix. For each time t we will gather the activity of the network in a matrix $A(t)$. Let $A(t)$ be an $M \times N$ matrix, where M is the number of edges and N the number of vertices. $A(t)$ will be defined as follows:

$$A_{\sigma(e_{k,l}),i}(t) = \begin{cases} W(e_{k,l}, t) - \frac{\sum_{i=1}^T W(e_{k,l}, i)}{T} & S(e_{k,l}, v_i) = 1 \\ - \left(W(e_{k,l}, t) - \frac{\sum_{i=1}^T W(e_{k,l}, i)}{T} \right) & S(e_{k,l}, v_i) = -1 \\ 0 & S(e_{k,l}, v_i) = 0 \end{cases}$$

A holds the deviation of edge activity from its mean, along the entire sampled time period.

Lemma 1. Let $L(t) = A^T(t)A(t)$.

$L(t)$ satisfies all Laplacian properties - Proposition 1.

Proof: First, let us examine how L is built.

$$\begin{aligned}
[L(t)]_{i,j} &= [A(t)^T A(t)]_{i,j} \\
&= \sum_{k,l \text{ such that } e_{k,l} \in E} A_{\sigma(e_{k,l}),i}(t) A_{\sigma(e_{k,l}),j}(t) \\
&= \begin{cases} \sum_{l=1}^N \left(W(e_{i,l}, t) - \frac{\sum_{\dot{t}=1}^T W(e_{i,l}, \dot{t})}{T} \right)^2 & i = j \\ - \left(W(e_{i,j}, t) - \frac{\sum_{\dot{t}=1}^T W(e_{i,j}, \dot{t})}{T} \right)^2 & e_{i,j} \in E \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

We will define the two matrices D, W :

Let D be the diagonal of $L(t)$ for some t ; more formally, $D_{i,i} = L_{i,i}$ for all i , and $D_{i,j} = 0$ for all $i \neq j$.

We will define $W = D - L$.

We can now conclude:

Property 1 may easily be seen from the definition $e_{i,j} = e_{j,i}$.

Property 2 - its proof goes along the same lines, together with the fact that $\sum_j W_{i,j} = D_{i,i}$ for all i .

Property 3 may be seen from the fact that $L(t)$ is a Gramian matrix and hence positive-semidefinite.

Property 4 is derived directly from Property 3.

Property 5 - its proof may be repeated in the same way; note that $W_{i,j} = 0$ if $e_{i,j} \notin E$. ■

Lemma 2. Let $L = \frac{1}{T} \sum_t^T L(t)$.

L satisfies all Laplacian properties 1.

Proof:

Property 1 may easily be seen from the fact that $L(t)$ is symmetric.

Property 2 - every $L(t)$ satisfies this property and hence the sum satisfies the property as well.

Property 3 - $L(t)$ is positive-semidefinite and the sum of positive-semidefinite matrices is also positive-semidefinite.

Property 4 is derived directly from Property 3.

Property 5 - L does not represent any specific graph, hence the property has no meaning. ■

We argue that L may be regarded as an improved covariance matrix. If one wishes to compute the covariance matrix of vertex activity, the vertex activity vector v must be found first.

$$v_i(t) = \sum_{j=1}^N W(e_{i,j}, t)$$

(In this format, edge activity is seen as activity of the two end-point vertices).

We will define the mean vector $\bar{v} = \frac{1}{T} \sum_{\dot{t}=1}^T v(\dot{t})$.

The biased estimation of the covariance matrix is:

$$\Sigma = \frac{1}{T} \sum_{t=1}^T (v(t) - \bar{v})(v(t) - \bar{v})^T$$

Let us look at the i, j coordinate:

$$\begin{aligned}
[\Sigma]_{i,j} &= \frac{1}{T} \sum_{t=1}^T (v_i(t) - \bar{v}_i) (v_j(t) - \bar{v}_j) \\
&= \frac{1}{T} \sum_{t=1}^T \left(\sum_{k=1}^N \mathbf{W}(e_{i,k}, t) - \frac{1}{T} \sum_{\dot{i}=1}^T \sum_{l=1}^N \mathbf{W}(e_{i,l}, \dot{t}) \right) \left(\sum_{k=1}^N \mathbf{W}(e_{k,j}, t) - \frac{1}{T} \sum_{\dot{i}=1}^T \sum_{l=1}^N \mathbf{W}(e_{l,j}, \dot{t}) \right) \\
&= \frac{1}{T} \sum_{t=1}^T \left(\sum_{k=1}^N \left[\mathbf{W}(e_{i,k}, t) - \frac{1}{T} \sum_{\dot{i}=1}^T \mathbf{W}(e_{i,k}, \dot{t}) \right] \right) \left(\sum_{k=1}^N \left[\mathbf{W}(e_{k,j}, t) - \frac{1}{T} \sum_{\dot{i}=1}^T \mathbf{W}(e_{l,j}, \dot{t}) \right] \right) \\
&= \frac{1}{T} \sum_{t=1}^T \left(\sum_{k,l \text{ such that } e_{k,l} \in E} S(e_{k,l}, v_i) A_{\sigma(e_{k,l}),i}(t) \right) \left(\sum_{k,l \text{ such that } e_{k,l} \in E} S(e_{k,l}, v_j) A_{\sigma(e_{k,l}),j}(t) \right)
\end{aligned}$$

Comparing this with

$$\begin{aligned}
[L]_{i,j} &= \frac{1}{T} \sum_t L(t) \\
&= \frac{1}{T} \sum_t \sum_{k,l \text{ such that } e_{k,l} \in E} A_{\sigma(e_{k,l}),i}(t) A_{\sigma(e_{k,l}),j}(t)
\end{aligned}$$

- there is only one subtle change between the equations: whether we first sum over the edges, or rather first multiply the activity. This can be confirmed if we notice that for all $e_{i,j} \in E$ and for $k \in \{i, j\}$ $S(e_{i,j}, v_k)^2 = 1$. If $k \notin \{i, j\}$, it is easy to see that $S(e_{i,j}, v_k) = 0$ as well as $A_{\sigma(e_{i,j}),k} = 0$, for all i, j .

The notion behind the covariance is to measure the strength by which two variables correlate with each other. The weight of the edge in the graph represents how much the two vertices effect one another, and hence correlate. So by losing the information as to which edge is responsible for the activity, you add noise by regarding as equals the event that one vertex was active and hence its neighbor was active and the case that by chance the two vertices happened to be active simultaneously. Hence, the Laplacian matrix, which takes into account only correlations that are not due to chance but to active effect of one vertex on the other, is better in our opinion.

The difference between the two matrices may be summed to the following: The covariance matrix examines the activity of the vertices, ignoring the fact that it is deduced from edge activity; since the edge activity dictates a structure - a graph, by ignoring it one ignores the information that the activity came from a graph. The Laplacian, on the other hand, takes this fact into account, and hence better represents graph activity.

The natural next step is to define a variation of the multivariate normal distribution of the vertex activity v . For the role of the covariance matrix, we will use the Laplacian. Remembering Property 4, one cannot take the inverse of the Laplacian. For this purpose, we will define a pseudo-inverse of a matrix and mark it by \dagger . One way to invert a matrix is to take the inverse of all its eigenvalues. When some of the eigenvalues are zero, the matrix is singular, and cannot be inverted. The pseudo-inverse takes the inverse of all non-zero eigenvalues while keeping the zero eigenvalues unchanged - inverting only the image of the transformation; for more details see [38].

We can now define the probability.

$$P(V = v) = \frac{1}{(2\pi)^{\frac{N}{2}} \hat{\det}(L)^{\frac{1}{2}}} \exp \left(-\frac{1}{2} (v - \mu)^T L^\dagger (v - \mu) \right)$$

where \hat{det} is as defined above in 8.1.1, and μ is the mean of the vertex activity. The proof that this indeed is a distribution follows the same lines as in the previous section. Note that we have assumed the naive basic assumption that, given the distribution, the samples (edge activities) are independent. This clearly is a false assumption, but one which we will make for simplicity's sake.

8.3 Kullback-Leibler divergence and maximum likelihood

We have defined two distributions of node activity. We are now ready to return to the likelihood-ratio test. We will denote the typical graph as L , and the generated graph as \tilde{L} . So the test is formulated as

$$c \geq \frac{\text{Likelihood}(P_{\tilde{L}}|X)}{\text{Likelihood}(P_L|X)} = \frac{P_{\tilde{L}}(X)}{P_L(X)}$$

where c is a threshold such that if exceeded will declare the new probability $P_{\tilde{L}}$, and hence the new sample, as abnormal. Note that by definition $P_{\text{typical}} = P_L$.

\tilde{L} defines probability, not only on the activities observed in the new sample, but on the entire probability space. Therefore, it may be more informative to test the new probability on the entire space. This means taking the expectation of the likelihood-ratio test with the probability defined by \tilde{L} . So now the test will be defined as

$$\mathbf{E}_{P_{\tilde{L}}} \left[\frac{P_{\tilde{L}}(x)}{P_L(x)} \right]$$

If we will examine the $\log \text{Likelihood}$ instead, we will get the known Kullback-Leibler divergence:

$$DKL(P_{\tilde{L}}, P_L) = \int_{v \in \Omega} P_{\tilde{L}}(v) \log \frac{P_{\tilde{L}}(v)}{P_L(v)}$$

If the DKL will be greater than a certain threshold, we will conclude that indeed the first distribution is not some deviation from the second distribution, but rather an entirely different distribution - an anomalous one.

Before finding the DKL , we need to re-examine the equation. In DKL we execute an integral over the entire probability space. So for the DKL to be well-defined, we need to assume that the probability spaces of the two distributions merge and are the same subspace. Note that the same problem occurs in the likelihood-ratio test itself, where we need to assume that the typical distribution is defined over the entire new sample data.

This assumption for DKL may be otherwise phrased as each graph being connected. This is not at all far-fetched, since in real-life graphs, the different groups will always be connected, if only due to noise. So, the DKL is now well-defined, since the only subspace that is not included in the probability space of the two distributions, is the one spanned by the all-one vector.

Real-life networks are relatively easy to cluster [10], so their first eigenvalues of their Laplacian will be close to zero. Hence, the probability of a sample to be perpendicular to all the corresponding eigenvectors, tends to zero. If these subspaces of the distributions do not merge, the DKL will be governed by the events in which $P_{\tilde{L}}$ is probable while P_L is near zero, meaning the events which are perpendicular to the first eigenvectors of the typical distribution but not of the new distribution. Comparing this subspace gives us a good approximation of the DKL , which is exactly our divergence measure.

We will now find the DKL of the first distribution; the DKL for the second goes along the same lines. The basal activity will be denoted as v_L and $v_{\tilde{v}}$ for the typical and new distributions, respectively.

$$\begin{aligned}
DKL(P_{\tilde{L}}, P_L) &= \int_{v \in \Omega} P_{\tilde{L}}(v) \log \left[\frac{\left(\hat{\det} \tilde{L} \right)^{\frac{1}{2}} e^{-\beta(v - \bar{v}_{\tilde{L}})^T \tilde{L}(v - \bar{v}_{\tilde{L}})}}{\left(\hat{\det} L \right)^{\frac{1}{2}} e^{-\beta(v - \bar{v}_L)^T L(v - \bar{v}_L)}} \right] dv \\
&= \log \prod_{i=k+1}^N \left(\frac{\tilde{\lambda}_i}{\lambda_i} \right)^{\frac{1}{2}} + \beta \mathbf{E}_{P_{\tilde{L}}} \left[-(v - \bar{v}_{\tilde{L}})^T \tilde{L}(v - \bar{v}_{\tilde{L}}) + (v - \bar{v}_L)^T L(v - \bar{v}_L) \right] \\
&= \log \prod_{i=k+1}^N \left(\frac{\tilde{\lambda}_i}{\lambda_i} \right)^{\frac{1}{2}} + \beta \left(\frac{\beta^{N-k} \hat{\det} \tilde{L}}{(\pi)^{N-k}} \right)^{\frac{1}{2}} \\
&\quad \int_{v \in \Omega} e^{-\beta(v - \bar{v}_{\tilde{L}})^T \tilde{L}(v - \bar{v}_{\tilde{L}})} \left((v - \bar{v}_L)^T L(v - \bar{v}_L) - (v - \bar{v}_{\tilde{L}})^T \tilde{L}(v - \bar{v}_{\tilde{L}}) \right) dv \\
&= *
\end{aligned}$$

The base of the log is e , and we will measure the result in nats.

We perform the variable change $v - \bar{v}_{\tilde{L}} = \tilde{D}y$:

$$\begin{aligned}
* &= \log \prod_{i=k+1}^N \left(\frac{\tilde{\lambda}_i}{\lambda_i} \right)^{\frac{1}{2}} + \beta \left(\frac{\beta}{\pi} \right)^{\frac{N-k}{2}} \\
&\quad \int_{-\infty}^{\infty} e^{-y^T y} \left((y \tilde{D}^T - \bar{v}_L + \bar{v}_{\tilde{L}})^T L(\tilde{D}y - \bar{v}_L + \bar{v}_{\tilde{L}}) - y^T y \right) dy \\
&= \log \prod_{i=k+1}^N \left(\frac{\tilde{\lambda}_i}{\lambda_i} \right)^{\frac{1}{2}} \\
&\quad + \beta \left(\frac{\beta}{\pi} \right)^{\frac{N-k}{2}} \int_{-\infty}^{\infty} e^{-\beta \sum_{i=1}^{N-k} \beta y_i^2} \sum_{j,l,s,i=1}^{N-k} y_i \tilde{D}_{s,i} L_{s,l} \tilde{D}_{l,j} y_j - e^{-\sum_{i=1}^{N-k} \beta y_i^2} \sum_{i=1}^{N-k} y_i^2 dy \\
&\quad + \beta(-\bar{v}_L + \bar{v}_{\tilde{L}})^T L(-\bar{v}_L + \bar{v}_{\tilde{L}}) \\
&= \log \prod_{i=k+1}^N \left(\frac{\tilde{\lambda}_i}{\lambda_i} \right)^{\frac{1}{2}} \\
&\quad + \beta \left(\frac{\beta}{\pi} \right)^{\frac{N-k}{2}} \left(\sum_{j,i=1}^{N-k} \int_{-\infty}^{\infty} e^{-\beta \sum_{i=1}^{N-k} y_i^2} y_i y_j \sum_{l,s=1}^{N-k} \tilde{D}_{s,i} L_{s,l} \tilde{D}_{l,j} dy - \sum_{i=1}^{N-k} \int_{-\infty}^{\infty} e^{-\beta \sum_{i=1}^{N-k} y_i^2} y_i^2 dy \right) \\
&\quad + \beta(\bar{v}_{\tilde{L}} - \bar{v}_L)^T L(\bar{v}_{\tilde{L}} - \bar{v}_L) \\
&= **
\end{aligned}$$

y_i are uncorrelated Gaussian variables.

$$\begin{aligned}
** &= \log \prod_{i=k+1}^N \left(\frac{\tilde{\lambda}_i}{\lambda_i} \right)^{\frac{1}{2}} \\
&\quad + \beta \left(\frac{\beta}{\pi} \right)^{\frac{1}{2}} \left(\sum_{i=1}^{N-k} \int_{-\infty}^{\infty} e^{-\beta y_i^2} y_i^2 \sum_{l,s=1}^{N-k} \tilde{D}_{s,i} L_{s,l} \tilde{D}_{l,i} dy_i - \sum_{i=1}^{N-k} \int_{-\infty}^{\infty} e^{-y_i^2} y_i^2 dy_i \right) \\
&\quad + \beta(\bar{v}_{\tilde{L}} - \bar{v}_L)^T L(\bar{v}_{\tilde{L}} - \bar{v}_L) \\
&= ***
\end{aligned}$$

y_i are Gaussian variables with a variance of 1.

$$\begin{aligned} *** &= \log \prod_{i=k+1}^N \left(\frac{\tilde{\lambda}_i}{\lambda_i} \right)^{\frac{1}{2}} + \beta \text{Trace} \left(\tilde{D}^T L \tilde{D} \right) - \beta(N - k) + \beta(\bar{v}_{\tilde{L}} - \bar{v}_L)^T L (\bar{v}_{\tilde{L}} - \bar{v}_L) \\ &= \log \prod_{i=k+1}^N \left(\frac{\tilde{\lambda}_i}{\lambda_i} \right)^{\frac{1}{2}} + \beta \text{Trace} \left(L \tilde{L}^\dagger \right) - \beta(N - k) + \beta(\bar{v}_{\tilde{L}} - \bar{v}_L)^T L (\bar{v}_{\tilde{L}} - \bar{v}_L) \end{aligned}$$

The same can be done for the second distribution, with the result of

$$DKL(P_{\tilde{L}}, P_L) = \log \prod_{i=k+1}^N \left(\frac{\lambda_i}{\tilde{\lambda}_i} \right)^{\frac{1}{2}} + \text{Trace} \left(L^\dagger \tilde{L} \right) - (N - k) + (\bar{v}_{\tilde{L}} - \bar{v}_L)^T L^\dagger (\bar{v}_{\tilde{L}} - \bar{v}_L) \quad (8.3.1)$$

8.4 From Kullback-Leibler divergence to our divergence measure

The DKL of the two distributions is effected by four factors: the means \bar{v} , a constant, the eigenvalues and the eigenvectors. We again encounter the same question: are all these features relevant for the detection of anomalies, or do they describe changes of the distributions that are of no interest.

The part of the DKL that includes the mean vector, can be regarded as the Mahalanobis distance between the means of the two distributions. The main problem with the mean is that it is very sensitive to scaling, the multiplication of the mean by a constant. As we want to be able to ignore such irrelevant changes while still being able to detect small structural changes, we will omit this part from our criteria function.

Regarding the constant, its value depends only on the null space of the two probabilities. As was discussed above, the only subspace not included in the probability space is the space spanned by the all-one vector. Hence all probabilities will have the same value and therefore have no use in a divergence function.

The eigenvalues appear in two different parts of DKL . In the first part, they are used to compare the volumes of the transformations, checking the ratio between their determinants. In the second part, they may be thought of as weights to the eigenvectors. As was described in the case of the mean and explained previously in 5.2, we will use the eigenvalues only to distinguish between the two sets of eigenvectors.

We are left with the following measure for the first node activity distribution:

$$\begin{aligned} D(P_{\tilde{L}}, P_L) &= \text{Trace}(L \tilde{L}^\dagger) \\ &= \text{Trace}(L^T \left(\tilde{L}^\dagger \right)^T) \\ &= \text{Trace}(\Phi \Lambda \Phi^T \tilde{\Phi} \tilde{\Lambda}^\dagger \tilde{\Phi}^T) \end{aligned}$$

As discussed above we want to use the eigenvalues only for the selection of the eigenvectors, and hence, the selected eigenvectors will be the ones which have a significant weight. For the normal Laplacian L , we will remove the eigenvectors corresponding to the near-zero eigenvalues. Noting that the generated Laplacian eigenvalues matrix $\tilde{\Lambda}$ is in inverse, the selected eigenvectors will be the ones which correspond to the near-zero eigenvalues. We will set k to be as defined by Definition 7 - the index of the eigenvalues distinguishing between the near-zero eigenvalues and the rest. The measure is reduced to:

$$D(P_{\tilde{L}}, P_L) = \text{Trace}(\Phi_{[k+1..N]} \Phi_{[k+1..N]}^T \tilde{\Phi}_{[1..k]} \tilde{\Phi}_{[1..k]}^T)$$

Let $Q = \Phi_{[k+1..N]}^T \tilde{\Phi}_{[1..k]}$, then

$$D(P_{\tilde{L}}, P_L) = \text{Trace}(QQ^T)$$

Now, bearing Theorem 4 in mind, Φ will take the role of X here, and $\tilde{\Phi}$ - the role of Y . So, in our context Q is $X_2^T Y_1$, and denoting its non-zero singular values as S , by Eq. (6.1.6) we have

$$Q = U_2 \begin{pmatrix} S \\ 0 \end{pmatrix} V_1^T$$

We will denote

$$D = \begin{pmatrix} S \\ 0 \end{pmatrix}$$

$$\text{Trace}(QQ^T) = \text{Trace}(U_2 D V_1^T V_1 D^T U_2^T) \quad (8.4.1)$$

$$= \text{Trace}(DD^T) \quad (8.4.2)$$

$$= \text{Trace}(SS^T) \quad (8.4.3)$$

Using again Theorem 4 we have:

$$S^2 = I - C^2$$

Where $C = \Phi_{[1\dots k]}^T \tilde{\Phi}_{[1\dots k]}$ hence

$$D(P_{\tilde{L}}, P_L) = \text{Trace}(I - C^2) \quad (8.4.4)$$

$$= k - \text{Trace}(M^T M) \quad (8.4.5)$$

where M is defined as in Definition 8. We get exactly our Trace measure, as defined in Definition 10. If we repeat the process for the second distribution, we will get the same result.

Chapter 9

Our Algorithm

The algorithm we present in this work will try to detect the times in which the network/graph behaves in an anomalous way. For such an algorithm, two inputs are necessary: the first, a function that returns the sampled network at time t ; and the other, the sensitivity parameter which controls the ration between missing and false alarm. In our notation, the sampling function will be denoted as `Sample(G)` and the sensitivity parameter will be a .

We will present our algorithm by dividing it into a number of methods. Each method has a different roll in the process.

9.1 The basic method

The first method is essentially similar to a service method. First, it initializes the Detector, and then, for each time step, it samples the network and calls the Detector's detect method. If the Detector finds an anomaly, the method raises an alarm. Note that this method is general for all detecting algorithms.

Method 1.

```
Detecting(Sample(G), a) {
    initialize Detector;
    while(1) {
        S = Sample(G);
        alarm = Detector.detect(S, a);
        if(alarm == abnormal) {
            rise an alarm;
        }
    }
}
```

9.2 The initialization and detect methods

We will describe the two methods mentioned above. We will start with the initialization method, and through it describe the different variables of the Detector. Note that the variables of the detector are saved; they are the object variables and hence exist the entire time that the object exists.

Every system has a different level of variability. Hence, the divergence between the different normal samples of one system, differs from that of other systems - having a different variance and mean. Therefore, we need to learn these variables for each system.

The mean and variance are denoted as m and v , respectively. In this method we initialize them to zero. The `counter` is also initialized to zero; its roll is to count the times that the detect method was

called, and by this help calculate the mean and variance. The `learn` variable is a parameter which controls the duration of the period in which we consider all points on the graph as normal - the learning period. There is one more variable that is saved from call to call, `E` - the approximation of the underlying graph.

Method 2.

```
Initialize() {
    counter = 0;
    m = 0;
    v = 0;
    learn = 10;
}
```

The detect method has three different missions. The first, to acquire the divergence between the sampled graph and the approximation to the generating graph. The second, having that divergence, to decide if the sample is anomalous - is the new sample too far to be considered as normal. The third, to update our perspective of the system.

The first step, of acquiring the divergence, is achieved by method 4 (see below). Having the divergence between the center graph and the sampled graph, we can now check whether this sample is anomalous. If this divergence is not within the normal range, but rather is too high, it means that the graphs are dissimilar to each other, and so we conclude that the sample is anomalous. The decision is performed by checking whether the divergence is higher than a times the standard deviation. In the learning period we do not raise an alarm, and hence the method will always return `normal`.

The third part is a simple gathering of statistics of the typical divergence distribution and the center of the normal graphs. As noted before, for the divergence distribution we gather the mean and variance. The center graph is simply the mean of the former samples.

Method 3.

```
detect(S, a) {

    if(counter == 0) {
        E=S;
        counter++;
        return normal;
    }
    dis = getDistance(S, E);
    if(dis-m > a*(sqrt(v)) && counter > learn) {
        alarm = abnormal;
        return abnormal;
    }
    counter++;
    E = E*(counter-1)/(counter)+S*1/counter;
    d = dis - m;
    m = m + d/counter;
    v = (v*(counter-1)+d*(dis-m))/counter;
    return normal;
}
```

9.3 getDistance method

In our algorithm, the graphs are represented as Laplacian. Three different definitions for the Laplacian have been given: Definition 4, Definition 5 and Definition 6. The `lap(S)` function implements one of these definitions.

In our divergence measure, the key parameter is the matrix M as defined by Definition 8. The first step in generating M is by finding the appropriate k as defined by Definition 7. We diverge from this definition by taking the log of the eigenvalues before comparing them. In other words, we look for the maximum ratio, and not difference, between two consecutive eigenvalues (remember that the eigenvalues are sorted in nondecreasing order). This choice can be explained by looking at the Eq. (4.3.2). Considering the result graph as an approximation of e^{-L} , the eigenvalues are $e^{-\lambda}$; hence for comparing the eigenvalues of L , we need to take the logarithm of the eigenvalues.

Having the two k 's, we extract the first k eigenvectors for each Laplacian. Note that we do not need all the eigenvectors; this fact will have considerable effect on the algorithm efficiency when monitoring large networks, since finding all the eigenvectors may be a tedious task, while finding the first k may be far more efficient. M is now calculated according to Definition 8.

The last step is to evaluate the magnitude of the matrix M . This is accomplished by one of the two divergence measures defined by Definition 9 or Definition 10. The function `divergence(M)` implements one of these definitions. We summarize all in the following method:

Method 4.

```
getDivergence(S, E) {
  // generate M
  L = lap(E);
  eL = eigenvalues of L;
  kL = max(log(eL_2) - log(eL_1), log(eL_3) - log(eL_2), ...
          , log(eL_N) - log(eL_{N-1}));

  Le = lap(S);
  eLe = eigenvalues of Le;
  V1 = the first kL eigenvectors of L
  kLe = max(log(eLe_2) - log(eLe_1), log(eLe_3) - log(eLe_2), ...
           , log(eLe_N) - log(eLe_{N-1}));
  V2 = the first kLe eigenvectors of Le

  if(kL > kLe){
    M = V1^T * V2;
  }else{
    M = V2^T * V1;
  }
  //acquire the divergence
  return divergence(M);
}
```

As noted above, the first method is a general one and may be thought of as a standard to this problem. The `initialize` and the `detect` methods are not specific for our divergence measure. They build on the assumption that the correct way to identify anomalies is by a certain divergence between the approximate normal graph and the new sample. In this work we did not optimize these methods and more sophisticated ones may be found, especially concerning the criteria for considering a sample as anomalous.

Furthermore, in this format the methods assume that the generating network is static - does not change over time. A simple extension of these methods is to consider not a single center graph but rather several such graphs, with certain limitations for changing from one model to another (an analog to an HMM with several states). Another simple extension is to allow the graph to slowly change/drift. This can easily be achieved by counting the last normal samples more than their fair share, which results in neglect of the old samples. Note that our algorithm is completely unsupervised. You can think of an online algorithm, in which according to the feedback from the user, a more accurate threshold is set.

We believe that the part of our algorithm which is least affected by a specific choice of network is the `getDistance`. In this method we tried to capture the more fundamental features of the graph, and hence we believe that it useful to many networks. In the next section we will demonstrate the quality of our divergence measure and our algorithm.

Chapter 10

Methods

The best way to test a detection algorithm is to apply it to the specific system one is trying to monitor. As the goal of this work is to find some universal technique for detecting anomalies in networks in general, and not in any specific system, we will not test our algorithm on any single specific network, but rather on several models of networks. In order to do this, we must first understand what the model which best describes real-world networks is.

10.1 The network model

The most basic model is the random graph by Erdős-Rényi, in which all edges have the same probability of belonging to the N-vertices graph. This model gives an equal probability to all N-vertices graphs with a fixed number of edges. As seen in the introduction, the graph thus generated will not, with high probability, hold most of the properties that are observed in real-world networks. Moreover, throughout this work we have used some of these properties to justify our divergence measure. Networks that have an underlying structure induce their first eigenvectors to hold fundamental information. An example of one such underlying structure may be clusters (communities), a property observed in real-world networks and lacking in this model; thus, the fact that the generated graph lacks this property is crucial to our work.

For this reason, we take one of the following three models as our network model: The first two are the scale-free network model by Barabási, and the well-known small-world graph. The third model used is a high-clustered graph. We will now describe each model, and afterwards will present the results - mainly on the high-clustered graph.

10.1.1 Barabási model

The Barabási-Albert model was invented to generate scale-free networks. In order to do so, the following algorithm was suggested: First, a random network of size m_0 is constructed (the network is usually a clique, we use this option). Each vertex other than the first m_0 vertices chooses m vertices to be its neighbors, by the following distribution: $P(e_{i,j}) = \frac{d_j}{\sum_{k=1}^{i-1} d_k}$. This technique induces a preferential attachment to vertices with high degree, by this creating a scale-free network. In all our trials we used $m_0 = 4$ and $m = 3$ where the graph size was $N = 400$. One must note that this network is very poorly clustered - high-clustering being a feature observed in many real-world networks. Moreover, the generated network does not in any way have an underlying structure inducing any meaningful small eigenvectors. Hence, we do not expect our algorithm, and specifically our divergence measures, to apply to this model.

10.1.2 The small-world model

The small-world model is a very specific graph. The simplest way to describe how it is generated is by placing all the vertices on a circle. Each vertex is a neighbor of all D vertices before or after it in the circle. In order to generate the small-world phenomena, each vertex has a probability of p to be a neighbor of any vertex in the circle (not only its immediate neighbors). In all our trials we used $D = 3$ and $p = 0.1$ where the graph size was $N = 400$. This graph, although not highly-clustered, has a very distinct structure, and hence its small eigenvector holds vital information. We expect this model to work well with our algorithm.

10.1.3 The high-cluster model

A high-cluster graph is a graph which can be uniquely divided into c clusters, such that the probability of belonging to the graph is much higher for an edge with both end-points in the same cluster, than that of an edge with end-points in different clusters. We induce this feature on the graph by setting the probability of an edge between clusters to be $pOut$. Inside each cluster, we set the mean degree of the vertices to be D .

A graph with N vertices will be divided into c clusters by the following procedure: Each cluster size is denoted by c_i and its mean size is set by $m_i = \frac{N - \sum_{j=0}^{i-1} c_j}{c-i}$, $i \in \{0, \dots, N-1\}$. Now the cluster size is set to be $c_i = e_i + 0.5 * random * e_i$, where $random$ is a number uniformly selected between -1 and 1 . Regardless of the division to clusters, each vertex has a probability of $pOut$ to have an edge between it and every other vertex - the between-clusters edges. Inside the clusters, as previously noted, each vertex has a probability of $\frac{D}{c_i}$ to have an edge between it and each of the cluster vertices. Since we are dealing with a weighted graph, we set the weight for each edge in the graph to be $W(e_{i,j}) = random * (Wh - Wl) + Wl$, where $random$ is a number uniformly selected between 0 and 1 , and Wh and Wl are the maximum and minimum weights, respectively. If not otherwise stated, we used the following parameters' values in all our trials: $N = 400$, $D = 5$, $pOut = 0.001$, $Wh = 1$, $Wl = 0.1$.

From all the models here presented, the high-cluster model seems to best meet our demands. On the one hand, it guarantees the crucial underlying structure - a feature lacking from both Barábsi and Erdős-Rényi models. On the other hand, it is not as restrict as the small-world model. For this reason, most of our trials will be on this model.

An additional note must be made. In the literature, many models for real-world networks have been suggested. We did not find any model which holds both features - scale-free and highly-clustered. Most models that are highly-clustered and create communities, have numerous parameters that have to be configured and the effect of which on the performance of our algorithm has to be correctly understood. Moreover, other than the three models presented, we could not find any specific model which leaves its mark on the community, and hence prefer our more simplified model.

10.2 Sampling from graphs

After generating the graph, we need to find a technique to sample it - to simulate an activity of the graph. In all our trials we observed edge activity - the times in which the edge is used by the entities it connects. The simplest way to simulate an activity of a graph is by a random walk on it. On a second look, one finds a fundamental shortcoming of this method. The choice of the first vertex is crucial; according to it a certain cluster is sampled, whereas other clusters could be sampled not at all; with the result of being oblivious to changes in or between these clusters.

The technique used here handles this disadvantage. The first vertex is sampled from the stationary distribution of the graph - the left eigenvector of the graph's adjacency matrix. After the first vertex is

sampled, an edge is sampled according to the different weights of the first vertex's edges. This process is repeated for every edge sampled. The number of edges we sample is denoted by T . Hence, on the one hand - we will not stay in specific cluster, and on the other hand - we sample according to the graph's distribution. From a different point of view, this process is simply a way to sample edges from a graph.

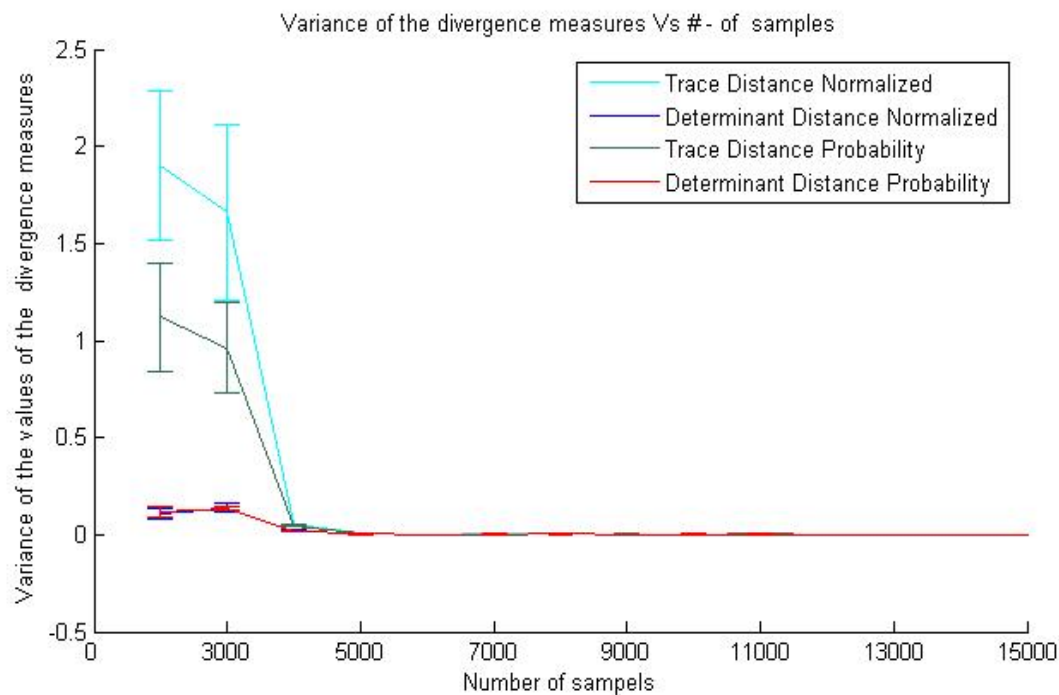


Figure 10.1: Variance

In Figure 10.1 we present the decrease in the variance of the different divergence measures between the base graph and the estimated graph. This figure shows that the sampled graph indeed converges with the underlying graph as the number of samples increases. Moreover, the convergence is rapid, if we consider that there are over 2000 edges, with different weights. This is due not only to our sampling technique, but also to the fact that the divergence measure uses only the small eigenvectors, which converge faster than the middle eigenvectors (ordered by their eigenvalues), as can be seen in Figure 5.1.

10.3 Anomalies

Before we can test our algorithm, one last subject must be considered: what we count as an anomaly, or more accurately - how we insert anomalies into our sampling. Theoretically, it is possible to just sample from a different graph; but this is so dramatic a change that it cannot be thought of as a realistic model for anomalies. Our suggestion is to add or remove edges while going against the graph's characteristic; hence every model will have a different technique for creating anomalies.

Anomaly technique should add or remove edges in a way that the new graph has a low probability of being created from the normal graph. The Erdős-Rényi model contradicts such a notion, as all edges in it have the same probability; hence, the anomaly divergence is only in the sense of edit distance, and is not suitable for our purpose.

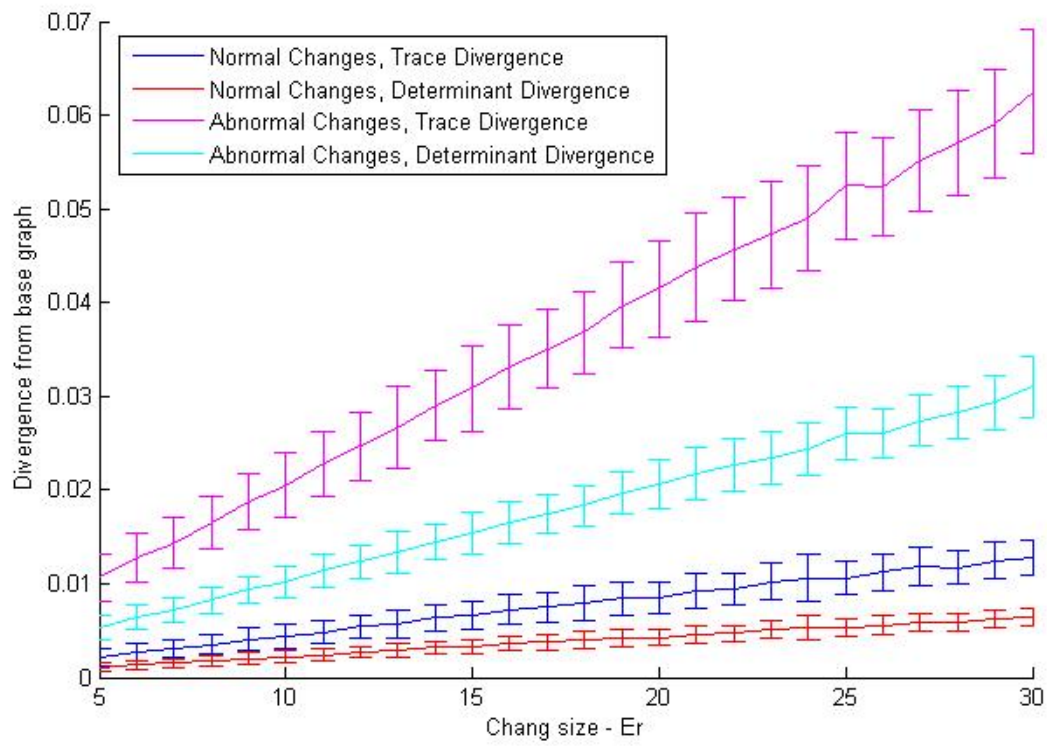
The Barábsi model is defined by the preferential attachment to high-degree vertices, so anomalous edges will be such that do not follow this distribution, e.g connect even vertices of low-degree. Note however, that as such edges are possible in the early stages of graph construction, it will be very difficult for any algorithm to distinguish between a normal and an abnormal edge .

The next two models have well-defined structures, and hence edges that ruin and disturb this structure will be counted as anomalous. In the small-world model, adding edges between vertices that are not close (in the circle) will be counted as inserting anomalies. Inserting anomalies in the high-clustered model will be by adding edges between the clusters, ruining the cluster structure.

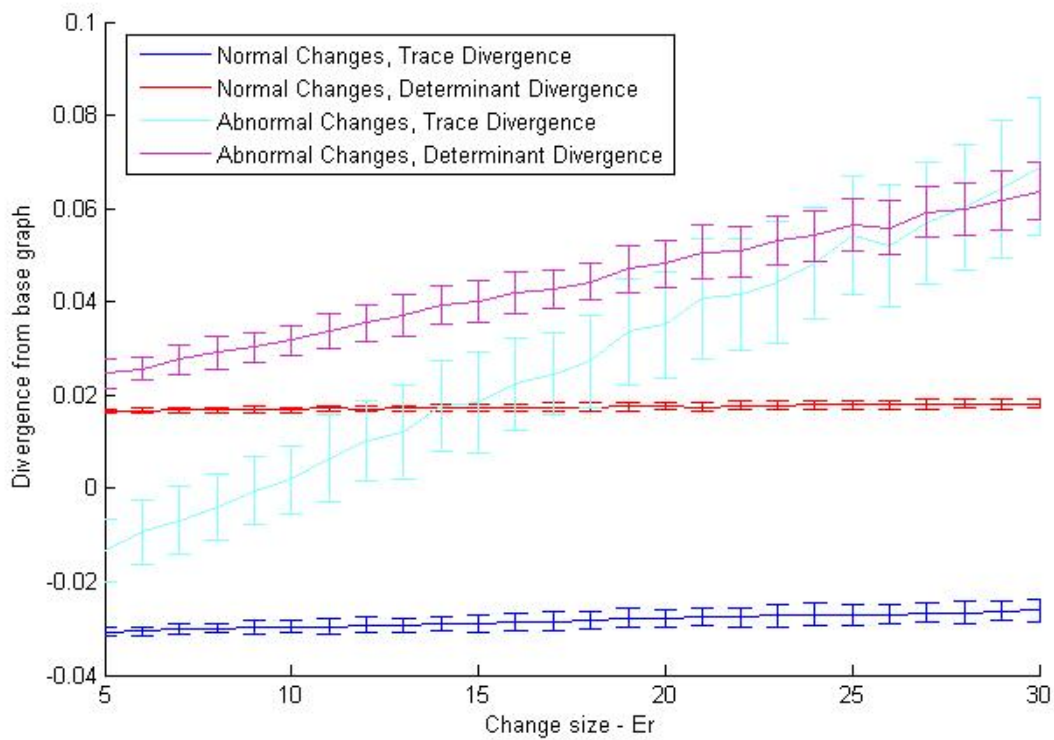
It may seem that these definitions are too strict, and that any random adding or removing of an edge will be anomalous. In this work we have stressed the point that not all divergence from the base graph is considered as anomalous. We argued that a change inside a cluster should not be regarded as anomalous, as opposed to a change between clusters, since the former change may be considered a part of the graph dynamics, whereas the latter is a rare incident in the normal dynamics of a graph, and hence can be treated as an anomaly. For example, in a social network, the probability of two individuals to be introduced is higher if they are part of the same group (having a higher probability of a common friend), relatively to if they are in different groups (with a very low probability of a common friend).

After having created an anomalous graph, we sample from it and not from the base graph (the normal one). The anomaly size is denoted as Er , which is the sum of weights of all edges inserted or removed anomalously.

We are now ready to see whether our divergence measure is indeed not sensitive to changes we call normal - adding and removing edges inside clusters, while sensitive to the anomalous changes - adding edges between clusters. Figure 10.2 shows that our divergence measure is indeed relatively unaffected by normal changes.



(a) Normalized Laplacian



(b) Probability Laplacian

Figure 10.2: Divergence from the underlying graph of normal vs abnormal changes

Chapter 11

Results

11.1 Our Divergence measures and algorithm

A good divergence measure for detecting anomalies will be one that places two normal instances close to each other, returning a small value, while when given an anomalous and a normal instance, will set the two far apart, returning a high value. The gap between the two values will be an important way to test the different divergence measures. Theoretically speaking, we can look at the expected divergence or the worst-case (smallest) divergence between the two groups of graphs. This approach entails a necessity to grasp the entire graphs' space, define a probability measure on it, or hold the ability to search for the minimum instance in it. Because of these intricate demands, this approach will be left for later work. In this work, we will use the cluster graph as an representative of the space. We will sample graphs from this model and check the expected behavior on it.

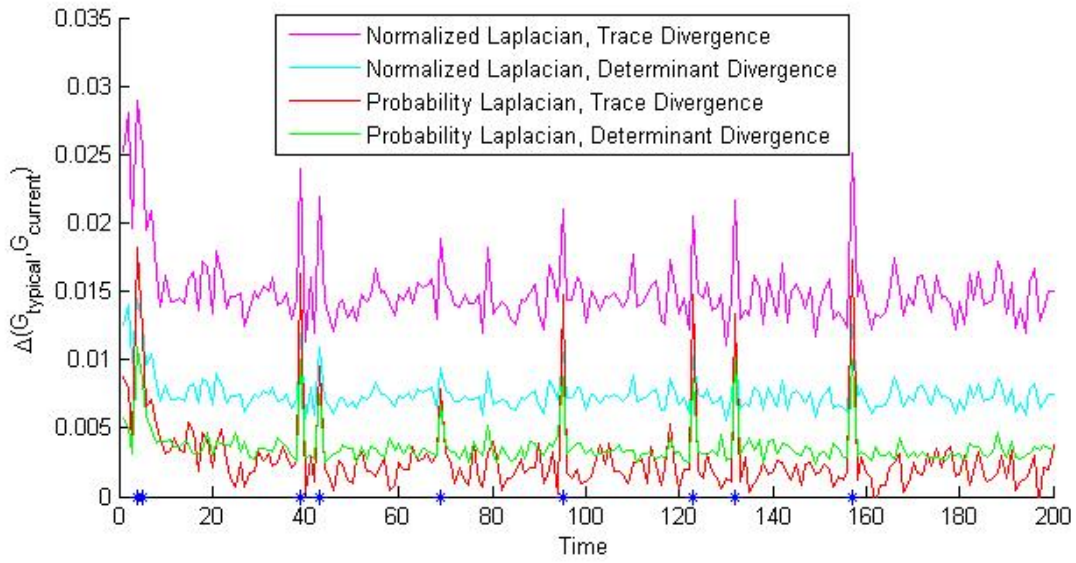
Using our graph simulation for testing the divergence measure brings us to the well-known concept of signal-to-noise ratio. The signal in our simulation will be the inserted anomaly in the abnormal time-points, while the noise may take several forms, such as : the fluctuation due to the sampling process, normal changes in the graph, the cause of the abnormality not being fully sampled, etc. The gap, or ratio, between the fluctuations due to the noise and the peaks when there is an anomalous instance, is our way to evaluate our results.

Since some noise arises from the fact that we do not have the graph but rather some estimation of it, we checked the divergence measures in two cases: the first, a large amount of noise and a large signal, meaning that we used a small sample size and a relatively large size of the inserted anomaly (Figure 11.2); and the second, having a large data sample and a relatively small signal (Figure 11.1).

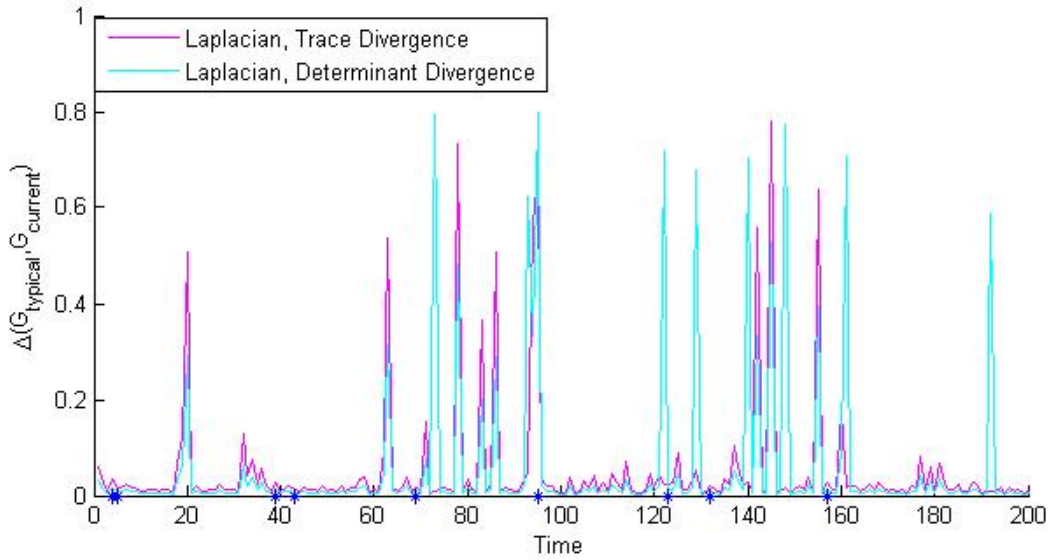
To stress the quality of our result, let us first examine Figure 11.1. This figure is the result of simulation of a high-cluster graph. In each time-unit, the base graph is sampled and the divergence between the current sample and the estimation of the base graph is calculated and presented. With a probability of 0.05 an anomaly is inserted into the the base graph (changing the underlying graph only for this specific time-unit), and only then is the base graph with the anomaly sampled. These time-points are marked by blue asterisks, and we expect the divergence measures to give a higher value there.

Sub-figure 11.1(a) clearly shows that the divergence measures using the normalized Laplacians identify the abnormal time-points with a considerable gap, while in sub-figure 11.1(b) it can be seen that in the unnormalized Laplacian, the signal cannot be separated from the noise. We have defined the signal as the insertion of edges of which each end-point is in a different cluster. One must remember that such edges exist in the normal graph as well. In this trial, the size of the inserted signal is $Er = 2.5$, which is only about 10% of the total weight of the edges between the clusters in the normal graph. Our algorithm can therefore detect an addition to the uncommon edges as small as 10%.

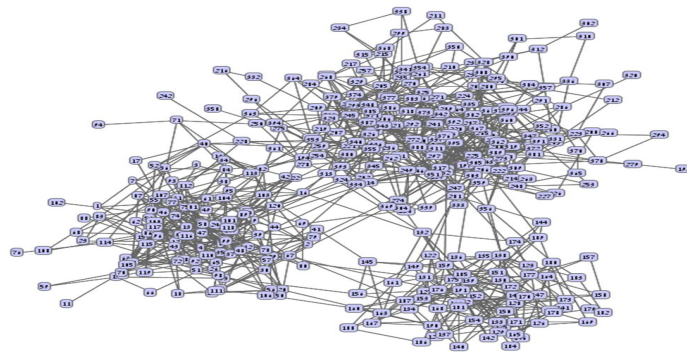
Figure 11.2 demonstrates the ability of our algorithm from the opposite point of view. The figure presents the same simulation as Figure 11.1, with a smaller sample size and a bigger anomaly size. The



(a) Normalized Laplacians Divergence measures

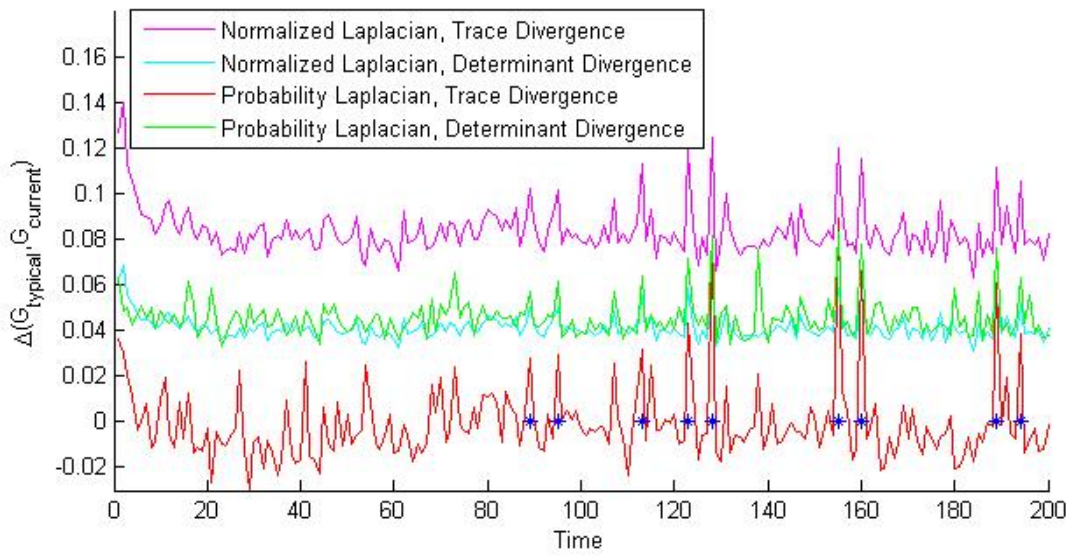


(b) Laplacian Divergence measures

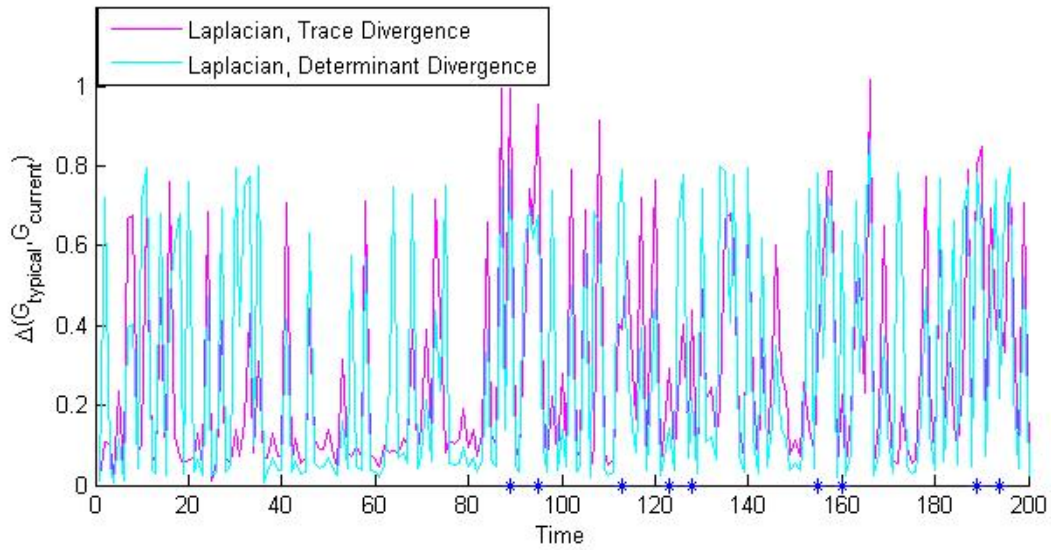


(c) The Graph

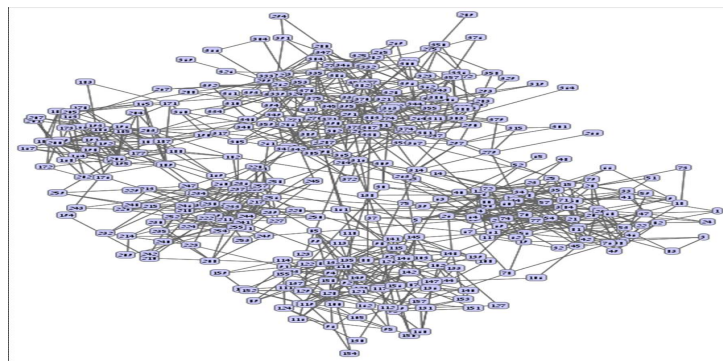
Figure 11.1: Time Series of High-Clustered Graph. $Er = 2.5, T = 15000$



(a) Normalized Laplacians Divergence measures



(b) Laplacian Divergence measures



(c) The Graph

Figure 11.2: Time Series of High-Clustered Graph. $Er = 7.5$, $T = 5000$

size of the anomaly is $Er = 7.5$, which is about 22% of the total weight of edges between the clusters in the normal graph. We sample only 5000 edges; remembering that we have at least 2000 weighted edges, this means that our sample is very meagre. Even under this condition, it is clear from sub-figure 11.2(a) that when using the normalized Laplacians, our divergence measure clearly identifies the abnormal times with a meaningful gap.

Until now we have presented the result of the two divergence measures and the three different Laplacian definitions. Figure 11.1, and more clearly Figure 11.2, prove the fact that the normalized Laplacians perform better than the unnormalized Laplacian. So from now on, the unnormalized Laplacian will be omitted. We wish to point out that, though we showed the existence of a gap between the fluctuations due to the noise and the ones due to our signal - the peaks at abnormal times, we did not give any estimation to this quantity. An explanation to this is the simple observation that, a divergence measure that fails to detect 25% of the signals but has very high peaks compared to the noise fluctuation, will have a higher ratio or gap than a divergence measure that detects 100% with small peaks in all these time-points. Therefore what is of interest is the existence of the gap, and not its exact magnitude.

The results presented up to this point were the ones of the high-cluster graph, with changes only in the anomalous time-points - no normal changes. We will now show some results of the other models of graphs.

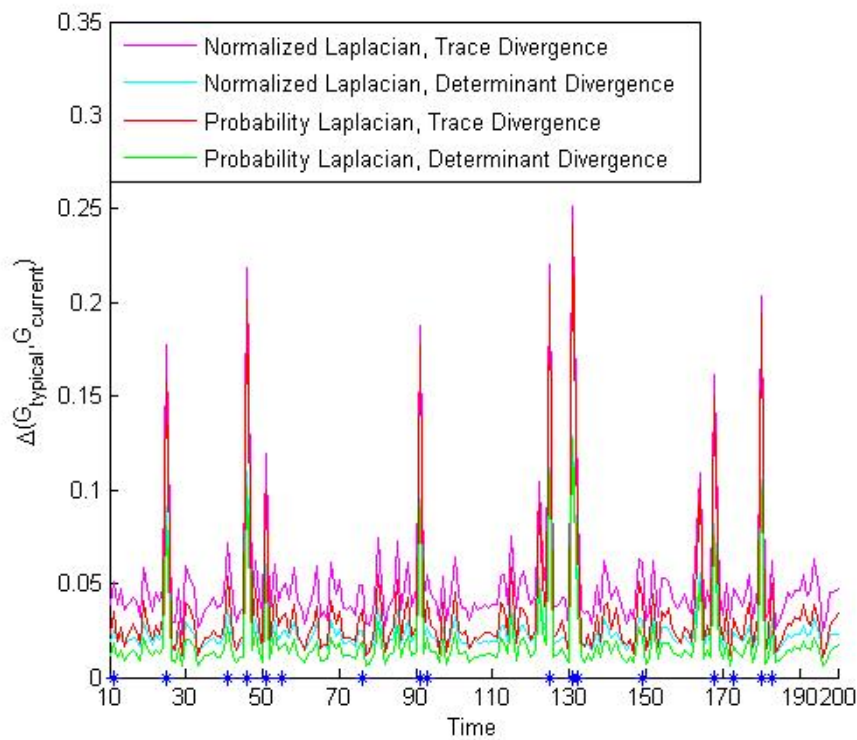
In Figure 11.3 we can see the time-series on the Small-World graph. It is clear that our method indeed may be applied successfully on this model, as can be seen from the large gap between the signal and the noise. The explanation to the success, as previously mentioned, is the dominant structure that is captured by the small eigenvectors.

As opposed to the results so far, Figure 11.1 shows the failure of the algorithm when applied on the graph generated by the Barábsi model. Since there is no dominant structure to this model, the small eigenvectors do not contain a sufficient amount of information to enable detection of anomalies in the graph. (See sub-figure 5.1 - the small eigenvectors of a random graph, or of any non-structure graph, do not converge.) Hence, no gap between the signal and the noise can be detected.

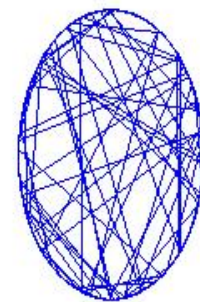
Up until this point the results focused on the divergence measures, as they are the focus of this work, and we attempted to minimize the effect of the surrounding algorithm on them. We now need to assess the algorithm itself. While when trying to evaluate our algorithm, two parameters are of interest: One, the number of anomalies which the algorithm detects, compared to the total number of anomalies. Two, the number of times in which the algorithm predicts an anomaly without there being one, compared to the total number of normal samples. The first parameter is called sensitivity, and the second - false-positive rate (or better known as $1 - \text{specificity}$). The graph created when comparing these two criteria is called the ROC curve.

Sub-figures 11.5(b) and 11.5(a) describe the results for the various divergence measures. It is clear that the graphs do not resemble a normal ROC curve. This is due to times in which the algorithm failed (declaring all times as anomalous). Since the focus of this work is not the detection algorithm, we believe that the graph we present is sufficient to manifest the utility of our divergence measures.

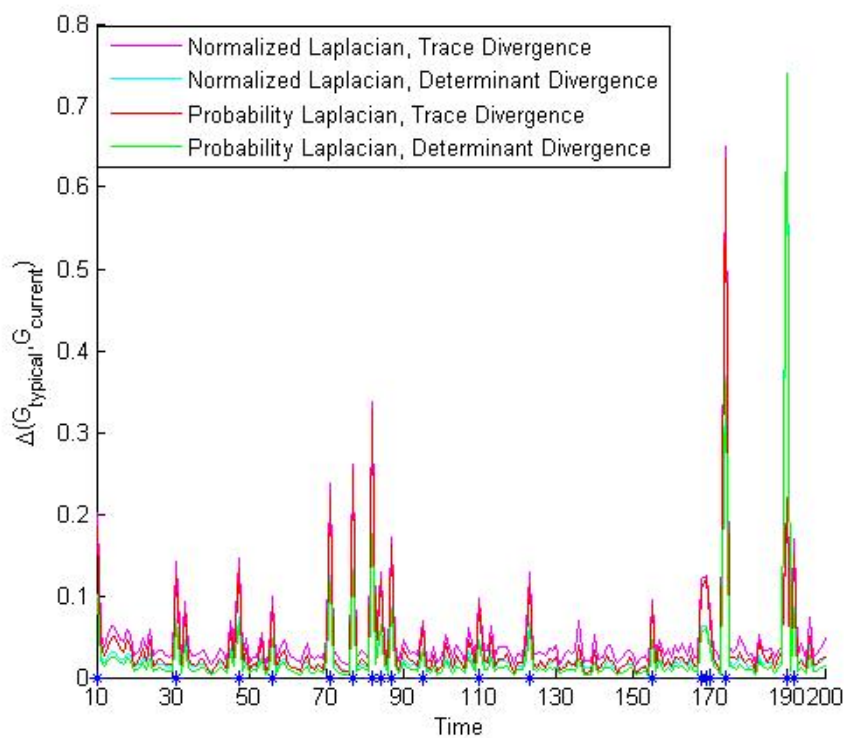
One of the more important characteristics of our divergence measures is their ability to disregard normal changes in the graph, while being very sensitive to anomalous ones. Figure 11.6 presents this ability. The red asterisk marks the times in which the underlying graph is changed by a normal change (as is in the case of the abnormal change, only the present time-unit is effected.) It is clear from this trial that our divergence measures can indeed effectively disregard the normal noise; note, however, that the normal noise does make this detection more difficult even for our divergence measure, but makes it impossible for any method based on the edit distance. When monitoring a network, one may monitor each vertex individually. This method will be insensitive to an anomaly which occurs on a very low scale in many vertices simultaneously. In our terms, this will mean that the size of the error will be unchanged, but the weight of the anomalous edges will be bound from above. Figure 11.7 is a trial in which the edges' weight is bound to between $Wh = 0.1$ and $Wl = 0$. It is clear that this indeed



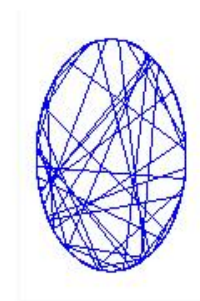
(a) Time Series $E = 5, T = 15000$



(b) Small-World Graph



(c) Time Series $E = 15, T = 15000$



(d) Small-World Graph

Figure 11.3: Time Series of Small-World Graph

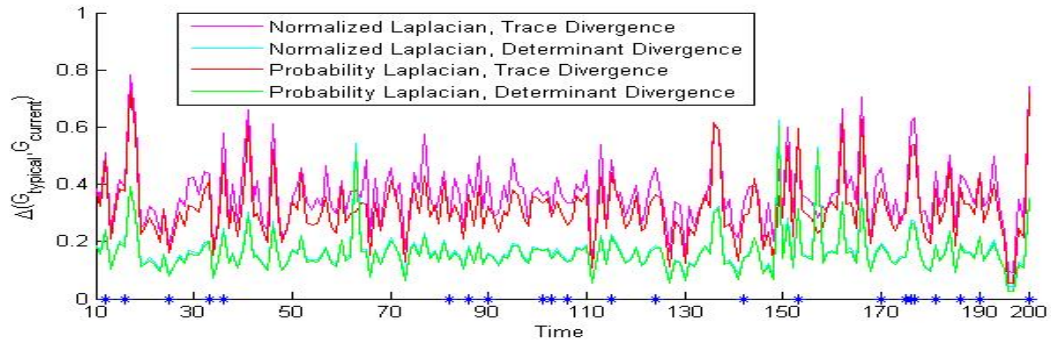
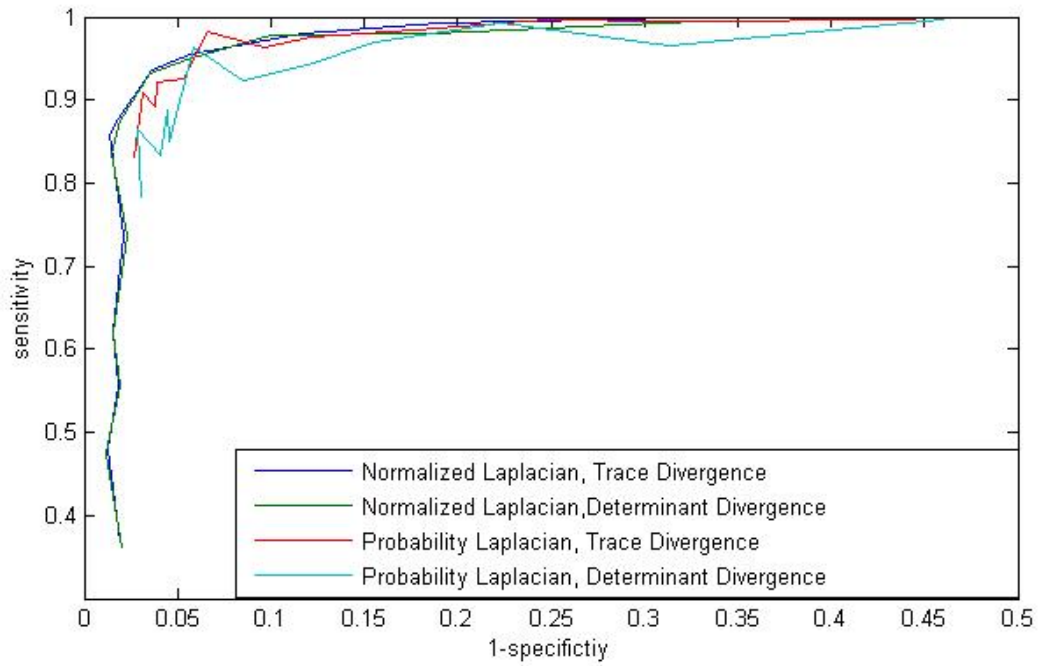
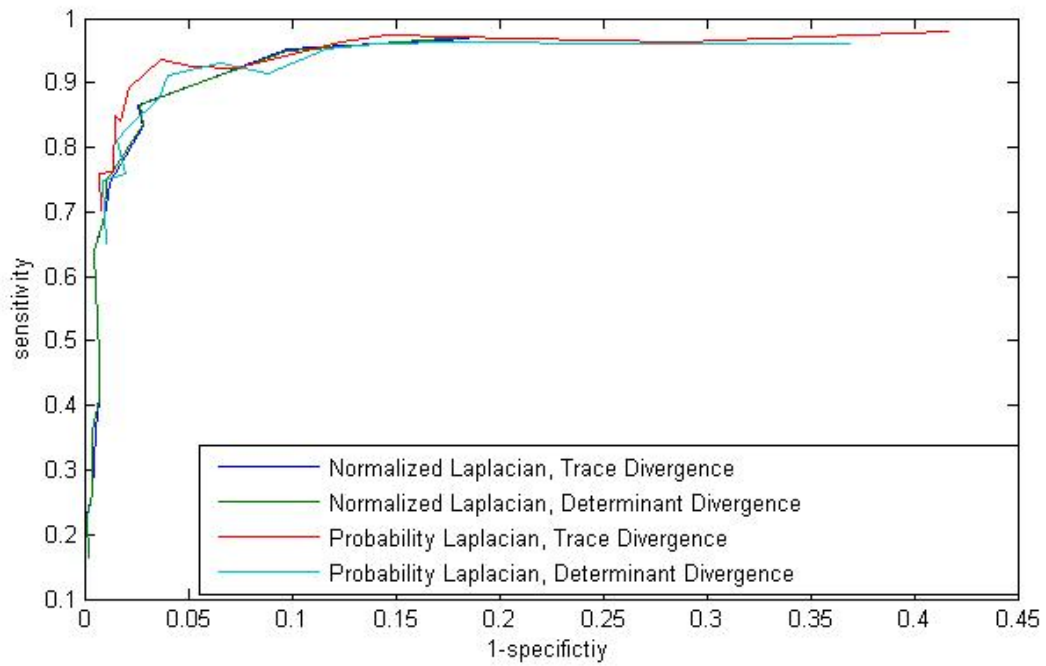


Figure 11.4: Time Series of Scale-Free Graph - Barabási model

makes little difference to the method's ability to detect anomalies, while these cannot be detected by any vertex-monitoring methods.

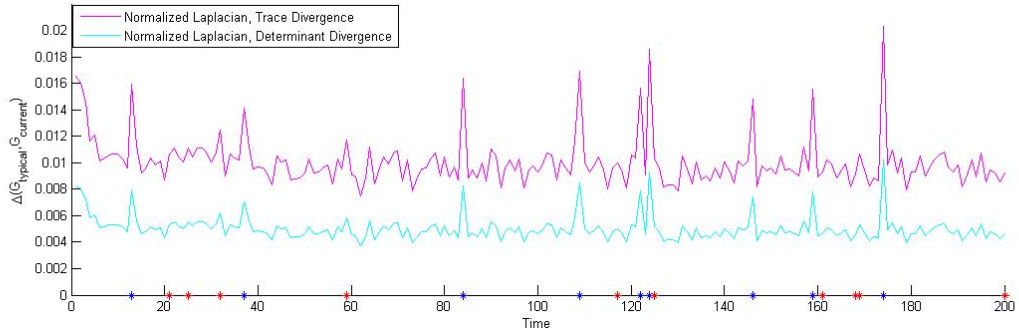


(a) $E = 15, T = 5,000$

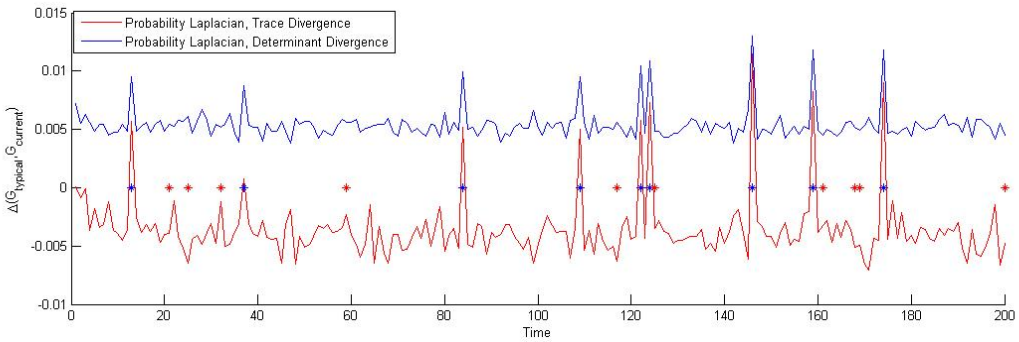


(b) $E = 5, T = 15,000$

Figure 11.5: ROC curve

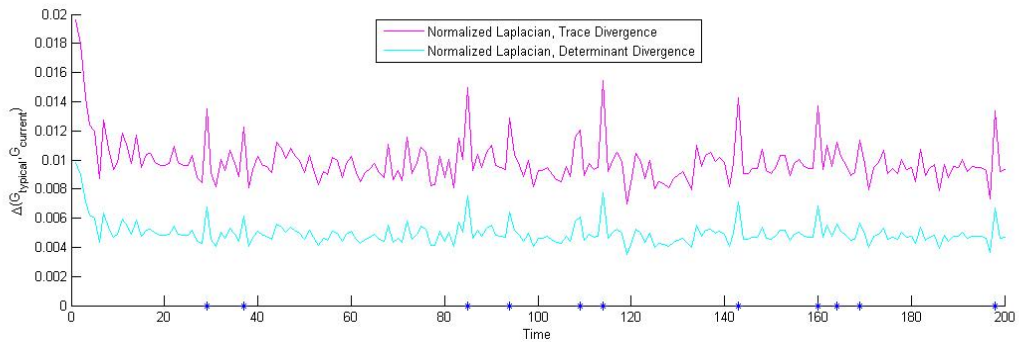


(a) Normalize Laplacian

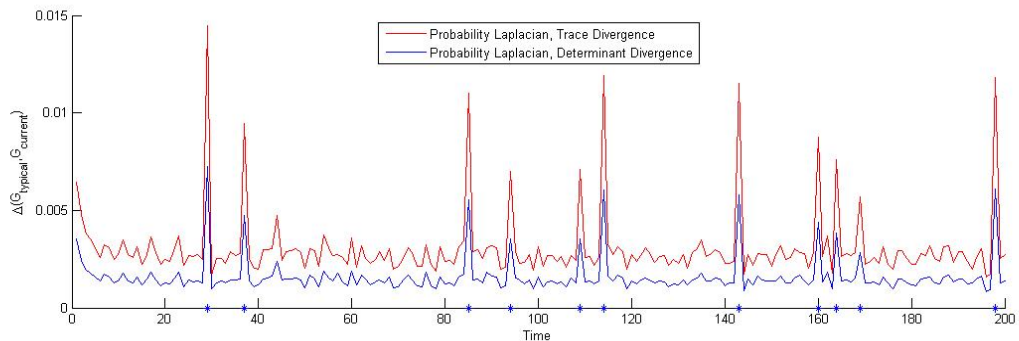


(b) Probability Laplacian

Figure 11.6: Time Series with normal noise. $T = 15000$, $Er = 4$, normal noise equals 4 as well



(a) Normalized Laplacian



(b) Probability Laplacian

Figure 11.7: Time Series with spread noise. $T = 15000$, $Er = 4$

Chapter 12

Conclusions & Future Work

The focus of this work was the two divergence measures we have presented, the Determinant measure Definition 9 and the Trace measure Definition 10. We explained the intuitions that guided us towards these functions. The results presented above prove, that these divergence measures indeed are able to detect anomalies, even in cases that are hidden from the straight-forward techniques.

We have presented an algorithm for detection of anomalies in networks. The algorithm's results, as can be seen in Figure 11.5, are usually good, but suffer from instability. We did not discover all the cases in which our algorithm failed, nor find all the reasons for this failure. However, a simple example of such a case is the unconnected graph - a situation possible through the generation technique of our graph, hence contaminating our results.

As noted above, the algorithm was not itself the focus of this work, and as such we consider it a suggested basis for future algorithms, and not an accomplished, ready-to-use one. In this chapter we will discuss several improvements, or more correctly adjustments, needed to be made before it could be used for real-life networks.

One of the most crucial stages in the algorithm is setting the threshold above which the sample is considered as an anomaly. In the current algorithm, we requested the parameter a from the user. This parameter is used as multiplier of the standard deviation; by that we are actually assuming that the divergence measure is has a Gaussian distribution, which is not necessarily true. We believe that more sophisticated techniques exist, such that take advantage of the rarity of the anomalies in a more direct way.

A very basic assumption on which our algorithm is founded, is the static nature of the underlying graph, and in all our experiments the underlying graph remains unchanged. Real-life networks change over time. If we were to examine the biologic cell, the chemical network in it changes during the different stages of the cell-cycle. Reactions that had taken place with a high rate in one stage (strong connection between the two chemicals - high-weighted edge) may be rare events in the other (low-weighted edge between the vertices symbolizing these chemicals).

In view of the last observation, when it is necessary to monitor a network, one of the basic questions is the time-scale in which the network should be monitored. In one time-scale we may want a change to be considered as an anomaly, while in a higher time-scale we might want the system to adjust to it. This will require some mechanism to allow the learned parameter to change over time. Such a mechanism can simply assign a higher weight to the recent events, and by this allow the system to "forget" the past.

This kind of dynamics of the network will not fit for normal changes in the network which are drastic - such as those observed in networks with several distinct states. If we monitor the social network of a community in an hourly rate, we will see some drastic changes during the day. The people one meets during working hours are different from the people one meets after work - the change is drastic and cannot be described by a drift. Hence the above mechanism will not fit here. This situation is best described by a Markov Hidden state. Each state will retain its underlying graph (and other parameters),

and we will define probabilities for switching from one state to another. Using this description, we will have a distinct (social) networks for work hours, and a different network for after-work hours, etc. A state will be considered as an anomaly if it is distant from this network, has low probability of switching to another state, and is distant from all probable states as well.

When looking at real-world networks, additional demands arise, such that involve changes to the divergence measures. One such demand is directionality. A detector network is a good example for a network requiring this property. This is a network of detectors (each detector is a vertex), in which the correlation of activity will be the edge weight. More explicitly, the weight of edge $e_{i,j}$ will be the correlation between the activity of the i and j detectors, possibly with a certain time-shift. In our framework, the edge weight function, as defined by Definition 1, must be both symmetric and positive. These two restraints prevent us from expressing any directionality, and therefore the detector network cannot be monitored by our divergence measure. Hence, a possible improvement of our divergence measure will be to remove the positivity of the weight function. Note that the demand of the adjacency matrix to be positive-semidefinite is the source of this constraint. Allowing the weights to be negative and directly guaranteeing the positive-semidefiniteness of the matrix is one simple option.

When the commerce network was presented, we considered the edge weight to be the total trade between the entities. A more fitting weight function may be one in which the sales of entity i to j will be the weight of the $e_{i,j}$, and the sales from entity j to i will be the weight of the $e_{j,i}$. This results in a directed graph with an asymmetric adjacency matrix. Definition 1 excludes such a network from being monitored by our divergence measures. Including directed graphs appears to be a difficult mission, since we may need to deal with adjacency matrices that are not positive-semidefinite.

To this date, most monitoring algorithms handle each instance individually, regardless of the connection to other monitored instances. We believe that this method is limited, and that the entire network should be monitored as a whole. For this purpose, the problem of network anomaly detecting has been defined here in two different ways. We have suggested two divergence measures and have explained the intuitions that brought us to them. We have presented an algorithm that detects anomalies, and have suggested several points of improvement that may be necessary in order apply it to real-world data. The results we have presented verify that our divergence measure indeed captures the anomalies in the graph, while having the ability to disregard normal changes in it. These are the most basic required features in any detection algorithm.

Further study is needed for our method to be fully evaluated. Such work includes theoretical analysis of the algorithm's performance; testing our algorithm on real-world networks; and finding criteria to determine the spectrum of graphs to which our algorithm may be applied, which in a more interesting perspective, we believe to be the same as asking in which graphs the first eigenvectors hold meaningful information.

Bibliography

- [1] H. Jeong *et al.*, Nature (London) **407**, (2000).
- [2] D. A. Fell and A. Wagner, Nature Biotechnology **18**, 1121 (2000).
- [3] D. A. Smith and D. R. White, Social Forces **70**, 857 (1992).
- [4] J. J. Hopfield, PNAS **79**, 2554 (1982).
- [5] H. Ebel, L.-I. Mielsch, and S. Bornholdt, Phys. Rev. E **66**, 035103 (2002).
- [6] M. Girvan and M. E. J. Newman, Proceedings of the National Academy of Science **99**, 7821 (2002).
- [7] A. Vázquez, R. P. Satorras, and A. Vespignani, Physical Review E (Statistical, Nonlinear, and Soft Matter Physics) **65**, (2002).
- [8] X. F. Wang and G. Chen, Circuits and Systems Magazine, IEEE **3**, 6 (2003).
- [9] R. Z. Albert, Ph.D. thesis, UNIVERSITY OF NOTRE DAME, 2001.
- [10] E. Ravasz and A. L. Barabasi, Physical Review E (Statistical, Nonlinear, and Soft Matter Physics) **67**, (2003).
- [11] M. Girvan and M. E. J. Newman, Community structure in social and biological networks, 2001.
- [12] H. Zhou, Physical Review E **67**, 061901+ (2003).
- [13] A. Sanfeliu and K.-S. Fu, IEEE transactions on systems, man, and cybernetics **13**, 353 (1983).
- [14] B. Luo and E. R. Hancock, Pattern Analysis and Machine Intelligence, IEEE Transactions on **23**, 1120 (2001).
- [15] L. S. Shapiro and J. M. Brady, Image Vision Comput. **10**, 283 (1992).
- [16] U. von Luxburg, Stat. Comput. **17**, 395 (2007).
- [17] M. Hein, J.-Y. Audibert, and U. von Luxburg, Graph Laplacians and their convergence on random neighborhood graphs, 2006.
- [18] R. I. Kondor and J. D. Lafferty, in *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning* (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002), pp. 315–322.
- [19] F. R. K. Chung, *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92) (Cbms Regional Conference Series in Mathematics)* (American Mathematical Society, AD-DRESS, 1997).

- [20] J. Shi and J. Malik, Pattern Analysis and Machine Intelligence, IEEE Transactions on **22**, 888 (2000).
- [21] A. Y. Ng, M. I. Jordan, and Y. Weiss, in *Advances in Neural Information Processing Systems 14* (MIT Press, ADDRESS, 2001), pp. 849–856.
- [22] R. C. Wilson and P. Zhu, Pattern Recognition **41**, 2833 (2008).
- [23] R. Bhatia, *Matrix analysis*, Vol. 169 of *Graduate Texts in Mathematics* (Springer-Verlag, New York, 1997), pp. xii+347.
- [24] R. M. Gray, *Toeplitz And Circulant Matrices: A Review (Foundations and Trends(R) in Communications and Information Theory)* (Now Publishers Inc., Hanover, MA, USA, 2006).
- [25] C. Shannon, Proceedings of the IRE **37**, 10 (1949).
- [26] R. R. Coifman *et al.*, Proceedings of the National Academy of Sciences **102**, 7426 (2005).
- [27] B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis, in *Advances in Neural Information Processing Systems 18*, edited by Y. Weiss, B. Schölkopf, and J. Platt (MIT Press, Cambridge, MA, 2006), pp. 955–962.
- [28] R. R. Coifman and S. Lafon, Applied and Computational Harmonic Analysis **21**, 5 (2006).
- [29] M. Belkin and P. Niyogi, Neural Computation **15**, 1373 (2003).
- [30] N. Tishby and N. Slonim, Advances in neural information processing systems 640 (2001).
- [31] B. Nadler, S. Lafon, R. Coifman, and I. Kevrekidis, *Principal Manifolds for Data Visualization and Dimension Reduction* (Springer Berlin Heidelberg, ADDRESS, 2007), Vol. 58, pp. 238–260.
- [32] C. C. Paige and M. A. Saunders, SIAM Journal on Numerical Analysis **18**, 398 (1981).
- [33] G. W. Stewart and J. G. Sun, *Matrix perturbation theory, Computer Science and Scientific Computing* (Academic Press Inc., Boston, MA, 1990), pp. xvi+365.
- [34] I. B. Risteski and K. G. T. cevski, Beiträge zur Algebra und Geometrie **42**, 289 (2001).
- [35] C. Davis and W. M. Kahan, SIAM J. Numer. Anal. **7**, 1 (1970).
- [36] L. Huang *et al.*, Technical support, U.C. Berkeley, Berkeley, CA (unpublished).
- [37] N. Biggs, *Algebraic Graph Theory (Cambridge Mathematical Library)* (Cambridge University Press, ADDRESS, 1994).
- [38] I. Gutman and W. Xiao, Bulletin: Classe des sciences mathe’matiques et naturelles - Sciences mathe’matiques **129**, 15 (2004).