CITY UNIVERSITY
LONDON

MSc Data Science Project Report 2016

Analysing the evolution of communication patterns in email data through an extended dynamic network analysis toolkit

Arshad Ahmed

Supervised by: Dr Cagatay Turkay
Submission Date: 23 September 2016

Declaration

By submitting this work, I declare that this work is entirely my own except those parts duly identified and referenced in my submission. It complies with any specified word limits and the requirements and regulations detailed in the assessment instructions and any other relevant programme and module documentation. In submitting this work, I acknowledge that I have read and understood the regulations and code regarding academic misconduct, including that relating to plagiarism, as specified in the Programme Handbook. I also acknowledge that this work will be subject to a variety of checks for academic misconduct.

Signed: Arshad Ahmed

Abstract

Dynamic networks arise in many modern applications such as email, social and telecommunication networks. These networks evolve over time through the addition and deletion of nodes and edges. Thus similarity measures for these networks are critical for us being able to characterise and understand this change. This study explores the use of similarity measures to support dynamic network analysis. To this end we survey existing and proposed methods, apply selected methods to the Enron email dataset and validate against these benchmark measures novel measures introduced in this study. These measures are inspired by the fields of Seismic Data Analysis and Music Information Retrieval. Also we propose novel visualisation techniques to support the analysis of dynamic networks such as the Radon, Frequency-Wavenumber, Audio Waveform and Log Panel Plots. These novel measures are derived using a variety of integral transforms such as the Fourier, Hilbert and Abel transforms. Analysing these attributes derived from the Modularity, Adjacency and Normalised Laplacian graph matrices we find that they are best derived from the Normalised Graph Laplacian. To integrate all the existing and novel measures we also propose two aggregation schemes that allow for the derivation of meta attributes such as the RMS and NRMS of the attribute volume derived from the graph time series. These measures act as effective snapshots of network activity at different time steps.

Keywords: Dynamic Network Analysis, Attribute Analysis, Multi-Attribute Visualisation

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

Networks arise in many modern day applications such as email, social, transport and telecommunication applications. These networks represent convenient structures to analyse phenomenon that would be very difficult to analyse otherwise. However, with most modern networks a key feature is that these networks evolve over time. These dynamic networks require analytical methods that would allow easy comparison between them at the different time steps. Thus similarity measures that allow us to assess and track change over time in such networks are instrumental for us to be able to study such networks. Therefore similarity analysis on dynamic networks is of critical importance if we are to be able to understand this evolution over time. This study is motivated by the need to explore the effectiveness of traditional Network Analysis metrics such as centrality analysis in the context of dynamic networks in addition to exploring novel measures.

The traditional metrics were developed mainly for the analysis of static graphs thus for modern applications their suitability needs to be assessed and their shortcomings determined. In this study we find that although these measures can be utilised in a dynamic network context they are useful for benchmarking of other new measures. The field of Dynamic Network Analysis spawned with the explicit aim of expanding our toolkit to handle such complex evolving networks.

In this study we propose novel measures of similarity which are motivated by the fields of Seismic Data Analysis and Music information Retrieval. These novel measures proposed here are calculated from the Normalised Graph Laplacian. Some of these new attributes behave very similarly to traditional centrality measures while being sensitive to smaller changes in the network that traditional measures do not pick up on as well.

In addition to capturing network dynamics we show that by utilising such attributes some aggregated network measures such as Average Degree could be predicted with a good level of accuracy using a Gradient Boosting Regression Technique. This is useful because having metrics that allow us to potentially model the change of the network under consideration can help us understand the key drivers even better. For example this method enables feature ranking of attributes in this data the at-

tribute derived from the Abel Transform is found to be particularly useful. This is interesting because this attribute appears to capture hidden dynamics which become apparent through mapping to these alternative spaces but the traditional metrics do not feature highly on this list. So there is a case for these attributes to be used in a predictive context.

This study also introduces a number of visualisation techniques to support the joint analysis of these attributes. These are namely the Radon, Frequency-Wavenumber (FK), Log Panel and Audio Waveform Plot. The Radon and FK plot serve as a form of dimensionality reduction and allow us to visualise the whole derived attribute volume in these alternative spaces in 2 dimensions. The Log Panel allows the analysis of multiple attributes by placing them along panels side by side. These can be arranged in a number of ways such as the attributes can be grouped by type such as Centrality measures, Seismic and Music attributes or they can be sorted by cluster indices from a hierarchical clustering procedure. This name is inspired by Well Log Panels which are used in hydrocarbon exploration to help map lithologies using a variety of logs captured while drilling.

The next two sections are included to provide the reader with a high level background of the topic. A detailed literature review is presented in Chapter 2.

1.2 Paradigms of Network Analysis

The literature suggests 3 main paradigms of network analysis these inform the framing of research questions and investigative approach. These can be broadly described as: [1]

- Social Network Analysis (SNA)
- Dynamic Network Analysis (DNA
- Supernetwork Analysis (SuNA)

1.2.1 Social Network Analysis (SNA)

SNA is concerned with the study of relationships between entities and its focus of research is of two types: whole network analysis and self-centred network analysis. Whole network analysis is concerned with understanding the structure of relationships between different roles in a group and is used to investigate network structure changes with the time and the contact pattern of network entities. Self-cantered network analysis is concerned with how the individual behaviour of network entities are influenced by the membership of the network. [1]

1.2.2 Dynamic Network Analysis (DNA)

DNA was proposed as an extension to the SNA. The strength of DNA is that it is able to handle large scale dynamic, multi-modal, multi-lateral network with various levels of uncertainty. The edges are probabilistic and the nodes behave like agents in a multi agent environment so this enables perturbations or changes in the network

to propagate through the network and result in some global reconfiguration. The evolution of a network in The application of machine learning and multi agent modelling in the same environment is enabled by DNA's use of the meta matrix.[1]

1.2.3 Super Networks Analysis (SuNA)

Super networks can be thought of as networks of networks that exist above and beyond existing networks. These have the characteristic of being multi-layered, multi-dimensional, multi-attributed and multi-levelled with additional features such as congestion and coordination. These have been applied predominantly in supply chain management, finance, traffic and ecology among others. These networks are analysed using either variational inequality and/or hyper graph theory. [1]

1.3 Recent Work

Li and Liao [2] notes the complexity and difficulty of Dynamic Network Analysis as a field of study. They note that modern applications such as the Internet of Things and mobile social networks we are faced with networks that are not only large but having complex dynamics associated with them. This is driven by the fact that node numbers and connections grow exponentially with connections constantly being added and broken. The properties of these networks evolve with time. This is the defining feature of a Dynamic Network. They are characterised by their time dependent topology due to fluctuations in the underlying network activity. As traditional data mining techniques are deemed to computationally expensive or insufficient when faced with such complex networks they suggest a visualisation based method in which the links in the dynamic graph are broken down by the time dimension. Each segment as a result represents a time step of the evolution of some property of the dynamic network. The key contribution is that their approach is a static view based approach which does not require the end user to have a mental map of the previous steps to enable analysis as in animation based methods. As a result it is easier for the end user to detect patterns and identify potential anomalies.

Pereira, Amo, and Gama [3] introduce the notion of evolving centralities in temporal networks in the context of social networks. They analyse data from Twitter but in order to understand the reaction of the structural position of the user with the underlying network evolution they utilise follower/followee networks are analyse the centrality evolution over time. Their approach is different in the sense that their approach is based on temporal graph theory. This enables more sophisticated analysis as the shortest path can be interpreted as a function of time and therefore they are able to recalculate the closeness and betweenness centrality using these fastest paths. This leads to their insight that Twitter users are dynamic and can assume or leave central positions in a network.

Wu, Liu, and Niu [4] propose a graph based decay function to update the frequency of user interactions in a social network and then use a community detection algorithm to detect communities at each time step. They find that most studies ignore the temporal information in the study of community structure in networks and show that by incorporating such information is very helpful in the analysis of community

evolution analysis in social networks.

Hu and Cao [5] use probabilistic graphic models to detect time-evolving influence among objects from dynamic heterogeneous graphs. They note that the dynamic nature of such heterogeneous graphs where nodes and edges are added or removed dynamically are not sufficiently addressed by current studies. Therefore to handle the dynamics of the network and to learn the time evolving influence structure they propose to use probabilistic graphic models using the graphs at discrete time stamps.

Mahyari and Aviyente [6] discuss signal processing on graphs and introduce the Fourier Transform for dynamic networks. They do this by finding a common subspace across a modified common Laplacian matrix for the dynamic networks. The eigenvectors of this Laplacian form the Fourier basis for the networks. From this common subspace the eigenvalues correspond to frequency components. Therefore high eigenvalues correspond to high frequency and vice versa. The Fourier transform is used extensively in this study for the derivation of the multiple attributes and supports many visualisation techniques as will be discussed in Chapter 3. The key difference between this study and our work is that they first find a subspace and then find the Fourier Transform of it and we apply the integral transform directly to the matrix and derive attributes from Fourier space.

Lansing [7] propose to assess the temporal evolution of networks by first transforming networks into signals through Classical Multidimensional Scaling based on the resistance distance and then constructing a tensor based on the spectra of each signal across time. We use the average resistance distance as an attribute to map change in the graph time series. Also we use Non-metric Multi Dimensional Scaling to visualise the relationship between the attributes.

1.4 Motivations and Aims

From the brief overview presented above it is clear that a lot effort and attention has been given to the topic of dynamic network analysis. These papers touch on various methods that we use in our study. But the wide variety of our methods and the nature of our application to dynamic networks to the best of our knowledge is unique and not encountered in the literature.

By treating the dynamic network as a time series we characterise each point in the series of the graphs by a number of attributes. These serve as a compact feature based representation of the network. The attributes are then collapsed into a single value either by taking the average or the Frobenius Norm where appropriate.

This is done for traditional centrality measures as well as the novel metrics. We show that by directly applying the integral transform to the Normalised Graph Laplacian there is no requirement to find a common subspace first. The application gives us a common subspace in the alternative space such as Fourier, Hilbert and Abel etc. We can then efficiently calculate multiple attributes in these spaces which can then be compared over the time range in question.

Also the treatment of networks as a music signal is novel. This approach opens the doors for many methods from digital music to be applied to networks. Therefore we can confidently state that we are addressing a real need for metrics designed to capture dynamism in networks. As a result we open the doors to many more methods to be imported not just from the fields from which we take inspiration but from other fields which use similar techniques.

The Fourier Transform forms the basis of many techniques explored in this study. However, in contrast to this work where the authors try to characterise the change in the spectral content of a common Laplacian. We in this study show that by using the Normalised Graph Laplacian we are able to derive a frequency representation for the graph time series. From the frequency of the individual graphs we can derive attributes such as the Norm of the Abel transform of the magnitude of the Fourier Spectrum for the dynamic network.

As mentioned that the analysis of dynamic networks is necessitated by the need to understand and model complex phenomenon. Undoubtedly there is great value in understanding the dynamics of such networks. The beneficiaries of this work are practitioners who are interested in or need to study dynamic networks. These could range from intelligence agencies monitoring changes in terrorist networks, social media companies monitoring and understanding the dynamics of their networks to telecommunications companies trying to identify interesting network dynamics that can drive their business forward.

The main aim of this study is to explore these metrics using the Enron Email Network Data. This will serve as a proof of concept for these attributes and visualisation techniques as well as outline a generalised work flow for the systematic robust analysis of dynamic networks.

A evaluation strategy is highlighted in this strategy that allows for a logical exploration and validation of these attributes and can serve as a road map for future work in this area.

Underlying all these analysis is the fact that these measures serve as a snapshot of a network at a point in this time. Thus sampling these measures at the different time intervals it is possible to derive a time series of attributes from the graph time series. This time series approach allows utilisation of methods from signal processing because by treating the graphs at the different time steps as a time series and the derived attributes also as a time series we can apply methods from signal processing

From the field of Seismic Data Analysis we present a number of attributes which are based on the notion of the Complex Trace. This involves using the Hilbert Transform and using the Real and Imaginary components to derive attributes such as Amplitude, Phase and Frequency. From this a number of other attributes are derived. wen addition matrix decomposition methods such as Kernel Principal Components Analysis (KPCA) and Non-negative Matrix Factorisation (NMF) are used to derive additional measures.

From the field of Music Information Retrieval we implement two measures the Zero Crossing Rate and the Spectral Centroid. This is possible by treating the networks as a audio signal by the use of the Fourier Transform on the Normalised Graph Laplacian.

Both sets of similarity measures that we will call Seismic and Music attributes for convenience are well established in their respective but their application to dynamic networks is novel. The motivation to utilise these measures is driven by the need to have measures that are scalable, have a relatable interpretation and are particularly suited for dynamic analysis. For example the seismic attributes are commonly analysed for the identification of hydrocarbon reservoirs and in 4D seismic two surveys are compared at different time steps to characterise change as a result of production from a field. So these measures become particularly suited for analysis with a time component.

Therefore the key motivations for this project can be summarised as firstly to develop an understanding of the use of traditional metrics to define a signal in a graph time series. Secondly, to complement existing measures with novel ones which will enhance our analytical capabilities for the structure of dynamic graphs. Third, to develop visualisation techniques to enable multiple attribute analysis and finally aggregation measures which can serve as a proxy for network activity.

1.4.1 Research Question

The research question can be stated formally as follows:

How can similarity measures be used to analyse dynamic email networks? How can similarity measures in alternative spaces support the analysis of dynamic networks?

The alternative spaces here refers to integral transforms of the data to new mathematical spaces. These are used extensively to derive the novel attributes for analysis. The key spaces are Hilbert, Fourier and Abel spaces. The Fourier space mainly returns the frequency while the Hilbert space is composed of the Complex Frequency. The Abel space is a form of dimensionality reduction which derives a lower dimensional cylindrical projection of a higher dimensional object.

1.5 Objectives

The objectives that will help us answer our research question are as follows:

- Explore similarity measures proposed for the analysis of networks
- Explore the use of these measures in a practical context
- Evaluate how such measures can be applied to the analysis of dynamic networks and develop a generalised work flow

- Derive novel attributes based on signal processing type approaches and benchmark their behaviour against existing measures
- Develop visualisation techniques to support multiple attribute analysis on dynamic networks
- Suggest aggregation schemes to serve as a snapshot of network activity over time

1.6 Overview of Methodology

The methodology followed in this study is as follows:

1. Source an appropriate email network network dataset with timestamps
2. Break data into yearly and monthly sets with no temporal aggregation
3. For the graph time series at the monthly and yearly level analyse the networks through traditional measures such as Centrality, Algebraic Connectivity, Density and Average Clustering Coefficient. These form the benchmark measures against which other measures are compared.
4. For the novel measures proposed determine which of the 3 graph matrices: Modularity, Adjacency and Normalised Laplacian yield the most stable attributes. This is done by analysis of the Signal to Noise Ratio, Mean Absolute Deviation, numerical magnitude of the attributes and their ability to model the signal in the benchmark measures.
5. Pick the Matrix yielding the most stable attributes using this to explore correlation among the measures, new visualisation approaches and aggregation schemes.
6. Scale all attributes to [-1,1] interval for comparability.
7. Perform correlation, regression and manifold reduction analysis to understand the relationship between all the different attributes
8. Derive node level trends for common nodes at the yearly level for centrality measures
9. Explain trends in the yearly and monthly trends relating to network visualisations while evaluating the ability of the measures to represent the graph time series signal

sectionBeneficiaries Although the analysis presented in this study uses an email network the methods proposed here are generalisable to any dynamic or static network. This is because the attributes are calculated from the matrix of the graph structure like the Normalised Graph Laplacian. Therefore any problem that can be modelled as a network can be expressed as a Normalised Laplacian Matrix and these attributes can be easily calculated from it. This opens up these methods to practitioners and analysts from a wide variety of fields as a result.

The main beneficiaries are practitioners or analysts who are involved or have an interest in the study of dynamic networks. This can range from social media companies trying to understand structural changes in their network and being able to identify easily the key players. wet could be useful to intelligence agencies analysing threat networks they have more tools to characterise the structural change in networks and identify times and nodes of interest. wet could also benefit telecommunications companies from example to understand how fast their network is expanding or contracting and more importantly predict some fundamental network property such as Average Degree at a time step in the future. Understanding potential trends in advance has the capacity to drive changes in strategy, enhance planning and deliver increased value for the business.

1.7 Report Structure

This report is structured into 6 key chapters. The in Chapter 1, the main aims, motivations, objectives and research questions are stated for clarity and final evaluation of the outcomes. In Chapter 2, we present a literature review where all the relevant information is summarised and presented. This serves to inform the rest of the analysis conducted. The analytical methods are presented in detail in Chapter 3. Chapter 4, shows all the results due to the applications of the measures in Chapter 3. We also present a detailed discussion noting our observations on structures and trends from the data. Chapter 5, presents a detailed evaluation of the methods and we answer the research question that we stated in Chapter 1. Chapter 6, concludes with our reflections, suggestions for further work and final conclusions of this study.

Chapter 2

Critical Context

2.1 Graph Terminology

The origins of graphs theory can be traced back to Leonhard Euler and his approach to solving the Konigsberg Bridge Problem. This city was located on the Pregel River in Prussia. The river divided this city into 4 distinct areas which included an island all of which were connected by a total of 7 bridges. Euler's representation of this problem of the individual areas as nodes and the bridges as edges is considered one of the first applications of graph theory.[8]

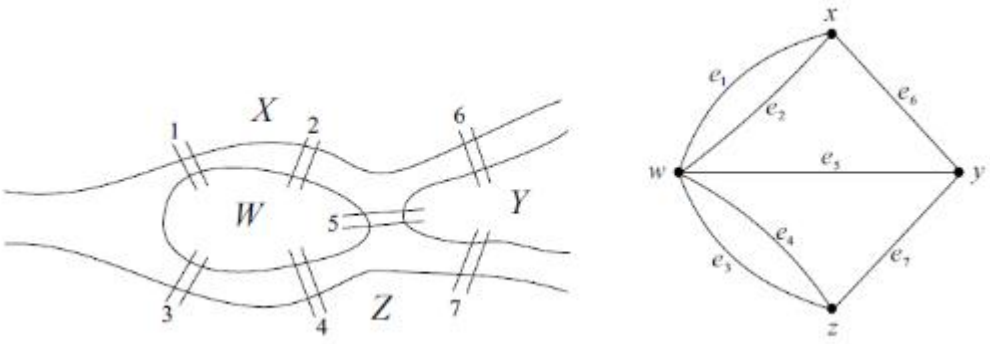


Figure 2.1: Euler's graphical representation of the Konigsberg Bridge Problem.

A graph, G can be described as a triple which consists of a set of edges E (G), a set of vertices or nodes V (G) and a relationship that connects the vertices to these edges. Finite graphs are those that have V and E as a finite set. Simple graphs are those that have no loops or multiple edges. A path is simple graph in which the vertices can be ordered where two vertices can be adjacent only if they are consecutively ordered. A cycle is defined as a simple graph where the vertices can be cyclically ordered such that two vertices are adjacent only if they are consecutive in cyclical ordering. A subgraph can be thought of cycles and paths within a larger graph, where the edge relations between the subgraph and the large graph are the same. [8]

2.2 The Limitations of traditional SNA

In Social Network Analysis (SNA), traditionally bounded networks are considered with maybe 2 or 3 connection or link types such as friendship or advice between a node types such as people sometimes another node type such as events are also considered together. [9]

If we consider more critically the interactions possible within our problem context of email networks we can have email networks within an organisation which are bounded and also with other organisations, clients and stakeholders and then the network does become unbounded. These networks can then be thought of as a higher order networks and as Carley [9] notes many tools developed for simpler networks do not scale well to increased network size and complexity and in some cases experience degradation through increased susceptibility to Type 1 and Type 2 errors.

The dynamics in these networks can arise from different processes depending on the context of the problem. Natural evolutionary processes would be learning, births, deaths and ageing Others could be as a result of intervention measures such removal or addition of nodes i.e. removing those who lead the system, communities forming or disintegrating. The data associated with such systems are also often incomplete and contain errors which make the process of analysis and evaluation of these systems.[10]

Analysis approaches that go beyond traditional SNA and link analysis are therefore necessary. Within the context of such dynamic networks analysis can be performed to identify of key individuals, locating hidden groups and estimate performance. The data analysis process on such networks then involve: [9], [10]

- Relationship identification among nodes
- Network structure characterisation
- Locating the elite within the network
- Identifying points of vulnerability
- Comparing networks

The approaches that enables effective analysis of such dynamic networks and help quantify their evolution over time is the motivation for this research.

2.3 Dynamic Network Analysis as an Extension to SNA

Dynamic network analysis (DNA) aims to extend the methods, tools and techniques used in traditional Social Network Analysis (SNA) to the analysis of networks which are able to handle big dynamic multi-mode, multi-link networks with varying levels of uncertainty. Dynamic networks also allow for probabilistic connection between

nodes. [9], [10]

In Carley, Diesner, Reminga, et al. [10] DNA was explored within the context of terrorism networks. Here an additional layer of complexity is added by the fact that an act of measurement changes its properties and this change propagates through the network and its state changes. Another key point is that the nodes in this network have the ability to learn. So the nodes themselves can be thought of being probabilistic compared to the more static nature of SNA nodes.

In a DNA representation system can be represented as relational data. This relational data structure can lend flexibility in defining multiple node types defined as multi-modal, have various types of connections among such nodes called multi-plex. The underlying attributes of both node, edges and the data change over time hence the dynamic part. [10]

In Carley, Diesner, Reminga, et al. [10] the key advances that allow for the analysis of such dynamic networks are identified as:

- The meta matrix
- Probabilistic edges between nodes
- Combining social networks with cognitive science and multi-agent systems

2.3.1 The Meta Matrix

The Meta matrix is a method used in operations research and organisational management that seeks to represent the entity and class relationships as a collection of networks. In the DNA context this translates as a multimode, multiplex approach to representing systems. Therefore, the Meta matrix can contain a social network, a membership network and knowledge network and allow us to explore and analyse the connections between them. [9]–[11]

2.3.2 Probabilistic Ties

The ties or connections in the Meta matrix are probabilistic with various factors affecting their probability. This allows for inclusion of the observers' uncertainty and the likelihood that the tie is present at the time of observation. These probabilities themselves and their temporal evolution maybe estimated by the Bayesian methods, cognitive inferencing and models of social and cognitive change. Carley

2.3.3 Multi Agent Network Models

As previously discussed the SNA treatment of nodes as static agents unable to learn is insufficient when dynamic networks are concerned. In DNA the nodes are able to take actions, learn from experience and alter their networks as a result. Some social and cognitive processes that influence the agent's interactions are relative similarity, relative expertise and co-workers. The dynamic behaviour of the network emerges from these interactions and experience a shared evolution. [9]

We briefly discuss some of the more common measures associated with networks which relate to their global and local properties. These will be important when we discuss similarity because one of the ways to assess similarity is to consider snapshots of a network attribute at different time intervals.

2.4 Network Measures

Centrality measures are a fundamental statistic in network analysis.Two paradigms of centrality definitions are suggested. One is the means based definition of centrality or the graph theoretic and the other is the ends-based definition which is a dynamic model based view that focuses on the outcome for the nodes in a network where there is flows across the nodes . However, both approaches agree that this measure is a node level property. [12]–[14]

The network measures used in this analysis are:

- Degree Centrality
- Closeness Centrality
- Betweeness Centrality
- Katz Centrality
- Load Centrality
- Density
- Average Clustering Coefficient
- Algebraic Connectivity

These measures are described in detail in Chapter 3, but the reason they are chosen is that the centrality measures are well understood node level properties and Density, Average Clustering Coefficient and Algebraic Connectivity are good network level properties that indicate the growth or contraction of the network as well as its communities.

2.5 Overview of Similarity Methods

Similarity in a networks is classified as being of structural, content or keyword based. The Structural similarity or link based similarity considers the similarity of links between the nodes in the graph e.g. Cosine, Jaccard, Hub Promoted and Hub Depressed Index etc. Content similarity considers the attributes of the node in the graph. For example, on a social network this could be birth dates or hobbies of individuals. Keyword similarity aims to find similarity based on nodes representing word collections. Global Structural Similarity can be classified as being:

- Local vs. Global

- Parameter free vs. Parameter dependent
- Node Dependent vs. Path Dependent

The global structural measures aim to measure node similarity compared to the whole network. We will call them intra network similarity measures. [15]

Inter network similarity measures are described in [16]–[23]. These measures are classified by Ashby and Ennis [24] into three categories:

- Distance Based
- Feature Based
- Probabilistic

2.5.1 Distance Based Approach

The distance based approach is perhaps the earliest of the methods encountered which is based on edit distance Bunke [23]. Essentially this boils down to finding a sequence of operations such as deletion, insertion, or substitution minimising some cost function that will turn one graph into another. These involve detection and comparison of the graph isomorphism, subgraph isomorphism and maximum common subgraph detection utilising the edit distance. Although these methods are guaranteed to converge to an optimal solution their exponential complexity makes them unsuitable for large graphs.

2.5.2 Feature Based Approach

This involves calculation of a network attribute such as degree, closeness, betweenness, and/or eigenvector centrality for the graphs and then applying a similarity measure on them that will characterise their similarity or dissimilarity. This has the benefit of being scalable to very large networks as the aggregated statistics are much smaller than the network themselves.

2.5.3 Probabilistic Approach

The methods that fall under this approach in the literature are vast. Some approaches under the probabilistic framework for graph matching are discussed here [21], [25]–[27]. But simply stated these methods define a probability distribution over mappings or graph embedding's [22], [27]. Graph embeddings are graphs whose nodes correspond to distinct points on a plane and the edges represents relationships connecting these points. The matching algorithm is strongly dependent upon the geometric information attached to the graphs[22], [27].

Graph matching allows for recovering point correspondences. In Zass and Shashua [21] the authors show that assuming that the assignment matrix that represents these correspondences are statistically independent the high order matching problem can be represented by a Kronecker product matrix. Also they show that that a high order tensor affinity tensor can be marginalised into a one dimensional vector of

probabilities. This probability vector is then updated by projection to a vector assignment space and then minimising a distance measure (Bregman measure) [28].

2.6 Graph Similarity

The problem of graph similarity or graph matching then becomes one of finding the equivalence of two graphs with potentially different number of nodes and edges and returning a measure within [0, 1] that captures their similarity or dissimilarity. [15]–[19], [21], [24], [29]–[31]

The key idea of graph matching in the context of dynamic networks can be summarised as finding a subgraph or an attribute that we can compare between two time instances. For example, if we consider the Degree Centrality of a network at time step 0 and then again calculate this measure at time step 1 we can apply a similarity measure on this attribute to quantify the change within the network. This can be done by means of a distance metric such as cosine similarity and others are possible.

The evaluation of the change in metrics over time will be done through a statistical control process. This is a concept that comes from quality engineering and it essentially involves calculating a statistic from a sequence of measurements of a random process and then comparing it so some control limit. This process translates to:

1. Calculating a cumulative sum control chart which is very good for detecting small changes in mean over time
2. Calculating a z-score for each time step $z_t = \frac{(x-\mu)}{\sigma}$
3. Construction of two charts to detect increase and decrease in the metric

2.7 Spectral and Tensor Methods

Spectral graph theory is the study of the eigenvectors and eigenvalues of graph matrices [26]. The spectrum of a finite graph is the spectrum of the adjacency matrix, which is the eigenvectors and eigenvalues derived from the eigendecomposition of this matrix. For an undirected graph without loops, the Laplace Matrix is the matrix indexed by a vertex set of v , with zero row sums if D is the degree matrix of a graph and A is the adjacency matrix then the Laplacian Matrix, L can be defined as $L = D–A$, where $Q = D + A$ is called the signless Laplacian Matrix of the graph. [26], [32], [33]

Spielman [26] note that since the eigenvalues of a graph do not depend on the vertex ordering of the graph then they could be used to distinguish between pairs of non-isomorphic graphs. If the eigenvalues for the graphs are different then two graphs can be considered different. But he notes that this approach has problems such as the eignevectors being only determined up to sign i.e. v and –v can both be eigenvectors, so spectral embedding comparison would result in having to check 2 K possible ways of flipping their signs. The eigenvectors can provide coordinates

for each vertex in a graph which is independent of the vertex labels but for graphs for which non-trivial eigenvalues have high multiplicity the coordinate flips in addition to its rotation must also be considered. Also the coordinates denoted by the eigenvectors are not unique which means that all eigenspaces must be considered to guarantee uniqueness of the coordinates of the vertices. Hence, this approach is problematic in practice.

Kunegis, Fay, and Bauckhage [32], suggest that since evolution or changes in a graph over time will lead to changes in its spectrum therefore an Eigen decomposition of the adjacency matrix can be used to characterise the this change. They then use the networks spectrum for link prediction and also discuss a method to reducing this link prediction problem to a 1D curve fitting problem.

Duchenne, Bach, In-so, et al. [34] formulate the hypergraph matching problem as a maximization of a multilinear objective function over a tensor representing feature permutations. The tensor represents the affinity between tuples of features. A multidimensional power method is used to solve the problem and the solution is then projected onto to the closest assignment matrix. The power method utilises a tensor Eigen decomposition and is applied to point matching using some similarity measure.

2.8 Visual Methods

More recently, the authors in Behrisch, Bach, Riche, et al. [35] have proposed visualising dynamic networks and characterising change by visualising the adjacency matrix of these networks as a matrix cube. Representing the adjacency matrix as a stack of cubes rather than node link diagrams is found to be a much more useful paradigm for analysis of dynamic networks especially when these networks are dense.

Behrisch, Bach, Hund, et al. [36] have proposed Matrix Diagnostics which ranks matrix views according to the appearance of some visual patters such as lines and blocks. These are taken as a proxy for network features such as clustering. This approach has been designed to aid in the analysis, query and identification of matrices with similar patters when there are a large collection of matrices. This can also be used to judge the effectiveness of the matrix reordering methods. These methods are particularly suited to dynamic networks as it can identify most similar matrices in a graph time series by their appearance similarity.

2.9 Contributions of this study

As we have shown from our reading of the literature that the tools applied are unprecedented in the literature and in addition to filling gaps in the current body of work we are also pushing the research in a new direction. A vast array of tools and techniques are utilised in this study. We show how aggregated centrality and assortativity measures can be used to establish a benchmark signal for a graph time series. Then we show that traditional signal processing techniques such as Fourier and Hilbert transforms can be used for the derivation of attributes. The Hilbert

transform used to derive a whole range of Seismic Attributes which have intuitive explanations and are easy to understand. In addition more exotic integral transforms such as Abel and Radon can be used for dynamic network analysis and visualisation of derived attribute volumes. These transforms allow us to treat the network as a music signal and opens up the field to attributes which have no parallel in network analysis to be used from Music Information Retrieval and Digital Music analysis.

These measures can not only be used in aggregated in form but can be used to derive a meta attribute that can serve as a snapshot of network activity to this end we suggest RMS and NRMS measures of aggregation. Also by using correlation analysis, correlation networks, regression and manifold reduction analysis we show the relationship between these attributes. A lot of these novel attributes are fairly well correlated to existing measures while a significant number of them are not strongly correlated to any of the other metrics but from a regression perspective important in prediction of network properties such as average Degree.

Chapter 3

Methods

3.1 About the data

The dataset chosen for use in this study is the Enron email data set. However this data has many different versions with multitude of pre-processing applied to it and the provenance of which is difficult to ascertain. Therefore, for this study we use the version of the data from John Hopkins [37].

The data set is in (time, from, to) tuple format. Here the time is encoded as seconds elapsed from 1 Jan 1970. The from and to fields are node numberings. These could be potentially mapped to employee id's but a viable data set to perform this particular operation had not been found so it was decided not to perform this operation but continue the analysis with the node numbers.

The data has a total of 5 years and 48 months that it can be segmented into. However, there is an encoding error that the first year occurs in the 1970's this is clearly wrong and is excluded from the analysis. Also there is no agglomeration performed on the networks so the networks at each time step contain only those items hence the networks are not cumulative.

3.2 Summary of Analysis Approach

Step 1: Data Acquisition and Quality Control	**Step 2: Exploratory Network Analysis**	**Step 3: Attribute Analysis**
1. Acquire Enron email network data with timestamps 2. Check data quality by examining time stamps for errors and checking data shape 3. Discard data with incorrect time stamps from the analysis 4. Segment data into yearly and monthly chunks	5. From the segmented data create networks without temporal aggregation from the yearly and monthly data 6. On these yearly and monthly networks derive centrality and network statistics 7. Use the centrality and assortativity statistics to establish a signal for the graph time series. 8. Derive network visualisations using clustered indices sorted adjacency matrices, node link and waveform diagrams	9. Using the Hilbert to derive a complex trace derive attributes commonly used in seismic data analysis 10. Using the Fourier Transform to derive real frequency components to derive an audio signal derive Music Information Retrieval Attributes 11. Derive additional novel attributes and those from literature 12. For all the attributes above determine which graph matrix gives the best attributes by looking at the Normalised Laplacian, Modularity and Adjacency Matrix 13. Use Entropy and Signal to Noise Ratio in addition to comparison with the signal of the benchmark measures in Step 2 to determine the best attribute set
Step 4: Attribute Volume Derivation	**Step 5: Attribute Voume Analysis**	**Step 6: Node Level Analysis**
14. Aggregate by averaging the Normalised Laplacian attributes derive an attribute volume for the graph time series at the monthly and yearly level scaling to [-1,1] interval 15. From the monthly attribute volume explore aggregation measures such as the RMS, NRMS and Emergence 16. Add these aggregation measures to the attribute volume to derive a final attribute volume	17. Use the attribute volume to compare signal of derived attributes to benchmark measures noting anomalies 18. From the attribute volume derive Pearson correlation matrice, Correlation Network, Correlation Network degree Histograms 19. Explore attribute similarity through additional techniques such as MDS and TSNE 20. Using the Average Degree as a predictive target perform Regression Analysis to derive feature ranking of attributes using a 50/50 train/test split	21. Use the yearly networks to identify common nodes 22. Use the centrality measures to explore the trends of these nodes over the course of the yearly timescales 23. Compare their behaviour to the yearly signal of the network in general noting correlations and anomalies.

For convenience the whole analytical process is summarised in the table above. The steps are grouped into logical stages that should make the reasoning behind the analysis much clearer.

3.2.1 Analytical Tools

The following tools are used in conjunction with the Python programming language for all the analysis in this study:

1. NetworkX - for graph analysis[38]
2. Numpy - for numerical computation [39]
3. Scipy - for statistical functions[40]
4. Pandas -for data structures and data analysis [41]
5. Matplotlib - for plotting [42]
6. Seaborn - for visualisation[43]
7. Scikit-learn - for matrix decompostion routines[44]
8. Librosa - for audio analysis[45]
9. PyAbel - for Abel Transforms [46]

Firstly, the network is decomposed into time steps at the year and monthly level. This was done by using the timestamps and converting them into dates. Once the dates had been derived the data was decomposed further into yearly time steps and the individual years were split further into monthly time steps. Not all months are available for all years so for the period considered the yearly data is for the years 1998-2002 and the monthly data starts from November 1998 to April 2002. This gives 5 yearly time steps and 48 monthly time steps. At each step of the analysis the yearly and monthly time steps a range of network measures are used to characterise the change over time.

The first step is to conduct some exploratory analysis of the data. This is done through primarily checking that the data has been subset properly into the years and months and then by visualising the networks. The visualisation of the networks also confirmed that the networks at both the monthly and year level are not unnecessarily dense. This could result if there is some aggregation in the data for example the network for 1999 contained data for the 1998 and so on. Therefore as an additional check the network sizes are compared with and without aggregation. The networks sizes and visualisations resulting serve as an additional Quality Control (QC) measure to give us confidence in the remainder of this analysis.

Once the networks at the different time steps had been created the exploratory analysis was conducted using traditional network measures. These measures serve as the benchmark measures for the graph time series. The reason being that these measures are widely used and understood. They serve to establish the potential

ground truth in this dataset and the signal that they represent can be used to asses the new measures proposed in this study.

The measures described in this section are then derived for both the monthly and yearly networks. All attributes are scaled to [-1,1] to enable comparison. Also this helps when we perform regression analysis for feature ranking.

These derived attributes are compared to the benchmark measures and deviations noted as points of interest. The correlation among all these measures are explored. This leads to the derivation of a correlation network of all attributes and correlation matrix which is reordered with cluster indices from hierarchical clustering and only plotted for measures where the correlation is > 0.7. In addition we utilise Multi Dimensional Scaling and TSNE of the final attribute volume to visualise the closeness of attributes in addition to the Correlation Network.

Also the issue of combining these various metrics into an overall global measure of network activity is explored. Here er use the Emergence and Persistence measures suggested by Wei and Carley[47]. Also we propose novel measures of aggregation based on the RMS and Normalised RMS or NRMS. The NRMS measure can be thought of as a normalised RMS difference between two traces. In this case the attributes are aggregated by RMS at each time step.

Regression Analysis is performed as an additional evaluation of these measures based on their predictive potential. For the regression analysis we use a 50/50 test /train set since the dataset is very sample and we want to prevent over fitting.This analysis enables feature ranking when trying to predict the Average Degree of the network at a future time step.

The final analysis attempts to look at node level dynamics in the network. This is node by finding common nodes across the years and then plotting centrality trends over time. This allows us to assess how individual nodes perform compared to the rest of the network.

3.2.2 Graph Matrix Representations

The attributes are derived from the matrix of the graph structure. The graph structure can be represented by different types of matrices such as the adjacency matrix, the Laplacian matrix and the Modularity matrix among others. It is important to note that in this study whenever the Laplacian is referred to the Normalised Laplacian is being referred to for brevity. This is because as Anderson[48] note that the eigenvalues derived from the Normalised Laplacian relate well to other graph invariants for general graphs that is not the case with other matrices such as the adjacency matrix. Another advantage noted is that the definition is more consistent with the eigenvalues derived from spectral geometry. Spectral geometry is an extension of spectral graph theory which incorporates more of an geometric approach in deriving graph properties through methods such as random walks and mixing of Markov Chains among others.

Brouwer and Haemers[33] describes the **Adjacency matrix** of a graph as a [0,1] matrix indexed by the vertex or node set of a graph $A_{xy} = 1$ if there is an edge from node x to node y and 0 otherwise. In the case of Multigraphs or graphs with loops this 1 is replaced by the count of the edges between nodes x and y.

The Normalised Laplacian or Laplacian in our case is defined by Anderson in the following manner. If we consider a matrix, L of a Graph where d_v is the degree of node v and the matrix L is defined as follows:[48]

$$L = \begin{cases} d_v, & \text{if } u = v \\ -1, & \text{if } u \text{ and } v \text{ are adjacent} \\ 0, & \text{otherwise} \end{cases}$$

Then the Laplacian matrix of the Graph cna be defined as:

$$\Gamma = \begin{cases} 1, & \text{if } u = v \text{ and } d_v \neq 0 \\ \frac{-1}{\sqrt{d_u d_v}}, & \text{if } u \text{ and } v \text{ are adjacent} \\ 0, & \text{otherwise} \end{cases}$$

This can be written as:

$$\Gamma = T^{-1/2} L T^{-1/2} \tag{3.3}$$

Here T is the diagonal matrix where containing the degree values, d_v.

The Modularity Matrix and Modularity function were developed as a means of better identifying community structure in graphs while minimising the influence of random factors. The modularity function, Q is the number of edges that fall in a community minus the the expected number of edges in an equivalent network with edges placed at random. This value is high for well formed communities and low for poor communities. This is defined as :[20]

$$Q = \frac{1}{2m} Trace(X^T M X) \tag{3.4}$$

Here

$$\sum_v k = 2m$$

where v are the degrees of node k and X is the assignment matrix where $\mathbf{X} = (x_{ih})$, $x_{ih} = 1$ if node i belongs to community h and $x_{ih} = 0$ otherwise. $\mathbf{M}$ is the modularity matrix defined as:

$$\mathbf{M} = A - \frac{k k^T}{2m} \tag{3.5}$$

Here k is a vector of degree of nodes and A is the adjacency matrix.

3.3 Selection of Graph Matrix Representation

In order to assess which matrix representation to use to Information Theoretic measures are used the Signal to Noise Ratio and Entropy.

The Signal to Noise Ratio (SNR) is defined as:

$$SNR = \frac{\mu}{\sigma} \tag{3.6}$$

The Entropy is defined as:

$$H = -\sum p(x) log_2 p(x) \tag{3.7}$$

The SNR in this context is to help us identify if attributes from one graph matrix are more noisy compared to the another. The Entropy measure is to help us asses the unpredictability of the information content. Hence lower average entropy would indicate less uncertainty. We assess the matrices by their mean entropy and high SNR attributes. The key deciding factor is how well these novel attributes from the different graph matrices represent the phenomenon identified by the benchmark measures. We find that the Normalised Laplacian attributes have lower average entropy than the attributes derived from the other matrices. This is discussed further in Chapter 4 and 5. The matrix finally chosen is due to the derived attributes ability to represent the signal identified by the benchmark measures. These information theoretic measures are the first step in this assessment process.

3.4 Benchmark Measures

The suite of benchmark measures are discussed here. These measures serve as snapshot of the network at each time step hence we can treat them as measures of similarity or dissimilarity of graphs within the graph time series.

Centrality measures are inherently node level properties and they can be broadly characterised as follows: [12], [14]

- Volume based measures – degree like centrality
- Length based measures – closeness like centrality
- Medial measures – betweenness like centrality

Volume and length based measures are what are called radial measures because they analyse walks that emanate from or terminate with a given node. Medial measures on the other hand are based on position so how many times does one encounter a node while trying to reach other nodes in the network.
We have mentioned some of the most popular centrality measures but there are numerous other variations mentioned in the literature and are beyond the scope of this work.

3.4.1 Degree Centrality

Degree Centrality is a special case of the k-path centrality that counts the all the paths of a length, k that originate from a given node. the k path centrality behaves like the degree centrality when k=1 . This is simply the count of the number of edges incident upon a given node which is equivalent to summing the rows of the adjacency matrix. This is the case in an undirected network while in directed network we have to take the difference between the number of nodes incident upon a node and the number of nodes emanating from a node to get a measure of degree. Since the network under consideration in this study is undirected we will restrict our discussion to undirected interpretations of our measures. [12]

$$c_i^{Deg} = \sum_j a_{ij} \tag{3.8}$$

3.4.2 Closeness Centrality

Closeness Centrality is the graph theoretic distance or the geodetic distance from a given to all the other nodes in a network. This essentially the marginals of a geodetic distance matrix. [12], [14]

$$c_i^{Clo} = \sum_j d_{ij} \tag{3.9}$$

3.4.3 Betweenness Centrality

Betweenness centrality counts the number of times that a certain node, x needs to pass another node, y to get to another node, z through the shortest path between them. [12], [14]

$$c_i^{bet} = \sum_x \sum_y \frac{g_{xky}}{g_{xy}} \tag{3.10}$$

Here g_{xy} is the number of geodesic paths from nodes x to y and g_{xky} is the number of these geodesic paths that pass through the node k.

3.4.4 Eigenvector Centrality

The Eigenvector Centrality is defined as the principal eigenvector of the adjacency matrix of a network. It captures the intuition that nodes that have high eigenvector centrality scores are likely to be close to other nodes which themselves have high values for this measure. [12], [14]

3.4.5 Katz Centrality

Katz[49] introduced this centrality measure which derives the centrality value of a node based on the centrality of its neighbours. This is a generalisation of the Eigenvector centrality measure. The Katz Centrality is defined as:

$$c_i^{katz} = \alpha \sum A_{ij} c_j + \beta \qquad \text{where;} \qquad \alpha < \frac{1}{\lambda_{max}} \tag{3.11}$$

Here **A** is the adjacency Matrix of the graph with eigenvalues λ and β controls the initial centrality.

Katz Centrality measures is also a measure of relative influence of a node within the network because it takes into account the number of immediate neighbours and also all the other nodes that connect to the node through these neighbours. When the β parameter is set to 0 the Katz centrality is identical to the Eigenvector Centrality. In this study this value is set to 1.

3.4.6 Load Centrality

Load Centrality is described by Goh, Kahng, and Kim, Newman[50], [51] as fraction of all shortest paths that pass through that node.

3.4.7 Density

This is defined as the total number of edges divided by the total number of possible edges. [29]

3.4.8 Diameter

The diameter of a network is the maximum geodesic distance between two nodes.[29]

3.5 Complex Trace Attributes

The Complex trace attributes make use of the Hilbert Transform. So before we introduce the measure an introduction to the Hilbert Transform is presented here.

3.5.1 The Hilbert Transform

The Hilbert Transform is an integral transform that extends a normal and real valued function into the complex plane. This allows the derivation of two useful attributes the Instantaneous Amplitude or Energy Envelope and the Instantaneous Frequency. These fundamental complex trace attributes form the basis for the derivation of the other attributes as they are derivatives of these quantities. A Hilbert transform does not change the domain of the function so a time domain function remains a time domain function while a frequency domain functions remains in the frequency domain. In the time domain this translates to a $\frac{\lambda}{4}$ shift for all frequencies and a $-90°$ phase shift for all spectral components in the frequency domain. [52]

3.5.2 The Complex Trace

As mentioned that these attributes are inspired by their use in seismic attribute analysis so the Complex Trace or Analytical Signal is introduced here here. Essentially it is the signal after the Hilbert Transform has been applied to it so that it has a Real and Imaginary component. Their use in seismic attribute analysis is discussed in detail by Li and Zhao, Subrahmanyam and Rao[53], [54]. The descriptions

included here are to build intuition around these attributes and to get a sense of what we can expect them to highlight.

The complex trace is defined as

$$C(t) = S(t) + iH(t) \tag{3.12}$$

Where:
$C(t)$= Complex Trace
$S(t)$= Real Data
$H(t)$= Hilbert Transform of data

Based on the above the Complex Trace Attributes introduced are shown.

1. Instantaneous Amplitude, IA
2. Power
3. Instantaneous Phase, IP
4. Instantaneous Frequency, IF
5. Derivative of Instantaneous Amplitude , dIA
6. Second Derivative of IA , d2IA
7. Instantaneous Acceleration, IAcc
8. Amplitude Weighted Instantaneous Phase
9. Amplitude Weighted Instantaneous Frequency

3.5.3 Instantaneous Amplitude

Instantaneous Amplitude is widely used in traditional tectonic and stratigraphic interpretation. As one of the basic parameters of the amplitude attribute, it helps delineate the high- or low-amplitude anomaly (bright or dark spots). In this context this should highlight the bright and dark spots in the network when used as an attribute map and should show the highest and lowest points when used as a time series. As I show later this amplitude has a high correlation (0.7) with the traditional centrality measures so its behaviour is very similar to those. Therefore it is highly plausible that for this data it should suffice to look at this attribute instead of many different centrality measures.

$$\sqrt{S(t)^2 + H(t)^2} \tag{3.13}$$

3.5.4 Power

The Power is calculated as the square value of the amplitude. In this case I use the Instantaneous Amplitude. This allows us to better understand the signal envelope as this attribute is smoother than the Amplitude.

$$IA^2 \tag{3.14}$$

3.5.5 Derivative of Instantaneous Amplitude

The derivative of IA highlights the change in reflectivity and shows sharp interfaces and discontinuities. Effectively this should highlight the big changes in the IA and the smaller changes should make the attribute smooth. This is what we observe because the attribute highlights the peaks observed in the IA plot and the rest of the signal is fairly smooth for the time range.

$$\frac{\mathrm{d}}{\mathrm{d}t}(IA) \tag{3.15}$$

3.5.6 2nd Derivative of Instantaneous Amplitude

The second derivative of the IA highlights the interfaces very well - the places of change. This attribute is not too sensitive to the amplitude and can highlight even weak events.We see this to be the case where the first derivative highlights the individual peaks the second derivative smoothes the individual peaks and gives a smooth peak over the range of months where they occur and does a better job of highlighting the peaks towards the end of the timeseries than the first derivative.

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}(IA) \tag{3.16}$$

3.5.7 Instantaneous Phase

Instantaneous Phase is expressed in degrees or radians at the selected sampling point. Instantaneous phase helps strengthen weak reflections in the inner parts of reservoirs and also strengthens the noise. Because the hydrocarbon accumulation often causes phase changes, this attribute can be used as a direct indication of hydrocarbons. The cosine of the instantaneous phase is derived from the instantaneous phase. It is commonly used to improve the variation display of the instantaneous phase because it has fixed boundary values (-1 to +1). In this context the change in the IP can be used as an indicator for the intervals which are the most interesting since they have a noticeable phase change. This is better illustrated by the cosine of IP especially the percentage change plot here we see the months with the greatest phase change really well and the peaks are well delineated in contrast to the just the IP plot.

$$\arctan(\frac{H(t)}{S(t)}) \tag{3.17}$$

3.5.8 Instantaneous Frequency

Instantaneous frequency is defined as the time derivative of the instantaneous phase and it is used for estimating seismic attenuation. Oil and gas reservoirs often cause the attenuation of high-frequency components, so this attribute is also conducive to measuring stratigraphic periodic intervals. In this context I expect this to show

clearly the peaks of the signal. This is exactly what we see especially in the monthly trend where the two large peaks are clearly delineated with a some smaller peaks highlighted the attribute is smooth over the rest of the series. So it has the property of finding the most significant peaks in our data.

$$\frac{\mathrm{d}}{\mathrm{d}t}(IP) \tag{3.18}$$

3.5.9 Instantaneous Acceleration

Instantaneous Acceleration is defined as the rate of change of the instantaneous frequency, which is often used to indicate the rate of attenuation and absorption. As gas (or oil, or water) can cause the attenuation of seismic waves, this attribute can represent a fluid interface in the high-resolution data. As a derivative attribute I expect it to highlight the peaks and troughs very effectively and be smooth in areas of less pronounced change. This is what is observed as the attribute highlights the two large peaks but with the opposite polarity of IF.

$$\frac{\mathrm{d}}{\mathrm{d}t}(IF) \tag{3.19}$$

3.5.10 Amplitude Weighted Instantaneous Frequency

Amplitude weighted Instantaneous Frequency provides a reliable smooth instantaneous frequency estimation in order to reduce the interference damage. I expcect this to better highlight frequency anomalies in the data and suppress insignificant anomalies. We see that the weighted IF is sharper in places where the IF is fairly smooth. This highlights the major peaks as well as minor anomalies not immediately obvious from the IF alone.

$$IA.IF \tag{3.20}$$

3.5.11 Amplitude weighted Instantaneous Phase

The amplitude weighted IP should cause the phase to become sharper with the peaks and troughs more accentuated. This has the effect of magnifying the signal as well as the noise. However, in this case we see that the trend in IP weighted and regular IP plot are identical with the only difference being the magnitude of the peaks.

$$IA.IP \tag{3.21}$$

3.6 Matrix and Matrix Decomposition Attributes

The novel matrix decomposition based metrics proposed in this study are:

1. Norm of the Forward Abel Transform

2. Mean Gaussian Curvature
3. Kernel PCA 3 Component Ratio Change
4. Norm NMF Ratio Change

In addition some measures suggested in the literature are also implemented such as:

1. Resistance Distance [55]
2. Stationarity Ratio [56]
3. Subgraph Stationarity [57]
4. Power Spectral Density [56]

3.6.1 Norm of Forward Abel Transform

The Abel transform finds a slice of a cylindrically symmetric 3D object and provides a 2D projection of it. In essence it does a dimensionality reduction by finding a lower dimensional subspace for the higher dimensional projection. The Inverse Abel Transform recovers the original higher dimensional data space. In this attribute I use the Forward Abel Transform through direct numerical integration of the Abel equations.

The Abel Transform attribute is calculated as follows:

1. For the Normalised Graph Laplacian, Γ take the Fourier Transform, $F(\Gamma)$.
2. Calculate the magnitude of the Fourier Transform of the Graph Laplacian $\sqrt{Freq_{real}^2 + Freq_{imag}^2}$
3. Take the Forward Abel transform of the magnitude of the Fourier Transform $A(|F(\Gamma)|)$
4. Take the Norm of the resulting matrix at each time step, $\|A(|F(\Gamma)|)\|$

This results in a lower dimensional representation of the Frequency content of the Laplacian Matrix. This is useful for highlighting change over time as this attribute and highlights when the underlying networks have expanded or contracted with regards to their activity.

3.6.2 Mean Gaussian Curvature

There are many ways to define curvature I use the Gaussian Curvature which is derived from the Hessian Matrix. A surface might be curved upward in places, curve downward in places, or even be flat in places. Also, at some given point, the surface may be curved upward in some directions and downward in others. The curvature measure helps us detect such change. Gaussian curvature can be positive, negative, or zero. A useful property of the curvature attribute is that it is independent of orientation of the surface. If we place vectors on this surface the would indicates where the curve bends i.e., the vectors are either diverging, converging, or parallel.

This can correspond to the negative, positive or zero value of curvature.

$$K = \frac{f_{xx}f_{yy} + f_{xy}f_{yx}}{(1 + f_x^2 + f_y^2)^2} \tag{3.22}$$

The mean curvature is the trace of the eigenvalues of the Hessian Matrix

$$K_{mean} = trace(\lambda_{hessian}) \tag{3.23}$$

3.6.3 Kernel PCA 3 Components Ratio

Here I calculate the Kernel PCA 3 Component ratio from the Normalised Graph Laplacian. This is the ratio of the main element difference between two networks. The reason a Kernel PCA is used is because of its ability to handle non-linearity through kernels. Here the RBF kernel is used to fit the Normalised Graph Laplacian and then the ratio is calculated as follows:

Step 1: Fit and transform the Normalised Graph Laplacian keeping only the first 3 components: PC1, PC2 and PC3

Step 2: Calculate the Kernel PCA Ratio

$$KPCA_r = \frac{PC1 - PC3}{PC1 - PC2} \tag{3.24}$$

Step 3: For all the networks calculate the change in the KPCA Ratio as:

$$\Delta K_r = \frac{K_{rt0}}{K_{rt1}} \tag{3.25}$$

recursively to derive a rolling measure of the change in KLPCA Ratio over time.

The KLPCA is one of the main element analysis methods in seismic attribute analysis where it is used to calculate the correlation in a multitrace window. A small value represents a degree of intermittent or no correlation of geological phenomena. It is also used to detect discontinuities, such as faults and unconformities. Here the purpose is to locate big discontinuities in our networks over time. This represents a scalalble and easy way to locate the biggest changes even when we are dealing with a large number of dynamic networks. Most of the values encountered are relatively small. But from the monthly plot we see that there is a large discontinuity in June and July 1999 whose scale dominates the plot. Hence the log of this attribute is used and we see other smaller signals emerge as a result.

3.6.4 Norm Non-Negative Matrix Factorisation Ratio

This is another metric derived from the Non-negative Matrix Factorisation of the graph matrix. The NMF involves finding two non-negative matrices (W, H) whose product approximates the non- negative matrix X. I take the Frobenius Norm of the NonNegative components derived from the data and calculate change in this ratio over time.

I calculate the NMF Ratio change as:

$$NMF_r = \frac{||WH_{t1}||}{||WH_{t0}||} \tag{3.26}$$

3.6.5 Resistance Distance

The resistance distance between vertices i and j of a graph G is defined as the effective resistance between the two vertices (as when a battery is attached across them) when each graph edge is replaced by a unit resistor. This resistance distance is a metric on graphs.
I calculate the resistance distance as:

M =Graph Matrix
N = Length of M
P = Moore-Penrose Pseudo Inverse of M
D = The diagonal of P

$$R_d = (D \otimes (N,1))^T + (D \otimes (N,1))^T - P - P^T \tag{3.27}$$

Here $\otimes$ denotes the Kronecker or outer product

3.6.6 Stationarity Ratio

The Stationarity Ratio is based on the SVD decomposition of the graph matrix and then calculated as a ratio of the norm of the diagonal elements to the norm of the derived matrix. This is calculated as follows:

L = graph matrix
U = eigenvalues of L
C = covariance of L

$$\begin{aligned} CF &= L(U^T C U) \\ S_r &= \frac{\|diag(CF)\|}{\|CF\|} \end{aligned} \tag{3.28}$$

3.6.7 Subgraph Stationarity

This is computed from the adjacency matrix of graphs and is done in two steps. Essentially this is comparing the common subraphs between two networks and deriving a correlation score. This is done in two steps.
Step 1: Calculate Correlation, Ct between the graphs at time, t and time, t+1.

$$C(t) = \frac{A(t) \cap A(t+1}{A(t) \cup A(t+1)} \tag{3.29}$$

Step 2: Calculate the Subgraph Stationarity, ζ

$$\zeta = \frac{\sum_{t=0}^{tmax-1} C(t, t+1)}{tmax - t0 - 1} \tag{3.30}$$

From the Subgraph Stationariy ζ we can calculate the amount of members that change at each time step as:

$$1 - \zeta \tag{3.31}$$

3.6.8 Power Spectral Density

The Power Spectral Density (PSD) describes the distribution of power over frequency for a given time series or signal. The Power in this case can be thought of as not necessarily as physical power but as squared of the signal. This is similar to the Power attribute we introduced earlier but this allows us to understand its distribution. [58]

This is related to the auto correlation function and thus we can utilise this attribute like the amplitude to identify Amplitude or Power Anomalies.

3.7 Music Attributes

3.7.1 The Fourier Transform

I briefly describe the Fourier Transform as this is essential in derivation of the Music Attributes. The Fourier Transform is used here to extract the frequency components of the Normalised Graph Laplacian. Since for audio applications we need either a stereo or a mono channel we combine the frequency components into 1 audio channel by averaging across the rows of the frequency components. This results in a single audio channel which represents the underlying graph. From this it is possible to extract a number of Music Attributes such as the Zero Crossing Rate and Spectral Centroid. These two are chosen as they have intuitive interpretations which can be easily understood. Other attributes related to beats and tempo are more difficult to explain thus could be the subject of further work but not actively explored in this study.

The Fourier Transform is discussed in detail by Tao[59]. As with the Hilbert Transform the focus here is to build intuition and the reader is referred to materials cited for more thorough treatment of the subject.

The Fourier Transform allows us to decompose functions in a systematic way into a superposition of symmetric functions such as trigonometric functions. These are related to physical concepts such as frequency and energy. The Fourier Transform is a reversible linear transform and is fundamental in the study of groups and is related to many linear algebra topics such as representing a vector as a linear combination of an orthonormal basis or as linear combinations of eigenvectors of a given vector. Changing the basis of the eigenvectors allows for the calculation of the Fourier Basis on Graph which will allow for more broader range of signal processing methods to be applied. [56]

3.7.2 Zero Crossing Rate

The Zero Crossing Rate (ZCR) is perhaps one of the most widely used measure in the field of speech and audio analysis. This is simply the number of zero crossings or the number of time the signal crosses the zero line within a defined region or a window over the signal. [60], [61]

The ZCR measure can be evaluated over a fixed, moving or variable sized window. This measure estimates for the waveform complexity in the time domain. It is very useful in prividing a general trends with regards to the level of the overall frequency content. For this study we utilise this measure in the frequency domain this is also called the Spectral Zero Crossing Rate but the Zero Crossing Rate is used in the discussion because the underlying data is audio. The ZCR in the frequency domain provides information related to the transients location within a time window. Since the dataset is essentially a graph time series. The audio waveforms can also be thought of as a Audio Time Series. Hence we are essentially characterizing the whole audio signal at each time step by their average ZCR. This gives a measure of similarity for comparing the waveforms. The higher ZCR is indicative of greater change in the signal envelope. In this case since we are using the average of the ZCR we can interpret this as highlighting the fact the signal here is changing more rapidly compared to others which also leads to these regions having higher Power. This helps identify interesting time periods in the graph time series. [61]

3.7.3 Spectral Centroid

The Spectral Centroid is measure for the characterisation of spectra and is widely used in digital music to classify the brightness of a sound. It calculates the centre of mass of a signal using the weighted mean of the frequencies. The frequencies are derived via the Fourier Transform and the weights used are the magnitudes which are similar to the Amplitude. This could be thought of as being similar to the Amplitude weighted Frequency attribute introduced earlier. But instead of the Hilbert Transform the the audio signal derived from the Fourier Transform is used instead. [62]

3.8 Aggregation Schemes

3.8.1 Persistence and Emergence

The Persistence measure used in this study utilised the time averaging of attributes. This can be stated as

$$P_{it} = Agg(m_1, ..., m_n) \tag{3.32}$$

The aggregation measure here is averaging but could be any number of aggregation functions such as linear or exponential aggregations are also possible. This measure is essentially all the measures at a time step averaged into one value and then normalised by the length of the time series or $max(t)$. The Emergence measure is then just the normalised Persistence measure depending on whether the Persistence values are positive or negative.

$$E_{i,t} = \begin{cases} 0, & \text{if } P_{i,t} = P_{i,t-1} \quad \text{or } t = 1 \\ \frac{P_{i,t} - P_{i,t-1}}{N_{i,t}} & \text{otherwise} \end{cases}$$

$$N_{i,t} = \begin{cases} max(P_{i,t}, P_{i,t-1}) \text{ for non-negative Persistence} \\ \\ |||P_{i,t}|| + ||P_{i,t-1}|| \text{ otherwise} \end{cases}$$

This is comprehensive measure that allows all the above measures to be combined into a single measure for network characterisation. The Emergence measure is an indicator of network activity bursts can indicate periods of growth while troughs can indicate contraction. They help give an high level overview of the trend in the graph time series and captures network dynamics well when we use the attributes because they characterise the trends in the network the traditional measures do not highlight as well.

3.8.2 NRMS of Network Attributes

The NRMS measure can be though of as a normalised RMS similarity measure in the sense that when two traces are similar the NRMS value will be close to 200 and when there is a great dissimilarity it will be less than 200. This constant helps to exaggerate the trends in the data while suppressing noise due to the use of the RMS operator.
The NRMS measure is defined as follows:[63]

$$NRMS = \sqrt{\frac{200RMS(a-b)}{RMS(a) + RMS(b}} \qquad RMS = \sqrt{\frac{\sum (x)^2}{n}}$$

3.9 Visualisations

3.9.1 F-K Plot

The F-K Plot is a visualisation technique used in seismic data analysis that isolates the signal in a central cone and separates the noise to enable easy filtering in the FK domain. The FK plot consists of the Frequency,f and Wavenumber, k. The Frequency is derived by taking the Fourier Transform of the attribute volume. The wave number is then calculated as the reciprocal of the frequency. Plotting this also highlights that most of the signal is concentrated near the the central cone with k values around 0 and that the outliers have high/low values and are thus separated from the signal cone. These frequency indices from the F-K plot can be used to order the data and we can highlight which the months that standout on the F-K plot.

In addition the Frequency, F and Wavenumber, k can be visualised individually as attribute maps with mappings to the months and attribute names. This allows for the identification of anomalies.

3.9.2 Radon Plot

The Radon Transform is an integral transform that has various applications ranging from image processing, medical imaging, computer vision and seismic analysis. The transform can be defined as

$$R(p, \tau)[f(x, y)] = \cup(p, \tau) \tag{3.35}$$

Here the p is the slope and τ is the intercept. In the image analysis context the Radon Transform computes projections along a axis. Since the Radon transform used here comes from an image analysis package this is the form utilised in this study. The plots of the projection of the Radon components are plotted.

The Radon transform is used in seismic data analysis to seperate events based on their velocities. This is possible with the $\tau - p$ implementation of the Radon Transform. However, in image analysis the projection method is used. Since this particular implementation computes a sinogram consisting of projections of the input at different angles the analogy to velocity here is not possible. Typically the Radon plot is used where the p trace represents the iverse of velocity and the τ represents time at zero offset or intercept. But the transformed space is still very useful as a form of visualisation for the high dimensional attribute volume.

This plot helps easily visualise clusters in the data and is an additional visualisation tool that can also be used for outlier detection and filtering. The original data can also be reordered from the Radon Matrix in order to see which are the highest and lowest values in the context of the Radon plot. The resulting Radon Matrix can also be visualised as a Heatmap hence the trends observed in the Radon plot can be linked to the graph time series.

3.9.3 Log Panel

The Log Panel is a multi-attribute visualisation technique where the y-axis is the time series and the x-axis represents the range of the values. This helps to put multiple attributes side by side and enable the tracking of peaks of troughs in addition to being very conducive for multiple attribute interpretation.

3.9.4 Audio Waveform Plot

The Audio Waveform Plot is a plot the amplitude envelope of a waveform. In our case the audio signal is one channel so we get a monophonic plot. The length indicates the length of the frequency trace which is a proxy for the length of the matrix. The frequency content is determined by the Laplacian Matrix which is a proxy for Degree. So high Degree correlates with high frequency which correlates to spikes in the waveform plot. This is a very compact representation of the graph matrix as we are able to gain both an overview of the characteristic of the network as well as a sense of the node level dynamics driving those changes.

3.10 Manifold Dimensionality Reduction

3.10.1 Multi Dimensional Scaling

Multi Dimensional Scaling is a visualisation technique that allows us to see the similarity of items in a dataset. The purpose of using the MDS is to explore visually the attribute similarity since the MDS would attempt to plot similar attributes closer together.

These methods that work on a distance matrix derived from the original data. The MDS routine is an optimisation technique hence the solutions are non unique. This means that for the same data we can get a different representation each time the routine is run. But despite the layout variance the proximity of similar objects will be similar. Hence more similar objects will be closer while dissimilar objects will be farther. The MDS algorithm tries to position each item in a pre-defined N dimensional space while preserving inter object distances through some cost function. [64]

3.10.2 T-Distributed Stochastic Neighbourhood Embedding (TSNE)

TSNE is a non-linear dimensionality reduction technique where the probability distribution over pairs of high dimensional objects are created such that similar objects have a high probability of being picked and dissimilar objects have a low probability of being picked. This is achieved by defining a low dimensional map over the points and then minimising the Kullback-Leibler (KL) divergence between the two distributions according to the locations on the lower dimensional map. This results in a lower dimensional mapping which shows the similarities present in the high dimensional dataset.[65]

This is used as an alternative to the MDS visualisation as the TSNE method is more sophisticated. The use of the probability distributions with KL divergence allows it to handle arbitrarily large dimensions. Also it is able to utilise a wider range of distance metrics so we can also see how the choice of the distance metric will affect our perception of the similarity of the attributes.

Chapter 4

Results

4.1 Discussion of Results

In this section we have shown the results of the methods discussed extensively in Chapter 3. To begin with we started with some familiar visualisations such as node link and matrix diagrams of the networks at the yearly and monthly intervals. These are denoted in Figures 4.1, 4.2, 4.4 and 4.5.

4.1.1 Yearly Network Visualisations

From the yearly plots we see that the network is most sparse in 1998 and gradually gets more dense throughout the time period and then shrinks again in 2002. This is reflected in the waveform plots in Figures 4.3 and 4.6. Here we see that the signal for 1999 is very blocky indicating probably that not many nodes are present in the network hence the amplitude envelope is smooth and blocky. But in the cases where the networks are dense we see that the signal is spiky indicating the greater number of nodes who have an associated amplitude and some have higher amplitude than others. These nodes are probably the more central players in the network. This is reflected at the monthly level as well as the first few months of the graph time series the networks are comparatively sparse compared to the middle. The network starts to thin out in the last few months of the time range.

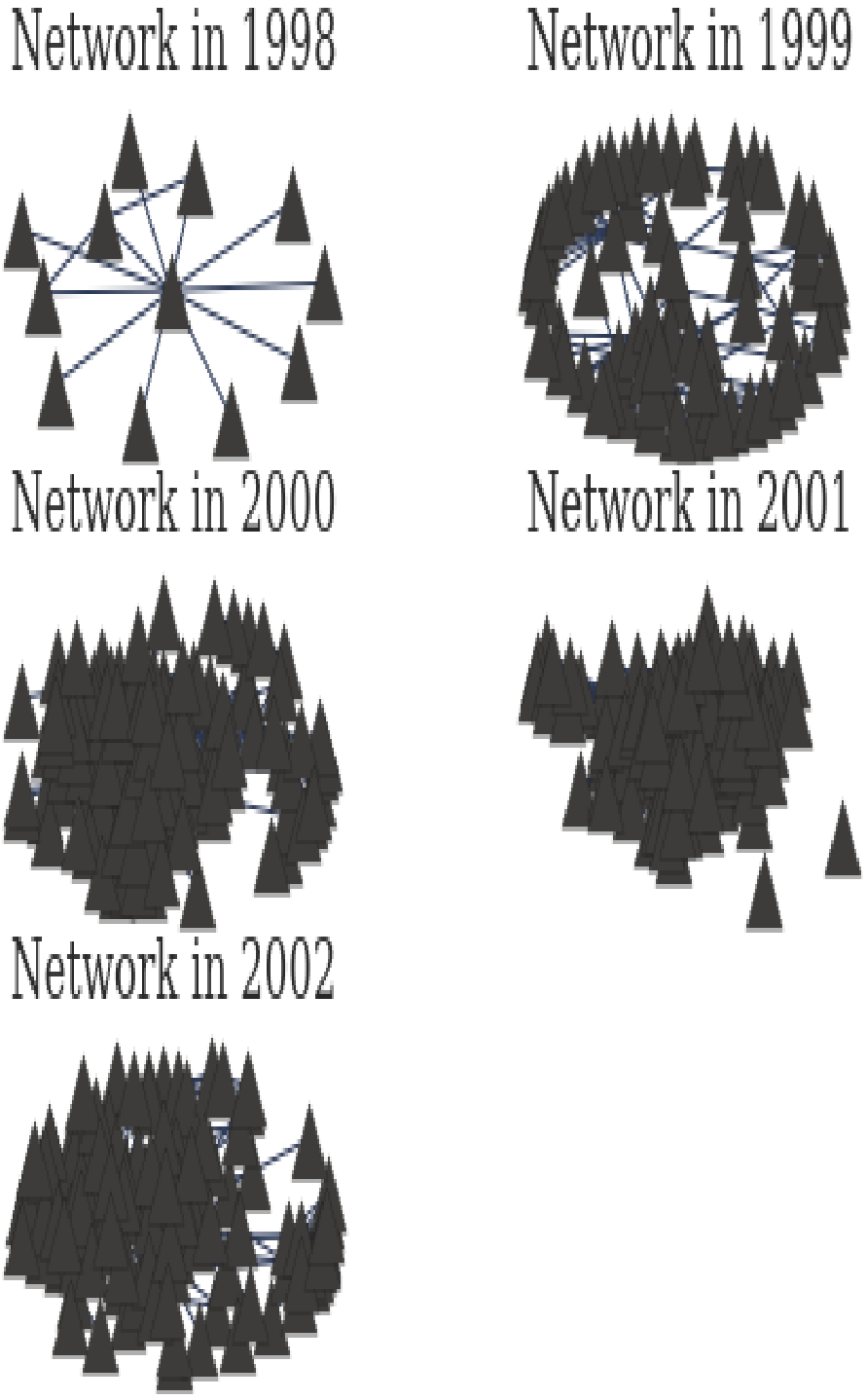


Figure 4.1: Node Link Diagram of yearly networks. This shows that as the network density increases this plot becomes a hairball and unusable.

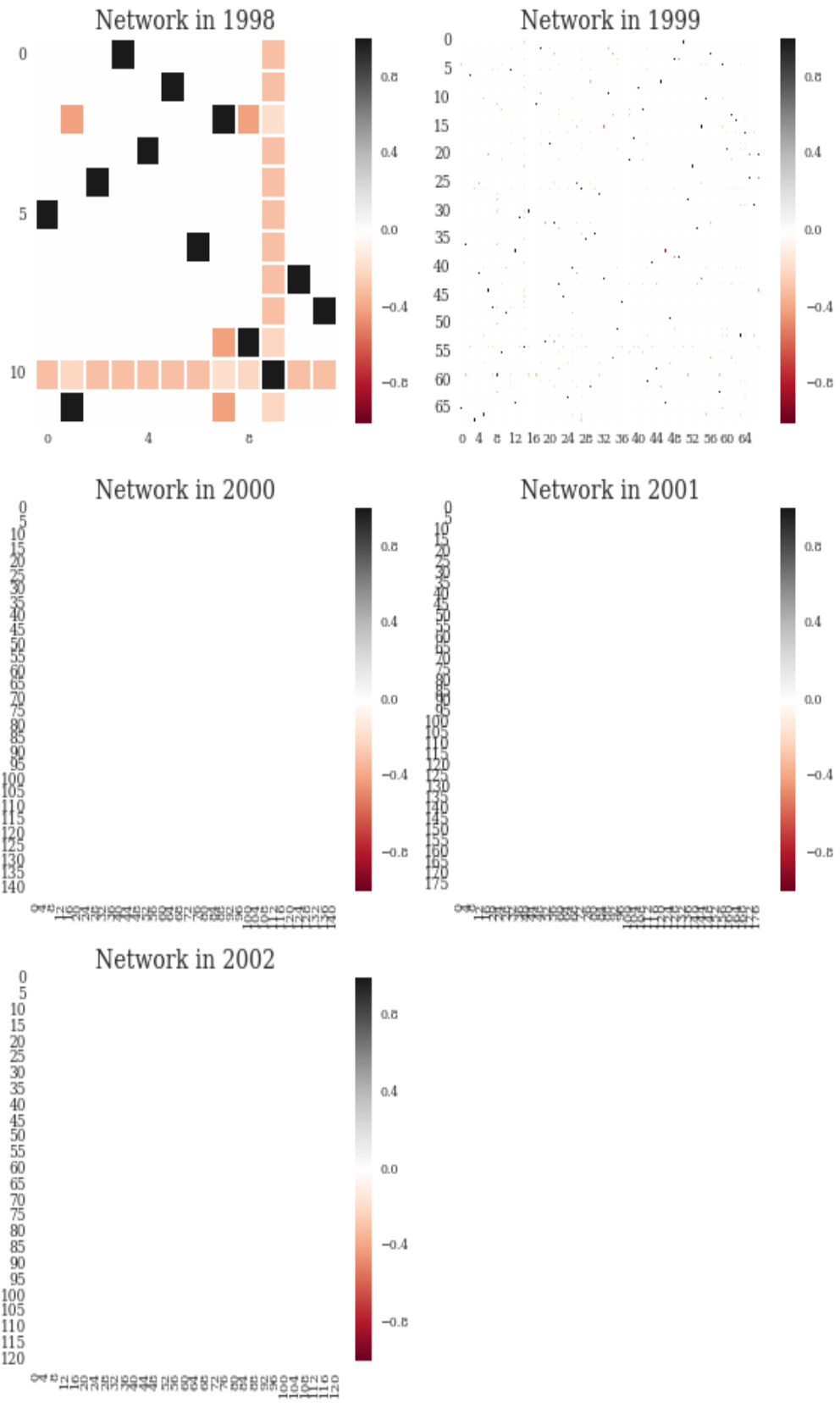


Figure 4.2: Reordered Matrix Diagram of yearly networks. The Matrix view is also not very useful when represented in a static setting such as a report and requires a dynamic representation to enable us to utilise it fully.

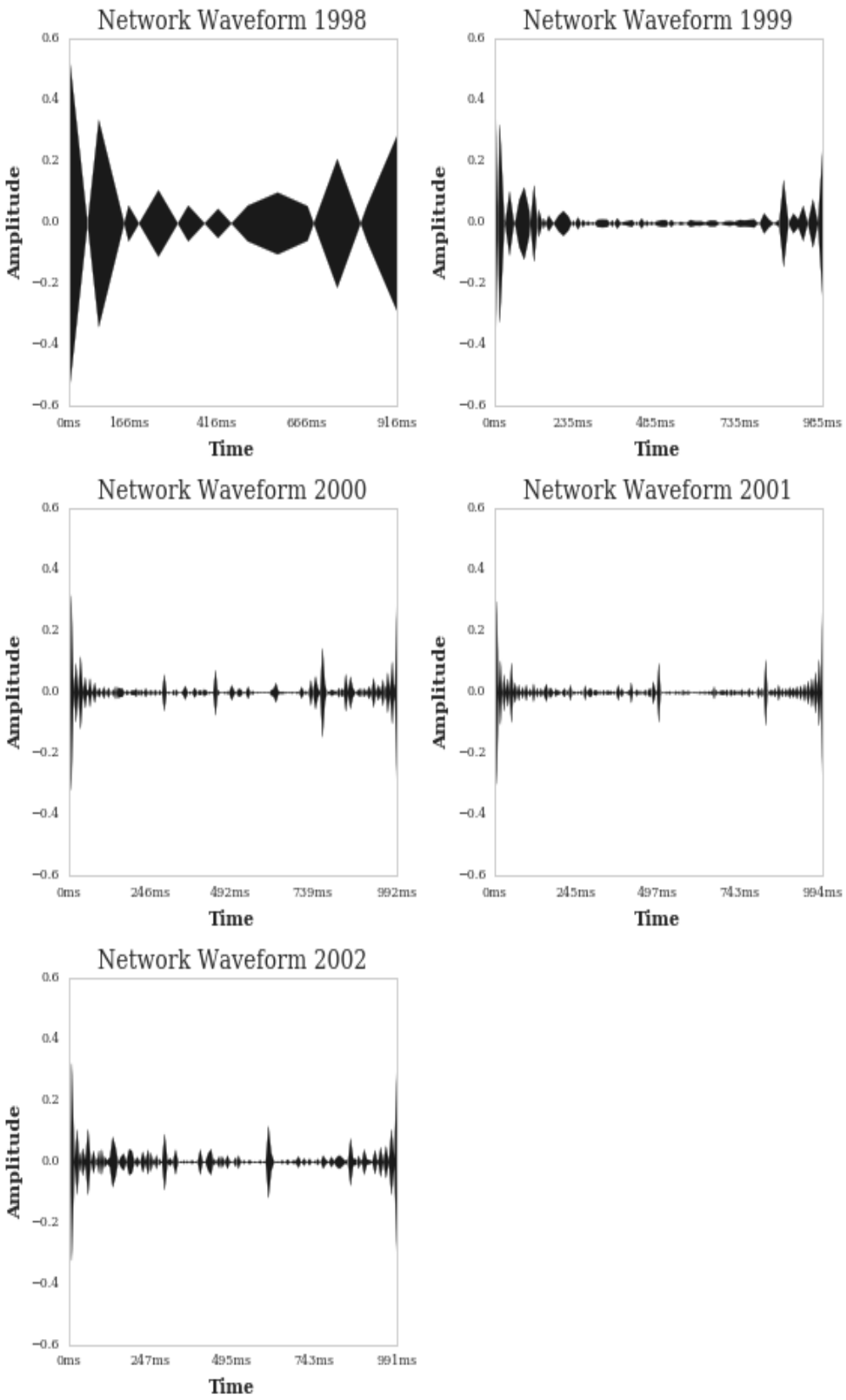


Figure 4.3: Audio Waveform plot of yearly networks. This is the more useful out of the previous two visualisations as we can get a good overview regardless of network density.

4.1.2 Monthly Network Visualisations

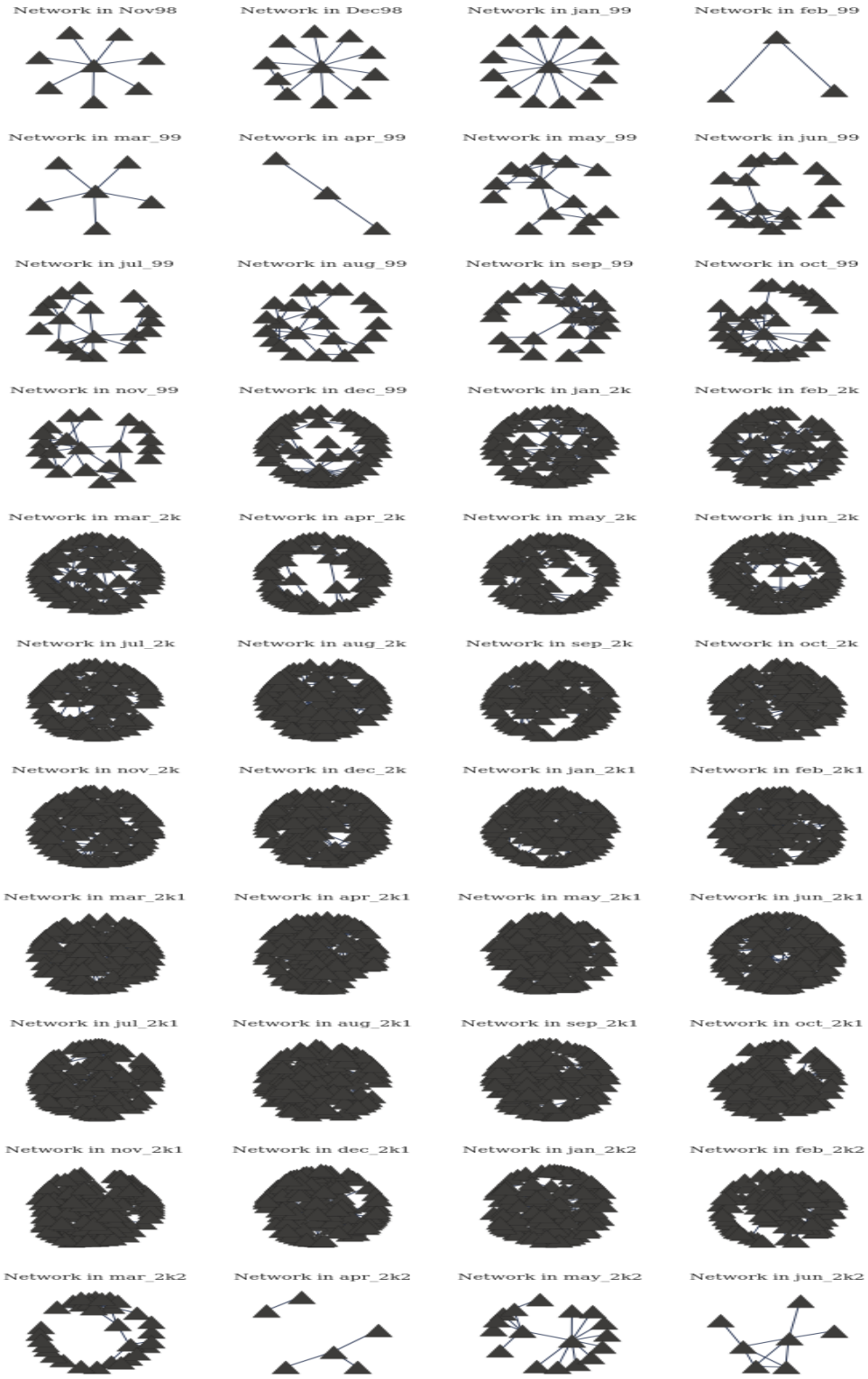


Figure 4.4: Node Link Diagram of monthly networks. We see the same hairball problem as the network density increases over time.

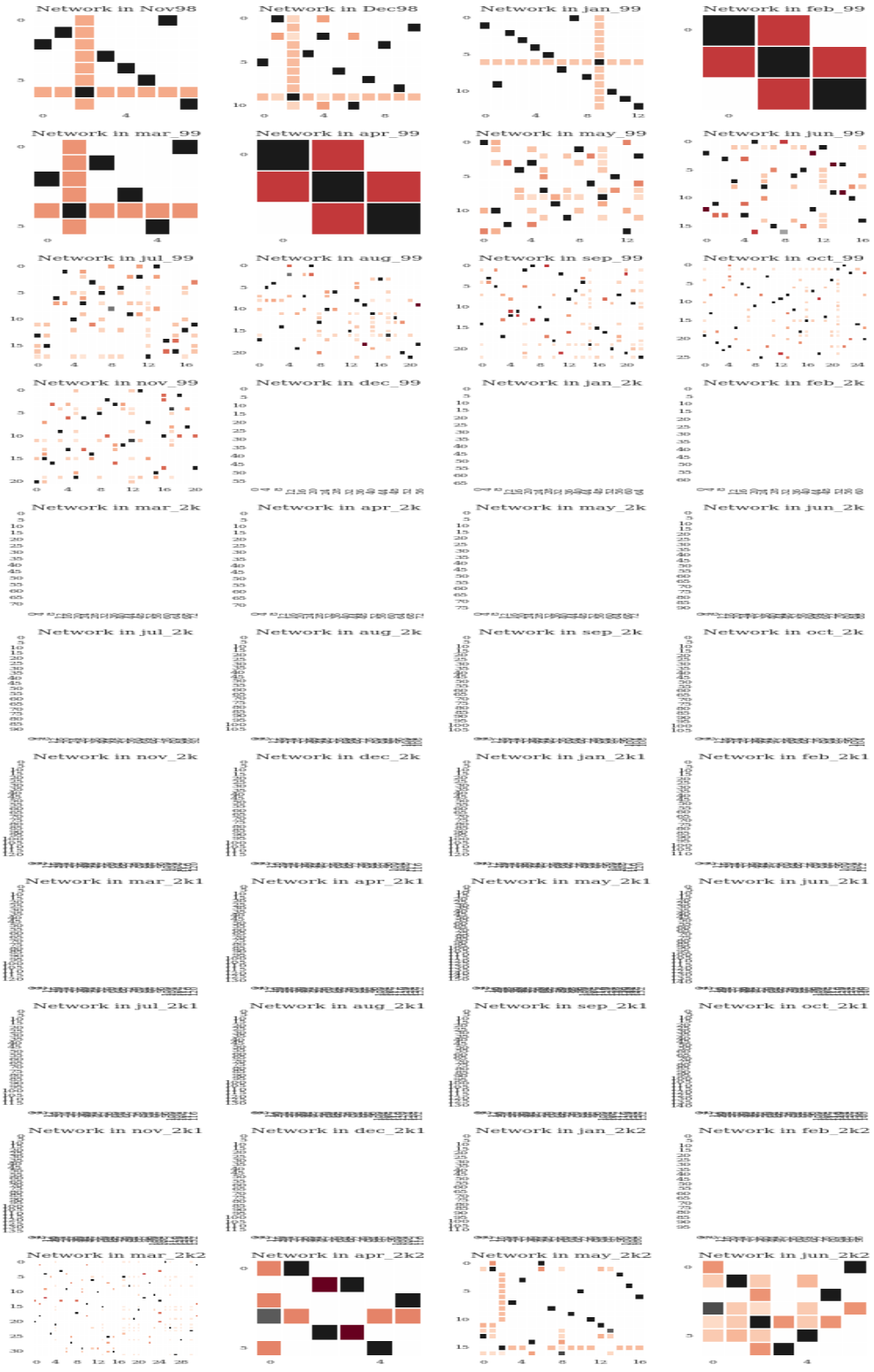


Figure 4.5: Reordered Matrix of monthly networks. Given that this is a highly compressed view we are not able to see anything useful over the dense parts of the network.

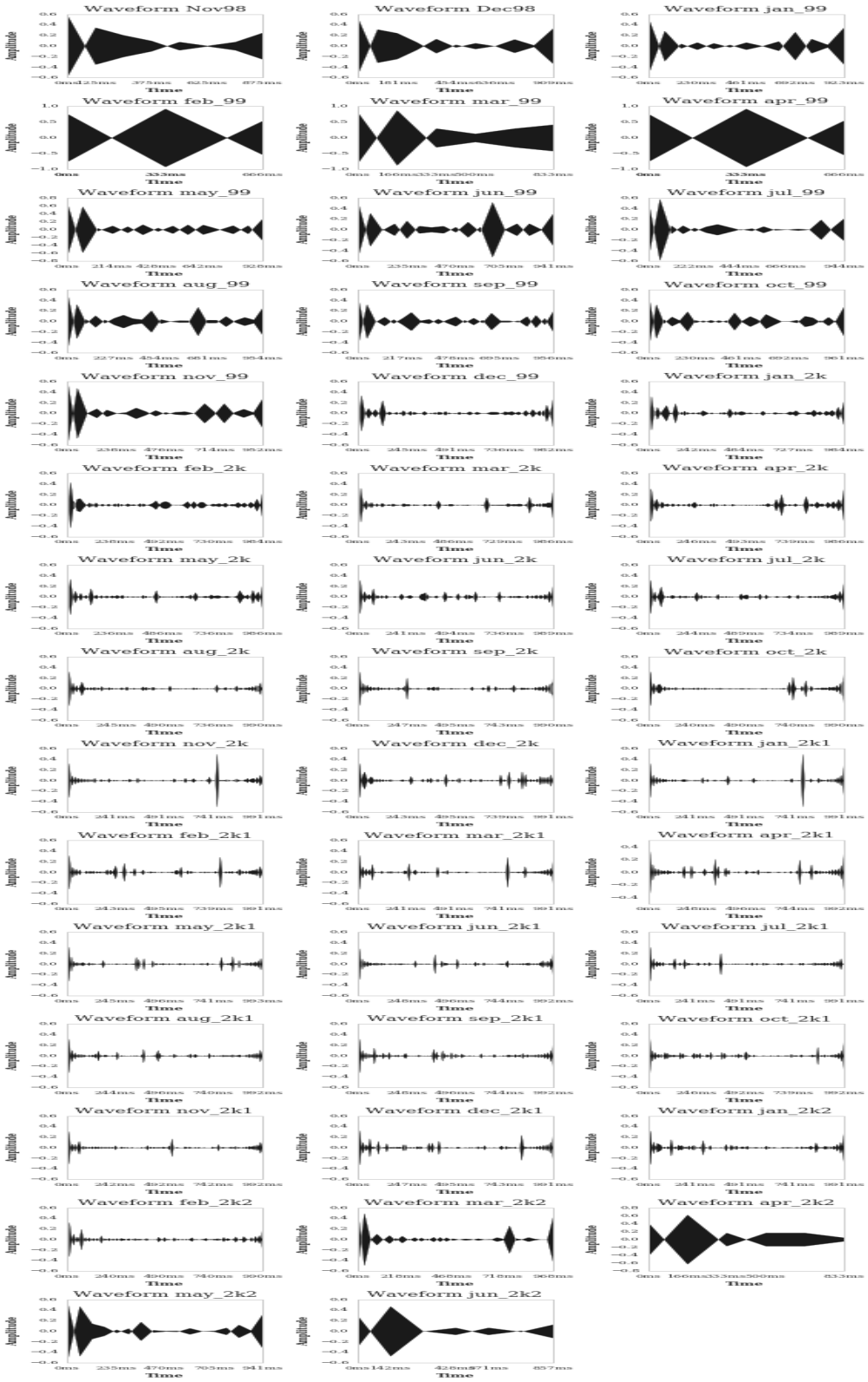


Figure 4.6: Audio Wavefrom of monthly networks. Dspite the large number of plots at the monthly level we can still visualise the trends in the network reasonably well in this plot.

4.2 Exploratory Analysis: Centrality Measures

The exploratory centrality histograms at the year and monthly level are what is observed in networks more generally. This is that they exhibit a long tail with most nodes having low values and some nodes having high values. The only other key observation from the figures in Section 4.2 are the extremes at the beginning of the time series where the networks are thin. This is reflected by the small number of nodes overall in both the monthly and yearly networks. In the case of the monthly networks a similar trend is observed where there are an overall smaller number of nodes due to thinning of the network which results in a sort of bimodal appearance of the centrality histograms. For most of the time series we see the long tail in centrality value distribution.

4.2.1 Yearly Analysis

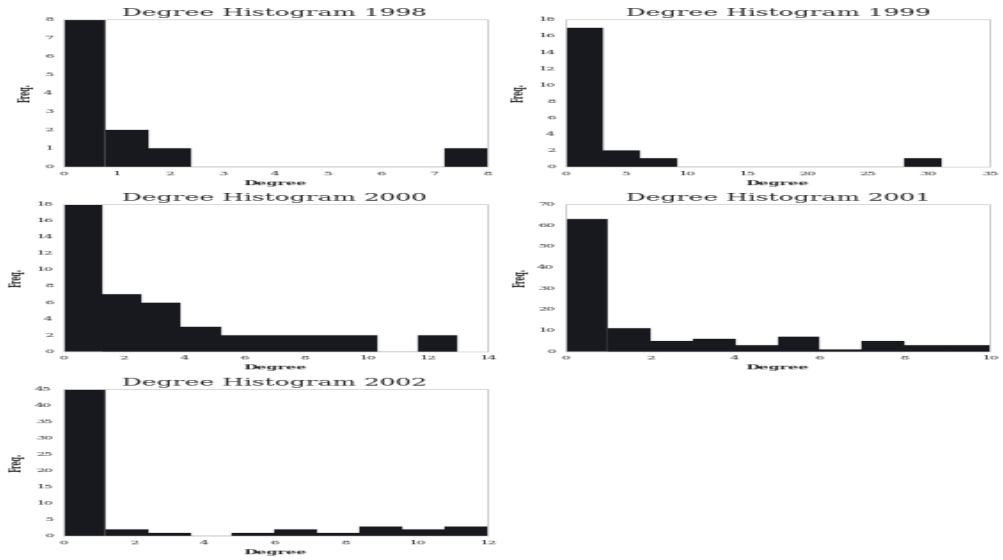


Figure 4.7: Degree Histogram for yearly networks showing typical power law behaviour of the long tail.

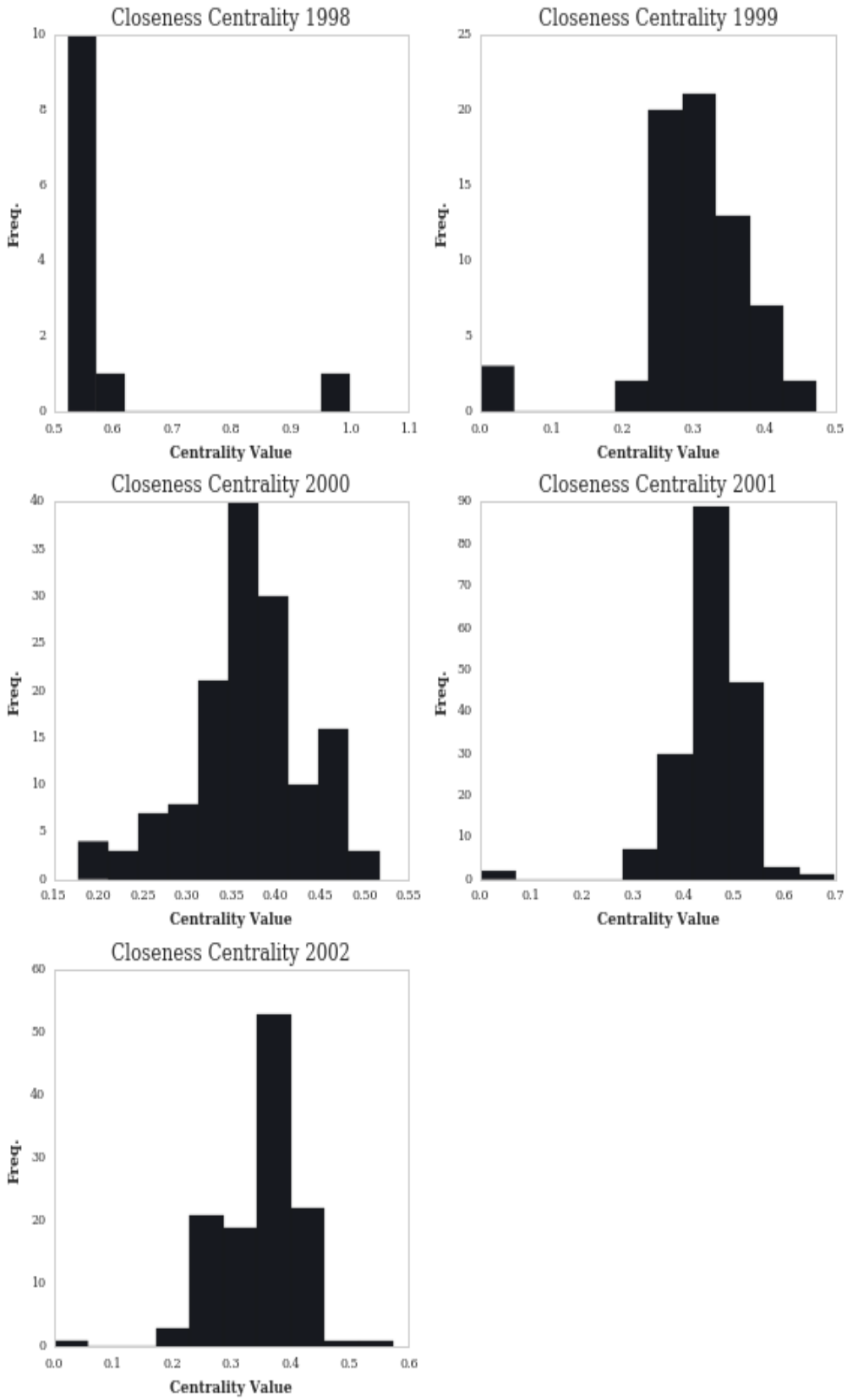


Figure 4.8: Yearly Closeness Centrality Histogram

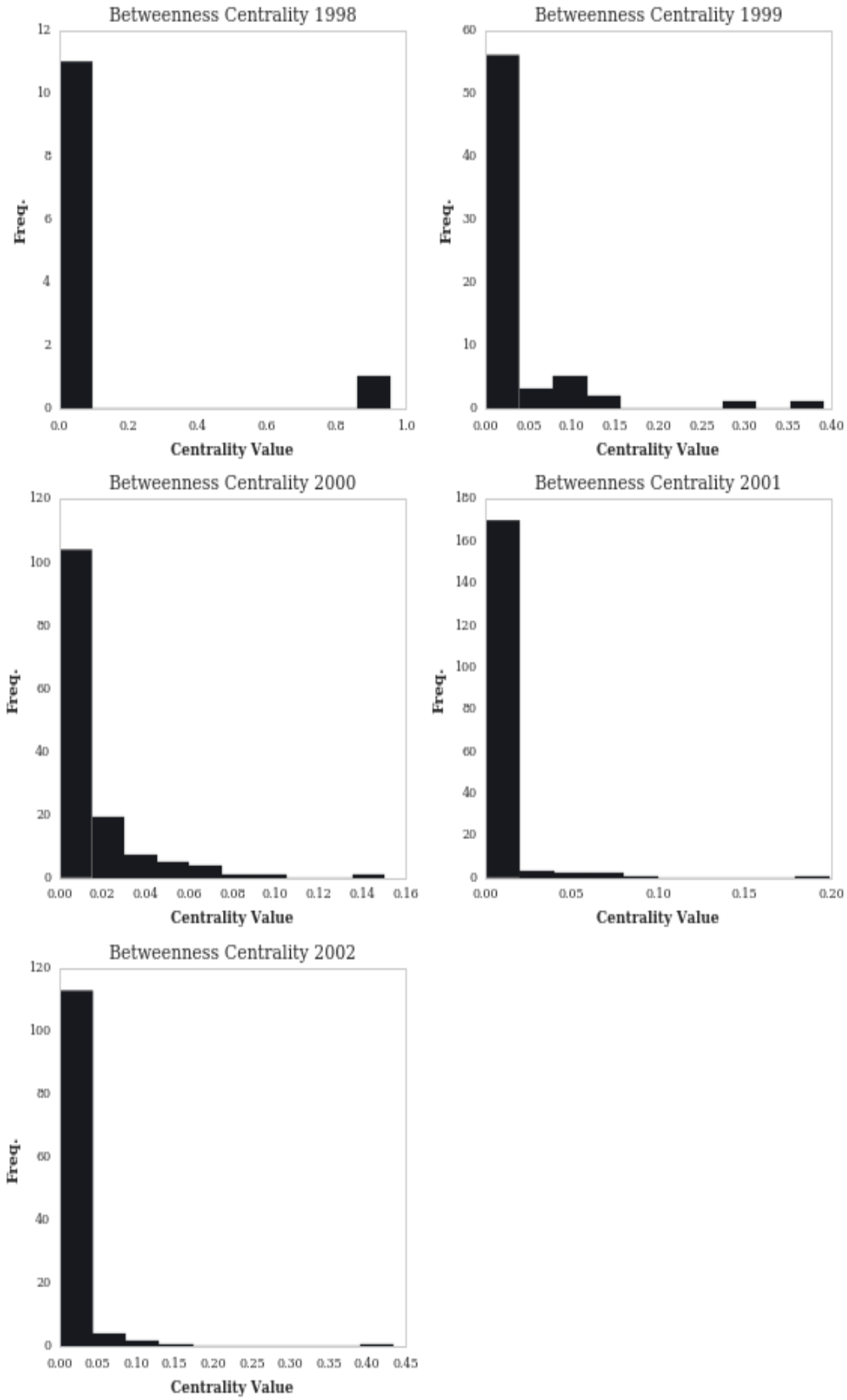


Figure 4.9: Yearly Betweenness Histogram

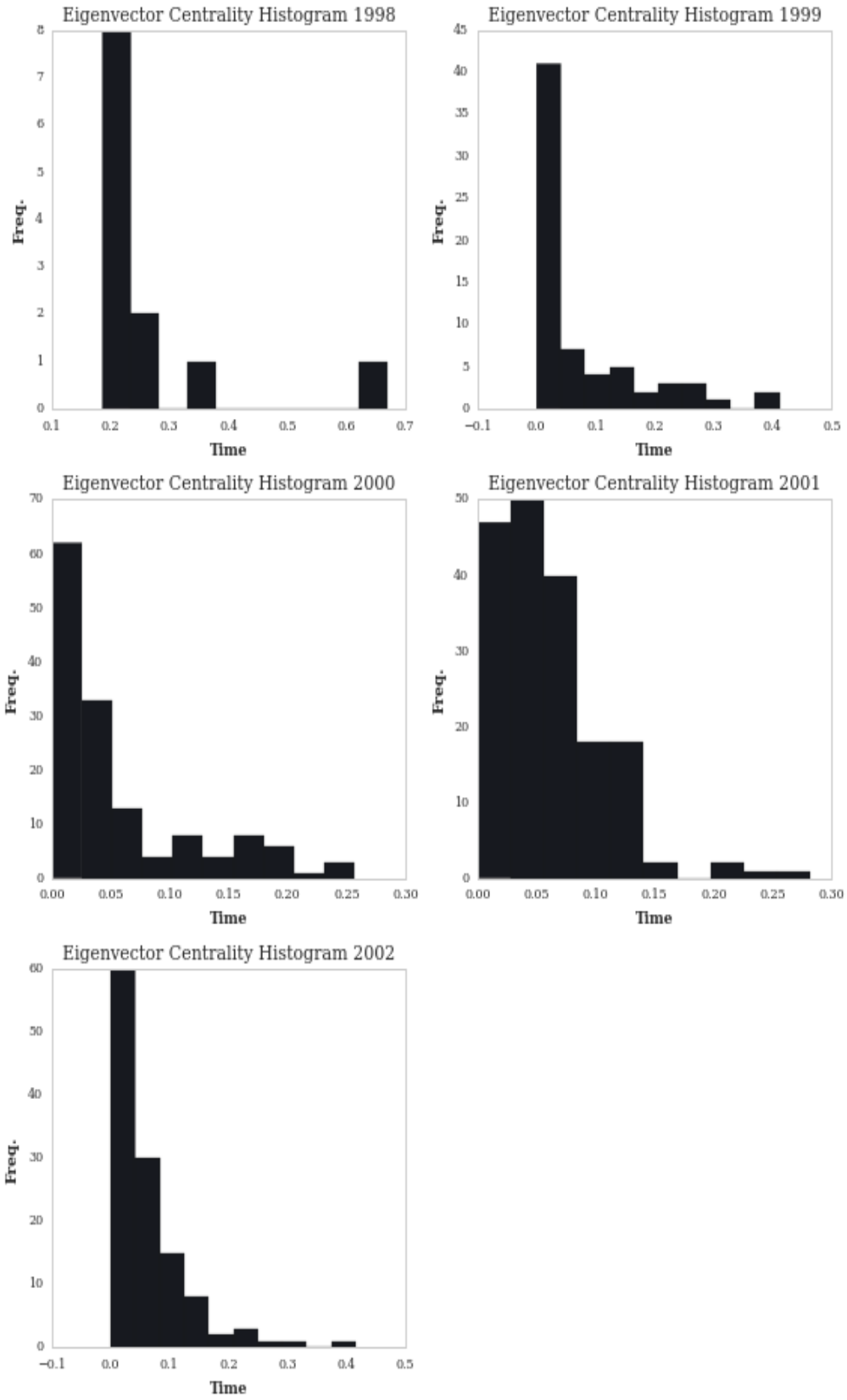


Figure 4.10: Yearly Eigenvector Centrality Histogram

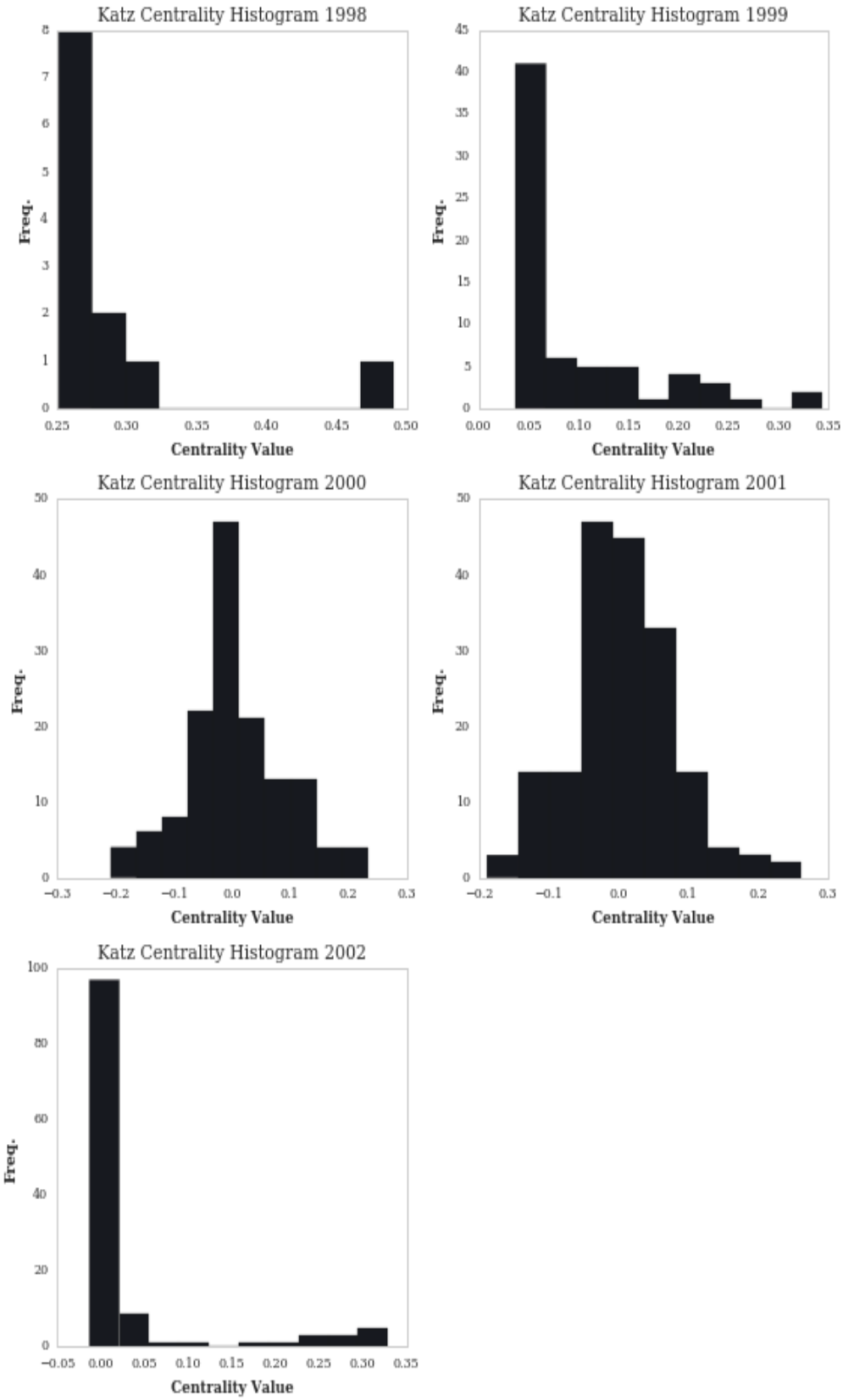


Figure 4.11: Yearly Katz Centrality Histogram

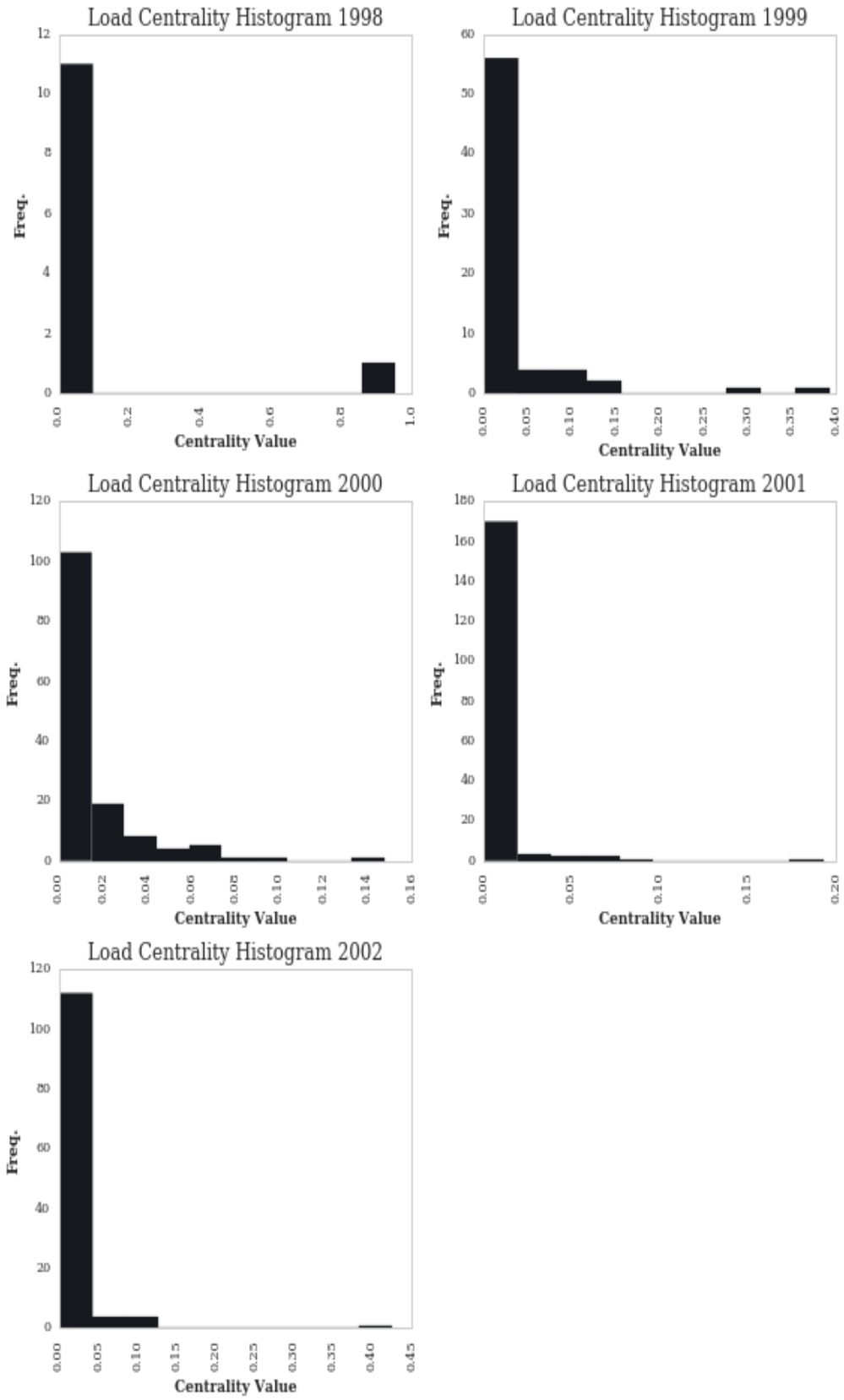


Figure 4.12: Yearly Load Centrality Histogram

4.2.2 Monthly Analysis

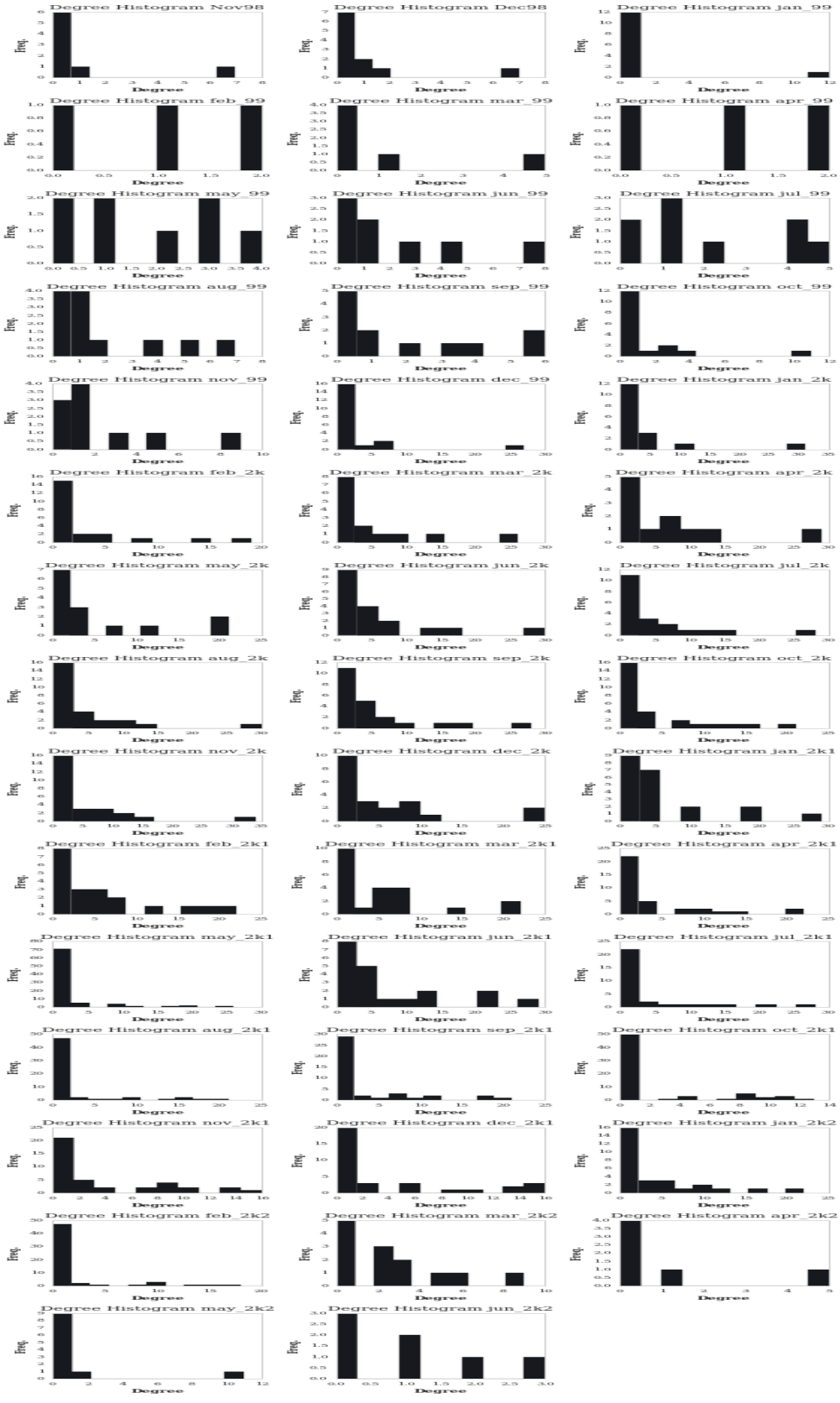


Figure 4.13: Monthly Degree Histogram

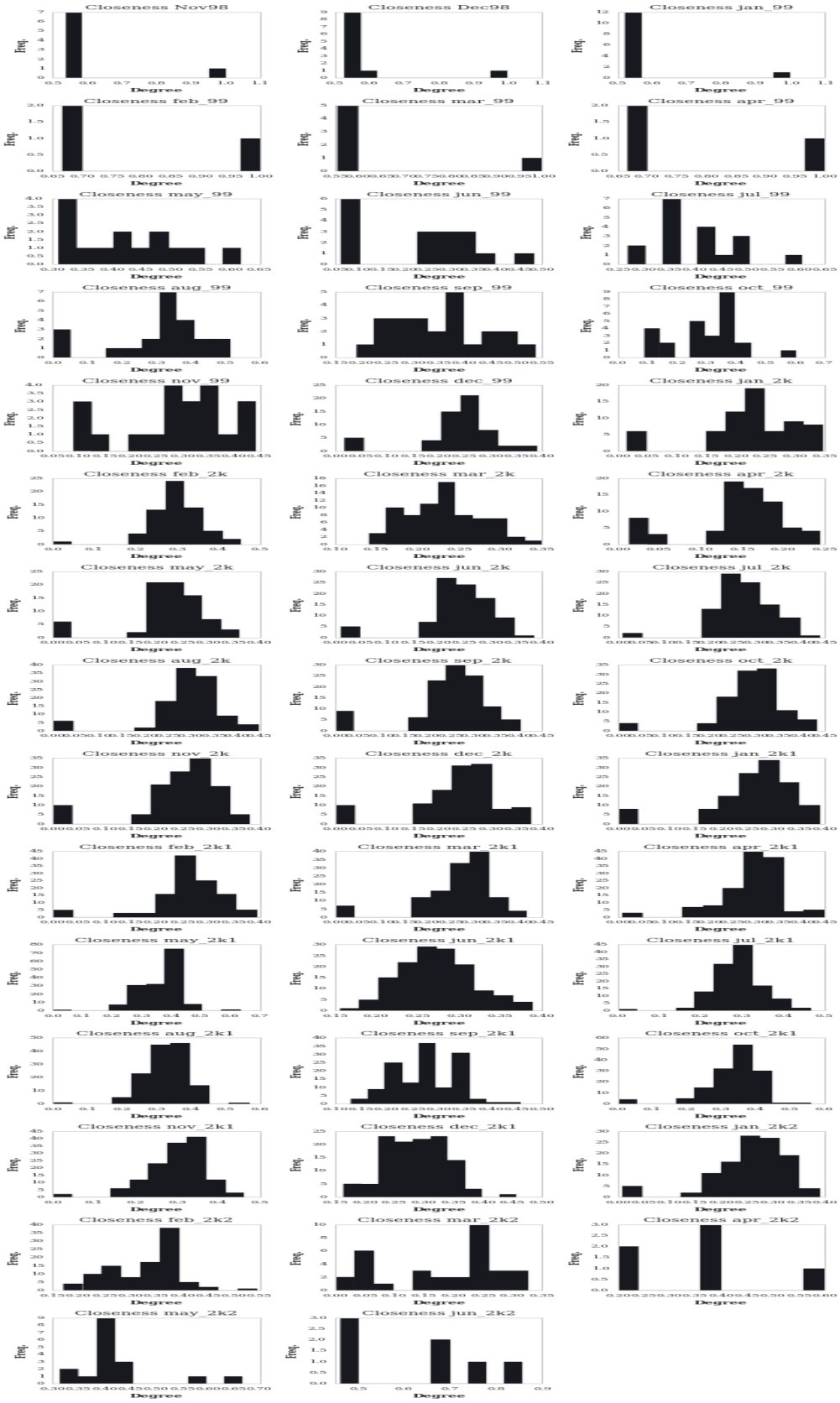


Figure 4.14: Monthly Closeness Centrality Histogram. The Closeness Centrality has a much more normal like distribution while the other measures have a much more skewed distribution.

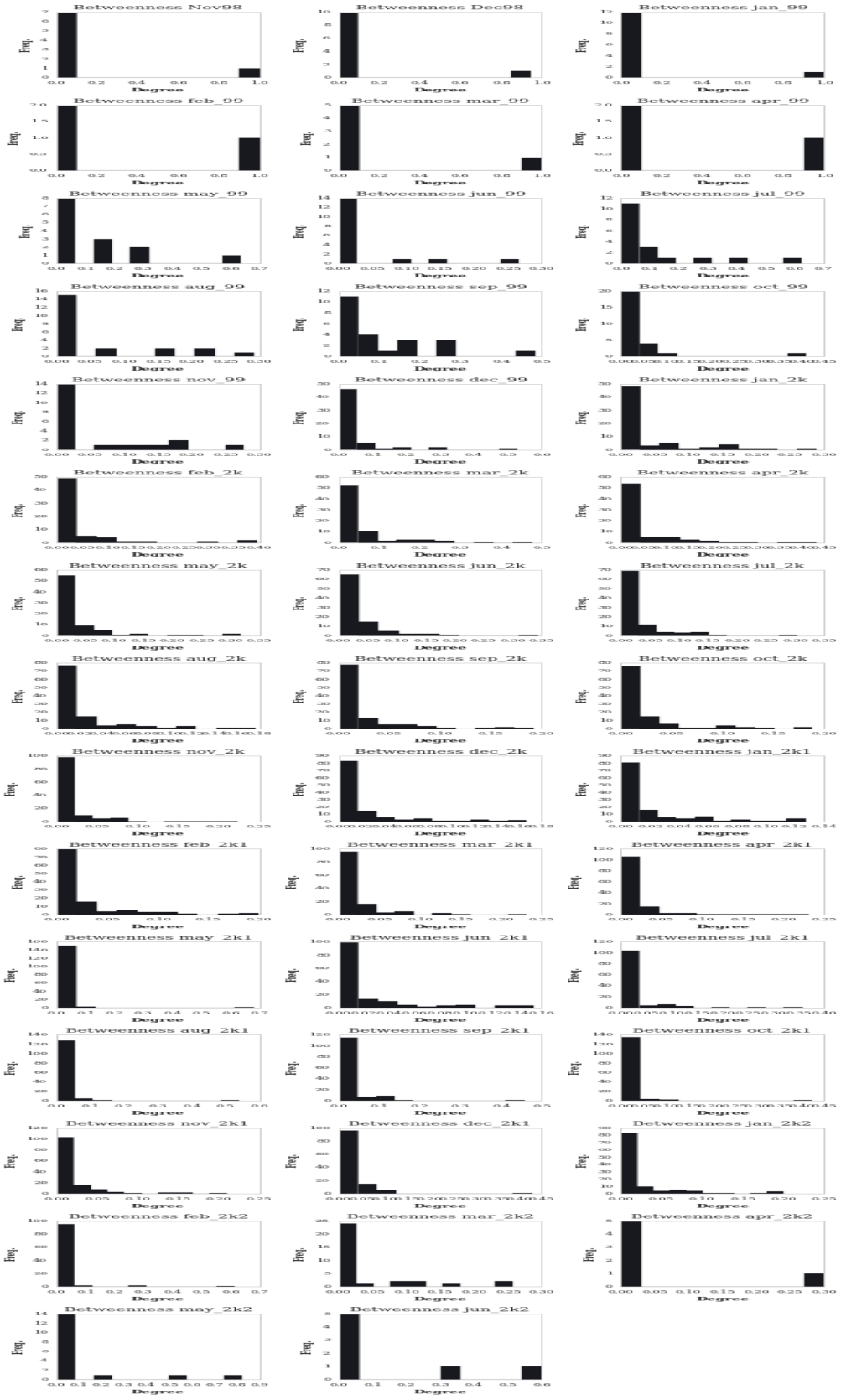


Figure 4.15: Monthly Betweenness Histogram

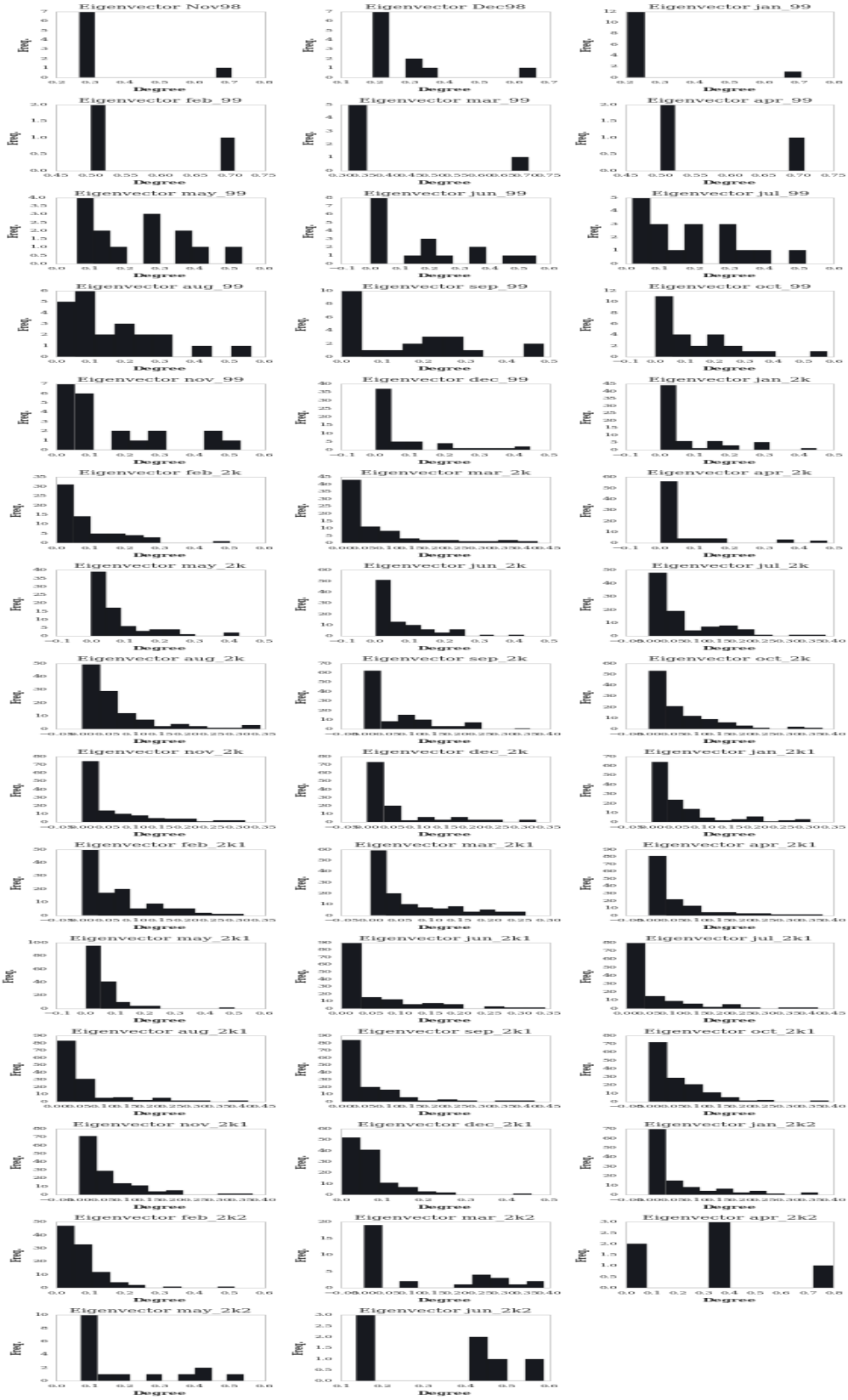


Figure 4.16: Monthly Eigenvector Centrality Histogram

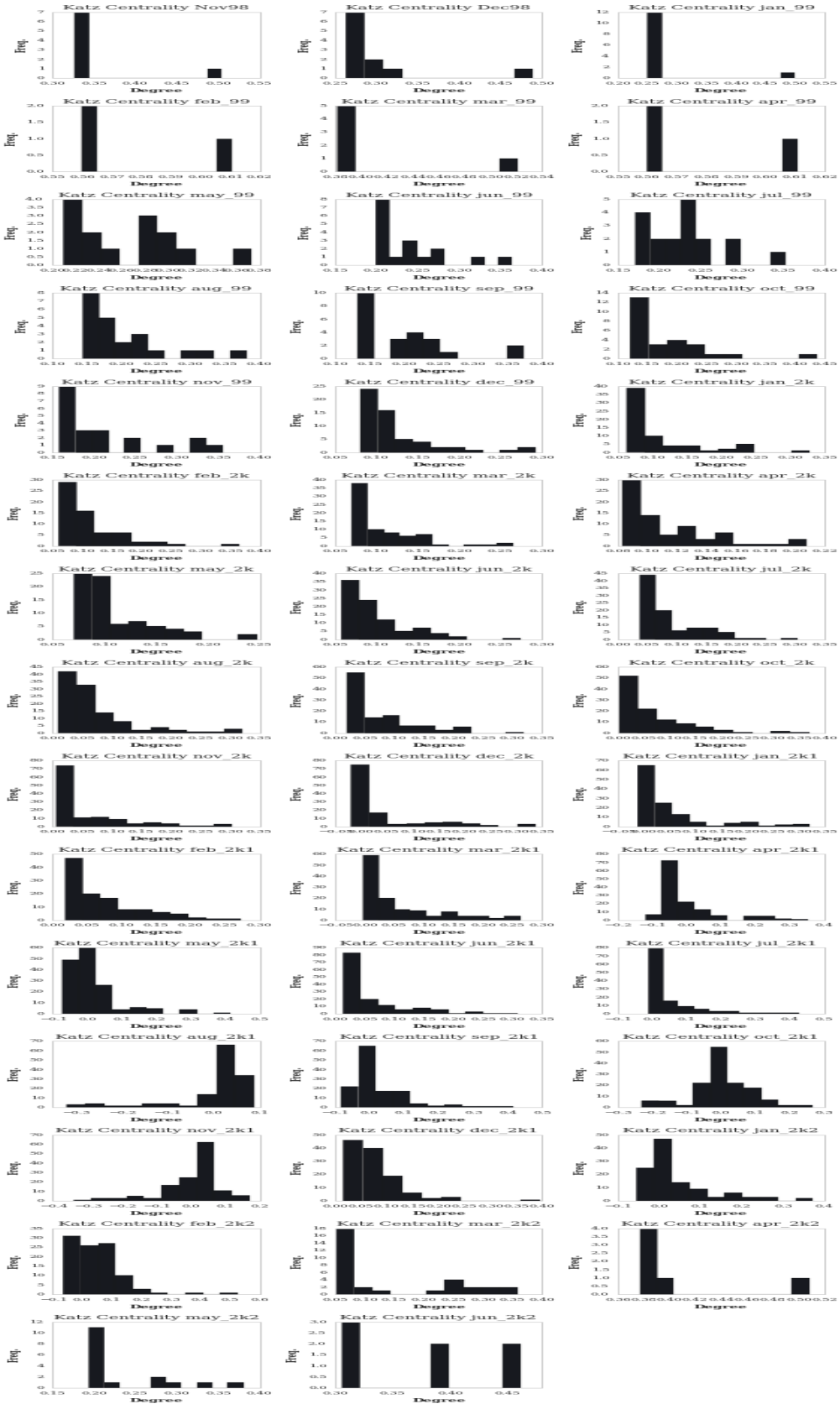


Figure 4.17: Monthly Katz Centrality Histogram

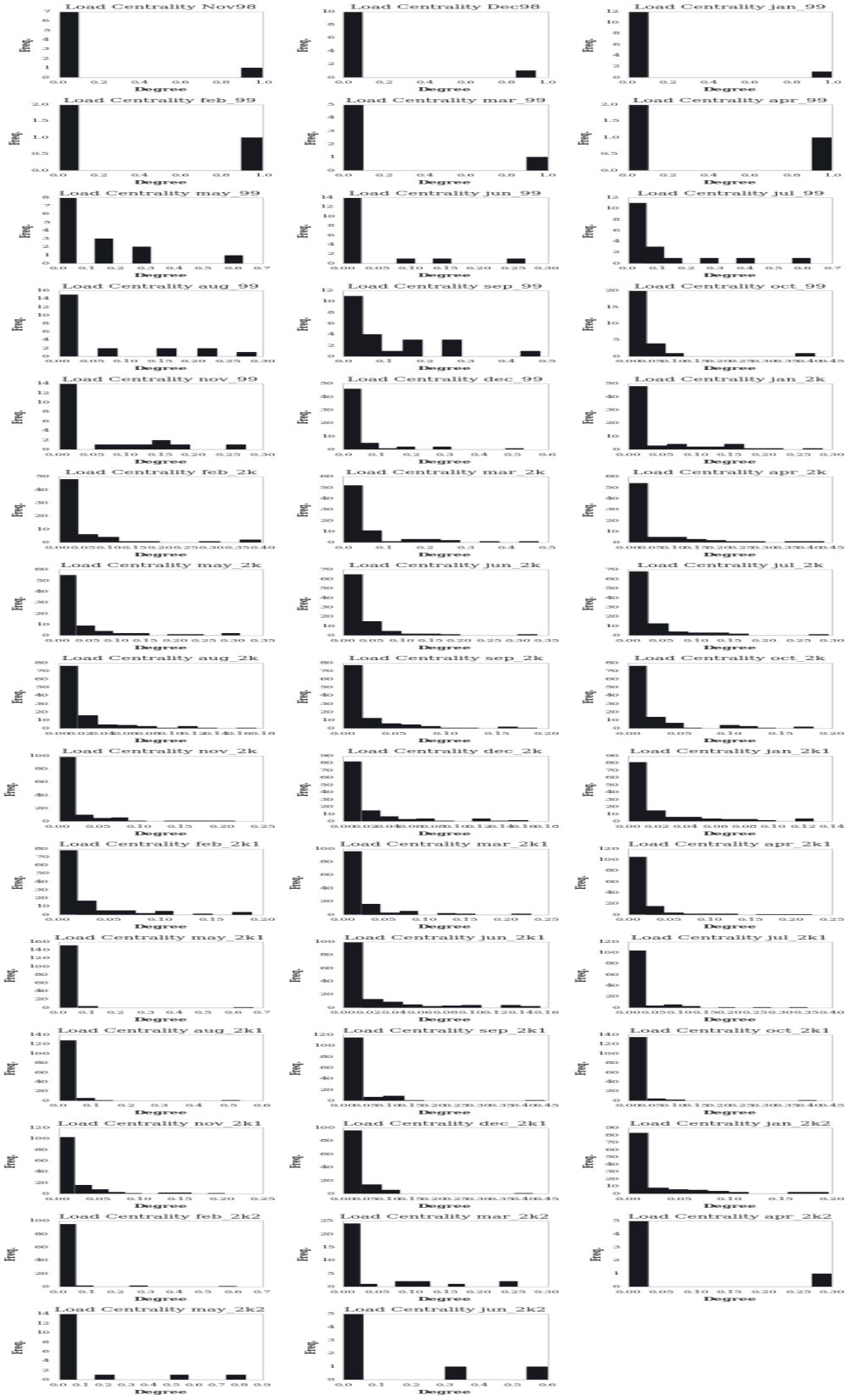


Figure 4.18: Monthly Load Centrality Histogram

4.3 Benchmark Measures

A key component of the analysis of the novel measures proposed in this study was to compare their behaviour to a set of benchmark measures. These measures are shown for the yearly and monthly networks in Section 4.3. From the yearly level plot Figure 4.19 we see that the change point in the network is the year 1999 and 2001. The network seems to be expanding from 1999 by all measures and contracting from 2001 onwards. From the monthly plot Figure 4.20 we see an interesting structure emerge. The graph time series seems to be characterised by two large peaks for the Feb - Apr 1999 period and two slightly smaller peaks between Apr 2002 and June 2002. In addition Feb 2000 appears to be interesting but only the Average Clustering Coefficient highlights this period with any conviction out of the benchmark metrics. Therefore, we expect our new measures at a minimum to replicate these features and highlight any additional ones.

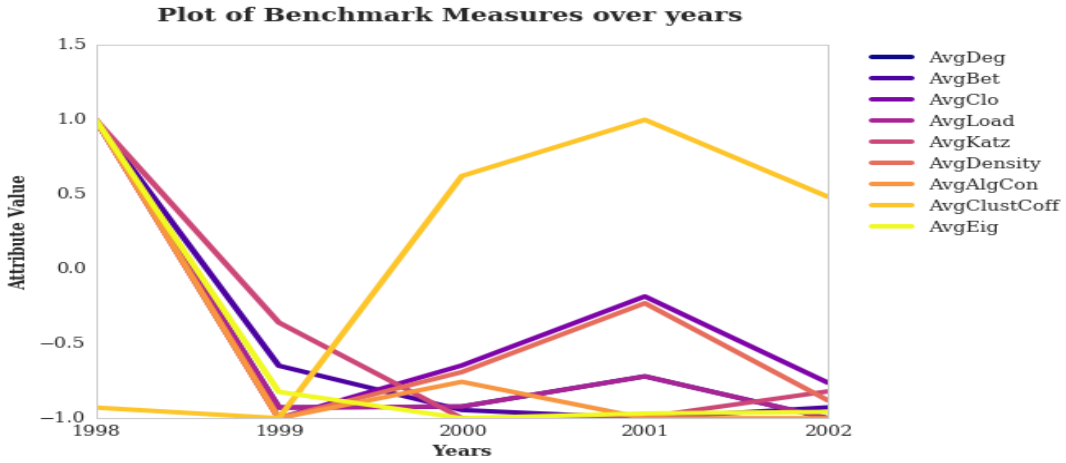


Figure 4.19: Plot of Benchmark Measures over Years

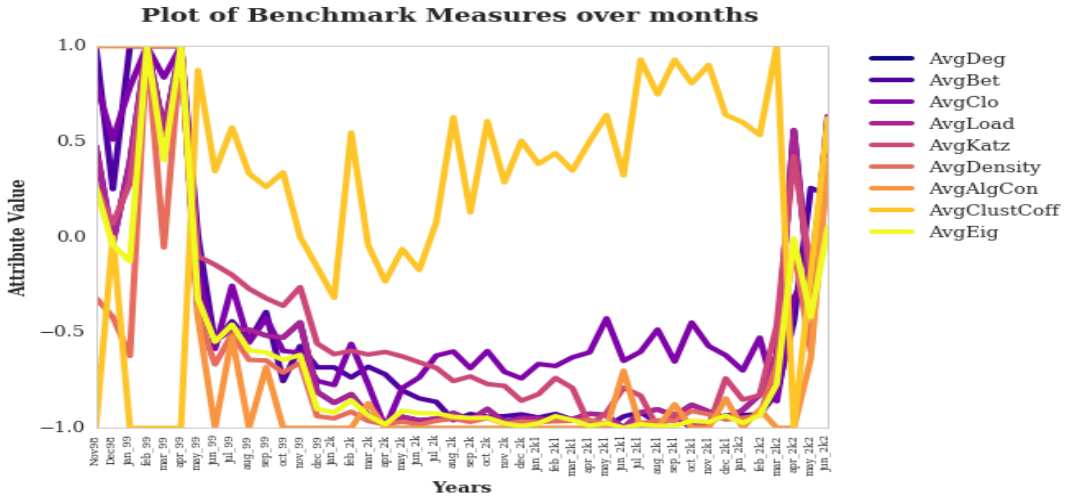


Figure 4.20: Plot of Benchmark Measures over Months

4.4 Attribute Analysis

In the figures in Section 4.4 we highlight the steps undertaken to decide which graph matrix to use for the derivation of the attributes. From the three matrices: Normalised Laplacian, Modularity and Adjacency we chose the Normalised Laplacian. The reasoning for this is highlighted in Figures 4.21 to 4.24. The first step was to look at the Signal to Noise ratio of all the proposed attributes calculated from the different matrices. The reasoning being that high SNR means better attributes. By this account it would appear that the Modularity and Adjacency matrices would be better choices because they have higher number of attributes with high SNR compared to the Laplacian. However, upon closer inspection of both the high and low SNR attributes we found this to be a bit misleading. The attributes which have low SNR from the Modularity and Adjacency matrices fail to recover the signal we established. But the attributes with high SNR to do this well. However for the Laplacian attributes the low SNR attributes are able to recover the signal that the same attributes from the other matrices are not. This can be seen in detail from Appendix E. So we decided to refine our analysis and consider another information theoretic measure, Entropy. From the Entropy plots we see that all three matrices seem to be similar with respect to their average Entropy but the Laplacian has lower average Entropy meaning lower uncertainty compared to the other two matrices. This combined with the observation that the attributes from the Laplacian are better at recovering the signal observed in the benchmark measures. We us the Laplacian attributes in all further results that are presented.

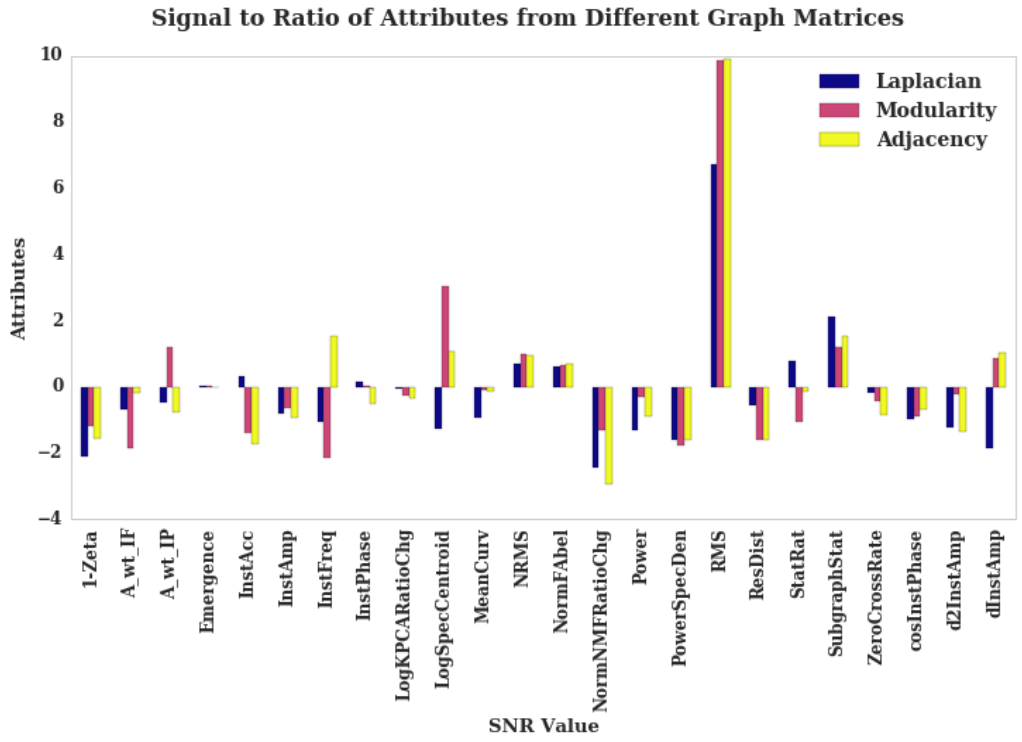


Figure 4.21: Plot of Signal to Noise Ratio of Attributes calculated from 3 different Graph Matrices

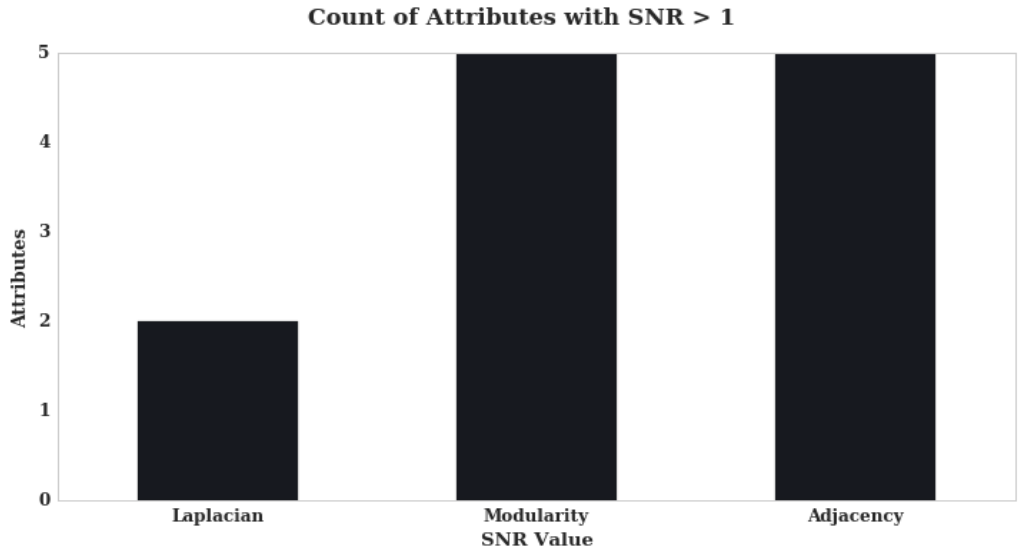


Figure 4.22: Plot of Count of Number attributes with $SNR > 1$ from the different graph matrices

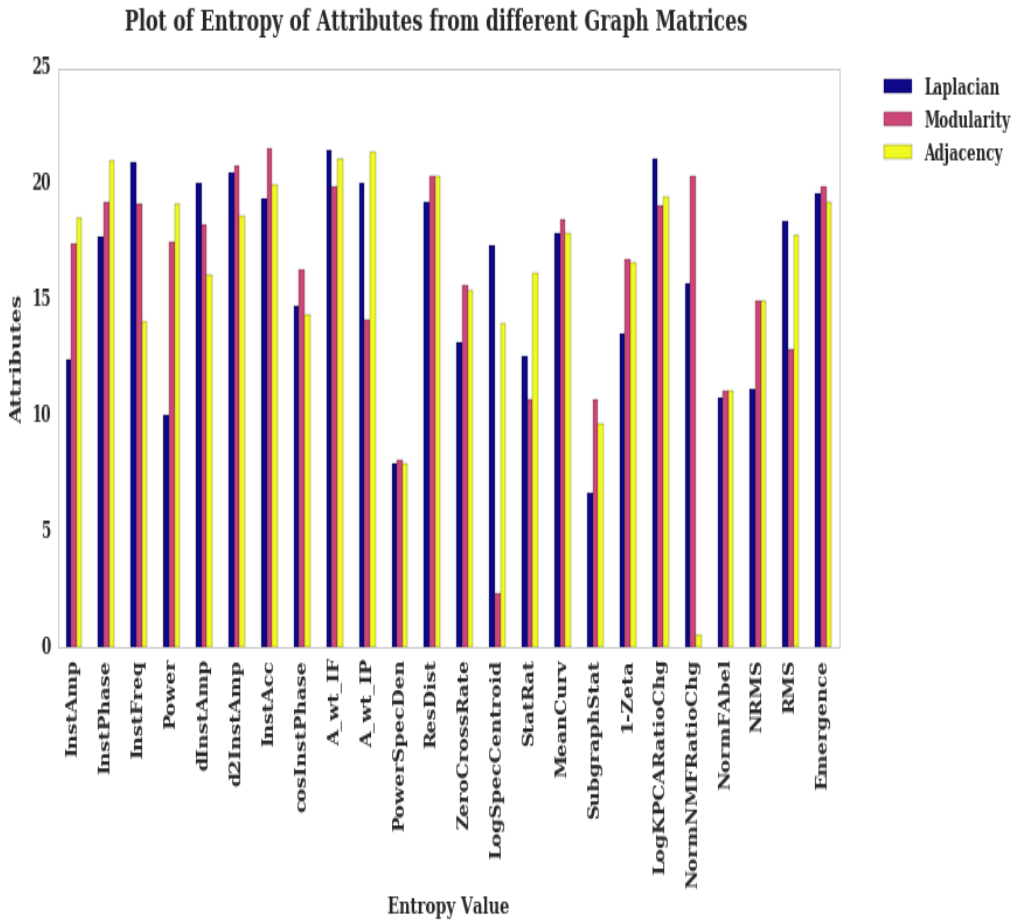


Figure 4.23: Plot of Entropy of attributes from the different graph matrices

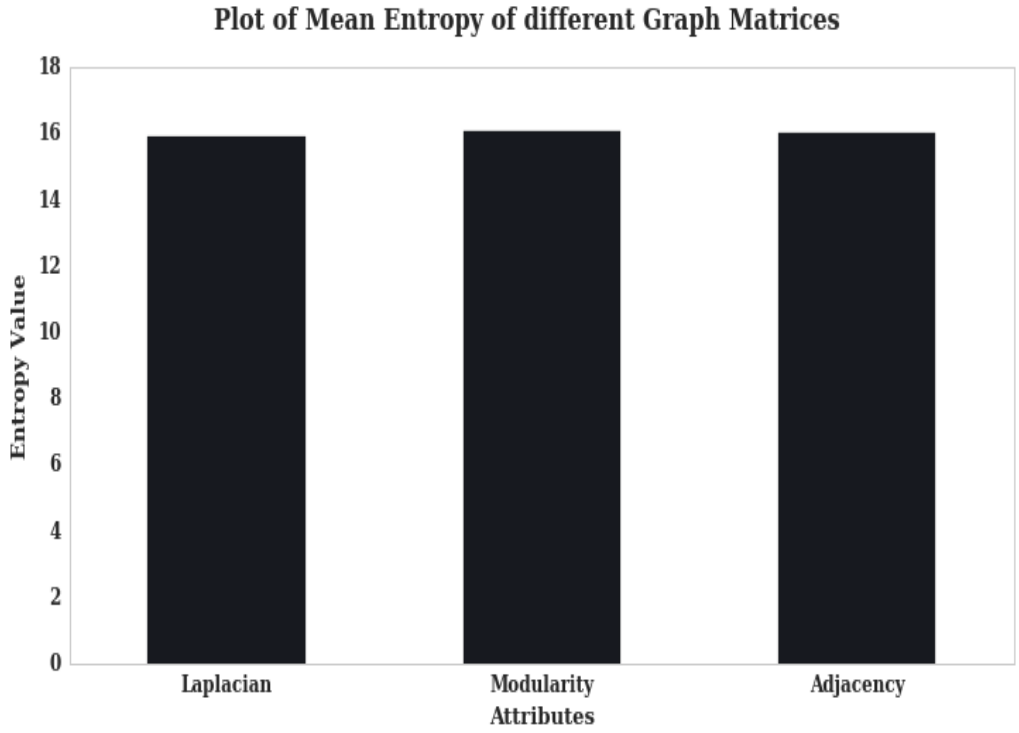


Figure 4.24: Plot of Mean Entropy of different graph matrices

4.4.1 Complex Attributes

The Complex attributes are the first of the Seismic Attribute that we present. From Figures 4.25 and 4.26 we see that the prominent peaks are retrieved by all the measures clearly with the Amplitude and the Cosine of Instantaneous Phase highlighting additional areas hinted at by the Average Clustering Coefficient. For the yearly the the 1999 change point is very clear among all the measures and suggests the network is contracting from 2001 onwards. Both sets of measures agree that the most significant time period is 2001.

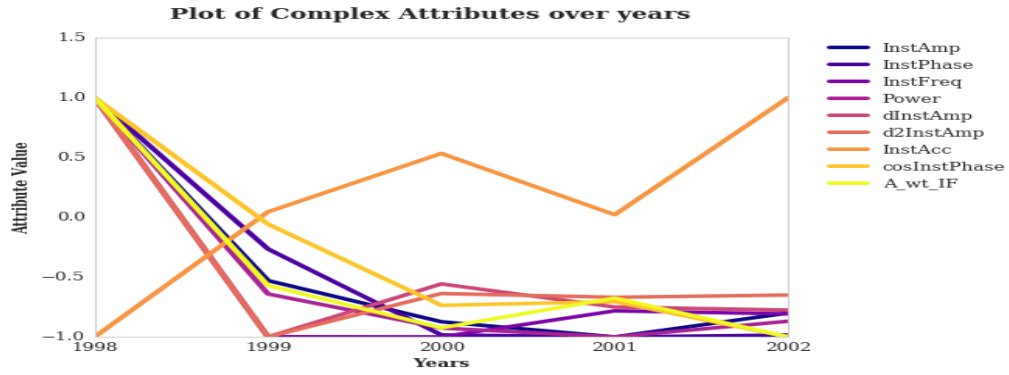


Figure 4.25: Plot of Complex Attributes over Years

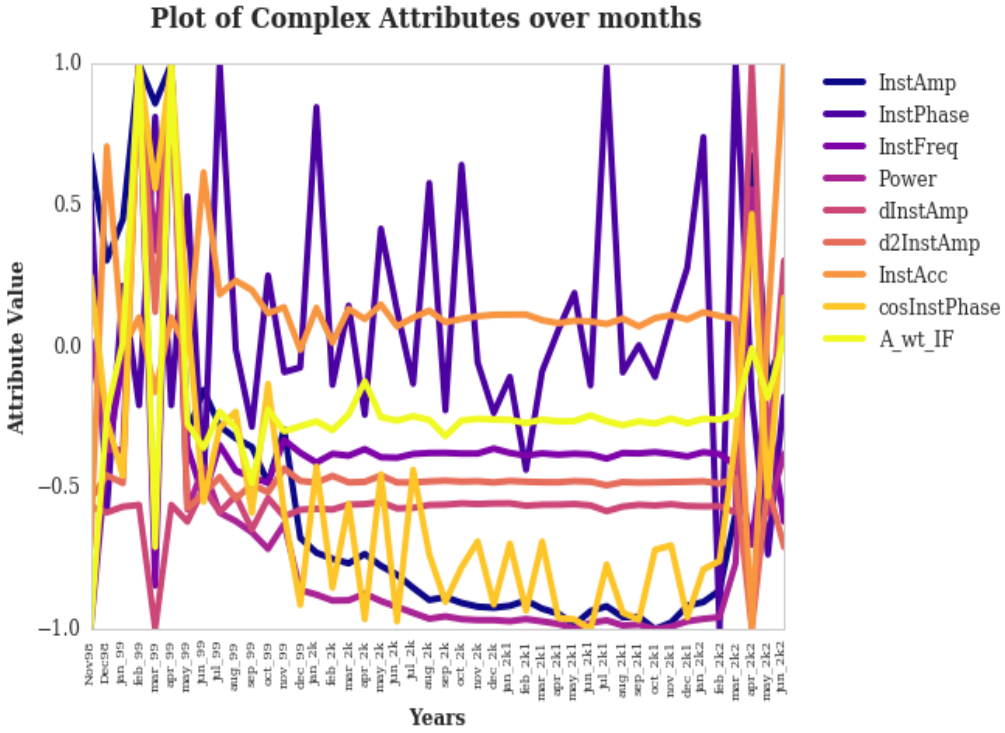


Figure 4.26: Plot of Complex Attributes over Months

4.4.2 Matrix Attributes

The matrix attributes shown in Figures 4.27 and 4.28 are also interesting in the sense that they also retrieve the key structures but are more smooth in general compared to the complex trace attributes. For the matrix attributes at the year level we see a large number of them converge to a peak in 2001 while a few converge to a trough. This is indicating the same phenomenon although with the opposite polarity. The monthly plot highlights the strong peaks as with the other attributes but in addition Nov 2000 and Oct 2001 are indicated to periods of strong change. Although the Average Clustering Coefficient indicates these periods it does not mark these periods as strongly as the matrix attributes but finds them to be roughly similar. So here we have a means of gaining additional insight into some of the structure suggested by the benchmark measures. These periods also do not show particularly strongly among the centrality measures.

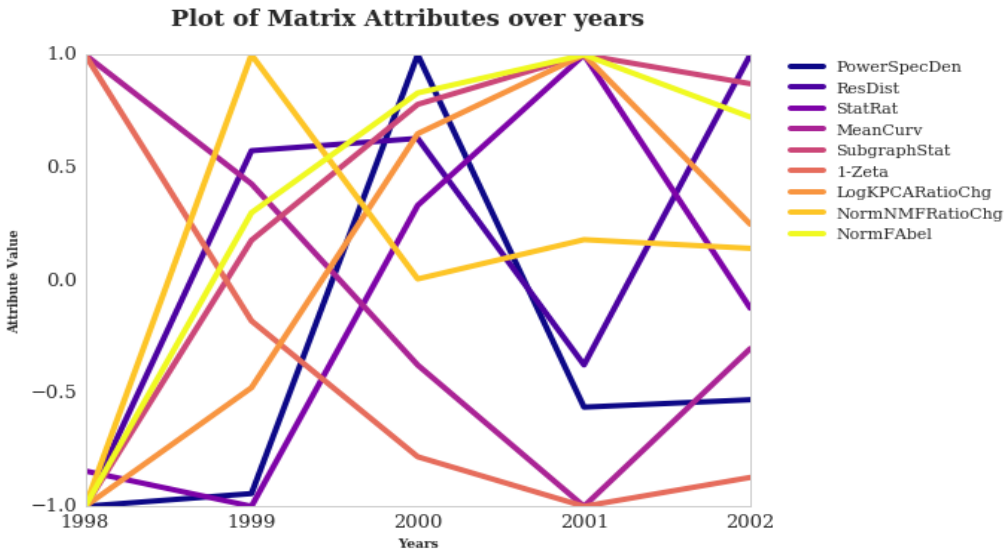


Figure 4.27: Plot of Matrix Attributes over Months

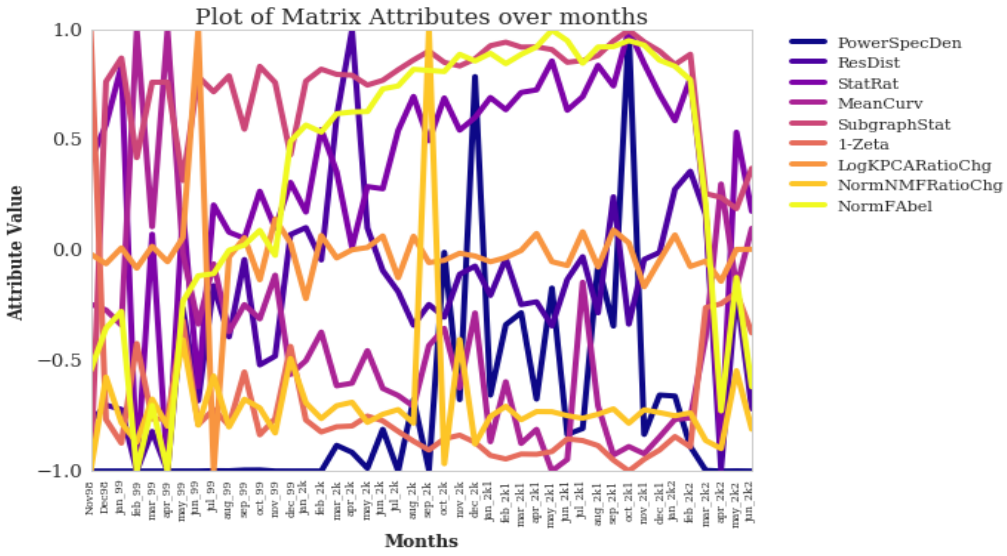


Figure 4.28: Plot of Matrix Attributes over Months

4.4.3 Music Attributes

The music attributes shown in Figures 4.29 and 4.30 are also able to retrieve the structure observed in the benchmark measures convincingly. From both the Zero Crossing Rate and the Spectral Centroid attribute we see they agree on the yearly peak being the year 2001 and declining thereafter. Interestingly the Zero Crossing Rate finds the peaks in the early and later part of the monthly time series but the peaks are smaller indicating that since the networks are smaller at these steps there are a a limited number of crossings available compared to when the network is larger and the signal is longer. Hence this makes sense in the context of the peaks later in the time series where the networks are denser. The Spectral Centroid shows a familiar structure while finding the additional peaks we have identified from the benchmark, complex and matrix attributes.

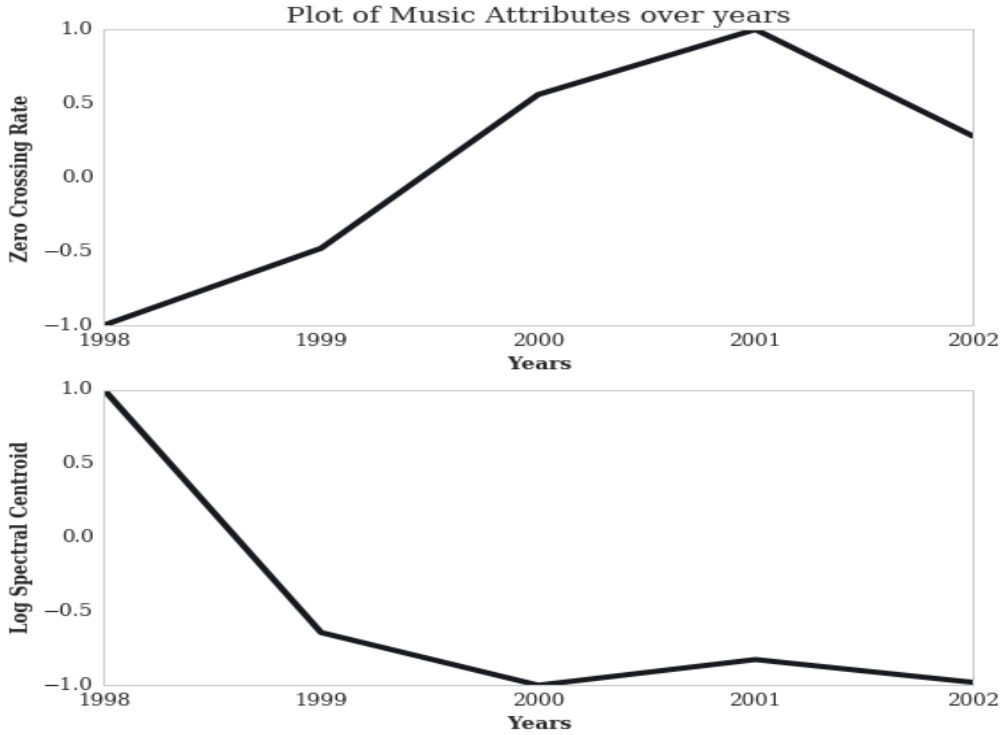


Figure 4.29: Plot of Music Attributes over Years

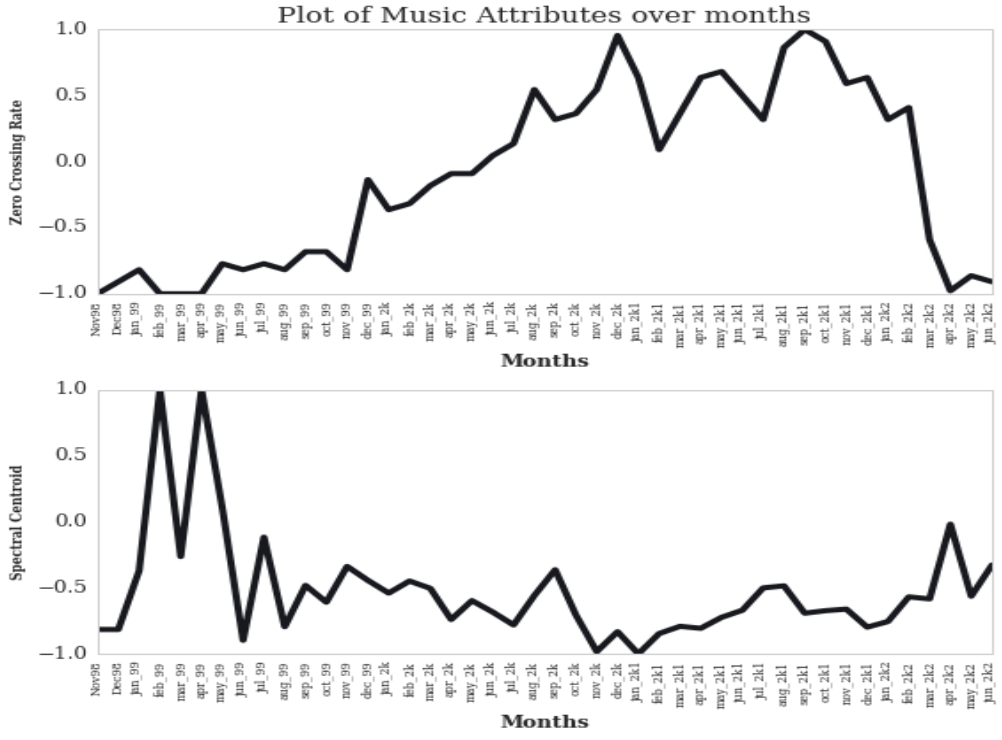


Figure 4.30: Plot of Music Attributes over Months

4.4.4 Average Attributes

Of the remaining attributes shown in Figures 4.31 and 4.32 we some interesting behaviour for example the KLPCA Ratio is used to detect unconformities. Unconformities are breaks in the depositional surface separating the rock masses into two different ages. This indicates a break in the sedimentation sequence. In our case we can think of the unconformity as being a large break. Therefore it is interesting to see that the change points in 1999 are highlighted very well by this attribute.

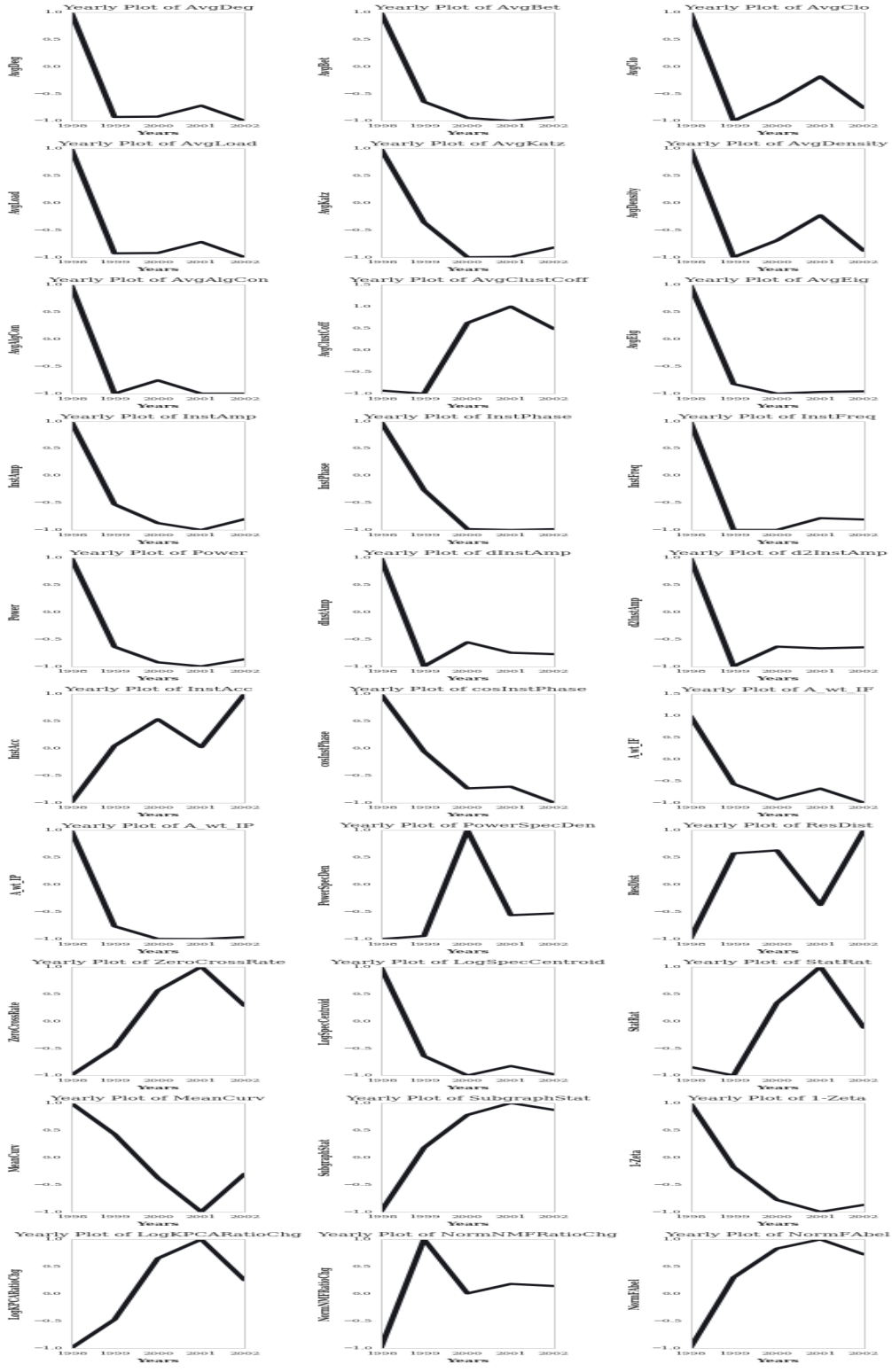


Figure 4.31: Plot of Average Attributes over Years

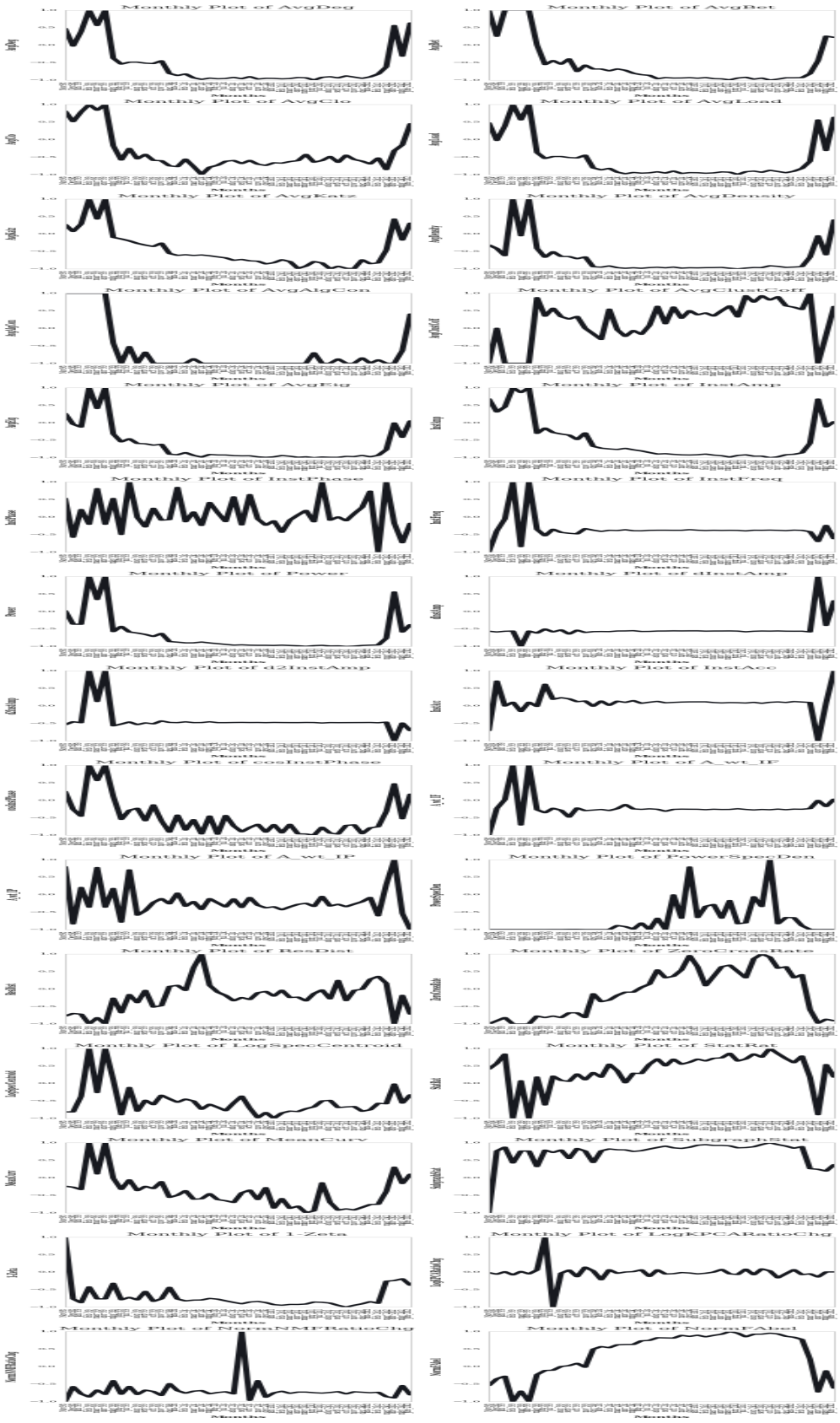


Figure 4.32: Plot of Average Attributes over Months

The Phase attributes such as Instantaneous Phase, Cosine of Instantaneous Phase and Amplitude Weighted Instantaneous Phase have similar behaviour in the sense

that they are sensitive to other changes apart from the instances when the network is small such as in early 1998. We see that the Phase attribute suppresses the two peaks when the network is really small but highlights other changes along the time series. However, this attribute is rather noisy so typically in a seismic attribute analysis the Cosine of Phase is used as it is better constrained and the polarity can be controlled. In this study we choose to look at peaks but one could easily flip the sign and look at troughs. This attribute is less noisy and is sensitive to the big and small changes in the network. The Amplitude weighted Phase attribute also highlights the bigger changes while suppressing the smaller changes. This is hardly surprising as the Amplitude attribute finds the amplitude anomalies in the network which are at the beginning and end of the series. The first and second derivatives of the Amplitude are smoother versions of the Amplitude and find the greatest change in the Amplitude while suppressing smaller changes.

The Frequency and Frequency derived attributes is smooth attribute that also highlights the biggest change periods in the network. The derivative of the Frequency Instantaneous Acceleration however is more interesting as it is able to provide additional granularity to the periods that the Frequency attribute highlights as having the greatest change. For example in the period from 1998 to early 1999 we see the two peaks in the Frequency but the Acceleration highlights within this time frame that Dec 98 and May 99 are particularly significant. The May 99 period is hinted at by the Frequency but its magnitude compared to the neighbouring periods is lower. The Amplitude weighted Frequency shows a similar propensity of being more sensitive to big change points.

The Stationarity measures such as the Subgraph Stationarity, Stationarity Ratio, 1-Zeta show the change points along the time series but capture slightly different dynamics. The 1- Zeta which gives a sense of how the members have changed between time periods rightly highlights the periods of expansion early and contraction later in the series. Where the network is fairly dense the attribute is smooth. The Subgraph stationarity which utilises the union and the intersection of adjacent graph matrices also captures this dynamics that the graph dissimilarity shows as a trough, while similarity shows as a fairly flat line. The Stationarity is more sensitive in the sense that the contractions in the network show as large troughs while the expansion shows a largely positive trend.

The Power attributes such as the Power and Power Spectral Density are interesting that while the Power attributes behaves largely like a smoother Amplitude attribute with the large changes enhanced. The Power Spectral Density is less sensitive to these contraction changes but rather highlights structure in the dense parts of the network which the other attributes do not do as well. The Power Spectral Density and the Zero Crossing Rate are similar in this respect. Both these attributes find greater changes in denser part of the network.

The Norm of the Forward Abel Transform of the magnitude of the Fourier Transform (Norm FAbel) captures the trend established from all the metrics so far the best. This is because the early part of the network, when it is small show up as troughs which are smaller in magnitude that those at the denser part of the network

when it is expanding and also much smaller than the period in the end when the network is contracting again.

The Resistance Distance and Curvature measures also highlight the periods of greatest change and more sensitive to large changes in both the dense and sparse parts of the network. For example the Mean Curvature best highlights the Aug - Sep 2001 change period most clearly out of all the attributes. The Resistance Distance likewise highlights the Mar - Apr 2000 period as being significant more clearly than most other attributes.

4.5 Correlation Analysis

The correlation analysis is performed using the monthly level attributes because the yearly data has a very small sample size. Also the correlation measure used is the Pearson Correlation.

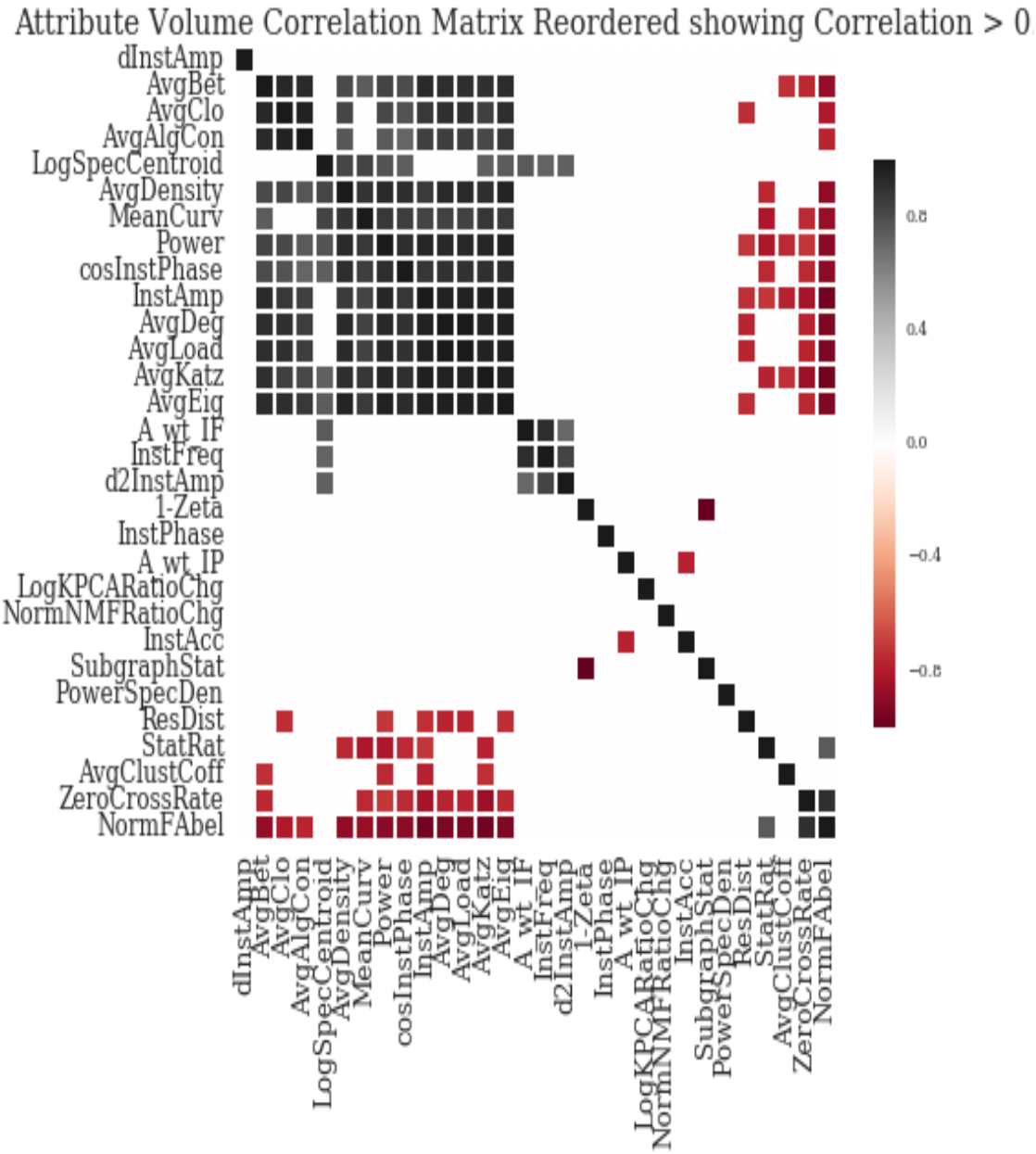


Figure 4.33: Reordered Correlation Heatmap showing correlation > 0.7

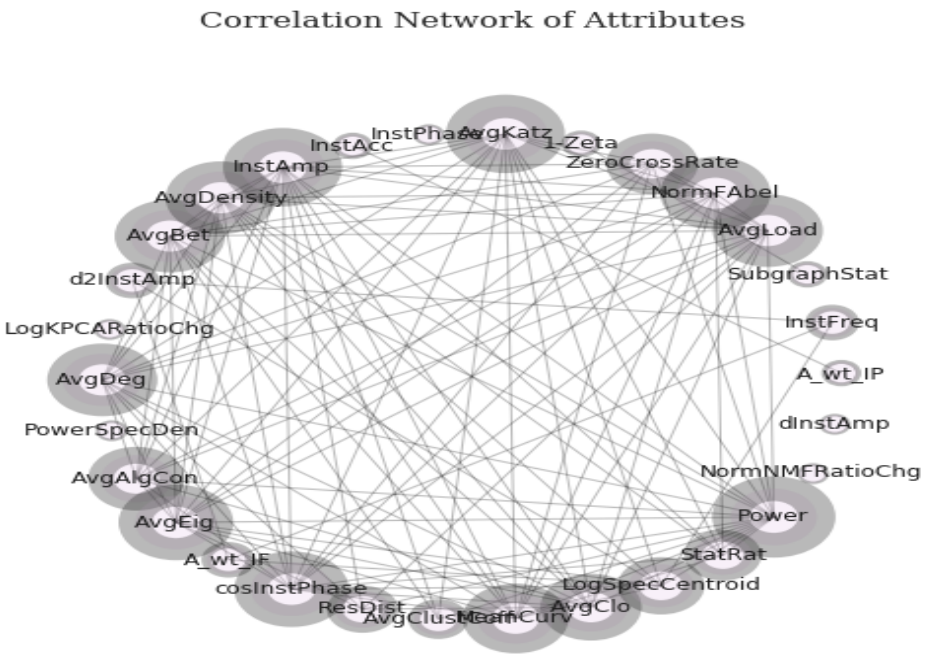


Figure 4.34: Correlation Network of Attributes. The thickness of the borders of the nodes indicates high degree while low thickness of borders around the nodes indicate low degree.

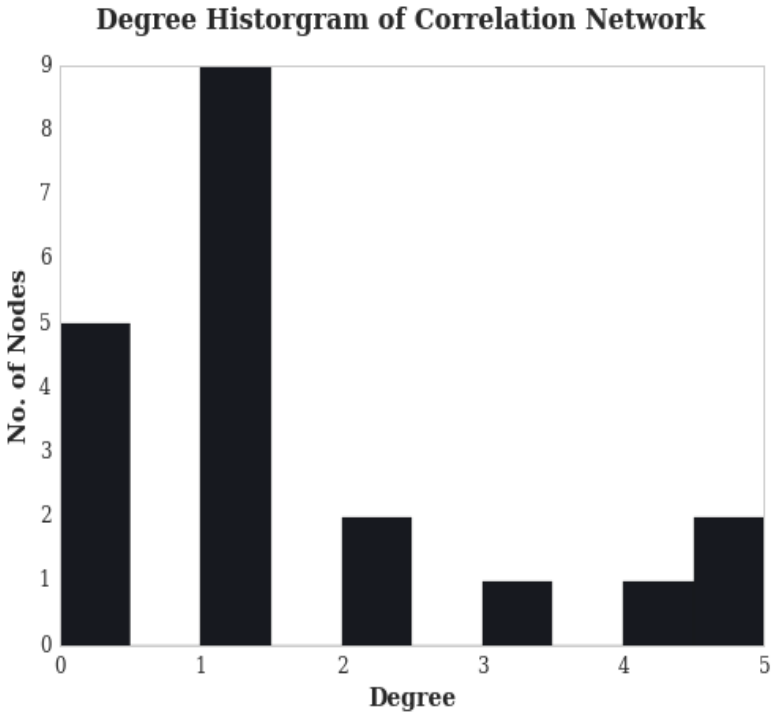


Figure 4.35: Correlation Network Degree Histogram

The figures in Section 4.5 shows the Pearson Correlation among the different attributes. The correlation matrix shown in Figure 4.33 shows the correlations above

0.7 and everything else has been set to zero. This allows us to see to the highly correlated items only. Also a hierarchical clustering is used to groupd the variables and then these indices are used to sort the rows and columns of this matrix so we are able to derive a reordered correlation matrix showing the high and low correlation items in clear clusters. As an extension to this analysis we take this matrix and derive a correlation network of attributes. Here the degree of a node is determined by the number of other nodes that it is highly correlated with. So in Figure 4.34 we see the network with the Degree encoded as the thickness of the border of the nodes. From Figure 4.35 we see that 9 attributes have at least 1 other attribute that they are correlated with 5 attributes being completely disconnected such as the Power Spectral Density, Log KLPCA Ratio Change, NMF Ratio Change and Derivative of Instantaneous Amplitude. A small number of attributes have very high Degree such as Power. We could say that a lot of the metrics capture similar information to the centrality measures even though the way they are calculated are very different.

4.6 Feature Ranking with Regression Analysis

The Regression method used for the results in Section 4.6 are from a Gradient Boosting Regressor which is mainly used for feature ranking. The dataset size is limited since we are only working with 44 monthly networks with a total of 30 attributes. To get a good idea of feature importance and prevent over fitting we separate the data into 50/50 test/train chunks. The aim was to rank features by importance in predicting a fundamental property such as Average Degree. We see predict these fundamental properties for a graph time series with reasonable accuracy ($> 95pc$). From the feature ranking we see that the features that are not well correlated to other features are paticularly important in this predictive model. Also we noted earlier that the Abel Transform attribute captured the insight we gained from the network visualisations that the early part of the network was sparse compared to the later parts so it is encouraging to see that this attribute is capturing useful dynamics from a predictive context as well.

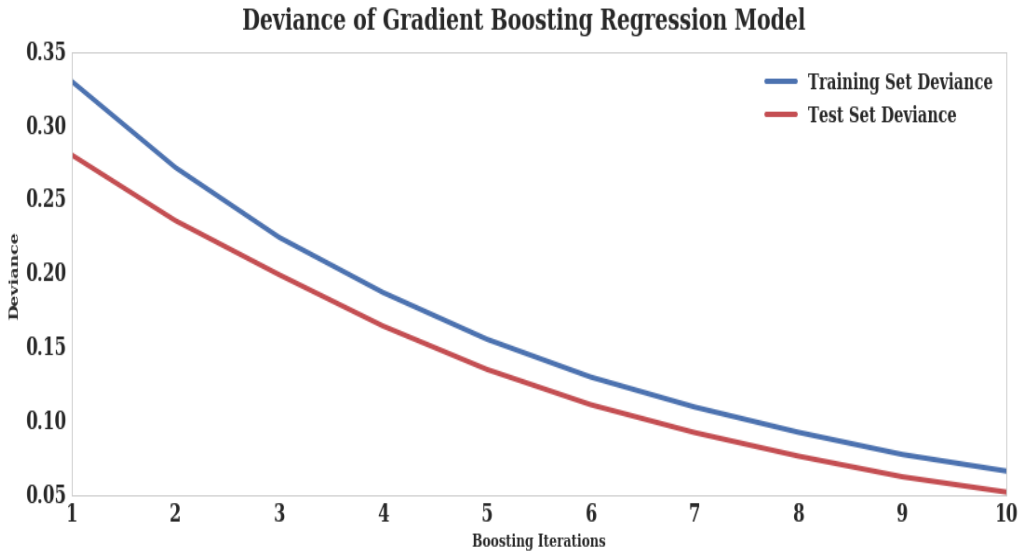


Figure 4.36: Regression Deviance Plot after 10 boosting iterations to train the mode. Here we predict the target based on the held out data.

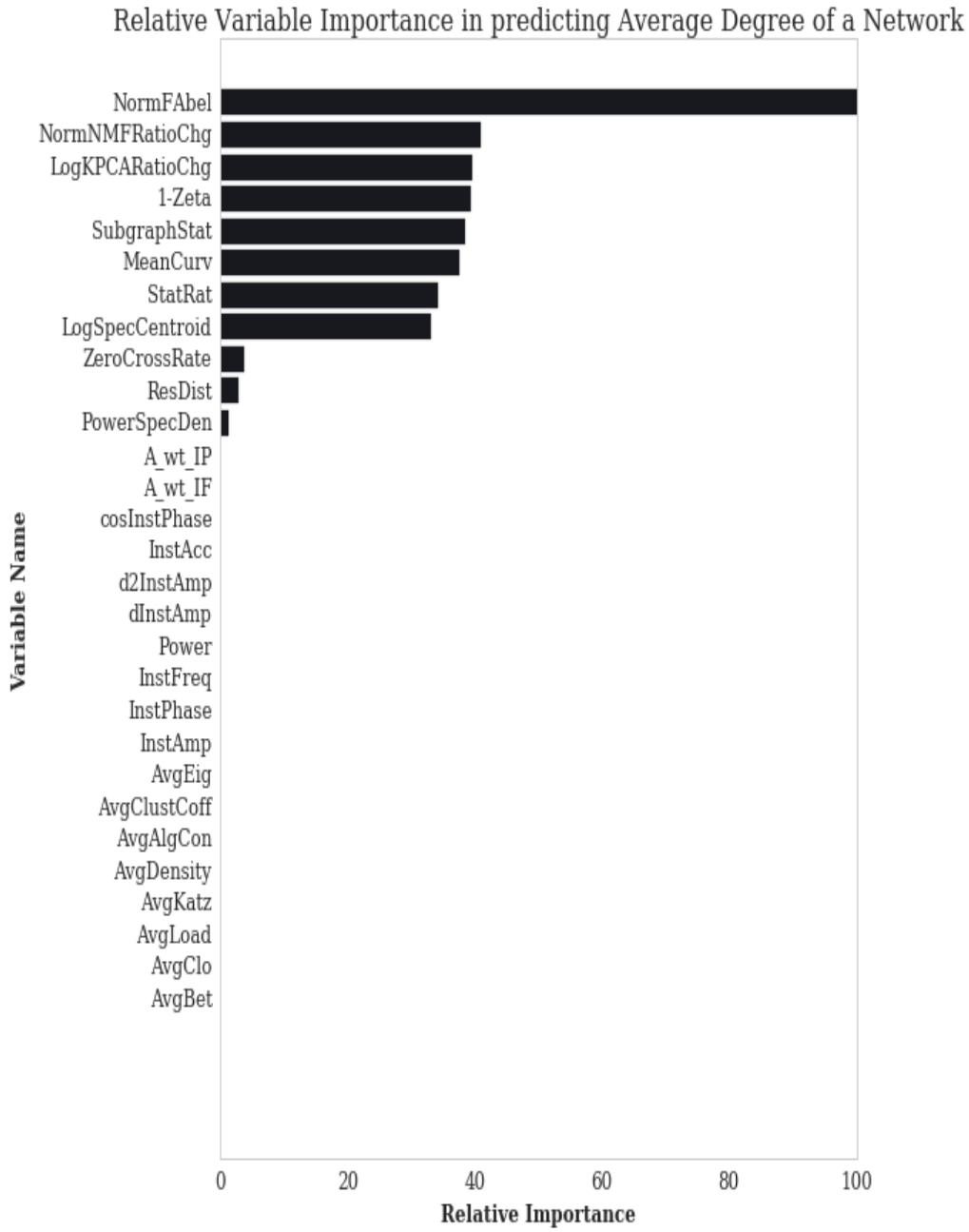


Figure 4.37: Feature Ranking by Gradient Boosting Regressor for predicting Average Degree of the Network at a future time step.

4.7 Aggregation Schemes

In Section 4.7 we show the aggregation measures proposed in this study the RMS and NRMS in addition to the Emergence measure proposed in literature. The Emergence measures as expected shows the bursts really clearly but the rest of the changes are fairly damped. The RMS is better in this respect that the bursts as well as the drop due to the sparsification towards the end of the time range. The period of densification is also clear from the positive trend. The NRMS is constrained in this case between [0, 0.4] so it is easier to interpret changes from zero. The NRMS measure segments the time series into periods of large and smaller changes. When the networks are or similar size the normalised difference is small compared to when the sizes are different. The box plot in Figure 4.39 shows the extent of the attributes. It is interesting to see that the Abel Transform attribute does not have any values beyond the whiskers of the plot while the first and second derivatives of Amplitude are particularly squashed with large values on both sides. This illustrates the necessity of scaling because otherwise the scale of some attributes would have dominated the Regression. Also another observation is that the Zero Crossing Rate is similar to the Abel attribute in the sense that it does have any values beyond the whiskers and is fairly centered around 0 but comparatively it does seem as important in the Regression context.

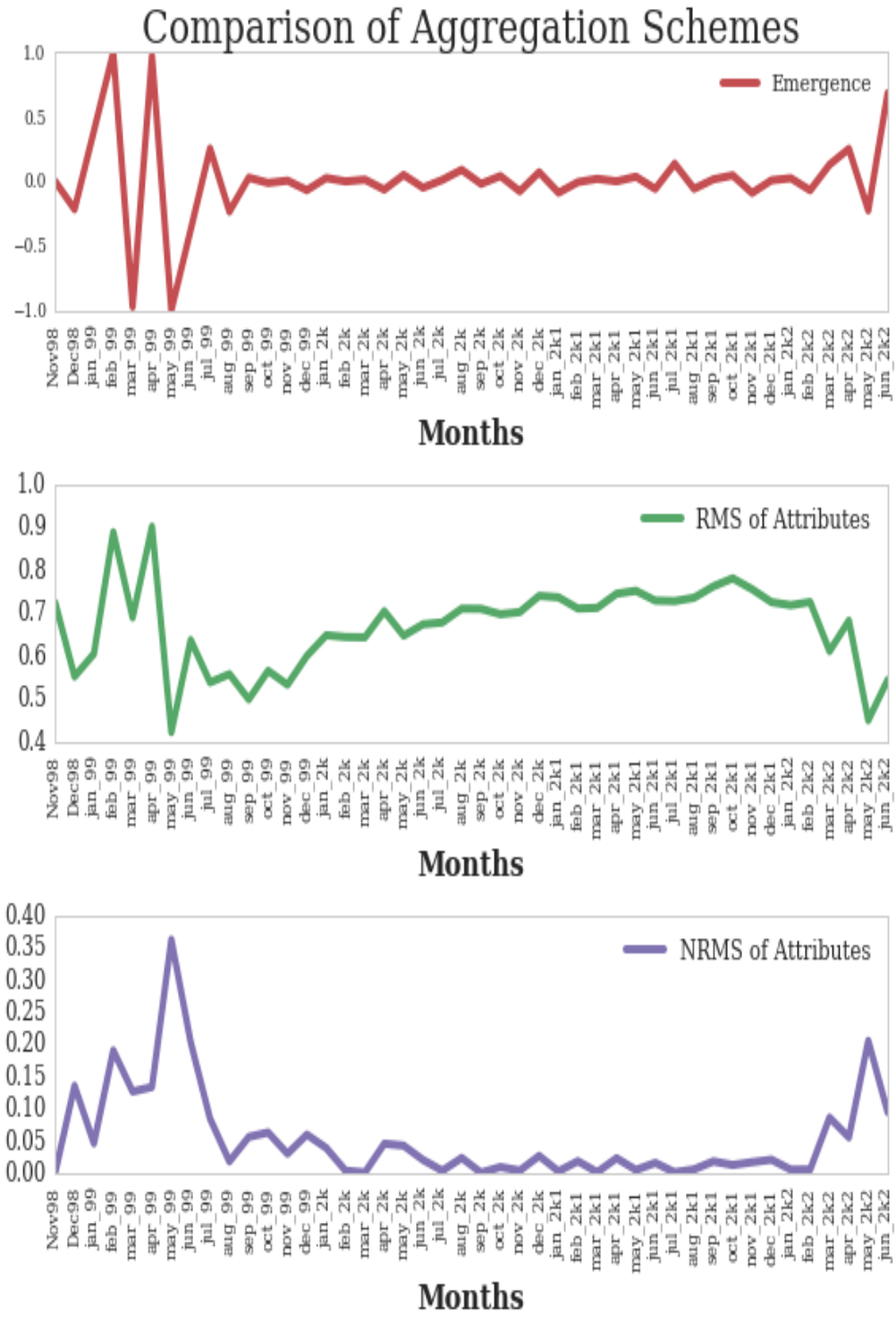


Figure 4.38: Comparison of different aggregation schemes: Emergence, RMS and NRMS.

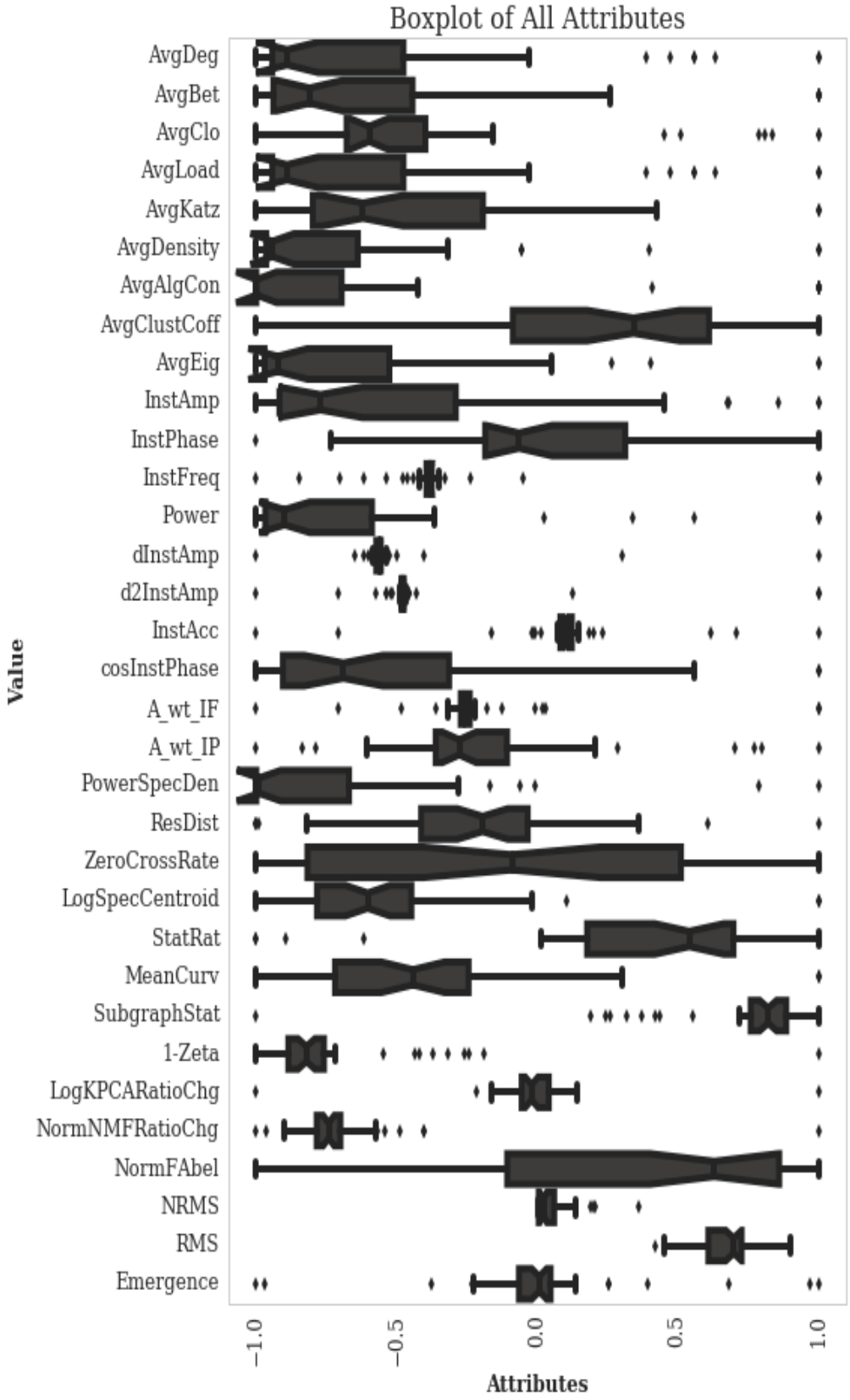


Figure 4.39: Boxplot of All Attributes

4.8 Manifold Visualisation of Attribute Volume

The Manifold Dimensionality Reduction plots using Multidimensional Scaling and TSNE highlighted in Section 4.8 illustrate that the perceived similarity is highly distance metric dependant. We see slightly different results from the TSNE for the different metrics. However, the key observation is that the Centrality and Complex Trace attributes appear near to each other and in this case the Correlation distance is not a good measure since from the Correlation analysis we have determined a high degree of correlation. But this serves as an additional way to visualise the correlations.

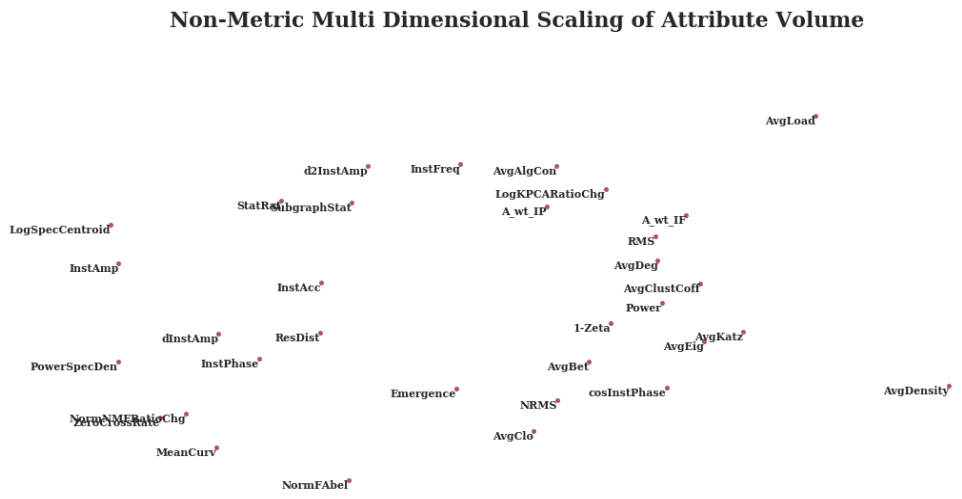


Figure 4.40: Non-Metric Multidimensional Scaling of Attribute Volume

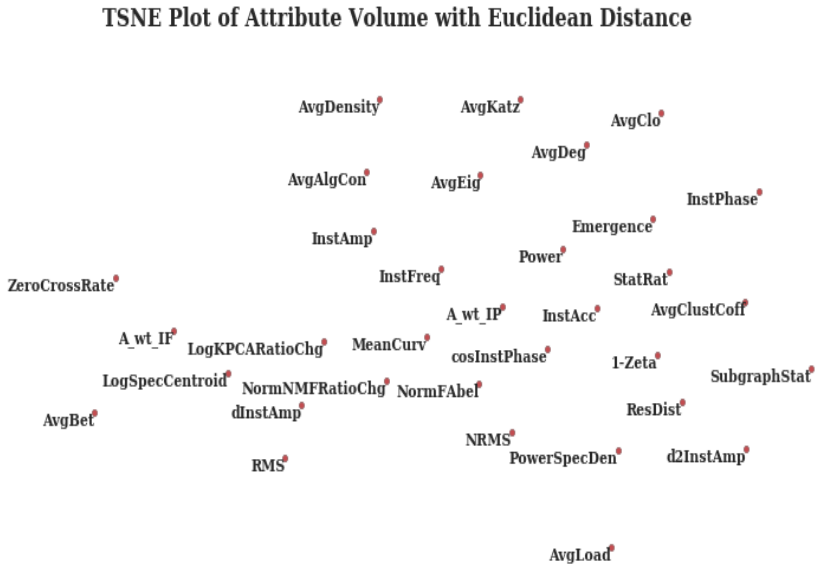


Figure 4.41: TSNE Plot of Attribute Volume with Euclidean Distance

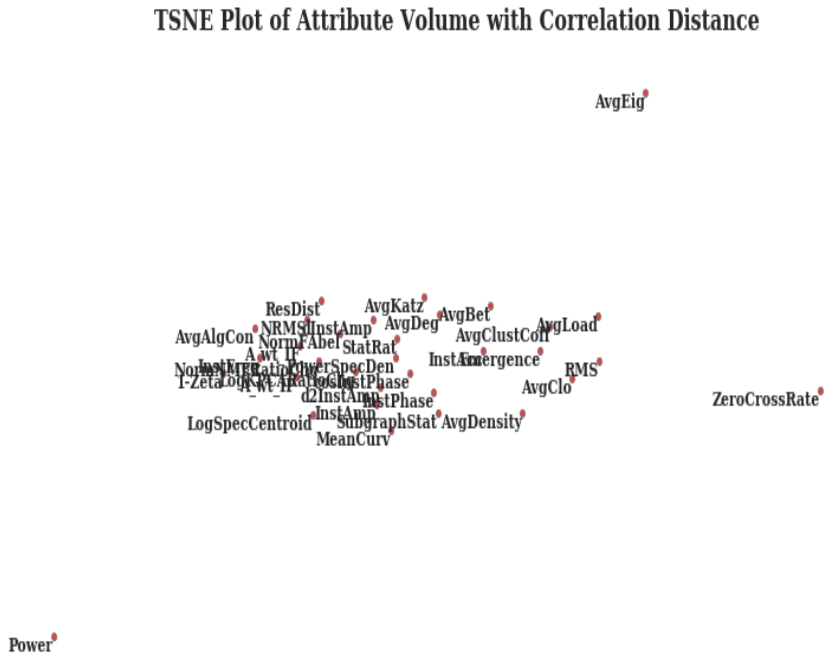


Figure 4.42: TSNE Plot of Attribute Volume with Correlation Distance

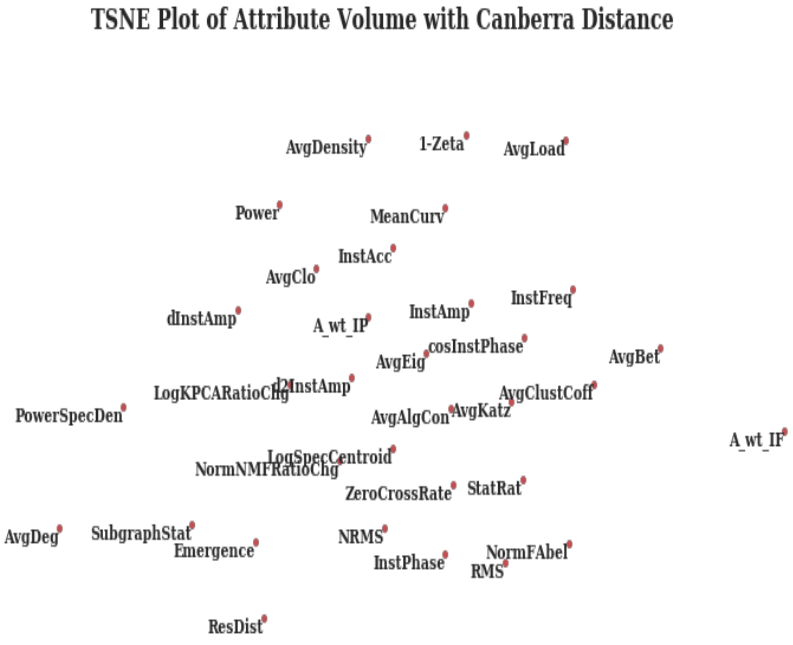


Figure 4.43: TSNE Plot of Attribute Volume with Canberra Distance

4.9 FK and Radon Plots

The FK and Radon Plots are common domains for data visualisation and processing. In a seismic context both plots allow for separation of signal and noise based on signal and velocity. In the FK plot we have transformed the whole volume into Frequency and wavenumber so a high dimensional volume can be visualised in 2 dimensions. The FK components can be visualised as slices through the volume or as the whole volume shown in Figure 4.44. It shows a central cone of the signal centered at F= 0 and we can see the outliers on either side. These correspond to the dense and sparse parts of the network. The Radon plot shown in Figure 4.45 maps the data to rotation angles so the two axes represent the angles. This is very useful because it shows clusters in the data. We can identify at least 4 different clusters where the first clusters consists of the networks that are extremely sparse, the second and third clusters where the networks are expanding but at different rates and the fourth when the network is contracting. The Log Panel is used to look at multiple attributes simultaneously in a way that allows for us to track peaks or troughs across multiple attributes over time. This plot can be set up in a number of ways for example the attributes could be sorted by cluster indices as was the case in the Correlation Heatmaps or in this case they are grouped by type. So the centrality, seismic, music and aggregation measures are grouped and shown. We observe the same trends that have been noted already but this visualisation is much easier to work with than the scatter matrix format.

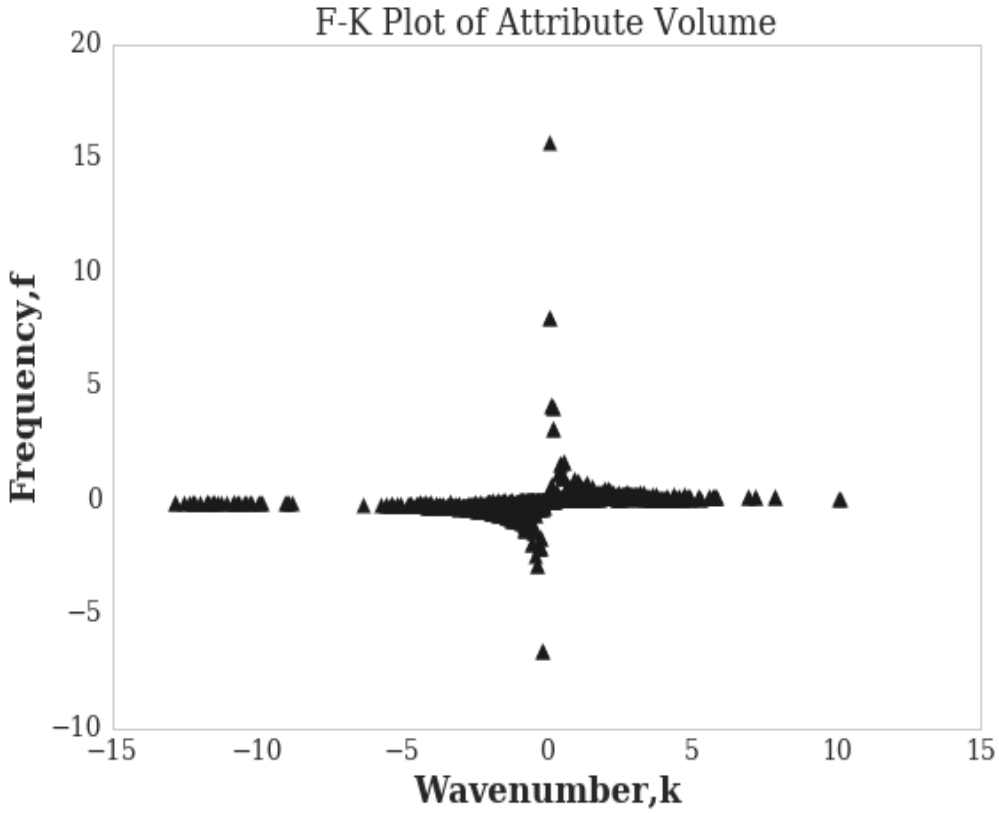


Figure 4.44: Frequency Wavenumber (FK) Plot of Attribute Volume

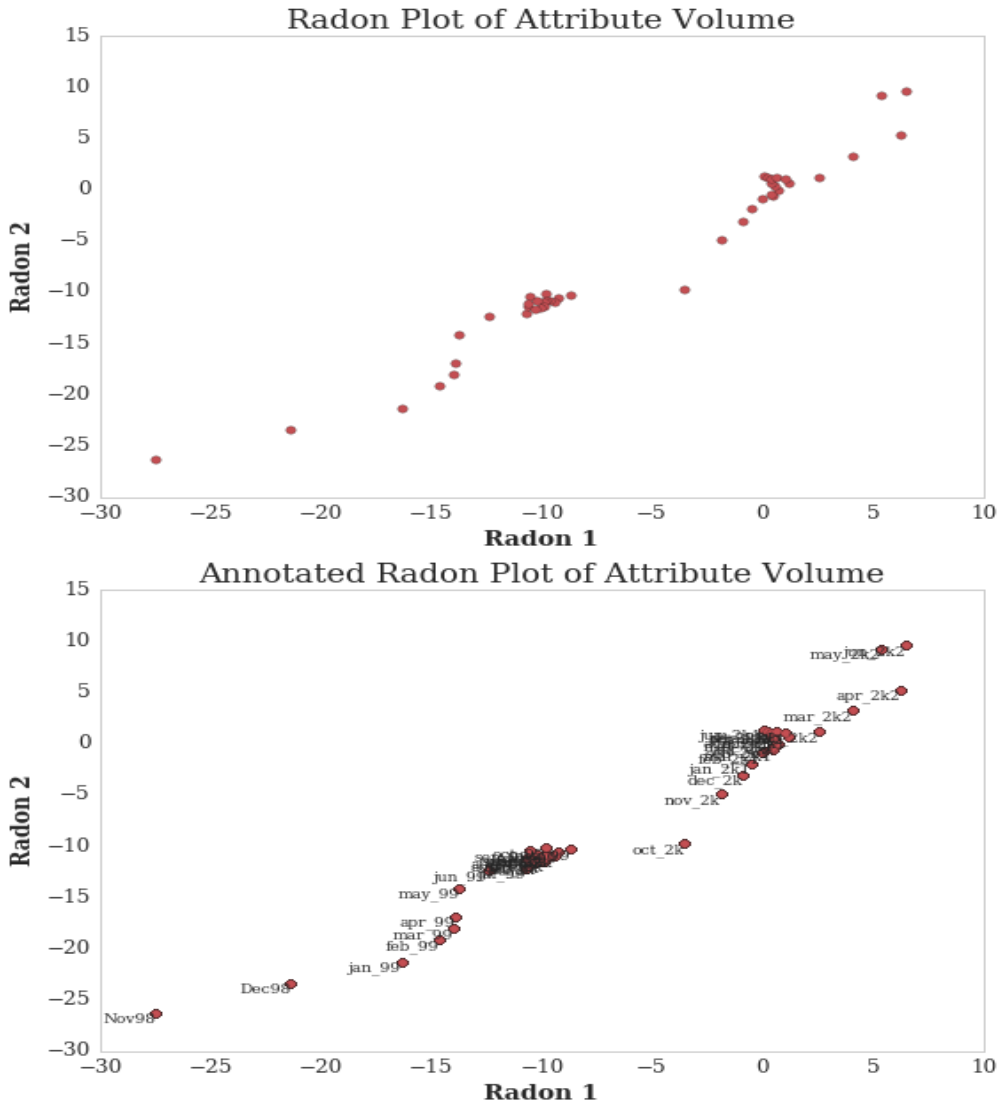


Figure 4.45: (top) Radon Plot of Attribute Volume, (bottom) Annotated Radon Plot

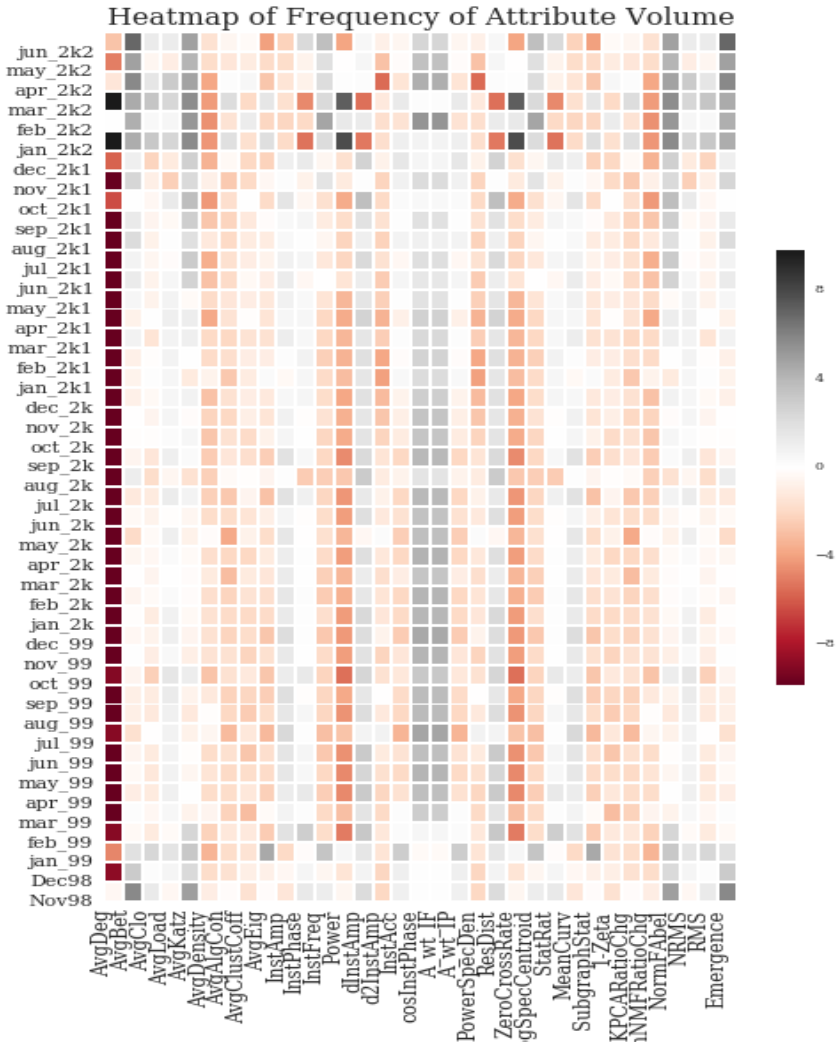


Figure 4.46: Heatmap of Frequency of Attribute Volume. This is the F component derived from the FK Plot

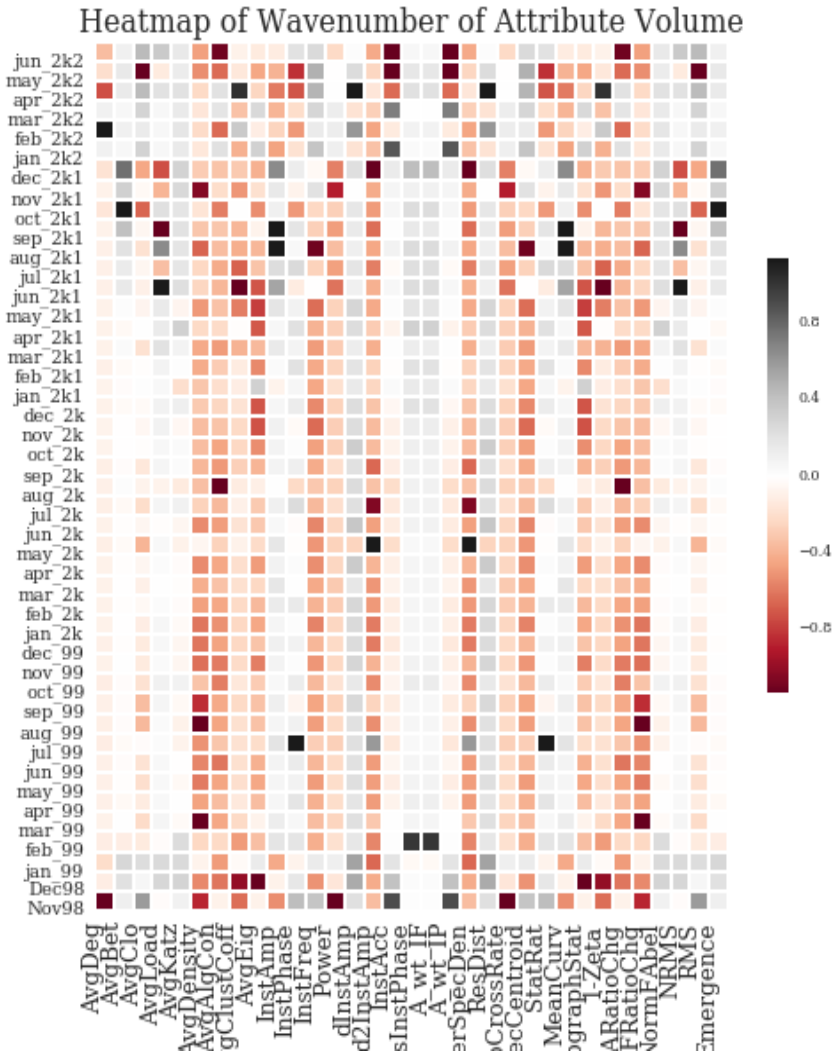


Figure 4.47: Heatmap of Wavenumber of Attribute Volume. This is the K component derived from the FK Plot

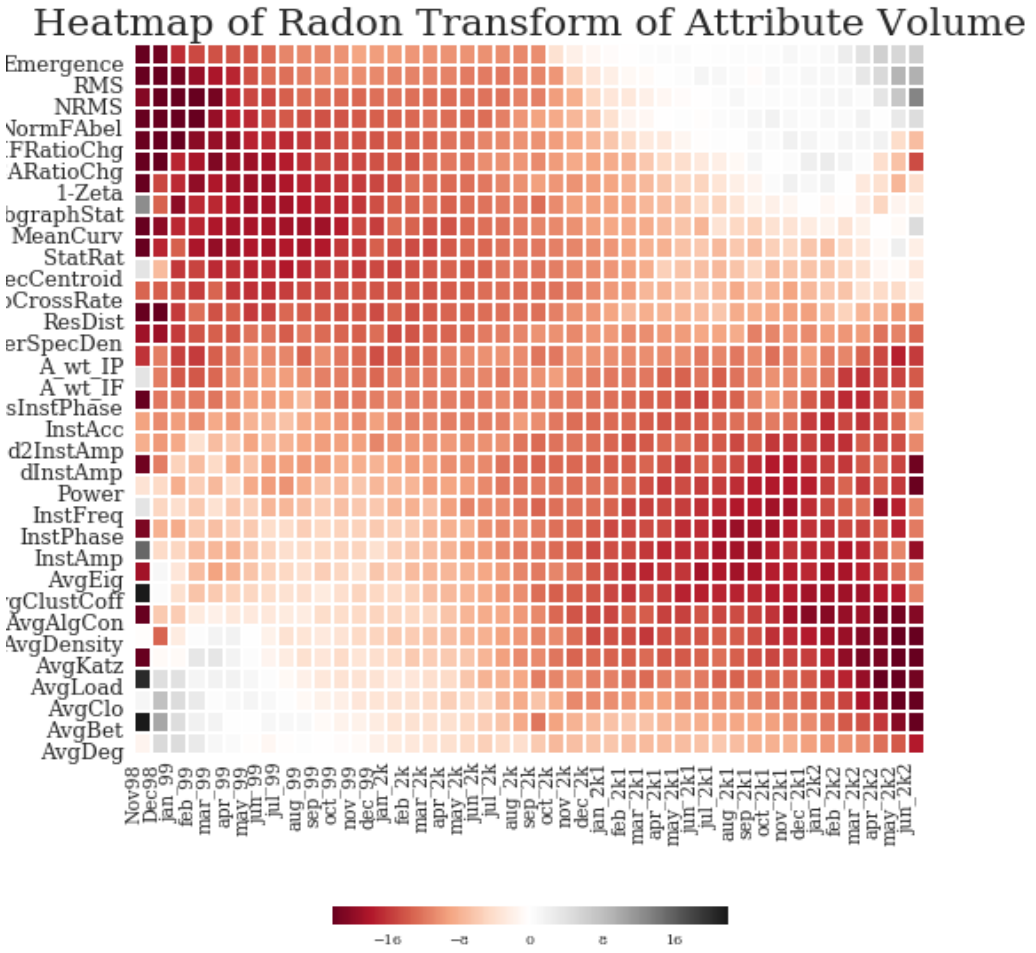


Figure 4.48: Heatmap of Radon Transform of Attribute Volume with point labels.

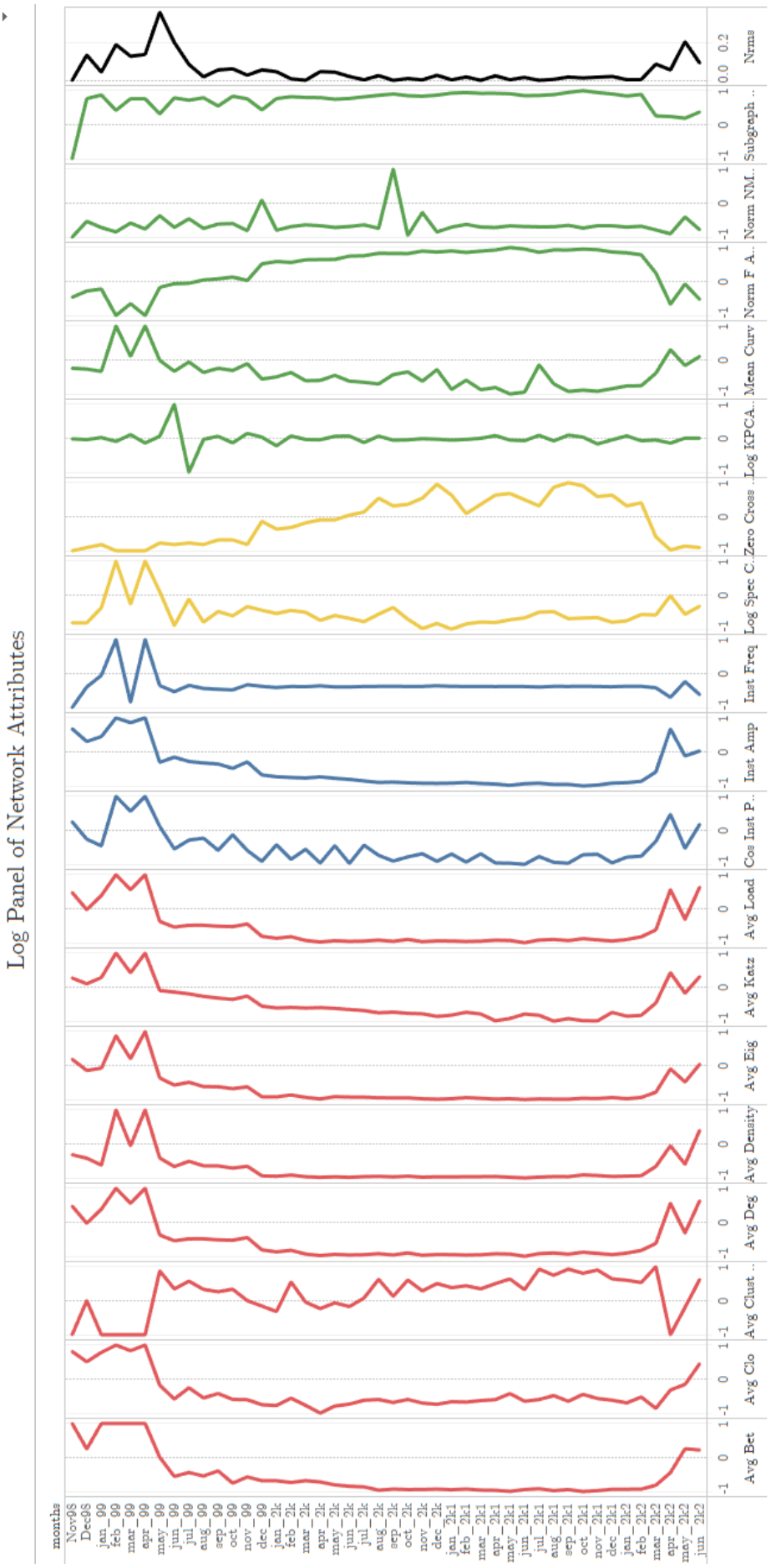


Figure 4.49: Log Panel of selected attributes. Benchmark attributes are shown in red. Seismic Attributes shown in blue. Music Attributes shown in yellow. Matrix Attributes shown in green and NRMS aggregation measure in black.

4.10 Node Level Trends

The node level trends shown in **??** shows the common nodes over 5 years and their trends. We note that node 169 is the most interesting as it has high centrality values over all the years compared to the other nodes except for Degree and Closeness Centrality in 2001 where it experiences a fall when the others experience a rise. In this way we are able to link our analysis at the network level to key nodes that are potentially driving these changes.

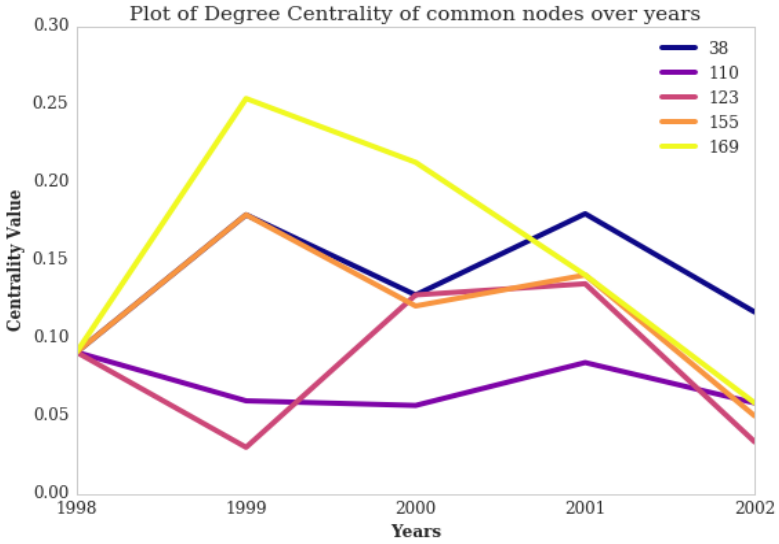


Figure 4.50: Plot of Common Node Degree over years

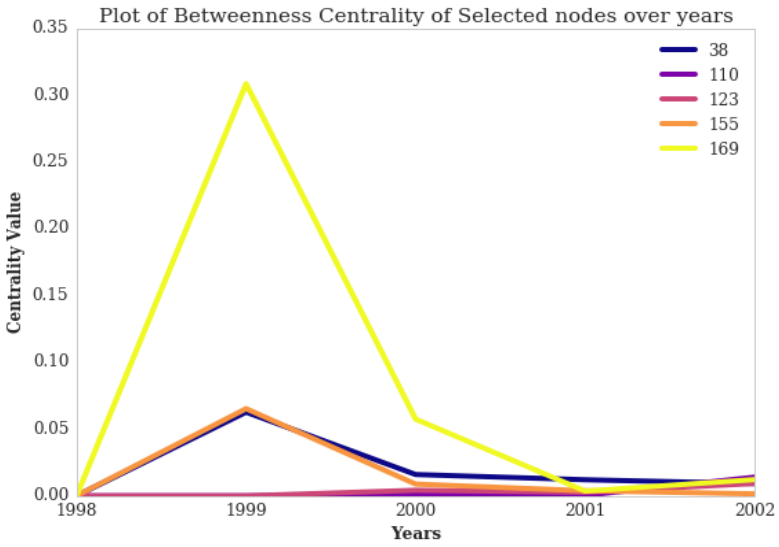


Figure 4.51: Plot of Common Node Betweenness Centrality over years

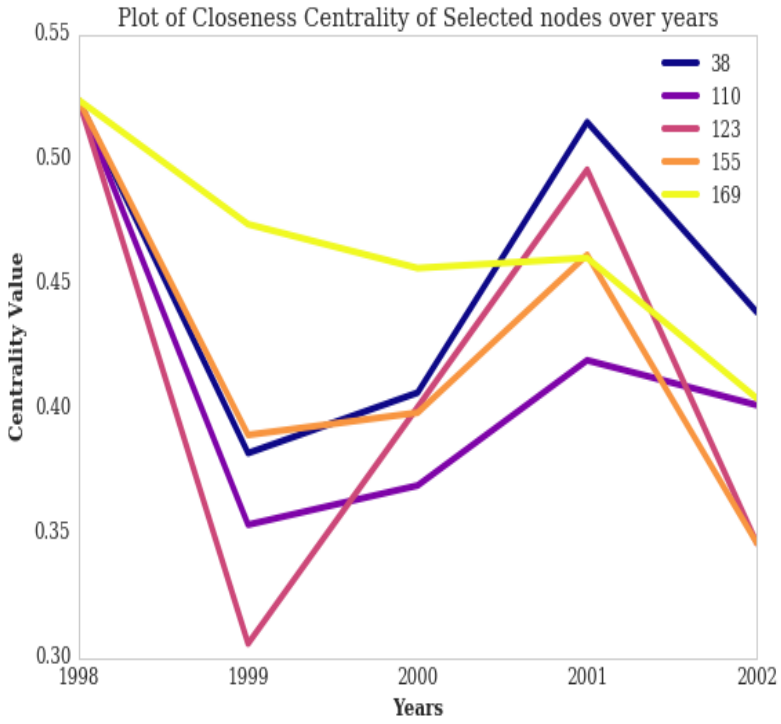


Figure 4.52: Plot of Common Node Closeness Centrality over years

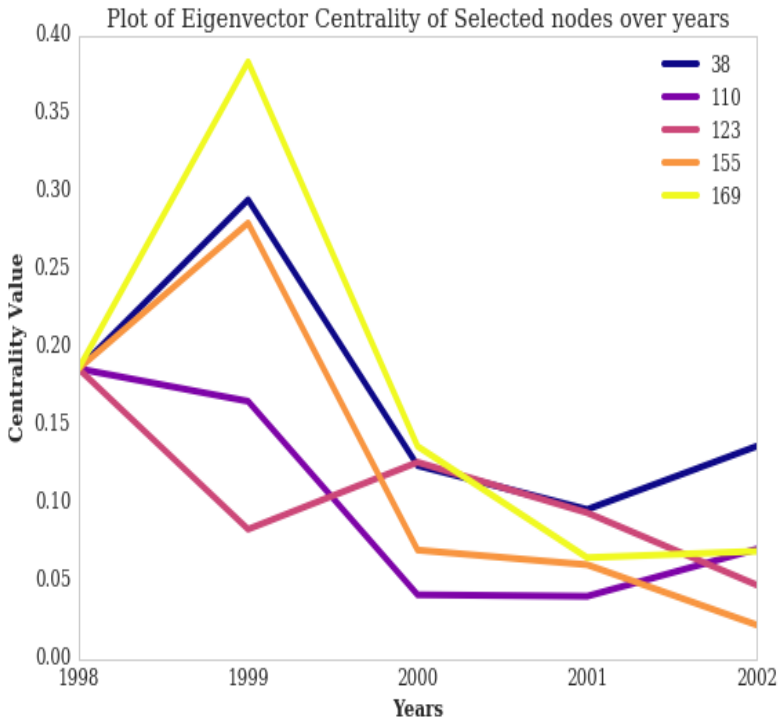


Figure 4.53: Plot of Common Node Eigenvector Centrality over years

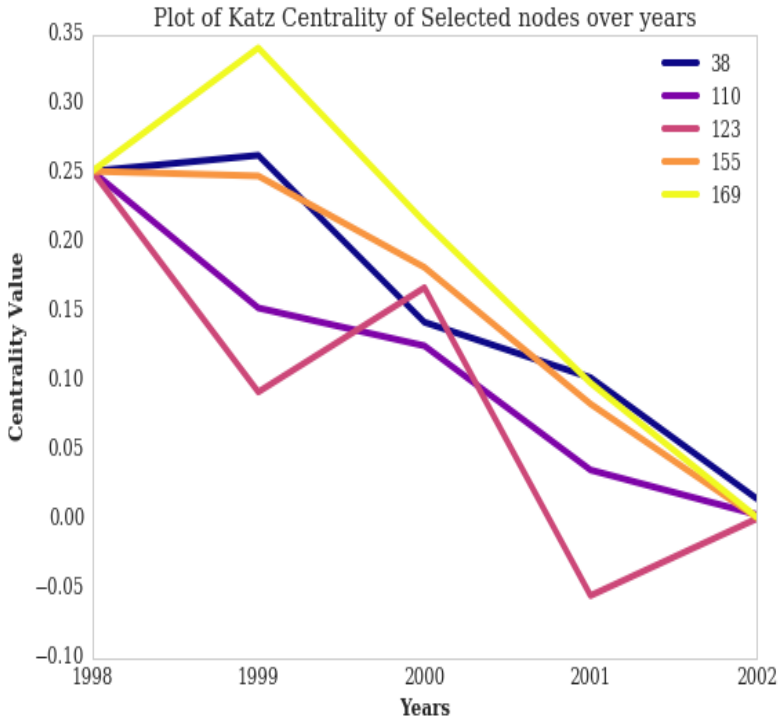


Figure 4.54: Plot of Common Node Katz Centrality over years

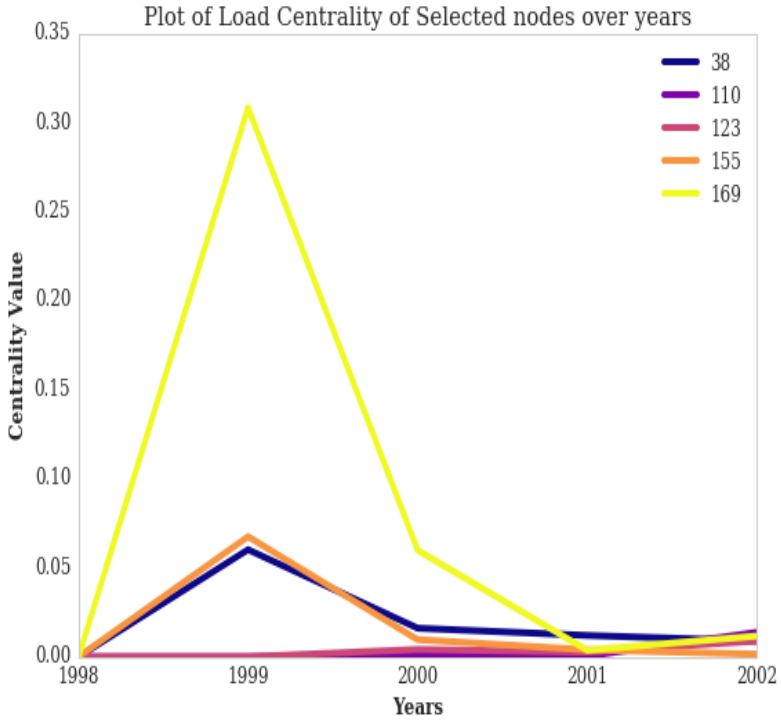


Figure 4.55: Plot of Common Node Load Centrality over years

Chapter 5

Discussion

5.1 Evaluation of Results

In this study we have presented a work flow for the systematic analysis of dynamic networks through the use of multitude of similarity measures and visualisation techniques. The methods surveyed and utilised range from traditional graph measures such as Centrality and Linear Algebra based measures to novel measures based on Hilbert, Fourier, Abel and Matrix Decomposition techniques.

The intention was to present these novel methods from a theoretical perspective and validate them through their application on real data to demonstrate their merit. As an additional level of validation all these measures were ranked on their predictive capability for a fundamental network measure such as Average Degree at a future time step. From this analysis it is clear that the novel measures proposed in this study especially the measures that are not well correlated to existing measures such as Norm of the Abel Transform and Norm of the NMF Ratio Change are particularly well suited for this specific task.

The analysis was conducted at both the year and monthly level on three different graph matrix representations. The most suitable was determined to be the Normalised Graph Laplacian as discussed in Chapter 3. From this basis a large number of measures were derived for these networks which come from the field of Seismic Data Analysis and Music Information Retrieval. These derived measures are then compared against a set of benchmark measures to assess their suitability. These measures are interesting because they are able to capture the underlying signal that is suggested by the benchmark measures and highlight additional areas of interest.

We use two information theoretic measures like the Signal to Noise Ratio (SNR) and Entropy to help assess which matrix is better.To do this all the attributes from the different matrices I calculate the SNR ratio and Entropy. For completeness the attributes are derived from 3 different graph matrices the Normalised Laplacian, Modularity and Adjacency Matrix as shown in Figures 4.21 and 4.23.

The SNR plot is useful as a first step to get an idea of the information content of the attributes derived and which matrix might be better. From Figures 4.21 and 4.22 we could conclude that any other matrix apart from the Laplacian would be a good

choice because all the other matrices have a high number of attributes with a high SNR value meaning that their deviation is low compared to their mean. But upon inspection of the attributes with high and low SNR values this turns out to be misleading. This is because when the SNR is high the attributes behave identically from the three matrices and they represent well the trends highlighted by the benchmark measures. However, for the attributes that have a low SNR the attributes from the Modularity and Adjacency Matrix attributes do not model the trend in the benchmark measures and they fail to recover some of the basic characteristics of the signal observed such as the prominent peaks at the beginning and end of the time series due large variations in network size. But the Laplacian attributes do not suffer from this issue they model the trends observed in the benchmark measures well in both cases of high and low SNR value.

Thus analysis by SNR alone was deemed insufficient. As an additional level of validation we opted to use the Entropy measure. As Figure 4.24 shows the mean Entropy and the Entropy of the attributes Figure 4.23 are very similar for the attributes from the different matrices. Thus the attributes must be judged on their ability to represent the basic characteristics of the signal from the benchmark measures shown in Section 4.3 and suggest potentially interesting areas. This is deemed to the Laplacian attributes in this study through inspection of the attributes against the benchmark measures so these attributes are used in all subsequent analysis in this work.

The attributes from Seismic Data Analysis are predominantly derived from the Complex Trace which relies on the Hilbert Transform. The Music attributes rely on the frequency waveform which rely on the Fourier or Stockwell Transform but the Fourier transform is used in this study. In addition the Abel transform is used which finds a lower dimensional slice from a 3D cylindrical symmetric projection onto a lower 2D surface. The frequency component from the Normalised Laplacian is transformed into this subspace through the Abel transform and gives us one of the more interesting attributes.

The Norm of Abel Transform attribute we note from Figure 4.32 shows that the attribute finds the initial two peaks that are associated with the Feb - Apr 1999 period. This period as we see from Section 4.1.2 that these periods correspond to very small network sizes. In the node link diagram this is highlighted by a small number of connections. In the Matrix plot by a small number of edge connections and in the audio waveform plot by the blocky appearance of the plots. The audio waveform is particularly useful because as this is based on the frequency of the Normalised Laplacian a low number of peaks is indicative of low frequency meaning there are not many edge connections. This makes intuitive sense when we look at the remaining plots all which show that the waveform has many spikes along its length and does not share the distinctive blocky appearance. When considering a large number of networks as in this study we see that the node link diagrams quickly turn into a hairball once the network sizes grow. Another problem with this visualisation is that the placement of the nodes are recalculated each time and are not invariant so the physical location of the nodes are meaningless.

The matrix visualisations on the other hand are more useful to a certain extent and reordering them by cluster indices is a useful strategy. But unless the matrices can be visualised as a cube this is also limited if static views are required. It would be very hard to present these visualisations in report form for a end user but their utility is more for the analyst. But the waveform plots can be intuitively understood in the sense that if we take the x-xis of the waveform to be indicative of the number of nodes in the network. So the waveform length becomes symbolic of network size then the spikes correspond to the frequency of their connections. In a large network we would then expect to see lots of spikes and a longer length of the waveform. Although the Fourier Transform to derive this requires a certain length in which case the trace is padded to a minimum length but even then the frequency of the spikes and their wavelength gives us an indication of the underlying network in a concise way in a manner that is invariant to position and order. This makes it a visualisation better suited for both static and interactive views.

The node link and matrix diagram gives us an additional level of validation for the explanation of these plots. In the case of the Norm of Abel Transform attribute the trend of the attribute captures this intuition well. The reason being that the peaks in the Feb-Mar 1999 period at the beginning of the time series to the Apr - Jun 2002 period at the end of the time series correspond to the most dramatic changes in the network size. In the 1999 period the network is unusually small and at the end the network is in the process of thinning out which corresponds to the large spikes in the signal again. The remaining time periods in the time series correspond to increasing or larger size of the network in comparison to these periods. Hence the Centrality and the other traditional measures show these periods as having smaller values. This part of the series for these benchmark measures are dominantly smooth with the exception of the Average Clustering Coefficient. The reason being that an increasing size of the network usually results in more local clustering which this metric measures. This metric captures the intuition well that as the interactions in the network grow the metrics should show a similar trend. However, the centrality measures show this is in a counter intuitive way and so do most of the Seismic Attributes which are highly correlated to these centrality measures.

From the seismic attributes such as Amplitude and the attributes derived from it such as Power, the first and second derivative of Amplitude, highlight the big changes in the network very well. These measures in Seismic Attribute analysis are used to detect amplitude anomalies and in this context these periods which cause the spikes observed correspond to such anomalies. The Amplitude anomalies and by extension the Power anomalies thus correspond to the time periods when the network is unusually small as compared to the other time periods. It is encouraging to see that the analogy from seismic holds well in the context of dynamic network analysis.

The Phase (IP) attribute is used as a direct indicator of hydrocarbons as the presence of fluids in the rock causes phase changes for the waves passing through it in a seismic experiment. This highlights additional periods which undergo change. The raw phase attribute is not particularly smooth and difficult to interpret as it appears very noisy. Therefore it is common to look at the cosine of the Instantaneous Phase for display purposes as taking the cosine constrains the signal to a $[-1, 1]$ range and

reverses the polarity of the signal to that which is more preferable i.e looking at peaks or troughs. The cosine of Instantaneous Phase highlights additional areas in addition to these anomalies already noted well.

Also the Phase derived attributes such as the Frequency and Acceleration represent a smoother attribute than the raw Phase attribute. The Frequency attribute highlights the major anomalies particularly well and highlights some of the peaks highlighted by the Phase and cosine IP attributes and the Average Closeness Centrality measure. However these additional areas of potential interest are not very clear from these measures but the Acceleration highlights some of these additional periods such as Mar 2000 well.

The Phase and Frequency attributes weighted by Amplitude are better in the sense that they are able to distinguish the peaks caused by the small network size and highlight additional spikes better such as Oct - Nov 1999 periods. The correlation between the Seismic Attributes and the Centrality and Assortativity measures are surprising because the centrality measures are predominantly calculated from the adjacency matrix while the Seismic Attributes are calculated form the Normalised Laplacian.

The matrix based attributes such as the Curvature, Resistance Distance, Stationarity, Subgraph Stationarity and Power Spectral Density highlight other features of the signal apart from these two peaks. The Power Spectral Density suggests that the periods between Aug - Oct 2001 are more interesting that the early 1999 and late 2002 periods. This attribute suppresses these parts of the time series and suggests this period as being more interesting. This is interesting because from what we know about the network the beginning and the end are due to network size being abnormally small but this part of the time series the networks are fairly dense thus it has the ability to highlight interesting areas among denser parts of the graph time series and suppress noisy parts. It is possibly that this period has the most dense part of the network which translate to higher Power Distribution over time compared to the rest of the time periods.

The Resistance Distance measures the resistance between nodes analogous to the flow of current in electrical networks thus it could reasonably be assumed that given more paths in a network that the connections will find the path of least resistance. It is this intuition that we expect this measure to capture. So we see peaks corresponding to the early and late periods but the bigger peaks correspond to later time periods when the networks are more dense. This suggests that during these periods even though the networks are getting dense that they are not fully connected. This is suggested by the Algebraic Connectivity as well but the Resistance Distance is clearer about these time periods in the sense that it highlights peaks in these areas such as Feb - Jun 2000.

The Music Attributes are interesting because they capture fairly different behaviour from the waveforms. The Zero Crossing Rate which is the number of times the signal crosses zero over time is greater when the network is dense and less so when the networks are sparse. The Spectral Centroid measure which calculates the centre of

gravity of a spectrum is lower for denser parts than it is for the sparse parts which is expected because it is essentially weighted mean of the frequency components which are likely to get smoothed out when the network is denser in contrast to when it is more sparse.

The Mean Curvature attribute finds the July to Nov 2001 period as being particularly interesting which is corroborated by the other matrix attributes and not as clear from the benchmark measures. In Seismic Attribute Analysis changes in curvature are associated with structures such as faults, fractures and discontinuities. It is encouraging that the discontinuities in this chase the sparse networks of the graph time series are highlighted in addition to interesting parts from the denser section of the graph time series.

The stationarity attributes such as the Stationarity and the Subgraph Stationarity highlight the sparse parts of the network as troughs and denser parts as peaks and point to the July to Nov 2001 as having undergone the most change. In the context of the stationarity measures this translates to large positive changes in the size of the network.The 1-zeta attribute which shows the proportion of members that change at a time step shows that the early and later parts of the time series as having undergone the greatest change in the proportion of members.

The KLPCA Ratio is used to detect large changes in seismic data such as unconformities from this we see that the largest change in this attribute is the Mar - July 1999 range this just after the sparse parts of the network at the point where the network starts to become dense. This is very interesting as it is a change point in the network that is not clear from the other measures. Most of the other measures highlight either aspects of the sparse part or the dense part of the network but the KLPCA ratio highlights this change point that would have been missed otherwise.

The Norm NMF Ratio highlights the periods identified by the other attributes but refines the range from Aug - Nov 2000. The other changes are far less prominent in this attribute.

The Abel Transform attribute captures the intuition very well that the sparse parts of the network shows up as smaller peaks and from the change point there is peak for the areas of the time series where the network is dense. The trough at the end of the time series corresponds again to the thinning of the network. This measure captures very well the dynamics of the whole time series highlighting the insights gained from all the remaining measures.

In terms of the trends for the years all the attributes agree that the biggest change in the series was between 1999-2001. This corresponds to the densification of the network and is picked up by all measures with different levels of clarity. The yearly aggregation is not very interesting as there are not many samples but this is the main signal from the yearly time series.

From the MDS and TSNE plots in Figures 4.40 and 4.41 we get a sense of the clustering of the attributes. As expected the Centrality measures are close to each other

but the surprise is the Load and Density metrics which are separate from everything else on the MDS plot. Also on the TSNE plot with the Euclidean Distance the Load stands out as being separate from the rest while the Norm attributes such as the NMF and Abel attributes are close together. The Centrality measures are also close together as are the seismic attributes to which they are well correlated. Another curious feature with this distance metric for the TSNE is that the aggregation measures are not close to each other but are fairly well separated while with the other two distance metrics they are fairly close to each other.

The Correlation distance seems to suggest that most of the attributes have low correlation distance hence the presence of the central large cluster with the Average Eigenvector Centrality, Zero Crossing Rate and Power attributes standing out. This is counter to what the correlation matrix has shown for starters that most of the attributes are weakly correlated, while some are negatively correlation and some are strongly correlated. So this distance metric is not recommended for this application. The Canberra distance is better because it gives good separation among the attributes. Also interestingly the Music Attributes are placed close to each other. So overall the MDS visualisation is probably better to use as its lack of options for the dissimilarity metrics makes the analytical options limited which is helpful in this case.

The Radon and FK Plots in addition to their attribute map give us an additional way to visualise the whole attribute volume over time. From the Radon plot it is clear that there is some structure in this data with 3 apparent clusters consisting of the early part when the network is very sparse the growth and the final decay of the network. Also the FK plot allows us to visualise the attribute volume not only together but as slices through the FK attribute volume.

The Log Panel gives us a way to easily analyse multiple attributes in the style of well log analysis used for hydrocarbon exploration. The idea here is that we have panels of different attributes and in this the case the attributes which are highly correlated are clustered together in the panel display. This allows for easy peak and trough tracking across the time series for multiple attributes at the same time. This gives as an easy way to gain insight from the graph time series utilising not just traditional graph measures but the Seismic and Music Attributes introduced in this study.

Also I have identified 5 key nodes at the yearly aggregation level. The trends in the centrality of these nodes broadly mimic the trend observed at the yearly scale. They are identified by taking the intersection of the node lists from the yearly networks. These nodes appear to experience the largest change between 1999-2001. Node 167 seems to be particularly important as it seems to have high centrality values across all the measures over the time period compared to the rest of the nodes. For node 199 values for the centrality measures rise sharpest from 1999-2000 but falls thereafter while node 38 has a sharp rise in the measures from 2000-2001 compared to the rest of the nodes. Although it was not possible to locate employee id data for these nodes so they could be merged but one could posit that these nodes could probably represent senior managers whose influence changed but did not diminish completely over time.

From the correlation analysis shown in Figures 4.33 to 4.35 we see that the traditional metrics are well correlated to a lot of the new metrics proposed. Most particularly it appears that the centrality measures are particularly well correlated to some of the Seismic and Music attributes such as Instantaneous Amplitude, Cosine of Instantaneous Phase, Mean Curvature, Spectral Centroid and Power.

The Spectral Centroid appears to be strongly correlated to the Eigenvector and Katz Centrality which are based on the eigenvalue analysis of the graph matrices as opposed to the other centrality measures. The remaining attributes such as the matrix attributes and the Zero Crossing Rate do not seem to be well correlated to the other metrics but rather negatively correlated to most of the metrics. Particularly interesting are the Abel attribute which is negatively correlated and the NMF attribute which is not correlated to the other metrics but appear to have high predictive capacity. The predictive importance of these attributes is shown in Figure 4.37 and the dominant attributes are the Abel and NMF attributes which are not well correlated to the other measures. Thus a case can be made that these measures are capturing important network dynamics while the correlated measures are capturing similar information to the centrality measures. In addition it could be argued that the matrix attributes such as the Norm of the Abel Transform, Norm of the NMF Ratio Change, Stationarity type attributes and Curvature capture vital network dynamics that enable the prediction of fundamental network attribute at a time step in the future. Thus we have a framework for linking all our results.

The aggregation schemes are important as they enable us to develop a single metric that serves as a measure of network activity. Therefore having a suite of measures which capture different aspects of network dynamics well should reflect in the final aggregation. But there is always a potential drawback that such aggregation measure smoothes out smaller changes. Hence the Log Panel of multiple attributes can provide additional context. Of the 3 measures introduced the NRMS is preferable because it is a normalised RMS measure so it already has a degree of noise suppression due to the RMS operator. But since all the attributes are scaled to the $[-1, 1]$ range this is perhaps less important. But what it highlights well is that it picks up bursts in the network in addition to smaller changes , is numerically better constrained and overall much smoother than the RMS and Emergence measure.

Therefore it can be said that from these measures we can identify not only interesting time steps but also interesting nodes over time. The node level analysis is only presented for the centrality measures because since they are well correlated with a lot of the Seismic and Music attributes so they would not tell us much new information. Also since these nodes seem to mimic the trend in the time series it is easier to think about them in terms of traditional centrality measures because the matrix attributes are more network level attributes.

5.2 Generalisation of Analytical Work flow

Therefore generalising this analytical work flow for any dynamic network scenario is possible. This can be done by the approach followed in this study. First, for the network under consideration decide the granularity at which to conduct the analysis such as yearly, monthly, weekly or daily. Although depending on the type of network there might not be sufficient network density at too fine a granularity such as daily for meaningful analysis but this can be evaluated against the data available and the application. Once the granularity has been decided segment the network into the required granularity and for each time step derive the Normalised Graph Laplacian. The attribute volume can be generated easily from this and an aggregation scheme can be chosen such as the RMS, NRMS and/or Emergence. Then as more data is available the process can be repeated at different time step resulting in a snapshot of the network through these metrics. The correlation analysis will suggest which measures are likely to be redundant and thus a subset of the more interesting measure can be selected. This can be seen as a pruning step. Not all measures will be useful in every application and it is therefore important to discern how each measure performs depending on the domain. Since these measures have only been applied in an email network context they might behave differently in another domain. This should be checked for different application domains.

Once the measures and the aggregation schemes are picked a potential dashboard could consist of an overview panel of the aggregation measure highlighting network activity. Then the Log Panel of selected metrics could provide additional insight of network dynamics. The Radon plots can be used to assess structure in the data and the FK plot can be used for outlier detection.

When interesting time steps are identified the Waveform plot of the network would give a good overview of the underlying network with further investigation through the Matrix and Node link views.

5.3 Discussion of Aims and Objectives

We can formulate some questions around our objectives that will structure our discussion of them.

- What are some of the graph similarity measures that have been proposed in the literature?
- How have these been used in a practical context such as email data?
- How can such measures be applied to dynamic networks?
- How are such measures evaluated?

What are some of the graph similarity measures that have been proposed in the literature?

An extensive review has been presented in Chapter 2, regarding the issue of graph similarity and matching. Some of the methods suggested for similarity and graph matching are visual, spectral, tensor and distance based approaches. The different methods proposed have differing levels of complexity but far the most convenient approach was determined to be the feature based approach. This is because features are more compact snapshots of a network and we can have a large number of them to characterise the network and this gives us a way to treat the features or collections of features called the Attribute Volume in this study to be treated as a time series. This allows many analytical methods form time series analysis and signal processing to be applied to this volume.They allow us to easily analyse a large graph time series in a highly convenient manner.

How have these been used in a practical context such as email data? How can such measures be applied to dynamic networks?

The traditional network measures used in this study have typically been used in the case of static networks. However, in the case of dynamic networks the measures have been used as a time series with control charts used to determine signal. But in this study we use scaled attributes so we can detect spikes in the attribute time series and identify time steps of interest.

So for email networks Degree measures could represent the communication pattern in the network. Thus nodes with high centrality measures can be detected. Also the novel measures proposed can used to highlight interesting time periods. So a time period identified as a amplitude or frequency anomaly for example could be investigated for the drivers of such change. This can lead to node level analysis through centrality measures as highlighted by the node level analysis presented in Chapter 4.

How are such measures evaluated?

The key thing to keep in mind when evaluating such measures is that there is not a right answer that we necessarily know of to begin with. There might be in some cases such as the Enron case where we know that in 2002 it went bankrupt when the scandal emerged. This is reflected in the graph time series through spikes in the attribute volume indicating a network that is undergoing some rapid change in this case the network was thinking out and dying. But in most cases we might be interested in finding the interesting times that we do now know off but would like to identify. With this is mind we can devise a more thorough evaluation procedure.

The evaluation approach used in this study is multi pronged in the sense that the first step was to try an establish some notion of ground truth. This is done through the use of the benchmark measures. These measures are chosen as they are widely understood in the field and have well developed interpretations. From these measures we get a notion of the signal in the graph time series as to where the peaks and troughs are. For example the Average Clustering Coefficient is large when the networks are getting dense this correlates well to the Stationarity Ratio, Resistance Distance, Power Spectral Density and Zero Crossing Rate. These measures recovers the characteristics of the Average Clustering Coefficient and highlights the char-

acteristic two peaks of the other centrality measures. So comparison to existing benchmark measures is the first step. Secondly, we perform correlation analysis as well as an MDS and TSNE to get a notion of their similarity. Most of the attributes appear well correlated and some attributes are negatively correlated while a lot of the attributes have weak correlation. This indicates that from a correlation perspective only a few attributes are highly correlated so potentially have redundant information but encouragingly most of the attributes are negatively or weakly correlated to existing measures suggesting they are capturing dynamics that the traditional measures are not.

The Correlation Network shows high degree for the correlated attributes while the Degree Histogram shows that 5 attributes are not connected to anything as there is not sufficient correlation such as the Instantaneous Phase and Derivative of the Amplitude attribute. The TSNE plots can make use of a range of distance functions so to get a sense how things are affected a few different distance measures such as the Euclidean, Canberra and Correlation Distance measures are tried. The Euclidean distance gives a somewhat comparable result to the MDS experiment whereas the Correlation distance gives one large cluster of attributes and the Canberra Distance gives a good separation among the attributes. This gives us an additional level of understanding that our novel attributes are in fact fairly unique and are capturing additional network dynamics that the traditional measures are not.

As a final level of evaluation and validation I propose regression testing to predict a fundamental network attribute based on the attribute volume. In this instance I chose the Average Degree this enables us to perform feature ranking and measure how well the novel attributes are at capturing network dynamics. If these metrics were not very good one would reasonably expect that the regression model would rank them in low importance but surprisingly we find that the novel measures are much more important to build a predictive a model of the dynamic network. In addition it appears that measures that are not well correlated are more important than those that appear to be well correlated. From this we can conclude that these measures are capturing the basic signal hinted at by the benchmark measures, capturing the dynamics sufficiently well so they are highly correlated thus containing redundant information and they have predictive potential.

The original aims were as follows:

- To provide a fairly comprehensive overview of graph theory as is relevant to the understanding of the derivation of similarity measures
- Conducting an in depth literature review on the graph similarity measures proposed and that have been demonstrated to be useful in a practical context.
- Compare the utility and performance of these measures on appropriate data
- Explore the viability of developing a novel similarity measure based on Fourier and Hilbert Analysis of networks

Based on the discussion so far we can assert that all the aims and objectives for this project have been met. In addition they have been exceeded with metrics derived

not just from Fourier and Hilbert spaces but Abel spaces and from Digital Music. Their interpretations have been discussed as well as their inclusion in a systematic and reproducible framework of analysis.

5.4 Research Question

How can similarity measures be used to analyse dynamic email networks? How can similarity measures in alternative spaces support the analysis of dynamic networks?

The research question has been addressed in many different forms over these chapters but to discuss it conclusively we can say that similarity measures can be used as a way of convenient analysis of dynamic networks. This is because the feature based approach as illustrated in this study enables us to treat our dynamics networks as a graph time series and thus allow us to apply signal processing and time series analysis approaches.

The feature based approach used in this study is used to derive a attribute volume that is then used to support multiple attribute interpretation and characterisation of the graph time series through novel visualisation techniques. Also it allows us to derive aggregate measures that serve as useful measures of network snapshot and support the overview first, zoom and details on demand type analysis. This is discussed further in Chapter 6, where the generalisation of this approach is discussed.

Utilisation of similarity measures in the context of email networks or any dynamic network serve as convenient snapshots that allow us to build a time series to enable analysis of such networks at any level of granularity.

The derivation of attributes from alternative spaces such as Fourier, Hilbert and Abel spaces allow for novel attributes and visualisations to be applied. The Fourier Transform forms the basis of the Music Attributes that are derived because we turn the Normalised Laplacian into a frequency trace which can be then visualised as a waveform and then this waveform allows derivation of attributes such as the Zero Crossing Rate and Spectral Centroid. In addition the Fourier Transform forms the basis of the FK plot helps to visualise the large high dimensional attribute volume in a 2D space and identifies outliers. The Frequency and Wavenumber components of this plot can also be visualised as Heatmaps. Also the Abel Transform attribute takes the magnitude of the real and imaginary components of the frequency derived from the Fourier Transform and finds the lower dimensional representation of it. This attribute as the regression analysis shows is very useful for building a predictive model for this particular dynamic network. The Hilbert Transform forms the basis of the derivation of the Seismic Attributes as it enable the creation of a complex analytical trace from a real valued function. These attributes have proved useful in identifying anomalies in the time series.

Another class of attributes that were not in the original plan were the so called matrix decomposition attributes such as the PCA Ratio and NMF Ratio which have

proved their ability to detect change points and have been shown to be useful for the Regression model.

Therefore the use of similarity measures can aid the analysis of dynamic networks by creating an attribute volume over time that can be used for network interpretation and characterisation. The use of alternative spaces allows for the derivation of new and creative attributes as well as visualisation approaches to facilitate analysis as well as predictive model building for dynamic networks.

Chapter 6

Evaluation, Reflection and Conclusions

6.1 Reflections

The aim of this study was to firstly conduct a systematic review on the topic of dynamic network analysis. More specifically the use of similarity measures to support such analysis.

From the literature we see that a plethora of methods have been proposed and applied by researchers to the problem of dynamic network analysis. These approaches can be broadly be summarised into but not limited to distance, feature and visualisation based methods. Informed by these ideas we developed some novel measures of our own and implemented some of the measures from the literature such as the Persistence and Emergence measures.

These were tested on the Enron Email network data split at the yearly and monthly level. At the planning stage the acquisition of suitable data was a concern as many versions of this data exist online with the preprocessing provenance unclear. But we used the John Hopkins version as this had time stamps that could be used to segment the data.

This study used as a starting point traditional measures of network analysis such as centrality measures along with other network statistics such as density and clustering coefficient to derive a time series of features from the graph time series.These benchmark measures were then used to establish a signal for the network and network visualisations aided in developing the explanations. But the main interest was to see whether additional novel measures could be developed and applied to dynamic networks. As a result of this we were able to introduce and demonstrate the use of novel metrics from the fields of Seismic Data Analysis and Music Information Retrieval.

The key challenge in figuring out how to implement the Seismic attributes were to establish which data representation to use. Since graphs can be represented by a variety of matrices such as Adjacency, Modularity, Laplacian and Incidence which would yield the most stable attributes was the first concern. The convenient part of

this process was however that there exist good libraries for Python for the task of signal processing so the key methods required such as Hilbert and Fourier Transform were readily available. However, the more exotic methods such as Abel Transforms and Radon Transforms came from more specialised packages. But their general availability contributed greatly to the success of this work.

The Hilbert Transform formed the basis of most of the Seismic attributes and some image processing libraries were used for their implementations of the Hessian Matrix Eigenvalues which was required for curvature calculations.

For the Music attributes implementation was almost abandoned due to the not being able to source well documented code for the implementation of the Stockwell Transform which is a common transformation used in the digital music domain. However, it was realised that the Frequency content that we were trying to derive could be derived also by the Fourier Transform. This realisation allowed us to be able to develop the Music attributes and the waveform visualisations for the dynamic networks.

From the start it was realised that this project would be a highly technical and that the success would depend on being agile. From the Agile methodology some key approaches that were followed in this work were:

1. Active user involvement is key
2. Requirements evolve but the time scale is fixed
3. Focus on frequent delivery of results
4. Testing is integrated throughout the project life cycle - evaluate what works otherwise move on
5. Adopt a collaborative and cooperative approach between all stakeholders

Taking this approach meant that we were able to circulate weekly reports of the results and have regular meetings around them to discuss the methods being applied. The key consideration was to always establish how these metrics would be useful in analysis and the best way to demonstrate it. This would require us to develop a story for the interpretation of these metrics so there is some intuition as what we could expect these metrics to capture.

To this end the Seismic attributes were developed and evaluated first since there exists a large body of literature on their varied use in the hydrocarbon exploration. Metaphors are used to build intuition around the Seismic attributes based on their use in industry.

The derivation of the music attributes were greatly simplified by using the Fourier Transform to derive the frequency components and then collapsing the frequency trace into a single channel by averaging. Although a large number of measures exist for Music Information Retrieval some of the measures such as Cepstral based attributes were not possible due to the short length of the monthly networks which

would require a large number of padding for the Fourier window. Thus the resulting output would be difficult to interpret. So these are left for future exploration.

Thus we are able to present a large body of metrics for the analysis of dynamic metrics derived from completely unrelated to the the field of network analysis. We show that they work well have strong interpretations and analytical potential. Integral Transforms were key to deriving these attributes. So there is a scope for future work to explore more exotic transforms such as the Mellin, Hankel transforms and their potential applications.

As a result of adopting and implementing a solid project plan coupled with effective management the project could be delivered comfortably within the set timescale. In addition the original aims were exceeded because initially we set out to explore some alternative spaces such as Hilbert and Fourier spaces this was mainly with the aim of deriving potential seismic attributes. The fact that we could use the Fourier space to derive a suite of Music attributes and visualisations was an unexpected realisation but a highly interesting side effect.

6.2 Suggestions for Future Work

The key recommendation for future work would be to take the novel measures proposed in this work and apply them to data sets from different domains and validate their performance. Also we have shown that we can use some exotic integral transforms such as the Abel Transform to derive some very interesting attributes. The utility and effectiveness other integral transforms such as Mellin, Hankel and others could be investigated. Some music attributes such as those based on Cepstral methods were not explored in this study but we have shown that this can be done on networks thus other work could explore the application of Cepstral and Mel Frequency Cepstral techniques for the derivation of additional novel attributes.

6.3 Conclusions

In conclusion, it can be said that the original aims and objectives for this study have been met and exceeded. Not only have we shown that we can derive many exciting attributes from integral transforms from other fields of study but we have highlighted that more measures from these fields can be imported.

These integral transforms are not only useful for the derivation of attributes but they can be used to support visualisation techniques such as the Radon, FK and Waveform plots.

With regards to the dataset we noticed in the raw data that there were some mislabelled data which we did not consider. This left us 5 years of data with 44 months. Utilising the yearly and monthly aggregations we illustrated our generalisable work

flow for the systematic analysis of dynamic networks.

This consisted of firstly separating the network into the required aggregation granularity without successive agglomeration of the time periods. We then derive an attribute volume which consists of a set of benchmark and additional measures. These measures include meta attributes which serve to provide snapshots of the network activity at each time step by collapsing the attribute volume through some function. We show that the RMS and NRMS aggregation schemes work very well in this context.

From the network analysis it is clear that the bursts are noticed by traditional measures when the networks are sparse and change points between sparse and dense parts of the network are not very clear. The newer measures proposed some such as the Kernel PCA Ratio and Norm of the Abel Transform attribute highlight this well. The Abel Transform attribute in addition also captures the trend in the time series that the graph time series is sparse in the beginning, it gets dense and thins out when the scandal hits Enron.

We build correlation matrices, networks as well as MDS and TSNE plots to explore the relationship between the large number of attributes. From these we discover that there is large amount of correlation among existing and new measures proposed. This is surprising because graph measures have been designed specifically with the topological characteristics in mind but these measures are completely reliant on integral transforms and their derivatives. this could suggest that these measures have analogues in other fields and could be calculated differently. There is also the possibility that this is a feature of this dataset and more verification on this aspect is required.

The work flow and methods used are highlighted in the Appendix section in order to allow full reproduction of results.

Bibliography

[1] Anurat Chapanond, Mukkai S. Krishnamoorthy, and Bülent Yener. "Graph theoretic and spectral analysis of Enron email data". In: *Computational and Mathematical Organization Theory* 11.3 (2005), pp. 265–281. ISSN: 1381298X. DOI: 10.1007/s10588-005-5381-4.

[2] Ting Li and Qi Liao. "IEEE International Conference on Cloud Computing and Big Data Analysis Dynamic Networks Analysis and Visualization through Spatiotemporal Link Segmentation". In: (2016), pp. 209–214.

[3] Fabiola S.F. Pereira, Sandra de Amo, and Joao Gama. "Evolving Centralities in Temporal Graphs: A Twitter Network Analysis". In: *2016 17th IEEE International Conference on Mobile Data Management (MDM)* (2016), pp. 43–48. DOI: 10.1109/MDM.2016.88. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7551567.

[4] Zhenyu Wu, Yu Liu, and Jianwei Niu. "A Novel Graph-Based Method to Study Community Evolutions in Social Interactions". In: *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom)* 1 (2015), pp. 62–67. DOI: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP.2015.33. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7518210.

[5] Chuan Hu and Huiping Cao. "Discovering time-evolving influence from dynamic heterogeneous graphs". In: *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015* iv (2015), pp. 2253–2262. DOI: 10.1109/BigData.2015.7364014.

[6] Arash Golibagh Mahyari and Selin Aviyente. "Fourier transform for signals on dynamic graphs". In: *Conference Record - Asilomar Conference on Signals, Systems and Computers* 2015-April (2015), pp. 2001–2004. ISSN: 10586393. DOI: 10.1109/ACSSC.2014.7094822.

[7] East Lansing. "TEMPORAL NETWORK TRACKING BASED ON TENSOR FACTOR ANALYSIS OF GRAPH SIGNAL SPECTRUM Marisel Villafa ˜ Department of Electrical and Computer Engineering ," in: (2016), pp. 1–5.

[8] Allen Dickson. "Introduction to Graph Theory The Konigsberg Bridge Problem". In: *October* October (2006), pp. 1–15.

[9] Kathleen M. Carley. "Dynamic Network Analysis". In: September (2001).

[10] Kathleen M. Carley, Jana Diesner, Jeffrey Reminga, et al. "Toward an interoperable dynamic network analysis toolkit". In: *Decision Support Systems* 43.4 (2007), pp. 1324–1347. ISSN: 01679236. DOI: 10.1016/j.dss.2006.04.003.

[11] Tom A B Snijders. "Introduction To Dynamic Social Network Analysis". In: *BRiMS 2015* November (2015).

[12] Stephen P. Borgatti, Kathleen M. Carley, and David Krackhardt. "On the robustness of centrality measures under conditions of imperfect data". In: *Social Networks* 28.2 (May 2006), pp. 124–136. ISSN: 03788733. DOI: 10.1016/j.socnet.2005.05.001. URL: http://www.sciencedirect.com/science/article/pii/S0378873305000353.

[13] Stephen P. Borgatti, Kathleen M. Carley, and David Krackhardt. "On the robustness of centrality measures under conditions of imperfect data". In: *Social Networks* 28.2 (May 2006), pp. 124–136. ISSN: 03788733. DOI: 10.1016/j.socnet.2005.05.001. URL: http://www.sciencedirect.com/science/article/pii/S0378873305000353.

[14] Stephen P. Borgatti. "Centrality and network flow". In: *Social Networks* 27.1 (2005), pp. 55–71. ISSN: 03788733. DOI: 10.1016/j.socnet.2004.11.008.

[15] Ahmad Rawashdeh and Anca L Ralescu. "Similarity Measure for Social Networks – A Brief Survey". In: (2012).

[16] Matthias Dehmer, Frank Emmert-Streib, and Jürgen Kilian. "A similarity measure for graphs with low computational complexity". In: *Applied Mathematics and Computation* 182.1 (Nov. 2006), pp. 447–459. ISSN: 00963003. DOI: 10.1016/j.amc.2006.04.006. URL: http://brainmaps.org/pdf/similarity1.pdf%20http://linkinghub.elsevier.com/retrieve/pii/S0096300306003523.

[17] D Koutra, a Parikh, a Ramdas, et al. "Algorithms for Graph Similarity and Subgraph Matching". In: (2011). URL: http://www.cs.cmu.edu/%7B~%7D./aramdas/reports/DBreport.pdf.

[18] Sucheta Soundarajan, T Eliassi-Rad, and Brian Gallagher. "A guide to selecting a network similarity method". In: *Sdm* 1 (2014). DOI: 10.1137/1.9781611973440.118. URL: http://epubs.siam.org/doi/abs/10.1137/1.9781611973440.118.

[19] Laura A. Zager and George C. Verghese. "Graph similarity scoring and matching". In: *Applied Mathematics Letters* 21.1 (2008), pp. 86–94. ISSN: 08939659. DOI: 10.1016/j.aml.2007.01.006.

[20] Gaoxia Wang, Yi Shen, and Enjie Luan. "Measure of centrality based on modularity matrix". In: *Progress in Natural Science* 18.8 (2008), pp. 1043–1047. ISSN: 10020071. DOI: 10.1016/j.pnsc.2008.03.015.

[21] Ron Zass and Amnon Shashua. "Probabilistic graph and hypergraph matching". In: *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2008), pp. –7. ISSN: 1063-6919. DOI: 10.1109/CVPR.2008.4587500.

[22] D. Conet, P. Foggia, C. Sansone, et al. "Thirty Years of Graph Matching in Pattern Recognition". In: *International Journal of Pattern Recognition and Artificial Intelligence* 18.03 (2004), pp. 265–298. ISSN: 0218-0014. DOI: 10.1142/S0218001404003228.

[23] H Bunke. "Recent developments in graph matching". In: *Pattern Recognition, 2000. Proceedings. 15th ...* 2 (2000), pp. 117–124. ISSN: 1051-4651. DOI: 10.1109/ICPR.2000.906030. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=906030$%5Cbackslash$nhttp://ieeexplore.ieee.org/xpls/abs%7B%5C_%7Dall.jsp?arnumber=906030.

[24] F. Ashby and Daniel Ennis. "Similarity measures". In: *Scholarpedia* 2.12 (2007), p. 4116. ISSN: 1941-6016. DOI: 10.4249/scholarpedia.4116. URL: http://www.scholarpedia.org/article/Similarity%7B%5C_%7Dmeasures.

[25] H Yaghi and H Krim. "Probabilistic graph matching by canonical decomposition". In: *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on* (2008), pp. 2368–2371. DOI: 10.1109/ICIP.2008.4712268.

[26] D.a. Spielman. "Spectral Graph Theory and its Applications". In: *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)* 1 (2007), pp. 29–38. ISSN: 0272-5428. DOI: 10.1109/FOCS.2007.56.

[27] Tib??rio S. Caetano, Julian J. McAuley, Li Cheng, et al. "Learning graph matching". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.6 (2009), pp. 1048–1058. ISSN: 01628828. DOI: 10.1109/TPAMI.2009.28. arXiv: 0806.2890v1.

[28] A. Egozi, Y. Keller, and H. Guterman. "A Probabilistic Approach to Spectral Graph Matching". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.1 (2013), pp. 18–27. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2012.51. URL: http://ieeexplore.ieee.org/ielx5/34/6353858/06152128.pdf?tp=%7B%5C&%7Darnumber=6152128%7B%5C&%7Disnumber=6353858$%5Cbackslash$nhttp://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=%7B%5C&%7Darnumber=6152128%7B%5C&%7DsortType=desc%7B%5C_%7Dp%7B%5C_%7DPublication%7B%5C_%7DYear%7B%5C&%7DqueryText=Maximum+Likelihood+Estimation.

[29] Robert a Hanneman and Mark Riddle. "Introduction to Social Network Methods". In: *Riverside, CA: University of California, Riverside. On-line textbook* 46.7 (2005), pp. 5128–30. ISSN: 10744770. DOI: 10.1016/j.socnet.2006.08.002. URL: http://www.faculty.ucr.edu/%7B~%7Dhanneman/nettext/.

[30] Colin McDiarmid and Fiona Skerman. "Modularity in random regular graphs and lattices". In: *Electronic Notes in Discrete Mathematics* 43 (2013), pp. 431–437. ISSN: 15710653. DOI: 10.1016/j.endm.2013.07.063. URL: http://linkinghub.elsevier.com/retrieve/pii/S1571065313002151.

[31] Maciej Komosinski and Marek Kubiak. "Quantitative measure of structural and geometric similarity of 3D morphologies". In: *Complexity* 16.6 (2011), pp. 40–52. ISSN: 10762787. DOI: 10.1002/cplx.20367.

[32] Jérôme Kunegis, Damien Fay, and Christian Bauckhage. "Spectral evolution in dynamic networks". In: *Knowledge and Information Systems* 37.1 (Oct. 2013), pp. 1–36. ISSN: 0219-1377. DOI: 10.1007/s10115-012-0575-9. URL: http://link.springer.com/10.1007/s10115-012-0575-9.

[33] Andries E. Brouwer and Willem H. Haemers. *Spectra of Graphs*. Vol. 1542. Universitext. New York, NY: Springer New York, 2012, pp. 33–36. ISBN: 978-1-4614-1938-9. DOI: 10.1007/978-1-4614-1939-6. arXiv: arXiv:1011.1669v3. URL: http://link.springer.com/10.1007/978-1-4614-1939-6.

[34] Olivier Duchenne, Francis Bach, Kweon In-so, et al. "A Tensor-Based Algorithm for High-Order Graph Matching". In: (2014).

[35] Michael Behrisch, Benjamin Bach, Nathalie Henry Riche, et al. "Matrix Reordering Methods for Table and Network Visualization". In: 35 (2016).

[36] Michael Behrisch, Benjamin Bach, Michael Hund, et al. "Magnostics: Image-based Search of Interesting Matrix Views for Guided Network Exploration". In: *IEEE Transactions on Visualization and Computer Graphics* (2016), pp. 1–1. ISSN: 1077-2626. DOI: 10.1109/TVCG.2016.2598467. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7534849.

[37] Y Park, C.E. Priebe, and D. J. Marchette. *Scan Statistics on Enron Hypergraphs*. 2008. URL: http://www.cis.jhu.edu/%7B~%7Dparky/Enron/.

[38] Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. "Exploring network structure, dynamics, and function using NetworkX". In: *Proceedings of the 7th Python in Science Conference (SciPy2008)*. Pasadena, CA USA, Aug. 2008, pp. 11–15.

[39] Stéfan van der Walt, S. Chris Colbert, and Gaël Varoquaux. "The NumPy Array: A Structure for Efficient Numerical Computation". In: *Computing in Science Engineering* 13.2 (2011), pp. 22–30. DOI: http://dx.doi.org/10.1109/MCSE.2011.37. URL: http://scitation.aip.org/content/aip/journal/cise/13/2/10.1109/MCSE.2011.37.

[40] Eric Jones, Travis Oliphant, Pearu Peterson, et al. *SciPy: Open source scientific tools for Python*. [Online; accessed 2016-08-04]. 2001–. URL: http://www.scipy.org/.

[41] Wes McKinney. "Data Structures for Statistical Computing in Python". In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman. 2010, pp. 51–56.

[42] J D Hunter. "Matplotlib: A 2D graphics environment". In: *Computing In Science & Engineering* 9.3 (2007), pp. 90–95.

[43] Michael. Waskom. *Seaborn: statistical data visualization*. 2012. URL: http://stanford.edu/%7B~%7Dmwaskom/software/seaborn/ (visited on 04/06/2016).

[44] F Pedregosa and G Varoquaux. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine ...* 12 (2011), pp. 2825–2830. arXiv: 1201.0490. URL: http://jmlr.csail.mit.edu/papers/volume12/pedregosa11a/pedregosa11a.pdf.

[45] Brian McFee, Matt McVicar, Colin Raffel, et al. *librosa: 0.4.1*. Oct. 2015. DOI: 10.5281/zenodo.32193. URL: http://dx.doi.org/10.5281/zenodo.32193.

[46] Daniel D. Hickstein, Roman Yurchak, Dhrubajyoti Das, et al. *PyAbel (v0.7): A Python Package for Abel Transforms*. Mar. 2016. DOI: 10.5281/zenodo.47423. URL: http://dx.doi.org/10.5281/zenodo.47423.

[47] Wei Wei and Kathleen M. Carley. "Measuring Temporal Patterns in Dynamic Social Networks". In: *ACM Transactions on Knowledge Discovery from Data* 10.1 (2015), pp. 1–27. ISSN: 15564681. DOI: 10.1145/2749465. URL: http://www.scopus.com/inward/record.url?eid=2-s2.0-84938355062%7B%5C&%7DpartnerID=tZOtx3y1.

[48] William N Anderson. "Eigenvalues of the Laplacian of a graph". In: *Linear and Multilinear Algebra* 18.2 (1985), pp. 141–145. ISSN: 0308-1087. DOI: 10.1080/03081088508817681. URL: http://www.informaworld.com/openurl?genre=article%7B%5C&%7Ddoi=10.1080/03081088508817681%7B%5C&%7Dmagic=crossref.

[49] Leo Katz. "A new status index derived from sociometric analysis". In: *Psychometrika* 18.1 (1953), pp. 39–43. ISSN: 1860-0980. DOI: 10.1007/BF02289026. URL: http://dx.doi.org/10.1007/BF02289026.

[50] K.-I. Goh, B. Kahng, and D. Kim. "Universal Behavior of Load Distribution in Scale-Free Networks". In: *Physical Review Letters* 87.27 (Dec. 2001), p. 278701. ISSN: 0031-9007. DOI: 10.1103/PhysRevLett.87.278701. URL: http://link.aps.org/doi/10.1103/PhysRevLett.87.278701.

[51] M E J Newman. "Scientific collaboration networks. II. Shortest paths, weighted networks, and centrality". In: *Phys. Rev. E* 64.1 (June 2001), p. 16132. DOI: 10.1103/PhysRevE.64.016132. URL: http://link.aps.org/doi/10.1103/PhysRevE.64.016132.

[52] Mathias Johansson. "The Hilbert transform". PhD thesis. 1999, pp. 1–33. URL: http://www.fuchs-braun.com/media/d9140c7b3d5004fbffff8007fffffff0.pdf.

[53] Ming Li and Yimin Zhao. "Seismic Attribute Analysis". In: *Geophysical Exploration Technology* (2014), pp. 103–131. DOI: 10.1016/B978-0-12-410436-5.00005-8. URL: http://linkinghub.elsevier.com/retrieve/pii/B9780124104365000058.

[54] D Subrahmanyam and P H Rao. "Seismic Attributes-A Review". In: ().

[55] D. J. Klein and M. Randić. "Resistance distance". In: *Journal of Mathematical Chemistry* 12.1 (Dec. 1993), pp. 81–95. ISSN: 02599791. DOI: 10.1007/BF01164627. URL: http://link.springer.com/10.1007/BF01164627.

[56] Nathanael Perraudin, Johan Paratte, David Shuman, et al. "GSPBOX: A toolbox for signal processing on graphs". In: (2016).

[57] Amarnath Gupta. *Graph Analytics for Big Data - University of California, San Diego — Coursera*. URL: https://www.coursera.org/learn/big-data-graph-analytics.

[58] Scott L. Miller and Donald Childers. "Power Spectral Density". In: *Probability and Random Processes* (2012), pp. 429–471. DOI: 10.1016/B978-0-12-386981-4.50013-8. URL: http://linkinghub.elsevier.com/retrieve/pii/B9780123869814500138.

[59] Terence Tao. "Fourier Transform". In: *arXiv* (2012), pp. 1–5. ISSN: 00222852. DOI: 10.1006/jmsp.1998.7620.

[60] Fabien Gouyon, François Pachet, and Olivier Delerue. "ON THE USE OF ZERO - CROSSING RATE FOR AN APPLICATION OF CLASSIFICATION OF PERCUSSIVE SOUNDS". In: ().

[61] Jesus David Terrazas Gonzalez and Witold Kinsner. "Zero-crossing analysis of Lévy walks for real-time feature extraction". In: *2016 IEEE International Conference on Electro/Information Technology, 2016 EIT*. 2016. ISBN: 978-1-4673-9985-2. DOI: 10.1109/EIT.2016.7535276.

[62] John M. Grey. "Perceptual effects of spectral modifications on musical timbres". In: *The Journal of the Acoustical Society of America* 63.5 (1978), p. 1493. ISSN: 00014966. DOI: 10.1121/1.381843.

[63] Ed Kragh and Phil Christie. "Seismic repeatability, normalized rms, and predictability". In: *The Leading Edge* 21.7 (July 2002), pp. 640–647. ISSN: 1070-485X. DOI: 10.1190/1.1497316. URL: http://library.seg.org/doi/10.1190/1.1497316.

[64] Florian Wickelmaier. *An introduction to MDS*. Denmark: Aalborg Universitetsforlag, 2003.

[65] L J P Van Der Maaten and G E Hinton. "Visualizing high-dimensional data using t-sne". In: *Journal of Machine Learning Research* 9 (2008), pp. 2579–2605. ISSN: 1532-4435. DOI: 10.1007/s10479-011-0841-3. URL: http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=pubmed%7B%5C&%7Dcmd=Retrieve%7B%5C&%7Ddopt=AbstractPlus%7B%5C&%7Dlist%7B%5C_%7Duids=79114314791487345 48related:VOiAgwMNy20J.

Appendix A

Project Proposal

TABLE OF CONTENTS

1. INTRODUCTION

1.1. TITLE

Analysing the evolution of communication patterns in email data through an extended dynamic network analysis toolkit

1.2. RESEARCH QUESTIONS

How can similarity measures be used to analyse dynamic email networks? How can similarity measures in alternative spaces support the analysis of dynamic networks?

1.3. PURPOSE

The origins of graphs theory can be traced back to Leonhard Euler and his approach to solving the Konigsberg Bridge Problem. This city was located on the Pregel River in Prussia. The river divided this city into 4 distinct areas which included an island all of which were connected by a total of 7 bridges. Euler's representation of this problem of the individual areas as nodes and the bridges as edges is considered one of the first applications of graph theory. [1]

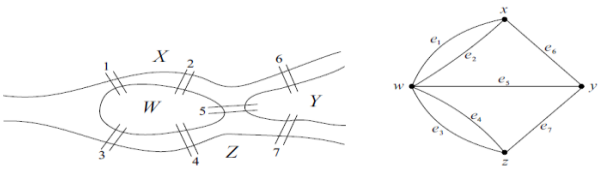


Figure 1: Euler's graphical representation of the Konigsberg Bridge Problem. [1]

Using Euler's insights into this problem in modern time's graphs have become a powerful tool with which to model and analyse communication and information networks. More specifically communities and their evolution in time [2]. The aim of this project is to explore the use of similarity measures to characterise such change in these networks. The viability of such analysis has been well demonstrated in the literature more specifically by [3], who modelled the email data released during the Enron investigation as a network and have made it available freely. Using this dataset [4] propose novel methods for community detection in real world information networks.

We intend to add to the existing body of knowledge regarding graph similarity measures and contribute a comparative and evaluative analysis of such measures on the emerging field of dynamic network analysis.

1.4. OBJECTIVES

The objectives that will help us answer our research question are as follows:

- What are some of the graph similarity measures that have been proposed in the literature?
- How have these been used in a practical context?
- How can such measures be applied to dynamic networks?
- How are such measures evaluated?

The aims of this project can be summarised as follows:

- To provide a fairly comprehensive overview of graph theory as is relevant to the understanding of the derivation of similarity measures.
- Conducting an in depth literature review on the graph similarity measures proposed and that have been demonstrated to be useful in a practical context.
- Compare the utility and performance of these measures on appropriate data
- Explore the viability of developing a novel similarity measure based on Fourier and Hilbert Analysis of networks

1.5. MOTIVATIONS & AIMS

The purpose of this research is twofold: firstly, we demonstrate the practical need for such analysis and secondly to develop a reference document for end user interested in applying such analysis to domains other than our stated domain of email networks.

The issue of graph similarity is critical in the analysis of dynamic networks. This is because when considering such networks an important concern is to understand the evolution of the network through time. Utilising similarity measures represents a very convenient method of achieving this as we can compare the same measure at different points in time to get a sense of network evolution. This is necessary because within our chosen context of email data, these methods can give insight into how communications patterns have evolved over time. Also we can gain insight into communities within such data and their birth, death and evolution. It can also serve as a means of anomaly detection. The set of analysis that such measures can enable are limited by one's creativity. The purpose is to show that there are very practical and real need to address the extension of similarity analysis to dynamic network analysis. We present many methods proposed in the literature and will also build on recent work on the application of Signal Processing methods on graphs to assess the potential of novel similarity methods.

The aim is to provide a comprehensive summary of graph similarity measures that can be applied to dynamic networks. We will explore their theoretical basis and methods of evaluation. In addition to the above it is hoped that the material developed as part of this project will provide a useful base reference to anyone interested in applying graph analytics to solve problems.

Also, we will explore the extent to which the measures that we encounter are implemented in common analysis packages such as IPython [5], NetworkX [6], Gephi [7] and Cytoscape [8] among others. The key point to make regarding tools selected here are that they are open source so they are freely available and the source code is viewable. This will allow if the need arises for extensions to be made to the code and shared freely.

2. CRITICAL CONTEXT

2.1. LIMITATIONS OF TRADITIONAL SNA

In Social Network Analysis (SNA), traditionally bounded networks are considered with maybe 2 or 3 connection or link types such as friendship or advice between a node types such as people sometimes another node type such as events are also considered together. [9]

If we consider more critically the interactions possible within our problem context of email networks we can have email networks within an organisation which are bounded and also with other organisations, clients and stakeholders and then the network does become unbounded. These networks can then be thought of as a higher order networks and as [9] notes many tools developed for simpler networks do not scale well to increased network size and complexity and in some cases experience degradation through increased susceptibility to Type 1 and Type 2 errors.
The dynamics in these networks can arise from different processes depending on the context of the problem. Natural evolutionary processes would be learning, births, deaths and ageing Others could be as a result of intervention measures such removal or addition of nodes i.e. removing those who lead the system, communities forming or disintegrating. The data associated with such systems are also often incomplete and contain errors which make the process of analysis and evaluation of these systems. [10]

Analysis approaches that go beyond traditional SNA and link analysis are therefore necessary. Within the context of such dynamic networks analysis can be performed to identify of key individuals, locating hidden groups and estimate performance. The data analysis process on such networks then involve: [9], [10]

- ⊕ Relationship identification among nodes
- ⊕ Network structure characterisation
- ⊕ Locating the elite within the network
- ⊕ Identifying points of vulnerability
- ⊕ Comparing networks

The approaches that enables effective analysis of such dynamic networks and help quantify their evolution over time is the motivation for this research.

2.2. GRAPH TERMINILOGY

A graph, G can be described as a triple which consists of a set of edges E (G), a set of vertices or nodes V

(G) and a relationship that connects the vertices to these edges. Finite graphs are those that have V and E as a finite set. Simple graphs are those that have no loops or multiple edges. A path is simple graph in which the vertices can be ordered where two vertices can be adjacent only if they are consecutively ordered. A cycle is defined as a simple graph where the vertices can be cyclically ordered such that two vertices are adjacent only if they are consecutive in cyclical ordering. A subgraph can be thought of cycles and paths within a larger graph, where the edge relations between the subgraph and the large graph are the same. [1]

2.3. DYNAMIC NETWORK ANALYSIS AS AN EXTENSION TO SNA

Dynamic network analysis (DNA) aims to extend the methods, tools and techniques used in traditional Social Network Analysis (SNA) to the analysis of networks which are able to handle big dynamic multi-mode, multi-link networks with varying levels of uncertainty. Dynamic networks also allow for probabilistic connection between nodes. [9]

In [9] DNA was explored within the context of terrorism networks. Here an additional layer of complexity is added by the fact that an act of measurement changes its properties and this change propagates through the network and its state changes. Another key point is that the nodes in this network have the ability to learn. So the nodes themselves can be thought of being probabilistic compared to the more static nature of SNA nodes.

In a DNA representation systems can be represented as relational data. This relational data structure can lends flexibility in defining multiple node types defined as multi-modal, have various types of connections among such nodes called multi-plex. The underlying attributes of both node, edges and the data change over time hence the dynamic part. [10]

In [9] the key advances that allow for the analysis of such dynamic networks are identified as:

- ⊕ The meta matrix
- ⊕ Probabilistic edges between nodes
- ⊕ Combining social networks with cognitive science and multi-agent systems

The Meta Matrix

The Meta matrix is a method used in operations research and organisational management that seeks to represent the entity and class relationships as a collection of networks. In the DNA context this translates as a multi-mode, multi-plex approach to representing systems. Therefore the Meta matrix can contain a social network, a membership network and knowledge network and allow us to explore and analyse the connections between them. [9]– [12]

Probabilistic Ties

The ties or connections in the Meta matrix are probabilistic with various factors affecting their probability. This allows for inclusion of the observers uncertainty and the likelihood that the tie is present at the time of observation. These probabilities themselves and their temporal evolution maybe estimated by the Bayesian methods, cognitive inferencing and models of social and cognitive change. [9]

Multi Agent Network Models

As previously discussed the SNA treatment of nodes as static agents unable to learn is insufficient when dynamic networks are concerned. In DNA the nodes are able to take actions, learn from experience and alter their networks as a result. Some social and cognitive processes that influence the agent's interactions are relative similarity, relative expertise and co-workers. The dynamic behaviour of the network emerges from these interactions and experience a shared evolution. [9]

We briefly discuss some of the more common measures associated with networks which relate to their global and local properties. These will be important when we discuss similarity because one of the ways to assess similarity is to consider snapshots of a network attribute at different time intervals.

2.4. NETWORK MEASURES: LOCAL

Centrality measures are a fundamental statistic in network analysis. In [13] two paradigms of centrality definitions are suggested. One is the means based definition of centrality or the graph theoretic and the other is the ends-based definition which is a dynamic model based view that focuses on the outcome for the nodes in a network where there is flows across the nodes [14]. However, both approaches agree that this measure is a node level property.

In both formulations of centrality measures noted above we characterise centrality measures as follows: [13]

⊕ Volume based measures – degree like centrality
⊕ Length based measures – closeness like centrality
⊕ Medial measures – betweenness like centrality

Volume and length based measures are what are called radial measures because they analyse walks that emanate from or terminate with a given node. Medial measures on the other hand are based on position so how many times does one encounter a node while trying to reach other nodes in the network.

Degree Centrality
Degree Centrality is a special case of the k–path centrality that counts the all the paths of a length, k that originate from a given node. Degree Centrality is then defined as the number of edges incident on a node. Which translates to summing all the rows of the adjacency matrix of a network. [13], [14]

Closeness Centrality
Closeness Centrality is the graph theoretic distance or the geodetic distance from a given to all the other nodes in a network. [13], [14]

Betweenness Centrality
Betweenness centrality counts the number of times that a certain node, x needs to pass another node, y to get to another node, z through the shortest path between them. [13], [14]

Eigenvector Centrality
The Eigenvector Centrality is defined as the principal eigenvector of the adjacency matrix of a network. It captures the intuition that nodes that have high eigenvector centrality scores are likely to be close to other nodes which themselves have high values for this measure. [13], [14]

We have mentioned some of the most popular centrality measures but there are numerous other variations mentioned in the literature and are beyond the scope of this work the interested reader is referred to the following starter references. [15]– [18]

2.5. NETWORK MEASURES: GLOBAL

Network Size
This can be defined by counting the number of nodes in a network. [19]

Density
This is defined as the total number of edges divided by the total number of possible edges. [19]

Diameter
The diameter of a network is the maximum geodesic distance between two nodes.

Modularity
The modularity function finds partitions within the graph where a large proportion of the edges fall entirely within that partition and biases against those partitions that have too few or unequal sized parts. [20]

3. APPROACHES

The purpose of detailing the some of the key properties and attributes are they can be used as to derive similarity measures on graphs. These measures will form the basis of our exploration of existing and new similarity measures for DNA.

The problem of graph similarity or graph matching then becomes one of finding the equivalence of two graphs with potentially different number of nodes and edges and returning a measure within [0, 1] that captures their similarity or dissimilarity. [18], [21]– [27]

The key idea of graph matching in the context of dynamic networks can be summarised as finding a subgraph or an attribute that we can compare between two time instances. For example, if we consider the Degree Centrality of a network at time step 0 and then again calculate this measure at time step 1 we can apply a similarity measure on this attribute to quantify the change within the network. This can be done by means of a distance metric such as cosine similarity and others are possible.
Therefore the analysis methodology can be generalised as follows:

⊕ For network X at time step, t extract some attribute
⊕ At time step, t +1 extract the same attribute
⊕ Perform similarity analysis on attribute
⊕ Track the change in the similarity metric over time through some control process.

The evaluation of the change in metrics over time will be done through a statistical control process as suggested in [28]. This is a concept that comes from quality engineering and it essentially involves calculating a statistic from a sequence of measurements of a random process and then comparing it so some control limit. This process translates to:

- ⊕ Calculating a cumulative sum control chart which is very good for detecting small changes in mean over time
- ⊕ Calculating a z-score for each time step (Eq. 1)
- ⊕ Construction of two charts to detect increase and decrease in the metric as shown.

$$Z_i = (x_i - \mu_0)/\sigma$$
$$C_t^+ = \max\{0,\ Z_t - k + C_{t-1}^+\}$$
$$C_t^- = \max\{0,\ -Z_t - k + C_{t-1}^-\}$$

Equation 1: Control chart to detect increase and decrease in metric over time

In the following section will present a brief treatment of the subject of similarity.

3.1. SIMILARITY METHODS

The problem of finding similarity between nodes which are similar in a network can be thought of as a problem of finding a set of nodes which are similar to a given node according to some attributes which are represented as connections [25].

Similarity in a networks is classified as being of structural, content or keyword based The Structural similarity or link based similarity considers the similarity of links between the nodes in the graph e.g. Cosine, Jaccard, Hub Promoted and Hub Depressed Index etc. Content similarity considers the attributes of the node in the graph. For example on a social network this could be birth dates or hobbies of individuals. Keyword similarity aims to find similarity based on nodes representing word collections. Global Structural Similarity can be classified as being:

- ⊕ local vs. global
- ⊕ parameter-free vs. parameter-dependent
- ⊕ node-dependent vs. path- dependent

The global structural measures aim to measure node similarity compared to the whole network. We will call them intra network similarity measures. [25]

Inter network similarity measures are described in [22], [24], [26], [27], [29]– [32]. These measures are classified by [21] into three categories 1) Distance Based 2) Feature Based and 3) Probabilistic.

Distance based approach

The distance based approach is perhaps the earliest of the methods encountered which is based on edit distance [32]. Essentially this boils down to finding a sequence of operations such as deletion, insertion, or substitution minimising some cost function that will turn one graph into another. These involve detection and comparison of the graph isomorphism, subgraph isomorphism and maximum common subgraph detection utilising the edit distance. Although these methods are guaranteed to converge to an optimal solution their exponential complexity makes them unsuitable for large graphs.

Feature Based approach

We have already hinted at the feature based approach above in the Al-Qaeda example above. But more formally this involves calculation of a network attribute such as degree, closeness, betweenness, and/or eigenvector centrality for the graphs and then applying a similarity measure on them that will characterise their similarity or dissimilarity. This has the benefit of being scalable to very large networks as the aggregated statistics are much smaller than the network themselves.

A taxonomy of the methods that have been proposed to solve these problems are shown in Fig 3. We include this for illustrative purposes and their discussion will form part of the extended review performed for the final report.

Probabilistic Approach

The methods that fall under this approach in the literature are vast. Some approaches under the probabilistic framework for graph matching are discussed here [29], [33]– [35]. But simply stated these methods define a probability distribution over mappings or graph embedding's [31], [34]. Graph embeddings are graphs whose nodes correspond to distinct points on a plane and the edges represents relationships connecting these points. The matching algorithm is strongly dependent upon the geometric information attached to the graphs [31], [34].

Graph matching allows for recovering point correspondences. In [29] the authors show that assuming that the assignment matrix that represents these correspondences are statistically independent the high order matching problem can be represented by a Kronecker product matrix. Also they show that

that a high order tensor affinity tensor can be marginalised into a one dimensional vector of probabilities. This probability vector is then updated by projection to a vector assignment space and then minimizing a distance measure (Bregman measure) [33]. Spectral Methods involve looking at the spectra of a graph which are defined by the eigenvalues of the adjacency matrix. In [36] the high order matching is expressed a tensor Eigen decomposition and applied to point matching using some similarity measure.

Visual Representation

More recently, the authors in [37] have proposed visualising dynamic networks and characterising change by visualising the adjacency matrix of these networks as a matrix cube. Representing the adjacency matrix as a stack of cubes rather than node link diagrams is found to be a much more useful paradigm for analysis of dynamic networks especially when these networks are dense.

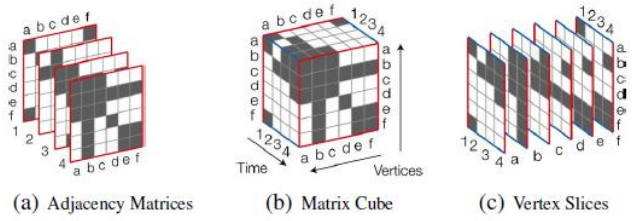


Figure 2: The matrix cube view of dynamic networks that allow for visual characterisation of network evolution over time. [37]

We will explore some representative methods from these classes of methods and assess their utility for DNA.

3.1. ORIGINAL IDEAS

Given the author's background as a Geophysicist in performing seismic data analysis, there was a strong interest in exploring potential ideas that could be imported from seismic analysis to the analysis of dynamic networks. The reasons for this is because seismic data is essentially a time series on which signal processing approaches are utilised. Since there is a time component to these networks the attributes of these networks could be viewed as a time series and some of these methods can then be tested for effectiveness.

Firstly, for visualisation of dynamic network attributes we suggest something similar to how well log analysis is performed in reservoir characterisation of hydrocarbon reservoirs shown in Fig 3.

From Fig 6, we propose to develop a dynamic network attribute panel that allows analysis of selected network attributes over time such as Degree, Closeness, and Density etc. [38]

The Coherence attribute is multi trace similarity of the seismic waveform that is used for event detection and is very useful when the amplitude of the signal is low compared to the noise. For DNA we can explore the viability of developing a Coherence for network similarity measures or network attributes that also allows for event detection shown in Fig 7. More details regarding this method can be found here [39].

Some recent work has focused on the application of signal processing methods to graphs. In [40] a generalisation of the Fourier Transform and its associated methods such as convolutions to graphs. In [41], [42] transforms and tomograms for graph application are discussed. An interesting application is that of the Radon transform commonly used for noise removal in seismic data analysis, which the authors in [42] generalise to arbitrary pairs of non-commuting operators. These are positive bilinear transforms with a probabilistic interpretation and provide a full characterization of the signals while being robust to noise. These developments although very interesting do not have mainstream implementations in software packages and must be conducted in a more ad-hoc manner. This is a potential limitation of their evaluation in our study.

However, we present some preliminary analysis on a synthetic geographical threshold graph model which places n nodes uniformly at random in a rectangular domain with 500 nodes. We extract the Degree Centrality and then apply the Fourier and Hilbert Transform to it to extract its periodic features and complex trace information.

From Fig 3, 4 we note that these transforms serve as a useful visualisation tool of network attributes. The Fourier Transform picks out the maximum degree centrality while the Double Hilbert Transform shows some outliers.

A correlation between the Fourier or Hilbert transforms of a measure would give a similarity value shown in Fig 8. The autocorrelation of the measures with themselves would give an indication of the self-similarity of the network.

From the autocorrelation functions we see that these have the potential to highlight anomalies. We will explore these measures over time to assess their value in DNA.

In summary, our analysis approach will focus on network measures calculated through some of the methods we identified previously. We will consider the temporal component as an important aspect of network characterisation.

The key consideration of the analysis will be to use already implemented method and not be distracted by extensive development work as this is not the purpose of the research. Hence, we identify promising areas to focus on but if time constraints do not permit or the software packages are found to be lacking we will include them as future areas of research.

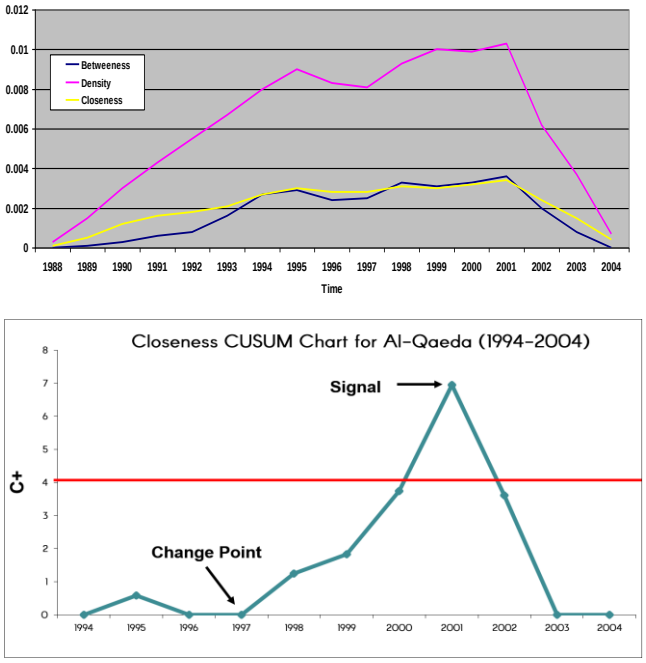


Figure 5: From [28] (top) shows the application of this method on the Al Qaeda network for the Betweenness, Density and Closeness measures. Bottom modified from [28] shows the change in Closeness Centrality for the network through the control chart. The control limit is shown in red

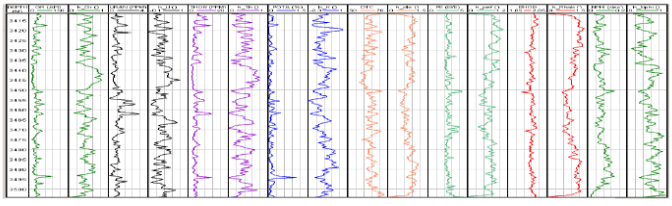


Figure 6: Well Log panel showing the different measurements typically recorded while drilling. These attributes are then used for characterisation.

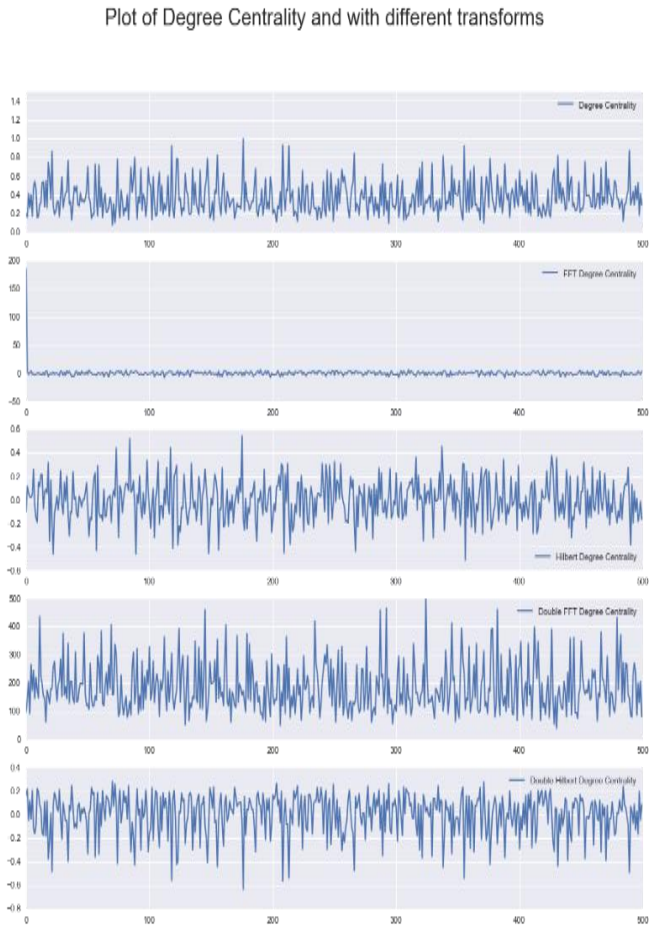


Figure 3: Plot of (1st) Degree Centrality (DC), (2nd) Fourier Transform of DC, (3rd) Hilbert Transform of DC, (4th) Double Fourier Transform of DC, (5th) Double Hilbert Transform of DC

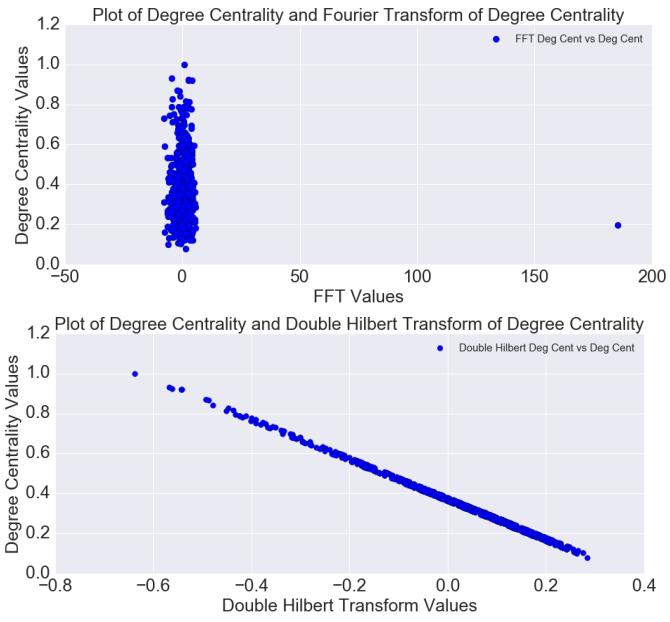


Figure 4: (top) Scatter plot of Fourier Transform of DC and DC (bottom) Double Hilbert Transform of DC and DC.

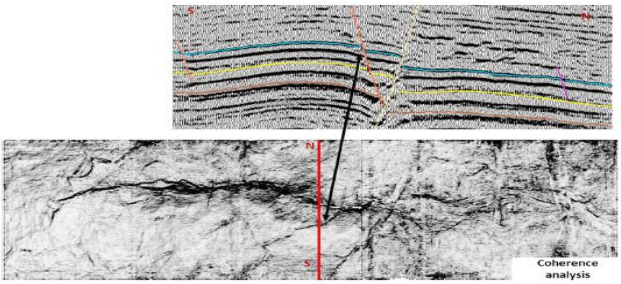


Figure 7: Example of a coherence attribute cube for a seismic volume. Coherence is essentially a multi trace similarity measure used for event detection. [39]

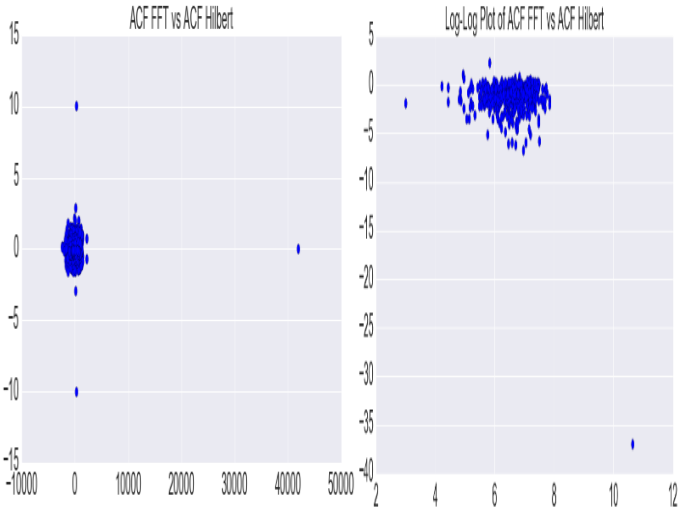


Figure 8: Plot of autocorrelation functions of the Fourier and Hilbert Transforms of Degree Centrality. (Left) Log-Log Plot of ACF's.

3.2. EVALUATION STRATEGIES

We have presented an array of methods and techniques proposed in the lecture and our own novel ideas based on the literature. Evaluation of such measures and their interpretation will form a key part of the research. The interpretation will be especially important when we consider our suggestions of measures of similarity in alternative spaces such as the Fourier and Hilbert spaces. These Fourier Transform extracts periodic features from a time series by extracting its frequency while the Hilbert transform extends the signal to the complex plane. What these mean in the context of our measures will be explored.

Therefore the evaluation strategy will be as follows:

- For the dataset used search the literature to see what other authors have found especially with regards to interesting features, signals and/or structures
- Then evaluate whether our measures are able to find these features and how difficult or easy they are in comparison with the original measures
- If such signals are not found in the literature, we can look for them in the data through other analysis and use our measures to confirm alternatively we can insert a test signal in the data and see how well the measures are able to find them.

This is deemed the most practical way to evaluate the research given the short time horizon. Other methods such as involving groups or surveys to assess participant's interactions with these analytical tools would not be practical for this study.

3.1. DATA SOURCES

This research will utilise open and publicly available data so the results are reproducible. To this end we identify some sources of network data available to us:

- Enron email data : http://www.cs.cmu.edu/~enron/
- Stanford Large Network Collection: http://snap.stanford.edu/data/#email
- UCI Network Data Repository : https://networkdata.ics.uci.edu/resources.php
- Open Dynamic Network Data: http://www.aviz.fr/~bbach/opendynamicnetworks

4. WORK PLAN

The research will be conducted between the periods of June – September 2016. A detailed work plan is highlighted in the Gantt chart attached with this report.

5. ETHICS

Since this is predominantly a research and analysis task on open data there are ethical concerns. Completed ethics questionnaire is also appended with this report.

6. RISK REGISTER

The standard types of technical and non-technical risks apply to this project as any other. A completed risk register is appended with this report.

7. REFERENCES

[1] A. Dickson, "Introduction to Graph Theory The Konigsberg Bridge Problem," *October*, no. October, pp. 1–15, 2006.

[2] A. Chapanond, M. S. Krishnamoorthy, and B. Yener, "Graph theoretic and spectral analysis of Enron email data," *Comput. Math. Organ. Theory*, vol. 11, no. 3, pp. 265–281, 2005.

[3] B. Klimt and Y. Yang, "Introducing the Enron Corpus," *Mach. Learn.*, vol. stitutep1, p. wwceasccaers2004168, 2004.

[4] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters," *Internet Math.*, vol. 6, no. 1, pp. 29–123, 2011.

[5] F. Pérez and B. E. Granger, "IPython: a System for Interactive Scientific Computing," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 21–29, May 2007.

[6] H. Aric, D. Schult, and S. Pieter, "NetworkX." .

[7] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: An Open Source Software for Exploring and Manipulating Networks," *Third International AAAI Conference on Weblogs and Social Media*, 2009. [Online]. Available: http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154 \npapers2://publication/uuid/CCEBC82E-0D18-4FFC-91EC-6E4A7F1A1972. [Accessed: 25-Mar-2016].

[8] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker, "Cytoscape: A software Environment for integrated models of biomolecular interaction networks," *Genome Res.*, vol. 13, no. 11, pp. 2498–2504, Nov. 2003.

[9] K. M. Carley, "Dynamic Network Analysis," no. September, 2001.

[10] K. M. Carley, J. Diesner, J. Reminga, and M. Tsvetovat, "Toward an interoperable dynamic network analysis toolkit," *Decis. Support Syst.*, vol. 43, no. 4, pp. 1324–1347, 2007.

[11] K. M. Carley, "Ora: Organization risk analyzer," *Science (80-.).*, no. January, p. 49, 2004.

[12] T. A. B. Snijders, "Introduction To Dynamic Social Network Analysis," *BRiMS 2015*, no. November, 2015.

[13] S. P. Borgatti and M. G. Everett, "A Graph-theoretic perspective on centrality," *Soc. Networks*, vol. 28, no. 4, pp. 466–484, 2006.

[14] S. P. Borgatti, "Centrality and network flow," *Soc. Networks*, vol. 27, no. 1, pp. 55–71, 2005.

[15] G. Costantini, S. Epskamp, D. Borsboom, M. Perugini, R. Mõttus, L. J. Waldorp, and A. O. J. Cramer, "State of the aRt personality research: A tutorial on network analysis of personality data in R," *J. Res. Pers.*, vol. 54, pp. 13–29, 2015.

[16] K. R. Harrison, M. Ventresca, and B. M. Ombuki-Berman, "A meta-analysis of centrality measures for comparing and generating complex network models," *J. Comput. Sci.*, 2015.

[17] R. Kaur and S. Singh, "A survey of data mining and social network analysis based anomaly detection techniques," *Egypt. Informatics J.*, 2015.

[18] R. a. Hanneman and M. Riddle, "Measures of similarity and structural equivalence," *Introduction to social network methods*, 2011. [Online]. Available: http://www.faculty.ucr.edu/~hanneman/nettext/C13_ Structural_Equivalence.html. [Accessed: 26-Mar-2016].

[19] R. a Hanneman and M. Riddle, "Introduction to Social Network Methods," *Riverside, CA Univ. California, Riverside. On-line Textb.*, vol. 46, no. 7, pp. 5128–30, 2005.

[20] C. McDiarmid and F. Skerman, "Modularity in random regular graphs and lattices," *Electron. Notes Discret. Math.*, vol. 43, pp. 431–437, 2013.

[21] F. Ashby and D. Ennis, "Similarity measures," *Scholarpedia*, vol. 2, no. 12, p. 4116, 2007.

[22] M. Dehmer, F. Emmert-Streib, and J. Kilian, "A similarity measure for graphs with low computational complexity," *Appl. Math. Comput.*, vol. 182, no. 1, pp. 447–459, Nov. 2006.

[23] M. Komosinski and M. Kubiak, "Quantitative measure of structural and geometric similarity of 3D morphologies," *Complexity*, vol. 16, no. 6, pp. 40–52, 2011.

[24] D. Koutra, a Parikh, a Ramdas, and J. Xiang, "Algorithms for Graph Similarity and Subgraph Matching," 2011.

[25] A. Rawashdeh and A. L. Ralescu, "Similarity Measure for Social Networks – A Brief Survey," 2012.

[26] S. Soundarajan, T. Eliassi-Rad, and B. Gallagher, "A guide to selecting a network similarity method," *Sdm*, no. 1, 2014.

[27] L. A. Zager and G. C. Verghese, "Graph similarity scoring and matching," *Appl. Math. Lett.*, vol. 21, no. 1, pp. 86–94, 2008.

[28] "Dynamic Social Network Analysis of Al-Qaeda." .

[29] R. Zass and A. Shashua, "Probabilistic graph and hypergraph matching," *26th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR*, pp. 0–7, 2008.

[30] C. Wang, L. Wang, and L. Liu, "Improving graph matching via density maximization," *Proc. IEEE Int. Conf. Comput. Vis.*, pp. 3424–3431, 2013.

[31] D. Conet, P. Foggia, C. Sansone, and M. Vento, "Thirty Years of Graph Matching in Pattern Recognition," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 18, no. 03, pp. 265–298, 2004.

[32] H. Bunke, "Recent developments in graph matching," *Pattern Recognition, 2000. Proceedings. 15th ...*, vol. 2, pp. 117–124, 2000.

[33] A. Egozi, Y. Keller, and H. Guterman, "A Probabilistic Approach to Spectral Graph Matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 18–27, 2013.

[34] T. S. Caetano, J. J. McAuley, L. Cheng, Q. V. Le, and A. J. Smola, "Learning graph matching," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 6, pp. 1048–1058, 2009.

[35] H. Yaghi and H. Krim, "Probabilistic graph matching by canonical decomposition," *Image Process. 2008. ICIP 2008. 15th IEEE Int. Conf.*, pp. 2368–2371, 2008.

[36] O. Duchenne, F. Bach, K. In-so, J. Ponce, O. Duchenne, F. Bach, K. In-so, J. Ponce, A. T. Algorithm, O. Duchenne, F. Bach, I. Kweon, and J. Ponce, "A Tensor-Based Algorithm for High-Order Graph Matching," 2014.

[37] B. Bach, E. Pietriga, and J.-D. Fekete, "Visualizing dynamic networks with matrix cubes," *32nd Annu. ACM Conf. Hum. Factors Comput. Syst. CHI 2014*, pp. 877–886, 2014.

[38] L. Aliouane, S. Ouadfeul, and A. Boudella, "Well-Logs Data Processing Using the Fractal Analysis and Neural Network," 2012.

[39] C. Outline, "Seismic Coherence Technique," pp. 63–74, 2014.

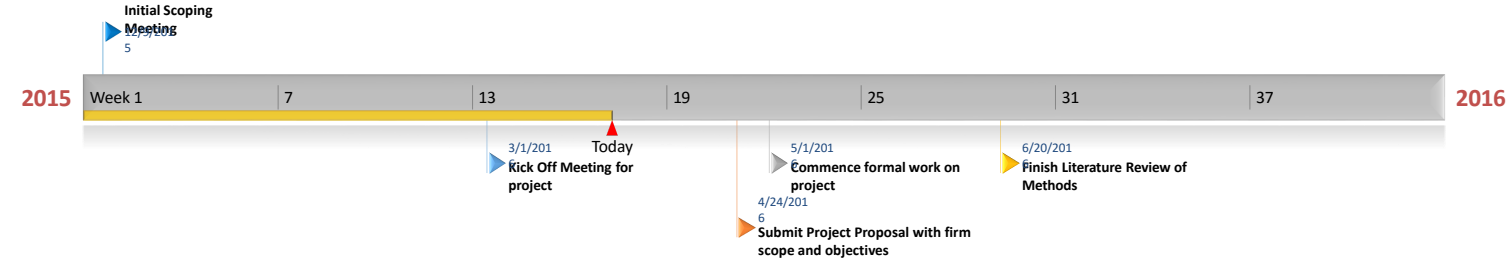
[40] M. Begú, "Fourier Analysis on Graphs."

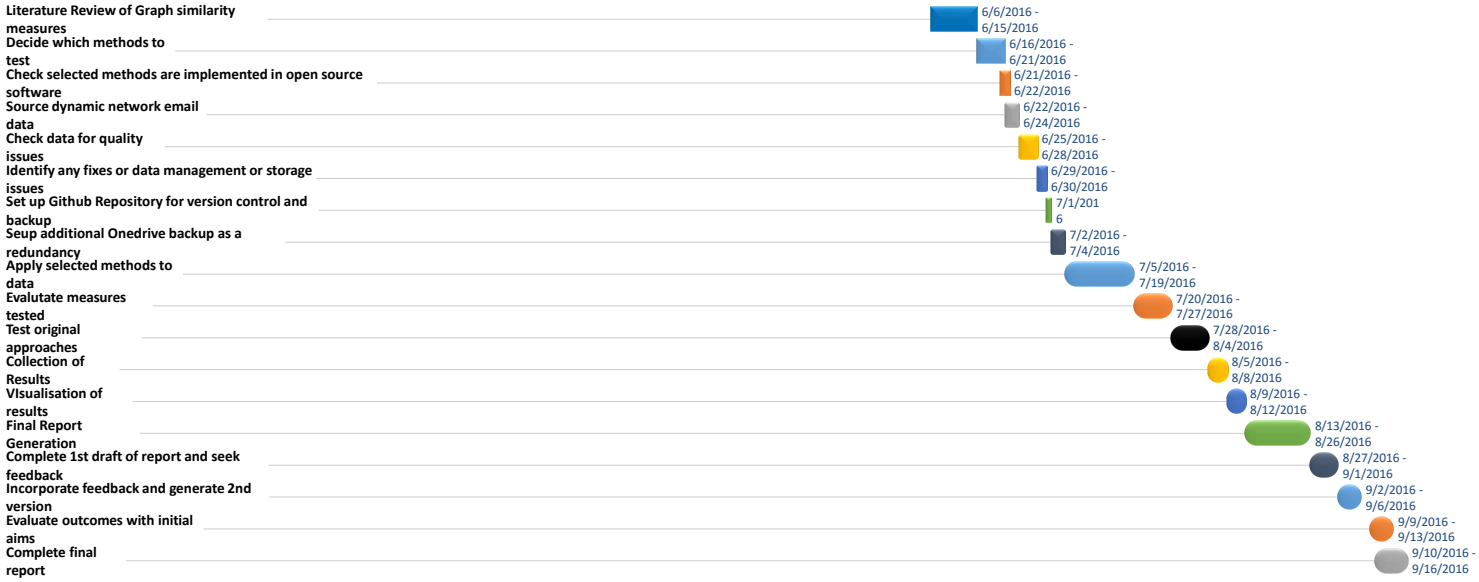
[41] M. A. Man'ko, R. V. Mendes, and V. I. Man'ko, "Tomograms and other transforms: a unified view," *J. Phys. A. Math. Gen.*, vol. 34, no. 40, pp. 8321–8332, 2001.

[42] A. Sandryhaila and J. M. F. Moura, "Discrete Signal Processing on Graphs," *IEEE Trans. Signal Process.*, vol. 61, no. 7, pp. 1644–1656, Apr. 2013.

Appendix B

Project Gantt Chart


Initial Scoping Meeting
5
2015
Week 1
7
13
19
25
31
37
2016
Today
3/1/201
Kick Off Meeting for project
5/1/201
Commence formal work on project
6/20/201
Finish Literature Review of Methods
4/24/201
6
Submit Project Proposal with firm scope and objectives


Literature Review of Graph similarity measures
6/6/2016 - 6/15/2016
Decide which methods to test
6/16/2016 - 6/21/2016
Check selected methods are implemented in open source software
6/21/2016 - 6/22/2016
Source dynamic network email data
6/22/2016 - 6/24/2016
Check data for quality issues
6/25/2016 - 6/28/2016
Identify any fixes or data management or storage issues
6/29/2016 - 6/30/2016
Set up Github Repository for version control and backup
7/1/201 6
Seup additional Onedrive backup as a redundancy
7/2/2016 - 7/4/2016
Apply selected methods to data
7/5/2016 - 7/19/2016
Evalutate measures tested
7/20/2016 - 7/27/2016
Test original approaches
7/28/2016 - 8/4/2016
Collection of Results
8/5/2016 - 8/8/2016
VIsualisation of results
8/9/2016 - 8/12/2016
Final Report Generation
8/13/2016 - 8/26/2016
Complete 1st draft of report and seek feedback
8/27/2016 - 9/1/2016
Incorporate feedback and generate 2nd version
9/2/2016 - 9/6/2016
Evaluate outcomes with initial aims
9/9/2016 - 9/13/2016
Complete final report
9/10/2016 - 9/16/2016

Appendix C

Project Test List

This shows the different stages for the analysis process and an overview of the decisions made. This also helps with versioning and creates a provenance for the analytical process so that it could be replicated easily.

Table C.1: Test List for Analysis

Dynamic Network Analysis of Enron Email Network Data: Test List

Test Number	Description
00	First look at data and initial checks
01	Break into networks but with cumulative yearly aggregation and look at centrality and other network statistics
02	Break data into monthly and yearly aggregations without cumulative aggregation and choose benchmark measures. Derive Seismic attributes from Laplacian matrix to evaluate their plausibility
03	Deriving Seismic attributes from 3 graph matrices
04	Comparison of attributes from 3 graph matrices to benchmark measures
05	Consolidate analysis and remove analysis that
06	Derivation of Music attributes as a proof of concept and consolidated analysis for report.
06.1	Derivation of Seismic and Music attributes from 3 graph matrices
06.2	Comparison of all attributes from the 3 graph matrices using SNR and Entropy

Appendix D

Jupyter Notebook: Python Code and Results

To enable full reproducibility and transparency of analysis and results all code and results presented in this study are attached in this section. The Jupyter Notebook format is convenient because it enables placing of code and its results together in a powerful way and can be exported to a wide variety of formats.

This shows clearly how every single attribute has been implemented with code, which libraries have been used and what the outputs were. This should enable practitioners to copy the code and use it in their own analysis as well as be able to reproduce any result presented here.

DNA_06

September 4, 2016

1 Table of Contents

2 Introduction

Dynamic Network Analysis of Enron Email Network Data.

I use the Enron email network data from John Hopkins which has time, sender and receiver pair format data.

Here I consolidate earlier analysis to make presentation of the final results easier and more readable.

From the JHU data, I have done the following in Excel: - The first column represents seconds elapsed since 1 January 1970, so I convert this in to days - I then add these days to the date to

get time stamps for all nodes - From the timestamps, I extract the year field - The network can be partitioned by the required level of granularity

```
In [1]: import pandas as pd
        import numpy as np
        import networkx as nx
        import seaborn as sns
        import scipy as sc
        import random
        from scipy.signal import *
        from numpy.linalg import *
        from sklearn.decomposition import *
        from sklearn.preprocessing import *
        import librosa
        import abel
        #plotting parameters
        %matplotlib inline
        sns.set(style="whitegrid", color_codes=True, context='paper')
```

```
In [221]: from matplotlib import rcParams
          rcParams['font.family'] = 'serif'
          rcParams['font.sans-serif'] = ['CMU Serif']
```

```
In [3]: import matplotlib.pyplot as plt
```

```
In [222]: plt.rc('axes', grid=False, titlesize='large', labelsize='medium',labelwei
          plt.rc('lines', linewidth=4)
          plt.rc('figure', figsize = (8,6),titlesize='large',titleweight='black')
          plt.rc('font', weight='medium', size=9)
          plt.rc('grid',linewidth=3)
```

```
In [5]: sns.palplot(sns.cubehelix_palette(10,hue=0.3, reverse=True, rot=-0.55, dark
```

```
In [6]: sns.set_palette(sns.cubehelix_palette(10, hue=0.3, reverse=True, rot=-0.55,
```

```
In [7]: data = pd.read_excel("../Data/data 03.2.xlsx")
```

```
In [8]: years = sorted(set(data.year))
        years = years[1:]
        years
```

```
Out[8]: [1998, 1999, 2000, 2001, 2002]
```

```
In [9]: months = ['Nov98', 'Dec98', 'jan_99', 'feb_99', 'mar_99', 'apr_99', 'may_99
                  'nov_99', 'dec_99', 'jan_2k', 'feb_2k', 'mar_2k', 'apr_2k', 'may_
                  'nov_2k', 'dec_2k', 'jan_2k1', 'feb_2k1', 'mar_2k1', 'apr_2k1', '
                  'oct_2k1', 'nov_2k1', 'dec_2k1', 'jan_2k2', 'feb_2k2', 'mar_2k2',

In [10]: df_98 = data[data.year==years[0]]
         df_99 = data[data.year==years[1]]
         df_2k = data[data.year==years[2]]
         df_2k1 = data[data.year==years[3]]
         df_2k2 = data[data.year==years[4]]

In [11]: def create_graph(df):
             tmp = df.values[:,1:3]
             G= nx.Graph()
             G = nx.from_edgelist(tmp)

             return G

In [12]: Gt0 = create_graph(df_98)
         Gt1 = create_graph(df_99)
         Gt2 = create_graph(df_2k)
         Gt3 = create_graph(df_2k1)
         Gt4 = create_graph(df_2k2)

In [13]: nov_98 = df_98[df_98.month==11]
         dec_98= df_98[df_98.month==12]

         G_nov98 = create_graph(nov_98)
         G_dec98 = create_graph(dec_98)

In [14]: jan_99=df_99[df_99.month==1]
         feb_99=df_99[df_99.month==2]
         mar_99=df_99[df_99.month==3]
         apr_99=df_99[df_99.month==4]
         may_99=df_99[df_99.month==5]
         jun_99=df_99[df_99.month==6]
         jul_99=df_99[df_99.month==7]
         aug_99=df_99[df_99.month==8]
         sep_99=df_99[df_99.month==9]
         oct_99=df_99[df_99.month==10]
         nov_99=df_99[df_99.month==11]
         dec_99=df_99[df_99.month==12]


         G_jan_99=create_graph(jan_99)
         G_feb_99=create_graph(feb_99)
         G_mar_99=create_graph(mar_99)
         G_apr_99=create_graph(apr_99)
         G_may_99=create_graph(may_99)
```

```
G_jun_99=create_graph(jun_99)
G_jul_99=create_graph(jul_99)
G_aug_99=create_graph(aug_99)
G_sep_99=create_graph(sep_99)
G_oct_99=create_graph(oct_99)
G_nov_99=create_graph(nov_99)
G_dec_99=create_graph(dec_99)
```

In [15]:
```
jan_2k=df_2k[df_2k.month==1]
feb_2k=df_2k[df_2k.month==2]
mar_2k=df_2k[df_2k.month==3]
apr_2k=df_2k[df_2k.month==4]
may_2k=df_2k[df_2k.month==5]
jun_2k=df_2k[df_2k.month==6]
jul_2k=df_2k[df_2k.month==7]
aug_2k=df_2k[df_2k.month==8]
sep_2k=df_2k[df_2k.month==9]
oct_2k=df_2k[df_2k.month==10]
nov_2k=df_2k[df_2k.month==11]
dec_2k=df_2k[df_2k.month==12]
```

```
G_jan_2k=create_graph(jan_2k)
G_feb_2k=create_graph(feb_2k)
G_mar_2k=create_graph(mar_2k)
G_apr_2k=create_graph(apr_2k)
G_may_2k=create_graph(may_2k)
G_jun_2k=create_graph(jun_2k)
G_jul_2k=create_graph(jul_2k)
G_aug_2k=create_graph(aug_2k)
G_sep_2k=create_graph(sep_2k)
G_oct_2k=create_graph(oct_2k)
G_nov_2k=create_graph(nov_2k)
G_dec_2k=create_graph(dec_2k)
```

In [16]:
```
jan_2k1=df_2k1[df_2k1.month==1]
feb_2k1=df_2k1[df_2k1.month==2]
mar_2k1=df_2k1[df_2k1.month==3]
apr_2k1=df_2k1[df_2k1.month==4]
may_2k1=df_2k1[df_2k1.month==5]
jun_2k1=df_2k1[df_2k1.month==6]
jul_2k1=df_2k1[df_2k1.month==7]
aug_2k1=df_2k1[df_2k1.month==8]
sep_2k1=df_2k1[df_2k1.month==9]
oct_2k1=df_2k1[df_2k1.month==10]
nov_2k1=df_2k1[df_2k1.month==11]
dec_2k1=df_2k1[df_2k1.month==12]
```

```
G_jan_2k1=create_graph(jan_2k1)
G_feb_2k1=create_graph(feb_2k1)
G_mar_2k1=create_graph(mar_2k1)
G_apr_2k1=create_graph(apr_2k1)
G_may_2k1=create_graph(may_2k1)
G_jun_2k1=create_graph(jun_2k1)
G_jul_2k1=create_graph(jul_2k1)
G_aug_2k1=create_graph(aug_2k1)
G_sep_2k1=create_graph(sep_2k1)
G_oct_2k1=create_graph(oct_2k1)
G_nov_2k1=create_graph(nov_2k1)
G_dec_2k1=create_graph(dec_2k1)
```

In [17]:
```
jan_2k2=df_2k2[df_2k2.month==1]
feb_2k2=df_2k2[df_2k2.month==2]
mar_2k2=df_2k2[df_2k2.month==3]
apr_2k2=df_2k2[df_2k2.month==4]
may_2k2=df_2k2[df_2k2.month==5]
jun_2k2=df_2k2[df_2k2.month==6]
jul_2k2=df_2k2[df_2k2.month==7]
aug_2k2=df_2k2[df_2k2.month==8]
sep_2k2=df_2k2[df_2k2.month==9]
oct_2k2=df_2k2[df_2k2.month==10]
nov_2k2=df_2k2[df_2k2.month==11]
dec_2k2=df_2k2[df_2k2.month==12]
```

```
G_jan_2k2=create_graph(jan_2k2)
G_feb_2k2=create_graph(feb_2k2)
G_mar_2k2=create_graph(mar_2k2)
G_apr_2k2=create_graph(apr_2k2)
G_may_2k2=create_graph(may_2k2)
G_jun_2k2=create_graph(jun_2k2)
G_jul_2k2=create_graph(jul_2k2)
G_aug_2k2=create_graph(aug_2k2)
G_sep_2k2=create_graph(sep_2k2)
G_oct_2k2=create_graph(oct_2k2)
G_nov_2k2=create_graph(nov_2k2)
G_dec_2k2=create_graph(dec_2k2)
```

In [18]:
```
all_year_G  =tuple([Gt0,Gt1,Gt2,Gt3,Gt4])
```

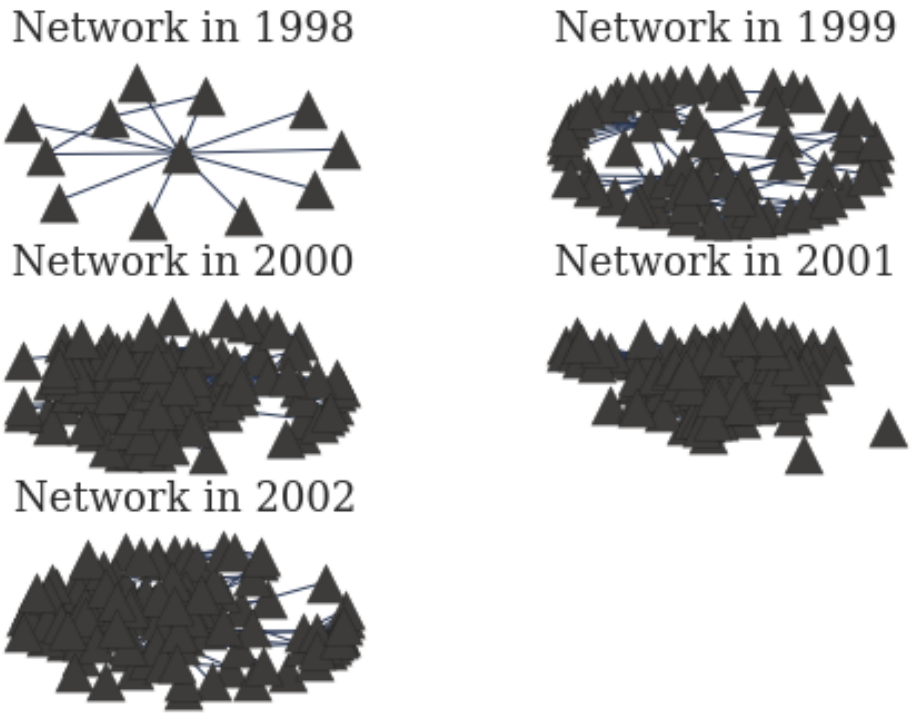
In [19]:
```
all_month_G = tuple([G_nov98,G_dec98,G_jan_99,G_feb_99,G_mar_99,G_apr_99,G
                     G_nov_99,G_dec_99,G_jan_2k,G_feb_2k,G_mar_2k,G_apr_2k,G_may_
                     G_oct_2k,G_nov_2k,G_dec_2k,G_jan_2k1,G_feb_2k1,G_mar_2k1,G_a
                     G_aug_2k1,G_sep_2k1,G_oct_2k1,G_nov_2k1,G_dec_2k1,G_jan_2k2,
```

3 Network Visualisation

- Node Link Diagram
- Audio Waveform
- Matrix Visualisation

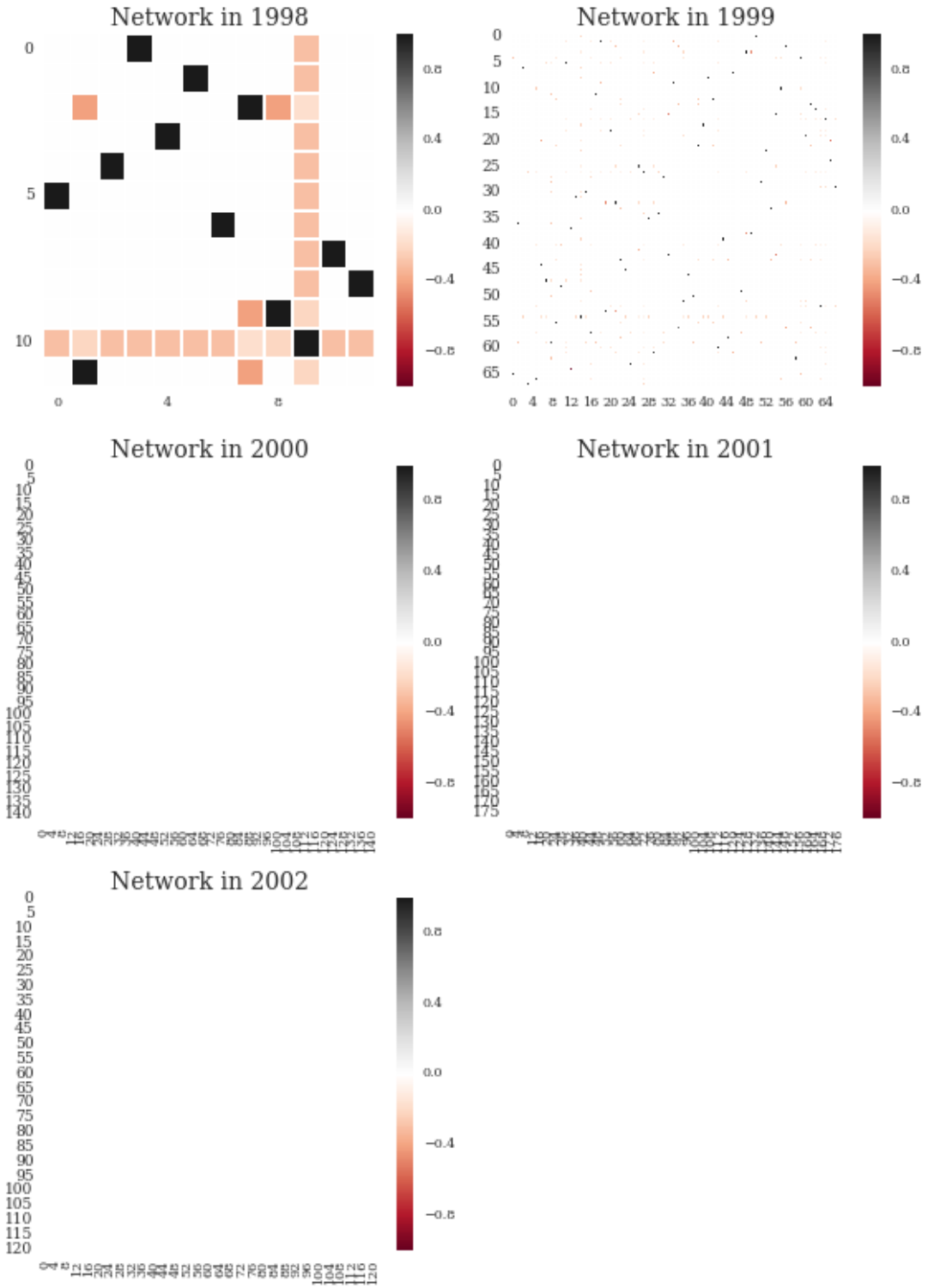
3.1 Yearly Networks

```
In [111]: plt.figure(figsize=(8,11))
          for i in range(len(all_year_G)):
              plt.subplot(6,2,i+1)
              nx.draw_spring(all_year_G[i],  node_color='#3D3C3A',node_shape='^',ed
              plt.title("Network in "  + str(years[i]), fontsize=14)
          plt.savefig('images/yearly_net.png')
```

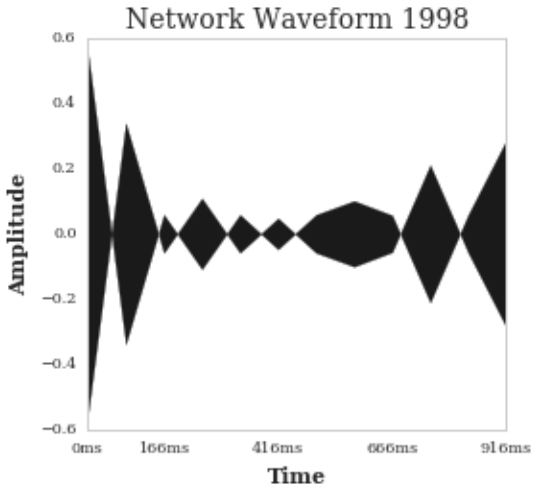


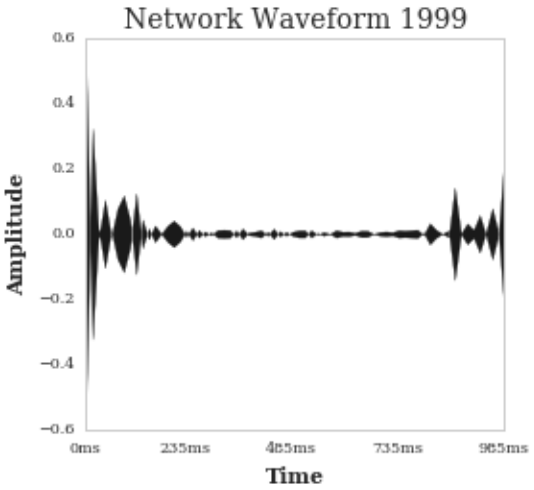

```
In [112]: plt.figure(figsize=(8,11))
          for i in range(len(all_year_G)):
              m = nx.normalized_laplacian_matrix(all_year_G[i]).todense()
              g = sns.clustermap(m)
              plt.close()
              ind = g.dendrogram_row.reordered_ind
              plt.subplot(3,2,i+1)
              sns.heatmap(m[ind][ind], cmap='RdGy', linewidths=1, xticklabels=4,yti
```

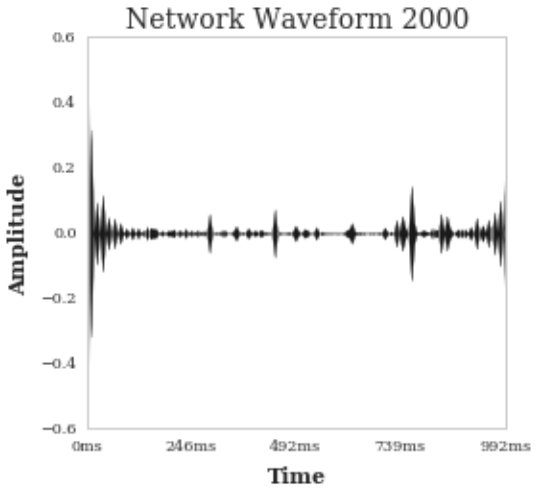
```
        plt.title("Network in "  + str(years[i]), fontsize=14)
        plt.yticks(fontsize=9, rotation=360)
        plt.tight_layout()
    plt.savefig('images/yearly_net_mat.png')
```

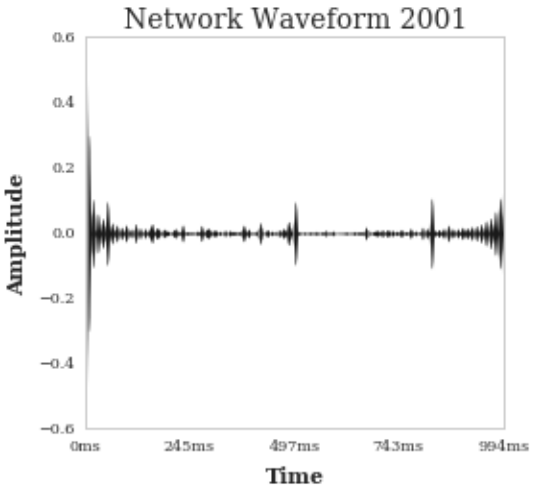


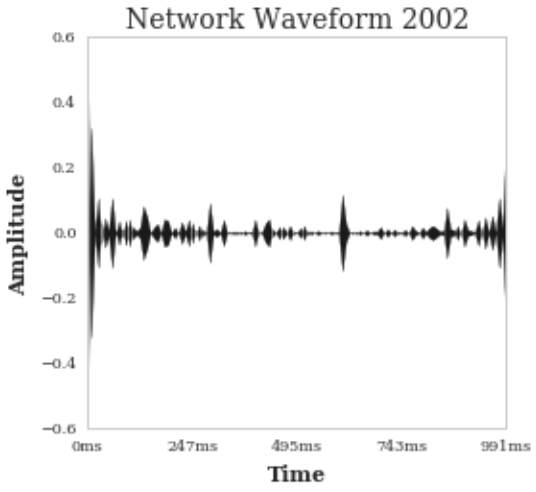
```
In [122]: plt.figure(figsize=(8,11))
          for i in range(len(all_year_G)):
              f = sc.fftpack.rfft(nx.normalized_laplacian_matrix(all_year_G[i]).toc
              fs = f.shape[0]

              plt.subplot(3,2,i+1)
              librosa.display.waveplot(f, fs,color='k')
              plt.title("Network Waveform " + str(years[i]), fontsize=14)
              plt.xlabel("Time", fontsize=11)
              plt.ylabel("Amplitude", fontsize=11)
              plt.grid(False)
              plt.tight_layout()
          plt.savefig('images/yearly_net_audio.png')
```


Network Waveform 1998
Amplitude
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0ms
166ms
416ms
666ms
916ms
Time


Network Waveform 1999
Amplitude
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0ms
235ms
485ms
735ms
985ms
Time

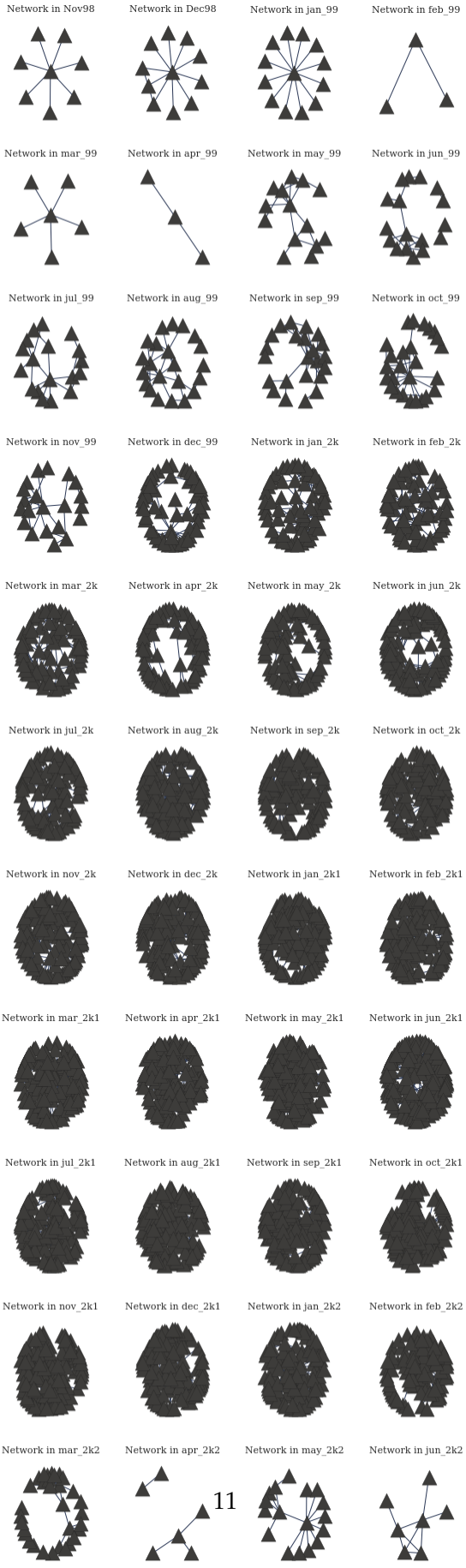

Network Waveform 2000
Amplitude
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0ms
246ms
492ms
739ms
992ms
Time


Network Waveform 2001
Amplitude
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0ms
245ms
497ms
743ms
994ms
Time

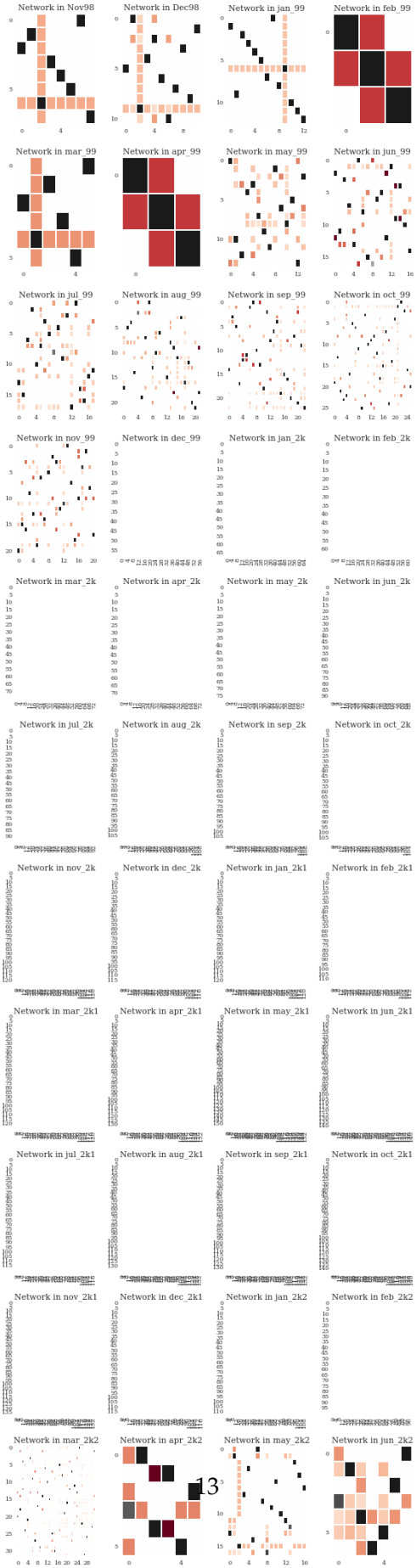

Network Waveform 2002
Amplitude
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0ms
247ms
495ms
743ms
991ms
Time

3.2 Monthly networks

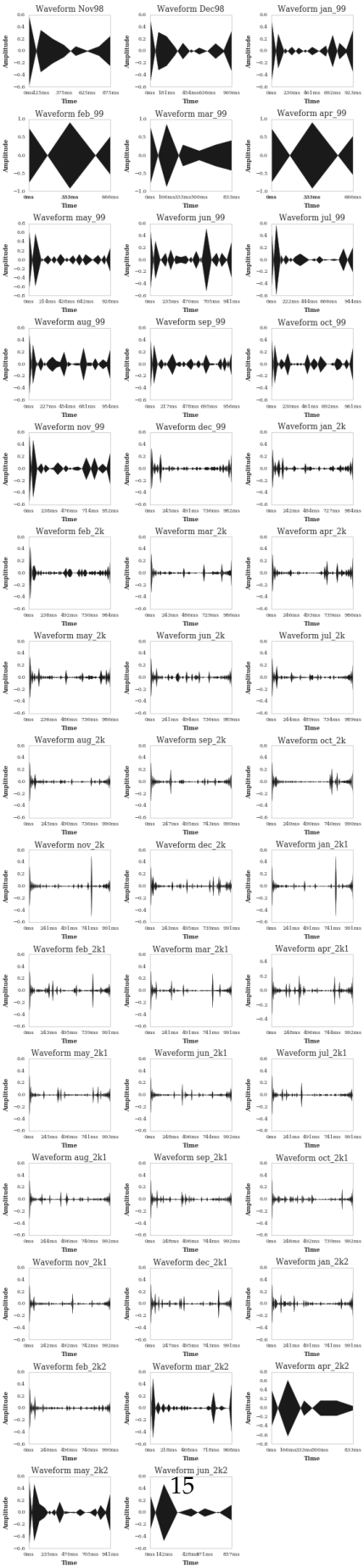
```
In [121]: plt.figure(figsize=(8,28))
          for i in range(len(all_month_G)):
              plt.subplot(12,4, i+1)
              nx.draw_spring(all_month_G[i],  node_color='#3D3C3A',node_shape='^',e
              plt.title("Network in "  + str(months[i]), fontsize=11)
              plt.tight_layout()
          plt.savefig('images/mth_net_nodelink.png')
```


Network in Nov98
Network in Dec98
Network in jan_99
Network in feb_99
Network in mar_99
Network in apr_99
Network in may_99
Network in jun_99
Network in jul_99
Network in aug_99
Network in sep_99
Network in oct_99
Network in nov_99
Network in dec_99
Network in jan_2k
Network in feb_2k
Network in mar_2k
Network in apr_2k
Network in may_2k
Network in jun_2k
Network in jul_2k
Network in aug_2k
Network in sep_2k
Network in oct_2k
Network in nov_2k
Network in dec_2k
Network in jan_2k1
Network in feb_2k1
Network in mar_2k1
Network in apr_2k1
Network in may_2k1
Network in jun_2k1
Network in jul_2k1
Network in aug_2k1
Network in sep_2k1
Network in oct_2k1
Network in nov_2k1
Network in dec_2k1
Network in jan_2k2
Network in feb_2k2
Network in mar_2k2
Network in apr_2k2
Network in may_2k2
Network in jun_2k2

```
In [124]: plt.figure(figsize=(8,32))
          for i in range(len(all_month_G)):
              m = nx.normalized_laplacian_matrix(all_month_G[i]).todense()
              g = sns.clustermap(m)
              plt.close()
              ind = g.dendrogram_row.reordered_ind
              plt.subplot(12, 4, i+1)
              sns.heatmap(m[ind][ind], cmap='RdGy', linewidths=1, xticklabels=4,yti
              plt.title("Network in "  + str(months[i]), fontsize=11)
              plt.yticks(rotation=360)
              plt.tight_layout()
          plt.savefig('images/mth_net_mat.png')
```


Network in Nov98
Network in Dec98
Network in jan_99
Network in feb_99
Network in mar_99
Network in apr_99
Network in may_99
Network in jun_99
Network in jul_99
Network in aug_99
Network in sep_99
Network in oct_99
Network in nov_99
Network in dec_99
Network in jan_2k
Network in feb_2k
Network in mar_2k
Network in apr_2k
Network in may_2k
Network in jun_2k
Network in jul_2k
Network in aug_2k
Network in sep_2k
Network in oct_2k
Network in nov_2k
Network in dec_2k
Network in jan_2k1
Network in feb_2k1
Network in mar_2k1
Network in apr_2k1
Network in may_2k1
Network in jun_2k1
Network in jul_2k1
Network in aug_2k1
Network in sep_2k1
Network in oct_2k1
Network in nov_2k1
Network in dec_2k1
Network in jan_2k2
Network in feb_2k2
Network in mar_2k2
Network in apr_2k2
Network in may_2k2
Network in jun_2k2

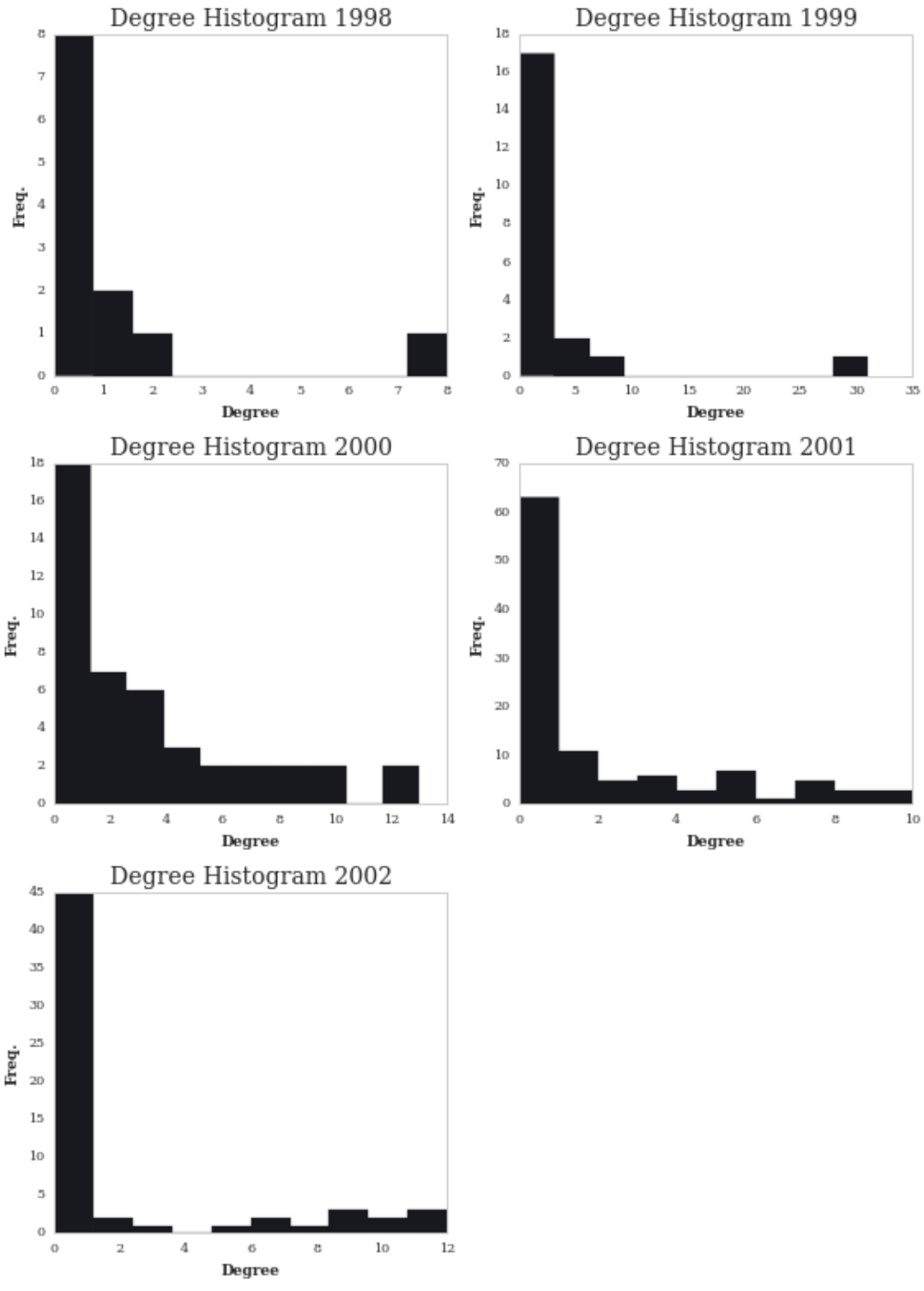
```
In [135]: plt.figure(figsize=(8,36))

          for i in range(len(all_month_G)):
              f = sc.fftpack.rfft(nx.normalized_laplacian_matrix(all_month_G[i]).to
              fs = f.shape[0]

              plt.subplot(16, 3, i+1)
              librosa.display.waveplot(f, fs,color='k')
              plt.title("Waveform " + str(months[i]), fontsize=12)

              plt.xlabel("Time")
              plt.ylabel("Amplitude")
              plt.grid(False)
              plt.tight_layout()
          plt.savefig('images/mth_net_audio.png')
```


Waveform Nov98
Waveform Dec98
Waveform jan_99
Waveform feb_99
Waveform mar_99
Waveform apr_99
Waveform may_99
Waveform jun_99
Waveform jul_99
Waveform aug_99
Waveform sep_99
Waveform oct_99
Waveform nov_99
Waveform dec_99
Waveform jan_2k
Waveform feb_2k
Waveform mar_2k
Waveform apr_2k
Waveform may_2k
Waveform jun_2k
Waveform jul_2k
Waveform aug_2k
Waveform sep_2k
Waveform oct_2k
Waveform nov_2k
Waveform dec_2k
Waveform jan_2k1
Waveform feb_2k1
Waveform mar_2k1
Waveform apr_2k1
Waveform may_2k1
Waveform jun_2k1
Waveform jul_2k1
Waveform aug_2k1
Waveform sep_2k1
Waveform oct_2k1
Waveform nov_2k1
Waveform dec_2k1
Waveform jan_2k2
Waveform feb_2k2
Waveform mar_2k2
Waveform apr_2k2
Waveform may_2k2
Waveform jun_2k2
Amplitude
Time

4 Exploratory Analysis

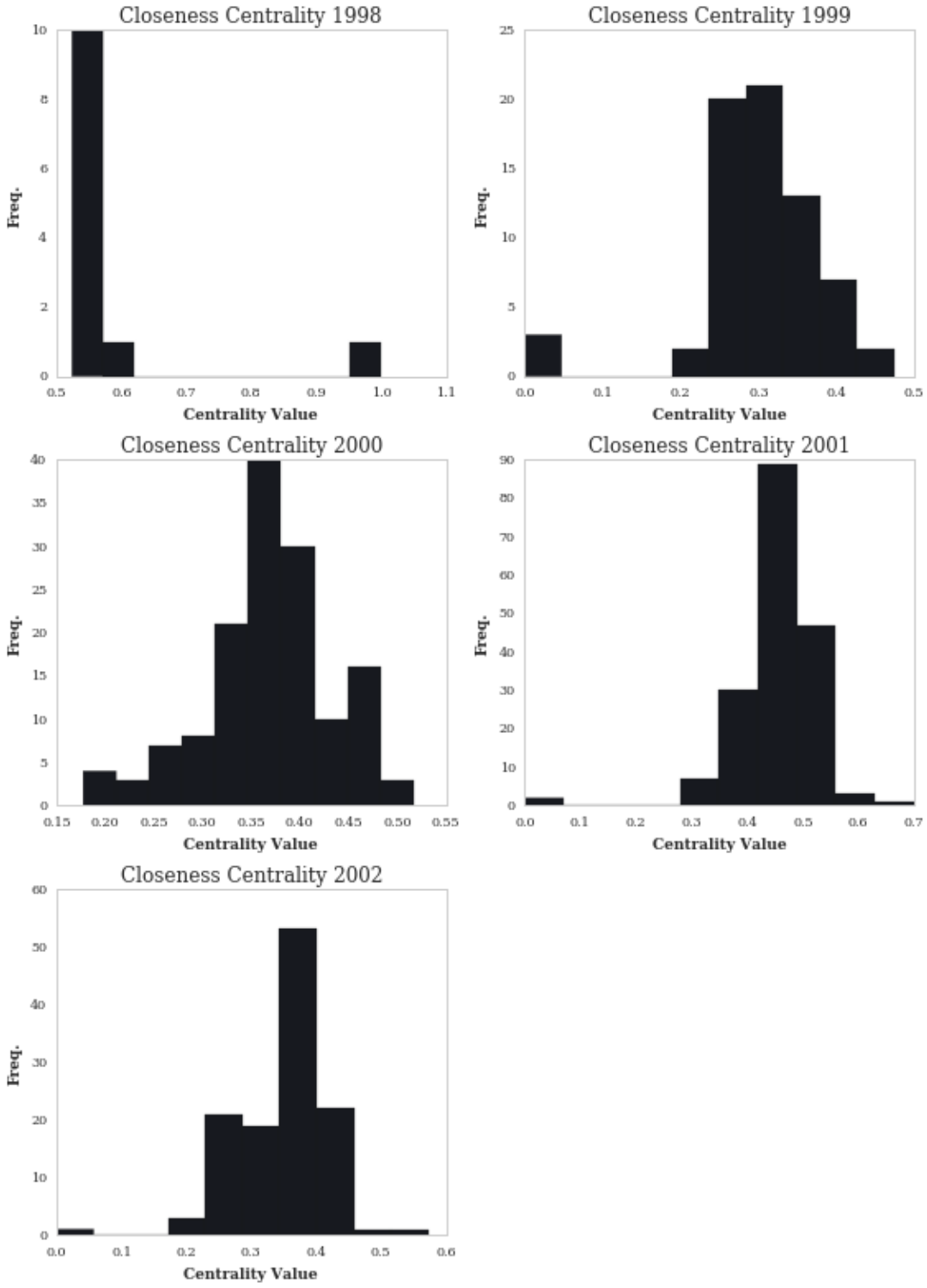
4.1 Yearly Networks

```
In [126]: plt.figure(figsize=(8,11))
          for i in range(len(all_year_G)):
              deg = nx.degree_histogram(all_year_G[i])
              plt.subplot(3, 2, i+1)
              plt.hist(deg)
              plt.title("Degree Histogram " + str(years[i]), fontsize=14)
              plt.xlabel("Degree")
              plt.ylabel("Freq.")
              plt.grid(False)
              plt.tight_layout()
          plt.savefig('images/year_deghist.png')
```



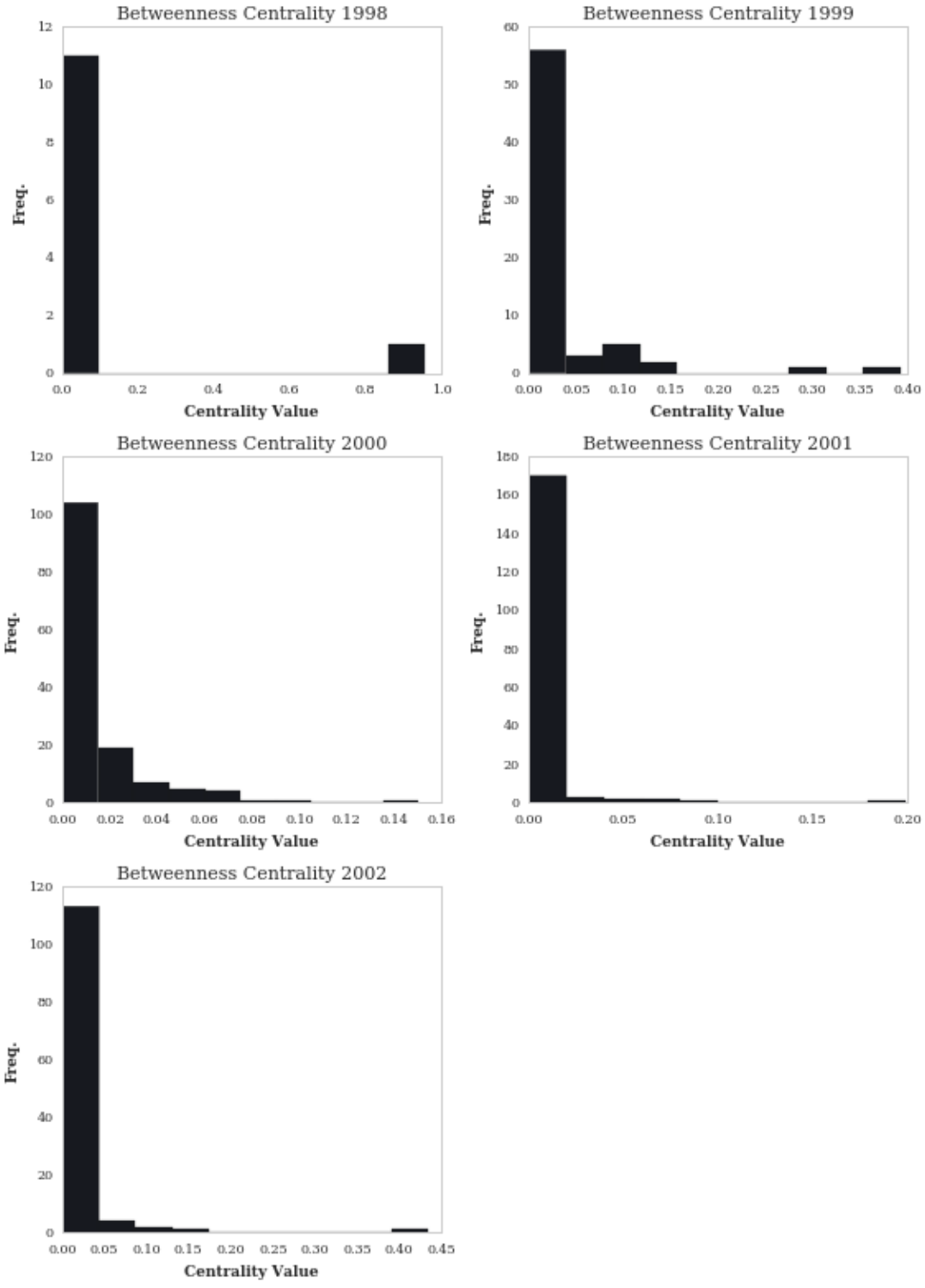

```
In [136]: plt.figure(figsize=(8,11))
          for i in range(len(all_year_G)):
```

```
    deg = nx.closeness_centrality(all_year_G[i]).values()
    deg = sorted(deg)
    plt.subplot(3, 2, i+1)
    plt.hist(deg)
    plt.title("Closeness Centrality " + str(years[i]), fontsize=12)
    plt.xlabel("Centrality Value")
    plt.ylabel("Freq.")
    plt.grid(False)
    plt.tight_layout()
plt.savefig('images/year_clohist.png')
```



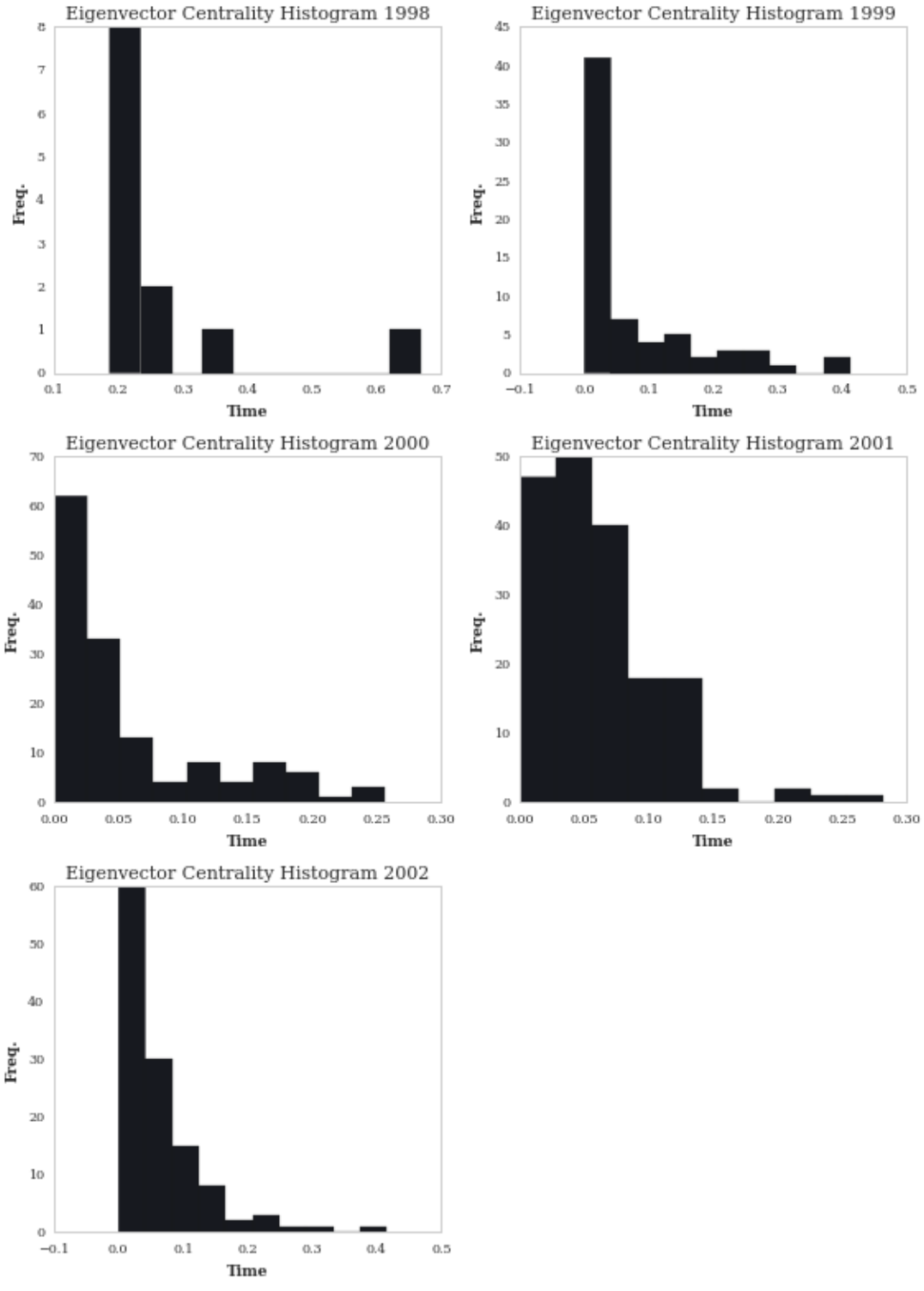

```
In [137]: plt.figure(figsize=(8,11))
          for i in range(len(all_year_G)):
```

```
        deg = nx.betweenness_centrality(all_year_G[i]).values()
        deg = sorted(deg)
        plt.subplot(3, 2, i+1)
        plt.hist(deg)
        plt.title("Betweenness Centrality " + str(years[i]), fontsize=11)
        plt.xlabel("Centrality Value")
        plt.ylabel("Freq.")
        plt.grid(False)
        plt.tight_layout()
    plt.savefig('images/year_bethist.png')
```



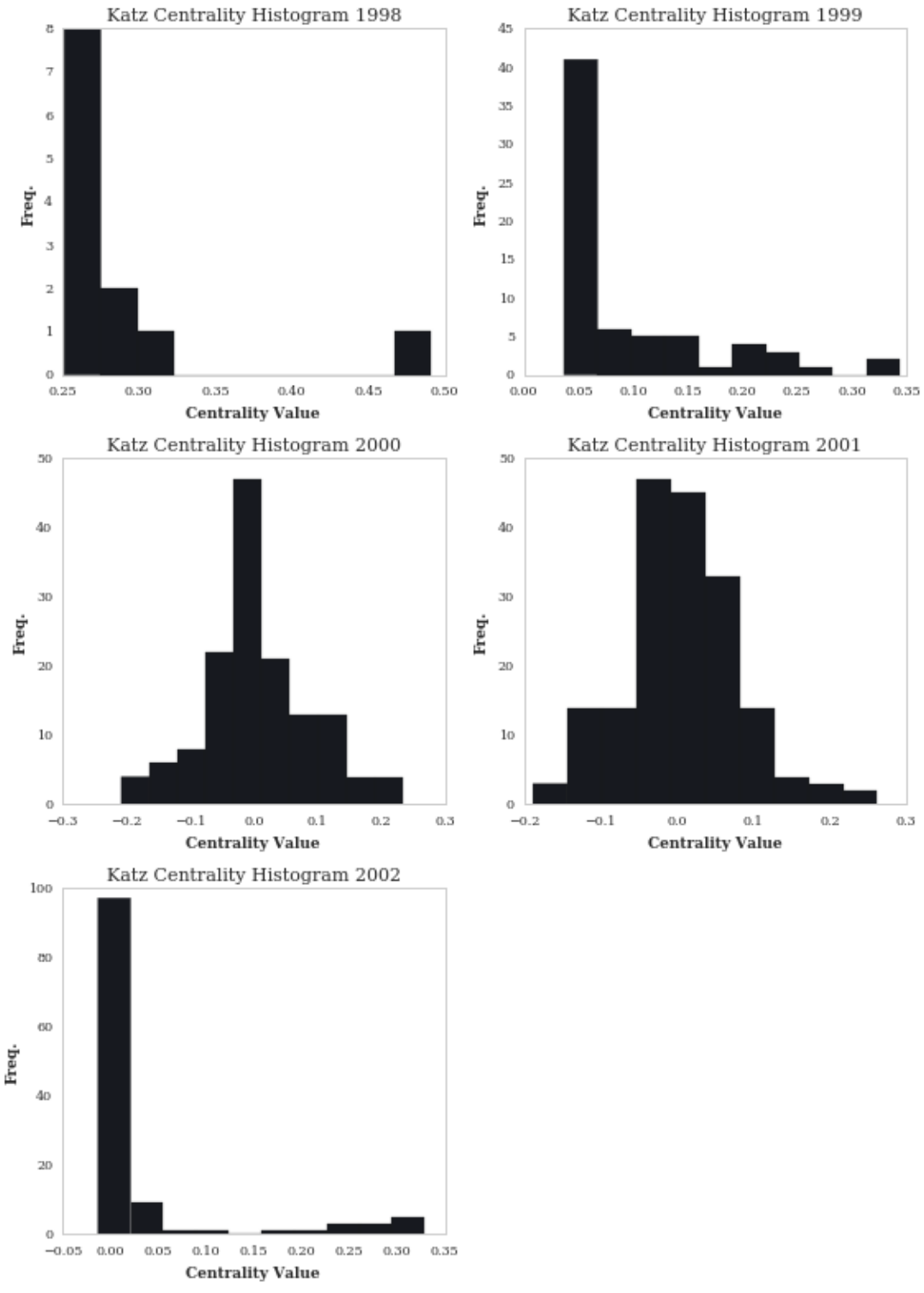

```
In [138]: plt.figure(figsize=(8,11))
          for i in range(len(all_year_G)):
```

```
        deg = nx.eigenvector_centrality_numpy(all_year_G[i]).values()
        deg = sorted(deg)
        plt.subplot(3, 2, i+1)
        plt.hist(deg)
        plt.title("Eigenvector Centrality Histogram " + str(years[i]), fontsi
        plt.xlabel("Time")
        plt.ylabel("Freq.")
        plt.grid(False)
        plt.tight_layout()
    plt.savefig('images/year_eighist.png')
```



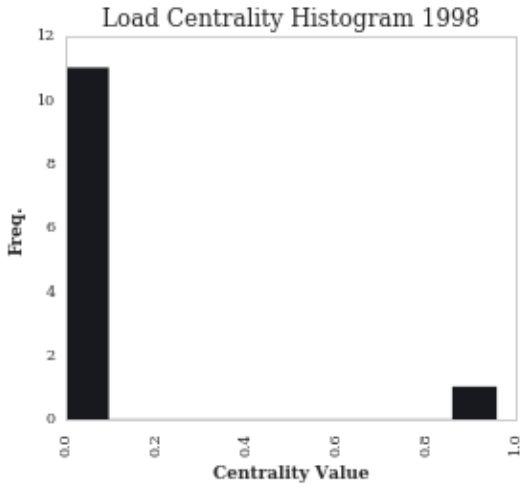

```
In [139]: plt.figure(figsize=(8,11))
          for i in range(len(all_year_G)):
```

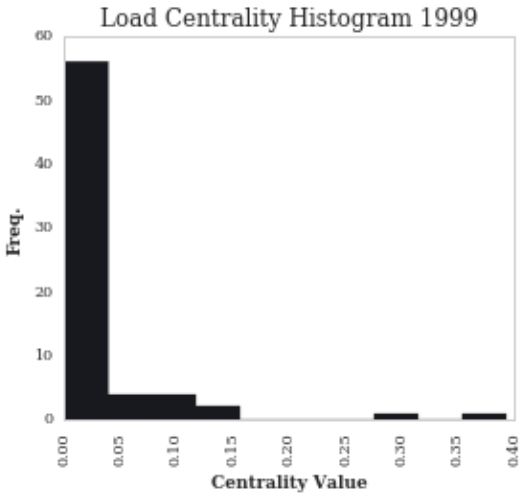
```
        deg = nx.katz_centrality_numpy(all_year_G[i]).values()
        deg = sorted(deg)
        plt.subplot(3, 2, i+1)
        plt.hist(deg)
        plt.title("Katz Centrality Histogram " + str(years[i]), fontsize=11)
        plt.xlabel("Centrality Value")
        plt.ylabel("Freq.")
        plt.grid(False)
        plt.tight_layout()
    plt.savefig('images/year_katzhist.png')
```

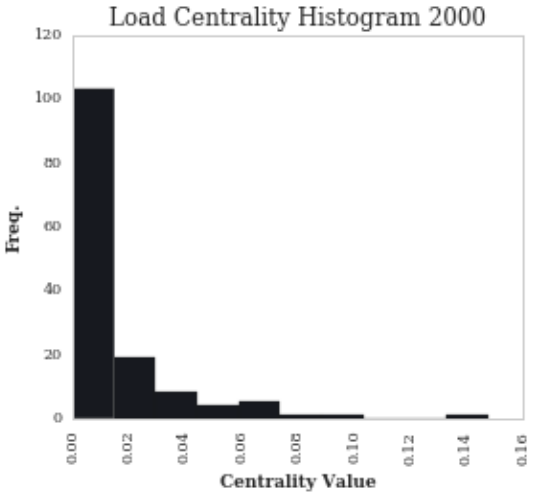


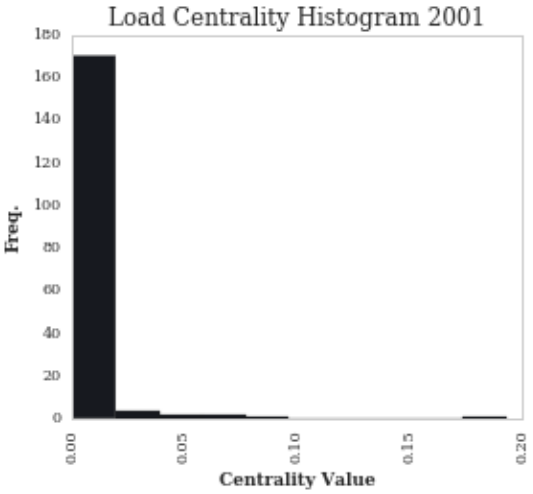

```
In [140]: plt.figure(figsize=(8,11))
          for i in range(len(all_year_G)):
```

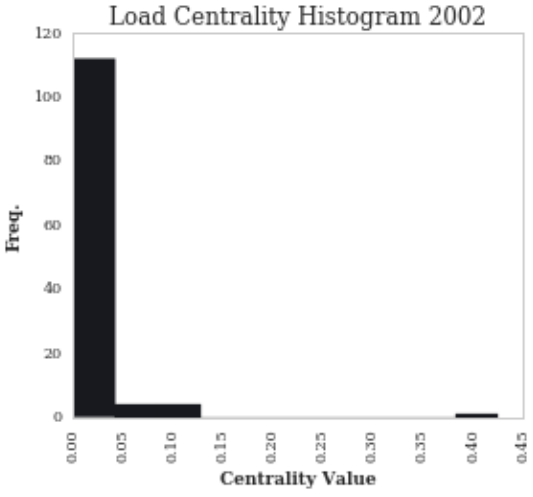
```
        deg = nx.load_centrality(all_year_G[i]).values()
        deg = sorted(deg)
        plt.subplot(3, 2, i+1)
        plt.hist(deg)
        plt.title("Load Centrality Histogram " + str(years[i]), fontsize=12)
        plt.xticks(rotation=90)
        plt.xlabel("Centrality Value")
        plt.ylabel("Freq.")
        plt.grid(False)
        plt.tight_layout()
    plt.savefig('images/year_loadhist.png')
```


Load Centrality Histogram 1998
Freq.
Centrality Value


Load Centrality Histogram 1999
Freq.
Centrality Value

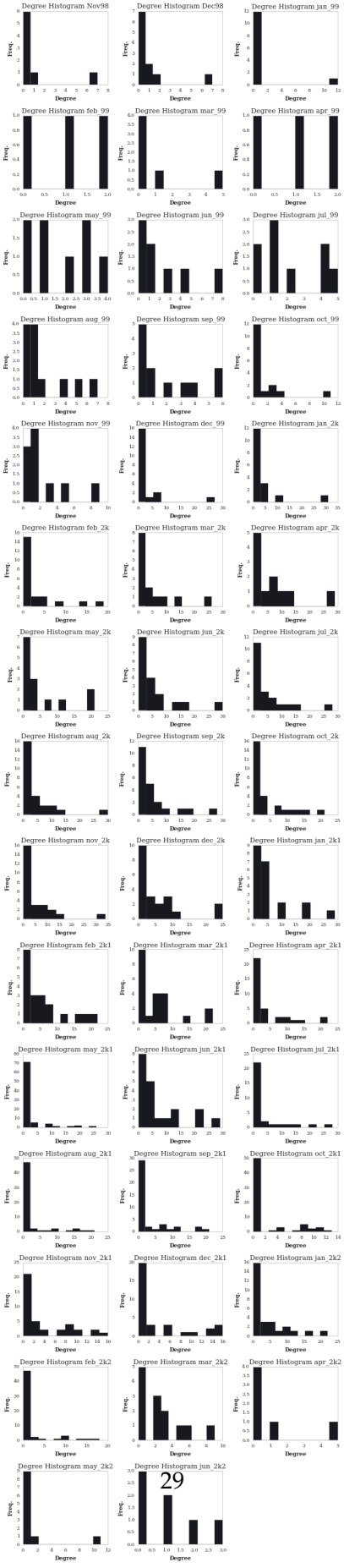

Load Centrality Histogram 2000
Freq.
Centrality Value


Load Centrality Histogram 2001
Freq.
Centrality Value

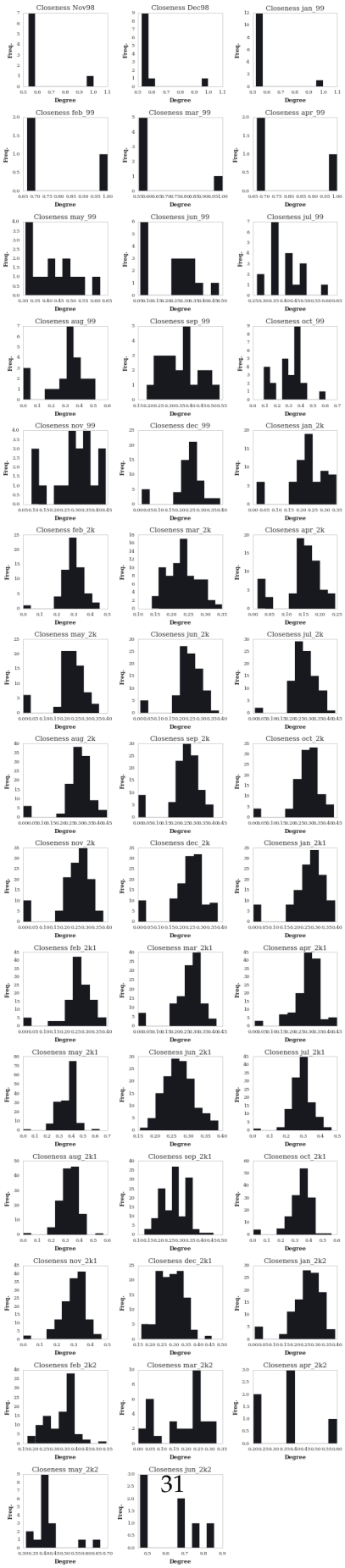

Load Centrality Histogram 2002
Freq.
Centrality Value

4.2 Monthly Networks

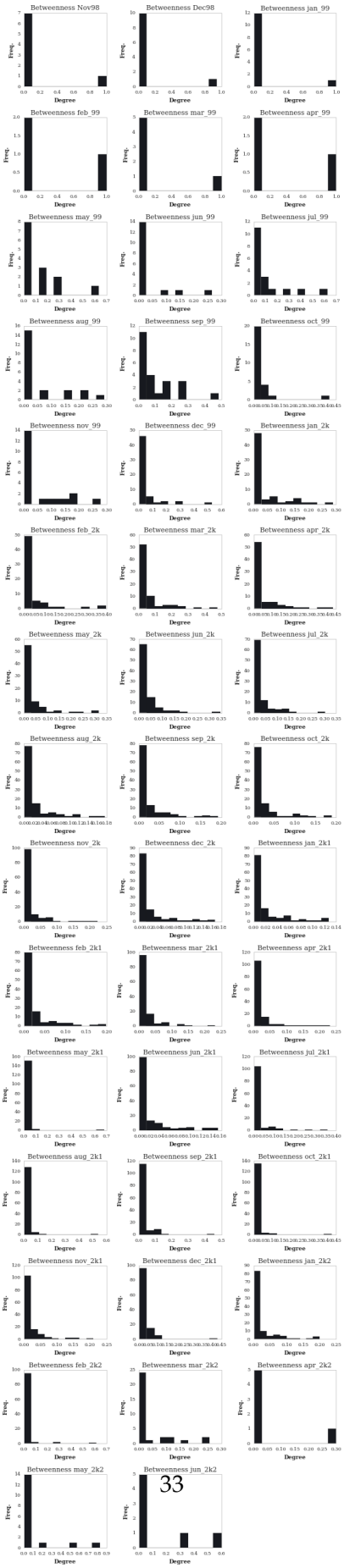
In [141]:
```
plt.figure(figsize=(8,38))
for i in range(len(all_month_G)):
    deg = nx.degree_histogram(all_month_G[i])
    plt.subplot(16, 3, i+1)
    plt.hist(deg)
    plt.title("Degree Histogram " + str(months[i]), fontsize=11)
    plt.xlabel("Degree")
    plt.ylabel("Freq.")
    plt.grid(False)
    plt.tight_layout()
plt.savefig('images/mth_deghist.png')
```


Degree Histogram Nov98
Degree Histogram Dec98
Degree Histogram jan_99
Degree Histogram feb_99
Degree Histogram mar_99
Degree Histogram apr_99
Degree Histogram may_99
Degree Histogram jun_99
Degree Histogram jul_99
Degree Histogram aug_99
Degree Histogram sep_99
Degree Histogram oct_99
Degree Histogram nov_99
Degree Histogram dec_99
Degree Histogram jan_2k
Degree Histogram feb_2k
Degree Histogram mar_2k
Degree Histogram apr_2k
Degree Histogram may_2k
Degree Histogram jun_2k
Degree Histogram jul_2k
Degree Histogram aug_2k
Degree Histogram sep_2k
Degree Histogram oct_2k
Degree Histogram nov_2k
Degree Histogram dec_2k
Degree Histogram jan_2k1
Degree Histogram feb_2k1
Degree Histogram mar_2k1
Degree Histogram apr_2k1
Degree Histogram may_2k1
Degree Histogram jun_2k1
Degree Histogram jul_2k1
Degree Histogram aug_2k1
Degree Histogram sep_2k1
Degree Histogram oct_2k1
Degree Histogram nov_2k1
Degree Histogram dec_2k1
Degree Histogram jan_2k2
Degree Histogram feb_2k2
Degree Histogram mar_2k2
Degree Histogram apr_2k2
Degree Histogram may_2k2
Degree Histogram jun_2k2
Freq.
Degree

```
In [142]: plt.figure(figsize=(8,38))
          for i in range(len(all_month_G)):
              deg = nx.closeness_centrality(all_month_G[i]).values()
              deg = sorted(deg)
              plt.subplot(16, 3, i+1)
              plt.hist(deg)
              plt.title("Closeness " + str(months[i]), fontsize=11)
              plt.xlabel("Degree")
              plt.ylabel("Freq.")
              plt.grid(False)
              plt.tight_layout()
          plt.savefig('images/mth_clohist.png')
```

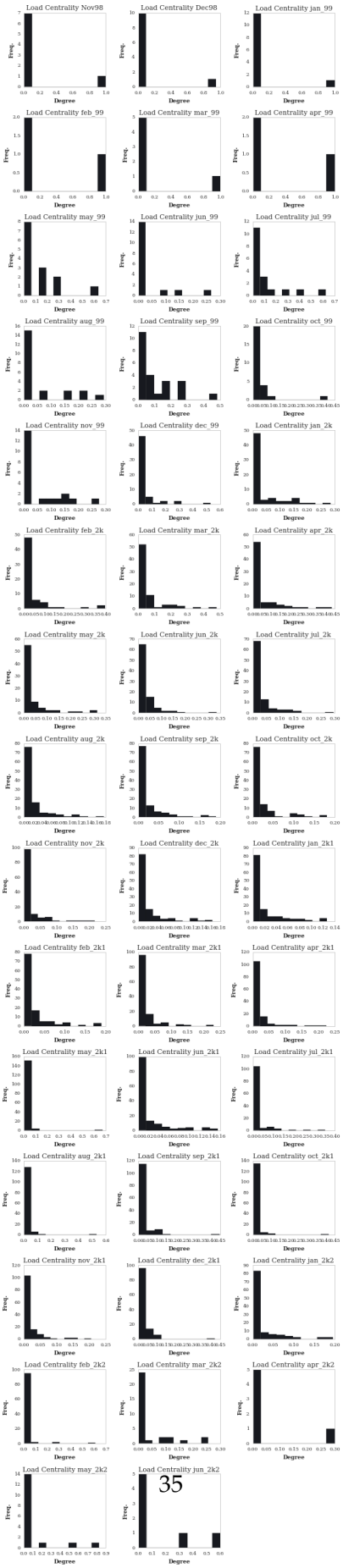

Closeness Nov98
Closeness Dec98
Closeness jan_99
Closeness feb_99
Closeness mar_99
Closeness apr_99
Closeness may_99
Closeness jun_99
Closeness jul_99
Closeness aug_99
Closeness sep_99
Closeness oct_99
Closeness nov_99
Closeness dec_99
Closeness jan_2k
Closeness feb_2k
Closeness mar_2k
Closeness apr_2k
Closeness may_2k
Closeness jun_2k
Closeness jul_2k
Closeness aug_2k
Closeness sep_2k
Closeness oct_2k
Closeness nov_2k
Closeness dec_2k
Closeness jan_2k1
Closeness feb_2k1
Closeness mar_2k1
Closeness apr_2k1
Closeness may_2k1
Closeness jun_2k1
Closeness jul_2k1
Closeness aug_2k1
Closeness sep_2k1
Closeness oct_2k1
Closeness nov_2k1
Closeness dec_2k1
Closeness jan_2k2
Closeness feb_2k2
Closeness mar_2k2
Closeness apr_2k2
Closeness may_2k2
Closeness jun_2k2
Freq.
Degree

```
In [143]: plt.figure(figsize=(8,38))
          for i in range(len(all_month_G)):
              deg = nx.betweenness_centrality(all_month_G[i]).values()
              deg = sorted(deg)
              plt.subplot(16, 3, i+1)
              plt.hist(deg)
              plt.title("Betweenness " + str(months[i]), fontsize=11)
              plt.xlabel("Degree")
              plt.ylabel("Freq.")
              plt.grid(False)
              plt.tight_layout()
          plt.savefig('images/mth_bethist.png')
```


Betweenness Nov98
Betweenness Dec98
Betweenness jan_99
Betweenness feb_99
Betweenness mar_99
Betweenness apr_99
Betweenness may_99
Betweenness jun_99
Betweenness jul_99
Betweenness aug_99
Betweenness sep_99
Betweenness oct_99
Betweenness nov_99
Betweenness dec_99
Betweenness jan_2k
Betweenness feb_2k
Betweenness mar_2k
Betweenness apr_2k
Betweenness may_2k
Betweenness jun_2k
Betweenness jul_2k
Betweenness aug_2k
Betweenness sep_2k
Betweenness oct_2k
Betweenness nov_2k
Betweenness dec_2k
Betweenness jan_2k1
Betweenness feb_2k1
Betweenness mar_2k1
Betweenness apr_2k1
Betweenness may_2k1
Betweenness jun_2k1
Betweenness jul_2k1
Betweenness aug_2k1
Betweenness sep_2k1
Betweenness oct_2k1
Betweenness nov_2k1
Betweenness dec_2k1
Betweenness jan_2k2
Betweenness feb_2k2
Betweenness mar_2k2
Betweenness apr_2k2
Betweenness may_2k2
Betweenness jun_2k2
Freq.
Degree

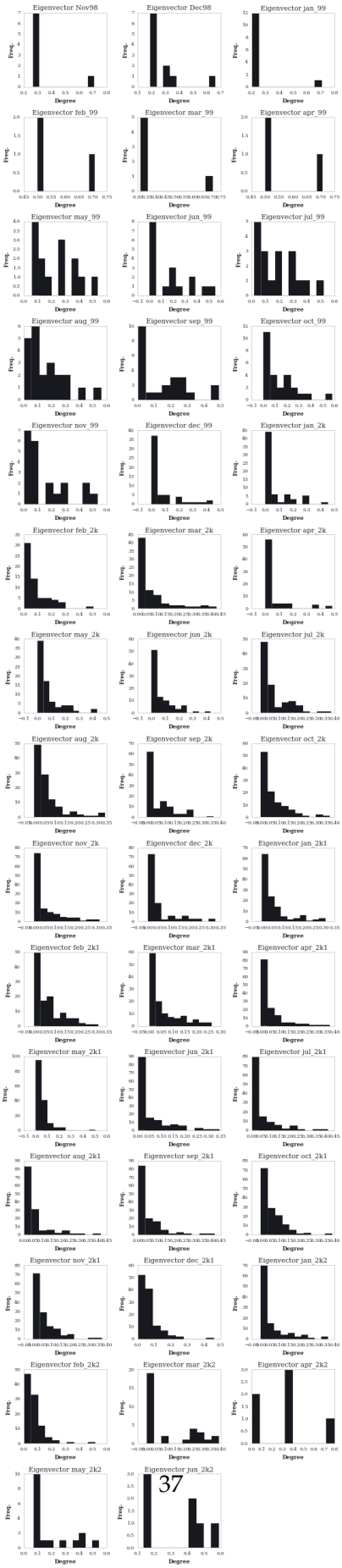
```
In [229]: plt.figure(figsize=(8,38))
          for i in range(len(all_month_G)):
              deg = nx.load_centrality(all_month_G[i]).values()
              deg = sorted(deg)
              plt.subplot(16, 3, i+1)
              plt.hist(deg)
              plt.title("Load Centrality " + str(months[i]), fontsize=11)
              plt.xlabel("Degree")
              plt.ylabel("Freq.")
              plt.grid(False)
              plt.tight_layout()
          plt.savefig('images/mth_loadhist.png')
```

```
/home/arshad/anaconda3/lib/python3.5/site-packages/matplotlib/figure.py:1742: UserW
  warnings.warn("This figure includes Axes that are not "
```

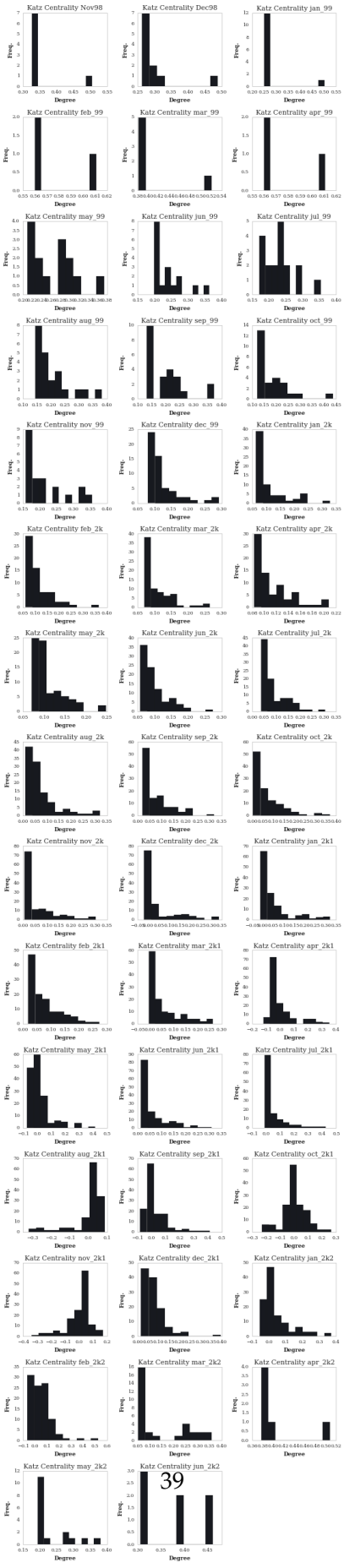

Load Centrality Nov98
Load Centrality Dec98
Load Centrality jan_99
Load Centrality feb_99
Load Centrality mar_99
Load Centrality apr_99
Load Centrality may_99
Load Centrality jun_99
Load Centrality jul_99
Load Centrality aug_99
Load Centrality sep_99
Load Centrality oct_99
Load Centrality nov_99
Load Centrality dec_99
Load Centrality jan_2k
Load Centrality feb_2k
Load Centrality mar_2k
Load Centrality apr_2k
Load Centrality may_2k
Load Centrality jun_2k
Load Centrality jul_2k
Load Centrality aug_2k
Load Centrality sep_2k
Load Centrality oct_2k
Load Centrality nov_2k
Load Centrality dec_2k
Load Centrality jan_2k1
Load Centrality feb_2k1
Load Centrality mar_2k1
Load Centrality apr_2k1
Load Centrality may_2k1
Load Centrality jun_2k1
Load Centrality jul_2k1
Load Centrality aug_2k1
Load Centrality sep_2k1
Load Centrality oct_2k1
Load Centrality nov_2k1
Load Centrality dec_2k1
Load Centrality jan_2k2
Load Centrality feb_2k2
Load Centrality mar_2k2
Load Centrality apr_2k2
Load Centrality may_2k2
Load Centrality jun_2k2
Freq.
Degree

```
In [228]: plt.figure(figsize=(8,38))
          for i in range(len(all_month_G)):
              deg = nx.eigenvector_centrality_numpy(all_month_G[i]).values()
              deg = sorted(deg)
              plt.subplot(16,3, i+1)
              plt.hist(deg)
              plt.title("Eigenvector " + str(months[i]), fontsize=11)
              plt.xlabel("Degree")
              plt.ylabel("Freq.")
              plt.grid(False)
              plt.tight_layout()
          plt.savefig('images/mth_eighist.png')
```

```
/home/arshad/anaconda3/lib/python3.5/site-packages/matplotlib/figure.py:1742: UserW
  warnings.warn("This figure includes Axes that are not "
```


Eigenvector Nov98
Eigenvector Dec98
Eigenvector jan_99
Eigenvector feb_99
Eigenvector mar_99
Eigenvector apr_99
Eigenvector may_99
Eigenvector jun_99
Eigenvector jul_99
Eigenvector aug_99
Eigenvector sep_99
Eigenvector oct_99
Eigenvector nov_99
Eigenvector dec_99
Eigenvector jan_2k
Eigenvector feb_2k
Eigenvector mar_2k
Eigenvector apr_2k
Eigenvector may_2k
Eigenvector jun_2k
Eigenvector jul_2k
Eigenvector aug_2k
Eigenvector sep_2k
Eigenvector oct_2k
Eigenvector nov_2k
Eigenvector dec_2k
Eigenvector jan_2k1
Eigenvector feb_2k1
Eigenvector mar_2k1
Eigenvector apr_2k1
Eigenvector may_2k1
Eigenvector jun_2k1
Eigenvector jul_2k1
Eigenvector aug_2k1
Eigenvector sep_2k1
Eigenvector oct_2k1
Eigenvector nov_2k1
Eigenvector dec_2k1
Eigenvector jan_2k2
Eigenvector feb_2k2
Eigenvector mar_2k2
Eigenvector apr_2k2
Eigenvector may_2k2
Eigenvector jun_2k2
Freq.
Degree

```
In [145]: plt.figure(figsize=(8,38))
          for i in range(len(all_month_G)):
              deg = nx.katz_centrality_numpy(all_month_G[i]).values()
              deg = sorted(deg)
              plt.subplot(16,3, i+1)
              plt.hist(deg)
              plt.title("Katz Centrality " + str(months[i]), fontsize=11)
              plt.xlabel("Degree")
              plt.ylabel("Freq.")
              plt.grid(False)
              plt.tight_layout()
          plt.savefig('images/mth_katzhist.png')
```


Katz Centrality Nov98
Katz Centrality Dec98
Katz Centrality jan_99
Katz Centrality feb_99
Katz Centrality mar_99
Katz Centrality apr_99
Katz Centrality may_99
Katz Centrality jun_99
Katz Centrality jul_99
Katz Centrality aug_99
Katz Centrality sep_99
Katz Centrality oct_99
Katz Centrality nov_99
Katz Centrality dec_99
Katz Centrality jan_2k
Katz Centrality feb_2k
Katz Centrality mar_2k
Katz Centrality apr_2k
Katz Centrality may_2k
Katz Centrality jun_2k
Katz Centrality jul_2k
Katz Centrality aug_2k
Katz Centrality sep_2k
Katz Centrality oct_2k
Katz Centrality nov_2k
Katz Centrality dec_2k
Katz Centrality jan_2k1
Katz Centrality feb_2k1
Katz Centrality mar_2k1
Katz Centrality apr_2k1
Katz Centrality may_2k1
Katz Centrality jun_2k1
Katz Centrality jul_2k1
Katz Centrality aug_2k1
Katz Centrality sep_2k1
Katz Centrality oct_2k1
Katz Centrality nov_2k1
Katz Centrality dec_2k1
Katz Centrality jan_2k2
Katz Centrality feb_2k2
Katz Centrality mar_2k2
Katz Centrality apr_2k2
Katz Centrality may_2k2
Katz Centrality jun_2k2
Freq.
Degree

5 Attribute Analysis

5.1 Traditional Measures

5.1.1 Centrality

- Degree
- Betweeness
- Closeness
- Katz
- Load

5.1.2 Assortativity & Linear Algebra

- Density
- Average Clustering Coefficient
- Algebraic Connectivity

5.2 Complex Trace Attributes

- Instantaneous Amplitude
- Instantaneous Frequency
- Instantaneous Phase
- Derivative of Amplitude
- Second Derivative of Amplitude
- Power
- Cosine of Instantaneous Phase
- Amplitude weighted Frequency
- Amplitude weighted Phase
- Power Spectral Density

5.3 Matrix

- Resistance Distance
- Stationarity Ratio
- Subgraph Stationarity
- Curvature
- Norm of Abel Transform

5.4 Matrix Decomposition

- KLPCA Ratio Change
- Norm NMF Ratio Change

5.5 Music Attributes

- Zero Crossing Rate
- Spectral Centroid

5.6 Aggregation Measures

- Persistence
- Emergence
- NRMS of Attributes

```
In [38]: def calc_seisatt(net):
             M = nx.normalized_laplacian_matrix(net).todense()
             Ht = hilbert(M)
             rHt = np.real(Ht)
             iHt = np.imag(Ht)

             #Basic attributes IA, IP, IF
             IA = np.real(np.nan_to_num(np.sqrt(np.dot(rHt,rHt)+np.dot(iHt,iHt))))
             IP = np.real(np.nan_to_num(np.arctan(iHt/rHt)))
             IF,_ = np.real(np.nan_to_num(np.asarray(np.gradient(IP))))
             P = np.square(IA)

             #Derivatives
             dIA,_ = np.nan_to_num(np.asarray(np.gradient(IA)))
             d2IA,_ = np.nan_to_num(np.asarray(np.gradient(dIA)))
             IAcc,_ =np.nan_to_num(np.asarray(np.gradient(IF)))

             #Derived Attributes
             cosIP = np.cos(IP)
             IA_weit_IF = IA * IF
             IA_weit_IP = IA*IP

             return [IA,IP,IF,P, dIA,d2IA, IAcc,cosIP,IA_weit_IF,IA_weit_IP]

In [39]: def stationarity_ratio(G):
             #stationarity ratio with laplian
             L = nx.normalized_laplacian_matrix(G).todense()
             U = eigvals(L)
             C =np.cov(L)
             CF = np.dot(L,np.dot(np.dot(U.T,C),U))
             r = np.linalg.norm(np.diag(CF))/np.linalg.norm(CF)

             return r

In [40]: #cite:`klein1993resistance`
         def resistance_distance(net):
             M = nx.normalized_laplacian_matrix(net).todense()
             pseudo = pinv(M)
```

```
            N = M.shape[0]
            d = np.diag(pseudo)
            rd  = np.kron(d,np.ones((N,1))).T+np.kron(d,np.ones((N,1))).T - pseudo

            return rd.mean()

In [41]: def curvature(net):
            from skimage.feature import hessian_matrix, hessian_matrix_det, hessia
            M = nx.normalized_laplacian_matrix(net).todense()
            M = np.float64(M)
            fx, fy = np.gradient(M)
            Hxx, Hxy, Hyy = hessian_matrix(M)
            K = np.divide((np.dot(Hxx,Hxy)-np.dot(Hxy,Hxy)),\
                          (1+np.dot(fx,fx)+np.dot(fy,fy)))
            He1,_ = hessian_matrix_eigvals(Hxx,Hxy,Hyy)
            mean_curv = np.trace(He1)

            return mean_curv

In [42]: def kpca_ratio(net):
            from sklearn.decomposition import KernelPCA
            kpca = KernelPCA(n_components=3,kernel='rbf')
            m = nx.normalized_laplacian_matrix(net).todense()
            X_kpca = kpca.fit_transform(m)
            pc1 = X_kpca[:,0]
            pc2 = X_kpca[:,1]
            pc3 = X_kpca[:,2]
            kpca_rat = norm(pc1-pc3/(pc1-pc2))
            return kpca_rat

In [43]: def kpca_att(net):
            kpca_chg = []
            for i in range(len(net)-1):
                x = int(i)
                y = 1+x
                kpcar1= kpca_ratio(net[x])
                kpcar2= kpca_ratio(net[y])
                kpca_chg.append(np.divide(kpcar2,kpcar1))
            kpca_chgpad = np.zeros(len(kpca_chg)+1);
            kpca_chgpad[1:] = kpca_chg

            return kpca_chgpad

In [44]: def nmf_att(net):
            nmf_chg = []

            nmf = NMF(init='nndsvda',solver='cd', random_state=0, l1_ratio=1)
            for i in range(len(net)-1):
                x = int(i)
```

```
            y = 1+x
            m1= abs(nx.normalized_laplacian_matrix(net[x]).todense())
            m2= abs(nx.normalized_laplacian_matrix(net[y]).todense())
            nmf1 = norm(nmf.fit_transform(m1))
            nmf2 = norm(nmf.fit_transform(m2))
            nmf_chg.append(np.divide(nmf2,nmf1))
        nmf_chgpad = np.zeros(len(nmf_chg)+1);
        nmf_chgpad[1:] = nmf_chg

        return nmf_chgpad
```

In [45]:
```
def pad_shape(x,ref,offset=0):
    result = np.zeros_like(ref)
    result[0:x.shape[0]+0,0:x.shape[1]+0] = x

    return result

def subgraph_stat(net1,net2):
    net1_int_net2 = net1.copy()
    net1_int_net2.remove_nodes_from(n for n in net1 if n not in net2)
    net1_u_net2 = nx.disjoint_union(net1, net2)
    int_adjmat = nx.normalized_laplacian_matrix(net1_int_net2).todense()
    uni_adjmat = nx.normalized_laplacian_matrix(net1_u_net2).todense()
    int_adjmat_pad = pad_shape(int_adjmat,uni_adjmat)

    Ct = np.divide(norm(int_adjmat_pad),norm(uni_adjmat))

    return Ct

def zeta(n):

    Ct_m = []
    for i in range(len(n)-1):
        x = int(i)
        y = x +1
        Ct_m.append(subgraph_stat(n[x],n[y]))
    Ct_m_pad = np.zeros(len(Ct_m)+1);
    Ct_m_pad[1:] = Ct_m
    return Ct_m_pad
```

In [46]:
```
def music_att(n):
    #music attributes
    f = sc.fftpack.rfft(nx.normalized_laplacian_matrix(n).todense()).mean
    zcr = librosa.feature.zero_crossing_rate(f)[0,0]
    spc = librosa.feature.spectral_centroid(f)[0,0]

    return [zcr,spc]
```

In [47]:
```
def norm_fabel(x):
```

```
            import abel
            fabel_att =[]
            for n in x:
                tmp = nx.normalized_laplacian_matrix(n).todense()
                tmp2 = sc.fftpack.fft2(tmp)
                fabel2 = abel.Transform(tmp2, direction='forward', method='direct'
                mag = np.sqrt(np.square(np.real(fabel2)) + np.square(np.imag(fabel
                fabel_att.append(norm(mag))

            return np.log10(fabel_att)

In [48]: def cal_avg_atts(x):

            #define attributes placeholders here
            deg_y = []
            bet_y = []
            clo_y = []
            load_y = []
            eig_y = []
            katz_y = []
            den_y = []
            algc_y = []
            cluscof_y = []
            IA_y = []
            IP_y = []
            IF_y = []
            P_y = []
            dIA_y = []
            d2IA_y =[]
            IAcc_y = []
            cosIP_y = []
            IA_weit_IF_y = []
            IA_weit_IP_y = []
            psd_y = []
            rd_y = []
            zcr_y = []
            spc_y = []
            statr_y = []
            meank_y = []

            #matrix decompostion attributes
            zeta_y = zeta(x)
            nmf_ratio_y = nmf_att(x)
            kpca_chg_y= kpca_att(x)
            prop_members_chg_1_zeta = 1-zeta_y
            fabel = norm_fabel(x)

            for n in x:
```

```
        deg = np.mean(sorted(set(nx.degree_centrality(n).values())))
        bet = np.mean(sorted(set(nx.betweenness_centrality(n).values())))
        clo = np.mean(sorted(set(nx.closeness_centrality(n).values())))
        katz = np.mean(sorted(set(nx.katz_centrality_numpy(n).values())))
        eig = np.mean(sorted(set(nx.eigenvector_centrality_numpy(n).values
        load = np.mean(sorted(set(nx.degree_centrality(n).values())))
        den = nx.density(n)
        algc = nx.algebraic_connectivity(n)
        clustcof = nx.average_clustering(n)

        #all network metrics
        deg_y.append(deg), bet_y.append(bet), clo_y.append(clo), load_y.ap
        den_y.append(den),  algc_y.append(algc), cluscof_y.append(clustcof

        #complex trace attributes
        IA,IP,IF,P, dIA,d2IA, IAcc,cosIP,IA_weit_IF,IA_weit_IP = calc_seis
        IA_y.append(IA.mean())
        IP_y.append(IP.mean())
        IF_y.append(IF.mean())
        P_y.append(P.mean())
        dIA_y.append(dIA.mean())
        d2IA_y.append(d2IA.mean())
        IAcc_y.append(IAcc.mean())
        cosIP_y.append(cosIP.mean())
        IA_weit_IF_y.append(IA_weit_IF.mean())
        IA_weit_IP_y.append(IA_weit_IP.mean())

        psd, _ = plt.psd(nx.laplacian_matrix(n).todense());
        plt.close()
        psd_y.append(psd.mean())

        zcr, spc = music_att(n)
        zcr_y.append(zcr)
        spc_y.append(np.log10(spc))

        #matrix attributes
        rdm = resistance_distance(n)
        rd_y.append(rdm)

        statrat = stationarity_ratio(n)
        statr_y.append(statrat)

        meank = curvature(n)
        meank_y.append(meank)



colnames = ['AvgDeg','AvgBet','AvgClo','AvgLoad','AvgKatz','AvgDensity
```

```
                        'AvgEig' ,'InstAmp','InstPhase','InstFreq','Power','dInstAm
                        'A_wt_IF','A_wt_IP','PowerSpecDen','ResDist','ZeroCrossRate

            attvol_y = pd.DataFrame([deg_y,bet_y, clo_y ,load_y ,katz_y ,den_y, al
                          P_y ,dIA_y ,d2IA_y,IAcc_y ,cosIP_y ,IA_weit_IF_y ,IA_weit
                          statr_y , meank_y]).T

            attvol_y.columns = colnames
            attvol_y['SubgraphStat']=zeta_y
            attvol_y['1-Zeta'] = prop_members_chg_1_zeta
            attvol_y['LogKPCARatioChg'] = np.log10(kpca_chg_y)
            attvol_y.LogKPCARatioChg[0]=0
            attvol_y['NormNMFRatioChg']= nmf_ratio_y
            attvol_y['NormFAbel'] = fabel
            attvol_y_sc = attvol_y.apply(lambda x: minmax_scale(x, feature_range=

            return attvol_y_sc
```

```
In [49]: attvol_y = cal_avg_atts(all_year_G)
```

```
/home/arshad/anaconda3/lib/python3.5/site-packages/abel/transform.py:341: ComplexWa
  self.IM = self.IM.astype('float64')
```

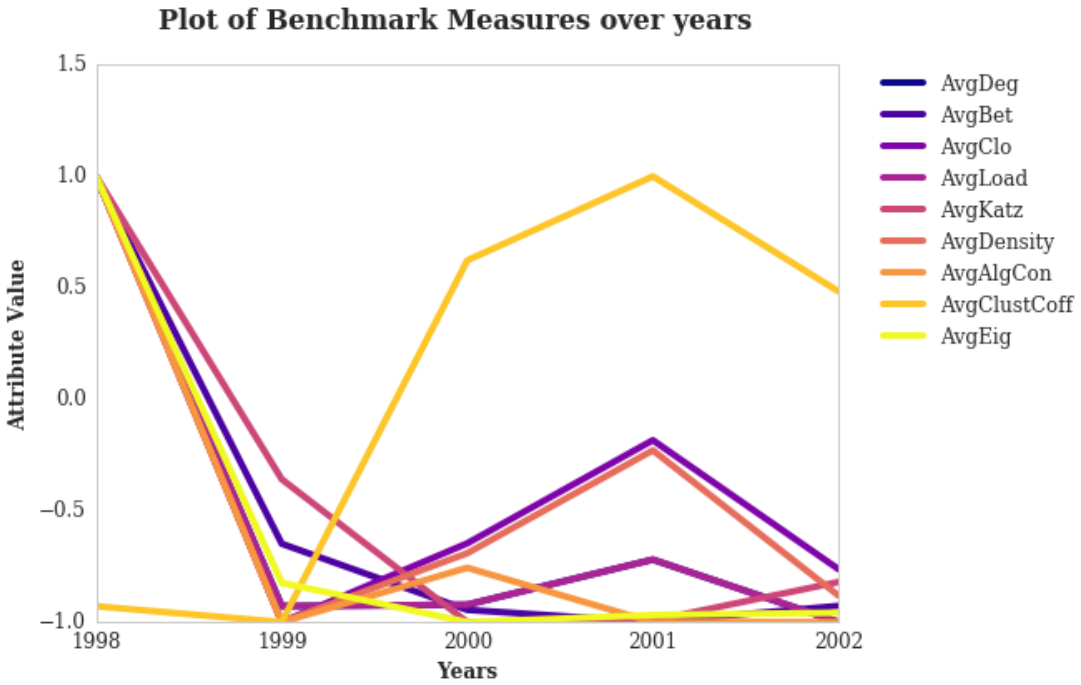
```
In [50]: attvol_m = cal_avg_atts(all_month_G)
```

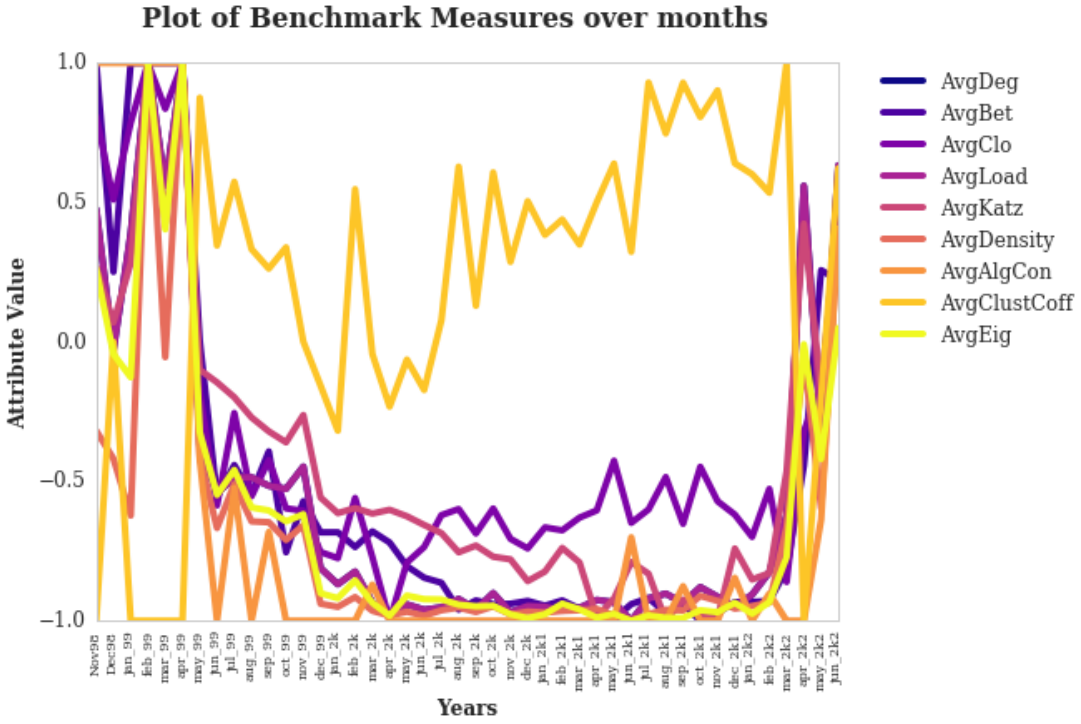
```
/home/arshad/anaconda3/lib/python3.5/site-packages/abel/transform.py:341: ComplexWa
  self.IM = self.IM.astype('float64')
```

6 Overview Plots

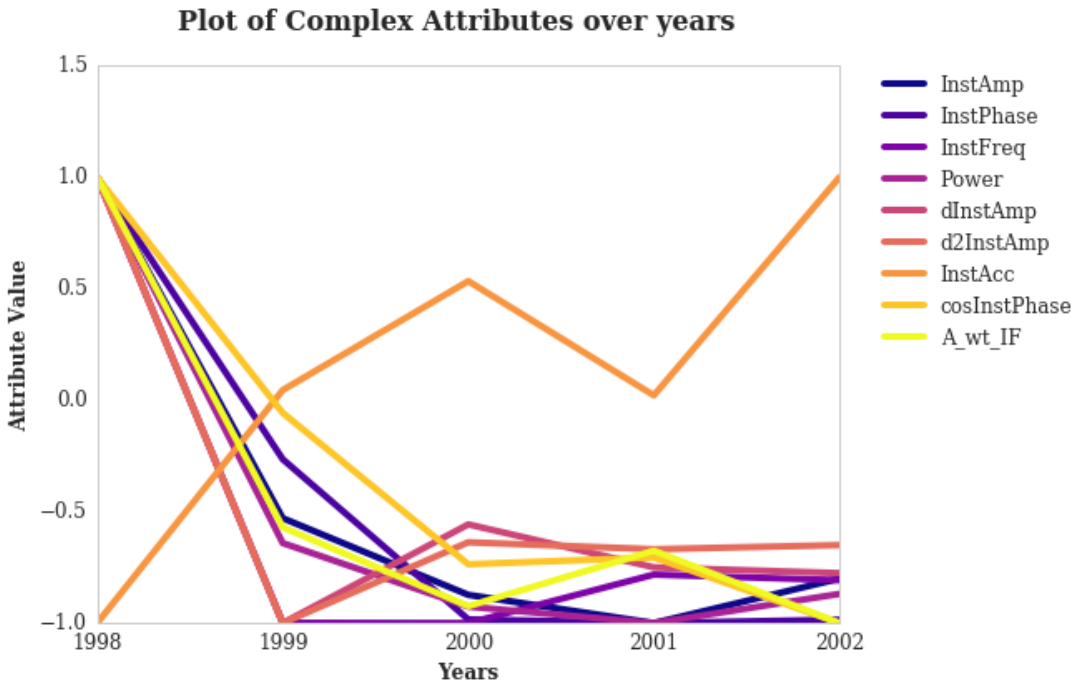
```
In [153]: attvol_y.iloc[:,:9].plot.line(use_index=True,cmap='plasma')
          plt.xlabel("Years", fontsize=12)
          plt.ylabel("Attribute Value", fontsize=12)
          plt.legend(fontsize=12, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.
          plt.xticks(np.arange(len(years)), years,fontsize=12)
          plt.yticks(fontsize=12)
          plt.suptitle("Plot of Benchmark Measures over years", fontsize=16)

          attvol_m.iloc[:,:9].plot.line(use_index=True,cmap='plasma', rot=90)
          plt.xlabel("Years", fontsize=12)
          plt.ylabel("Attribute Value", fontsize=12)
          plt.legend(fontsize=12, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.
          plt.xticks(np.arange(len(months)), months)
          plt.yticks(fontsize=12)
          plt.suptitle("Plot of Benchmark Measures over months", fontsize=16)
```

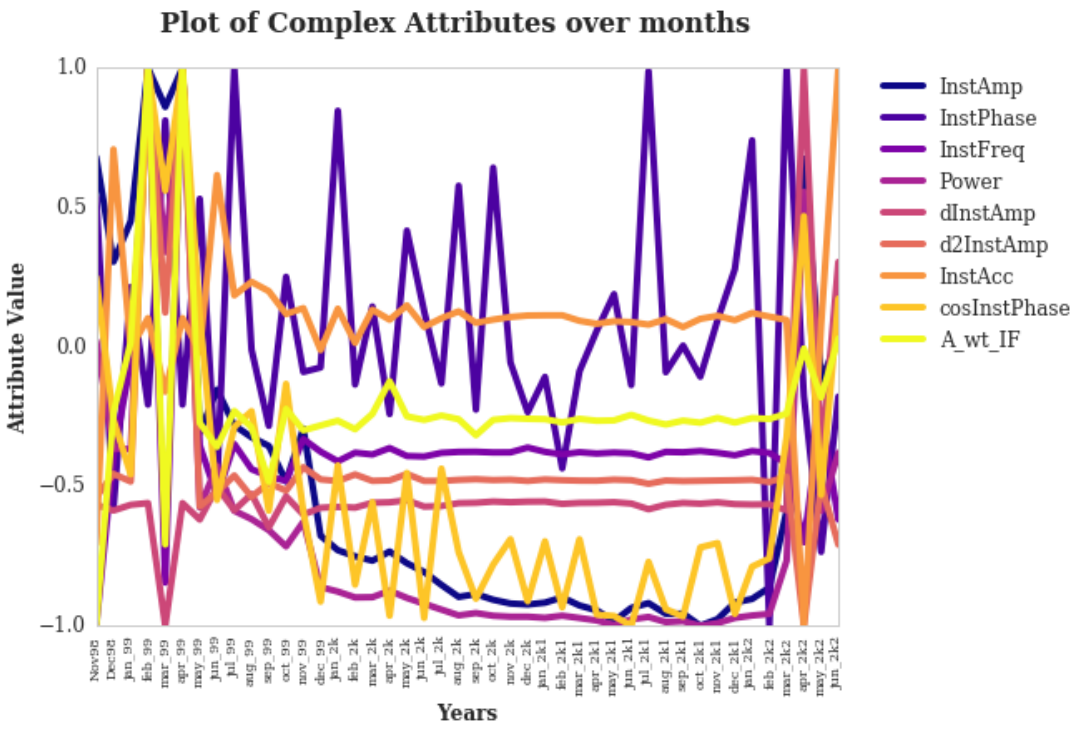
```
Out[153]: <matplotlib.text.Text at 0x7f625cd4bfd0>
```


Plot of Benchmark Measures over years
Attribute Value
1.5
1.0
0.5
0.0
−0.5
−1.0
1998
1999
2000
2001
2002
Years
AvgDeg
AvgBet
AvgClo
AvgLoad
AvgKatz
AvgDensity
AvgAlgCon
AvgClustCoff
AvgEig

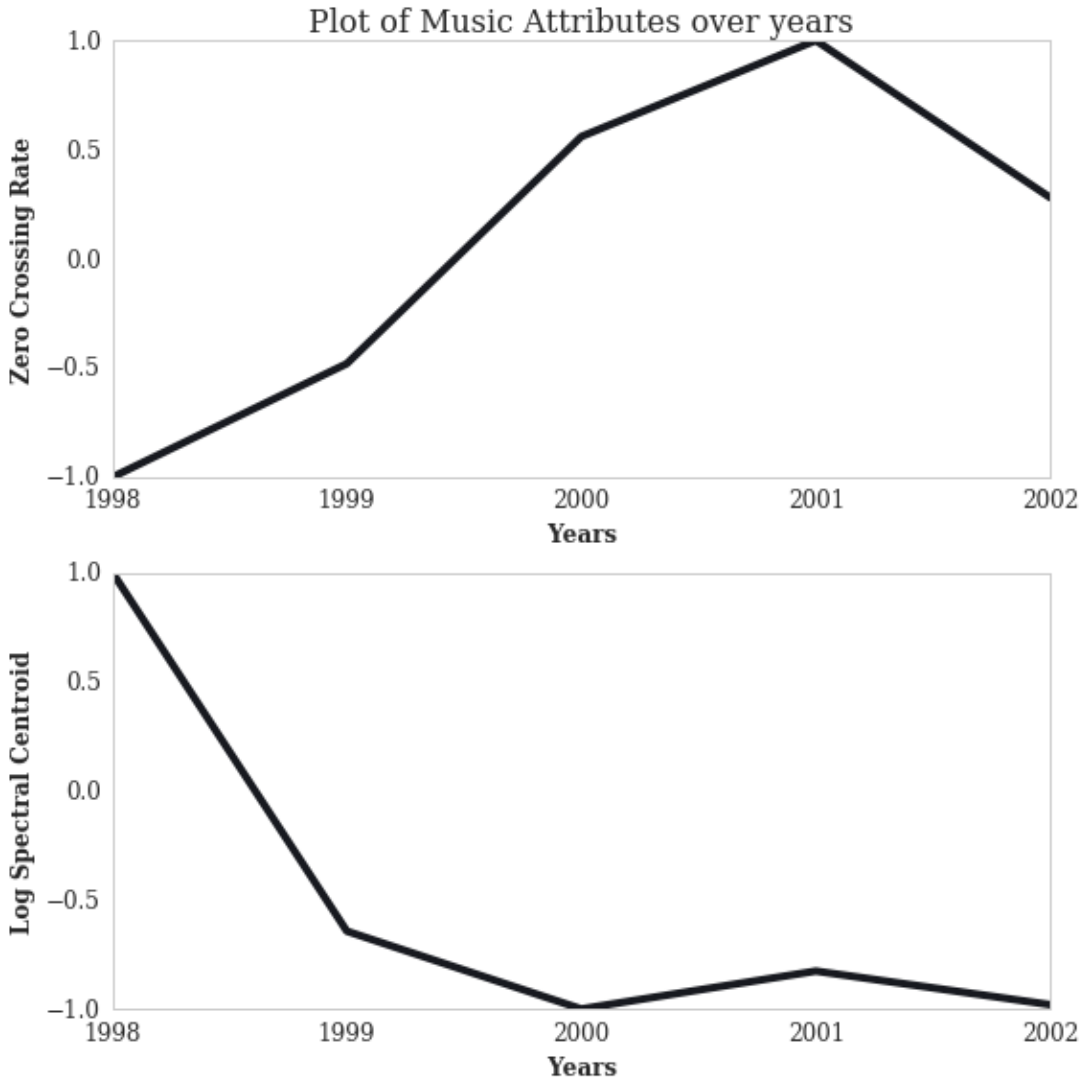

Plot of Benchmark Measures over months
Attribute Value
1.0
0.5
0.0
−0.5
−1.0
Years
AvgDeg
AvgBet
AvgClo
AvgLoad
AvgKatz
AvgDensity
AvgAlgCon
AvgClustCoff
AvgEig

```
In [155]: attvol_y.iloc[:,9:18].plot.line(use_index=True,cmap='plasma')
          plt.xlabel("Years", fontsize=12)
          plt.ylabel("Attribute Value", fontsize=12)
          plt.legend(fontsize=12, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.
          plt.xticks(np.arange(len(years)), years,fontsize=12)
          plt.yticks(fontsize=12)
          plt.suptitle("Plot of Complex Attributes over years", fontsize=16)
          plt.autoscale()

          attvol_m.iloc[:,9:18].plot.line(use_index=True,cmap='plasma', rot=90)
          plt.xlabel("Years", fontsize=12)
          plt.ylabel("Attribute Value", fontsize=12)
          plt.legend(fontsize=12, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.
          plt.xticks(np.arange(len(months)), months)
          plt.yticks(fontsize=12)
          plt.suptitle("Plot of Complex Attributes over months", fontsize=16)
          plt.autoscale()
```



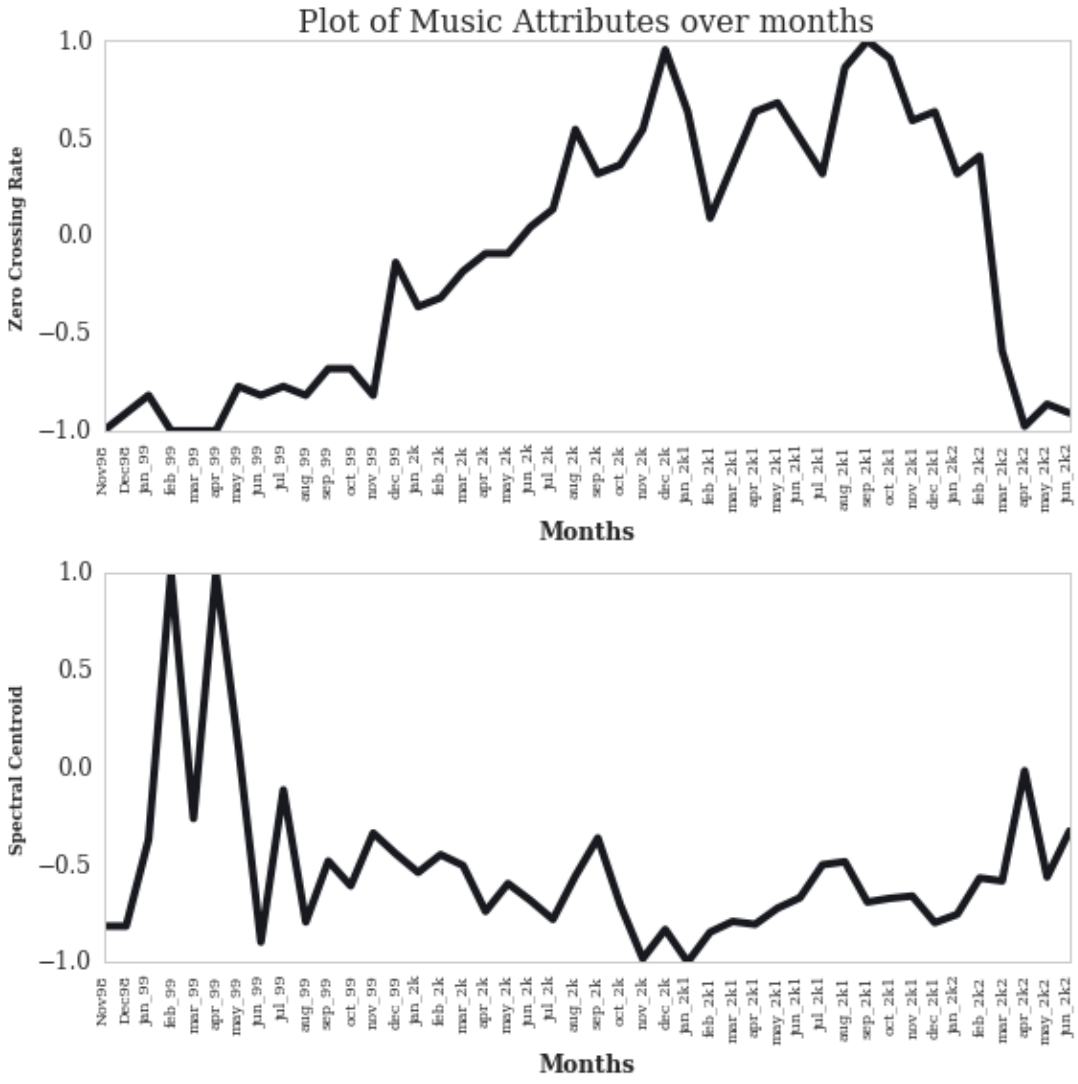



```
In [156]: plt.figure(figsize=(8,8))

          plt.subplot(2,1,1)
          plt.title("Plot of Music Attributes over years", fontsize=16)
          plt.plot(attvol_y.ZeroCrossRate)
          plt.xlabel("Years", fontsize=12)
          plt.ylabel("Zero Crossing Rate", fontsize=12)
          plt.xticks(np.arange(len(years)),years,fontsize=12)
          plt.yticks(fontsize=12)
          plt.autoscale()

          plt.subplot(2,1,2)
          plt.plot(attvol_y.LogSpecCentroid)
          plt.xlabel("Years", fontsize=12)
          plt.ylabel("Log Spectral Centroid", fontsize=12)
          plt.xticks(np.arange(len(years)),years,fontsize=12)
          plt.yticks(fontsize=12)
          plt.tight_layout()
          plt.savefig('images/musicatt_yrs.png')
```



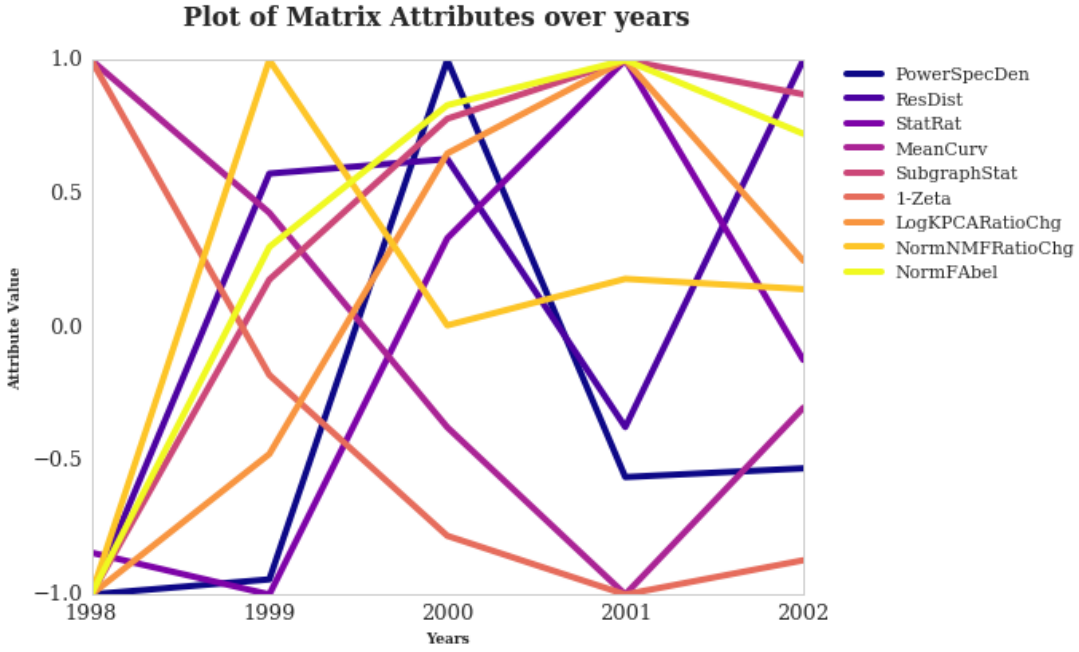

```
In [157]: plt.figure(figsize=(8,8))

          plt.subplot(2,1,1)
          plt.title("Plot of Music Attributes over months", fontsize=16)
          plt.plot(attvol_m.ZeroCrossRate.values)
          plt.xlabel("Months", fontsize=12)
          plt.ylabel("Zero Crossing Rate")
          plt.xticks(np.arange(len(months)), months, rotation=90)
          plt.yticks(fontsize=12)
          plt.autoscale()

          plt.subplot(2,1,2)
          plt.plot(attvol_m.LogSpecCentroid.values)
```

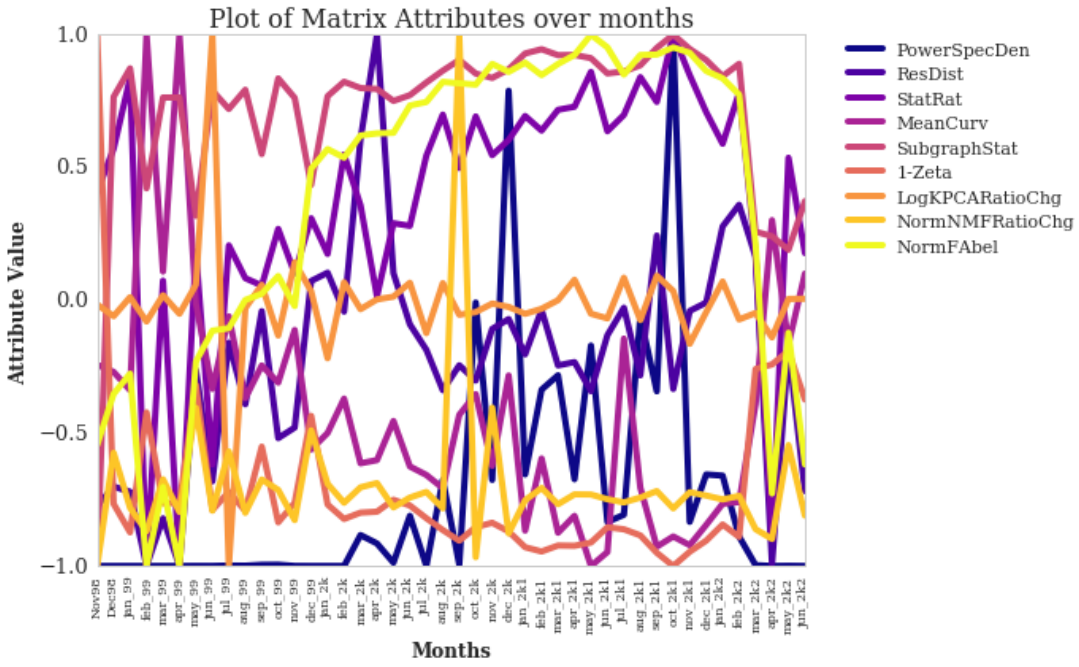
```
plt.xlabel("Months", fontsize=12)
plt.ylabel("Spectral Centroid")
plt.xticks(np.arange(len(months)), months, rotation=90)
plt.yticks(fontsize=12)
plt.autoscale()
plt.tight_layout()
plt.savefig('images/musicatt_mth.png')
```



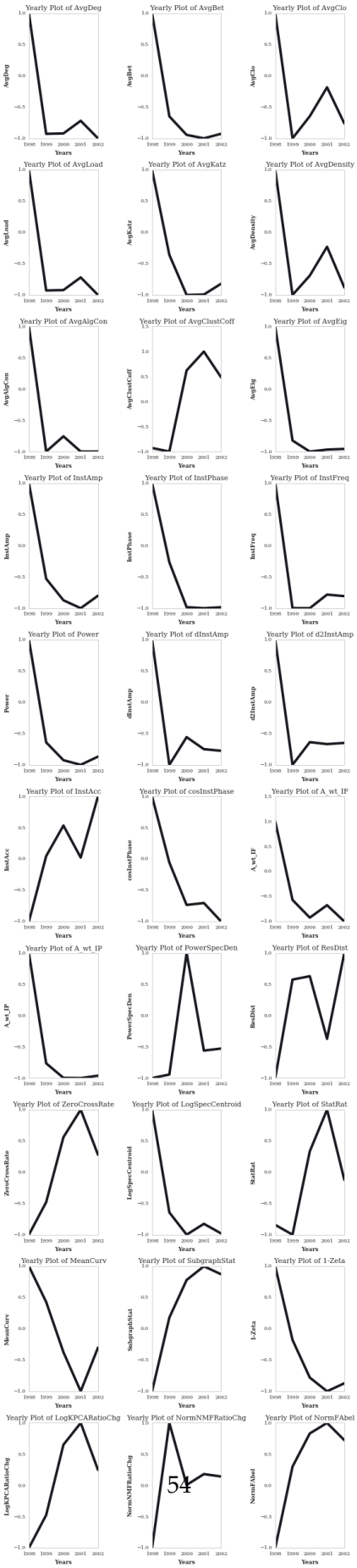

```
In [160]: attvol_y.drop(['ZeroCrossRate','LogSpecCentroid'], axis=1).iloc[:,19:].pl
          plt.xlabel("Years")
          plt.ylabel("Attribute Value")
          plt.legend(fontsize=11, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.
          plt.xticks(np.arange(len(years)),years,fontsize=13)
```

```
plt.yticks(fontsize=13)
plt.suptitle("Plot of Matrix Attributes over years", fontsize=16)
plt.autoscale()

attvol_m.drop(['ZeroCrossRate','LogSpecCentroid'], axis=1).iloc[:,19:].pl
plt.xlabel("Months", fontsize=12)
plt.ylabel("Attribute Value", fontsize=12)
plt.legend(fontsize=11, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.
plt.xticks(np.arange(len(months)), months)
plt.yticks(fontsize=13)
plt.title("Plot of Matrix Attributes over months", fontsize=16)
plt.autoscale()
```

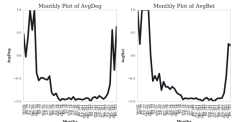




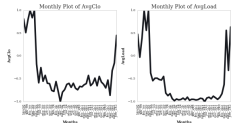

```
In [56]: all_cols = list(attvol_m.columns)

In [57]: attvol_y.shape

Out[57]: (5, 30)

In [162]: plt.figure(figsize=(8,38))

          for i in range(len(all_cols)):
              plt.subplot(11,3,i+1)
              attvol_y.iloc[:,i].plot.line(use_index=True)
              plt.xlabel("Years")
              plt.ylabel(all_cols[i])
              plt.xticks(np.arange(len(years)), years)
              plt.title("Yearly Plot of " +all_cols[i],fontsize=11)
              plt.tight_layout()
          plt.savefig('images/avg_allatt_yrs.png')
```


Yearly Plot of AvgDeg
AvgDeg
Yearly Plot of AvgBet
AvgBet
Yearly Plot of AvgClo
AvgClo
Yearly Plot of AvgLoad
AvgLoad
Yearly Plot of AvgKatz
AvgKatz
Yearly Plot of AvgDensity
AvgDensity
Yearly Plot of AvgAlgCon
AvgAlgCon
Yearly Plot of AvgClustCoff
AvgClustCoff
Yearly Plot of AvgEig
AvgEig
Yearly Plot of InstAmp
InstAmp
Yearly Plot of InstPhase
InstPhase
Yearly Plot of InstFreq
InstFreq
Yearly Plot of Power
Power
Yearly Plot of dInstAmp
dInstAmp
Yearly Plot of d2InstAmp
d2InstAmp
Yearly Plot of InstAcc
InstAcc
Yearly Plot of cosInstPhase
cosInstPhase
Yearly Plot of A_wt_IF
A_wt_IF
Yearly Plot of A_wt_IP
A_wt_IP
Yearly Plot of PowerSpecDen
PowerSpecDen
Yearly Plot of ResDist
ResDist
Yearly Plot of ZeroCrossRate
ZeroCrossRate
Yearly Plot of LogSpecCentroid
LogSpecCentroid
Yearly Plot of StatRat
StatRat
Yearly Plot of MeanCurv
MeanCurv
Yearly Plot of SubgraphStat
SubgraphStat
Yearly Plot of 1-Zeta
1-Zeta
Yearly Plot of LogKPCARatioChg
LogKPCARatioChg
Yearly Plot of NormNMFRatioChg
NormNMFRatioChg
Yearly Plot of NormFAbel
NormFAbel
Years

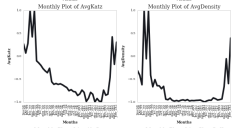
In [163]:
```
plt.figure(figsize=(8,64))

for i in range(len(all_cols)):
    plt.subplot(15,2,i+1)
    attvol_m.iloc[:,i].plot.line(rot=90)
    plt.xlabel("Months")
    plt.ylabel(all_cols[i])
    plt.xticks(np.arange(len(months)), months)
    plt.title("Monthly Plot of " +all_cols[i],fontsize=13)
    plt.tight_layout()
plt.savefig('images/avg_allatt_mth.png')
```


Monthly Plot of AvgDeg
AvgDeg
Months

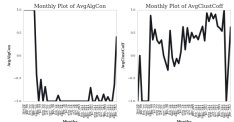
Monthly Plot of AvgBet
AvgBet
Months


Monthly Plot of AvgClo
AvgClo
Months

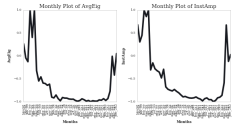
Monthly Plot of AvgLoad
AvgLoad
Months


Monthly Plot of AvgKatz
AvgKatz
Months

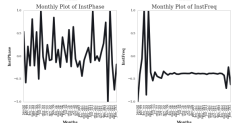
Monthly Plot of AvgDensity
AvgDensity
Months


Monthly Plot of AvgAlgCon
AvgAlgCon
Months

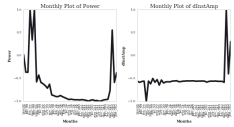
Monthly Plot of AvgClustCoff
AvgClustCoff
Months


Monthly Plot of AvgEig
AvgEig
Months

Monthly Plot of InstAmp
InstAmp
Months


Monthly Plot of InstPhase
InstPhase
Months

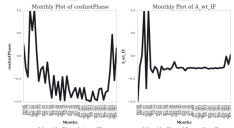
Monthly Plot of InstFreq
InstFreq
Months


Monthly Plot of Power
Power
Months

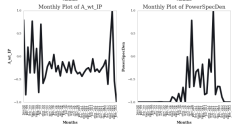
Monthly Plot of dInstAmp
dInstAmp
Months


Monthly Plot of d2InstAmp
d2InstAmp
Months

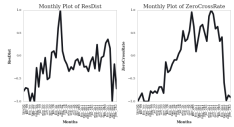
Monthly Plot of InstAcc
InstAcc
Months


Monthly Plot of cosInstPhase
cosInstPhase
Months

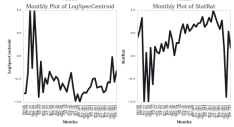
Monthly Plot of A_wt_IF
A_wt_IF
Months


Monthly Plot of A_wt_IP
A_wt_IP
Months

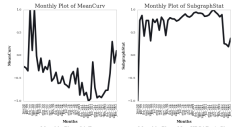
Monthly Plot of PowerSpecDen
PowerSpecDen
Months


Monthly Plot of ResDist
ResDist
Months

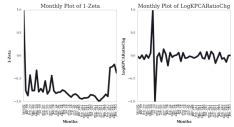
Monthly Plot of ZeroCrossRate
ZeroCrossRate
Months


Monthly Plot of LogSpecCentroid
LogSpecCentroid
Months

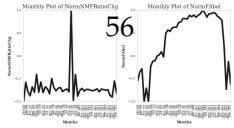
Monthly Plot of StatRot
StatRot
Months


Monthly Plot of MeanCurv
MeanCurv
Months

Monthly Plot of SubgraphDist
SubgraphDist
Months


Monthly Plot of 1-Zeta
1-Zeta
Months

Monthly Plot of LogKPCARatioChg
LogKPCARatioChg
Months


Monthly Plot of NormNMFRatioChg
NormNMFRatioChg
Months

Monthly Plot of NormFAfel
NormFAfel
Months

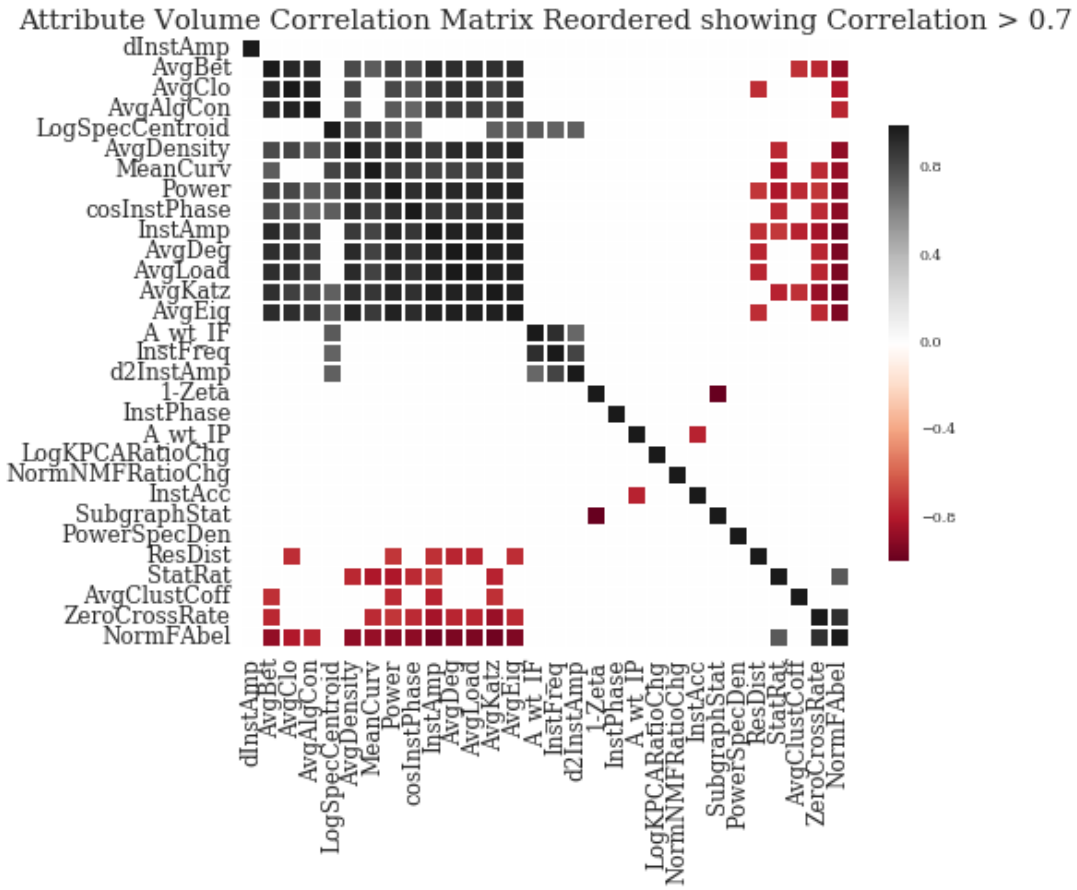
7 Correlation Analysis

```
In [165]: corr_m = attvol_m.corr()
          g = sns.clustermap(corr_m, metric='chebyshev')
          plt.close()
```

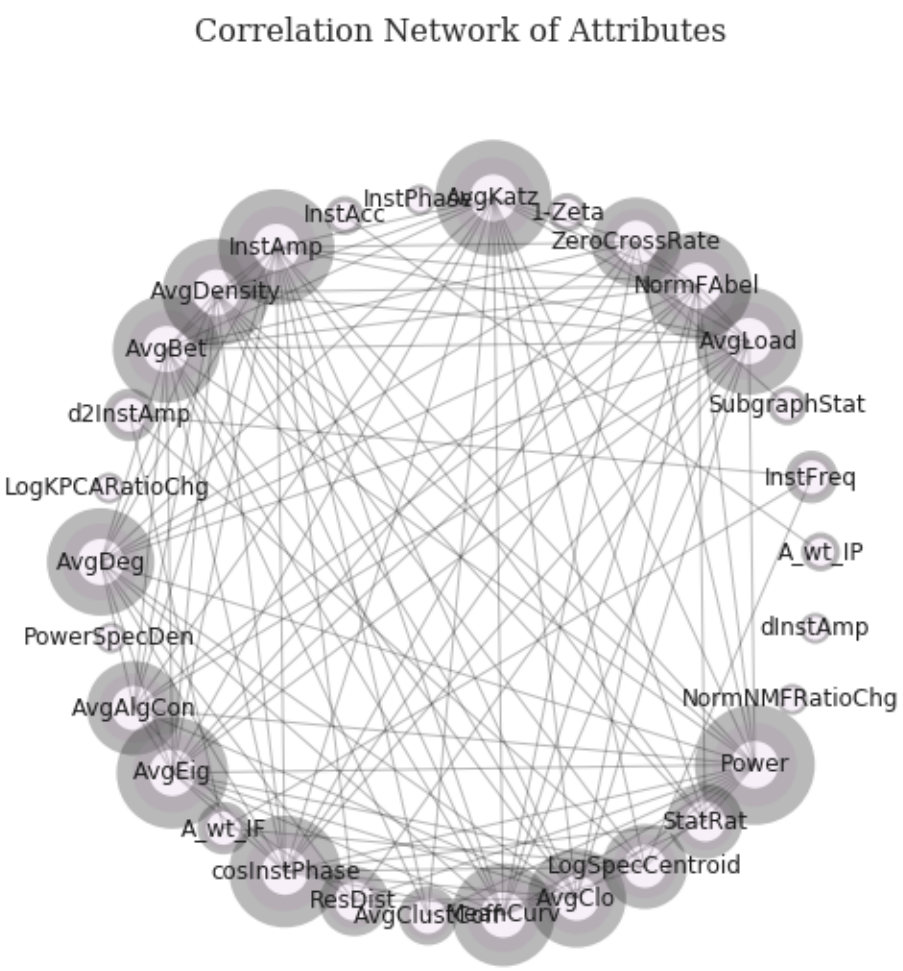
7.1 Correlation > 0.7

```
In [166]: threshold = 0.7
          corr_m.values[np.where(abs(corr_m.values) < threshold)] = 0
```

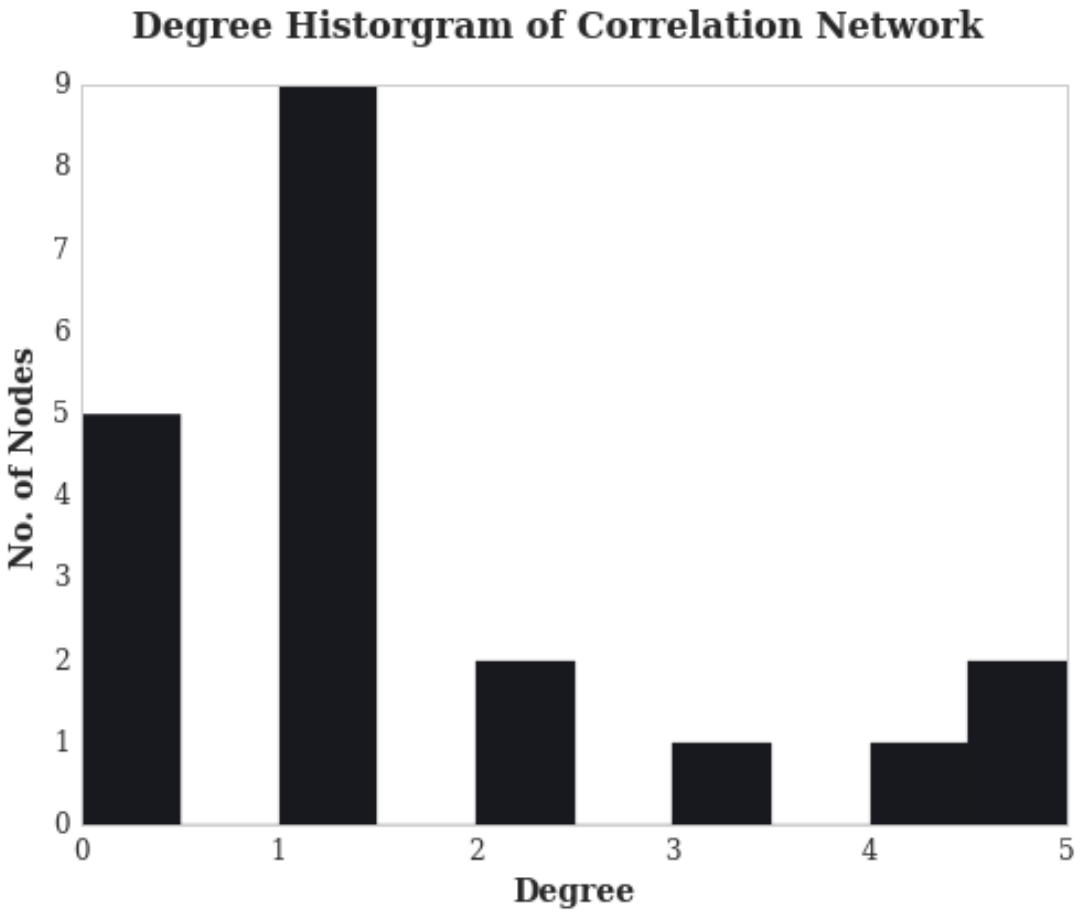
```
In [167]: plt.figure(figsize=(8,8))
          sns.heatmap(corr_m.iloc[g.dendrogram_col.reordered_ind, g.dendrogram_row.
                      cmap='RdGy',robust=True, fmt='d', linewidths=1, square=True,\
                      cbar_kws={'orientation':'vertical', 'shrink':0.5})
          plt.title("Attribute Volume Correlation Matrix Reordered showing Correlat
          plt.xticks(rotation=90, fontsize=12)
          plt.yticks(rotation=360, fontsize=12)
          plt.tight_layout()
          plt.savefig('images/reordered_corrmat.png')
```




```
In [169]: names = corr_m.index.values
          G_corr = nx.Graph(corr_m.values)
          pos=nx.fruchterman_reingold_layout(G_corr, iterations=1000, k=200)

In [172]: #ref: https://github.com/traims/correlation-based-networks/blob/master/cr
          components = nx.connected_components(G_corr)
          plt.figure(figsize=(8,8))
          plt.title("Correlation Network of Attributes", fontsize=16)
          plt.axis('off')
          for i in components:
              component = G_corr.subgraph(i)
              nx.draw_networkx(component, with_labels = True,node_size = [component
                  labels = dict([(x, names[x]) for x in component.nodes()]), pos=po
                               edge_color='k', node_color='#E9CFEC', linewidths=[com
                                fontsize=11)
          plt.tight_layout()
          plt.savefig('images/corrnet.png')
```




```
In [173]: plt.hist(nx.degree_histogram(G_corr))
          plt.suptitle("Degree Historgram of Correlation Network", fontsize=16)
          plt.xticks(fontsize=12)
          plt.yticks(fontsize=12)
          plt.xlabel("Degree", fontsize=14)
          plt.ylabel("No. of Nodes", fontsize=14)
          plt.savefig('images/corrnet_deghist.png')
```



8 Regression Analysis for Feature Ranking

```
In [66]: from sklearn.metrics import mean_squared_error
         from sklearn import ensemble

In [67]: def RMS(x, axis=None):
             rms = np.sqrt(np.mean(np.square(x), axis=axis))
             return rms

In [68]: X= attvol_m.drop(['AvgDeg'],axis=1)
         y = attvol_m.AvgDeg

In [70]: X[:5]

Out[70]:      AvgBet    AvgClo   AvgLoad   AvgKatz  AvgDensity  AvgAlgCon  \
         0  1.000000  0.810523  0.475524  0.267944   -0.318532        1.0
         1  0.250903  0.509903 -0.027972  0.069435   -0.419219        1.0
         2  1.000000  0.785809  0.388112  0.282044   -0.622808        1.0
```

```
3  1.000000  1.000000  1.000000  1.000000    1.000000        1.0
4  1.000000  0.835787  0.559441  0.423789   -0.054825        1.0

   AvgClustCoff    AvgEig   InstAmp  InstPhase     ...       ResDist  \
0     -1.000000  0.262262  0.679834   0.535455     ...     -0.774637
1     -0.000165 -0.044277  0.305093  -0.578269     ...     -0.701077
2     -1.000000 -0.127135  0.450396   0.214700     ...     -0.720470
3     -1.000000  1.000000  1.000000  -0.207114     ...     -1.000000
4     -1.000000  0.403511  0.860240   0.812100     ...     -0.820587

   ZeroCrossRate  LogSpecCentroid   StatRat  MeanCurv  SubgraphStat    1-2
0      -1.000000        -0.815713  0.407864 -0.242635     -1.000000  1.000
1      -0.909091        -0.816057  0.570795 -0.269438      0.766117 -0.766
2      -0.818182        -0.371199  0.836955 -0.337965      0.873376 -0.873
3      -1.000000         1.000000 -1.000000  1.000000      0.421859 -0.421
4      -1.000000        -0.264125  0.074177  0.108395      0.763599 -0.763

   LogKPCARatioChg  NormNMFRatioChg  NormFAbel
0        -0.015620        -1.000000  -0.543745
1        -0.059551        -0.573875  -0.352246
2         0.012093        -0.780922  -0.276008
3        -0.079774        -0.869723  -1.000000
4         0.019315        -0.673657  -0.702049

[5 rows x 29 columns]
```

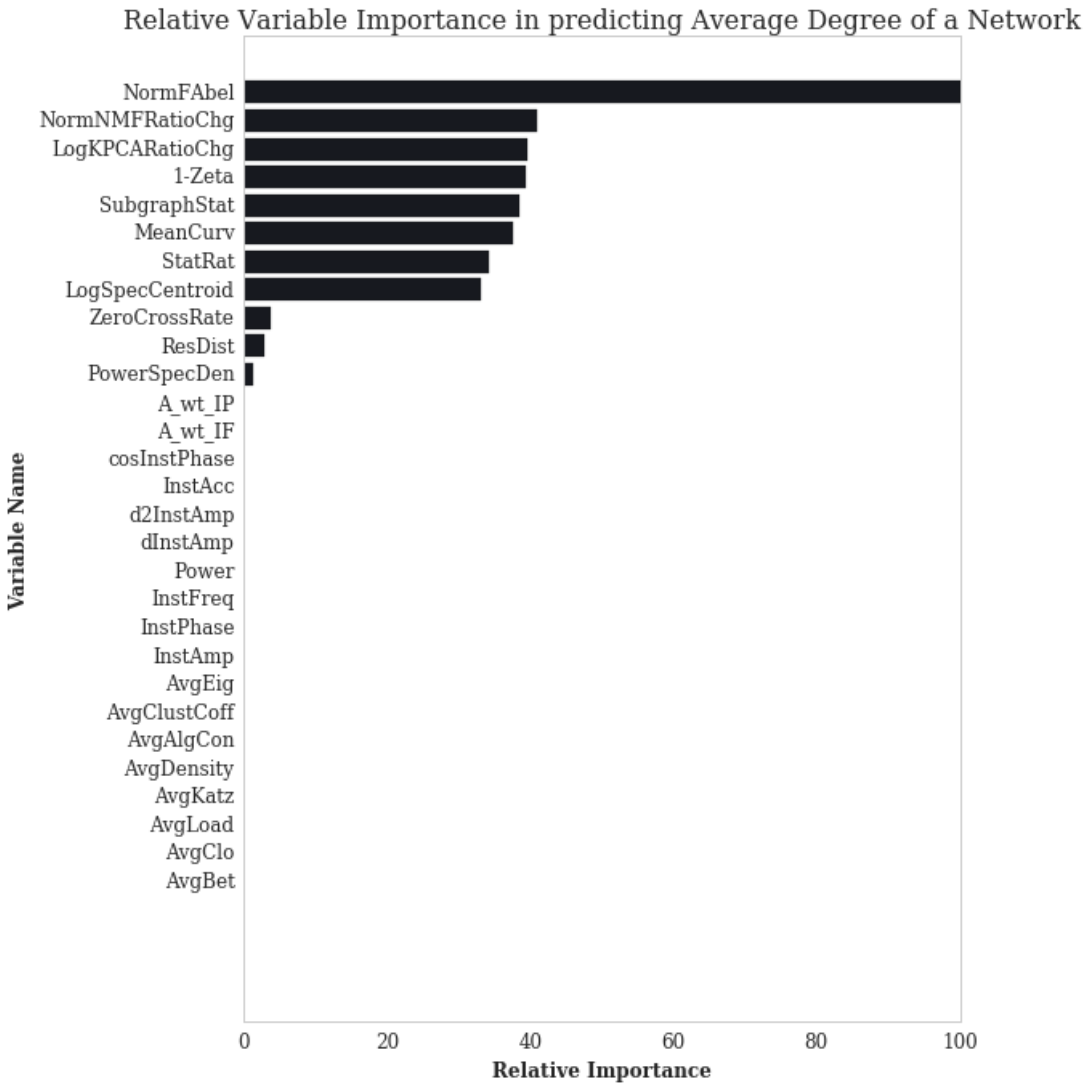
```
In [71]: X = X.astype(np.float32)
         offset = int(X.shape[0] * 0.5)
         X_train, y_train = X[:offset], y[:offset]
         X_test, y_test = X[offset:], y[offset:]
```

```
In [72]: params = {'n_estimators': 10, 'max_depth': 10, 'min_samples_split': 10,
                  'learning_rate': 0.1, 'loss': 'ls'}
         clf = ensemble.GradientBoostingRegressor(**params)
         clf.fit(X_train, y_train)
```

```
Out[72]: GradientBoostingRegressor(alpha=0.9, init=None, learning_rate=0.1, loss='l
                    max_depth=10, max_features=None, max_leaf_nodes=None,
                    min_samples_leaf=1, min_samples_split=10,
                    min_weight_fraction_leaf=0.0, n_estimators=10, presort='auto'
                    random_state=None, subsample=1.0, verbose=0, warm_start=False
```

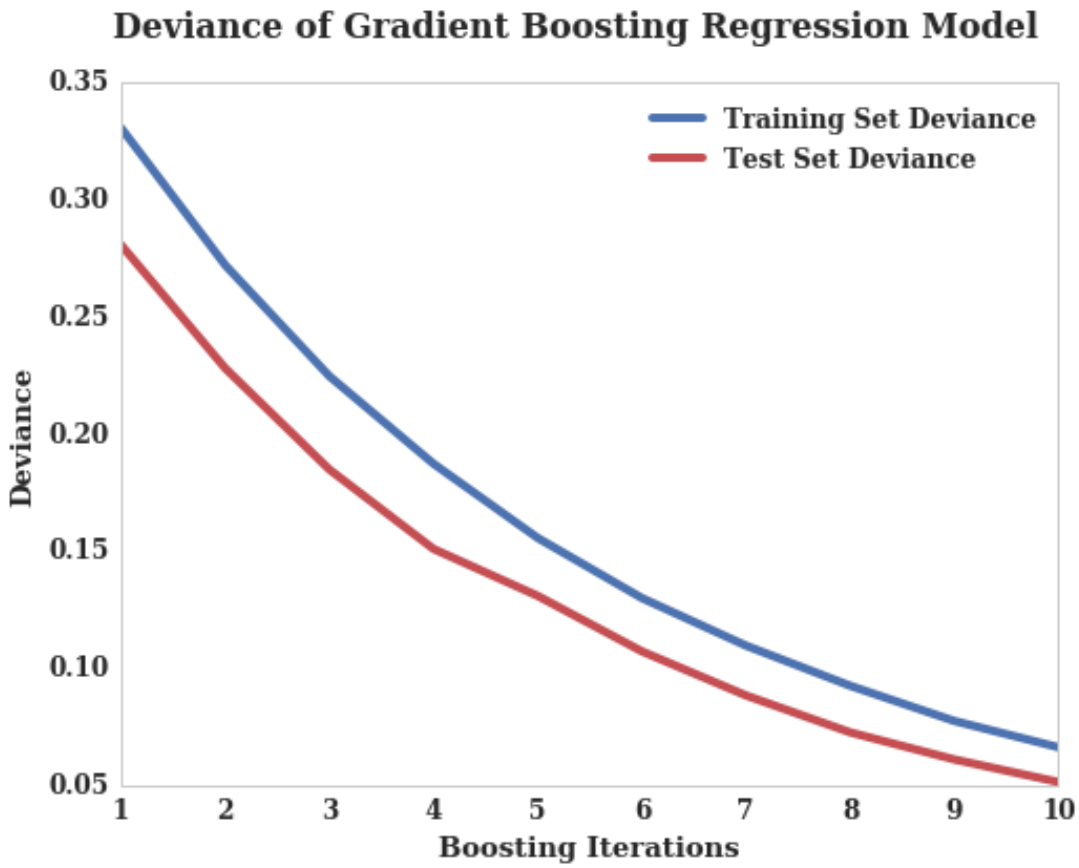
```
In [174]: feature_importance = clf.feature_importances_

          # make importances relative to max importance
          plt.figure(figsize=(8,11))
          feature_importance = 100.0 * (feature_importance / feature_importance.max
          sorted_idx = np.argsort(feature_importance)
          feature_names = list(X)
```

```
pos = np.arange(sorted_idx.shape[0])
plt.barh(pos, feature_importance[sorted_idx], align='center')
plt.yticks(pos, feature_names, fontsize=12)
plt.ylabel("Variable Name", fontsize=12)
plt.xlabel('Relative Importance', fontsize=12)
plt.title('Relative Variable Importance in predicting Average Degree of a
plt.xticks(fontsize=12)
plt.savefig('images/feature_ranking.png')
```



In [74]:
```
test_score = np.zeros((params['n_estimators'],), dtype=np.float64)

for i, y_pred in enumerate(clf.staged_predict(X_test)):
    test_score[i] = clf.loss_(y_test, y_pred)
```

```
plt.figure(figsize=(8, 6))
plt.suptitle('Deviance of Gradient Boosting Regression Model',fontsize=16)
plt.plot(np.arange(params['n_estimators']) + 1, clf.train_score_, 'b-',
         label='Training Set Deviance')
plt.plot(np.arange(params['n_estimators']) + 1, test_score, 'r-',
         label='Test Set Deviance')
plt.legend(loc=1, fontsize=12)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.xlabel('Boosting Iterations')
plt.ylabel('Deviance')
plt.savefig('images/reg_deviance.png')
```




```
In [75]: mse = mean_squared_error(y_test, clf.predict(X_test))
         print("Gradient Boosting MSE: " ,mse)

Gradient Boosting MSE:  0.0517054046745
```

9 Aggregation Measures

```
In [77]: persistence_m = attvol_m.mean(axis=1)/(attvol_m.shape[0]-1)
         persistence_y = attvol_y.mean(axis=1)/(attvol_y.shape[0]-1)
```

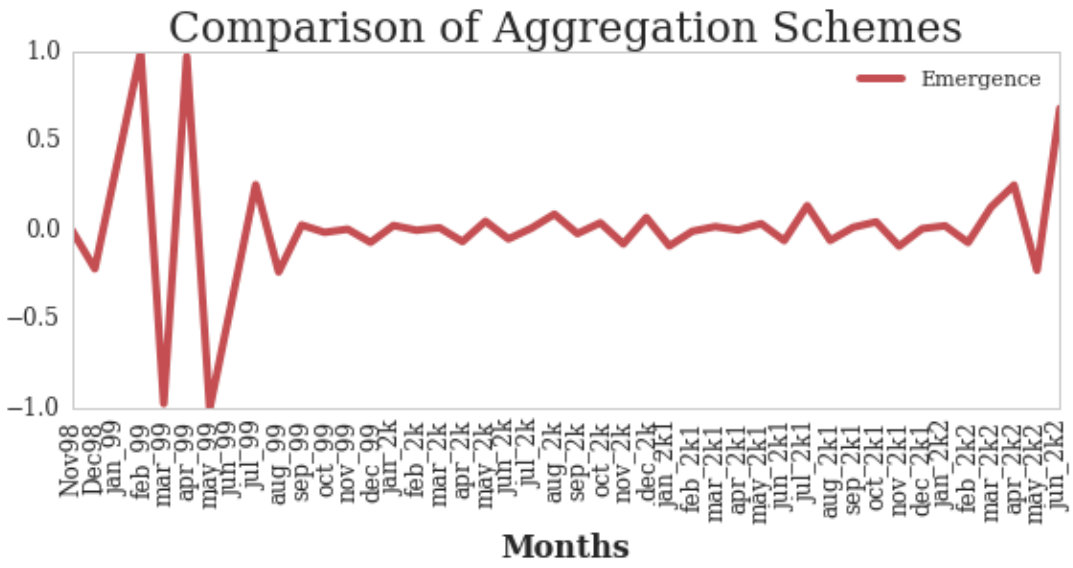
```
In [78]: def emergence(per):
             tmp = np.asarray(per)
             emerg = []
             for i in range(len(tmp)-1):
                 x= int(i)
                 y = x +1
                 #print(tmp[y], tmp[x])
                 if tmp[y]==tmp[x]:
                     emerg.append(0)
                 elif tmp[y] < (0) or tmp[x] < 0:
                     res = (tmp[y]-tmp[x])/(abs(tmp[y])+abs(tmp[x]))
                     emerg.append(res)
                 else:
                     res = (tmp[y]-tmp[x])/max([tmp[y],tmp[x]])
                     emerg.append(res)
             tmp2 = np.zeros(len(emerg)+1)
             tmp2[1:] = emerg

             return tmp2


         def NRMS(n):
             nrms = []
             for i in range(len(n)-1):
                 x= int(i)
                 y = x +1
                 a = n[x]
                 b = n[y]
                 nrms_ = np.divide((RMS(a-b)),(RMS(a)+RMS(b)))
                 nrms.append(nrms_)
             tmp2 = np.zeros(len(nrms)+1)
             tmp2[1:] = nrms

             return tmp2
```

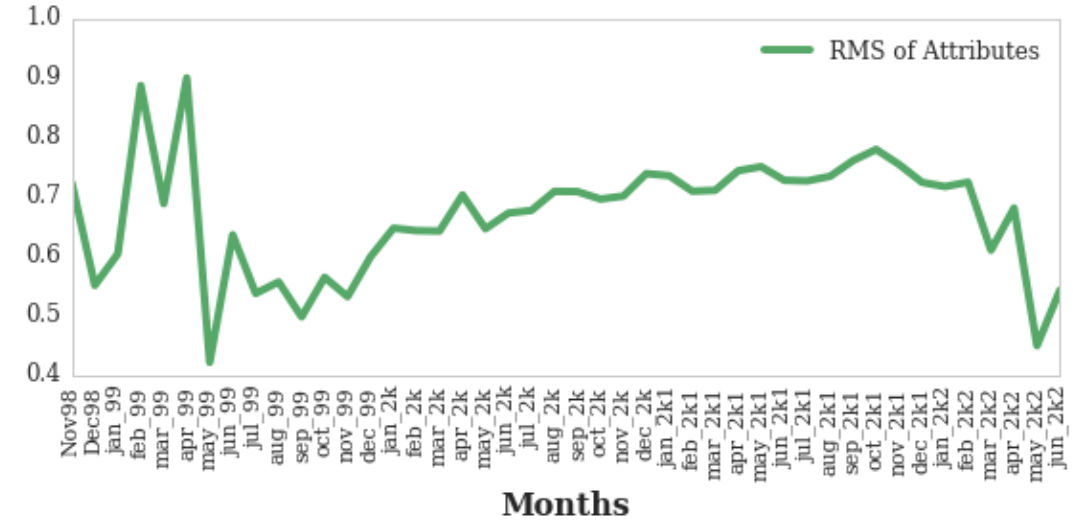
```
In [79]: rms_m = attvol_m.apply(lambda x: RMS(x), axis=1)
         nrms_m = NRMS(rms_m)
         emerg_m = emergence(persistence_m)
```

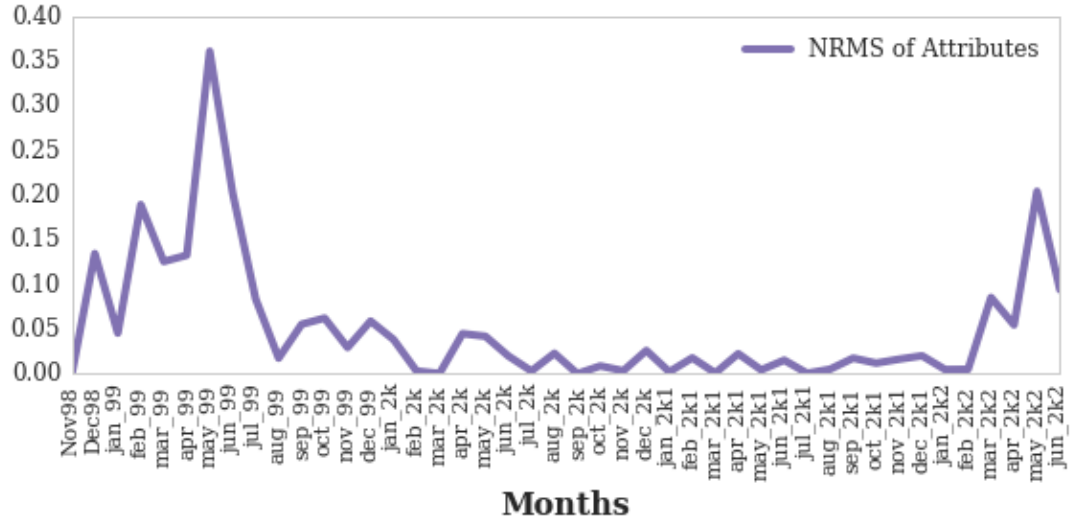
```
In [80]: emerg_y = emergence(persistence_y)
```

```
In [181]: plt.figure(figsize=(8,11))

          plt.subplot(3,1,1)
```

```
plt.plot(emerg_m,'r', label='Emergence')
plt.xticks(np.arange(len(months)), months,fontsize=12, rotation=90)
plt.yticks(fontsize=12)
plt.xlabel('Months', fontsize=16)
plt.legend(fontsize=11, loc=1)
plt.tight_layout()
plt.title("Comparison of Aggregation Schemes", fontsize=22)
plt.autoscale()

plt.subplot(3,1,2)
plt.plot(rms_m,'g',label='RMS of Attributes',)
plt.yticks(fontsize=12)
plt.xticks(np.arange(len(months)), months,fontsize=11, rotation=90)
plt.xlabel('Months', fontsize=16)
plt.legend(fontsize=12, loc=1)
plt.tight_layout()
plt.autoscale()

plt.subplot(3,1,3)
plt.plot(nrms_m,'m',label='NRMS of Attributes')

plt.xticks(np.arange(len(months)), months,fontsize=11, rotation=90)
plt.yticks(fontsize=12)
plt.xlabel('Months', fontsize=16)
plt.legend(fontsize=12, loc=1)
plt.autoscale()
plt.savefig('images/agg_comp.png')
```


Comparison of Aggregation Schemes
Emergence
1.0
0.5
0.0
−0.5
−1.0
Nov98 Dec98 jan_99 feb_99 mar_99 apr_99 may_99 jun_99 jul_99 aug_99 sep_99 oct_99 nov_99 dec_99 jan_2k feb_2k mar_2k apr_2k may_2k jun_2k jul_2k aug_2k sep_2k oct_2k nov_2k dec_2k jan_2k1 feb_2k1 mar_2k1 apr_2k1 may_2k1 jun_2k1 jul_2k1 aug_2k1 sep_2k1 oct_2k1 nov_2k1 dec_2k1 jan_2k2 feb_2k2 mar_2k2 apr_2k2 may_2k2 jun_2k2
Months


RMS of Attributes
1.0
0.9
0.8
0.7
0.6
0.5
0.4
Nov98 Dec98 jan_99 feb_99 mar_99 apr_99 may_99 jun_99 jul_99 aug_99 sep_99 oct_99 nov_99 dec_99 jan_2k feb_2k mar_2k apr_2k may_2k jun_2k jul_2k aug_2k sep_2k oct_2k nov_2k dec_2k jan_2k1 feb_2k1 mar_2k1 apr_2k1 may_2k1 jun_2k1 jul_2k1 aug_2k1 sep_2k1 oct_2k1 nov_2k1 dec_2k1 jan_2k2 feb_2k2 mar_2k2 apr_2k2 may_2k2 jun_2k2
Months

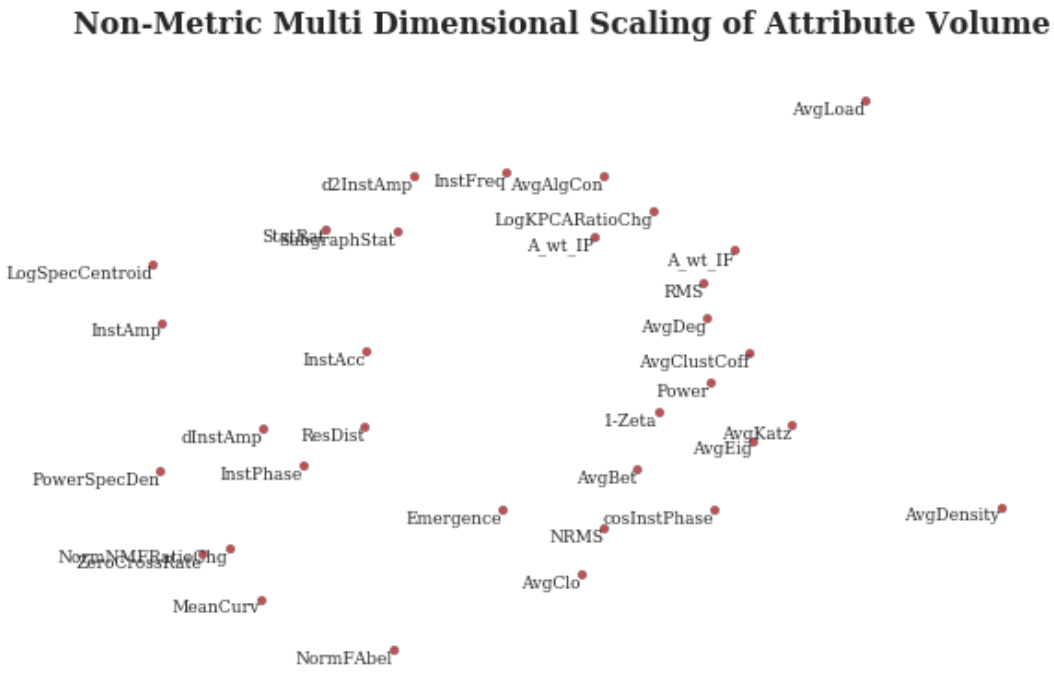

NRMS of Attributes
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Nov98 Dec98 jan_99 feb_99 mar_99 apr_99 may_99 jun_99 jul_99 aug_99 sep_99 oct_99 nov_99 dec_99 jan_2k feb_2k mar_2k apr_2k may_2k jun_2k jul_2k aug_2k sep_2k oct_2k nov_2k dec_2k jan_2k1 feb_2k1 mar_2k1 apr_2k1 may_2k1 jun_2k1 jul_2k1 aug_2k1 sep_2k1 oct_2k1 nov_2k1 dec_2k1 jan_2k2 feb_2k2 mar_2k2 apr_2k2 may_2k2 jun_2k2
Months

```
In [82]: final_attvol_m = attvol_m.copy()

         final_attvol_m['NRMS'] = nrms_m
         final_attvol_m['RMS']=rms_m
         final_attvol_m['Emergence'] =emerg_m
```

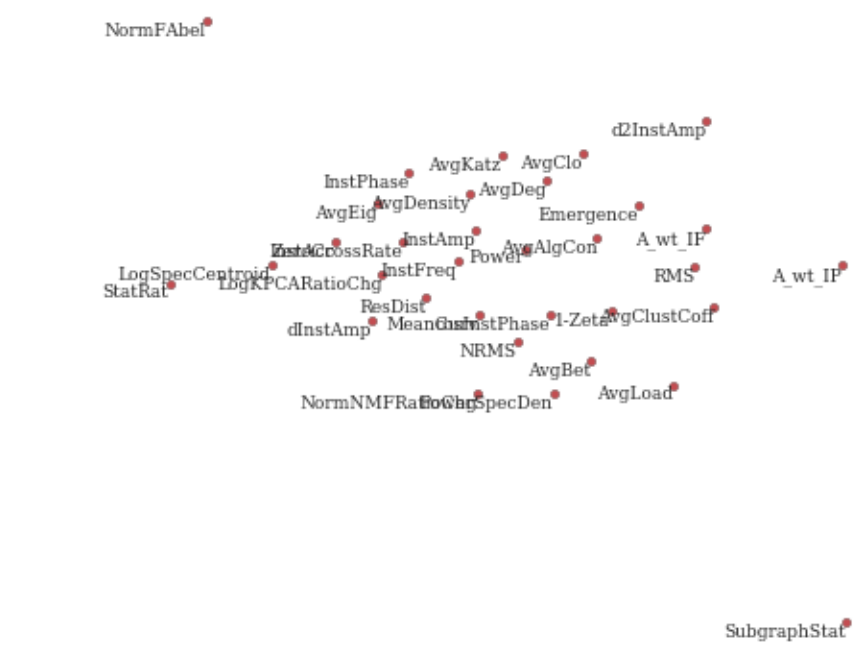
10 MDS and TSNE

```
In [83]: from sklearn.manifold import *
```

```
In [184]: mds = MDS(n_components=2, metric=False, random_state=0)
          Y = mds.fit_transform(final_attvol_m)

          plt.axis('off')
          plt.suptitle("Non-Metric Multi Dimensional Scaling of Attribute Volume",
          for i, txt in enumerate(final_attvol_m.columns):
              plt.scatter(Y[:, 1][i],Y[:, 0][i], c='r')
              plt.annotate(txt, (Y[:, 1][i],Y[:, 0][i]), horizontalalignment='right

          plt.autoscale()
          plt.tight_layout()
          plt.savefig('images/mds_plot.png')
```

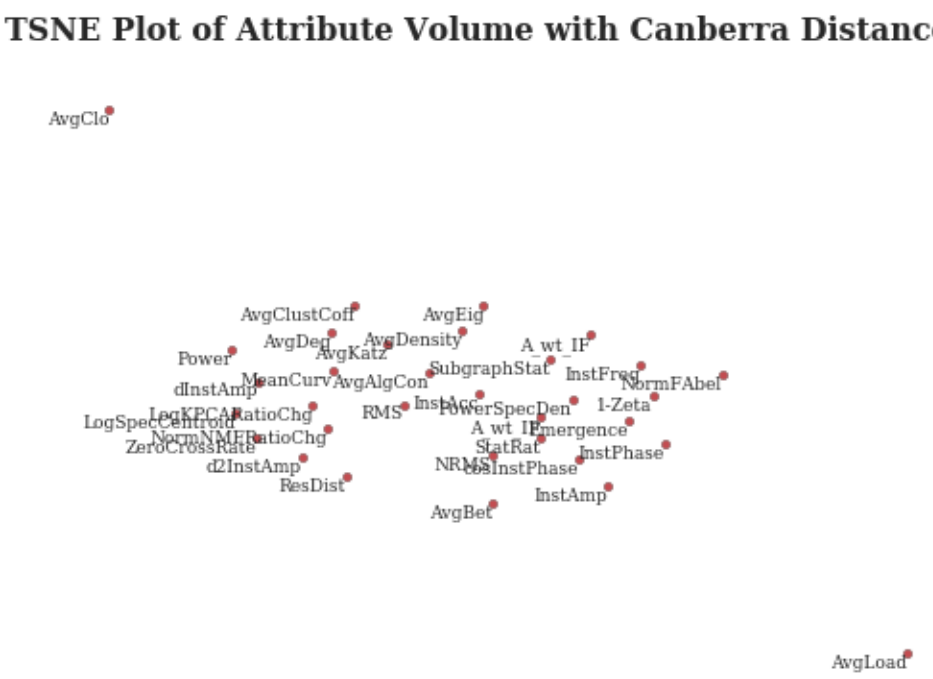


In [185]:
```
tsne = TSNE(metric='euclidean', random_state=0)

y_tsne = tsne.fit_transform(final_attvol_m)

plt.axis('off')
plt.suptitle("TSNE Plot of Attribute Volume with Euclidean Distance", for
for i, txt in enumerate(final_attvol_m.columns):
    plt.scatter(y_tsne[:, 1][i],y_tsne[:, 0][i], c='r')
    plt.annotate(txt, (y_tsne[:, 1][i],y_tsne[:, 0][i]), horizontalalignm

plt.autoscale()
plt.tight_layout()
plt.savefig('images/tsne_plot_euc.png')
```

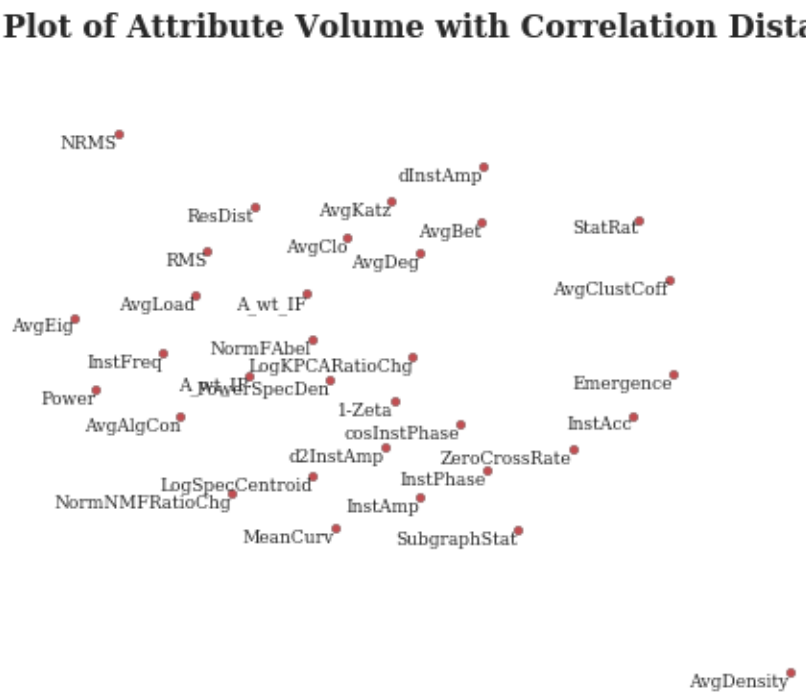


In [186]:
```
tsne = TSNE(metric='canberra', random_state=0)

y_tsne = tsne.fit_transform(final_attvol_m)
plt.axis('off')
```

```
    plt.suptitle("TSNE Plot of Attribute Volume with Canberra Distance", font
    for i, txt in enumerate(final_attvol_m.columns):
        plt.scatter(y_tsne[:, 1][i],y_tsne[:, 0][i], c='r')
        plt.annotate(txt, (y_tsne[:, 1][i],y_tsne[:, 0][i]), horizontalalignm

    plt.autoscale()
    plt.tight_layout()
    plt.savefig('images/tsne_plot.png')
```




```
In [187]: tsne = TSNE(metric='correlation', random_state=0)

    y_tsne = tsne.fit_transform(final_attvol_m)

    plt.axis('off')
    plt.suptitle("TSNE Plot of Attribute Volume with Correlation Distance", f
    for i, txt in enumerate(final_attvol_m.columns):
        plt.scatter(y_tsne[:, 1][i],y_tsne[:, 0][i], c='r')
        plt.annotate(txt, (y_tsne[:, 1][i],y_tsne[:, 0][i]), horizontalalignm

    plt.autoscale()
    plt.tight_layout()
    plt.savefig('images/tsne_plot_corr.png')
```



11 FK and Radon Plot

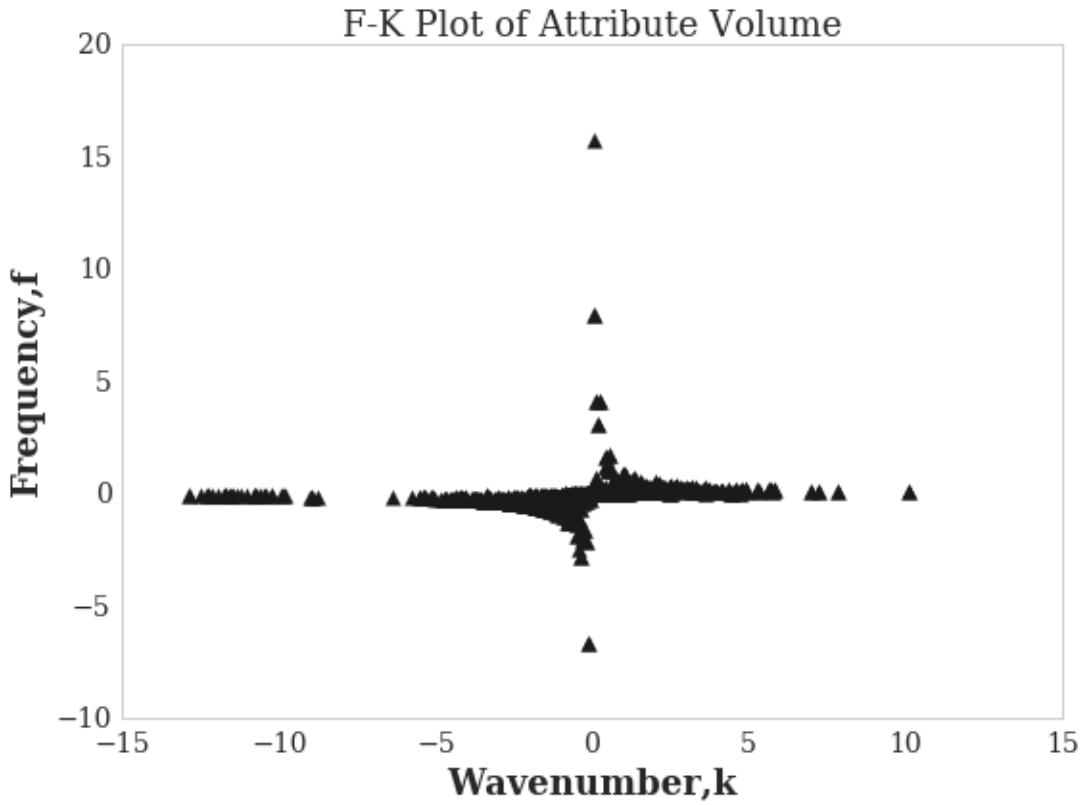
```
In [191]: def radon(m):
              from skimage.transform import radon
              theta = np.linspace(0., 180., max(m.shape), endpoint=False)
              sinogram = radon(m, theta=theta, circle=True)
              return sinogram
```

```
In [232]: def fk_plot(f):
              freq = sc.fft(f)
              wavnum = 1/freq

              return [freq, wavnum]
```

```
In [234]: f,k, = fk_plot(final_attvol_m)
          plt.scatter(f,k, s=60, marker='^', c='k')
          plt.title("F-K Plot of Attribute Volume", fontsize=18)
          plt.xticks(fontsize=14)
          plt.yticks(fontsize=14)
          plt.xlabel("Wavenumber,k", fontsize=18)
```

```
        plt.ylabel("Frequency,f", fontsize=18)
        plt.savefig('images/fkplot.png')
```

```
/home/arshad/anaconda3/lib/python3.5/site-packages/numpy/core/numeric.py:533: Compl
  return array(a, dtype, copy=False, order=order, subok=True)
/home/arshad/anaconda3/lib/python3.5/site-packages/matplotlib/figure.py:1742: UserW
  warnings.warn("This figure includes Axes that are not "
```

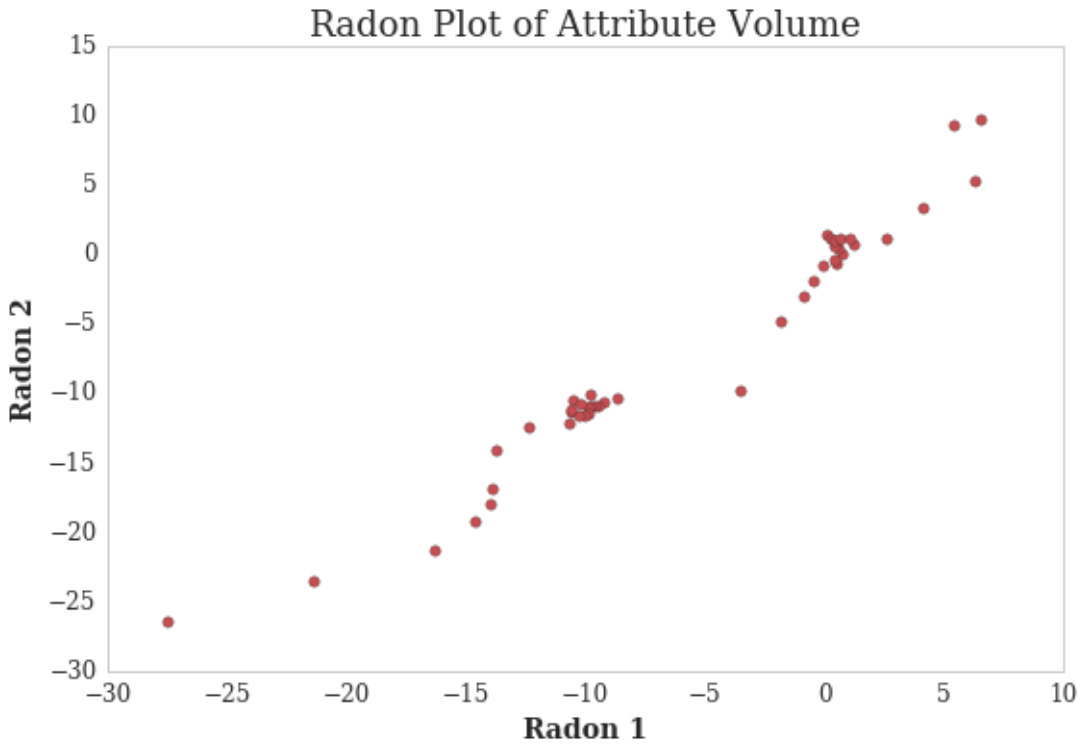


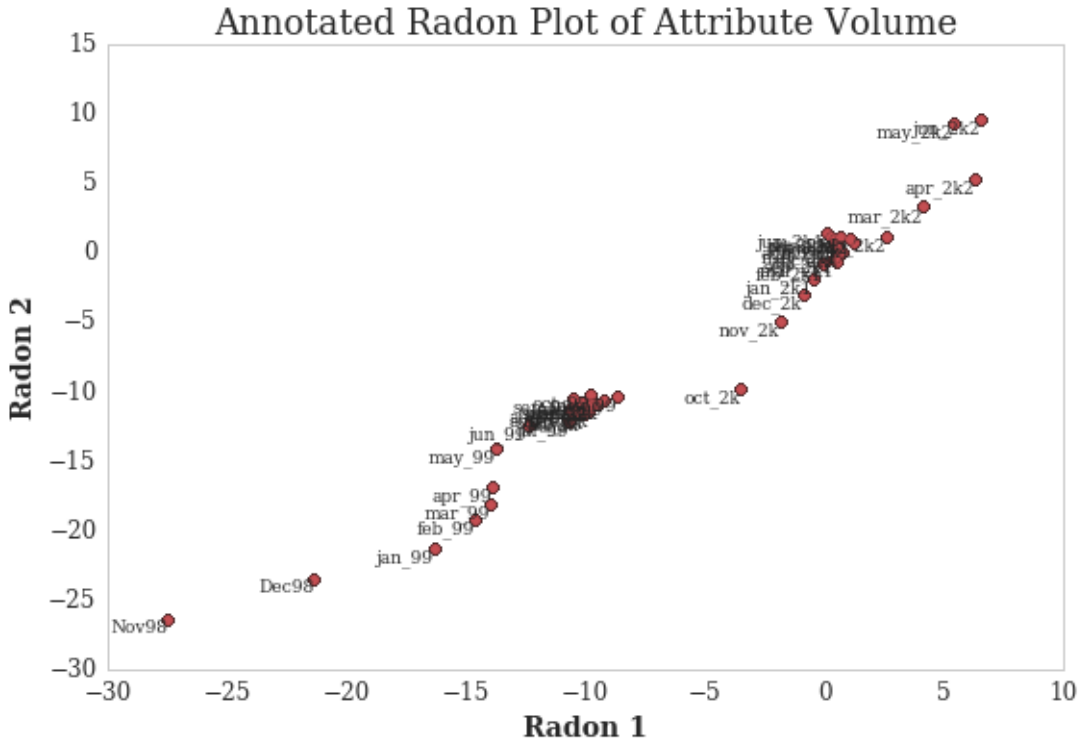
In [192]:
```
sgram = radon(final_attvol_m.values)
```

```
/home/arshad/anaconda3/lib/python3.5/site-packages/skimage/transform/radon_transfor
  warn('Radon transform: image must be zero outside the '
```

In [190]:
```
plt.figure(figsize=(8,11))

        plt.subplot(211)
        plt.scatter(sgram[0],sgram[1], c='r', s=30)
        plt.xticks(fontsize=12)
        plt.yticks(fontsize=12)
        plt.xlabel("Radon 1", fontsize=14)
```

```
plt.ylabel("Radon 2", fontsize=14)
plt.title("Radon Plot of Attribute Volume", fontsize=18)
plt.tight_layout()

plt.subplot(212)
plt.title("Annotated Radon Plot of Attribute Volume", fontsize=18)
for i, txt in enumerate(months):
    plt.scatter(sgram[0],sgram[1], c='r', s=30)
    plt.annotate(txt, (sgram[0][i],sgram[1][i]), horizontalalignment='rig
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.xlabel("Radon 1", fontsize=14)
plt.ylabel("Radon 2", fontsize=14)
plt.tight_layout()
plt.savefig('images/radonplot.png')
```






```
In [194]: plt.figure(figsize=(8,11))
```

Appendix E

Jupyter Notebook: Attribute comparison: 3 Matrices

This notebook shows the comparison of attributes derived from 3 different graph matrices the Normalised Graph Laplacian, Adjacency and Modularity matrices.

DNA_06.2

August 29, 2016

1 Dynamic Network Analysis of Enron Email Network Comparisons

In [1]:
```
import pandas as pd
import numpy as np
import networkx as nx
import seaborn as sns
import matplotlib.pyplot as plt
import scipy as sc
import random
from scipy.signal import *
from numpy.linalg import *
from sklearn.decomposition import *
from sklearn.metrics import mean_squared_error
from sklearn import ensemble
#plotting parameters
%matplotlib inline
sns.set(style="whitegrid", color_codes=True, context='paper')
```

In [2]:
```
from matplotlib import rcParams
rcParams['font.family'] = 'serif'
rcParams['font.sans-serif'] = ['CMU Serif']
rcParams['font.weight']=['heavy']
import matplotlib.pyplot as plt
```

In [4]:
```
plt.rc('axes', grid=False, titlesize='large', labelsize='large',labelweight
plt.rc('lines', linewidth=4)
plt.rc('figure', figsize = (12,6),titlesize='large',titleweight='black')
plt.rc('font', weight='heavy', size=11)
plt.rc('grid',linewidth=5)
```

In [7]:
```
sns.set_palette(sns.cubehelix_palette(10, hue=0.3, reverse=True, rot=-0.55,
```

2 Get attribute data

In [71]:
```
lap = pd.read_excel('attribute_data/lap_att.xlsx')
adj = pd.read_excel('attribute_data/adj_att.xlsx')
mod = pd.read_excel('attribute_data/mod_att.xlsx')
```

In [72]:
```
lap.head()
```

Out[72]:

	AvgDeg	AvgBet	AvgClo	AvgLoad	AvgKatz	AvgDensity	AvgAlgCor
0	0.475524	1.000000	0.810523	0.475524	0.267944	-0.318532	1.0
1	-0.027972	0.250903	0.509903	-0.027972	0.069435	-0.419219	1.0
2	0.388112	1.000000	0.785809	0.388112	0.282044	-0.622808	1.0
3	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0
4	0.559441	1.000000	0.835787	0.559441	0.423789	-0.054825	1.0

	AvgClustCoff	AvgEig	InstAmp	...	StatRat	MeanCurv	\
0	-1.000000	0.177194	0.679834	...	0.407864	-0.242635	
1	-0.000165	-0.044891	0.305093	...	0.570795	-0.269438	
2	-1.000000	-0.127797	0.450396	...	0.836955	-0.337965	
3	-1.000000	1.000000	1.000000	...	-1.000000	1.000000	
4	-1.000000	0.403160	0.860240	...	0.074177	0.108395	

	SubgraphStat	1-Zeta	LogKPCARatioChg	NormNMFRatioChg	NormFAbel	\
0	-1.000000	1.000000	-0.015620	-1.000000	-0.543745	
1	0.766117	-0.766117	-0.059551	-0.573875	-0.352246	
2	0.873376	-0.873376	0.012093	-0.780922	-0.276008	
3	0.421859	-0.421859	-0.079774	-0.869723	-1.000000	
4	0.763599	-0.763599	0.019315	-0.673657	-0.702049	

	NRMS	RMS	Emergence
0	0.000000	0.722464	0.000000
1	0.134441	0.551228	-0.205409
2	0.045739	0.604070	0.390743
3	0.189878	0.887236	1.000000
4	0.126025	0.688637	-0.972473

[5 rows x 33 columns]

In [73]:
```
coldrop = lap.columns[:9]
```

In [74]:
```
lap.drop(coldrop, axis=1, inplace=True)
mod.drop(coldrop, axis=1, inplace=True)
adj.drop(coldrop, axis=1, inplace=True)
```

In [75]:
```
lap.head()
```

Out[75]:

	InstAmp	InstPhase	InstFreq	Power	dInstAmp	d2InstAmp	InstAcc
0	0.679834	0.535455	-1.000000	0.021345	-0.565983	-0.539313	-0.707491
1	0.305093	-0.578269	-0.393695	-0.364336	-0.587010	-0.455499	0.708190
2	0.450396	0.214700	-0.052958	-0.367204	-0.566478	-0.483079	-0.006052
3	1.000000	-0.207114	1.000000	1.000000	-0.560284	1.000000	0.103005
4	0.860240	0.812100	-0.845407	0.340239	-1.000000	0.122943	-0.161020

	cosInstPhase	A_wt_IF	A_wt_IP	...	StatRat	MeanCurv	\
0	0.249334	-1.000000	0.796844	...	0.407864	-0.242635	

```
1     -0.261313 -0.235448 -0.838336    ...      0.570795 -0.269438
2     -0.457223  0.015942  0.204171    ...      0.836955 -0.337965
3      1.000000  1.000000 -0.359857    ...     -1.000000  1.000000
4      0.560408 -0.708792  0.771960    ...      0.074177  0.108395

   SubgraphStat    1-Zeta  LogKPCARatioChg  NormNMFRatioChg  NormFAbel  \
0     -1.000000  1.000000        -0.015620        -1.000000  -0.543745
1      0.766117 -0.766117        -0.059551        -0.573875  -0.352246
2      0.873376 -0.873376         0.012093        -0.780922  -0.276008
3      0.421859 -0.421859        -0.079774        -0.869723  -1.000000
4      0.763599 -0.763599         0.019315        -0.673657  -0.702049

       NRMS       RMS  Emergence
0  0.000000  0.722464   0.000000
1  0.134441  0.551228  -0.205409
2  0.045739  0.604070   0.390743
3  0.189878  0.887236   1.000000
4  0.126025  0.688637  -0.972473

[5 rows x 24 columns]
```

In [76]:
```
mod.max()
```

Out[76]:
```
InstAmp            1.000000
InstPhase          1.000000
InstFreq           1.000000
Power              1.000000
dInstAmp           1.000000
d2InstAmp          1.000000
InstAcc            1.000000
cosInstPhase       1.000000
A_wt_IF            1.000000
A_wt_IP            1.000000
PowerSpecDen       1.000000
ResDist            1.000000
ZeroCrossRate      1.000000
LogSpecCentroid    1.000000
StatRat            1.000000
MeanCurv           1.000000
SubgraphStat       1.000000
1-Zeta             1.000000
LogKPCARatioChg    1.000000
NormNMFRatioChg    1.000000
NormFAbel          1.000000
NRMS               0.193777
RMS                0.745719
Emergence          1.000000
dtype: float64
```

In [77]:
```
months = ['Nov98', 'Dec98', 'jan_99', 'feb_99', 'mar_99', 'apr_99', 'may_9
          'nov_99', 'dec_99', 'jan_2k', 'feb_2k', 'mar_2k', 'apr_2k', 'may
          'nov_2k', 'dec_2k', 'jan_2k1', 'feb_2k1', 'mar_2k1', 'apr_2k1',
          'oct_2k1', 'nov_2k1', 'dec_2k1', 'jan_2k2', 'feb_2k2', 'mar_2k2'
```

In [78]:
```
att_only = lap.join(adj,rsuffix='_Adj').join(mod,rsuffix='_Mod')
att_only.sortlevel(axis=1, inplace=True);
```

In [79]:
```
att_only.head()
```

Out[79]:

	1-Zeta	1-Zeta_Adj	1-Zeta_Mod	A_wt_IF	A_wt_IF_Adj	A_wt_IF_Mod	\
0	1.000000	1.000000	1.000000	-1.000000	-0.065973	-0.578144	
1	-0.766117	-0.500770	-0.562061	-0.235448	0.000403	-0.270839	
2	-0.873376	-0.808032	-0.688113	0.015942	-0.065973	-0.578144	
3	-0.421859	-0.009407	0.339333	1.000000	-0.065973	-0.578144	
4	-0.763599	-0.427517	-0.195048	-0.708792	0.101108	-0.578144	

	A_wt_IP	A_wt_IP_Adj	A_wt_IP_Mod	Emergence	...	\
0	0.796844	0.402595	0.448481	0.000000	...	
1	-0.838336	-1.000000	-1.000000	-0.205409	...	
2	0.204171	1.000000	0.448481	0.390743	...	
3	-0.359857	-0.245437	0.448481	1.000000	...	
4	0.771960	-0.359158	0.448481	-0.972473	...	

	ZeroCrossRate_Mod	cosInstPhase	cosInstPhase_Adj	cosInstPhase_Mod	\
0	-1.000000	0.249334	0.912276	0.869867	
1	-0.904762	-0.261313	-0.339427	-0.038949	
2	-0.952381	-0.457223	-0.597742	0.009933	
3	-0.952381	1.000000	0.795405	0.243197	
4	-0.952381	0.560408	0.741131	1.000000	

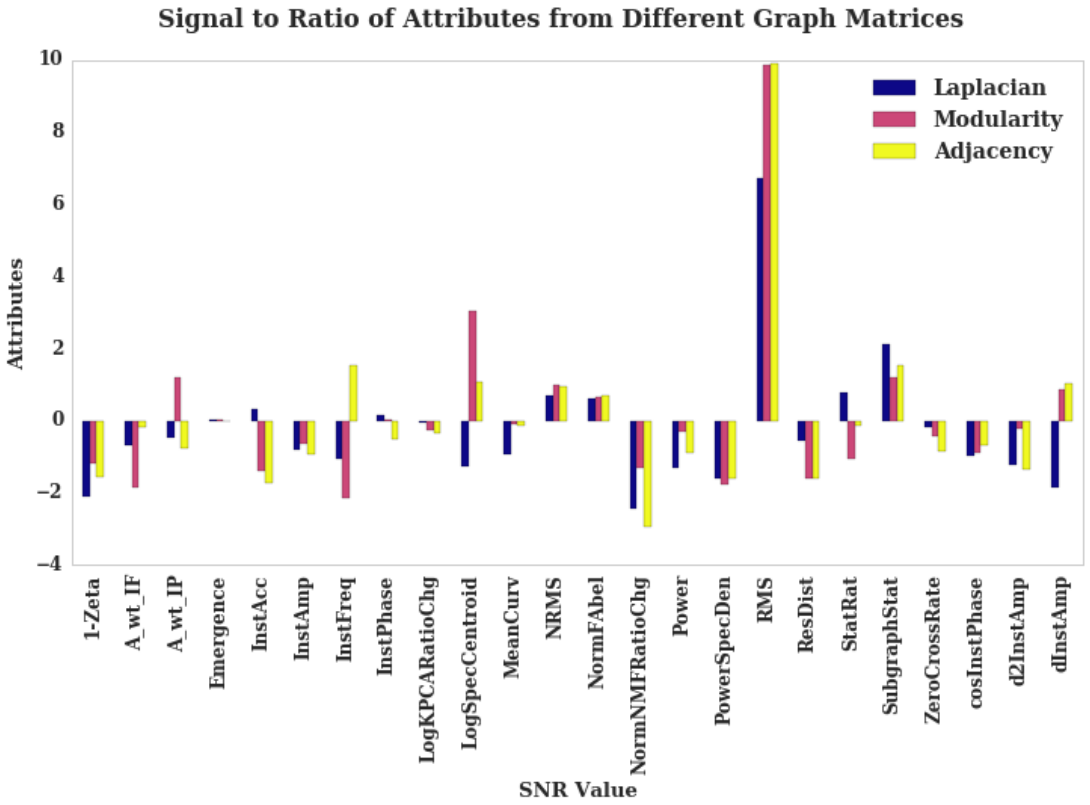
	d2InstAmp	d2InstAmp_Adj	d2InstAmp_Mod	dInstAmp	dInstAmp_Adj	\
0	-0.539313	-0.741787	-0.070128	-0.565983	0.375510	
1	-0.455499	-0.811824	0.594556	-0.587010	0.465306	
2	-0.483079	-0.602222	-0.070128	-0.566478	0.375510	
3	1.000000	1.000000	-0.070128	-0.560284	0.375510	
4	0.122943	0.342754	-0.070128	-1.000000	-0.107077	

	dInstAmp_Mod
0	0.197628
1	0.007620
2	0.197628
3	0.197628
4	0.197628

[5 rows x 72 columns]

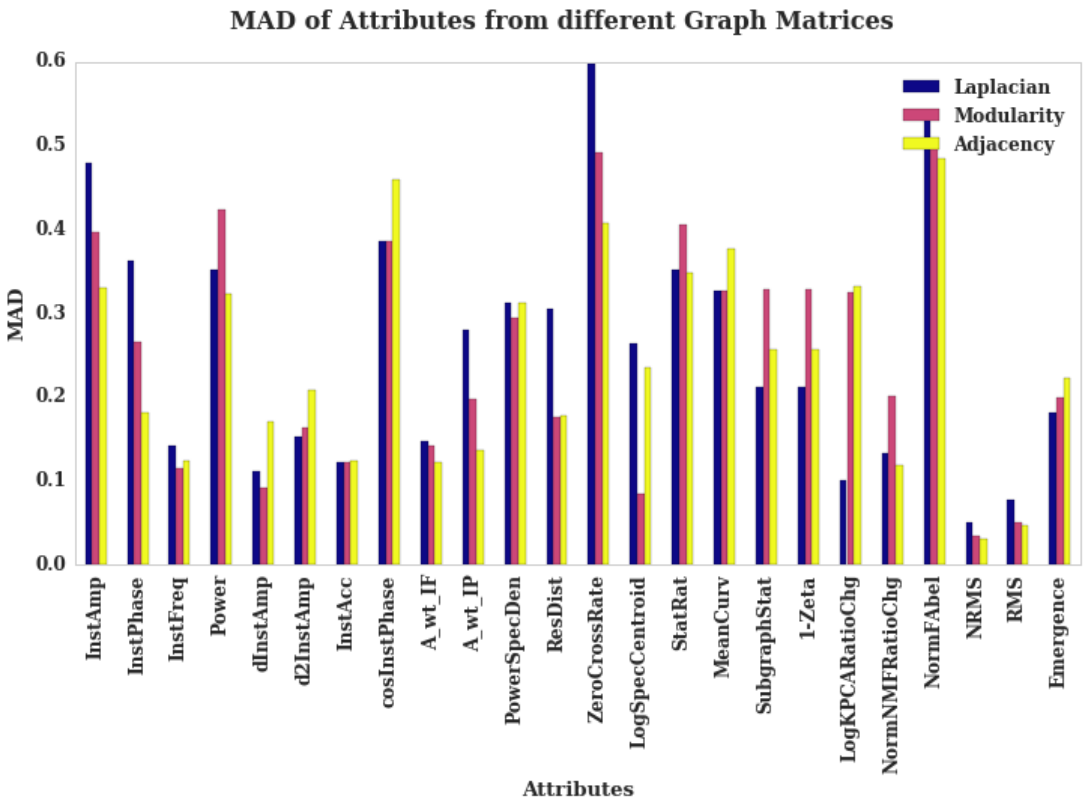
In [106]:
```
snr_lap = lap.mean()/lap.std()
snr_mod = mod.mean()/mod.std()
snr_adj = adj.mean()/adj.std()
```

```
In [111]: snr_df = pd.DataFrame([snr_lap,snr_mod,snr_adj])

In [114]: cname =  ['Laplacian','Modularity','Adjacency']

In [129]: snr_df =  snr_df.T
          snr_df.columns = cname
          snr_df.sortlevel(inplace=True)

In [248]: snr_df.plot.bar(figsize=(12,6), fontsize=12,cmap='plasma')
          plt.suptitle("Signal to Ratio of Attributes from Different Graph Matrices
          plt.legend(fontsize=14)
          plt.xlabel('SNR Value')
          plt.ylabel('Attributes')
          plt.savefig('images/snr_allatt_3mat.png')
```

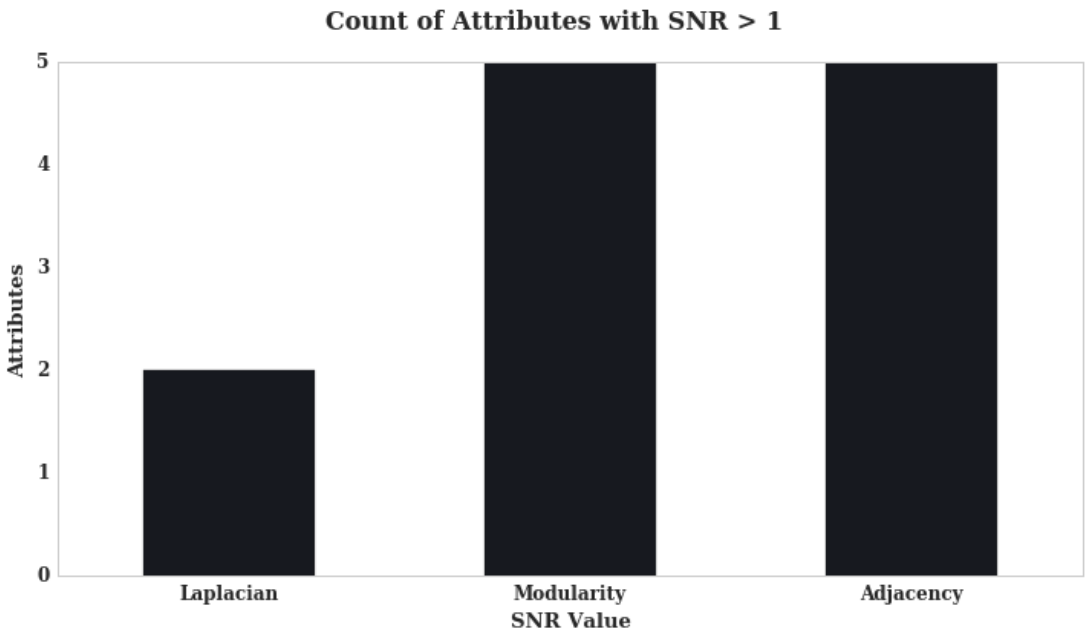



```
In [247]: mad_lap = lap.mad()
          mad_adj = adj.mad()
          mad_mod = mod.mad()

          mad_all = pd.DataFrame([mad_lap,mad_mod,mad_adj]).T
          mad_all.columns = ['Laplacian','Modularity','Adjacency']
```

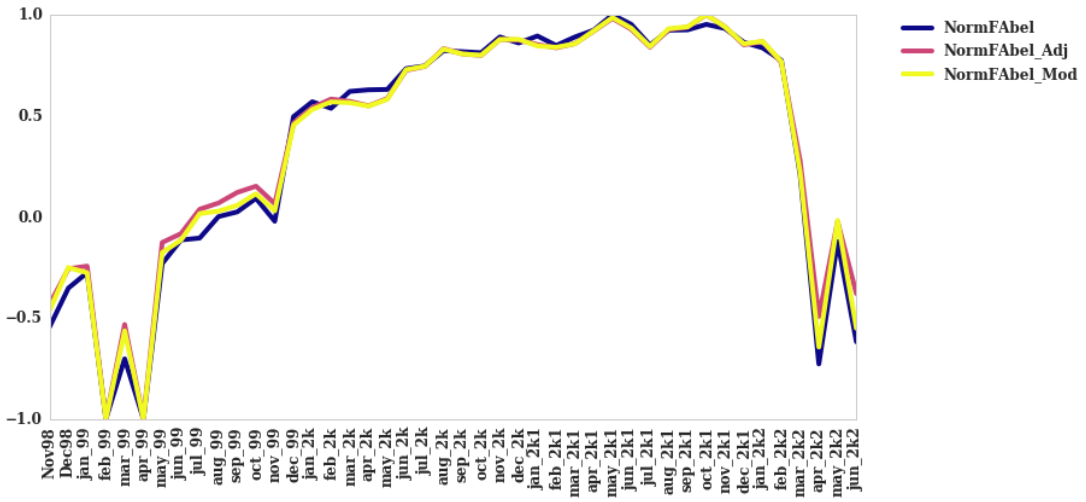
```
mad_all.plot.bar(fontsize=12, cmap='plasma',figsize=(12,6))
plt.suptitle('MAD of Attributes from different Graph Matrices', fontsize=
plt.xlabel('Attributes')
plt.ylabel('MAD')
plt.legend(loc=1,fontsize=12)
plt.savefig('images/mad_allatt_3mat.png')
```



In [232]:
```
snr_df[snr_df > 1].count().plot.bar(rot=0, fontsize=12)
plt.suptitle("Count of Attributes with SNR > 1", fontsize=16)
plt.xlabel('SNR Value')
plt.ylabel('Attributes')
plt.savefig('images/snrcount.png')
```




```
In [164]: mad_all.max()/2

Out[164]: Laplacian     0.299199
          Modularity    0.252281
          Adjacency     0.242036
          dtype: float64

In [168]:  att_only.columns[36:39]

Out[168]: Index(['NormFAbel', 'NormFAbel_Adj', 'NormFAbel_Mod'], dtype='object')

In [187]: att_only.iloc[:,36:39].plot.line(fontsize=12,cmap='plasma',rot=90)
          plt.xticks(np.arange(len(months)), months);
          plt.legend(fontsize=12, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.

Out[187]: <matplotlib.legend.Legend at 0x7f63f26db7f0>
```

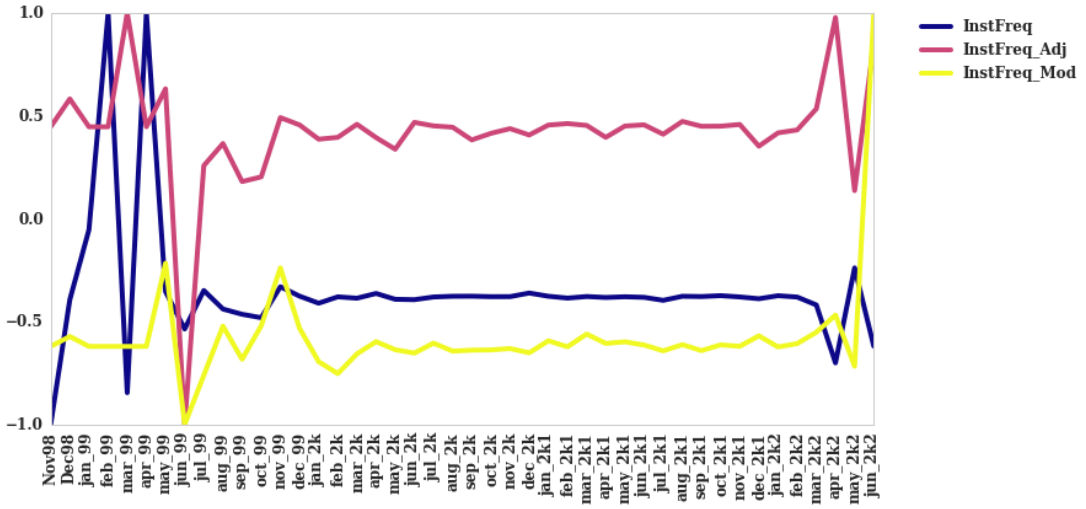


In [175]:
```
att_only.columns[18:21]
```

Out[175]:
```
Index(['InstFreq', 'InstFreq_Adj', 'InstFreq_Mod'], dtype='object')
```

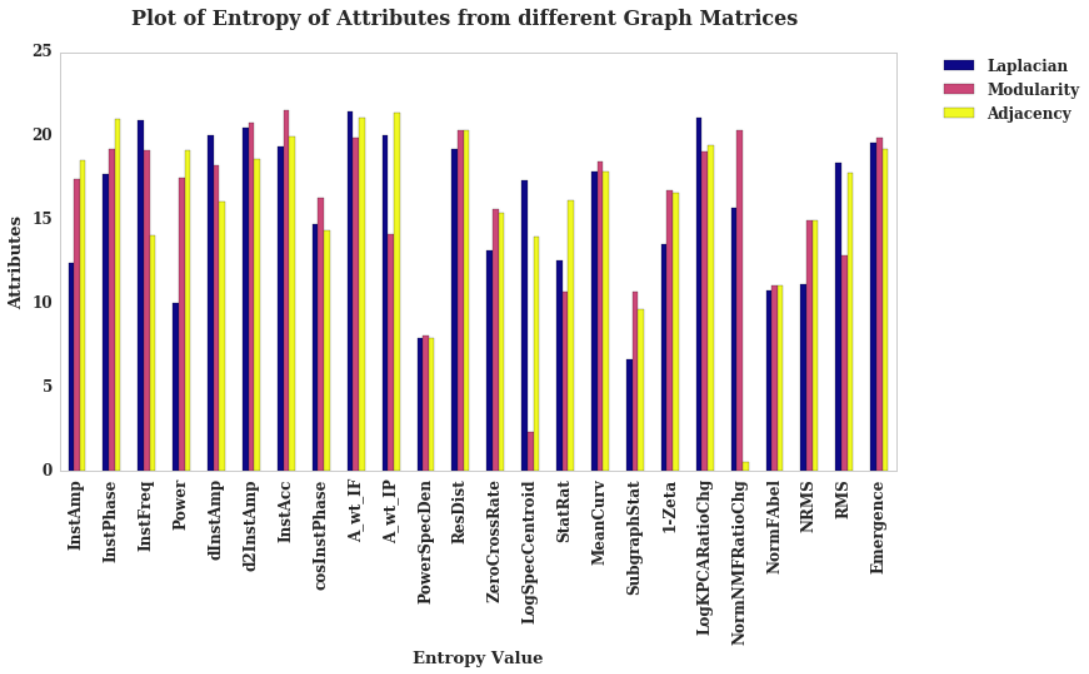
In [186]:
```
att_only.iloc[:,18:21].plot.line(fontsize=12,cmap='plasma', rot=90)
plt.xticks(np.arange(len(months)), months);
plt.legend(fontsize=12, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.
```

Out[186]:
```
<matplotlib.legend.Legend at 0x7f63f26ee080>
```

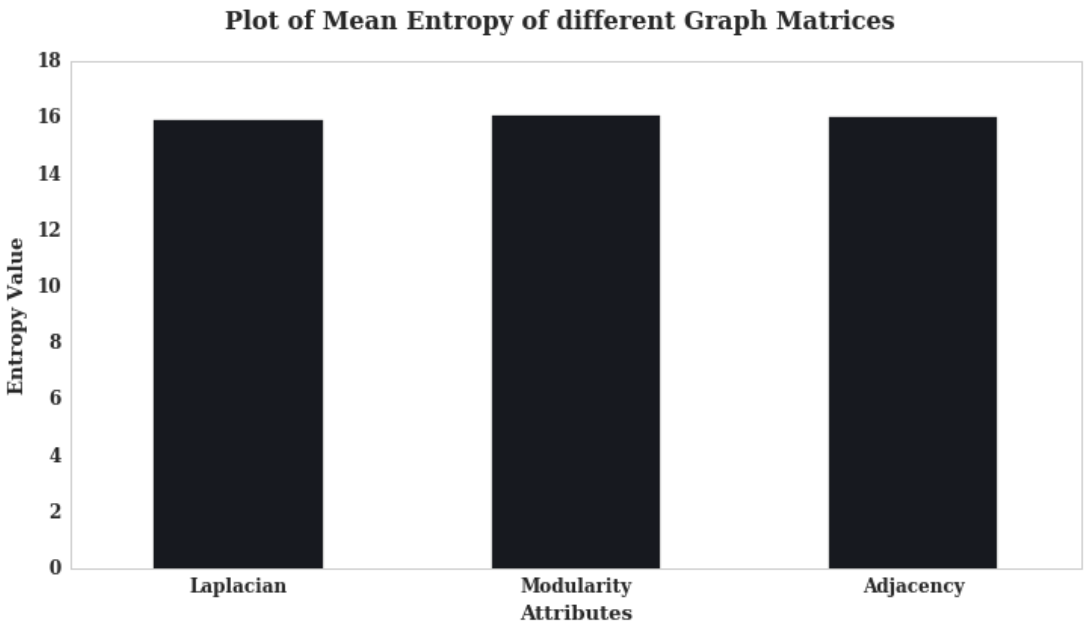


In [195]:
```
from sklearn.preprocessing import *
```

```
In [201]: px_lap = lap.apply(lambda x: minmax_scale(x, feature_range=[0,1]))
          px_mod = mod.apply(lambda x: minmax_scale(x, feature_range=[0,1]))
          px_adj = adj.apply(lambda x: minmax_scale(x, feature_range=[0,1]))

In [202]: ent_lap = -1* np.sum(px_lap*np.log2(px_lap))
          ent_mod = -1* np.sum(px_mod*np.log2(px_mod))
          ent_adj = -1* np.sum(px_adj*np.log2(px_adj))

In [203]: entropy_df = pd.DataFrame([ent_lap,ent_mod,ent_adj]).T
          entropy_df.columns = ['Laplacian','Modularity','Adjacency']

In [243]: entropy_df.plot.bar(figsize=(12,6), fontsize=12,cmap='plasma')
          plt.legend(fontsize=12, bbox_to_anchor=(1.05, 1), loc=2, borderaxespad=0.
          plt.suptitle("Plot of Entropy of Attributes from different Graph Matrices
          plt.xlabel('Entropy Value')
          plt.ylabel('Attributes')
          plt.savefig('images/entropy.png')
```




```
In [234]: entropy_df.mean().plot.bar(legend=False,fontsize=12, rot=0)
          plt.suptitle("Plot of Mean Entropy of different Graph Matrices", fontsize
          plt.ylabel('Entropy Value')
          plt.xlabel('Attributes')
          plt.savefig('images/entropy_mean.png')
```


Plot of Mean Entropy of different Graph Matrices
Entropy Value
18
16
14
12
10
8
6
4
2
0
Laplacian
Modularity
Adjacency
Attributes

In [ ]: