Complex Networks and the Generalized Singular Value Decomposition

Xiaolin Xiao

Department of Mathematics and Statistics

University of Strathclyde

Glasgow, UK

May 2011

This thesis is submitted to the University of Strathclyde for the degree of Doctor of Philosophy in the Faculty of Science.

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

Signed: Date:

To my parents and Qi, for their continued support and encouragement.

Acknowledgements

Firstly, I wish to thank my supervisor, Professor Des Higham, who has guided and assisted me enormously with my research. His energy and involvement have helped make my research experience so very rewarding.

The financial support of an EPSRC PhD studentship and the funding provided by the Department of Mathematics and Statistics in the University of Strathclyde are most gratefully acknowledged.

There are a number of people whom I would like to thank for their help, kind fellowship, and support generally during my PhD studies. In particular, I am very grateful for the assistance given by Dr. Neil Dawson from the Universities of Strathclyde and Glasgow who allowed use of their metabolic and brain data in Chapters 6 and 7, respectively. His cooperation in analyzing the corresponding data is also highly appreciated. Additionally, Professor Keith Vass and Dr. Jonathan Crofts, have been involved in many useful discussions and provided advice for my PhD studies. Professor Nataša Pržulj and Dr. Tijana Milenković from the University of California provided the protein data, which was presented in Chapter 5 and their cooperation with this and on analysis was invaluable to me. I would like to especially thank Dr. Jiazhu Pan, who assisted me with advice and suggestions in statistics, contributing significantly to validating my results. Many thanks are owed to Dr. David Gleich from Stanford University, and to Prof. Ernesto Estrada for their additional input into other aspects of my work. To my senior friend, Prof. Stephen Wilson, my gratitude for your time generously given. Data help from Dr. Darren Croft from the University of Exeter was particularly welcome, and I am thankful to all the colleagues involved in the same EPSRC funded project from the Universities of Strathclyde and Bath. In addition, I also appreciate the friendship and help provided by a dear couple, Drs. Yueyue Wang and Tom Fitzgerald.

To all of the preceding colleagues and friends be certain that, throughout my life, I shall treasure and will never forget, all your proffered help and support.

Closer to home and heart, I acknowledge the considerable love and supportive contribution of my parents–knowing how attentive you are to me I hope I will repay that by making you always proud of me wherever I go.

Finally, Qi, my love, thank you for your continued encouragement and all kinds of support. Without such love and support, this dissertation would not have been possible.

Contents

Abstract

Complex networks are everywhere. Many natural and synthesized phenomena can be modelled as complex networks, examples include: social interactions, brain functions or structures, protein-protein interactions, and even the Internet. Since complex networks have features and structures that are not present in simple networks, they receive a significant amount of attention and have become a young but active area of scientific research which brings together researches from many areas including mathematics, physics, biology, computer science and sociology. The majority of this research is focused on discovering properties and structures within a complex network.

Complex networks can be described as graphs and represented as matrices. The Generalized Singular Value Decomposition (GSVD) can factorize a pair of data matrices with the same column size simultaneously. The main aim of this thesis is to show that it is possible to evaluate differences and similarities between two complex networks using the GSVD.

In this dissertation, the GSVD was employed to compare structural differences between two complex netwoks in terms of clustering. A specific task was to develop an intuitive understanding of why the GSVD is useful for processing pairs of related data sets. Initially in this thesis, from an optimization viewpoint, algorithms have been derived in an attempt to shuffle nodes by exploiting the variational properties of the GSVD. Secondly, the standard algorithm for computing the GSVD was interpreted as an iterative method in order to justify the approach in another way. Algorithms were tested with both synthetic data and small scale real world complex networks. To verify the significance of all these findings, a cluster validation method was designed by computing the p-values. The corresponding p-values produced support that our findings are statistically significant. Subsequently, these real tests were extended to use within large scale biological complex data, such as the protein interaction networks, metabolic networks and brain networks. The corresponding results produced show our algorithms are useful for extracting some substructures which have specific biological functions. In addition, heuristic algorithms were proposed for processing a pair of nonsquare data matrices, corresponding to bipartitie graphs, from an optimization viewpoint and followed with a synthetic test. Finally, conclusions, and a number of directions for further research subjects to pursue subsequent to this work, were discussed.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 The SVD

The singular value decomposition (SVD) has been a well known entity for approximately a century as an important matrix factorization method which can be used to express a matrix $A \in \mathbb{R}^{M\times N}$ as the product $A = U\Sigma V^T$, where $U \in \mathbb{R}^{M\times M}$ and $V \in \mathbb{R}^{N\times N}$ are orthogonal matrices with columns containing the singular vectors, and $\Sigma = diag(\sigma_1, \cdots, \sigma_p) \in \mathbb{R}^{M\times N}$ with $p = \min(M, N)$ is diagonal containing the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$ [48]. A thin SVD is also defined; if $A = U\Sigma V^T \in \mathbb{R}^{M\times N}$ is the SVD of A and $M \geq N$, then $A = U_1\Sigma_1 V^T$ is called a thin SVD when $U_1 = [u_1, \cdots, u_N] \in \mathbb{R}^{M\times N}$ and $\Sigma_1 = diag(\sigma_1, \cdots, \sigma_N) \in \mathbb{R}^{N\times N}$ [48, 113]. A thin SVD is a trimmed down version of the standard SVD. In practice, the SVD can be conveniently computed by using the standard function `svd` in MATLAB[1].

The SVD is a well known spectral clustering tool since the singular vectors have been proved to be an extremely useful tool for data mining and dimension reduction [113]. Spectral clustering/ordering algorithms have been designed and

[1] `http://www.mathworks.com/`

implemented in many disciplines. Among all the related works, the study in [4] is probably the first important work describing the use of the SVD in analyzing genome-wide expression data, or more precisely, DNA microarray data. Microarray data can be regarded as an array $A \in \mathbb{R}^{M \times N}$, where a_{ij} records the activity of the ith gene in the jth sample. We will give more details of microarray data sets in the end of section 9.4 in Chapter 9. The authors in [4] demonstrate that the SVD is useful in finding an appropriate classification of the data so that the genes and samples having similar function, or similar cellular state, are put into the same group. This is completed by sorting the data according to the corresponding eigenvectors, more generally, the singular vectors, that are unique orthonormal superpositions of the genes and samples, respectively. In another way, this approach works since the SVD can reduce the data from genes × samples space to diagonalized "eigen genes" × "eigen samples" space.

Tracking the SVD in terms of some system parameters, such as clustering pattern, is a key approach to reveal emergent behaviour. Spectral clustering/ordering algorithms can be motivated from several different view points. In 2004, researchers formulated a discrete optimization problem on the symmetric weight matrix $W \in \mathbb{R}^{N \times N}$ and showed that spectral clustering may be viewed as maximum likelihood partitioning under the assumption that the data is an instance of a graph with random edge weights [62]. This viewpoint was only tested with numerical synthetic experiments. This unified viewpoint of spectral clustering was then generalized to the rectangular case $A \in \mathbb{R}^{M \times N}$ [61]. Then in [59], the authors proposed an optimization problem that can be used to give a simple derivation of a spectral clustering/ordering algorithm for symmetric data $W \in \mathbb{R}^{N \times N}$. Besides showing the numerical results on symmetric synthetic data, they also applied this approach to tumor microarray data $A \in \mathbb{R}^{M \times N}$ by forming the weighted matrix $W = A^T A \in \mathbb{R}^{N \times N}$ where the entry w_{ij} can be regarded as a measure of similarity between the corresponding two samples i

and j. All the results support their analysis that this approach can be used to find an accurate clustering of samples from the graph (microarray data), which summarizes similarity of gene activity across different tissue samples. They also generalized the above discrete optimization viewpoint of the spectral clustering method and, more generally, the SVD, to the case of unsymmetric, nonsquare data and applied it to tumor microarrays in [73]. The corresponding results show the algorithm reveals interesting gene activity across tissue samples. On the other hand, in [60] the authors showed that by interpreting the SVD as arising from both the optimization viewpoint and the power method applied to A^TA and AA^T, spectral clustering and reordering of cancer microarray data can be viewed as arising from a simple but maybe more intuitive iterative algorithm, which shuffles nodes (genes or samples) according to their correlation in the graph (network). Furthermore, the authors in [50] emphasizes the application of the SVD to the DNA expression data—Microarrays. They also state how the SVD comes into the application to these biological complex networks from the ideas given in previous related work [59, 60].

In summary, the SVD can be regarded as the basis of a spectral clustering/ordering algorithm motivated by variational properties, or interpreted as an iterative algorithm in order to shuffle nodes [59, 60]. The relevant algorithms have been applied to bioinformatics [4, 73], especially for microarray data [50, 59, 60, 61, 73, 75]. However, the SVD is mainly used on a single data matrix. Our work in this thesis is to develop an intuitive understanding of a more general tool, the GSVD, for processing pairs of related data sets, or more precisely, complex networks, simultaneously. Although spectral clustering/ordering algorithms can can be motivated from several different viewpoints, we believe that the power method viewpoint and the optimization viewpoint that used in interpreting the principles of the SVD in analyzing a single network may be more useful for our work.

1.2 The GSVD

In this thesis, we aim to develop and study a method which can explore a pair of networks simultaneously. One possible solution is to use the Generalized Singular Value Decomposition (GSVD). The GSVD was initially studied in the 1970's [129]. The GSVD expresses a pair of matrices $A \in \mathbb{R}^{M\times N}$ with $M \geq N$ and $B \in \mathbb{R}^{P\times N}$ as the products

$$A = UCX^{-1} \qquad \text{and} \qquad B = VSX^{-1}, \tag{1.1}$$

where $U \in \mathbb{R}^{M\times M}$, and $V \in \mathbb{R}^{P\times P}$ are orthogonal, $C \in \mathbb{R}^{M\times N}$ and $S \in \mathbb{R}^{P\times N}$ are diagonal with nonnegative entries such that $C = \mathrm{diag}(c_1, c_2, \ldots, c_N)$ and $S = \mathrm{diag}(s_1, s_2, \ldots, s_q)$ with $q = \min(P, N)$, and $X \in \mathbb{R}^{N\times N}$ is nonsingular [48]. By construction, the diagonal entries in C have an nondecreasing order so that $0 \leq c_1 \leq c_2 \leq \cdots \leq c_N$ and those of S have an nonincreasing order so that $s_1 \geq s_2 \geq \cdots \geq s_q \geq 0$ [48]. A trimmed down version is also available for the GSVD, which is called a thin size GSVD or alternatively economy-sized GSVD [16, 85]. In this case, the resulting factors $U \in \mathbb{R}^{M\times N}$ and $V \in \mathbb{R}^{P\times N}$, and diagonal matrices C and S are in $\mathbb{R}^{N\times N}$.

The authors of [95] construct a slightly different formulation of a GSVD of $A \in \mathbb{R}^{M\times N}$ and $B \in \mathbb{R}^{P\times N}$ with no restrictions on M, N, or P. Additionally, this idea has been widely used in standard software [16, 85] for computing the GSVD such that we have $A = UCQ^T$, $B = VSQ^T$ and $C^TC + S^TS = I$ for $A \in \mathbb{R}^{M\times N}$ and $B \in \mathbb{R}^{P\times N}$, where the orthogonal matrix U is still in $\mathbb{R}^{M\times M}$ and V is still in $\mathbb{R}^{P\times P}$. The matrix Q is in $\mathbb{R}^{N\times q}$ where $q = min(M + P, N)$ so that the common factor Q in this decomposition can be nonsquare while the common factor X expressed in (1.1) is always square. Here the condition $M \geq N$ or $P \geq N$ is not compulsory but this decomposition still requires that the matrices have the same column size. The economy-sized GSVD, or a thin GSVD, is just derived from this decomposition formulation for products U and

V which have at most N columns, and C and S have at most N rows.

In pratice, a widely used method for computing the GSVD of two matrices A and B is Paige's [95] algorithm [7, 8, 16, 48]. In this computational method, the two matrices are first factorized to the form $A = Q_1R$ and $B = Q_2R$ by using the Orthogonal-triangular (QR) factorization. The second step is to compute the Cosine-Sine (CS) decomposition $Q_1 = UCZ^T$ and $Q_2 = VSZ^T$ and order the diagonals of C and S to satisfy $0 \leq c_1 \leq c_2 \leq \cdots \leq c_N$ and $s_1 \geq s_2 \geq \cdots \geq s_N$. Then we compute orthogonal W and upper triangular T so $TW = Z^TR$, then $X = W^TT^{-1}$. In our computational examples in the following chapters, we used the `gsvd` routine built in to MATLAB for computing the GSVD. In MATLAB, `gsvd` uses the CS and the QR decompositions as described above for the widely used computation method for the GSVD [85]. In MATLAB, the QR decomposition is implemented by a separate `qr` function (M-file), whereas the CS decomposition is implemented in a subfunction (`csd`) in the `gsvd` M-file by using the built in `qr` and `svd` functions. Hence, the total computational complexity of computing the GSVD depends on the computational cost of the subfunctions (`csd`, `svd` and `qr`). As we stated above, this standard software uses the formulation $A = UCQ^T$, $B = VSQ^T$ for computing the GSVD, so the output is $Q = W^TT^T$ instead of X. The only error message produced by `gsvd` occurs when the two input matrices, A and B, do not have the same column size.

Currently, the GSVD is attracting the interest of researchers in light of its applications within life science. Perhaps the most high-profile endeavors are presented in [5]. In [5], the GSVD was used to analyze simultaneously microarray data from two different organisms; a common set of genes across two different sets of samples. This work demonstrates that the GSVD provides a comparative mathematical framework for two data sets in identifying common patterns or exclusive patterns. The two data matrices have a different number of rows, which are used to represent the genes or genelets, over the same size of columns, which are samples from two different organisms. The expression data they examined can be characterized as signals over a period/time so that the common or exclusive patterns of the features of these signals, such as shape or frequency as examples, are captured. Other biological applications of the GSVD can be seen in [14, 15, 109]. In [14, 15], the GSVD is utilized to analyze different types of genomic data by extracting similar and dissimilar patterns within both data types. In [109], the GSVD is applied to analyze two different types of transcriptional datasets produced from different lab platforms; this proves to be similar to the approach in [5] since the expression data can be characterized as signals, though it is claimed that two data sets over different sizes of arraylets, which are arranged as the matrices' columns, can be compared. In fact, the number of genes, which are arranged as rows of the matrices of the original data, are the same for two data sets. Hence the two data matrices are transposed to ensure the matrices have the same column number. Then the GSVD is applied to process the transformed data sets. However, these works merely report the application of the GSVD to biological data without justifying its use. This dissertation is aimed toward developing an intuitive understanding of why the GSVD works for processing pairs of related data sets and, furthermore, to pick out common or exclusive patterns of two data sets in terms of clustering.

1.3 Complex networks

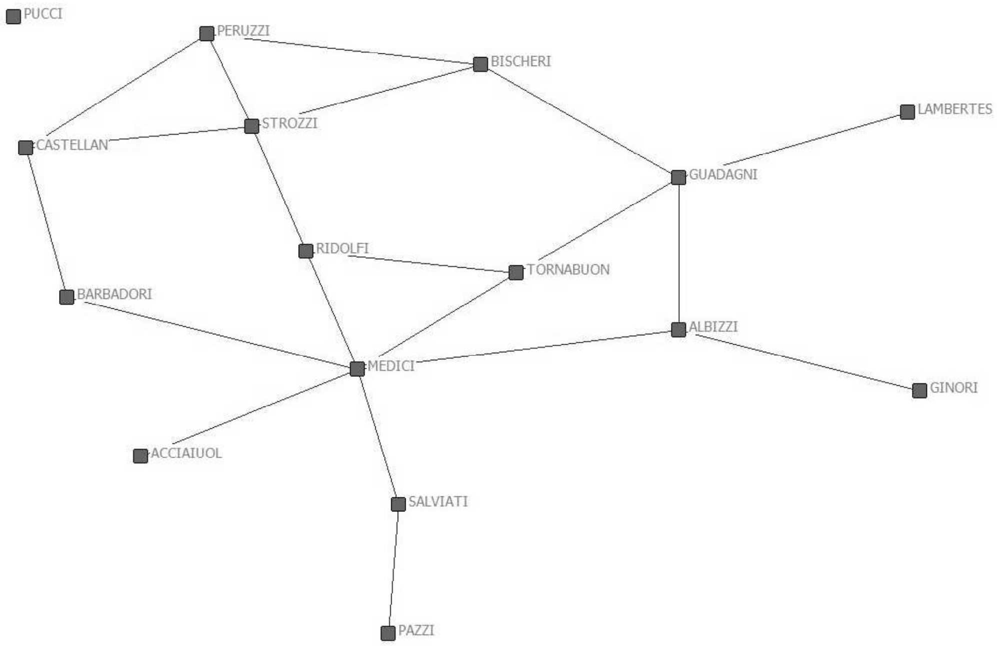
Networks are usually described as graphs in the mathematical literature. In mathematics and computer science (graph theory), a graph is denoted as G = (V, E) comprising a set V of vertices or nodes together with a set E of edges or lines [30]. The graph may be represented by a matrix of size $|V|$ (number of vertices) and the entries denote the relationship between the corresponding nodes, such as an adjacency matrix with the elements 0 or 1 to represent whether the two vertices have a connection.

There are more general types of networks, such as directed (directional) networks and undirected (bidirectional) networks. In a directed graph (network), all the edges have directions, whereas an edge in an undirected network runs in both directions. Many phenomena in nature and beyond can be modeled as a network, for example brain structures, protein-protein interaction networks, social interactions and the Internet and World Wide Web (WWW). All such systems can be represented in terms of nodes and edges indicating connections between nodes. Within the Internet, for example, nodes represent routers and the edges represent the physical connections between them. In the same way, in transport networks, the nodes represent cities and edges represent the highways that connect them. These edges can have weights, which can represent the flux of cars on a highway.

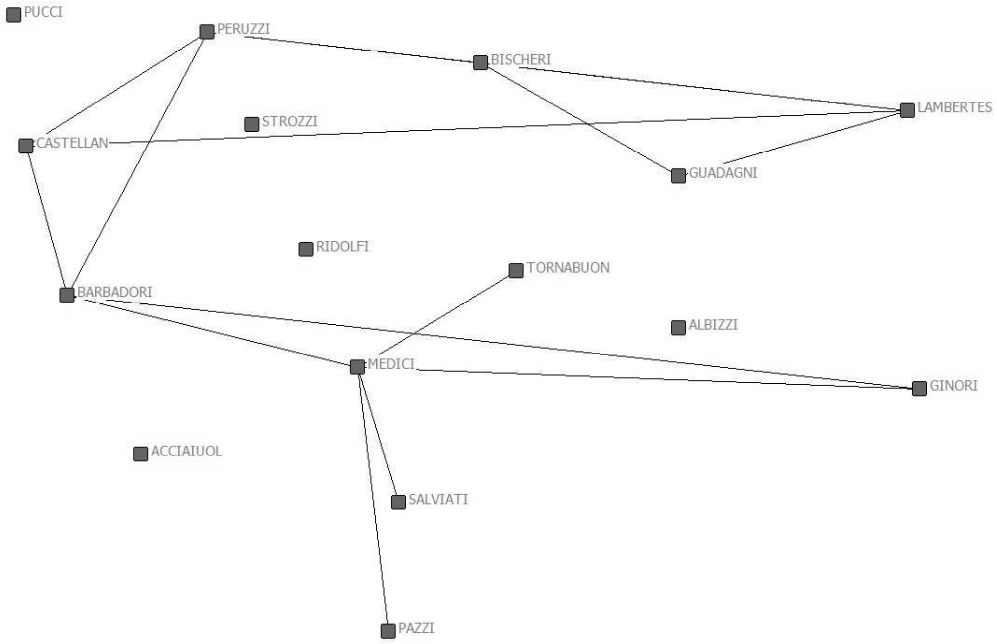
For clarity, researchers divide these real world complex networks into several main categories [91]: Social networks, information networks, technological networks and biological networks. A social network comprises sets or groups of people with some pattern of contact or interaction between them. Traditional social networks are small. One of the most well-known examples in complex information networks is scientific citation networks, which describe the networks of citation between academic papers. The third complex networks category is technological networks, which are man-made networks designed typically for distribution of some commodity or resource, such as electricity or information. A widely studied technological network is the Internet. Thus, experiments and applications encompassing some such real world complex networks are included in the following chapters of this dissertation; witness the studies of social networks, protein interaction networks, metabolic networks and brain networks in Chapters 2, 5, 6, 7 respectively.

Figure 1.1 shows a family network, which is a classic example of a social network. The upper and the lower pictures show two different types of interaction over the same 16 nodes. Each node is a 15th century FLORENTINE FAMILY, linked with the edges representing a business tie or a marriage tie for corresponding nodes [127]. Figure 1.2 is another example of social network. Social scientists use the Gahuku-Gama system of the eastern central highlands of New Guinea [103], as described by Read, to illustrate a clusterable signed graph. The signed graph [55] has been split into two patterns: friend tie, denoted as a black wide line, and enemy tie as represented by the thin line in this figure. Here, each node is a tribe.

Properties of complex networks have been studied and summarized in the literature [3, 30, 91, 118]. Perhaps the simplest useful model of a network is the classical random graph. A random graph is produced by placing edges at random between a set of n vertices. The most commonly studied random graph model is the *Erdös-Rényi* model [33, 122], denoted $\mathcal{G}(\mathsf{n}, \mathsf{p})$, in which every possible edge occurs independently with probability p. A closely related model, denoted $\mathcal{G}(\mathsf{n}, \mathsf{m})$, assigns equal probability to all graphs with exactly m edges. However, real networks are not random. Nowadays, a large number of real world complex networks have been studied and shown to exhibit the small world effect. The small world effect was first studied in an experiment carried out by Stanely Milgram in the 1960s [33, 132]. By using the average path length, the experiments indicate that most pairs of people in a social friendship network are connected by a short path. Many networks, such as social networks, interaction on the internet and complex brain networks, can be modeled by small world networks or exhibit the properties of small world networks [91]. To evaluate other properties of complex networks, a number of statistical measures have been defined [3, 30, 91]. The most frequently used statistics include degree distribution, clustering coefficient, average path length and network diameter. In



(a) Marriage tie.



(b) Business tie.

Figure 1.1: Family networks: marriage tie and business tie.

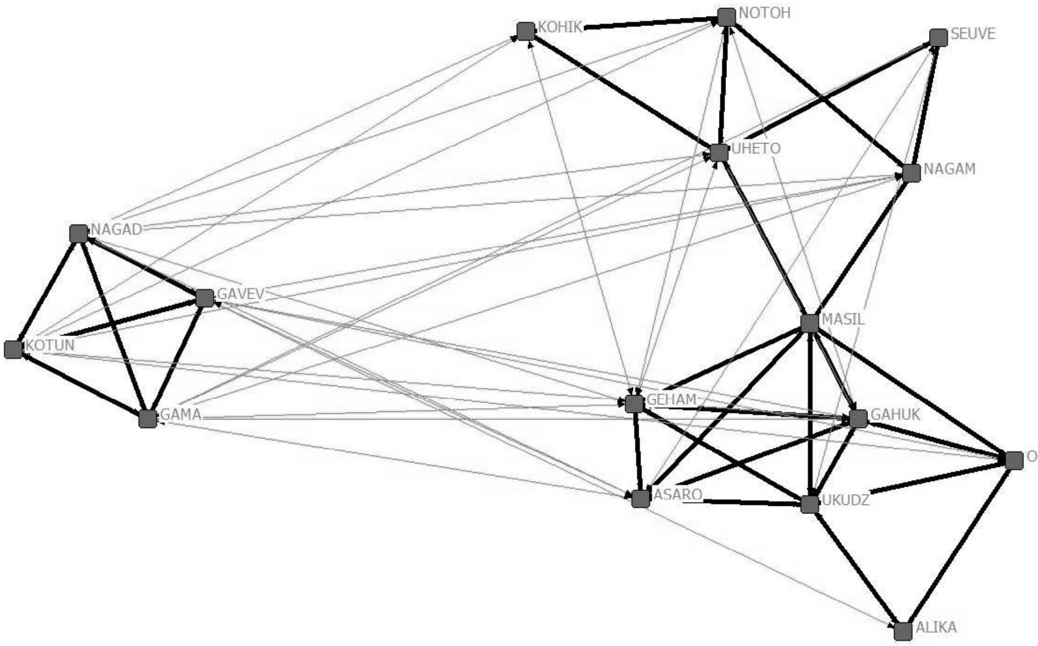


Figure 1.2: Read Highland Tribes: friend tie (wide line) and enemy tie (thin line).

graph theory, the degree of a node (vertex) is the number of edges connected to it. In a directed graph, each vertex (node) has both an in-degree and an out-degree, which are the numbers of in-coming and out-going edges, respectively. The degree distribution captures the local neighbourhood diversity in the network. The degree distribution $n(k)$ is the number of nodes of degree k for all $k \geq 0$. The clustering coefficient is a measure of the average local neighbourhoods in a graph/network. It defined by the probability that two neighbour nodes i and j of a third node l are themselves connected. In graph theory, the distance d_{ij}, which is also known as the geodesic distance or geodesic path, between two nodes i and j in a graph is defined through the minimum number of edges in a path connecting them. If the graph has n nodes, then the average path-length $< d >$ of the graph is defined as $< d >= \frac{2}{n(n-1)} \sum_{i=1,j=1,i\neq j}^{n} d_{ij}$. The diameter of a network is the maximum distance (or the length of the longest geodesic path between any two nodes) in the network.

The study of complex networks is a young and active area of scientific research which brings together researchers from many areas including mathematics, physics, biology, computer science, sociology and epidemiology. As we stated above, an important characteristic of these networks is that they are not random, but have a more structured architecture. This motivates us to study the problem of exploring pairs of complex networks with some mathematical method in terms of clustering. Recalling section 1.2, we can decompose two networks simultaneously by using the GSVD. Hence, in this dissertation, we will work under the general title of "Complex Networks and the Generalized Singular Value Decomposition". The key aim is to develop an intuitive understanding of why the GSVD is useful for processing pairs of related complex interaction networks. In addition, an extended outline of my thesis is given in section 1.4.

1.4 Outline of Thesis

In Chapter 2, two computational algorithms are derived based on an optimization problem combined with the variational properties of the GSVD. To show their effectiveness, the algorithms are tested on a synthetic data set and some real data sets from sociology and neuroscience.

To justify the significance of our findings in the tests in Chapter 2, a cluster validation process was proposed by checking the corresponding statistical significance in Chapter 3. The corresponding results support our findings in Chapter 2. Here, the cluster validation process works on binary adjacency matrices, though we will later generalize the corresponding process to the real-valued weighted case.

In Chapter 4, an algorithm for computing the GSVD is interpreted as an iterative method for attempting to shuffle the nodes. Interpreting the GSVD this way enables processing of pairs of symmetric, real valued weighted graphs.

In Chapter 5, our algorithms are used to compare large scale protein interaction networks: Protein-Protein Interaction (PPI) networks and Genetic Interaction (GI) networks. First, these two kinds of interaction networks are introduced using related research which also outlines especially our motivation towards studying their structural difference by indicating the biological process or behavior exclusive to one graph then the other. Since these networks are huge (over thousands of nodes), to avoid expensive computational cost, we use some techniques to trim the data sets before applying computational algorithms to them. All the details of these pre-processing steps are discussed in section 5.2 of Chapter 5.

Chapter 6 illustrates an example of the GSVD based algorithms being applied, this time to analyze metabolic data sets. First, the background in neuroscience and metabolomics of this study is explained, then the algorithms derived from Chapter 2 based on the optimization view are generalized from the binary case to the symmetric, real-valued weighted case. Third, the materials and methods used to prepare and analyze the metabolic data are described. Since the metabolic data matrices are weighted, some steps in the previous cluster validation method designed in Chapter 3 were no longer applicable. Accordingly, in this chapter, a different cluster validation process is proposed to validate the significance of the corresponding findings produced from the real-valued weighted matrices. Then the results of the metabolic pathway disruption in the subchronic phencyclidine model of Schizophrenia with our GSVD based algorithm derived from the optimization view are given, followed by a further discussion on metabolomics.

Similarly, work attempting to extend applications to brain networks is described in Chapter 7. As an introduction, general brain networks research is summarized. Then the materials and methods used to generate the brain data matrices are described. Our algorithms were applied to pick out potentially exclusive good clusters which were present in one graph but not in the other. In this chapter, we also validate our findings by applying the cluster validation method designed for the weighted graphs in Chapter 6, and then discuss the biological meaning of the results revealed in correspondence to neuroscience.

In Chapter 8, optimization problems for developing a theory for why the GSVD works on pairs of nonsquare data matrices are set up, followed by a numerical experiment. Testing the algorithm on synthetic data, it is shown that the GSVD works well on a pair of weighted, nonsquare data matrices in picking out good clusters which were present in one graph but not in the other. It is also indicated that the algorithms developed from these optimization problems can be used to process related nonsymmetric square data matrices too.

Finally, the entirety of this thesis is summarized in Chapter 9, with suggestions and proposals towards study of remaining outstanding questions discussed, and thereby indicating the likely directions of future work towards progress in this area.

1.5 Publications and Presentations

The material presented in Chapter 6 and much of the material presented in Chapters 2 and 3 has been written as an article

- *Exploring Metabolic Pathway Disruption in the Subchronic Phencyclidine Model of Schizophrenia with the Generalized Singular Value Decomposition*, X. Xiao, N. Dawson, L. MacIntyre, B. J. Morris, J. A. Pratt, D. G. Watson and D. J. Higham, accepted for BMC Systems Biology (2011).

The material presented in Chapter 7 and part of the material presented in Chapters 3 and 4 has also been written into the article

- *Sustained NMDA receptor hypofunction induces compromised neural systems integration and schizophrenia-like alterations in functional brain networks*, N. Dawson, X. Xiao, D. J. Higham, B. J. Morris and J. A. Pratt, submitted (2011).

The material presented in Chapter 7 and much of the material presented in Chapters 2 and 3 was presented as a poster entitled “Exploring Pairs of Brain Networks with the Generalized Singular Value Decomposition” at SNG2010 (Scottish Neuroscience Group Meeting 2010), University of Strathclyde, August 2010.

The material presented in Chapter 2 was first given as a presentation entitled “Exploring Pairs of Complex Networks with the Generalized Singular Value Decomposition” at the Interdisciplinary International Workshop on Complex Networks in Natural and Technological Science, University of Strathclyde, January 2009, and then was given as a presentation again together with material presented in Chapter 5 at the 23rd Biennial Conference on Numerical Analysis, University of Strathclyde, June 2009.

Chapter 2

Optimization Viewpoint

2.1 Background

Large, complex interaction networks arise across many applications in science and technology [3, 91, 118]. Spectral methods, based on information computed from eigenvectors or singular vectors, have been used successfully to reveal fundamental network properties. For example, we may wish to cluster objects into groups [112], put objects into order [58] or discover specific patterns of connectivity within subgroups [38, 40, 65, 89, 124]. In this chapter, we look at the case where two interaction data sets are available and the aim is to discover differences between the two sets in terms of clustering. Section 2.2 sets up the problem and shows how a spectral algorithm can be derived. An alternative viewpoint is given in section 2.3, and this leads to a variant of the algorithm. Section 2.4 tests the two approaches on a synthetic data set, where results can be judged accurately. In section 2.5 we then apply the most promising algorithm to real data sets arising in sociology and neuroscience, showing that informative patterns can be found.

2.2 Algorithm Derivation

Suppose that the square matrices A and B in $\mathbb{R}^{N\times N}$ represent two different types of interaction between a set of N nodes. We use the convention that a large weight a_{ij} or b_{ij} indicates that nodes i and j are strongly connected with respect to interactions of type A or B, respectively. For example, in section 2.5 we consider the example from Figure 1.1 for a set of families where A records inter-family marriage ties and B records inter-family business ties. In order to reveal interesting differences between the two types of connectivity data, we may then look for a set of nodes that form a good cluster with respect to A and a poor cluster with respect to B, or vice versa.

As a starting point for a computational algorithm, we consider the identity

$$\|Ax\|_2^2 = \sum_{k=1}^{N} x_k^2 \deg_k^A + \sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{l=1,l\neq k}^{N} a_{ik}a_{il}x_kx_l, \tag{2.1}$$

for $x \in \mathbb{R}^N$. Here $\|\cdot\|_2$ denotes the Euclidean norm and $\deg_k^A := \sum_{j=1}^{N} a_{kj}^2$ is one way to generalize the concept of degree to the case of a weighted network. Suppose we wish to split the nodes into two groups such that nodes within each group are well-connected but nodes across different groups are poorly connected. We could use an indicator vector $x \in \mathbb{R}^N$ to denote such a partition, with $x_s = 1$ if node s is placed in group 1 and $x_s = -1$ if node s is placed in group 2.

Fixing on two nodes, k and l, we could argue that the existence of a third node, i, such that a_{ik} and a_{il} are both large is evidence in favor of placing k and l in the same group (since they share a strong connection with node i). On the other hand, a small value for either of both of a_{ik} and a_{il} is evidence in favor of placing k and l in different groups. In terms of the indicator vector, this translates to

1. $a_{ik}a_{il}$ large ⇒ try to choose $x_kx_l = +1$,

2. $a_{ik}a_{il}$ small ⇒ try to choose $x_kx_l = -1$.

Returning to the right-hand side of (2.1), we see that $\sum_{k=1}^{N} x_k^2 \deg_k^A$ is independent of the choice of indicator vector, and $\sum_{i=1}^{N} \sum_{k=1}^{N} \sum_{l=1, l\neq k}^{N} a_{ik} a_{il} x_k x_l$ gives a measure of how successfully we have incorporated the (possibly conflicting) desiderata in points 1 and 2 over all pairs k, l and third parties i. So we could judge the quality of an indicator vector by its ability to produce a large value of $\|Ax\|_2^2$, provided other constraints, such as balanced group sizes, were satisfied.

Analogously, we can argue that making $\sum_{i=1}^{N} \sum_{k=1}^{N} \sum_{l=1, l\neq k}^{N} a_{ik} a_{il} x_k x_l$ as negative as possible is a good way to avoid forming well-connected subgroups, and so the problem

$$\max_{x_s \in \pm 1,\, 1 \leq s \leq N} \frac{\|Ax\|_2^2}{\|Bx\|_2^2} \tag{2.2}$$

is a good basis for picking out strong clusters in A that are not present in B.

In general, optimizing over a large, discrete set of possibilities is computationally intractable, and hence we will follow the widely used practice of relaxing to an optimization over $\mathbb{R}^N$ [59, 112]. So, instead of (2.2) we have

$$\max_{x \in \mathbb{R}^N,\, x \neq 0} \frac{\|Ax\|_2^2}{\|Bx\|_2^2}. \tag{2.3}$$

At this stage we recall from Chapter 1 that a general pair of matrices $A \in \mathbb{R}^{M \times N}$ with $M \geq N$ and $B \in \mathbb{R}^{P \times N}$ can be simultaneously factorized using the Generalized Singular Value Decomposition (GSVD) into

$$A = UCX^{-1} \qquad \text{and} \qquad B = VSX^{-1}, \tag{2.4}$$

where $U \in \mathbb{R}^{M \times M}$ and $V \in \mathbb{R}^{P \times P}$ are both orthogonal, $X \in \mathbb{R}^{N \times N}$ is invertible, $C = \operatorname{diag}(c_1, c_2, \ldots, c_N)$ and $S = \operatorname{diag}(s_1, s_2, \ldots, s_q)$ with $q = \min(P, N)$ are diagonal, and $0 \leq c_1 \leq c_2 \leq \cdots \leq c_N$ and $s_1 \geq s_2 \geq \cdots \geq s_q \geq 0$ [48]. The ratios $\lambda_i = c_i / s_i$ are the *generalized singular values* of A and B.

A key property of the GSVD is that the columns of X are stationary points of the function $f : \mathbb{R}^N \mapsto \mathbb{R}$ given by $f(x) = \| Ax \|_2 / \| Bx \|_2$, with the generalized singular values λ_i giving the corresponding stationary values [21]. Hence, we may tackle the problem (2.3) through the GSVD. Columns $1, 2, 3, \ldots$ of X are candidates for finding good clusters in B that are poor clusters in A and, analogously, columns $N, N-1, N-2, \ldots$ of X are candidates for finding good clusters in A that are poor clusters in B.

In this work, we take a visualization approach. We will display the interaction matrix that arises when x is used to reorder the nodes. More precisely, we relabel row and column i of A and B as row and column p_i, where

$$p_i \leq p_j \Longleftrightarrow x_i \leq x_j.$$

In this way, the existence or lack of clusters in the matrix becomes apparent from inspection of the heat map of the matrix. A heat map is a graphical representation of data where the values are represented as colors. It is widely used in displaying the results of a cluster analysis by permuting the rows and the columns of a matrix to place similar values near each other according to the clustering [37, 134]. There are several different kinds of heat map [134]. In a heatmap, larger values are usually represented by darker or warmer colors and smaller values by lighter or colder colors. It is, of course, possible to perform further computations in order to automate the process of finding and quantifying clusters; see [46, 112] for examples that apply to a single network. However, on the data sets used in this study, we found that visualization was intuitively revealing, especially for the case of binary (unweighted) graphs, where a straightforward nonzero pattern can be displayed. Furthermore, we will also quantify the significance of these visual findings by computing p-values [42, 86]. All the corresponding details will be given in Chapter 3.

2.3 A Variant of the Algorithm

In the context of this chapter, the matrices A and B are square, with $M = N = P$. In this case, when A and B are invertible it is known that the GSVD is closely related to the standard Singular Value Decompositions (SVD) of AB^{-1} and BA^{-1}. To see this, we could rearrange (2.4) into

$$AB^{-1} = UCS^{-1}V^T \qquad \text{and} \qquad BA^{-1} = VSC^{-1}U^T. \tag{2.5}$$

Alternatively, we may let $z = Ax$ or $y = Bx$ in (2.3), to obtain the quadratic problems

$$\max_{z \in \mathbb{R}^N,\, z \neq 0} \frac{\|z\|_2^2}{\|BA^{-1}z\|_2^2} \qquad \text{or} \qquad \max_{y \in \mathbb{R}^N,\, y \neq 0} \frac{\|AB^{-1}y\|_2^2}{\|y\|_2^2},$$

which can be solved through the standard SVD.

It is known from spectral graph theory that the dominant singular vectors give good directions in which to look for clusters [112, 117]. Inverting the weight matrix reverses their importance (the singular value σ becomes σ^{-1}) and hence a spectral clustering approach applied to A^{-1} will typically find the opposite of good clusters—poorly connected nodes will be grouped together [39]. So, intuitively, forming AB^{-1} in (2.5) should produce a data matrix for which the SVD approach finds good clusters for A and poor clusters for B. Analogously, the opposite holds for BA^{-1}.

Having interpreted the algorithm this way, it is then natural to consider the reverse products, $A^{-1}B$ and $B^{-1}A$, or, equivalently, to form the optimization problem

$$\max_{x \in \mathbb{R}^N,\, x \neq 0} \frac{\|B^{-1}x\|_2^2}{\|A^{-1}x\|_2^2}. \tag{2.6}$$

We may interpret (2.6) from the point of view that making $B^{-1}x$ large encourages poor clusters for B, while making $A^{-1}x$ small encourages good clusters for A. In this case, we would base our algorithm on the GSVD of A^{-1} and B^{-1}.

In the situation where A and B are both symmetric, corresponding to undirected networks, we have, from (2.4),

$$A^{-1} = (A^T)^{-1} = (X^{-T}CU^T)^{-1} = UC^{-1}X^T$$

and

$$B^{-1} = (B^T)^{-1} = (X^{-T}SV^T)^{-1} = VS^{-1}X^T.$$

Then we may appeal to the arguments in section 2.2 and use columns from the inverse of the third factor in the GSVD as the basis for reordering. With this approach we use columns of X^{-T} rather than columns of X. We emphasize that although this heuristic derivation used an assumption that A and B are invertible, the GSVD, and hence the final algorithm, applies in the non-invertible case. If A and B are singular (not invertible), we still can write $AB^{-1} = UCS^{-1}V^T$ and $BA^{-1} = VSC^{-1}U^T$ by using the pseudo-inverse of A and B. Another aspect we want to emphasize is that although this variant of the algorithm is based on an assumption that A and B are both symmetric, we could justify using columns from X^{-T} in this heuristic algorithm without the corresponding restrictions on A and B by setting up an alternative optimization problem. This alternative solution will be introduced in section 8.3 of Chapter 8.

To summarize, in terms of the GSVD (2.4), we will refer to the two reordering approaches as

Algorithm 1: reorder the network via a column of X and

Algorithm 2: reorder the network via a column of X^{-T}.

In both cases the first few columns should give orderings that favor clusters in B rather than A and vice versa for the final few columns.

2.4 Synthetic Test

In this section we test the algorithms in a simple, controlled case where we know the "correct" answer. We generated adjacency matrices A and B as shown in Figure 2.1. Here we have 20 nodes. In both networks, nodes 1–5 are well connected. In A there is a well connected cluster consisting of nodes 6–15, whereas in B there is a well connected cluster consisting of nodes 15–20. To make the test more realistic, the clusters are not perfect; there are both missing edges within the clusters and spurious edges outside the clusters. Our aim is to test whether the algorithms can identify the clusters that are particular to each data set.

We emphasize that the node labelling in Figure 2.1 was chosen purely to make the inherent structure visually apparent. Any spectral reordering algorithm should be invariant to a relabelling of the input data. In our context, this follows from the fact that for any permutation matrix P, the factorizations $A = UCX^{-1}$ and $B = VSX^{-1}$ are equivalent to $PAP^T = (PU)C(PX)^{-1}$ and $PBP^T = (PV)S(PX)^{-1}$. So, on the relabelled data matrices, (PX) plays the role that was played by X, and Algorithms 1 and 2 reorder based on the appropriately permuted columns of X and X^{-T}, respectively, as required. In Figure 2.2 we show the same two data sets with an arbitrary relabelling in order to illustrate that the inherent structure is no longer apparent. In essence, we are hoping that the algorithms will reveal the structure that is buried in Figure 2.2.

In Figure 2.3 we display the two adjacency matrices reordered with Algorithm 1; we show reorderings with eight different columns of X, four from each end of the spectrum. We see that none of these reorderings reveals the mutually distinct clusters.

In contrast, Figure 2.4 shows results for Algorithm 2; the two matrices are reordered with the first and last four columns of X^{-T}. In this case we see that

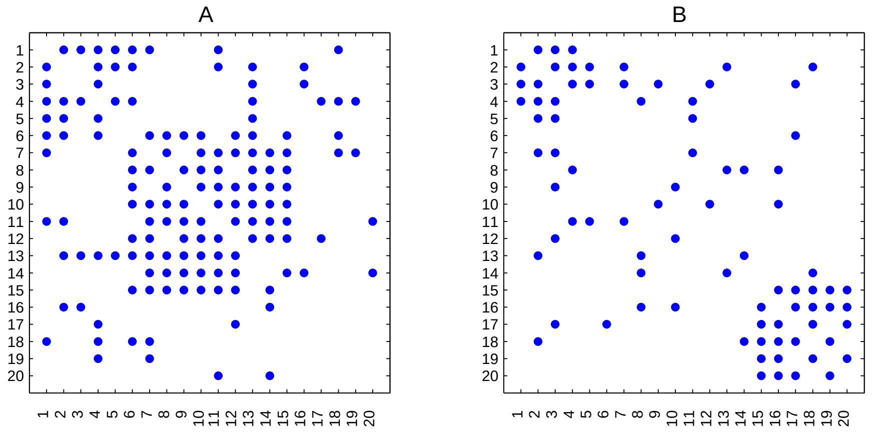


Figure 2.1: Adjacency matrices for the two synthetic graphs.

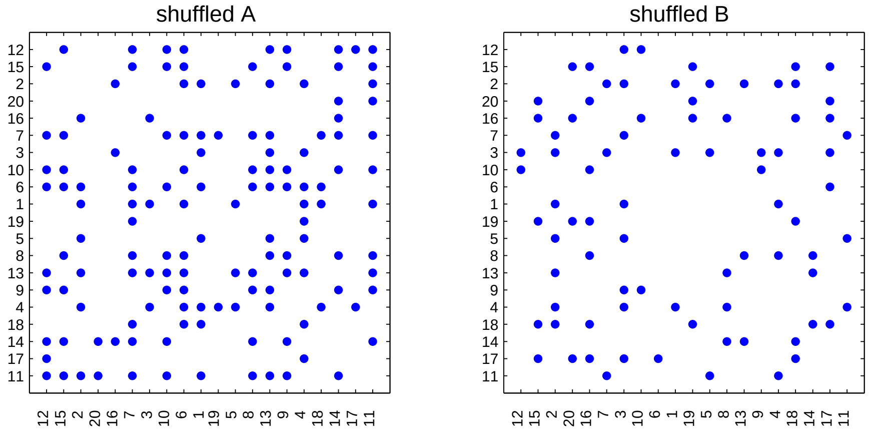


Figure 2.2: Shuffled versions of the synthetic graphs in Figure 2.1.

mutually exclusive structures have been uncovered. The reordering from the first column begins with nodes $18, 20, 16, 15, 19, 17$, which form a cluster in B, but not A. The final column begins by picking out nodes $7, 9, 10, 15, 14, 11, 6, 13$, which form the bulk of the 6–15 cluster in A. Nodes 8 and 12, which are missing from this sequential ordering, are placed at the head of the ordering in the penultimate column, which begins $12, 8, 7, 10, 15, 14, 9, 11$. So in summary, the 19th and 20th columns of X^{-T} each reveal almost complete information about the exclusive cluster in A, and between them they capture the full cluster.

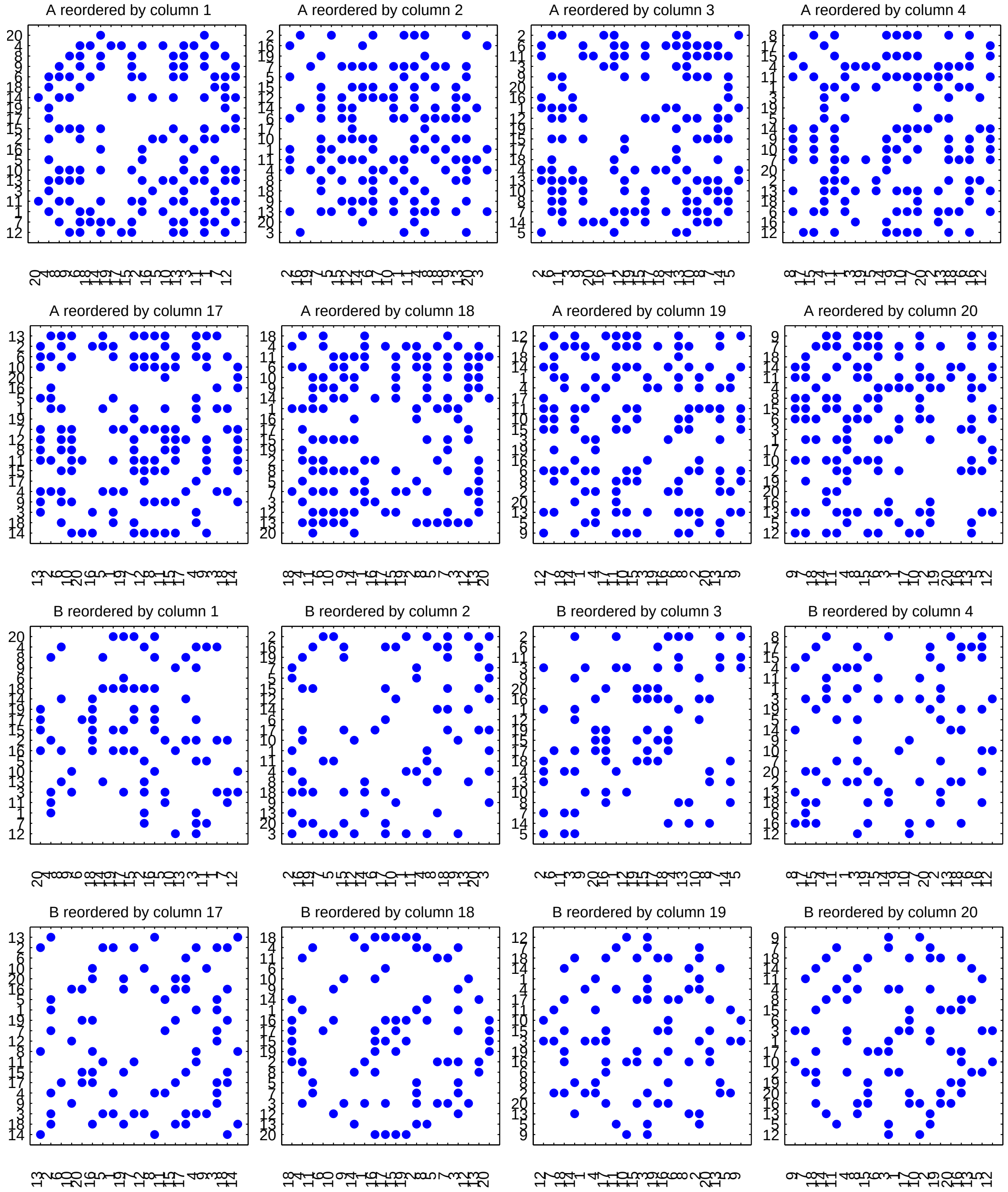


Figure 2.3: Graphs reordered by the columns from X.

We also applied the test to prefect data (the clusters are prefect without missing edges within the clusters and no spurious edges outside the clusters).

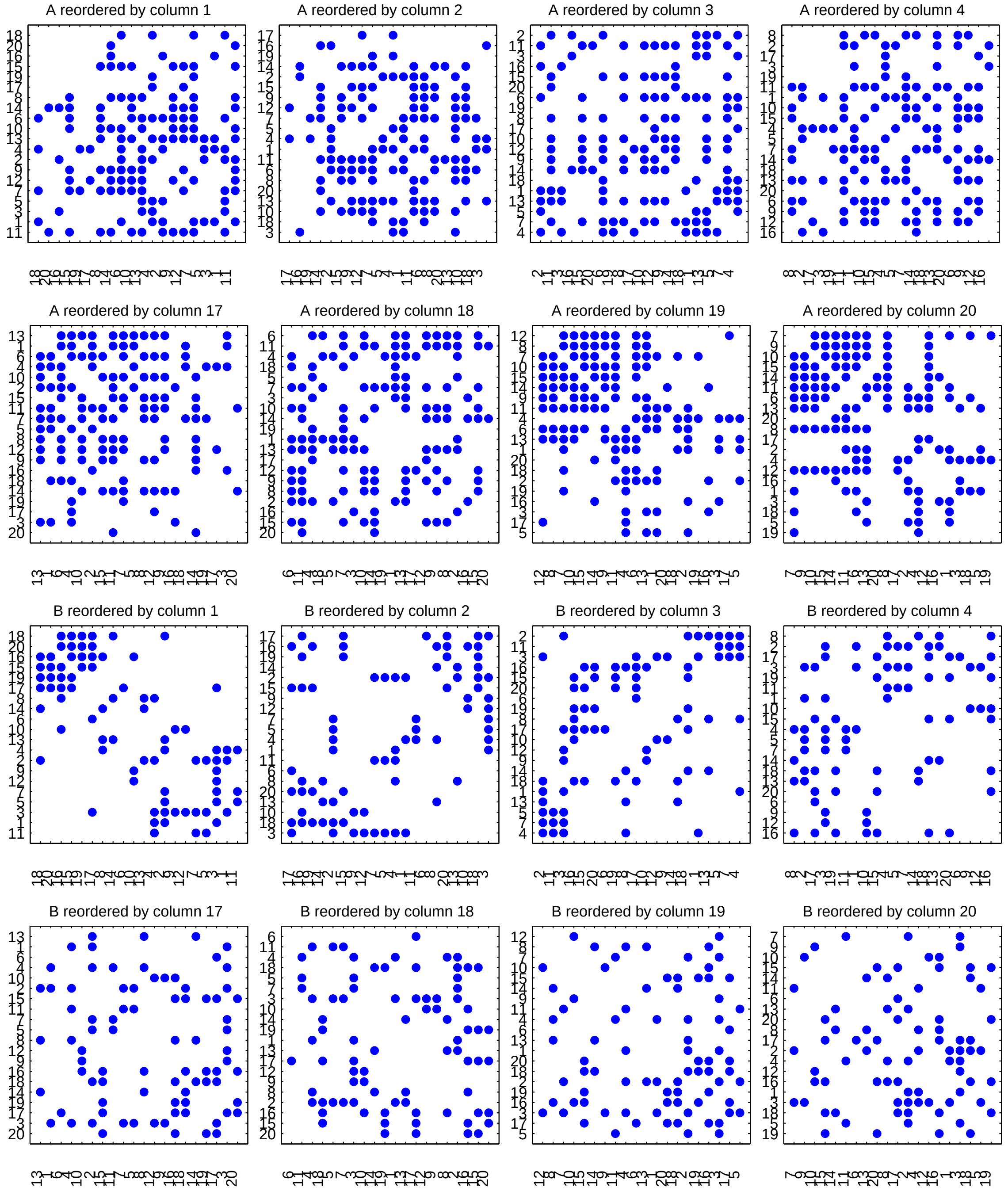


Figure 2.4: Graphs reordered by the columns from X^{-T}.

The clusters present in one synthetic data set but not in the other were all revealed by reordering the data with the columns of X^{-T}. Further experience on other synthetic data sets, and also on real networks, suggests that Algorithm 2 is more effective than Algorithm 1. Hence in the remainder of this work, we will focus on results obtained from the X^{-T} reordering.

2.5 Tests on Real Networks

The previous section indicated that the new algorithm can reveal the required structure when it is known to be present in the data. We now use the reordering approach to search for substructures in real networks. We begin with the adjacency matrices illustrated in Figure 2.5, which give the same information as Figure 1.1. Here, the nodes represent sixteen 15th century Florentine families. In matrix A, edges denote marriage ties and in matrix B they denote business ties. This data, which was taken from UCINET IV at

`http://vlado.fmf.uni-lj.si/pub/networks/data/Ucinet/UciData.htm`

has been well studied by social scientists [17] and by researchers in network science [41, 92], but we are not aware of any attempts to deal with the two networks simultaneously. Figure 2.6 shows the data reordered according to the first column of X^{-T}. We see that a set of six families, Lambertes, Peruzzi, Bischeri, Barbadori, Guadgani and Castellan, has been placed at the head of the list, and these families form a well-connected group for B (nine out of the possible fifteen edges present), but not for A (five out of the possible fifteen edges present). This pattern emerges despite the fact that B is a sparser network than A (fifteen compared with twenty edges). Reordering with the final columns of X^{-T} did not produce any reciprocal structure. Overall, the algorithm has uncovered a set of six families having strong business ties but weaker marriage ties, and there is no evidence of family groups having strong marriage ties but weaker business ties.

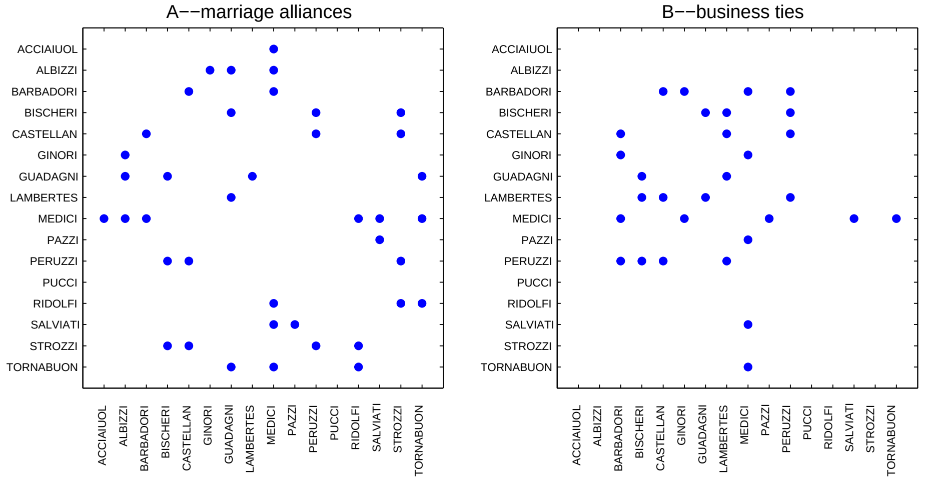


Figure 2.5: Adjacency matrices showing Florentine family connections.

Social network data from [51], also available at UCINET IV, appears in Figure 2.7. Here there are sixteen tribes from the central highland of New Guinea. In matrix A, two tribes are linked if they have a friendship relationship and in B they are linked if they have an enemy relationship. We would intuitively expect to find a group of tribes that shares strong friendship ties and weak enemy ties (a friend of my friend tends to be my friend rather than my enemy). On the other hand, there is no clear argument for finding groups of more than two tribes that have (a) strong mutual enemy ties and (b) weak mutual friendship ties (an enemy of my enemy could be my friend or my enemy).

The reordering results support this intuition. Figure 2.8 shows the reordering from the final column of X^{-T}, where six tribes, Masil, Gahuk, Ukudz, Asaro, Geham and Ove, have thirteen out of a possible fifteen friendship ties and no enemy ties. Also, four other tribes, Kotun, Nagad, Gama and Gavev are seen to share all six possible friendship links and no enemy links. Reciprocal patterns did not emerge from the initial columns of X^{-T}.

Figure 2.9 shows networks arising in neuroscience. The two adjacency matrices describe different kinds of interrelation within one hemisphere of the macaque

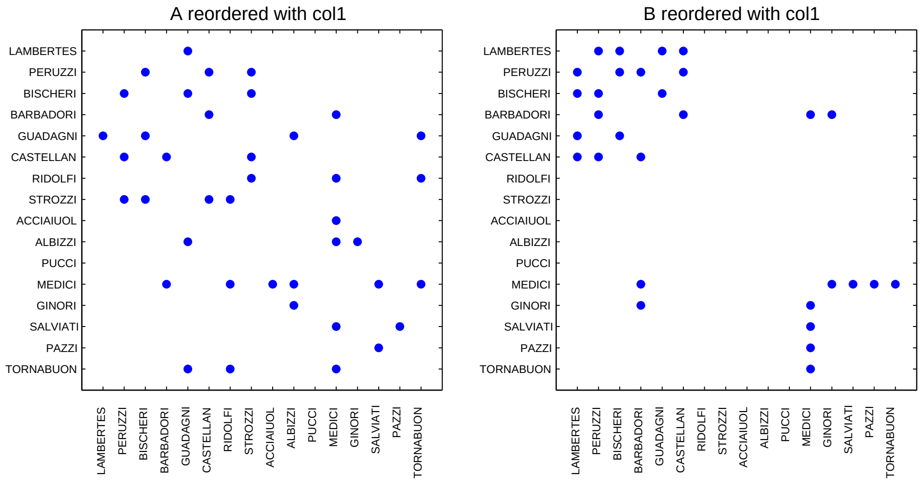


Figure 2.6: Florentine family ties reordered by the first column of X^{-T}.

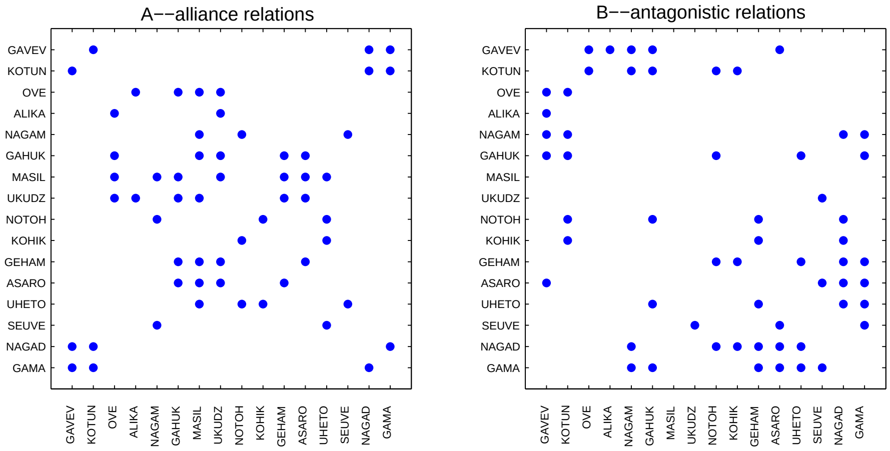


Figure 2.7: Adjacency matrices for highland tribe social connections.

brain, using data from [72] available at

`http://www.biological-networks.org/?page_id=25`

Here, each of the 94 nodes represents a region in the monkey brain. In the first matrix, $a_{ij} = 1$ if region i has an anatomical connection to region j and $a_{ij} = 0$ otherwise. In the second matrix, $b_{ij} = 1$ if regions i and j are physically close and $b_{ij} = 0$ otherwise. (More precisely, we computed the reciprocal of the

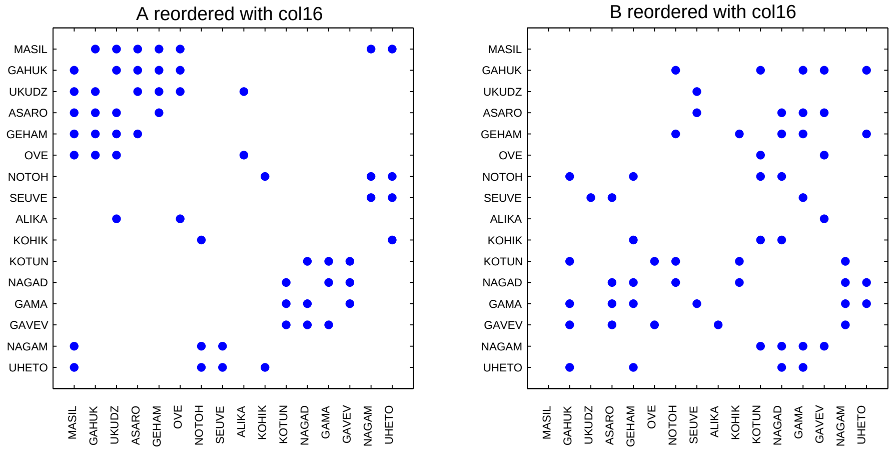


Figure 2.8: Highland tribe social networks reordered by the final column of X^{-T}.

Euclidean distance between pairs of nodes, and binarized so that B has the same number of edges as A.) We note that connections in A are directional, and A is unsymmetric; however the asymmetry is very mild, with $\| A - A^T \|_2 / \| A \|_2 = 0.52$. Furthermore, we stated in section 2.3 that we have another way to justify the use of the columns from X^{-T} even when A and/or B are asymmetric by optimization in Chapter 8. So it does not matter whether symmetric matrices are supplied or not.

This data set is larger than those presented earlier, and we found it more useful visually to display the difference, $A - B$, under the reorderings from the algorithm. In Figure 2.10 we show the result when the final column of X^{-T} is used for the reordering, with a gray scale. Because A and B are binary, the difference $A - B$ may only take values -1 (dark gray), 0 (light gray) or $+1$ (white). We see from the figure that there is a large group of nodes producing a cluster of predominantly $+1$ values in the upper left-hand corner. More precisely, with this reordering the leading 38×38 submatrix of A has 1054 nonzeros, whereas in B there are only 590. This corresponds to a set of nodes that have strong, direct, anatomical connections but are typically not geographically close. Kaiser

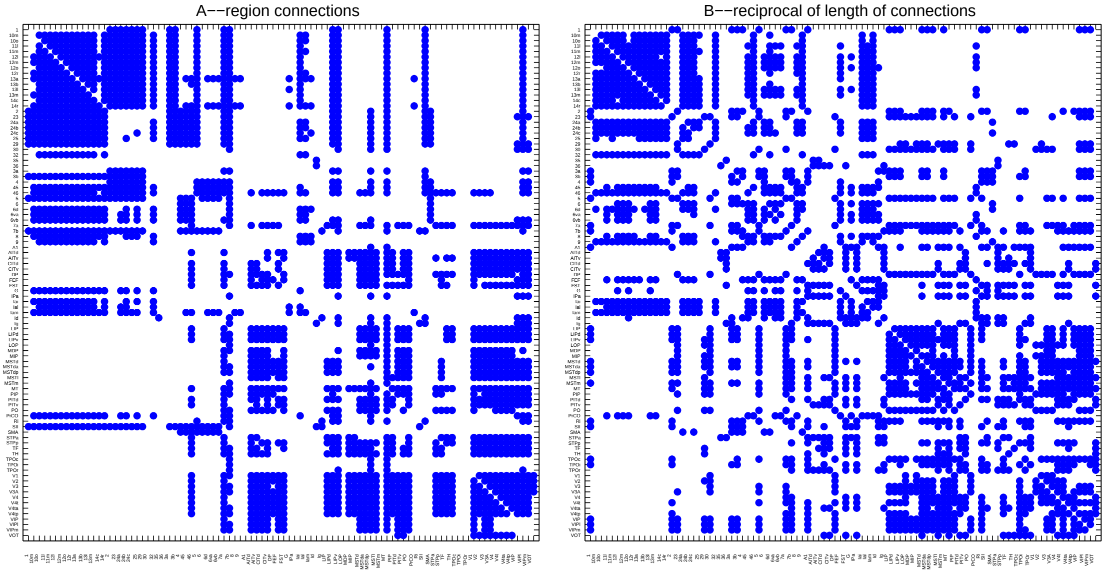


Figure 2.9: Adjacency matrices for macaque cortical connectivity networks.

and Hilgetag [72] argue that neural systems have a bias towards minimizing the "number of processing steps" between nodes (i.e. the pathlength from traditional graph theory), rather than the physical length of the connections. Figure 2.10 is consistent with this hypothesis, as it shows that the network harbors a well-coordinated long-distance subnetwork. On the other hand, the initial columns of X^{-T} did not reveal any large clusters for B that were weak clusters of A. This is intuitively reasonable—we would not expect to find a large group of brain regions that where physically close but poorly interconnected.

2.6 Conclusions

This work addresses the situation where a pair of networks describes two different types of connection between a common set of nodes. We argued from first principles that the generalized singular value decomposition (2.4) provides a very useful computational tool, and used data from social and neurological sciences to confirm that

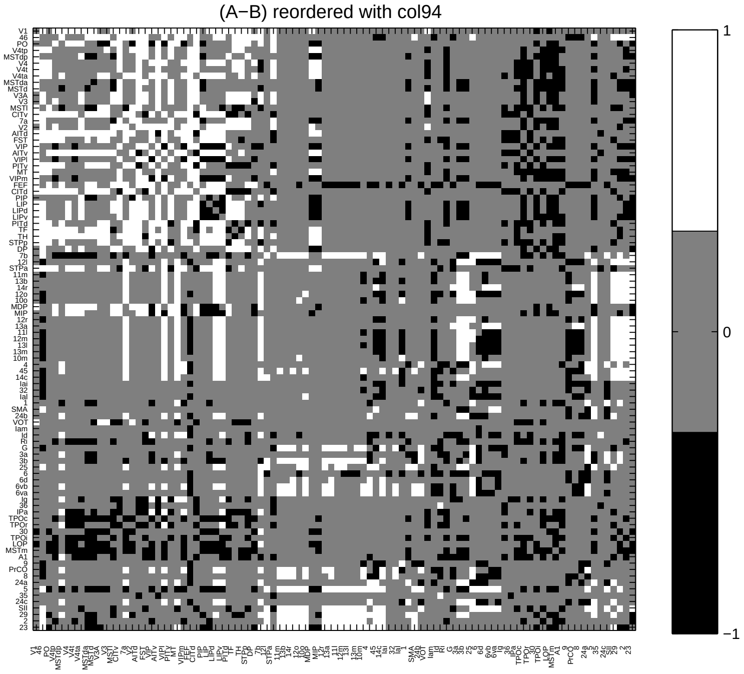


Figure 2.10: Macaque cortical connectivity networks reordered by the final column of X^{-T}.

- the third factor of the decomposition can reveal communities that are well connected in one network and poorly connected in the other, and
- these patterns exist in physical networks and have meaningful interpretations.

In this chapter, the algorithms were developed for binary adjacency matrices. These algorithms will be generalized to the real-valued weighted case in Chapter 6.

Chapter 3

Cluster Validation

3.1 Background

In section 2.4, Algorithm 2 was successful at finding differences between pairs of networks. It discovered nodes that give a good cluster for one graph while giving a bad one for the other. But how can we quantify the significance of our findings to reinforce the visual observations? We will discuss this issue in this chapter.

In statistics, a result is called statistically significant if it is unlikely to have occurred by chance [42, page 43][86]. Statistical significance is different from the standard use of the term "significance", which suggests that something is important or meaningful. In statistics, significant loosely means probably true. The amount of evidence required to accept that an event is unlikely to have arisen by chance is known as the significance level α or critical p-value. The p-value [42, 86] is the frequency or probability with which the observed event would occur in a traditional statistical hypothesis test if the null hypothesis were true. Hypothesis testing is the use of statistics to determine the probability that a given hypothesis is true. The usual process of hypothesis testing consists of four steps: formulate the null hypothesis H_0, identify a test statistic, compute the p-value and compare the p-value to an acceptable significance level α. If the obtained p-value is smaller than the significance level ($p \leq \alpha$), then the null hypothesis is rejected. In simple cases, the significance level is defined as the probability that a decision to reject the null hypothesis will be made when it is in fact true and should not have been rejected. After much research on hypothesis testing published in thousands of journals over the past years, a common consensus has emerged. Following this convention: we take $p \geq 5\%$ as "not significant", $1\% \leq p < 5\%$ as "significant" and $p < 1\%$ as "highly significant" [115].

A p-value is always computed in a hypothesis test. But in fact, there are several different possible ways to compute the p-value. A commonly used approach in computing a p-value is based on the interpretation of probability as the long-time frequency. In this case, the p-value is simply the proportion of samples which support the null hypothesis. The main challenge of this approach is to determine how many samples to take in the simulation. We do not need to estimate the density function $\widehat{f}$ of the randomized data and do not need to consider the error between the estimator $\widehat{f}$ and the real density function f. This approach will be described in the tests in section 3.3.1.1. An alternative way to compute a p-value is to fit a density curve to the frequency histogram. The disadvantage of this method is that we need to decide which type of density $\widehat{f}$ is a good estimator of f. We will focus our attention and efforts on the disadvantages and challenges of both computing approaches mainly in section 3.3.1.1 and section 3.3.1.2. For convenience, we label the commonly used approach *Approach 1* and the alternative *Approach 2* for computing the p-value in our context.

In computing the p-value by *Approach 2*, it would be useful to know the distribution F of the objective data in practice. The distribution is determined by its density f. In practice, it is not possible to compute the whole population data to find its true distribution. A common method is to use samples to estimate the population. To estimate a density curve this way is called **density estimation** [130]. Here we have samples $\xi_1, \ldots, \xi_n$ from a distribution F with density f, written

$$\xi_1, \ldots, \xi_n \sim f \tag{3.1}$$

and we want to estimate the probability density function f.

Perhaps the most widely used and the simplest way for estimating the density is to use the frequency histogram. A histogram is a simple method of nonparametric density estimation [130]. The frequency histogram estimator $\widehat{f}$ can be defined using intervals, or bins [57, 130]. Since histograms are not smooth, we can move to a kernel density estimation [57, 130] of f. Although a histogram or kernel density estimation may be sufficient to estimate the density curve and help to decide the distribution density type, they are not problem free: distributions which are almost the same can look different, depending on the choice of bins. The main challenge of using a histogram or kernel density estimate is to obtain the 'just right' density estimator $\widehat{f}$ of density f. The data are easily biased from f when they are oversmoothed and the variance of the data will be large if they are undersmoothed [130]. Particularly, for a kernel density estimate, the choice of a given smooth function, kernel K, is not crucial but the choice of bin width is important.

A more powerful way to decide a proper distribution for a set of data is using a "quantile-quantile plot" because this plot is independent of the choice of the bin width. In statistics, a quantile-quantile plot is a probability plot, a graphical method for comparing two probability distributions, by plotting their quantiles against each other. A quantile-quantile plot can also be used as a non-parametric method to compare two sets of data against each other, where the data-set sizes may be unequal [22, page 144], or as a parametric method to compare a data set against a theoretical model distribution [47, page 199][123, page 21], or, less commonly, two theoretical models against each other [45, page 144]. If the data sets agree or the observed set matches the theoretical, the plot will be on a straight line.

In this context, we use the quantile-quantile plot as another technique for estimating and examining the statistical distribution of data. That is, we do not have two sets of data in practice. We just use the quantile-quantile plot to see whether the randomized data quantiles match those of the candidate distribution. The following is a description of the idea of using quantile-quantile plot in this way.

For a given density function $f(x)$ and a given $0 < p < 1$ define the pth *quantile* of f as $z(p)$, where

$$\int_{-\infty}^{z(p)} f(x)dx = p. \tag{3.2}$$

Suppose we are taking T samples, Given a set of data points $\xi_1, \xi_2, \cdots, \xi_T$, a *quantile-quantile* plot is produced by

(i) placing the data points in increasing order: $\widehat{\xi_1}, \widehat{\xi_2}, \cdots, \widehat{\xi_T}$,

(ii) plotting $\widehat{\xi_k}$ against $z(k/(T+1))$.

The idea of choosing quantiles for equally spaced $p = k/(T+1)$ is that it 'evens out' the probability [57]. For large T, if the quantile-quantile plot produces points which lie approximately on a straight line of unit slope, then we may conclude that the data points 'look as though' they were drawn from a distribution corresponding to $f(x)$. It is easy to justify this. If we divide the x-axis into T bins where x is in the kth bin if it is closest to $z(k/(T+1))$, then, having evened out the probability, we would expect roughly one ξ_i value in each bin.

Figure 3.1 tests the quantile-quantile idea. There are $T = 100$ samples from $N(0,1)$ and $U(0,1)$ random number generators. Here $N(0,1)$ denotes the standard normal distribution: N stands for normal, 0 is the mean and 1 is the variance. $U(0,1)$ denotes a uniform distribution over $(0,1)$. The $N(0,1)$ sample means and variances approach the true values 0 and 1 and the $U(0,1)$ sample means and variances approach the true values $\frac{1}{2}$ and $\frac{1}{12}$. We plot $\xi_1, \xi_2, \cdots, \xi_T$ on the x-axis against quantiles $z(k/(T+1))$ on the y-axis. There are four pictures here to show the four possible combinations arising from $N(0,1)$ or $U(0,1)$ random number samples against $N(0,1)$ or $U(0,1)$ quantiles. A reference line of unit slope is added to each plot. As expected, the data set matches well with the 'correct' quantiles and very poorly with the 'incorrect' quantiles.

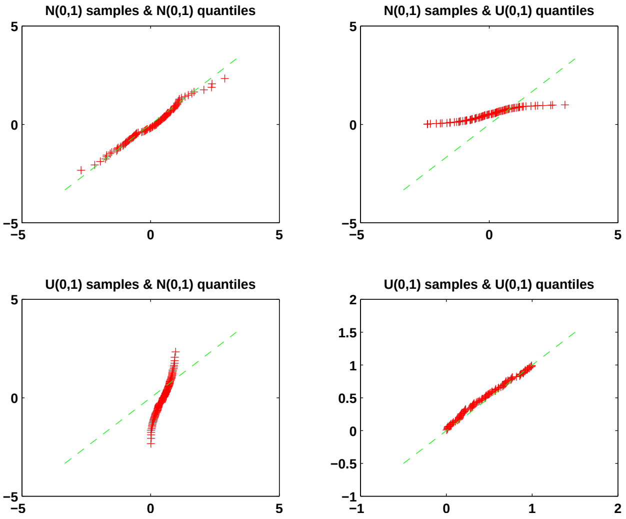


Figure 3.1: Quantile-quantile plots using $T = 100$ samples.

In the rest of this chapter, section 3.2 sets up a basic validation method that we use for the problem. We develop several alternative test routes in section 3.3 and apply these approaches to the synthetic data. In section 3.4, we pick out some typical approaches to test the findings in real data sets. Finally, we summarize the results in section 3.5.

3.2 Method

Now suppose we find τ nodes giving a good cluster for B but a poor cluster for A when the graphs are reordered by column ε from X^{-T}. The following general approach can be used in order to determine a p-value:

Step 1: Compute a measure of cluster quality, $c(A, B)$ ((3.3) or (3.4)), for the promising substructure consisting of those τ nodes in networks A and B reordered by column ε.

Step 2: Randomize the networks and obtain $\widehat{A}$ and $\widehat{B}$.

Step 3: Compute the GSVD for the randomized networks $\widehat{A}$ and $\widehat{B}$ and obtain a matrix $\widehat{X}^{-T}$.

Step 4: Compute the measure $c(\widehat{A}, \widehat{B})$ for the τ node 'cluster' in $\widehat{A}$ and $\widehat{B}$ reordered by column ε from $\widehat{X}^{-T}$.

Step 5: Repeat Step 2 to Step 4 T times and finally compute a p-value based on the value of $c(A, B)$ and the samples $c(\widehat{A}, \widehat{B})$.

In this process, we repeat Step 2 to Step 4 for many times and then for each instance of randomized networks $\widehat{A}$ and $\widehat{B}$, we get a measure $c(\widehat{A}, \widehat{B})$. After the loop, we now have a value $c(A, B)$ from our original experiment and lots of samples $c(\widehat{A}, \widehat{B})$ from randomized networks. Our goal is to test whether $c(A, B)$ is "unusually large".

For convenience and consistency, we use a random variable ξ to represent the randomized data $c(\widehat{A}, \widehat{B})$. We then can compute and plot the histogram of ξ and see whether $c(A, B)$ lies in a low-probability region. More formally, we are able to compute a p-value.

In order to produce a specific algorithm, we must decide how to compute the quality measure $c(A, B)$ in Step 1 and 4 and how to randomize the networks in

Step 2. In our test, we use two different ways to compute $c(A, B)$, which are denoted by c_1 and c_2 to avoid confusion.

$$c_1 = \frac{\text{density of edges of the cluster in } B}{\text{density of edges of the cluster in } A}, \tag{3.3}$$

$$c_2 = \frac{(\text{density of edges within the cluster in } B)\big/(\text{density of edges outside the cluster in } B)}{(\text{density of edges within the cluster in } A)\big/(\text{density of edges outside the cluster in } A)}. \tag{3.4}$$

The density $f(s)$ of a cluster s was defined as

$$f(s) = \frac{|E(s)|}{|s|}. \tag{3.5}$$

Here, $|E(s)|$ represents the actual number of edges in the cluster s, and $|s|$ is the maximum possible number of edges.

Here, computing c_1 is simpler than that of c_2. In both definitions of $c(A, B)$, a large ratio corresponds to a better result since we assume in the beginning of this section that we are aiming to validate a good cluster found in B which is a poor cluster for A. To validate the opposite pattern: a good cluster found in A which is a poor one in B, we can reciprocate the fraction in equation (3.3) and equation (3.4). Then a large value of $c(A, B)$ always corresponds to a better result. An empty substructure with no edges may occur in graphs A and B. So to avoid a zero value of the denominator in c_1 and c_2, we add one to the corresponding denominator if the density of edges within a cluster in one graph equals to 1. We also compute $c(\widehat{A}, \widehat{B})$ in the same way for each instance during the randomization in Step 4.

For the randomization in Step 2, we try these approaches:

1. *Erdös-Rényi*.

2. Permutation.

3. Redistribution.

In the late 1950s, Erdös and Rényi introduced two random graph models. These two models are usually referred to as $\mathcal{G}(\mathsf{n}, \mathsf{p})$ (Here, p is not a p-value) and $\mathcal{G}(\mathsf{n}, \mathsf{m})$ [33, 122]. $\mathcal{G}(\mathsf{n}, \mathsf{m})$ is a less commonly used version where one picks m edges out of the $\mathsf{n}(\mathsf{n}-1)/2$ possible ones between these n vertices at random. $\mathcal{G}(\mathsf{n}, \mathsf{p})$, sometimes called Gilbert [122], is a commonly studied approach to generate the *Erdös-Rényi* random graphs with each of the $\mathsf{n}(\mathsf{n}-1)/2$ possible edges between these n nodes assigned independently with probability p [33]. The properties of these random graph models have been well studied. When n is large($\mathsf{n} \to \infty$), these two models are closely related for

$$\mathsf{p} = \frac{2\mathsf{m}}{\mathsf{n}(\mathsf{n}-1)}.$$

Here, we can regard m as the average number of edges in $\mathcal{G}(\mathsf{n}, \mathsf{p})$. In our computational examples, we generated the Erdös and Rényi random graphs in MATLAB by using the functions from CONTEST[1], which is a controllable Test Matrix Toolbox for MATLAB. Particularly, we use the function `erdrey`(n, m) built in to CONTEST to generate a adjacency matrix for a $\mathcal{G}(\mathsf{n}, \mathsf{m})$ type random graph, where n is number of nodes in the graph and m is number of edges produced in the graph. The adjacency matrix for a $\mathcal{G}(\mathsf{n}, \mathsf{p})$ type random graph is produced by the function `gilbert`(n, p), where p is probability that any two nodes are neighbours among the total n nodes in the graph. More details can be found in [122]. We will show the corresponding results from the randomized data generated by *Erdös-Rényi* models in section 3.3.1.

In the second case, we permute the nodes in the original A and B. For each iteration, we choose the first τ nodes randomly after one permutation. We then simply apply the cluster quality measure onto it without computing the GSVD for each permutation in Step 3. The details and results will be described in the following section 3.3.2. The computation will take less time

[1] http://www.mathstat.strath.ac.uk/research/groups/numerical_analysis/contest/toolbox

than *Erdös-Rényi* and Redistribution since we ignore Step 3 in this method. For convenience, we simply call this method permutation in the following context. In our computational examples, we use the `randperm` routine built in to MATLAB to generate a random permutation of graphs A and B. For example, if we have n nodes originally ordered from 1 to n in graph A and B, we then use the reordering from `randperm(`n`)` to permute the graphs and produce the randomized graph $\widehat{A}$ and $\widehat{B}$. Then we directly compute the measure $c(\widehat{A}, \widehat{B})$ for the 'cluster' consisting of the first τ nodes in the randomized graphs.

We also try a third randomization way: First, we go along the rows of A one at a time and redistribute the numbers in that row. We do this for every row and then for every column of A to obtain $\widehat{A}$. Then we apply the same operation to B to produce $\widehat{B}$. We refer to this randomization method as redistribution for convenience in the following context. The corresponding results are listed in section 3.3.3.

In Step 5, we use both approaches mentioned in section 3.1, *Approach 1* and *Approach 2*, to compute the p-value. For *Approach 1*, the p-value corresponds to the proportion of randomly sampled networks for which a better clustering could be found than the clustering on the original data A and B in our context. For *Approach 2*, we will focus on the *log-normal* density and this choice will be described in section 3.3.1.2. Here, the p-value is our estimation of the probability that a random network would give a better clustering result than the given data set.

In our tests, our null hypothesis H_0 is that the cluster quality that we discovered could have arisen from the class of random networks defined by Step 2. If the p-value is less than 0.05, this null hypothesis will be rejected and then we can say that our finding is "statistically significant at the 5% level". Or we can express this significant finding in another way: it is very unlikely that this level of cluster quality or higher from the real data would arise if we take a random network from the class defined by Step 2. For an even smaller p-value, we can claim to classify that our findings is 'significant' if $1\% \leq p < 5\%$ or 'highly significant' if the p-value is even smaller than 0.01, as we stated in section 3.1.

3.3 Test on Synthetic Data

In this section, unless we specify, we will randomize the data in single test 1000 times.

3.3.1 *Erdös-Rényi*

In this section, we discuss the results from the randomized data generated by either of the *Erdös-Rényi* random graph models. Since Algorithm 2 is more promising, we apply our cluster validation method to the findings for the synthetic data with Algorithm 2 (Figure 2.4). In the previous synthetic test, we found the six nodes 18,20,16,15,19,17 form a cluster in B, but not in A, when reordered with the first column of X^{-T}. To verify the findings in the opposite pattern, we also repeat the experiment for the same synthetic graphs reordered by the final column from X^{-T}. There is a cluster formed by the eight nodes 7,9,10,15,14,11,6,13, which is a good cluster for A and a poor one for B. For clarity, we call the former cluster *Cluster 1* and the latter one *Cluster 2* in the following tests.

3.3.1.1 Compute a p-value by ***Approach 1***

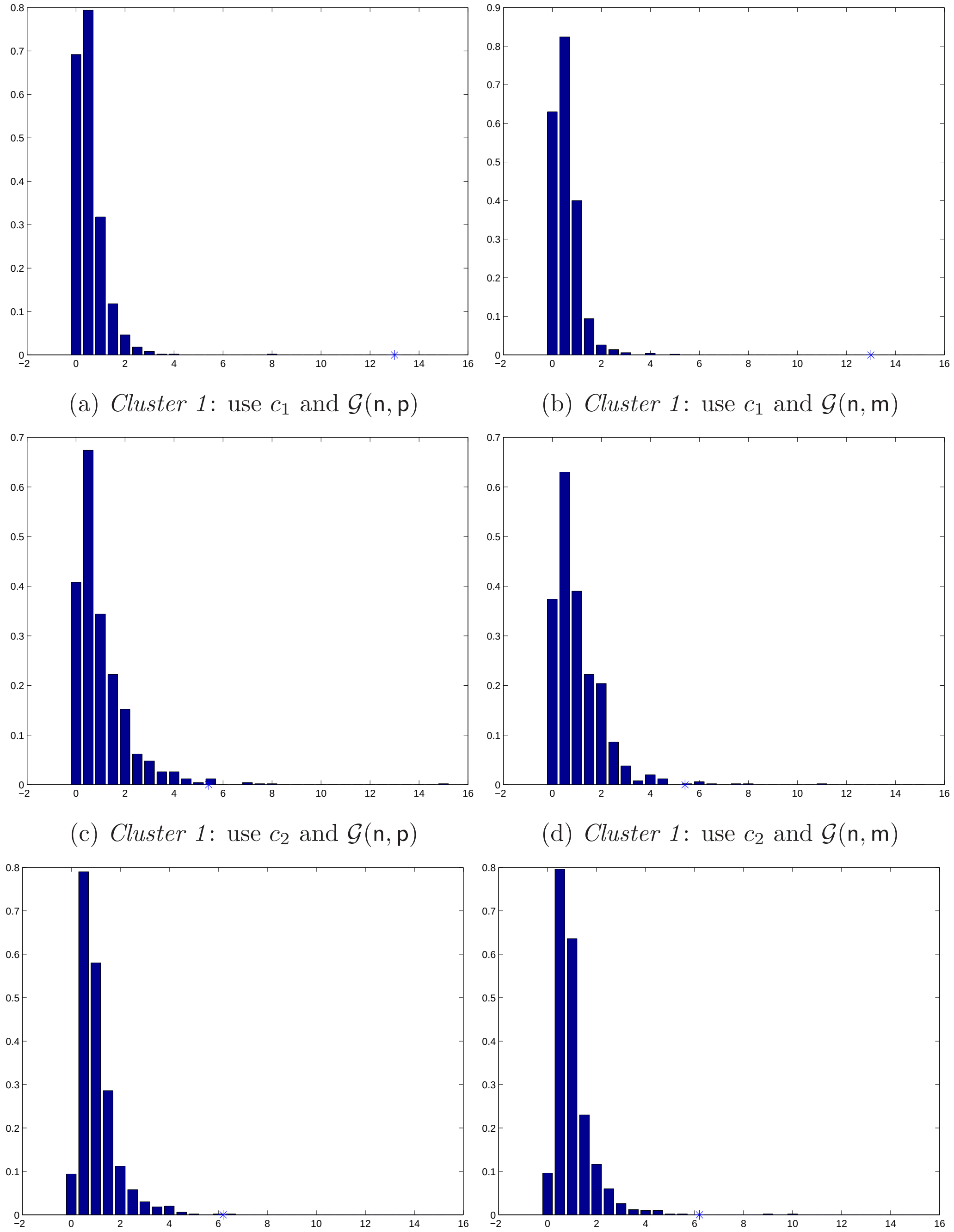
We begin with *Cluster 1*. We use c_1, the simpler definition of $c(A, B)$, in the initial trial. We use a marker '$*$' to point out the value of this original cluster quality measure, $c(A, B)$, in the histograms. Figure 3.2a and Figure 3.2b give the histograms for the data randomized by $\mathcal{G}(\mathsf{n}, \mathsf{p})$ and $\mathcal{G}(\mathsf{n}, \mathsf{m})$ separately. For both cases, $c_1 = 13$ and p-value $= 0$. As we stated in section 3.2 for the two random graph models of *Erdös-Rényi*, we emphasize that the p-value has nothing to do with the p of $\mathcal{G}(\mathsf{n}, \mathsf{p})$ model of *Erdös-Rényi*. A p-value of 0 is perfect in the sense that no observed event occurs to support H_0. We can then claim that this good cluster is highly significant in this simulation. More formally, we can conclude that the level of success of the good cluster is unlikely to arise by chance.

We then try c_2 (3.4), the second definition of $c(A, B)$, to check the same substructure again in Step 1 and Step 4 while other steps for the validation process are the same as the previous test. The corresponding histograms of the randomized data generated by $\mathcal{G}(\mathsf{n}, \mathsf{p})$ and $\mathcal{G}(\mathsf{n}, \mathsf{m})$ are given in Figure 3.2c and Figure 3.2d individually. Here, $c_1 \approx 5.4167$, p-value $= 0.01$ for $\mathcal{G}(\mathsf{n}, \mathsf{p})$ and p-value $= 0.007$ for $\mathcal{G}(\mathsf{n}, \mathsf{m})$. The p-values here are not zero but they are very small so that we can make the same conclusion as before that the finding is 'significant'.

We also extend the experiment to the object *Cluster 2* with the opposite pattern. We only use c_2 to compute $c(A, B)$ in this pattern but try both models of *Erdös-Rényi*. Figure 3.2e and Figure 3.2f give the corresponding histograms. In both cases, the marker '$*$' represents the value of $c_2 \approx 6.1986$ and p-value $=$ 0.002. That is, we can say that this good cluster is 'highly significant' in the graphs.

In the above tests, the p-values are all very small or zero. That is, our validation method is successful in showing that the good clusters we found in A and B are both significant. Then we turn back to the disadvantage of *Approach 1*, we need pay attention to the number of samples to be taken in the experiment by *Approach 1*.

In theory, the risk R, which is error (3.10) between the estimated density function and the real density function, decreases to 0 at rate $T^{-2/3}$ [130] with a choice of an optimal bin width for the histogram. This can be written in the



(a) *Cluster 1*: use c_1 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$

(b) *Cluster 1*: use c_1 and $\mathcal{G}(\mathsf{n}, \mathsf{m})$

(c) *Cluster 1*: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$

(d) *Cluster 1*: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{m})$

(e) *Cluster 2*: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$

(f) *Cluster 2*: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{m})$

Figure 3.2: Histograms of 10^3 samples produced by *Erdös-Rényi*.

following formulation:

$$R \sim \frac{C}{T^{2/3}}, \tag{3.6}$$

where $C = (3/4)^{2/3}(\int (f'(x))^2 dx)^{1/3}$ at a fixed sample x and T represents the number of samples and f denotes the probability density. More details of the optimal number of bins will be given in section 3.3.1.2.

In practice, *Approach 1* does not estimate the density so that in fact we do not need to consider the optimal bin width. Equation (3.6) indicates that the risk is smaller when we use a larger number of samples T. However, it is difficult to test a huge number T in practice since the whole computation will be very expensive. The risk will converge to 0 very slowly when the sample size increases. We try to find a threshold value of so that we can take it as a candidate as a proper sample size for the tests by *Approach 1.* We can complete this task by plotting a relationship figure between the risk and sample size. A difficulty is that we do not know the constant C in (3.6). We can simply use the widely used Gaussian density to compute C here, since estimating f is not our main task in *Approach 1.* Figure 3.3 shows the resulting relationship between the number of samples and the risk.

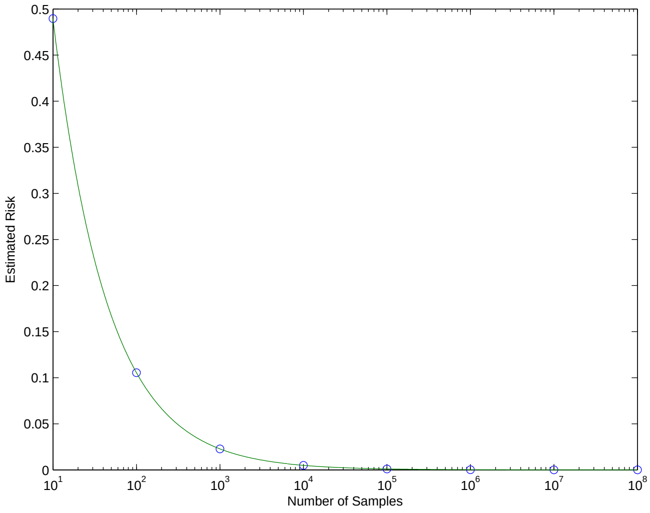
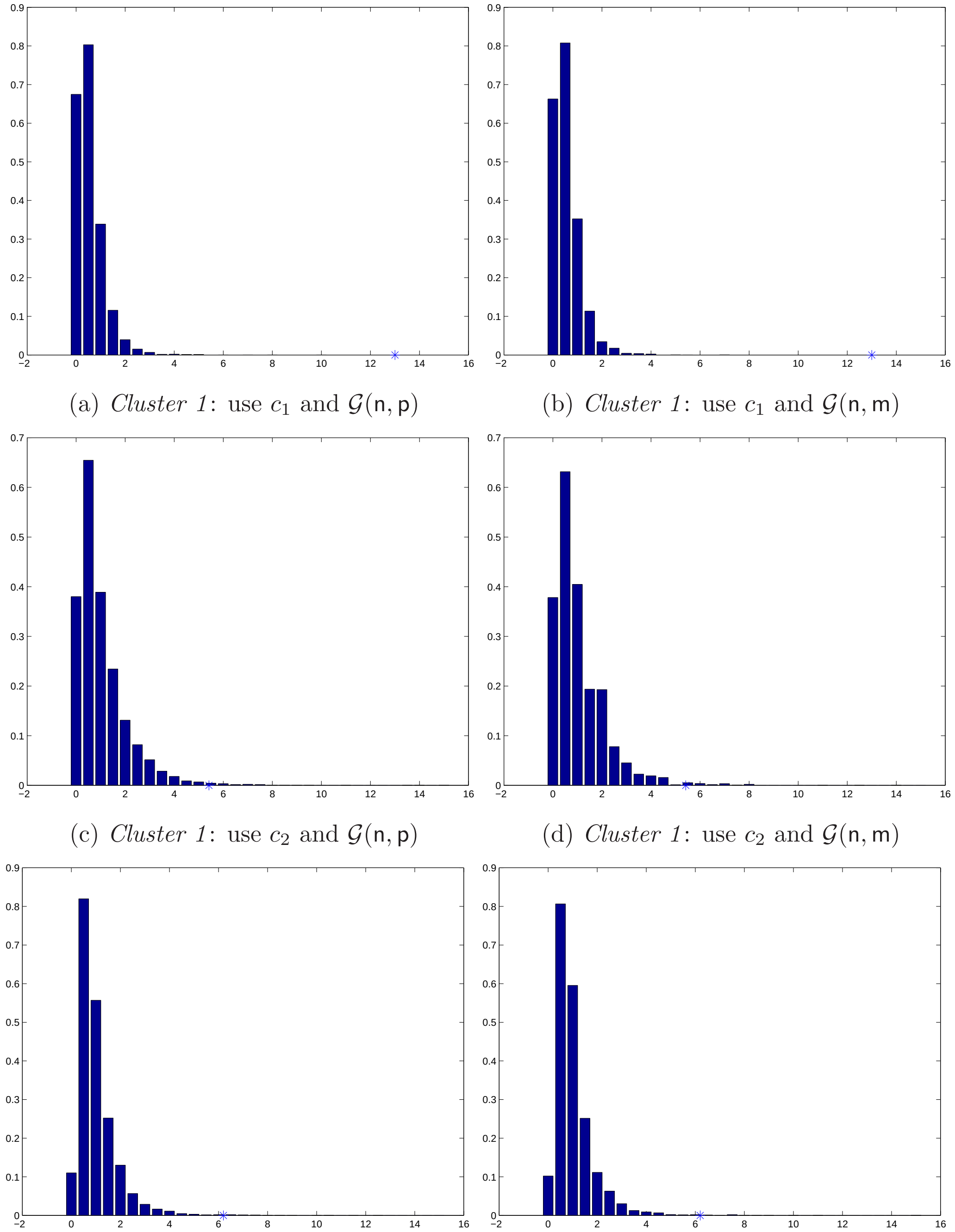


Figure 3.3: The estimated risk versus the number of samples.

An intrinsic choice of number of samples can be estimated by looking for the "elbow" at which this curve ceases to decrease significantly with added number of samples. In Figure 3.3, the most promising choice is around 10^3 or 10^4. Therefore, we extend our test to 10^4 samples with the same validation process used for sample size 10^3.

Figures 3.4a and 3.4b give the histograms using c_1 and either models of *Erdös-Rényi* to validate *Cluster 1* with 10^4 samples, the corresponding p-value = 0 for both tests. Figure 3.4c gives the results for *Cluster 1* using c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$, p-value = 0.0072 in this test. The p-value = 0.0063 when we use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{m})$ on 10^4 samples of *Cluster 1*, and the corresponding histogram is given in Figure 3.4d. The histograms for *Cluster 2* are given in Figure 3.4e and Figure 3.4f. They use $\mathcal{G}(\mathsf{n}, \mathsf{p})$ and $\mathcal{G}(\mathsf{n}, \mathsf{m})$ to generate random data individually with the same cluster quality measure c_2. The corresponding p-value is 0.0033 and 0.003, respectively.

Table 3.1 gives a summary of the computations in this section by *Approach 1*. All the p-values here are very small. That is, our findings are always significant. It could be argued that c_2 is more informative than c_1 because using c_1 always makes p-value = 0. c_2 is more expensive in computation but this does not make a big difference to the overall computation time for the whole test. The $\mathcal{G}(\mathsf{n}, \mathsf{p})$ and $\mathcal{G}(\mathsf{n}, \mathsf{m})$ models are similar in producing a small p-value but $\mathcal{G}(\mathsf{n}, \mathsf{p})$ is more sensitive to the number of samples. For example, from Figures 3.2c to the test given in Figures 3.4c, the difference of p-values produced by c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$ from 10^3 samples to 10^4 samples is $0.01 - 0.0072 = 0.0028$ while this difference is only $0.007 - 0.0063 = 0.0007$ by using $\mathcal{G}(\mathsf{n}, \mathsf{m})$ and c_2 in Figure 3.2d and Figure 3.4c. In this degree, $\mathcal{G}(\mathsf{n}, \mathsf{m})$ may be more reliable but $\mathcal{G}(\mathsf{n}, \mathsf{p})$ is still more widely used than $\mathcal{G}(\mathsf{n}, \mathsf{m})$ as a random graph model. With a larger sample size 10^4, the p-values are decreased slightly for *Cluster 1* but increased for *Cluster 2*. No matter what the slight differences in these p-values, the choices of sample size



(a) *Cluster 1*: use c_1 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$

(b) *Cluster 1*: use c_1 and $\mathcal{G}(\mathsf{n}, \mathsf{m})$

(c) *Cluster 1*: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$

(d) *Cluster 1*: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{m})$

(e) *Cluster 2*: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$

(f) *Cluster 2*: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{m})$

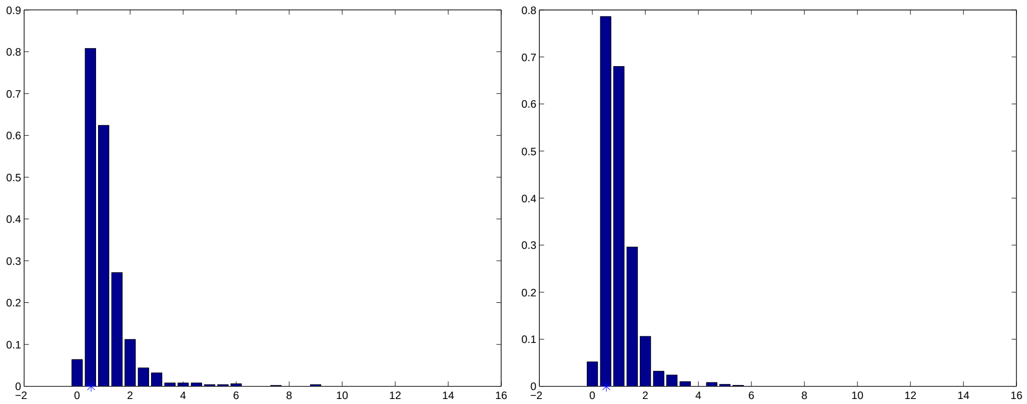
Figure 3.4: Histograms of 10^4 samples produced by *Erdös-Rényi*.

10^3 and 10^4 are both acceptable. Considering that the computation cost will increase rapidly with a increasing sample size, we generally take 10^3 samples in the following tests. We also list the value of $c(A, B)$ in the table, which denotes the location of the marker '$*$' in the corresponding histograms.

Table 3.1: Test on Randomized Data produced by *Erdös-Rényi* with *Approach 1.*

Object	$\mathbf{c(A, B)}$	**Graph model**	$\mathbf{T = 10^3}$	$\mathbf{T = 10^4}$
Cluster 1	$c_1 = 13$	$\mathcal{G}(\mathsf{n}, \mathsf{p})$	$p = 0$ (Figure 3.2a)	$p = 0$ (Figure 3.4a)
		$\mathcal{G}(\mathsf{n}, \mathsf{m})$	$p = 0$ (Figure 3.2b)	$p = 0$ (Figure 3.4b)
	$c_2 = 5.4167$	$\mathcal{G}(\mathsf{n}, \mathsf{p})$	$p = 0.0100$ (Figure 3.2c)	$p = 0.0072$ (Figure 3.4c)
		$\mathcal{G}(\mathsf{n}, \mathsf{m})$	$p = 0.0070$ (Figure 3.2d)	$p = 0.0063$ (Figure 3.4d)
Cluster 2	$c_2 = 6.1986$	$\mathcal{G}(\mathsf{n}, \mathsf{p})$	$p = 0.0020$ (Figure 3.2e)	$p = 0.0033$ (Figure 3.4e)
		$\mathcal{G}(\mathsf{n}, \mathsf{m})$	$p = 0.0.0020$ (Figure 3.2f)	$p = 0.0030$ (Figure 3.4f)

To show the applicability of our validation method, we also pick out some 'poor' patterns from the graphs and then apply the above validation process. We arbitrarily select an object with no visual pattern. This object is composed of the 12th to 18th components of the sorted final column from X^{-T} in the synthetic data (Figure 2.4). We use c_2 and both *Erdös-Rényi* models to test it. Figure 3.5a and Figure 3.5b give the corresponding results. The markers '$*$' in these histograms do not fall into a small probability area. The corresponding p-value is 0.769 and 0.798 for the two models. The large p-values show that validation fails since the object has no obvious pattern. In other words, this pattern is not classified as significant, which is consistent with our visual observation.



(a) use c_2 and $\mathcal{G}(\mathsf{n},\mathsf{p})$ (b) use c_2 and $\mathcal{G}(\mathsf{n},\mathsf{m})$

Figure 3.5: Histogram of bad pattern produced by *Erdös-Rényi* for 20th column of X^{-T}.

3.3.1.2 Compute a p-value by *Approach 2*

In this section, we use c_2 and $\mathcal{G}(\mathsf{n},\mathsf{p})$ for most of the tests from the empirical study in section 3.3.1.1. We use $\mathcal{G}(\mathsf{n},\mathsf{m})$ in some other tests, where specified. As we stated in section 3.1, the main difficulty in using *Approach 2* is to decide the density type against which to compare the data.

We now go back to check the shape of the original histograms given in section 3.3.1.1. Figure 3.2c is the histogram of the randomized data generated by c_2 and $\mathcal{G}(\mathsf{n},\mathsf{p})$ to validate *Cluster 1*. This histogram shows a more skewed distribution different to a normal distribution. It is caused by the fact that all the values of $c(\widehat{A},\widehat{B})$ are nonnegative by construction. There are some classic skewed continuous distributions, such as the *exponential* distribution and the *log-normal* distribution.

We try the *exponential* kernel first. Figure 3.6a gives the *exponential* kernel density estimate of the data. The curve looks to fit the histogram well but it is hard to conclude definitely that the data fits an *exponential* distribution because the histograms only provide a rough visual clue to the distribution of the randomized data. A more reliable method is to check the corresponding quantile-quantile plot. In some computational experiments, we used the optimal bin width but do not give the accompanying figures. Nevertheless, in practice, the optimal bin width, or the optimal number of bins, can be computed by a simple formula for computing the cross validation score $\widehat{J}(h)$ [105]:

$$\widehat{J}(h) = \frac{2}{h(T-1)} - \frac{T+1}{h(T-1)T^2} \sum_{j=1}^{b} \widehat{p}_j^2, \tag{3.7}$$

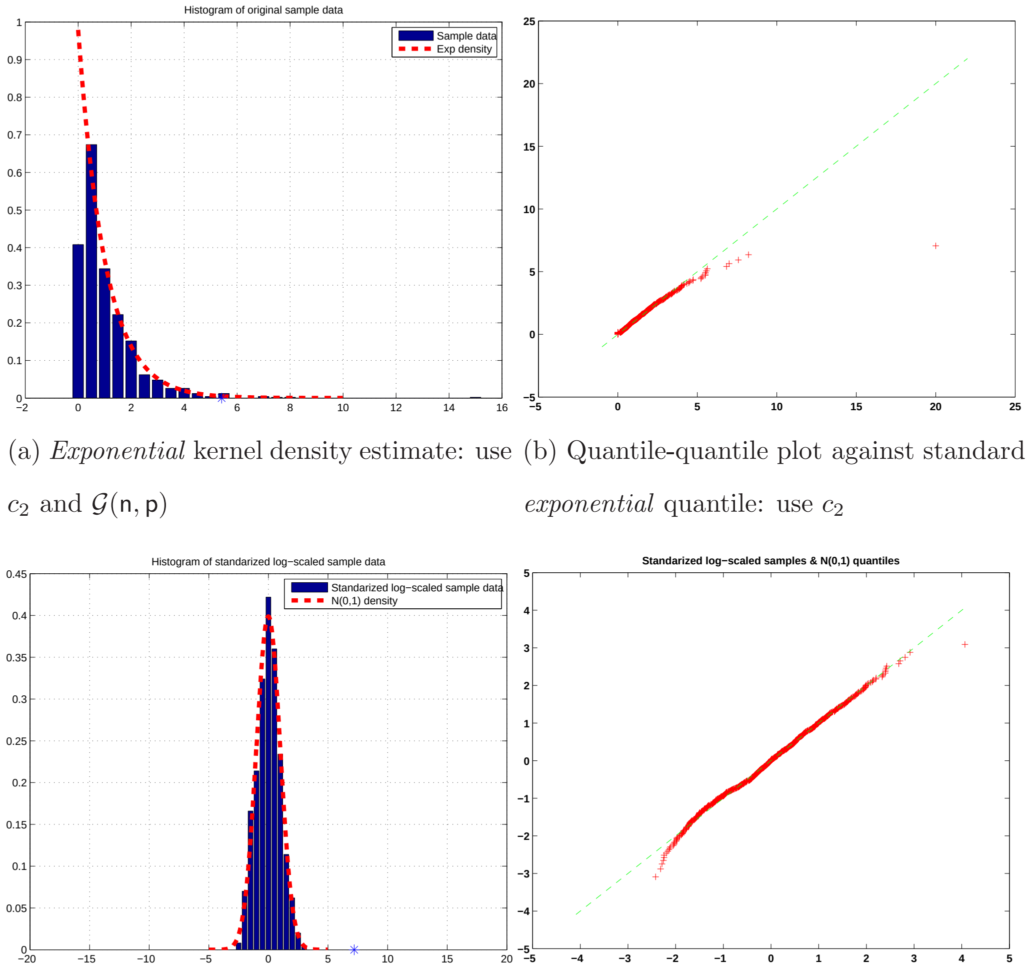
where T is the number of samples, b is the optimal number of bins, h is the optimal bin width and $h = 1/b$.

However, it does not matter whether we use a optimal bin width or we assign another fixed value of bin width. As we stated in section 3.1, we can use the quantile-quantile plot to check the fitness of the data from the given density further, because the quantile-quantile plot does not rely on the bin width of the histograms.

We show the corresponding quantile-quantile plot of the test on *Cluster 1* in Figure 3.6b. Here, p-value $= 0.004973$.

In Figure 3.6a and 3.6b, the *exponential* density seems be a good choice since the p-value is very small and the corresponding quantile-quantile plot is matching well. But closer inspection reveals some difficulties. First, the shape of the bars in the histogram is not monotonic as an *exponential* density curve should be, even if we apply an optimal bin width onto it. Furthermore, recent related research also provides evidence that skewed distributions in experimental science often closely fit the *log-normal* distribution [83]. The *log-normal* is a related continuous distribution to the normal distribution. We say $\xi \sim$ *Log-N*(μ, σ^2) is a *log-normal* distribution if $\xi = e^\zeta$ and $\zeta \sim N(\mu, \sigma^2)$. The data can be easily standardized to *log-normal*$(0, 1)$. Hence, what we need to do next is log scale the randomized data and see whether the log-scaled data fits a normal density.

To check the idea of using *log-normal* density, we then log scale ξ to ζ, then



(a) *Exponential* kernel density estimate: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$

(b) Quantile-quantile plot against standard *exponential* quantile: use c_2

(c) Gaussian kernel density estimate: use c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$

(d) Quantile-quantile plot against $N(0, 1)$ quantile: use c_2 and Value 3

Figure 3.6: Validating *Cluster 1* by *Erdös-Rényi* with *Approach 2*.

standardize the data ζ to mean $\mu = 0$ and standard deviation $\sigma = 1$. These transformations can be computed from the following equations:

$$\zeta = \ln(\xi), \tag{3.8}$$

$$\widehat{\zeta} = \frac{\zeta - \mu}{\sigma}. \tag{3.9}$$

Here, we use μ and σ to represent the sample mean value and standard deviation value of ζ, then use $\widehat{\zeta}$ to represent the standard data that will be tested for $\widehat{\zeta} \sim N(0, 1)$.

With a given normal kernel, we utilize both techniques, kernel density estimation and quantile-quantile plotting, to estimate the density of the randomized data $\widehat{\zeta}$ for *Cluster 1*. We show the Gaussian kernel density estimate in Figure 3.6c and the corresponding quantile-quantile plot in Figure 3.6d. Using the *log-normal* fit, the p-value≈ 0 in this test. We also use the marker '$*$' to point out the value of the original ratio $c(A, B)$ (after standardization) on the x-axis in the kernel density estimate figures.

Now, we have small p-values and good quantile-quantile plot for both kernels: *exponential* and the *log-normal*. How can we choose the best density type from these two candidates? In practice, there are several ways to measure the error between the kernel density estimator $\widehat{f_n}(x)$ and the exact density functions $f(x)$ [130]. Among them, the *squared error* (or L_2) loss function is a widely used one. The *squared error* is:

$$L(f(x), \widehat{f_n}(x)) = (f(x) - \widehat{f_n}(x))^2. \tag{3.10}$$

The result of this computation shows that the error of using *exponential* density is much larger than that of *log-normal* density, even though they both have good quantile-quantile plots. This dictates that we choose *log-normal* as our promising density type. This choice may also be explained by Central Limit Theorem as discussed earlier.

We also consider some other issues in the tests. For example, to log scale ξ to ζ, we remove all the zero values in ξ by letting them equal one of the followings items:

Value 1. let $\xi_i = \delta$ if $\xi_i = 0$ and $\delta = \min_{\xi \neq 0}(\xi_i)$.

Value 2. let $\xi_i = 1$ if $\xi_i = 0$.

Value 3. let the numerator of the fraction ((3.3) or (3.4)) of $\xi_i = 1$ if $\xi_i = 0$.

We try these different values in several tests: Figure 3.6c and Figure 3.6d we use a Value 3 and the quantile-quantile plot is good. We now look at Figure 3.7, this is an example using c_1, $\mathcal{G}(\mathsf{n}, \mathsf{p})$ and Gaussian kernel for $\widehat{\zeta}$ on the same *Cluster 1*. The only difference in Figure 3.7 is that we take Value 1 to replace all the zero ξ values in this case. There is an obvious vertical line at the beginning of the data and the whole sequence looks more discrete. This is consequence of rounding the zero values to the same non-zero threshold. Although Figure 3.7 can show the *log-normal* density of the data, we decide to use Value 3 in preference thereafter considering that Value 1 and Value 2 may cause some vertical reference line or pieces of discrete data to appear in the quantile-quantile plots. Another important reason to encourage us to choose Value 3 is that we can get a smaller L_2 score.

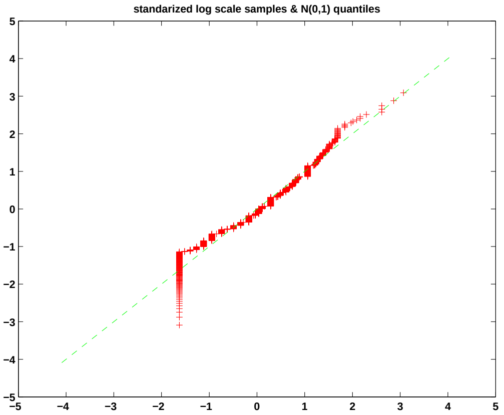
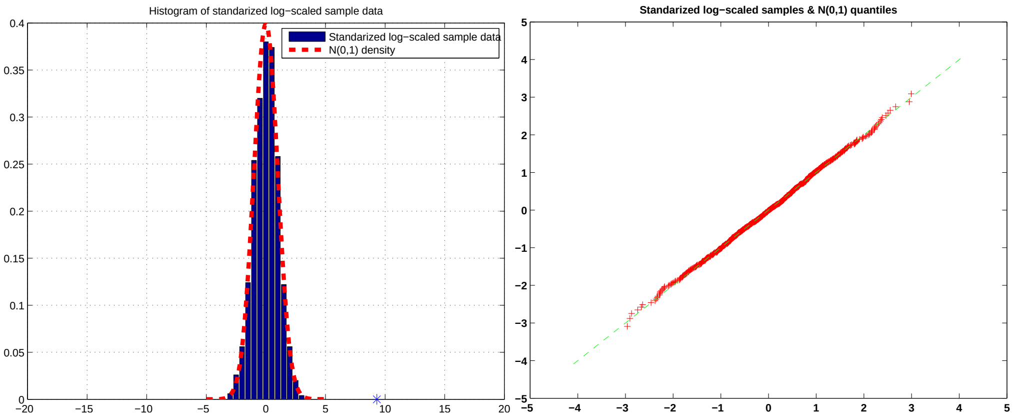


Figure 3.7: Quantile-quantile plot of ζ samples against $N(0, 1)$ quantile for *Cluster 1*: use c_1, $\mathcal{G}(\mathsf{n}, \mathsf{p})$ and Value 1.

To verify the findings in the opposite pattern, we repeat the experiment with c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{p})$ model on *Cluster 2*. In this test, $c_2 \approx 6.1986$ and the p-value $= 0$. Figures 3.8a and 3.8b give the corresponding results for both density estimation techniques.

We also test *Approach 2* with that same bad pattern that we use in section 3.3.1.1. We first use c_2, $\mathcal{G}(\mathsf{n}, \mathsf{p})$ and Value 3 to generate the histogram



(a) Gaussian kernel density estimate: use c_2 and $\mathcal{G}(\mathsf{n},\mathsf{p})$

(b) Quantile-quantile plot against $N(0,1)$ quantile

Figure 3.8: Validating *Cluster 2* by *Erdös-Rényi* with *Approach 2*.

in Figure 3.9a. The kernel density estimate is given in Figure 3.9c and the quantile-quantile plot is shown in Figure 3.9e. In this case, p-value$\approx$ 0.1523. This suggests that there is no pattern within the object, as we expected.

We change to $\mathcal{G}(\mathsf{n},\mathsf{m})$ instead of $\mathcal{G}(\mathsf{n},\mathsf{p})$ while other parameters stay the same, then produce the histogram in Figure 3.9b. The kernel density estimate and the quantile-quantile plot are given in Figure 3.9d and Figure 3.9f separately. Here, p-value$\approx$ 0.1172. Whether $\mathcal{G}(\mathsf{n},\mathsf{m})$ or $\mathcal{G}(\mathsf{n},\mathsf{p})$ is applied with *Approach 2*, the findings within this arbitrarily selected object with no pattern are not significant (p-values are larger than 0.05). These observations are consistent with those of the poor cluster in section 3.3.1.1.

3.3.2 Permutation

In section 3.3.1, we used the random graph model to generate randomized data. Here, we study the performance of the second way to randomize the data. As we stated in section 3.2, we skip Step 3 and only compute the GSVD for the original networks. We will describe results for *Approach 1* and *Approach 2* separately.

3.3.2.1 Compute a p-value by *Approach 1*

Based on the study in the previous sections, we begin the test on *Cluster 1* using the two cluster quality measures in (3.3) and (3.4). Figure 3.10a and Figure 3.10b give the corresponding histograms. We use c_1 in the former one with a resulting p-value $= 0$ and use c_2 in the latter test with a resulting p-value $= 0.029$.

We then use c_1 and c_2 separately on *Cluster 2* and produce the histograms in Figure 3.10c and Figure 3.10d by permutation. We show the corresponding p-values in Table 3.2.

Table 3.2: Test on Randomized Data produced by Permutation with *Approach 1*.

Object	$\mathbf{c(A, B)}$	$\mathbf{T = 10^3}$	$\mathbf{T = 10^4}$
Cluster 1	$c_1 = 13$	$p = 0$ (Figure 3.10a)	$p = 0$ (Figure 3.11a)
	$c_2 = 5.4167$	$p = 0.0290$ (Figure 3.10b)	$p = 0.0328$ (Figure 3.11b)
Cluster 2	$c_1 = 7.6667$	$p = 0.0090$ (Figure 3.10c)	$p = 0.0098$ (Figure 3.11c)
	$c_2 = 6.1986$	$p = 0.0.0070$ (Figure 3.10d)	$p = 0.0071$ (Figure 3.11d)

To address the main challenge of *Approach 1*, we just increase the sample size from 10^3 to 10^4. Figure 3.11a to Figure 3.11d show the histograms of the randomized data with this larger sample size produced by permutation combined with both c_1 and c_2.

Table 3.2 denotes the key features and corresponding results in all the tests in this section. All the p-values are smaller than 5% so that all the tests show our findings are significant for both 10^3 samples and 10^4 samples. Although it is hard to decide the exact number of samples that should be used for every test, we could compare a result with that of a larger sample size and make sure that the results are consistent. The data presented in Table 3.2 show that using a larger sample size gives less conclusive results. For example, using c_1 or c_2 on 10^3 samples makes a smaller difference than that of 10^4 samples. Nearly all the p-values are decreased slightly with a larger sample size 10^4. Furthermore, we are inclined to take 10^3 samples in the test because we also need to consider the computational cost. c_2 may be a better choice to produce randomized data, in the sense that it leads to a more stringent test, because c_1 always makes the p-value $= 0$.

3.3.2.2 Compute a p-value by ***Approach 2***

Based on the study of the density estimate in section 3.3.1.2, in this subsection we only adopt the *log-normal* distribution in kernel density estimate and the quantile-quantile plot. We use Value 3 to replace all the zero values in ξ before we log scale it.

Figures 3.12a-3.12d illustrate the corresponding kernel density estimate and the quantile-quantile plot for *Cluster 1* by c_1 and c_2 separately. The corresponding p-values both approximate zero.

For *Cluster 2*, we repeat the same validation process and corresponding density estimate results are given in Figures 3.13a-3.13d, using c_1 and c_2 separately. The p-values are zero in both tests.

We obtained very similar results to those produced by Permutation with *Approach 1*, and reached the same conclusions with *Approach 2* as we did with *Approach 1*.

3.3.3 Redistribution

In this section, we undertake the cluster validation on the randomized data by redistribution. We follow the same test route in section 3.3.2 except for the randomization method and show the corresponding results in the following two subsections. This randomization method changes the structure of the graphs and keeps the same number of edges for each graph, but destroys symmetry.

3.3.3.1 Compute a p-value by *Approach 1*

First, we use the two cluster quality measures on *Cluster 1* and *Cluster 2* with 10^3 samples. The corresponding results are given from Figure 3.14a-3.14d. We only validate *Cluster 1* with a larger sample size 10^4 and list the results in Figure 3.15a and Figure 3.15b. We list all the p-values for these tests in Table 3.3.

Table 3.3: Test on Randomized Data produced by Redistribution with *Approach 1*.

Object	$\mathbf{c(A,B)}$	$\mathbf{T = 10^3}$	$\mathbf{T = 10^4}$
Cluster 1	$c_1 = 13$	$p = 0$ (Figure 3.14a)	$p = 0$ (Figure 3.15a)
	$c_2 = 5.4167$	$p = 0$ (Figure 3.14b)	$p = 0.0001$ (Figure 3.15b)
Cluster 2	$c_1 = 7.6667$	$p = 0.0010$ (Figure 3.14c)	
	$c_2 = 6.1986$	$p = 0$ (Figure 3.14d)	

Table 3.3 gives the key features and corresponding results in all the tests using redistribution with *Approach 1*. We do not apply the validation process on *Cluster 2* with a larger sample size 10^4 because the computation for redistributing the graphs is the most expensive of the three randomization methods. Redistribution also produces very small p-values for all the tests in this section, quantifying that our findings are meaningful. There is only a small difference in p-values when we increase the sample size from 10^3 to 10^4. We also list the value of $c(A, B)$ in the table because it denotes the location of the marker '$*$' in

the corresponding histograms and helps us to confirm visually that the p-values are very small.

3.3.3.2 Compute a p-value by ***Approach 2***

We only use the *log-normal* density for kernel density estimation and only use $N(0,1)$ quantiles in the corresponding quantile-quantile plots for the randomized data $\widehat{\zeta}$ in this section. We only use the Value 3 before log-scaling the data ξ.

First, we validate *Cluster 1* with a kernel density estimate and quantile-quantile plots on different quality measures. We give the corresponding results in Figure 3.16a to Figure 3.16d. Then we continue to validate *Cluster 2* in the same way and show the corresponding results in Figure 3.17a to Figure 3.17d. Here, all the p-values equal zero. We can also see how small the p-value is by looking at the location of markers '$*$' on the histograms, since they denote which bin the original ratio $c(A,B)$ falls into on the histograms.

This randomization method is consistent with the other approaches for validating the patterns, and we can even say in this case that all the findings are highly significant.

3.4 Test on Real Data

In section 3.3, our validation methods confirm the significance of the clusters found in synthetic data. Based on the insights gained from this study, we designed two test routes for the real data sets, based on the two main approaches that we investigated. We call them Test 1 and Test 2 as described in the following:

Test 1. Compute the p-value by *Approach 2* combined with c_2, $\mathcal{G}(\mathsf{n},\mathsf{p})$ and use Value 3 to replace the zero values in ξ for a *log-normal* given density kernel.

Test 2. Compute the p-value by *Approach 1* combined with c_2, $\mathcal{G}(\mathsf{n}, \mathsf{p})$.

In both tests, we choose to use c_2 since c_1 always resulted in a zero p-value. Furthermore, in the aspect of randomization, we decided to adopt $\mathcal{G}(\mathsf{n}, \mathsf{p})$ of *Erdös-Rényi* in both tests since it is a widely used model to generate random graphs, and $\mathcal{G}(\mathsf{n}, \mathsf{m})$ performed very similarly. In Test 1, we choose a *log-normal* distribution density for *Approach 2* because it produced a good kernel density estimate, good quantile-quantile plot and a smaller $L2$ score than an *exponential* one. Value 3 is used to replace the zero values in ξ considering that it can produce a smoother quantile-quantile plot.

In most of the following tests, we only apply these two tests with a sample size of 10^3. But as an extra check we occasionally see larger values.

In the previous test on Florentine family networks, we see that the 6 families, Lambertes, Peruzzi, Bischeri, Barbadori, Guadgani and Castellan, were placed in a well-connected group in B but not for A when the data was reordered according to the first column of X^{-T} (Figure 2.6). Here, we test this cluster by Test 1 and Test 2 separately. Figures 3.18a and 3.18b give the corresponding kernel density estimate and quantile-quantile plot for Test 1, while Figure 3.18c shows the histogram obtained from Test 2.

In Test 1, we obtain a small p-value of approximately 2×10^{-6}. But in Test 2, the p-value $= 0.101$. This p-value is larger than 0.05. Using c_1 instead of c_2 and $\mathcal{G}(\mathsf{n}, \mathsf{m})$ instead of $\mathcal{G}(\mathsf{n}, \mathsf{p})$ with a larger sample size 10^4 with *Approach 1* in Test 2 makes a tiny difference, producing a p-value that remains larger than 0.05. Then we turn to the other two randomization methods, permutation and redistribution, with *Approach 1.* Figures 3.18d and 3.18e illustrate that smaller p-values are produced by *Approach 1* with these two alternative randomization methods. In these two alternative tests, we still use c_2. The corresponding p-values are 0.045 for permutation and only 0.033 for the redistribution. These p-values are shown in Table 3.4. These tests suggests that the cluster is statistically significant, since the majority of the p-values are below 0.05. However, the results also emphasize that the p-value computation depends on a number of choices, most notably the underlying null hypothesis.

We also tested the cluster in family networks by *Approach 2* with a *exponential* density kernel. The results also suggest using a *log-normal* density type, since the corresponding sample quantiles are highly biased against the *exponential* reference line in the quantile-quantile plot.

In the previous test for Read highland tribes networks given in Figure 2.8, we see that a set of six tribes, Masil, Gahuk, Ukudz, Asaro, Geham and Ove, are well-connected in A but not in B from the reordering from the final column of X^{-T}. Meanwhile another set of four nodes, Kotun, Nagad, Gama and Gavev are also seen to be grouped well. To avoid confusion, here we call the former set of nodes the bigger cluster and the latter one the smaller cluster. Then we apply Test 1 and Test 2 to these two clusters separately. We give the corresponding kernel density estimate in Figure 3.19a and the quantile-quantile plot in Figure 3.19c for Test 1 on the bigger cluster, and use Figure 3.19e to show the histogram for Test 2 on the same bigger cluster. The corresponding p-values are given in Table 3.4.

Then we repeat the tests on the smaller cluster of Read highland tribes. Figure 3.19b and Figure 3.19d give the density estimate for this smaller cluster by Test 1 and Figure 3.19f is the histogram of that smaller cluster by Test 2. The p-value results are given in Table 3.4. These small p-values (< 0.05) suggest that the bigger cluster and the small cluster are both statistically significant.

Finally, we apply Test 1 and Test 2 to macaque cortical connectivity networks to validate that the leading 38×38 submatrix of A reordered with the final column of X^{-T} in Figure 2.10 is a good cluster. Figures 3.20a and 3.20b are density estimates from Test 1 while Figure 3.21 shows the histograms produced by Test 2. Table 3.4 lists the corresponding values of p for all these tests. The corresponding p-values obtained are 0 under both Test 1 and Test 2. This suggests that the cluster consisting of the submatrix is statistically significant.

In Table 3.4, the value for the item "Location of $*$" denotes the value of original cluster quality measure $c(A, B)$ or that original value after log-scale and standardization in Test 1. We can see where it lies in the histogram of the randomized data and judge whether it falls into a low probability area. The item "Test Method" is used to describe which test we apply to the object. Usually we only use Test 1 and Test 2 as we described in the beginning of this section but in some cases we also tested further with other randomization methods to verify that we obtained a small p-value and show we do find a significant cluster in the real data.

Table 3.4: Test on Real Data Sets.

Test method	**Object**	**Figure**	**Location of $*$**	**p-value**
Test 1	family	3.18a,3.18b	4.6026	2×10^{-6}
	bigger cluster in tribes	3.19a,3.19c	6.4906	4.2746×10^{-11}
	smaller cluster in tribes	3.19b,3.19d	7.9259	1.1102×10^{-15}
	macaque brain	3.20a,3.20b	16.3178	0
Test 2	family	3.18c	4.5000	0.1010
	bigger cluster in tribes	3.19e	5.6875	0.0140
	smaller cluster in tribes	3.19f	4.9565	0.0040
	macaque brain	3.21	2.4069	0
c_2+permutation+*Approach 1*	family	3.18d	4.5000	0.0450
c_2+redistribution+*Approach 1*	family	3.18e	4.5000	0.0330

3.5 Summary

In section 3.3 and section 3.4, we tested a pair of synthetic networks with built-in clusters and three kinds of real networks with different approaches. The small

p-values we obtain in these tests indicate that all of our findings are statistically significant and some of them are highly significant. As a separate test, we obtained large p-values for collections of nodes chosen arbitrarily, without attempt to understand the structure. These results suggest that the validation method described in section 3.2 is efficient and powerful. We summarize our findings as follows:

1. As we stated in section 3.2, we tried two different definitions of $c(A, B)$, c_1 and c_2, in Step 1 and Step 4. In most cases, they produce similar p-values. But we prefer c_2 to c_1 in the forthcoming tests for real data since c_1 always makes p-value $= 0$ while c_2 can produce a slightly bigger but still meaningful results.

2. The second key point arising from these computations concerns the randomization methods in Step 2. In fact, there are many possible ways to randomize the data but it is not practical to try every one. $\mathcal{G}(\mathsf{n}, \mathsf{p})$ and $\mathcal{G}(\mathsf{n}, \mathsf{m})$ are useful random graph models with a very similar output of p-values. We used $\mathcal{G}(\mathsf{n}, \mathsf{p})$ for most cases since this model is more commonly used in network science. Redistribution is much more expensive in computation time and breaks the symmetry of the original graphs. Permutation is perhaps the easiest and fastest way to randomize the data because we do not need to compute the GSVD in Step 3 for each instance for this method. In practice, the three randomization methods that we tried have similar performance.

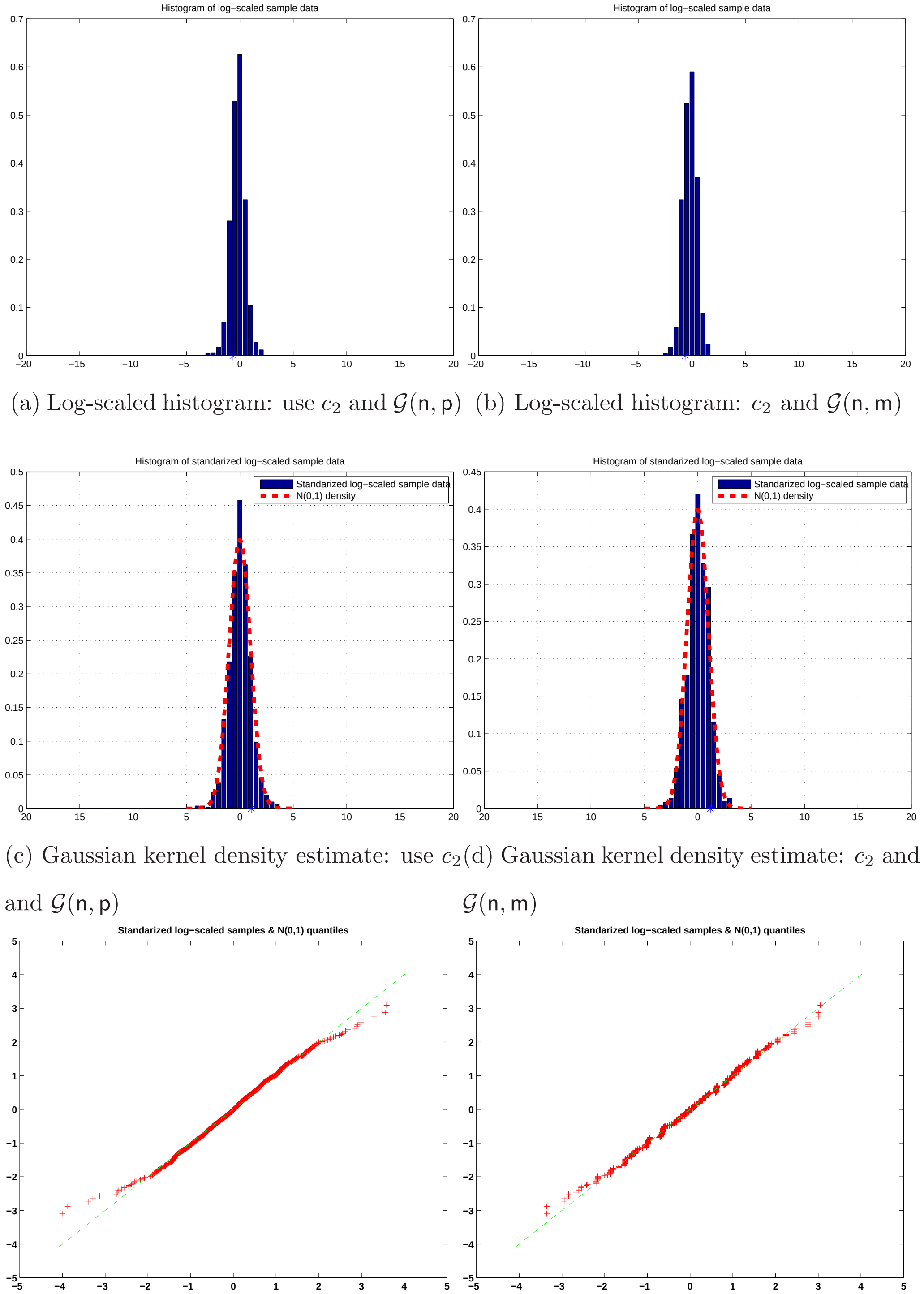
3. In the final Step 5, we developed two approaches to calculate a p-value: *Approach 1* and *Approach 2*. *Approach 1* is cheaper but more sensitive to the number of samples we take in the test. From all of these studies we can say that, with this size of network and cluster, 10^3 samples is enough since it just makes a small difference to the p-values if we extend to 10^4.

It is not practical for us to extend the experiment to a much larger sample size than 10^3 because the computation task would exceed that possible with our computers. So we adopted a sample size of 10^3 for the tests on real data of corresponding size. Furthermore, we also computed the estimated risk to support the choice of 10^3 for *Approach 1*. *Approach 2* also produced small p-values. For the techniques that we tried, which were kernel density estimate, quantile-quantile plot and $L2$ loss function score, we found that the *log-normal* distribution is a more appropriate choice than the *exponential* distribution.

4. Computationally, *Approach 1* may be better than *Approach 2* not only because it is easier to compute but also because it produces more informative nonzero p-values. To show our validation method is appropriate, we also checked it with a poor pattern in synthetic data and obtained big p-values, corresponding to a lack of statistical significance.

In summary, we can claim that (a) it is possible to quantify the significance of our results, and (b) our findings in previous chapters on real data are meaningful with small p-values.

The cluster validation method in this chapter was designed for binary matrices, that is, unweighted graphs. In Chapter 6, we will modify the corresponding concepts and generalize this cluster validation method to the case of weighted graphs.



(a) Log-scaled histogram: use c_2 and $\mathcal{G}(\mathsf{n},\mathsf{p})$

(b) Log-scaled histogram: c_2 and $\mathcal{G}(\mathsf{n},\mathsf{m})$

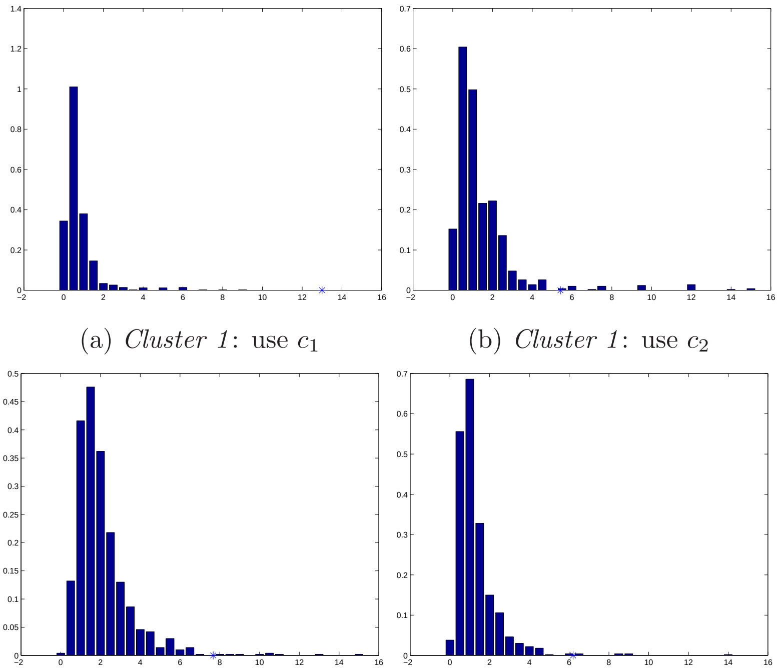
(c) Gaussian kernel density estimate: use c_2 and $\mathcal{G}(\mathsf{n},\mathsf{p})$

(d) Gaussian kernel density estimate: c_2 and $\mathcal{G}(\mathsf{n},\mathsf{m})$

(e) Quantile-quantile plot against $N(0,1)$ quantile: use c_2 and $\mathcal{G}(\mathsf{n},\mathsf{p})$

(f) Quantile-quantile plot against $N(0,1)$ quantile: use c_2 and $\mathcal{G}(\mathsf{n},\mathsf{p})$

Figure 3.9: Validating bad pattern by *Erdös-Rényi* with *Approach 2* for 20th column of X^{-T}.



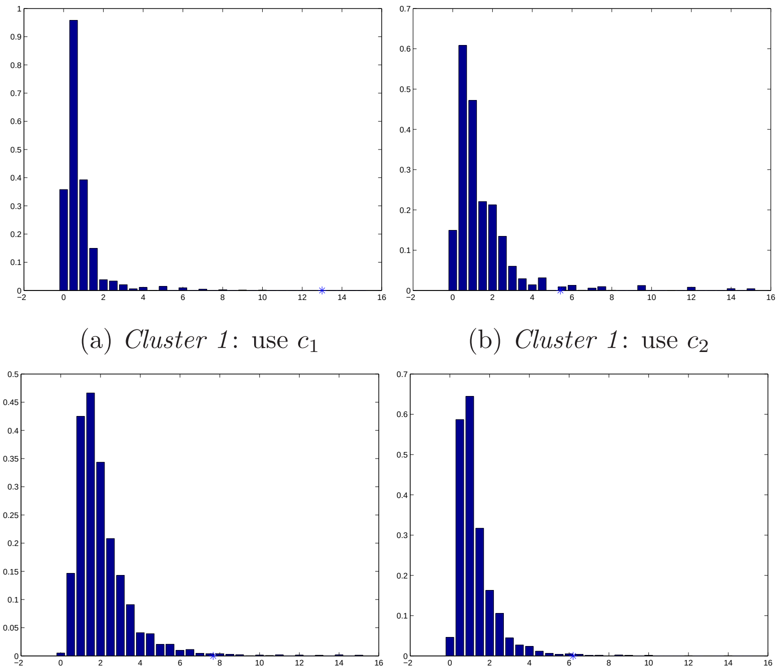
(a) *Cluster 1*: use c_1

(b) *Cluster 1*: use c_2

(c) *Cluster 2*: use c_1

(d) *Cluster 2*: use c_2

Figure 3.10: Histograms of 10^3 samples produced by Permutation.



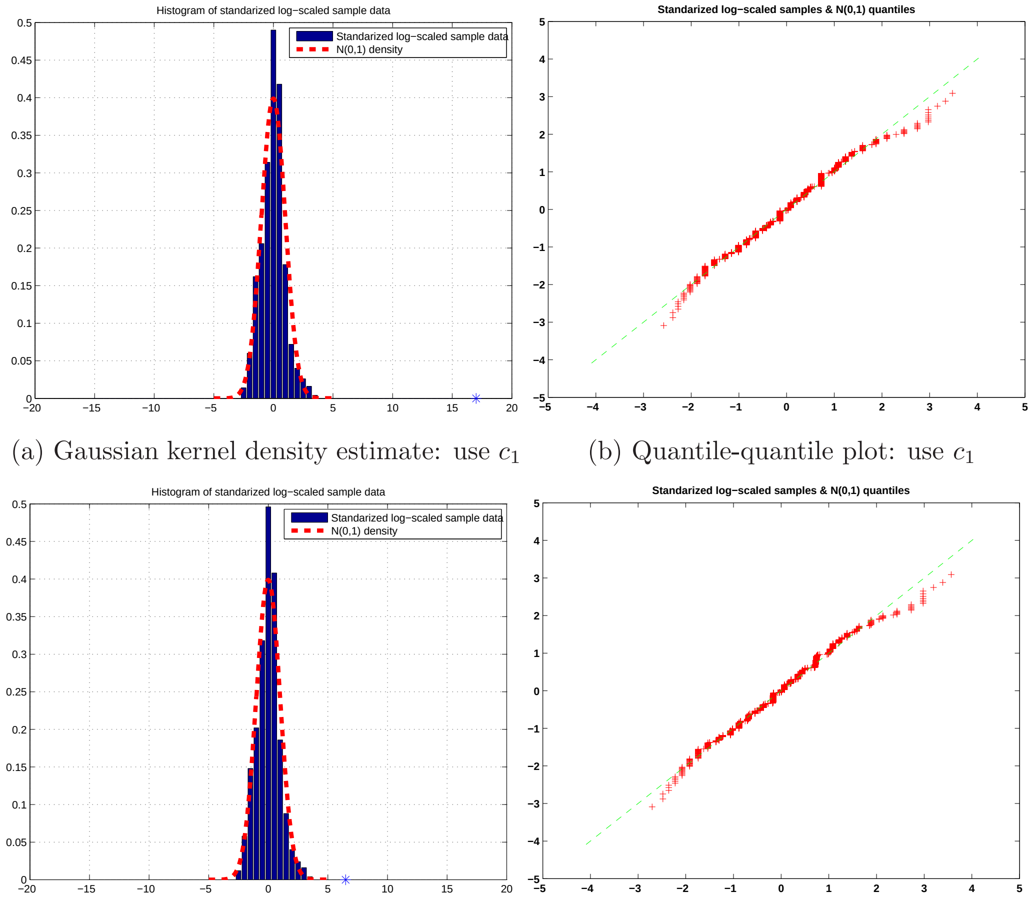
(a) *Cluster 1*: use c_1

(b) *Cluster 1*: use c_2

(c) *Cluster 2*: use c_1

(d) *Cluster 2*: use c_2

Figure 3.11: Histograms of 10^4 samples produced by Permutation.



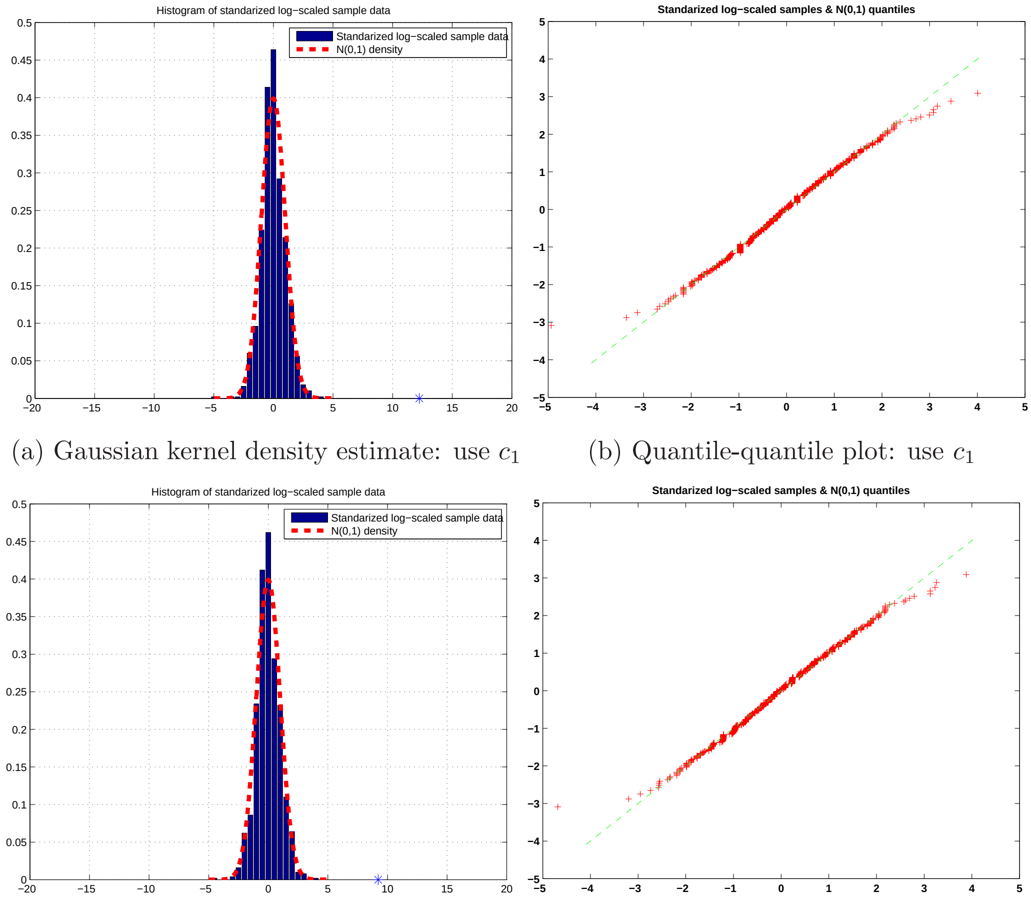
(a) Gaussian kernel density estimate: use c_1

(b) Quantile-quantile plot: use c_1

(c) Gaussian kernel density estimate: use c_2

(d) Quantile-quantile plot: use c_2

Figure 3.12: Validating *Cluster 1* by Permutation with *Approach 2*.



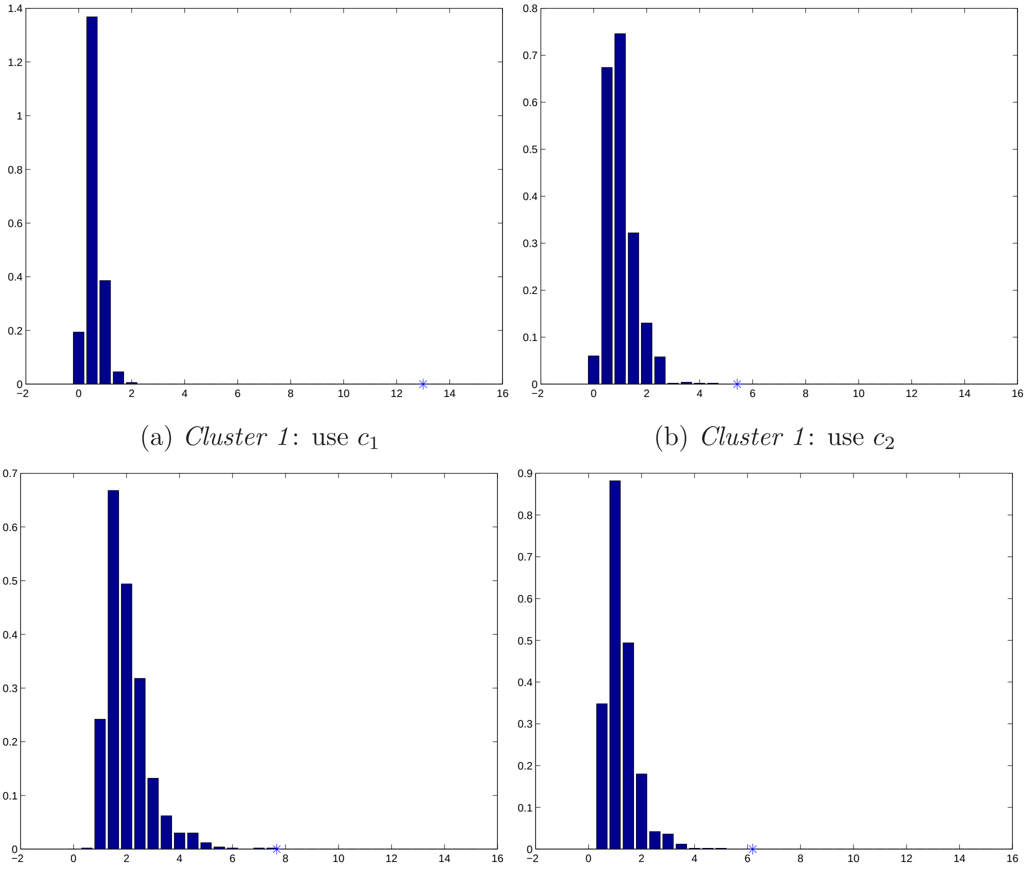
(a) Gaussian kernel density estimate: use c_1

(b) Quantile-quantile plot: use c_1

(c) Gaussian kernel density estimate: use c_2

(d) Quantile-quantile plot: use c_2

Figure 3.13: Validating *Cluster 2* by Permutation with *Approach 2*.



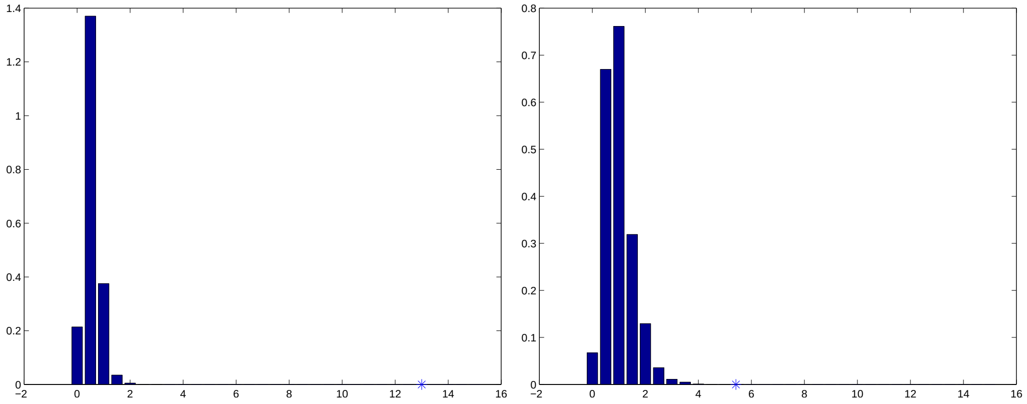
(a) *Cluster 1*: use c_1

(b) *Cluster 1*: use c_2

(c) *Cluster 2*: use c_1

(d) *Cluster 2*: use c_2

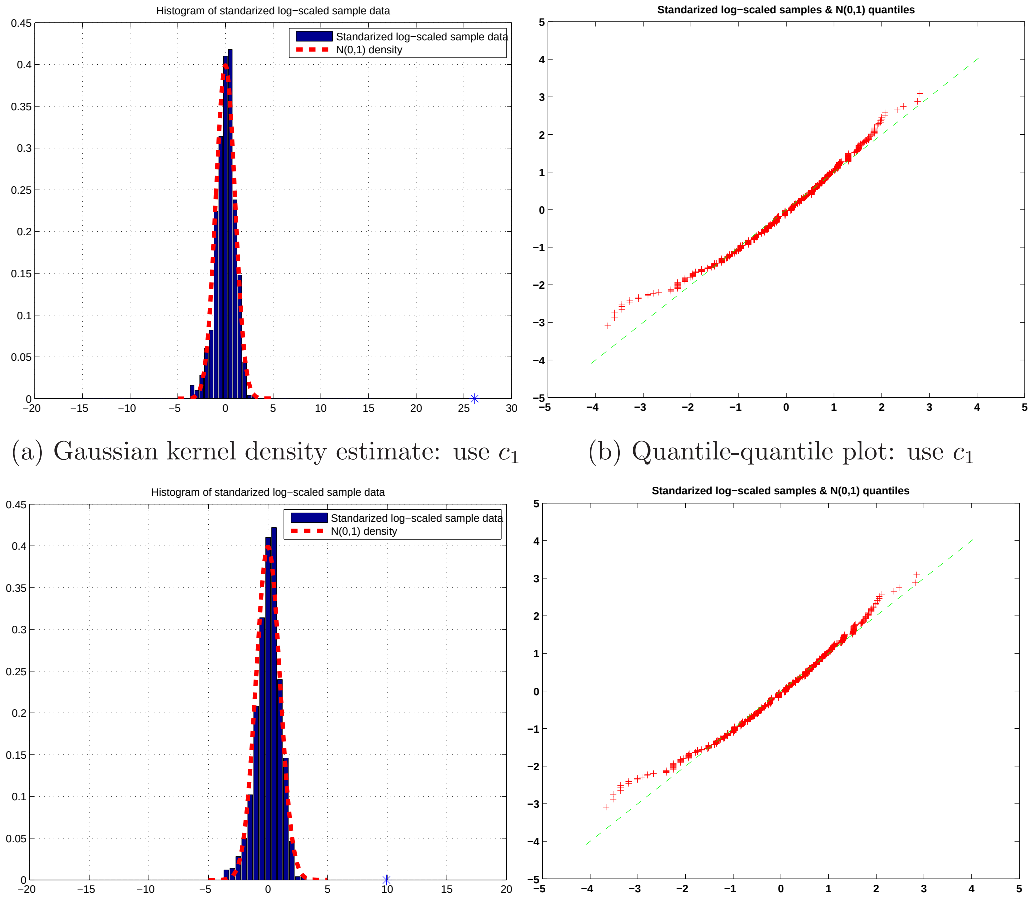
Figure 3.14: Histograms of 10^3 samples produced by Redistribution.



(a) *Cluster 1*: use c_1

(b) *Cluster 1*: use c_2

Figure 3.15: Histograms of 10^4 samples produced by Redistribution.



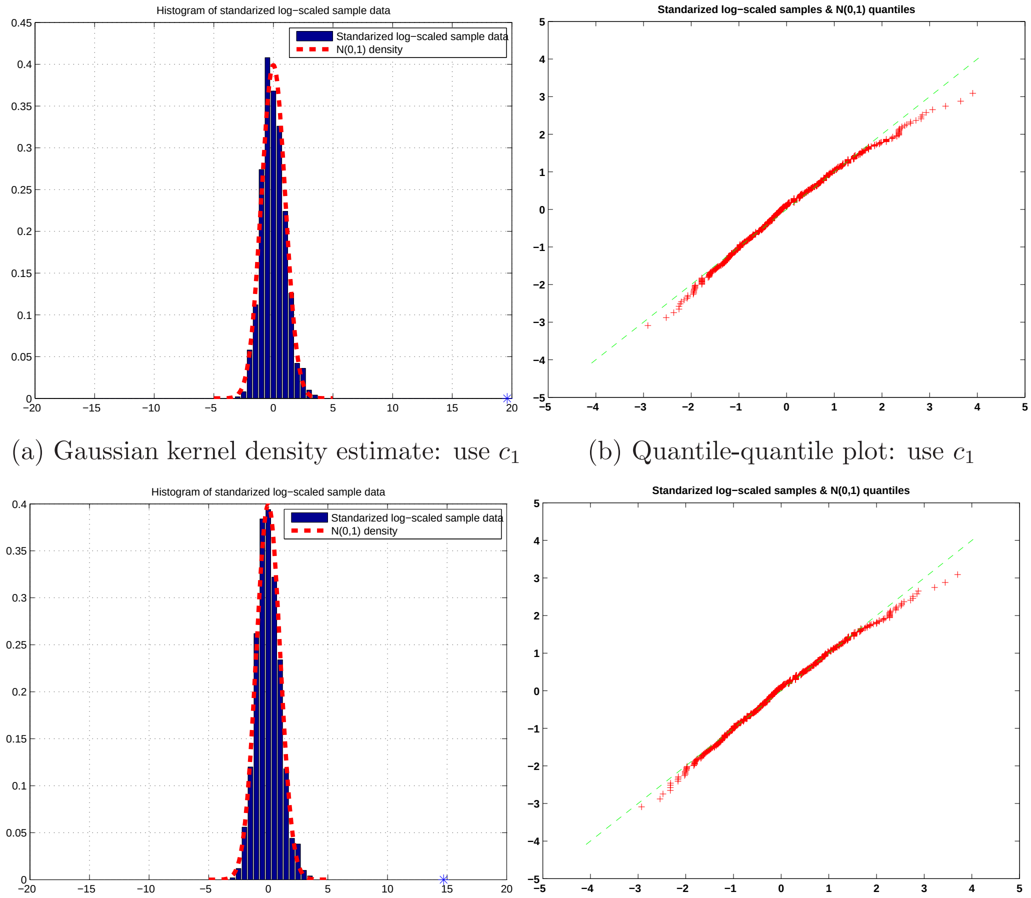
(a) Gaussian kernel density estimate: use c_1

(b) Quantile-quantile plot: use c_1

(c) Gaussian kernel density estimate: use c_2

(d) Quantile-quantile plot: use c_2

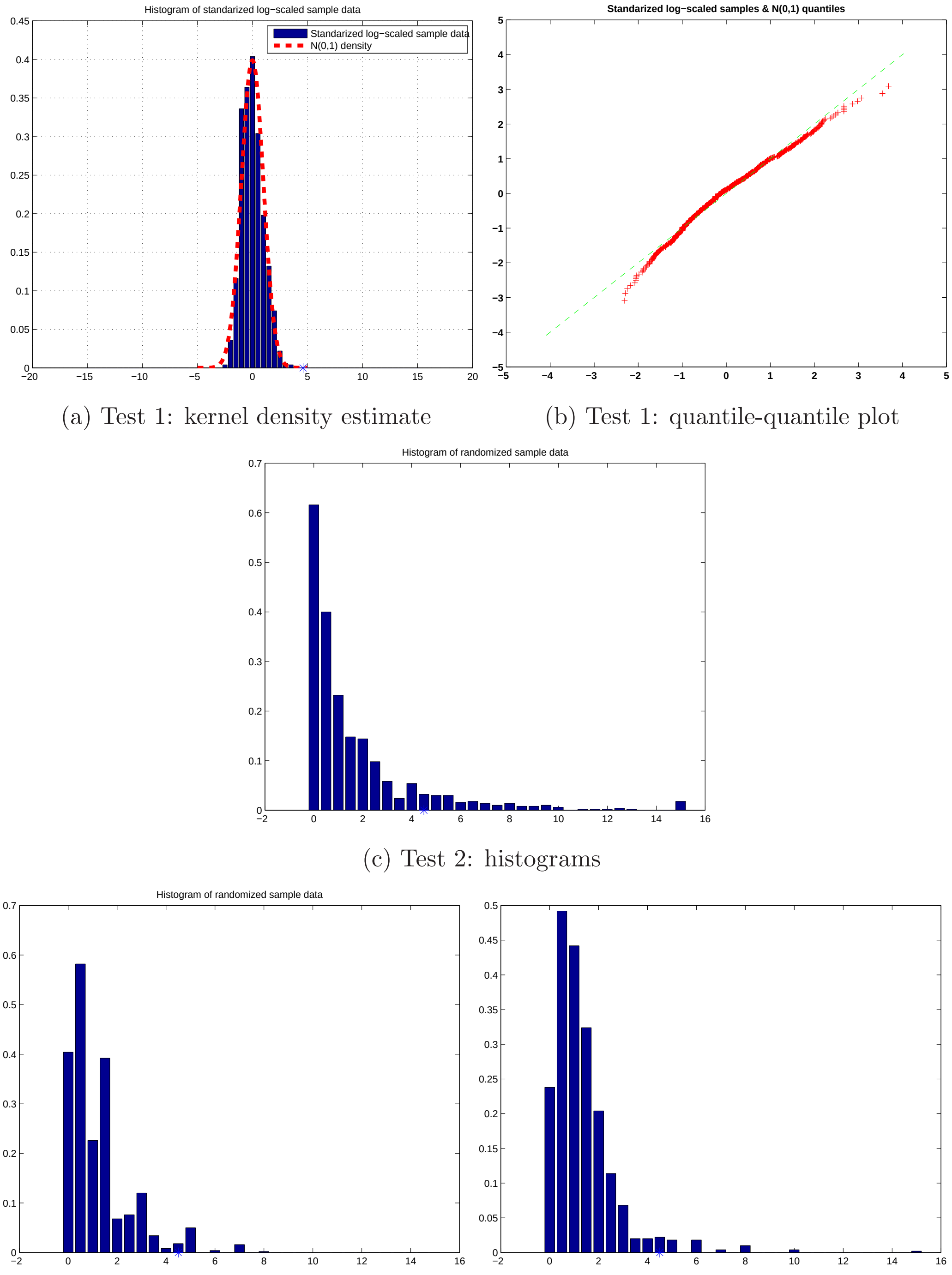
Figure 3.16: Validating *Cluster 1* by Redistribution with *Approach 2*.



(a) Gaussian kernel density estimate: use c_1 (b) Quantile-quantile plot: use c_1

(c) Gaussian kernel density estimate: use c_2 (d) Quantile-quantile plot: use c_2

Figure 3.17: Validating *Cluster 2* by Redistribution with *Approach 2*.



(a) Test 1: kernel density estimate

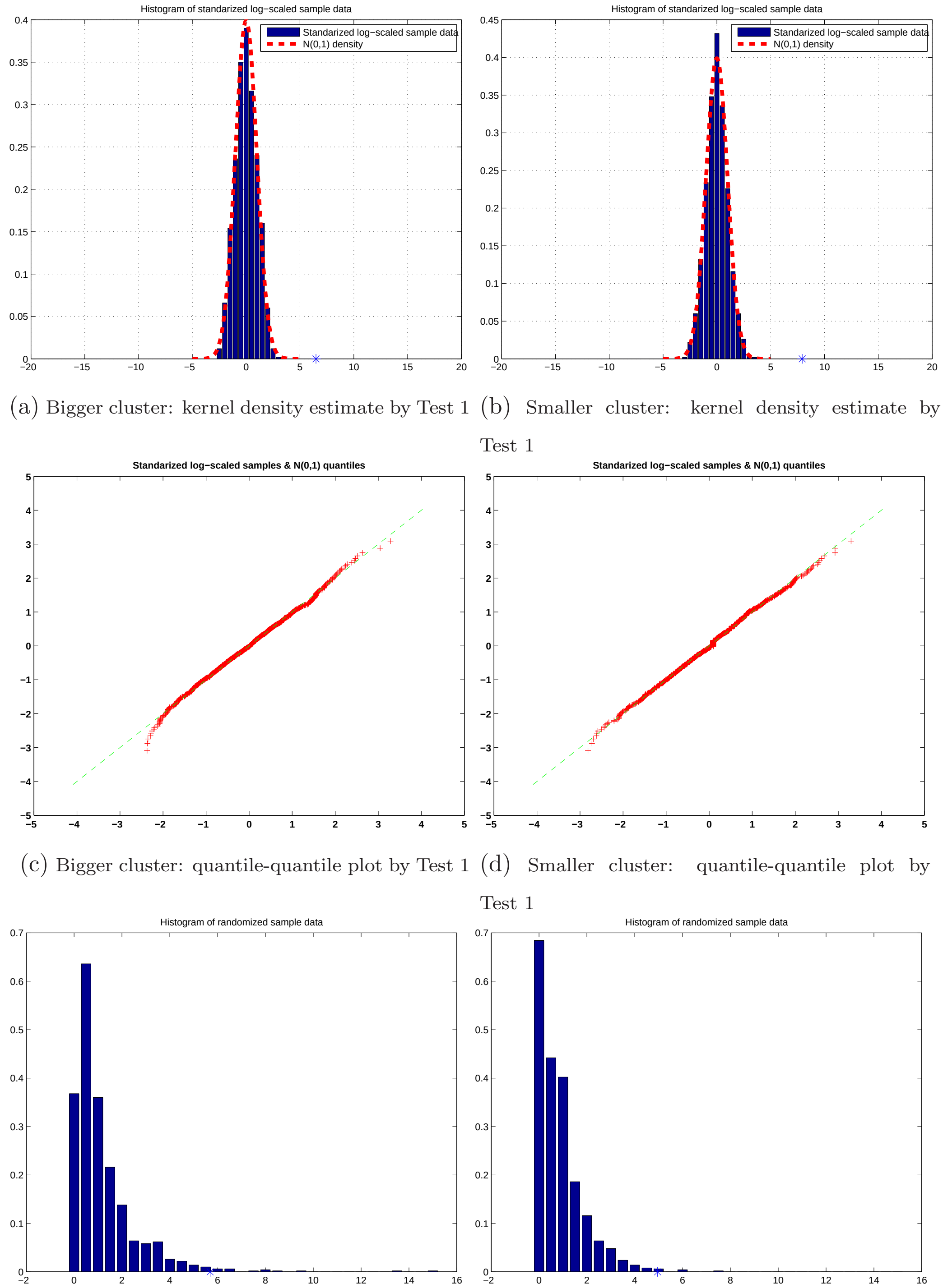
(b) Test 1: quantile-quantile plot

(c) Test 2: histograms

(d) Histograms: use c_2 and Permutation

(e) Histograms: use c_2 and Redistribution

Figure 3.18: Validating the cluster in family networks.

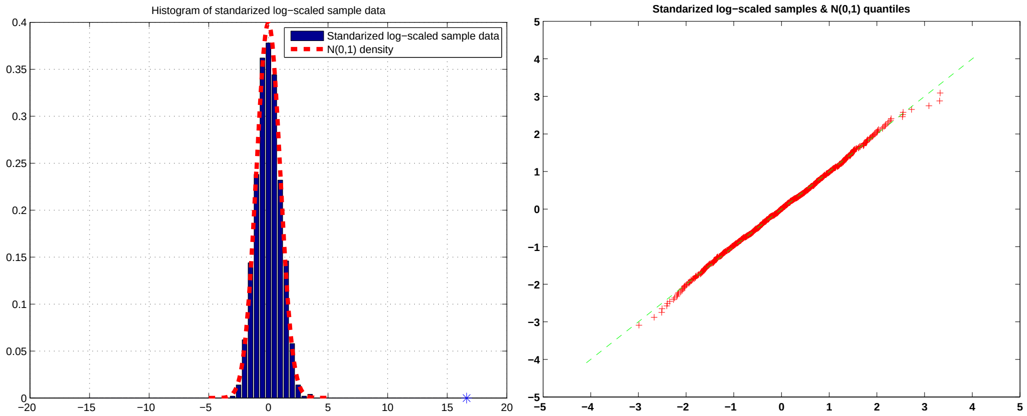


(a) Bigger cluster: kernel density estimate by Test 1 (b) Smaller cluster: kernel density estimate by Test 1

(c) Bigger cluster: quantile-quantile plot by Test 1 (d) Smaller cluster: quantile-quantile plot by Test 1

(e) Bigger cluster: histograms by Test 2 (f) Smaller cluster: histograms by Test 2

Figure 3.19: Validating the clusters in tribes networks.



(a) Kernel density estimate

(b) Quantile-quantile plot

Figure 3.20: Validating the cluster in macaque brain networks by Test 1.

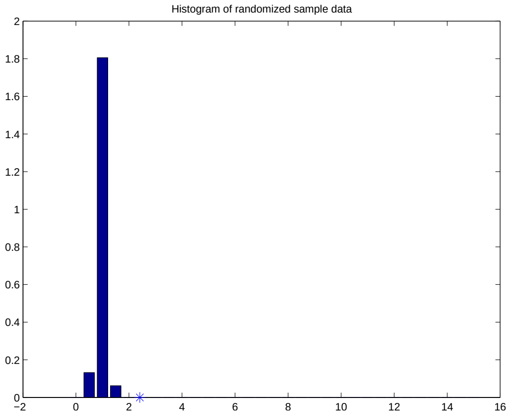


Figure 3.21: Histogram of macaque brain networks by Test 2.

Chapter 4

Power Method Viewpoint

4.1 Introduction

In Chapter 2, we introduced our reordering algorithms using an optimization viewpoint. In this chapter, we interpret the algorithms on an iterative basis. This provides an alternative justification.

4.1.1 Power Method

We recall that the power method applied to a general square matrix $W \in \mathbb{R}^{N \times N}$ takes the form [48, 131, 133]

(1) choose $y^{[0]} \in \mathbb{R}^N$, set $k = 0$,

(2) let $y^{[k+1]} = Wy^{[k]}/\| Wy^{[k]} \|$,

(3) repeat step(2) until some convergence criterion is satisfied.

In this form, given $y^{[0]} \in \mathbb{R}^N$, the power method produces a sequence of vectors $y^{[k]}$. If W has eigenvalues $|\sigma_1| > |\sigma_2| \geq \cdots \geq |\sigma_N|$, then we say that σ_1 is a unique *dominant eigenvalue* and $|\sigma_1|$ represents the maximum modulus of W's eigenvalues. The power method sequence $y^{[k]}$ converges to the corresponding eigenvector of W if σ_1 is unique and dominant.

The convergence rate of the power method is dictated by the ratio $|\sigma_2|/|\sigma_1|$. Generally, the error at the kth step is proportional to $(|\sigma_2|/|\sigma_1|)^k$. When the modulus of the dominant eigenvalue σ_1 is close to that of a subdominant eigenvalue σ_2, the power method has a slow convergence. This difficulty motivates alternatives to the power method, such as the shifted power method [106, 131, 133].

The existence of a unique dominant eigenvalue σ_1 is essential for confirming the power method converges. The behavior of this iterative method without the assumption would be different [79, 96, 131, 133]. A generalization of the power method, called orthogonal iteration or subspace iteration, is useful when $|\sigma_1| = |\sigma_2| = \cdots = |\sigma_r| > |\sigma_{r+1}| \geq \cdots \geq |\sigma_N|$, the corresponding iteration then converges to some vector lying in the subspace spanned by the eigenvectors. The corresponding rate of convergence is proportional to $|\sigma_{r+1}/\sigma_r|^k$.

Considering that the aim of this work is to interpret an algorithm for computing the GSVD as an iterative method that is attempting to reorder the nodes, we can simplify the iteration by replacing step (2) with $y^{[k+1]} = Wy^{[k]}$. This normalization does not change the relative order of the components in each $y^{[k]}$. We emphasize that our aim in this chapter is to develop a new interpretation of the algorithm rather than a practical implementation.

4.1.2 Notation and Assumptions

As in Chapter 2, we suppose that the square, symmetric, real-valued matrices A and B in $\mathbb{R}^{N\times N}$ represent two different types of interaction on the same set of N nodes. We assume that A and B are invertible with diagonal entries $a_{ii} = b_{ii} = 0$. We consider general, weighted edges and use the convention that a large weight a_{ij} or b_{ij} indicates strong connectivity between nodes i and j in

A or B, respectively.

Recalling (2.4), which introduced the definition of the GSVD in Chapter 2, a pair of square matrices A and B in $\mathbb{R}^{N\times N}$ can be expressed as $A = UCX^{-1}$ and $B = VSX^{-1}$, where $U \in \mathbb{R}^{N\times N}$ and $V \in \mathbb{R}^{N\times N}$ are orthogonal, $C \in \mathbb{R}^{N\times N}$ and $S \in \mathbb{R}^{N\times N}$ are diagonal with nonnegative entries that $0 \leq c_1 \leq c_2 \leq \cdots \leq c_N$ and $s_1 \geq s_2 \geq \cdots \geq s_N \geq 0$, and $X \in \mathbb{R}^{N\times N}$ is nonsingular. We assume that generalized singular values $\lambda_i = c_i/s_i$ satisfy

$$0 < \lambda_1 < \lambda_2 \leq \lambda_3 \leq \cdots \leq \lambda_{N-1} < \lambda_N. \tag{4.1}$$

Let X be defined by $X = [x^{[1]}, x^{[2]}, \cdots, x^{[N]}]$, so that $x^{[i]}$ represents the ith column of X. Analogously, we use $u^{[i]}$ to represent the ith column for orthogonal matrix U and $v^{[i]}$ for the other orthogonal matrix V, and $e^{[i]}$ denotes the ith column of the identity matrix I.

In Chapter 2, we derived two algorithms based on an optimization viewpoint. We use the columns from X in Algorithm 1 and the columns from X^{-T} in Algorithm 2 to shuffle the nodes. Both algorithms can be used to explore two different types of patterns between A and B. For clarity, we called these two patterns *Mode 1* and *Mode 2*:

- a group of nodes form good clusters in A that are not in B (*Mode 1*),
- another group of nodes form good clusters in B that are not in A (*Mode 2*).

In section 4.2, we will first set up the iteration for *Mode 1*, accompanied by an interpretation for the corresponding iteration to justify both algorithms from Chapter 2. We will also explain that *Mode 2* can be interpreted in the same way by a corollary to the lemma developed for *Mode 1*. The final section of the chapter summarizes the results.

4.2 Algorithm Derivation

In this section, we begin with *Mode 1* to derive the algorithm from an iteration view. The reverse case, *Mode 2*, can be explored similarly.

4.2.1 The iteration

From our previous work in Chapter 2, we have

Algorithm 1 → use final column of X (*Mode 1*),

Algorithm 2 → use final column of X^{-T} (*Mode 1*).

Recalling the form of the power method given in section 4.1.1, we have:

Lemma 4.2.1 *Under the assumptions in section 4.1.2, the iteration defined by*

- *solve*

$$B^T B z^{[k+1]} = A^T A z^{[k]}, \tag{4.2}$$

- *set* $z^{[k+1]} \leftarrow z^{[k+1]} / \| z^{[k+1]} \|$

converges to a multiple of the final column of X. Similarly, the iteration

- *solve*

$$(A^T A)^{-1} \widehat{z}^{[k+1]} = (B^T B)^{-1} \widehat{z}^{[k]}, \tag{4.3}$$

- *set* $\widehat{z}^{[k+1]} \leftarrow \widehat{z}^{[k+1]} / \| \widehat{z}^{[k+1]} \|$

converges to a multiple of the final column of X^{-T}.

Proof Write $z^{[k]} = B^{-1} y^{[k]}$, so $y^{[k+1]} = B z^{[k+1]}$. Then the iteration (4.2) becomes

$$B^T y^{[k+1]} = A^T A B^{-1} y^{[k]}.$$

So

$$\begin{aligned} y^{[k+1]} &= B^{-T}A^TAB^{-1}y^{[k]} \\ &= (AB^{-1})^T(AB^{-1})y^{[k]}. \end{aligned}$$

Recall the definition of the GSVD in (2.4). We have $(AB^{-1})^T(AB^{-1}) = VS^{-2}C^2V^T$, giving

$$y^{[k+1]} = VS^{-2}C^2V^Ty^{[k]}. \tag{4.4}$$

This is the power method on the square matrix $VS^{-2}C^2V^T \in \mathbb{R}^{N\times N}$. By construction, this matrix has eigenvalues c_i^2/s_i^2 and eigenvectors given by the columns of V. Considering the order of the elements c_i and s_i defined in (2.4) and the assumptions we made in (4.1), the ratio c_N^2/s_N^2 is dominant. Hence the iteration (4.4) will converge to the final column of V, which corresponds to the unique dominant eigenvalue given by c_N^2/s_N^2.

Then we have, as $k \to \infty$,

$$\begin{aligned} z^{[k]} &= B^{-1}y^{[k]} \\ &\to XS^{-1}V^Tv^{[N]} \\ &= XS^{-1}e^{[N]} \\ &= {s_N}^{-1}x^{[N]} \\ &\propto x^{[N]}, \end{aligned}$$

where $v^{[N]}$, $e^{[N]}$ and $x^{[N]}$ represent the final column from V, I and X, respectively, and s_N is the smallest of the diagonal entries in S. Hence $z^{[k]}$ converges to a multiple of the final column of X.

Analogously, let $\widehat{z}^{[k]} = A^T\widehat{y}^{[k]}$, so $\widehat{y}^{[k+1]} = A^{-T}\widehat{z}^{k+1}$. Then we can rewrite the iteration (4.3) as

$$A^{-1}\widehat{y}^{[k+1]} = (B^TB)^{-1}A^T\widehat{y}^k.$$

So we have

$$\begin{aligned}\widehat{y}^{[k+1]} &= AB^{-1}B^{-T}A^T\widehat{y}^{[k]} \\ &= (AB^{-1})(AB^{-1})^T\widehat{y}^{[k]}.\end{aligned}$$

Recalling the definition of the GSVD in (2.4), we have $(AB^{-1})(AB^{-1})^T = UC^2S^{-2}U^T$, giving

$$\widehat{y}^{[k+1]} = UC^2S^{-2}U^T\widehat{y}^{[k]}. \tag{4.5}$$

Iteration (4.5) is the power method on the square matrix $UC^2S^{-2}U^T \in \mathbb{R}^{N\times N}$. By construction, this matrix has eigenvalues c_i^2/s_i^2 and eigenvectors given by the columns of U. Considering the order of c_i and s_i in (2.4) and the assumptions made in (4.1), the power method will converge to the final column of U, which corresponds to the single dominant eigenvalue c_N^2/s_N^2 of matrix $UC^2S^{-2}U^T$.

Hence, as $k \to \infty$,

$$\begin{aligned}\widehat{z}^{[k]} &= A^T\widehat{y}^{[k]} \\ &\to X^{-T}CU^Tu^{[N]} \\ &= X^{-T}Ce^{[N]} \\ &= c_Nx^\star \\ &\propto x^\star,\end{aligned}$$

so $z^{[k]}$ converges to a multiple of the final column of X^{-T}. Here, $x^\star$ represents the final column vector from matrix X^{-T}, $u^{[N]}$ is the final column of U and c_N is the largest diagonal entry of S. This completes the proof.

The lemma shows that iteration (4.2) is equivalent to Algorithm 1 and iteration (4.3) is equivalent to Algorithm 2. Both iterations are described for *Mode 1* in finding the good clusters in A that are not in B.

4.2.2 Interpreting the iteration

In this section, we will use **Lemma 4.2.1** to justify Algorithm 1 and Algorithm 2. Our aim is to map the nodes onto the real line. Each node, i, will be given a real coordinate, z_i, and we will reorder the network according to their positions on the real line. More precisely, our aim is as follows: If there is a *Mode 1* cluster then we aim to give the corresponding nodes the largest (most positive or most negative) coefficients. To be definite, we will suppose that they will be given large positive coefficients. Hence these nodes should appear together at one end of the reordering and they will reveal a strongly-weighted sub matrix in A that does not exist for B.

Consider

$$\bar{B}z^{[k+1]} = \bar{A}z^{[k]}. \tag{4.6}$$

This is a general iteration form for (4.2), with $\bar{A} = A^T A$ and $\bar{B} = B^T B$ in $\mathbb{R}^{N\times N}$. For clarity, we use $\bar{A}_{ij}$ and $\bar{B}_{ij}$ to represent the weights of $\bar{A}$ and $\bar{B}$, respectively.

We will argue that the iteration (4.2), or (4.6), can be regarded as an attempt to compute coefficients for the nodes, using the data in A and B. The iterations shuffle the locations until an appropriate order is produced.

We will ignore the normalization step, as this does not affect the relative ordering of the nodes. The iteration (4.6) may then be written as

$$z_i^{[k+1]} = \underbrace{\frac{\bar{A}_{ii}}{\bar{B}_{ii}}}_{\textbf{Part 1}} \left(z_i^{[k]} + \underbrace{\sum_{l=1,l\neq i}^{N} \frac{\bar{A}_{il}}{\bar{A}_{ii}} z_l^{[k]}}_{\textbf{Part 2}} \right) \underbrace{- \sum_{l=1,l\neq i}^{N} \frac{\bar{B}_{il}}{\bar{B}_{ii}} z_l^{[k+1]}}_{\textbf{Part 3}}, \tag{4.7}$$

where

$$\left.\begin{aligned} \bar{A}_{ii} &= \sum_{j=1}^{N} a_{ji}^2, \\ \bar{B}_{ii} &= \sum_{j=1}^{N} b_{ji}^2, \end{aligned}\right\} \quad \text{generalization of degree of node } i$$

$$\begin{aligned}\bar{A}_{il} &= \sum_{j=1,i\neq l}^{N}(a_{ji}a_{jl}),\\ \bar{B}_{il} &= \sum_{j=1,i\neq l}^{N}(b_{ji}b_{jl}).\end{aligned}$$

Here, $\bar{A}_{ii}$ and $\bar{B}_{ii}$ are generalizations of the degree of ith node in graph A and B, respectively, for the case of weighted edges. $\bar{A}_{il}$ represents the connectedness node i and node l for graph A: this value is large if nodes i and l have many strongly-connected neighbours in common, so this is a measure of similarity where more positive means more similar. Analogously, $\bar{B}_{il}$ denotes the relationship between the nodes i and l for graph B.

So, we can interpret the iteration as follows: suppose that a collection of nodes already has large positive coefficients because they represent a cluster in A that is not a cluster in B. (Large negative coefficients can be discussed in a similar way.) Should node i be moved closer or further away from this group? To answer this question, we look at the right hand side of (4.7) and then see that

Part 1. $\frac{\bar{A}_{ii}}{\bar{B}_{ii}}$ represents the relative importance of node i in A compared with B.

Part 2. $\frac{\bar{A}_{il}}{\bar{A}_{ii}}$, the relative strength of connection between node i and node l in A, is used as the coefficient of $z_l^{[k]}$.

Part 3. Here, we have a minus sign, so the relative strength of connection between nodes i and l in B counts negatively. That is, the relative strength of connection between these two nodes in B contributes negatively to the coefficient of $z_l^{[k]}$.

So, in (4.7), node i will tend to move towards this group if

- $\bar{A}_{ii}/\bar{B}_{ii}$ is large, so node i is relatively important in network A, and
- $\bar{A}_{il}/\bar{A}_{ii}$ is large for nodes in this current cluster, so that node i is relatively well connected with this group in network A.

However, because of the minus sign of **Part 3** in (4.7), the node i will move away from this current cluster if

- $\bar{B}_{il}/\bar{B}_{ii}$ is large for nodes in this current cluster, so that node i is relatively well connected with this group in network B.

Overall, node i will be given a large coefficient, and hence placed alongside the other appropriate nodes, if

a) it is strongly connected to these nodes in A,

b) it is not strongly connected to those nodes in B.

Now we have justified the use of the final column of X for Algorithm 1 by analyzing the ith component of iteration (4.2).

We could study the iteration (4.3) in a similar way. Alternatively, observe that

$$\begin{aligned} A^T A x^{[N]} &= X^{-T} C^2 X^{-1} x^{[N]} \\ &= X^{-T} C^2 e^{[N]} \\ &= {c_N}^2 x^{\star} \\ &\propto x^{\star}. \end{aligned} \tag{4.8}$$

Recalling the convergence of power method for Algorithm 1 in (4.2) and that for Algorithm 2 in (4.3), we have

$$\begin{gathered} \textit{Power Method of Algorithm 2} \text{ is equivalent to} \\ A^T A \times \textit{Power Method of Algorithm 1} \end{gathered} \tag{4.9}$$

for *Mode 1*.

In other words, the vector arising from Algorithm 2 can be generated by applying Algorithm 1 and then pre-multiplying by A^TA. This extra step can be interpreted as a single iteration from an algorithm that computes the dominant singular value of A. Hence, we could argue that this extra step is using only the information in A in order to improve that aspect of the clustering. As an aside, we should note that we can also obtain the following expression

$$\begin{aligned} B^TBx^{[N]} &= X^{-T}S^2X^{-1}x^{[N]} \\ &= X^{-T}S^2e^{[N]} \\ &= {s_N}^2x^\star \\ &\propto x^\star. \end{aligned} \qquad (4.10)$$

So we also have the following formulation

$$\begin{aligned} &\textit{Power Method of Algorithm 2} \text{ is equivalent to} \\ &B^TB \times \textit{Power Method of Algorithm 1} \end{aligned} \qquad (4.11)$$

for *Mode 1*. This can be interpreted in a similar way to (4.9). Then we can also argue that this improves to the formulation of good clusters in B. Obviously, this is inconsistent with the aims of the algorithm. So, how can we interpret this incompatibility?

The iteration (4.2) can be written as

$$z^{[k+1]} = (B^TB)^{-1}(A^TA)z^{[k]},$$

and we know that $z^{[k]}$ converges to a multiple of the final column of X. So

$$z^{[k]} = \underbrace{(B^TB)^{-1}(A^TA)}_{\textit{kth iteration}} \cdots \underbrace{(B^TB)^{-1}(A^TA)}_{\textit{2nd iteration}}\underbrace{(B^TB)^{-1}(A^TA)}_{\textit{1st iteration}} z^{[0]},$$

and hence, from (4.10),

$$(B^TB)\underbrace{(B^TB)^{-1}(A^TA)\cdots(B^TB)^{-1}(A^TA)}_{\textit{k times iteration of Algorithm 1(}\text{Mode 1}\textit{)}} z^{[0]} \to x^\star.$$

Replacing $(B^TB)(B^TB)^{-1}$ with I, we see that

$$(A^TA)\underbrace{(B^TB)^{-1}(A^TA)\cdots(B^TB)^{-1}(A^TA)z^{[0]}}_{\textit{k-1 times power iteration of Algorithm 1}}$$
$$\text{is equivalent to } \textit{Power Method of Algorithm 2} \tag{4.12}$$

for *Mode 1*. This expression (4.12) is, of course, equivalent to (4.9).

Hence, we can argue that performing the multiplication of B^TB in (4.10) is consistent with performing the multiplication of A^TA in (4.8). Overall, the vector arising from Algorithm 2 can be generated by applying Algorithm 1 and then is pre-multiplying with A^TA. Referring to the above arguments, using A^TA encourages the well connected nodes to group together in A.

Now we have interpreted the iteration (4.3). This means we have also justified for Algorithm 2 that the final column of X^{-T} can be used to group the strongly connected nodes together in A.

Corollary 4.2.1 *Under the assumptions in section 4.1.2, we can justify the use of the first column of X in Algorithm 1 and the use of the first column of X^{-T} in Algorithm 2 for* Mode 2 *by using the power method.*

Proof From **Lemma 4.2.1**, the iteration $A^TAz^{[k+1]} = B^TBz^{[k]}$ will converge to a multiple of the first column of X if we set $z^{[k+1]} \leftarrow z^{[k+1]}/\| z^{[k+1]} \|$, and the iteration $(B^TB)^{-1}\widehat{z}^{[k+1]} = (A^TA)^{-1}\widehat{z}^{[k]}$ will converge to a multiple of the first column of X^{-T} by setting $\widehat{z}^{[k+1]} \leftarrow \widehat{z}^{[k+1]}/\| \widehat{z}^{[k+1]} \|$. Hence, it follows that we can use these two iterations to justify Algorithm 1 and Algorithm 2, respectively.

4.3 Summary

This work presents an alternative theoretical viewpoint for giving an intuitive understanding of the GSVD algorithm, when it is used for exploring pairs of related data sets. The related data sets are a pair of square, symmetric, real-valued matrices representing two different weighted networks on the same set of nodes. Here, a motivation for Algorithm 1 and Algorithm 2, which were derived in Chapter 2 via the variational properties of the GSVD, was given by viewing the reordering vectors as the limiting values arising from an iterative method.

Chapter 5

Protein Networks Analysis

5.1 Background

5.1.1 Protein-Protein Interaction Networks

Physically-interacting proteins and their corresponding interactions can be represented in Protein-Protein Interaction (PPI) Networks [126]. In Protein-Protein Interaction networks, each node is a protein and an edge exists between a pair of proteins if a physical interaction has been detected by some techniques [98]. In this context, interaction corresponds to the physical binding of two molecules in three dimensional space.

Protein-Protein Interactions are intrinsic to virtually every cellular process. Mapping this kind of physical connection is crucial in understanding cellular system properties. There are various methods to discover Protein-Protein Interactions [20, 98]. Generally, these methods can be classified into three main kinds: physical methods, library-based method and genetic methods [98]. Physical methods directly detect proteins which bind to another protein [98]. A variety of indirect library-based methods have been developed to screen large libraries for genes or gene-fragments whose products may interact with a protein of interest. Genetic methods are maybe the most sophisticated strategies amongst all of these investigation techniques. They can be designed to uncover indirectly Protein-Protein Interactions via genes which show interactions with other genes [98, 126], since genes encode proteins. The library methods differ from the physical, biochemical, methods, and also contrast with the genetic methods in that they are generally performed on some special organisms such as bacteria or yeasts [98].

The two-Hybrid system (Y2H) is a useful way to detect proteins which interact. In [98], the Y2H is described as a library screening method. However, in fact, it can be also regarded as a genetic method because it uses transcriptional activity as a measure of Protein-Protein Interaction [98]. Furthermore, the Y2H is also a molecular biology technique used to discover Protein-Protein Interactions and protein-DNA interactions by testing for physical interactions (binding) between two proteins or a single protein and a DNA molecule, respectively. Some research shows the ability of this screening to produce high-quality binary protein interaction maps for large-scale yeast proteins [68, 128, 137].

The term "yeast" is often taken as a synonym for Saccharomyces cerevisiae (S. cerevisiae). This yeast species S. cerevisiae is closely related to people's everyday life experience as the most widely used yeast in food industry [80]. In fact, there are many different yeast species [78]. S. cerevisiae is a species of budding yeast that reproduces asexually by budding. Other yeast species may reproduce by fission. Schizosaccharomyces pombe (S. pombe) is the most famous example of a fission yeast species. Both yeast species, S. pombe and S. cerevisiae, have been extensively studied [93] and are of importance as model organisms in molecular and modern cell biology research. The yeast two-hybrid technique investigates the interaction between fission yeast, and has also successfully been applied to S. cerevisiae yeast [68, 128, 137]. Other physical methods can also be applied to investigate the map of S. cerevisiae yeast Protein-Protein Interactions [121].

Proteins, especially in yeast, can have direct or indirect interactions with each other [20, 68, 137]. Indirect interaction refers to being a member of the same functional module but without directly binding. In particular, two proteins may be said to interact if they both from part of a large multi-protein complex, even if they are not physically overlapping. In contrast, direct interaction refers to two amino acid chains that bind to each other. Both direct and indirect interactions reflect important information about the cell.

A Protein-Protein Interaction map is usually supposed to be non-directional and thus modeled with undirected graphs [20, 38, 56, 101, 102, 125]. However, a number of related studies consider the use of directed graphs for building Protein-Protein Interaction networks [19, 43, 54, 68], especially in the case where different types of interactions are presented for the whole system [43]. So PPI networks can be described by either undirected graphs or directed graphs, or even a mixture. Direct Protein-Protein Interactions, which are usually detected by the two-hybrid method, are mainly used to generate undirected graphs [110, 121]. On the other hand, some researchers propose that PPI interactions have directionality and therefore can be modeled by directed graphs [2, 88, 124].

Network visualization is a challenging task for large-scale Protein-Protein Interaction networks. Some of the popular analysis tools (methods) are based on directed graphs [19, 43, 54], others are built on undirected graphs [88] or a mix of the two [2]. GraphCrunch is a software tool which can compare large networks, including Protein-Protein Interaction networks, using global and local properties [88]. The current version of GraphCrunch can only be applied to undirected, simple, unweighted graphs, although self-loops (an edge that connects a node to itself), or self-associations, are possible within PPI networks [19, 54, 98, 125].

Protein-Protein Interaction networks have been identified by some authors as scale-free networks due to the observation that the degree distribution by some authors fits the power-law model best [6, 30, 126]. However, other observations indicate that the behavior of the presented protein data is not pure power-law. There have been studies applying geometric random graphs, which may be better at describing Protein-Protein Interaction networks than scale-free models [63, 88, 100], towards explaining these real biological networks.

5.1.2 Genetic Interaction Networks

Exploring interactions among genes is essential to understanding how a genome specifies the properties of an organism. Such gene interactions include protein-protein physical interactions, described in section 5.1.1, in addition to gene-gene and protein-gene interactions. The latter two types of interactions can be called Genetic Interactions. Genetic Interactions can be understood as the functional relationships between genes and the corresponding protein function in the pathway [44, 102, 104, 140]. Some other work also suggests that the complete genetic network is a map of functional relationships between genes [126]. In a Genetic Interaction network, each node is a protein and the interactions between proteins can be directional or bidirectional [32].

Genetic Interactions generally report that the function of one gene depends on the presence of another one [104]. More specifically, Genetic Interactions can be specified and imagined as two proteins being connected if removing both of the corresponding genes or mutations in two genes causes a organism or cell to die [32, 34, 81, 102, 104, 120, 126, 140]. The key point is that an individual mutation from a pair of genes (who have Genetic Interactions between each other) does not affect the organism, whereas double-mutation may cause the organism to sicken or die [102].

The observed Genetic Interaction networks appear to behave like a small-world network: the shortest path length between the node pairs tends to be small and the Genetic Interaction networks tends to exhibit a densely connected local neighborhood [132].

5.1.3 Comparing Protein-Protein Interaction Networks and Genetic Interaction Networks

As we described in section 5.1.1 and 5.1.2, Protein-Protein Interactions and Genetic Interactions represent two different types of relationships between proteins, and both of the interactions are essential to biological processes. Protein-Protein Interaction networks contain the physical bindings while Genetic Interaction networks map functional connections [126]. Protein-Protein Interaction and Genetic Interaction data for different organisms are available from some online databases such as `http://www.thebiogrid.org`.

Experimentally collected protein interactions can be given direction according to whether the protein was the bait or the prey. In some literature, the researchers use negative or positive to indicate the direction of the Protein-Protein Interactions [124]. The labeling of these positive or negative Protein-Protein Interaction is completely arbitrary, therefore the edges of the related graphs appear undirected. This is a reason why the Protein-Protein Interaction networks are usually represented as undirected (symmetric) graphs [20, 38, 101, 102, 124, 125] although the edge directions exist. Protein-Protein Interaction networks allow self-interactions, which are noted as self-loops in the corresponding undirected graph model [30, 38, 124, 125]. The connections of Genetic Interactions networks also exist in different types, which can be directional or bidirectional [32]. The Genetic Interaction networks can be modeled or visualized in a similar way to the Protein-Protein Interaction networks. Thus Genetic Interaction data can also be represented by undirected graphs [104, 126] or directed graphs [94]. In summary, although Genetic and Protein-Protein Interactions are different types of protein interactions, their structures are similar. Both of them are usually drawn using undirected graphs [30]. The density of the interaction networks can be different. For example, the yeast Genetic Interaction network is much denser than its Protein-Protein Interaction network [126].

Proteins of known function and cellular location tend to cluster together; many interactions occurring between proteins have a common function and many interactions occurring between proteins are found in the same subcellular compartment [110]. Based on this common feature of Genetic Interaction networks and Protein-Protein Interaction networks, we may predict the function of the uncharacterized yeast proteins that have physical connections to the partner proteins of known function [68, 110]. More generally, the gene or corresponding protein functions can be predicted from Protein-Protein Interactions in many cases (organisms), since the newly uncovered genes encode proteins that physically interact with proteins encoded by the known genes [98].

Although the genetically interacted genes do encode proteins in the complex, and the Genetic Interaction and Protein-Protein Interaction between the corresponding gene pairs have a common part, this is limited in quantity [126]. However, the number of common neighbors between two genes in a Genetic Interaction network relates to a known Protein-Protein Interaction between the corresponding proteins [6]. So we can also predict Protein-Protein Interactions from Genetic Interaction networks by common neighbours. As we described in section 5.1.1, the genetic method is but one of the useful methods to screen (detect) or confirm the Protein-Protein Interactions [98, 121, 126].

In this work, we aim to investigate the difference, or exclusive part, existing between Protein-Protein and Genetic Interactions. There is biological interest in finding a set of proteins which are well connected in Protein-Protein Interaction networks but not in Genetic Interaction networks, or vice versa. Dense clusters in a Protein-Protein Interaction network which are sparse in a Genetic Interaction network may correspond to protein complexes, while parts which are dense in a Genetic Interaction network but sparse in a Protein-Protein Interaction network would correspond to pathways [102, 104].

In section 5.1, we have reviewed two types of interaction networks between proteins: the Protein-Protein Interaction networks and Genetic Interaction networks. In Chapter 2, algorithms were applied to some real networks. However, most of these real data sets arose from considering social networks. The protein interaction network is coarsely distinguished as one of the main types of molecular networks, which are important biological networks [30]. This motivates us to apply our algorithms to these important large scale complex biological networks. The remaining parts of this chapter are laid out as follows: section 5.2 describes how the raw protein data was preprocessed. Section 5.3 presents the numerical results from the algorithm by reordering the trimmed data, and discusses how we identify and validate the good clusters present in one graph that are not in the other from the visual observations. Some interesting candidate protein clusters are suggested in section 5.4.

5.2 Materials and Methods

Herein, the case where a Protein-Protein Interaction network and a Genetic Interaction network on the same group of proteins are available is studied. The aim is to discover differences between these two data sets in terms of clustering detected with our algorithms. In other words, the aim is to find a group of proteins which form dense clusters in a Protein-Protein Interaction network but not in a Genetic Interaction network, and vice versa. Recalling section 5.1.3, these clusters correspond to protein complexes and pathways, separately.

Our protein interaction data was originally taken from the web database http://www.thebiogrid.org of 5 different organisms: S. cerevisiae yeast (for convenience, we use yeast to refer to S. cerevisiae yeast in the remaining parts of this chapter), human, fly, mouse and worm. For each organism, there are three edge list files: one stores the corresponding physical connections, another contains the genetic (function) connections of the proteins, and the other saves both types of connections between the corresponding node pairs. There are 4388 nodes in the yeast data sets, 7892 nodes in the human data, 2486 nodes in the fly data, 329 nodes in the mouse data and 1702 nodes in the worm data. These protein data have been used and well studied by some computer scientists [88]. However, their work is focused on visualizing large scale graphs, and we are not aware of any previous attempts to explore the different patterns with these two types of protein interaction networks simultaneously.

First, all the edge list files were imported into MATLAB so that the protein data could be stored and organized as adjacency matrices. After this conversion, two adjacency matrices plus a vector are formed for each organism. One matrix contains the Protein-Protein Interaction network over a set of proteins and the other represents the corresponding Genetic Interaction network over the same set of proteins, whose names are saved in the accompanying vector. In the matrix, each node is a protein, and each edge represents the corresponding nodes which are strongly connected.

In the process of converting (importing), it was noted that the raw edge list data consisted of a mix of bidirectional edges and directional edges between the corresponding pairs of nodes (proteins). This phenomenon can be explained by the background knowledge introduced in the previous section 5.1.3. In our experiment, all the directions of the edges were ignored and both protein interaction networks were represented with undirected binary graphs. On the other hand, all the self-loops in the graphs were retained. Therefore, non-zero diagonal entries may be present in the corresponding adjacency matrices.

Then further examination of the protein data matrices resolved some technical problems. The problems arose from two main aspects: first, much inconsistency appears in the protein node names: the same protein may be denoted by a different string of characters (this is caused by the fact that these protein data were generated by different laboratories with slightly different naming conventions); second, the sizes of protein interaction networks across some organisms are very large, which is expensive for our facilities to compute. Consequently, two steps were taken to resolve these problems in these experiments.

The first step was to remove the name inconsistency throughout the proteins from the same organism. Techniques used in this step included converting all the letters in the protein names to upper case, and removing the stop, comma, hyphens and brackets in the protein names. Subsequent results reveal that the yeast protein names are more consistent than the names across other organisms.

The second step was to reduce size in the protein interaction networks. Related studies suggest that the degree distribution in both Protein-Protein Interaction networks and Genetic Interaction networks is similar to a power law degree distribution. In general, a power law degree distribution indicates that there are a large number of nodes of very low degree, whereas a few nodes have a high degree. Considering that fact, two different ways to trim the data size were then designed with a given threshold: one way removed the nodes which have lower degrees than the threshold in both the Protein-Protein Interaction network and the Genetic Interaction network; the second only kept the nodes which have higher degrees than the threshold in both networks. For simplicity and convenience, these were called *Way One* and *Way Two* in the following context. These trim methods are both simple, but perform differently. We can imagine that *Way Two* will remove more nodes than *Way One* if we apply a same threshold to the two methods in trimming a same pair of networks simultaneously, considering the existence of the nodes whose degree are higher than the given threshold in one network but lower than that in the other network, or vice versa. The trimming results suggest *Way Two* was more promising than *Way One* since *Way Two* enabled us to use a more consistent threshold, such as a median value or a mean value of the degrees of the node in both protein interaction networks, to trim all the protein data sets.

Related work suggests that the S. cerevisiae yeast data sets may be the most consistent in protein names amongst 5 organisms [88]. This viewpoint is also supported by the findings arising from the above two steps. In addition, it was found that the protein interaction networks of some organisms (except yeast) were too sparse to form a cluster of nodes. Considering that the aim of this study is to pick out the pattern differences between related protein data sets in terms of clustering, we then omitted further tests with these organisms. So yeast data was focused on for the remainder of this work.

Figure 5.1 shows the two adjacency matrices for the two different types of connections on YEAST proteins. In Figure 5.1, nz represent the number of non-zero entries in a matrix. This number is approximately twice of the number of edges in a matrix (because self-loops are only counted once). According to the previous introduction in section 5.1, each node in both networks is a yeast protein. These proteins have two kinds of relationship. Matrix A is a Protein-Protein Interaction network which represents the physical connections of the yeast proteins, and matrix B is the corresponding Genetic Interaction network containing the genetic bindings of the same group of yeast proteins. As stated above, both matrices present in Figure 5.1 are symmetric and the directions of the edges are ignored. There are many hundreds of self-loops in matrix A, but only a few noted in Genetic Interaction matrix B. We can see that neither of the networks are sparse in Figure 5.1. Matrix B is denser than matrix A, which is consistent with the observation from [126] stated in section 5.1.3.

Figure 5.2 illustrates yeast protein data matrices trimmed by *Way Two*. In

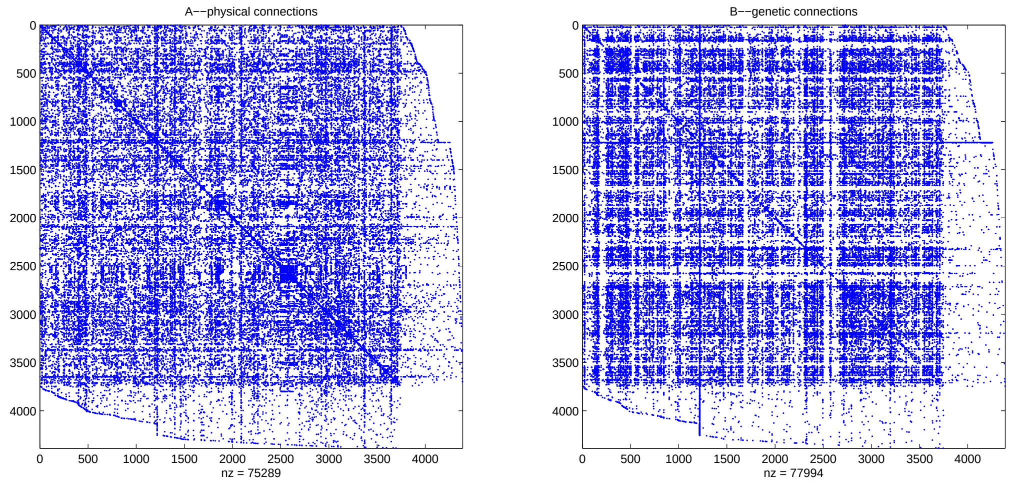


Figure 5.1: Adjacency matrices representing original yeast data.

Figure 5.2, the yeast data size has been reduced from 4388 to 458 with a given threshold (18, a mean degree of the nodes for both networks). Algorithms are then applied to this pair of trimmed protein interaction networks in the following section.

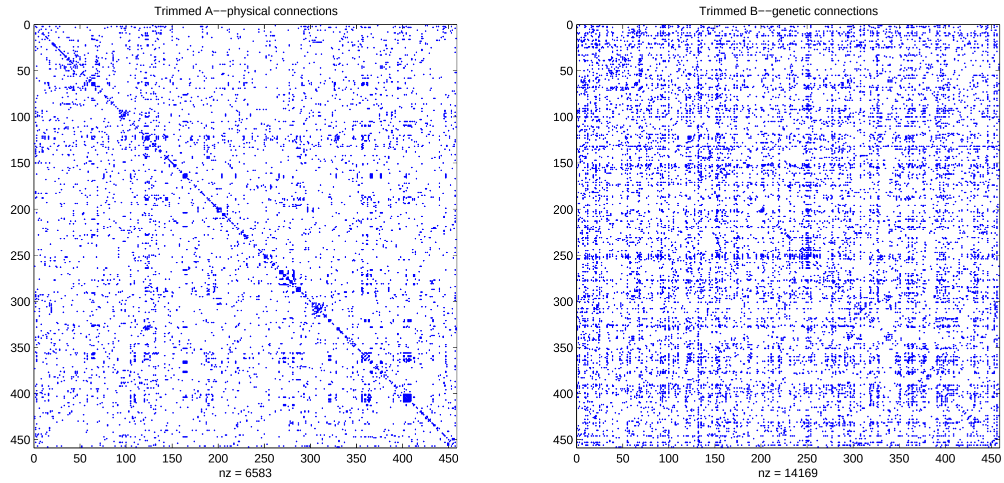


Figure 5.2: Adjacency matrices representing trimmed yeast data.

5.3 Results and Discussions

5.3.1 Reordering the Data

The trimmed yeast protein data shown in Figure 5.2 were tested with both algorithms introduced in Chapter 2. All the corresponding results suggest that Algorithm 2 is more effective than Algorithm 1 in picking out the clusters with the relative reorderings. So, in this chapter, only the results of reordering the yeast protein data with columns from X^{-T} are presented. To illustrate the performance of the algorithm, the corresponding reordered graphs are shown in Figure 5.3 and Figure 5.4. Yeast protein data reordered with the final column of X^{-T} is shown in Figure 5.3. Recalling our algorithm, Figure 5.3 supplies visual evidence identifying the exclusive clusters in graph A that are not present in graph B. Visually, in Figure 5.3, we see there appears to be one cluster in A which is not present in B (at the bottom right hand side) when both graphs reorder with the final column of X^{-T}. On the other hand, the first column from X^{-T} is used to reorder the yeast protein data in Figure 5.4. According to our algorithm, Figure 5.4 allows a visual identification of the exclusive clusters in graph B which are not present in graph A. There appears to be one cluster in B that is not in A in Figure 5.4 (at the bottom right hand side) when the data are reordered with the first column of X^{-T}. For convenience and clarity, we call the reordered graphs shown in Figure 5.3 *reordering 1* and the reordered graphs shown in Figure 5.4 *reordering 2*.

5.3.2 Producing the Cluster Name Lists

Based on the reordering results given in Figures 5.4 and 5.3 in section 5.3.1, we began by considering the final 89 nodes in *reordering 1* and the final 109 nodes of *reordering 2* as the candidate exclusive cluster nodes with respect to

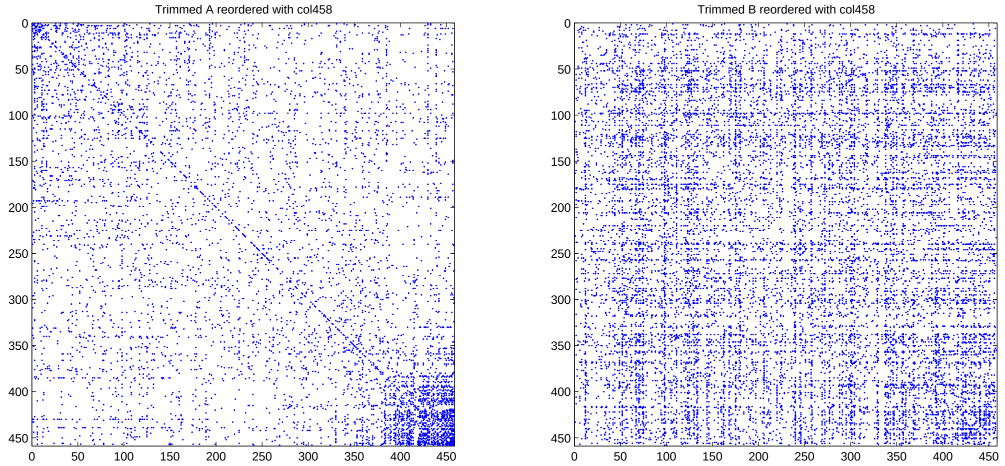


Figure 5.3: *Reordering 1*: Adjacency matrices representing yeast data reordered by the final column of X^{-T}.

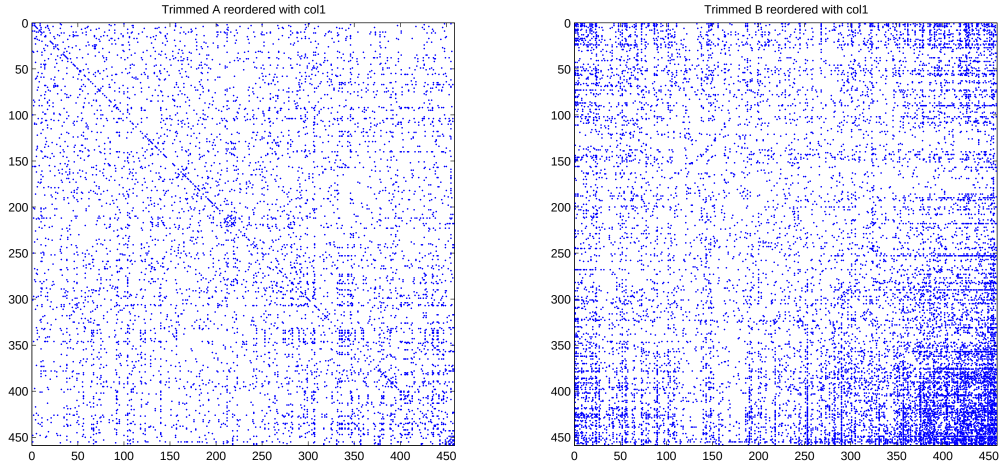


Figure 5.4: *Reordering 2*: Adjacency matrices representing yeast data reordered by the first column of X^{-T}.

graph A and B, separately. Then we checked these two protein clusters further and found 27 overlapping proteins between these two groups of nodes. Then these two groups of proteins were carefully visually scanned from one end to the other with the aim of selecting two new groups of proteins, which have smaller cluster sizes and much fewer overlaps (e.g. only 2-3 overlaps). Although it is hard to determine the size of a cluster automatically, the number of overlaps maybe a determining feature to help us to identify good clusters for both graphs. Overlaps are discussed further at a later point in this section.

Following this method, 3 pairs of candidate clusters with different sizes for Protein-Protein Interaction network A and Genetic Interaction network B of yeast proteins were found, separately. For convenience, these 3 pairs of clusters were labeled Choice 1, Choice 2 and Choice 3. These clusters (choices) are listed with the corresponding p-values in Table 5.1. The p-values are calculated using the approach introduced in Chapter 3. The small p-values ($< 5\%$) identify the significance of the clusters given in Table 5.1. As described in section 5.3.1, *reordering 1* reveals the corresponding significant clusters in network A but not in network B, and *reordering 2* represents the corresponding clusters in network B but not in network A in this table. In addition, within each choice, there are a few nodes overlapped between the identified cluster found in *reordering 1* and that found in *reordering 2*. In Table 5.1, it can be seen that there are 2 overlapping nodes between the significant cluster consisting of 33 nodes in *reordering 1* and the identified cluster consisting of 22 nodes of *reordering 2* for Choice 1. Analogously, 3 proteins are overlapped in Choice 2 between the identified cluster containing 63 proteins in *reordering 1* and that containing 22 proteins in *reordering 2*. With Choice 3, 2 nodes are found overlapped between the identified cluster consisting of 17 proteins in *reordering 1* and the significant cluster containing 31 proteins in *reordering 2*. In each choice, the overlapping nodes between the clusters in *reordering 1* and *reordering 2* are highlighted in red in Table 5.1.

Table 5.1: Identified clusters for PPI Network and GI Network of yeast proteins

Red highlights the overlapping nodes

Choice	**Reordering**	**Cluster**	p**-value**	**Cluster size**	**Protein names**
1	1	nodes 426-458	0.002	33	'SPT7'
					'TAF6'
					'UBP8'
					'NUT1'
					'HHF1'
					'CSE2'
					'RSC2'
					'SEC27'
					'SPT8'
					'CDC39'
					'SSN3'
					'TAF7'
					'MOT1'
					'CDC28'
					'GCN5'
					'SRB2'
					'RVB2'
					'STH1'
					'ISW1'
					'SIN4'
					'NGG1'
					'ADA2'
					'YAP1'
					'RSC8'
					'GAL11'
					'TAF9'
					'VPS1'
					'MED4'
					'TAF5'
					'TAF12'
					'TAF14'
					'SPT15'

Continued on Next Page

Choice	Reordering	Cluster	p-value	Cluster size	Protein names
					'GCN4'
	2	nodes 437-458	0.000	22	'LSM1' 'KEM1' 'SEC22' 'HFI1' 'ADA2' 'SSN2' 'EPL1' 'UBC4' 'CDC73' 'UBP3' 'SRB2' 'RFA2' 'HTZ1' 'CDC20' 'PHO23' 'YPT6' 'ESA1' 'YNG2' 'HSP82' 'BRE5' 'ARP4' 'SWC4'
2	1	nodes 396-458	0.000	63	'SLT2' 'CDC36' 'NHP10' 'SGF11' 'RAP1' 'RSC1' 'MED1' 'BDF1' 'MED2' 'HOG1' 'YKU80' 'PGD1'

Continued on Next Page

Choice	**Reordering**	**Cluster**	p**-value**	**Cluster size**	**Protein names**
					'SGF29'
					'TAF4'
					'TAF1'
					'NPL6'
					'MED8'
					'HTL1'
					'SRB5'
					'DBF2'
					'CCR4'
					'MRE11'
					'CDC24'
					'NOT5'
					'TAF13'
					'SPT20'
					'SPT3'
					'RSC58'
					'SGF73'
					'SSN2'
					'SPT7'
					'TAF6'
					'UBP8'
					'NUT1'
					'HHF1'
					'CSE2'
					'RSC2'
					'SEC27'
					'SPT8'
					'CDC39'
					'SSN3'
					'TAF7'
					'MOT1'
					'CDC28'
					'GCN5'
					'SRB2'
					'RVB2'

Continued on Next Page

Choice	Reordering	Cluster	p-value	Cluster size	Protein names
					'STH1'
					'ISW1'
					'SIN4'
					'NGG1'
					'ADA2'
					'YAP1'
					'RSC8'
					'GAL11'
					'TAF9'
					'VPS1'
					'MED4'
					'TAF5'
					'TAF12'
					'TAF14'
					'SPT15'
					'GCN4'
	2	nodes 437-458	0.000	22	'LSM1'
					'KEM1'
					'SEC22'
					'HFI1'
					'ADA2'
					'SSN2'
					'EPL1'
					'UBC4'
					'CDC73'
					'UBP3'
					'SRB2'
					'RFA2'
					'HTZ1'
					'CDC20'
					'PHO23'
					'YPT6'
					'ESA1'
					'YNG2'
					'HSP82'

Continued on Next Page

Choice	Reordering	Cluster	*p*-value	Cluster size	Protein names
					'BRE5'
					'ARP4'
					'SWC4'
3	1	nodes 442-458	0.005	17	'RVB2'
					'STH1'
					'ISW1'
					'SIN4'
					'NGG1'
					'ADA2'
					'YAP1'
					'RSC8'
					'GAL11'
					'TAF9'
					'VPS1'
					'MED4'
					'TAF5'
					'TAF12'
					'TAF14'
					'SPT15'
					'GCN4'
	2	nodes 428-458	0.000	31	'RPD3'
					'RRP6'
					'SRB8'
					'RSC1'
					'DIA2'
					'NPL6'
					'CTR9'
					'TAF14
					'GCN5'
					'LSM1'
					'KEM1'
					'SEC22'
					'HFI1'
					'ADA2'
					'SSN2'

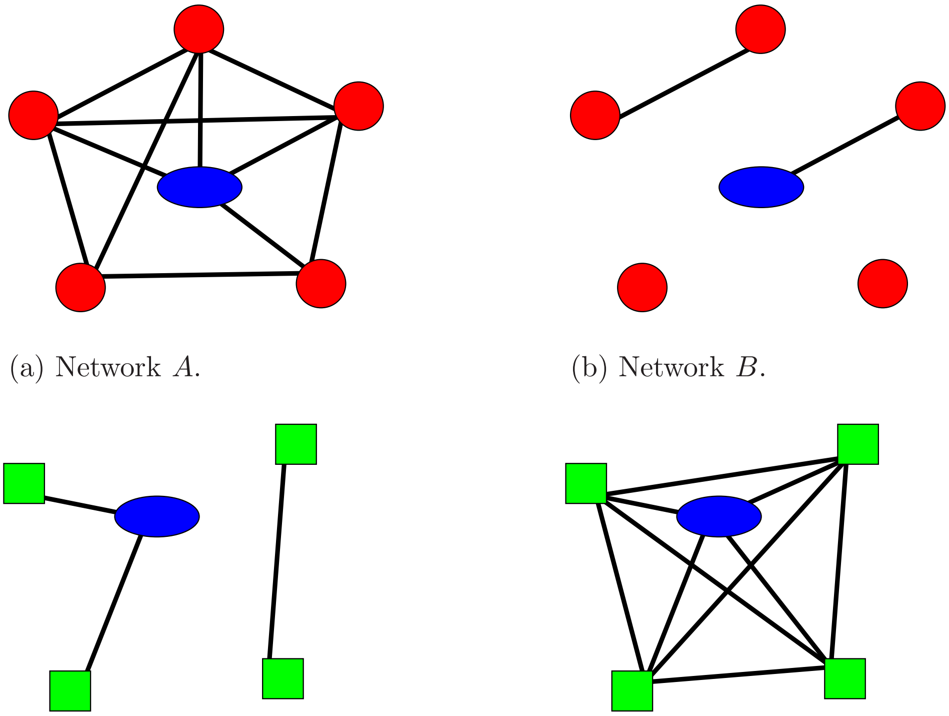
Continued on Next Page

Choice	**Reordering**	**Cluster**	***p*-value**	**Cluster size**	**Protein names**
					'EPL1'
					'UBC4'
					'CDC73'
					'UBP3'
					'SRB2'
					'RFA2'
					'HTZ1'
					'CDC20'
					'PHO23'
					'YPT6'
					'ESA1'
					'YNG2'
					'HSP82'
					'BRE5'
					'ARP4'
					'SWC4'

The appearance of overlapping nodes may seem counterintuitive. The same nodes are being proposed as members of strong clusters for both networks. However, this phenomenon can be explained by the fact that a node may be a member of a different cluster in each network. Figure 5.5 gives a visually intuitive explanation for the occurrence of overlaps. In Figure 5.5, the ellipse filled in blue represents an overlapping node well connected to a group of nodes, which are represented with circles filled in red, in graph A and also well connected to a different group of nodes, which are represented as squares filled in green, in graph B. Here, picture (a) of Figure 5.5 illustrates the subgraph consisting of nodes (a number of circles filled in red plus an ellipse filled in blue) forming a good cluster in graph A and picture (b) shows that this same group of nodes are poorly clustered in graph B. In this case, the overlapping node (ellipse filled in blue) has many connections to the remaining nodes (circles filled in red) within the cluster given in (a) but few nodes (circles filled in red) are connected to this overlapping node (ellipse filled in blue) in (b). Picture (c) shows a poor cluster consisting of the same overlapping node (ellipse filled in blue) plus a different group of nodes (a number of squares filled in green) in graph A, whereas these nodes are well connected in graph B, as shown in picture (d). There are few nodes (squares filled in green) connected to the overlapping node (ellipse filled in blue) in the poor cluster given in (c) but in picture (d) the overlapping node (ellipse filled in blue) has many connections to the remaining nodes (squares filled in green) within the same cluster. The example given in Figure 5.5 shows that the existence of an overlapping node is reasonable. However, in this case, it is emphasized that the overlapping node is well connected to different groups of nodes in *reordering 1* and *reordering 2*, separately.

5.4 Summary

This chapter shows the results of applying our algorithm to protein interaction networks, which have attracted much research interest in recent years. First, the raw protein data was preprocessed and trimmed to a computationally convenient size. The promising algorithm introduced in Chapter 2 was applied to the trimmed protein data and some candidate clusters were selected from the reordering graphs. The clusters were validated by computing the p-values. As for the overlapping nodes appearing in both *reordering 1* and *reordering 2*, the inner structure of the clusters was investigated carefully by checking the number of connections to the overlaps. After the investigation, it was concluded that



(a) Network *A*.

(b) Network *B*.

(c) Network *A*.

(d) Network *B*.

Figure 5.5: Overlapping node, represented by a yellow ellipse, in *reordering 1* and *reordering 2*. (a)(b): *reordering 1* (a cluster present in network *A* that is not present in network *B*). (c)(d): *reordering 2* (a cluster present in network *B* that is not present in network *A*.)

some clusters are reasonable with the overlaps. There is an ongoing collaboration with colleagues in bioinformatics in order to interpret the protein clusters identified by the algorithm with reference to their known biological function. It is hoped that this biologically-informed post-processing stage will

- confirm that the clusters reflect known properties, and
- suggest new relationships that could be confirmed experimentally.

Chapter 6

Metabolic Networks Analysis

6.1 Background

Schizophrenia is characterized by deficits in cognition known to be dependent upon the functional integrity of the prefrontal cortex (PFC). Furthermore, compromised PFC function in schizophrenia is supported by a multitude of neuroimaging studies reporting hypometabolism (hypofrontality), as evidenced by decreased blood flow or glucose utilization [25, 64]. While the pathophysiological basis of PFC dysfunction in schizophrenia is not completely understood, a central role for N-methyl-D-Aspartic acid (NMDA) receptor hypofunction is widely supported. For example, subchronic exposure to the NMDA receptor antagonist phencyclidine (PCP) induces cognitive deficits and a hypofrontality which directly parallels that seen in schizophrenia [23, 28, 35]. Furthermore, subchronic PCP exposure induces alterations in GABAergic cell markers and 5-HT receptor expression in the PFC similar to those seen in this disorder [23, 36, 116]. While this evidence places NMDA receptor hypofunction central to the pathophysiology of PFC dysfunction in schizophrenia, the mechanisms through which

NMDA hypofunction promotes PFC dysfunction are poorly understood.

Metabolomics, the untargeted and comprehensive quantitative analysis of small bioactive molecule levels within a tissue of interest, represents a robust approach through which alterations in diverse metabolic pathways may be determined at a biological systems level. In this way a metabolomics approach may prove useful in further elucidating the pathophysiological mechanisms contributing to PFC dysfunction in schizophrenia. Furthermore, this approach may also allow for the identification of PFC metabolic biomarkers for the cognitive deficits in this disorder. While the metabolomics approach can provide a rich and comprehensive set of data, the appropriate quantitative analysis of this data has not been adequately developed. In particular, the identification of statistical differences in metabolic pathways between experimental groups rather than the identification of statistical differences in individual metabolites alone represents a major challenge to quantitatively identifying metabolic alterations at a systems level from metabolomic data. One method through which statistical differences in metabolic pathways can be identified from metabolomic data involves the representation of this data as a large, complex network of nodes (single metabolites) connected by real-value edges (the correlation coefficient between two metabolites). This form of representation has high face validity as the relationship between two metabolites, in a given pathway, is governed by a single or series of enzymatic reactions that can be viewed as being represented by the correlation between the concentrations of the two metabolites. Another advantage is that metabolomic data consist of a range of metabolites detected in both of the experimental groups of interest, meaning that these data can be expressed as two complex networks based upon the same set of nodes. This data structure is amenable to analysis through the application of the GSVD algorithm that we have developed in this thesis.

In this chapter we therefore describe the results of some collaborative work with experimental biologists using previously unpublished data. We focus here on the algorithmic and data aspects, referring to [136] for more biologically-oriented material.

6.2 Generalizing the Algorithms

In Chapter 2, the algorithm derivation was set up on a pair of square, binary matrices A and B. Here, we generalize the algorithms from the binary case to the weighted case. Now suppose that the square, symmetric, real-valued matrices A and B in $\mathbb{R}^{N\times N}$ represent two different types of interaction between a set of N nodes.We have in mind the case where the weights play the role of correlation coefficients. Our aim is to discover clusters, in the sense of subsets of nodes that are mutually, pairwise, strongly connected through positive weights. The algorithm will also discover clusters of strong negative connectivity, although in practice this type of pattern is less likely to be present. However, we note that the arguments given below and the resulting algorithm remain valid in the case where the weights are non-negative, with zero representing the minimal level of similarity.

We use the identity (2.1) as a starting point for a computational algorithm. We may consider the identity in the same way as we did in Chapter 2. Applying the same indicator vector $x \in \mathbb{R}^N$ introduced in Chapter 2 to the identity, now we could argue that the existence of a third node, i, on two nodes, k and l, such that a_{ik} and a_{il} are both large and positive or both large and negative, is evidence in favor of placing k and l in the same group (since they have in common a strong similarity or dissimilarity with node i). On the other hand, small or oppositely signed values for a_{ik} and a_{il} is evidence in favor of placing k and l in different groups. In terms of the indicator vector, this translates to

1. $a_{ik}a_{il}$ large and positive $\Rightarrow$ try to choose $x_k x_l = +1$,

2. $a_{ik}a_{il}$ small or negative $\Rightarrow$ try to choose $x_k x_l = -1$.

Returning to the right-hand side of the identity (2.1), we see that $\sum_{k=1}^{N} x_k^2 \deg_k^A$ is independent of the choice of indicator vector, and $\sum_{i=1}^{N}\sum_{k=1}^{N}\sum_{l=1,l\neq k}^{N} a_{ik}a_{il}x_k x_l$ gives a measure of how successfully we have incorporated the (possibly conflicting) desiderata in points 1 and 2 over all pairs k, l and third parties i. So we could judge the quality of an indicator vector by its ability to produce a large value of $\|Ax\|_2^2$, provided other constraints, such as balanced group sizes, were satisfied.

Thus, we could argue that

$$\max_{x\in\mathbb{R}^N,\, x\neq 0} \frac{\|Ax\|_2^2}{\|Bx\|_2^2}. \tag{6.1}$$

is a good basis for picking out strong clusters in A that are not present in B. In this way, we have generalized Algorithm 1 in Chapter 2 to case of the weighted graphs.

Having interpreted the algorithm in the same way as described in Chapter 2, it is then straightforward to generalize the justification for Algorithm 2 to the weighted case. That is, the optimization problem

$$\max_{x\in\mathbb{R}^N,\, x\neq 0} \frac{\|B^{-1}x\|_2^2}{\|A^{-1}x\|_2^2}. \tag{6.2}$$

is a good basis for picking out strong clusters in A that are not present in B. We emphasize that here A and B are symmetric, real-valued matrices.

6.3 Cluster Validation

In Chapter 3, we designed a cluster validation approach for identifying the significance of a cluster. There are 5 steps in the cluster validation method. For the cluster quality measure $c(A, B)$ we designed c_1 (3.3) and c_2 (3.4) in Chapter 3. In Chapter 3, we defined the density $f(s)$ (3.5) of a cluster s for the binary graphs.

For the experiments in this chapter, these concepts must be extended to allow for weighted edges. In the case where the cluster is dominated by positive weights, we will generalize $f(s)$ in (3.5) to

$$f(s) = \frac{w(s)}{|s|}. \tag{6.3}$$

Here, $w(s)$ denotes the average weight in block s, and $|s|$ is the maximum possible number of edges. We note that the denominator $|s|$ cancels when ratios are computed in the cluster validation algorithm. In the case where all weights are zero or one, the general version (6.3) collapses to (3.5).

As discussed in Chapter 3, we must also decide how to randomize the networks in Step 2 in order to produce a specific algorithm for validating a cluster. Three randomization methods were tested in Chapter 3, all of which gave similar results. Of those three methods, permutation extends most naturally to the case of weighted edges, so we use that approach here.

Suppose now that we find τ nodes giving a good cluster s for B but a poor cluster for A when the graphs are reordered by column v from X^{-T}. The following general approach can be used in order to determine a p-value:

Step 1: Compute a measure of cluster quality, $c(A, B)$ (3.4), for the promising substructure consisting of those τ nodes in networks A and B reordered by column ε.

Step 2: Randomize the networks and obtain new data sets $\widehat{A}$ and $\widehat{B}$.

Step 3: Compute the measure $c(\widehat{A}, \widehat{B})$ for the τ node 'cluster' in $\widehat{A}$ and $\widehat{B}$.

p-value: After performing T loops over Steps 1 to 2, compute a p-value as the proportion of $c(\widehat{A}, \widehat{B})$ samples that exceed $c(A, B)$.

Here, the number of steps is slightly different to those in Chapter 3. In this process, we repeat Step 2 to Step 3 for T times and then for each instance of randomized networks $\widehat{A}$ and $\widehat{B}$, we get a measure $c(\widehat{A}, \widehat{B})$. After the loop, we now have a value $c(A, B)$ from our original experiment and lots of samples $c(\widehat{A}, \widehat{B})$ from randomized networks. We use the cluster quality measure given in (3.4) accompanied with the density defined in (6.3) for our weighted graphs. We randomize the metabolic data matrices with the permutation test.

In this test, we use the same null hypothesis H_0 from Chapter 3 that the cluster quality that we discovered could have arisen from the class of random networks defined by Step 2. So our goal is to test whether $c(A, B)$ is "unusually large". In this case, as introduced in Chapter 3, the p-value is simply the proportion of samples which support the null hypothesis. We therefore use the following expression to compute the p-values in the last step

$$p = \frac{|c(\widehat{A}, \widehat{B}) \geq c(A, B)|}{T},$$

where $|c(\widehat{A}, \widehat{B}) \geq c(A, B)|$ represents the number of times that $c(\widehat{A}, \widehat{B})$ is larger than or equal to the original measure $c(A, B)$. So the p-value corresponds to the proportion of randomly sampled networks for which a better clustering could be found than the clustering on the original data A and B. If the p-value is less than 0.05, the null hypothesis will be rejected and then we can say that our finding is “statistically significant at the 5% level”. Or we can express this significant finding in another way: it is very unlikely that the value $c(A, B)$ from the real data would arise if we take a random network from the class defined by Step 2.

6.4 Results and Discussions

6.4.1 Quantitative determination of metabolic pathways disrupted in the Prefrontal Cortex of PCP-treated animals

SIEVE analysis (Thermo-Fisher Scientific) [84, 87, 107, 108, 111, 119, 139] performed by biological colleagues in this collaboration revealed significant PCP-induced alterations in the level of specific metabolites in the PFC of PCP-treated rats. This included multiple metabolites from the phenylalanine, tyrosine and tryptophan metabolic pathway (defined in the Kyoto Encyclopedia of Genes and Genomes (KEGG) metabolite pathway database) and two involved in the alanine, aspartate and glutamate pathway, butanoate metabolism and in purine metabolism. This suggested that these metabolic pathways are disrupted in the PFC of PCP-treated animals. However, this simple level of analysis prevents any quantitative and statistically rigorous determination of the pre-defined (KEGG) metabolic pathways disrupted in the PFC of PCP-treated animals, which motivates the use of our GSVD algorithm.

In the context of this study the aim of applying the GSVD algorithm to metabolomic data from control and PCP-treated animals was to quantitatively determine which pre-defined KEGG pathways. The inter-metabolite Pearson's correlation coefficient (partial correlation) was used as the metric of the functional association between each pair of metabolites and was generated from the metabolite peak intensities, as determined by Liquid Chromatography Mass Spectrometry (LC-MS), across all animals within the same experimental group (i.e. either control or PCP-treated). These correlations were Fisher transformed [84, 87, 107, 108, 111, 119, 139] to give the correlation data a normal distribution. This resulted in a pair of symmetric, square, real-valued $\{98 \times 98\}$ partial correlation matrices. Each within-group matrix represents the specific association strength between each of the 9506 possible pairs of metabolites in that experimental group. A simple biological interpretation is that the correlation coefficient between two metabolites (nodes) in the matrix represents the series of enzymatic reactions responsible for converting one metabolite into another. However, it should be noted that this simple interpretation does not account for the complex relationships that may influence the correlation between two metabolites, such as the involvement of metabolites in alternative, often parallel, metabolic pathways. In essence we can view the real valued edges in the matrix as defining how many molecules of each metabolite exist relative to other metabolite. Because of the nature of the data, our network treats interactions between molecules as bidirectional, and so the set of interactions between molecules forms an undirected weighted network. In essence the GSVD algorithm allows the re-ordering of the two experimental matrices A (control animals) and B (PCP-treated animals) with the aim of discovering a new node (metabolite) ordering that reveals clusters of nodes that exhibit strong connectivity (mutual weights) in one network but not the other. In the context of this data the aim of applying the GSVD algorithm was to identify clusters of metabolites present in one of the experimental groups that were not present in the other, in the hope of identifying those metabolic pathways in the PFC disrupted by PCP-treatment. Once the matrices had been re-ordered through the GSVD algorithm the significant presence of a cluster in the given network was statistically tested by comparison of the cluster quality measure in the real networks relative to that in 1000 random permutations of the initial matrices, as described in section 6.3.

As stated in section 2.2 in Chapter 2, we display here the metabolic data using the heat map of the matrices. The heatmap is nowadays widely used in visualizing biological data sets, especially gene expression data, from a cluster analysis view [37, 134]. The original metabolic networks are shown in Figure 6.1, where matrix A represents control animals and B represents PCP-treated animals. Here, the warmer colors indicate larger weights in the network and colder colors correspond to smaller entries. Figures 6.2 and 6.3 show the networks reordered by the first and the final column of X^{-T}, respectively. Figure 6.3 provides visual evidence of clustered nodes present in one experimental group but not the other. In this figure, two discrete clusters are visually apparent (top left hand side and bottom right hand side of the heatmap) signifying those clusters present in network A (control) but not in B (PCP). For Figure 6.3 the significance of the top cluster (first 22 nodes in the re-ordering, $p < 0.001$) and the bottom cluster (last 18 nodes in the re-ordering, $p < 0.001$) was confirmed, as outlined in section 6.3. Hence there were clusters of metabolites significantly present in control (A) animals that were not present in PCP-treated (B) animals. The identity of the metabolites, the KEGG pathways in which each metabolite is involved, and the PCP-induced alteration in the overt level of each metabolite (as determined by SIEVE analysis) are shown in Tables 6.1 and 6.3 for the top and bottom cluster, respectively. In contrast to the metabolite clustering shown in Figure 6.3 there was no evidence in Figure 6.2 for any significant cluster of metabolites present in PCP-treated animals (B) that was not present in control (A) animals; for example: (i) potential top cluster [first 10 nodes] $p = 0.421$; (ii) potential middle cluster [nodes 18-25] $p = 0.494$. Rigorous significance testing, involving multiple potential metabolite clusters, confirmed that there were no significant clusters of metabolites in PCP-treated animals that were not present in controls (Figure 6.2). Following significance testing of potential metabolite clusters in the GSVD re-ordered matrices, hypergeometric probability (described in the Supplementary Material at the end of this chapter) was applied to test the significance of KEGG defined metabolite pathway over-representation in these clusters. The results for hypergeometric probability testing are shown in Tables 6.2 and 6.4.

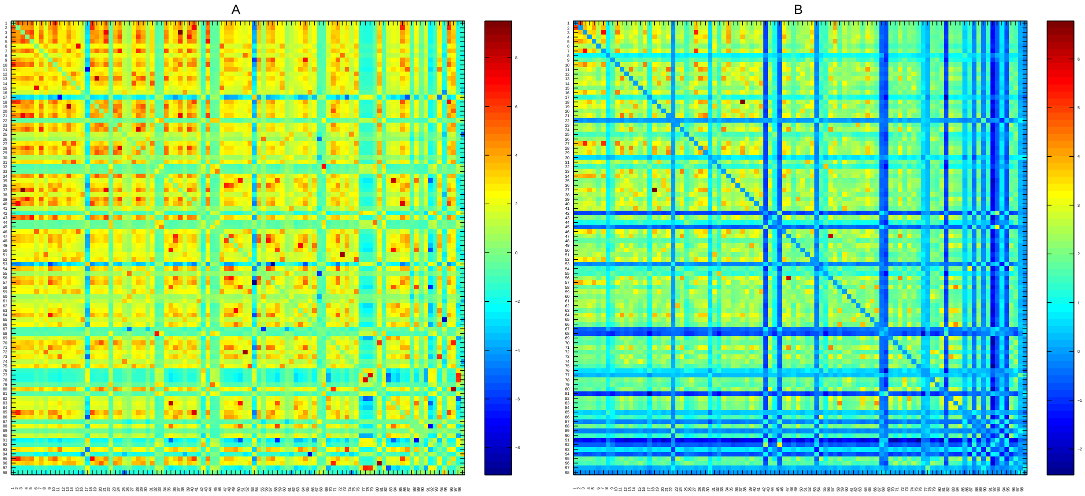


Figure 6.1: Control (A) and PCP (B) metabolic networks: original ordering.

Table 6.1 shows the top cluster of metabolites identified by the GSVD algorithm that are present in the PFC of control animals but not in PCP-treated animals (Figure 6.3). The molecular formula, tentative molecular identity and

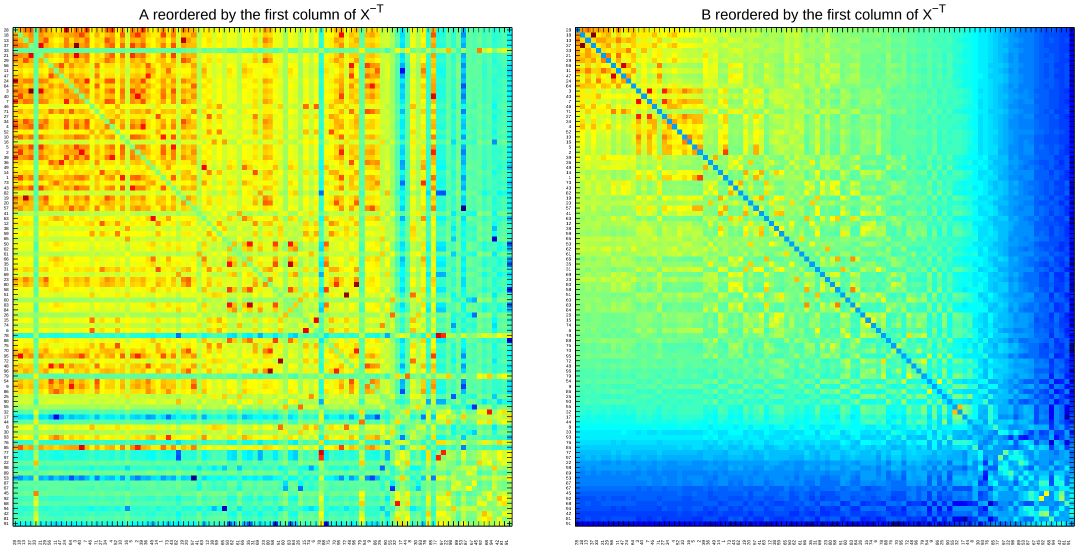


Figure 6.2: Control (A) and PCP (B) metabolic networks: reordered with the first column from X^{-T}.

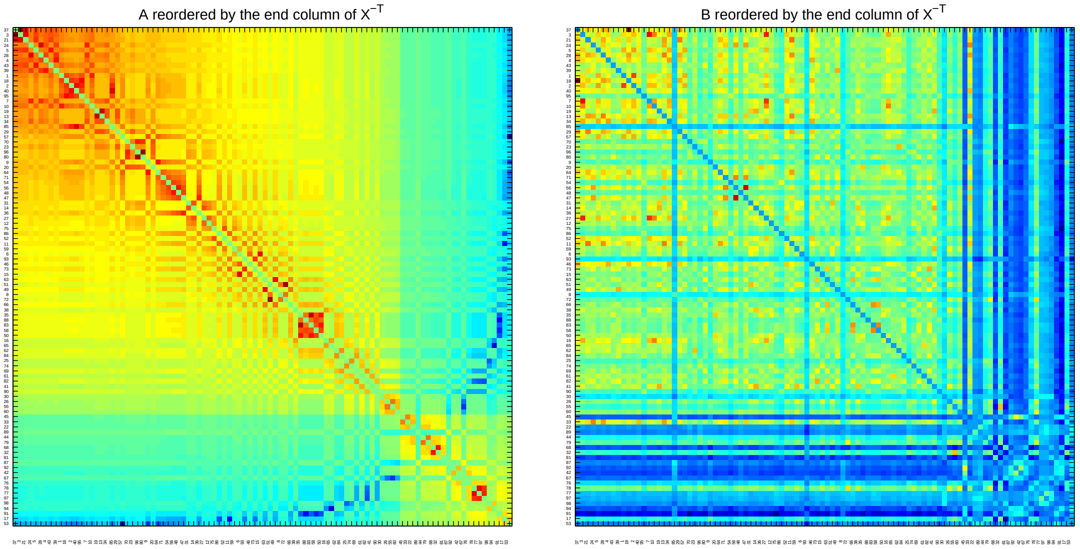


Figure 6.3: Control (A) and PCP (B) metabolic networks: reordered with the final column of X^{-T}.

the KEGG metabolic pathways in which a given metabolite is involved are shown. The p-values and ratio change reported for each metabolite in this cluster were calculated by SIEVE analysis. KEGG pathways identified in this

Table 6.1: Metabolite identities and their relevant KEGG pathways in the top cluster of Figure 6.3

Formula	**Metabolite Identity**	**KEGG Pathways**	***p*-value**	**PCP/ Control Ratio**	**Direction of Change**
$C_5H_10N_2O_3$	Glutamine	c, m, l, o	0.522	0.959	DECREASE
H_3PO_4	Phosphoric acid	w	0.254	0.915	DECREASE
$C_5H_7NO_3$	Pyrroline-4-hydroxy-2-carboxylate	c	0.781	0.981	DECREASE
$C_4H_9N_3O_2$	Creatine	c, g	0.551	0.953	DECREASE
$C_4H_9NO_2$	GABA	d, k, q	0.021	0.804	DECREASE
$C_4H_7NO_4$	L-Aspartate	c, d, g, j, n, p, q, r, s	0.319	0.916	DECREASE
$C_4H_7NO_2$	1-Aminocyclopropane-1-carboxylate	e, f, t	0.590	0.951	DECREASE
$C_5H_5N_5O$	Guanine	o	0.035	0.593	DECREASE
$C_5H_9NO_4$	Glutamate	c, e, m, q, r, x	0.845	0.985	DECREASE
C_4H_7NO	2-pyrrolidinone		0.098	0.842	DECREASE
$C_6H_6N_2O$	Nicotinamide	s	0.440	0.917	DECREASE
$C_4H_6O_2$	Butanedione	k	0.017	0.786	DECREASE
$C_6H_{12}O_4$	(R)-Pantoate	j	0.722	0.963	DECREASE
$C_{15}H_{23}N_5O_{14}P_2$	ADP-ribose	o	0.058	677.029	INCREASE
$C_3H_7NO_3$	L-Serine	e, f, g, u, v	0.316	0.856	DECREASE
$C_4H_5N_3O$	Cytosine	l	0.019	0.665	DECREASE
$C_2H_7NO_3S$	Taurine	h, q	0.936	0.995	DECREASE
$C_4H_5NO_3$	Maleamate	s	0.372	0.927	DECREASE
$C_2H_8NO_4P$	Ethanolamine phosphate	g	0.373	0.889	DECREASE
	Unknown ID		0.271	1.395	INCREASE
$C_5H_{11}NO_3$	Hydroxyvaline		0.585	0.946	DECREASE
$C_6H_{13}N_3O_3$	L-Citrulline	c,d	0.007	0.709	DECREASE

cluster included (c) arginine and proline metabolism (d) urea metabolism (e) cysteine metabolism (f) methionine metabolism (g) glycine, serine and threonine metabolism (h) taurine and hypotaurine metabolism (j) panthotheate and CoA metabolism (k) butanoate metabolism (l) pyrimidine metabolism (m) glutamate metabolism (n) alanine, aspartate and glutamate metabolism (o) purine metabolism (p) lysine metabolism (q) neuroactive ligands (r) histadine metabolism (s) nicotinamide metabolism (t) propanoate metabolism (u) sulphur metabolism, (v) sphingolipid metabolism, (w) oxidative phosphorylation and (x) glutathione metabolism. While SIEVE analysis fails to attribute significance ($p < 0.05$) to PCP-induced alterations in the overt concentration of many metabolites in this cluster, GSVD analysis reveals that the relationship between the levels of these metabolites are significantly altered by PCP-treatment ($p < 0.001$) highlighting the specific metabolic pathways that may be disrupted in the PFC of PCP-treated animals. The most prominent alterations in KEGG defined pathways in this cluster were in arginine and proline metabolism (6 metabolites (c)), neuroactive ligands (4 metabolites (q)) and glycine, serine and threonine metabolism (4 metabolites (g)).

In Table 6.2 we show the hypergeometric probability of at least the observed number of metabolites arising by chance for a given KEGG pathway in the top cluster of Figure 6.3 identified though the GSVD algorithm as being present in control animals but not in PCP-treated animals. Further computational details are given in the Supplementary Material at the end of this chapter. There was a significant over representation of metabolites of (c) arginine and proline metabolism, (g) glycine, serine and threonine metabolism, (m) glutamate metabolism (q) neuroactive ligands and (s) nicotinamide metabolism (highlighted in bold). This suggests that these metabolic pathways are disrupted in the prefrontal cortex (PFC) PCP-treated animals. Here, the cluster size is 22 metabolites from a total population of 98.

Table 6.3 concerns the bottom cluster of metabolites identified by the GSVD algorithm that are present in the prefrontal cortex (PFC) of control animals but not in PCP-treated animals. The table shows the molecular formula, tentative molecular identity and the KEGG pathways in which a given metabolite is involved. The p-values and ratio change reported for each metabolite in this cluster were calculated by SIEVE analysis. KEGG pathways identified in this cluster included (e) cysteine metabolism (f) methionine metabolism (g) glycine, serine and threonine metabolism, (h) taurine metabolism, (i) thiamine metabolism (j) panthoate and CoA biosynthesis, (k) butanoate metabolism (n) alanine, aspartate and glutamate metabolism, (o) purine metabolism, (r) histidine metabolism, (s) nicotinamide metabolism, (u) sulphur metabolism, (v) sphingolipid metabolism, (x) glutathione metabolism, (y) glycerophospholipid metabolism, (z) beta-alanine metabolism, (aa) fatty actid metabolism, (bb) glyconeolysis and glucogenesis (cc) pyruvate metabolism and (dd) pentose phosphate pathway. While SIEVE analysis fails to attribute significance ($p < 0.05$) to PCP-induced alterations in the overt concentration of many metabolites in this cluster, the PCP/Control ratio suggests that the levels of many of these metabolites are markedly altered. GSVD analysis reveals that the relationship between the levels of these metabolites in this cluster are significantly altered by PCP-treatment ($p < 0.001$) highlighting the specific metabolic pathways that may be disrupted in the PFC of PCP-treated animals. There appears to be an over-abundance of purine (4 metabolites (o)) and glycerophospholipid (2 metabolites (y)) in the bottom cluster.

Table 6.4 shows the hypergeometric probability of randomly selecting at least the observed number of metabolites of a given KEGG pathway in the bottom cluster of Figure 6.3, identified though the GSVD algorithm as being present in control animals but not in PCP-treated animals. There was no evidence for a particular over-abundance of metabolites from any given KEGG pathway in this cluster. Cluster size is 18 metabolites from a total population of 98.

6.4.2 Discussion

Through its application to metabolomic data we have demonstrated the added value that can be gained from applying the GSVD algorithm to two sets of complex, network data based upon the same set of nodes. In particular, we have demonstrated that the combined application of the GSVD algorithm with hypergeometric probability analysis provides an analytical framework by which statistical alterations in predefined metabolic pathways between experimental groups can be defined from complex metabolomic data. There is a great unmet need for this type of analytical approach in metabolomics, as well as in the other omics fields (e.g. transcriptomics), which allows the quantification of alterations at the biological systems (pathways) level rather than simply identifying significant alterations of discrete measures (i.e. single metabolites).

Through the application of this analytical approach, in collaboration with biological colleagues who are able to interpret the quantitative results, we identified statistically significant alterations in specific, pre-defined metabolic pathways (KEGG database pathways) that may contribute to PFC dysfunction in PCP-treated animals, and so in schizophrenia. This included the disruption of the (1) Arginine and Proline (2) Glycine, Serine and Threonine (3) Nicotinamide and (4) glutamate metabolic pathways as well as an imbalance in (5) neuractive ligands, as defined by the KEGG database (Table 6.2). The detection of compromised glutamate metabolism in the PFC of PCP-treated rats seems particularly pertinent given the reported alterations in extracllular glutamate availability in the PFC following repeated PCP treatment [90] and the central hypothesis of hypofunctional glutamatergic PFC neurotransmission in schizophrenia [53, 82].

Overall, our collaborators were able to draw a number of inferences concerning pathway disruption, some of which agree with previous studies and others of which appear to lead to novel insights . Further details concerning the biological interpretations can be found in [136].

Supplementary Material

Chemicals

The solvents used for the study were purchased from the following sources: Acetonitrile, methanol and chloroform (Fisher Scientific, Leicestershire, UK) and formic acid (VWR, Poole, UK). All chemicals used were of analytical reagent grade. A Direct Q-3® water purification system (Millipore, Watford, UK) was used to produce HPLC grade water which was used in all analysis. Standards for 90 common bio-molecules were also purchased which were used to characterize the ZIC-HILIC column (Sigma Aldrich, Dorset UK).

Animals

All experiments were completed using male Lister Hooded rats (Harlan-Olac, UK) housed under standard conditions (21℃, 45-65% humidity, 12-h dark/light cycle (lights on 0600h) with food and drinking water available *ad libitum*). All manipulations were carried out at least 1 week after entry into the facility and all experiments were carried out under the Animals (Scientific Procedures) Act 1986. Animals received either sub-chronic treatment with vehicle (0.9% saline, *i.p.*, $n = 5$) or $2.58mg.kg^{-1}$ PCP.HCl (*i.p.*, Sigma Aldrich, UK) once daily for five consecutive days ($n = 5$). At 72 hours after the final drug treatment dose animals were sacrificed and the brain rapidly dissected out and frozen in isopentane (-40℃) and stored at -80℃ until sectioning. Frozen brains were sectioned ($20\mu M$) in the coronal plane in a cryostat (-20℃). Tissue sections from the prefrontal cortex (PFC, Bregma $+4.70mm$ to Bregma $+3.20mm$) were collected in $4ml$ glass vials with reference to a stereotactic rat brain atlas [97] and stored at -80℃ until further preparation for LC-MS analysis.

Preparation of Polar Tissue Homogenates for LC-MS Analysis

Extraction of polar metabolites from brain tissue was carried out using the two-step extraction method described previously [135], using methanol, water and chloroform for the optimal extraction of polar metabolites. A hand held homogenizer was used to homogenize the samples once in solution. For preparation of samples for LC-MS analysis $200\mu l$ of the collected polar extract was added to $600\mu l$ of $1 : 1$ acetonitrile:water solution to produce a final solvent:sample ratio of $3 : 1$. The samples were then filtered using Acrodisc $13mm$ syringe filters with $0.2\mu m$ nylon membrane (Sigma Aldrich) before LC-MS analysis.

LC-MS Analysis of Polar Metabolites

Experiments were carried out using a Finnigan LTQ Orbitrap (Thermo Fisher, Hemel Hempstead, UK) using 30000 resolution. Analysis was carried out in positive mode over a mass range of 60-1000 m/z. The capillary temperature was set at 250℃ and in positive ionization mode the ion spray voltage was 4.5

kV, the capillary voltage 30 V and the tube lens voltage 105 V. The sheath and auxiliary gas flow rates were 45 and 15, respectively (units not specified by manufacturer). A ZIC-HILIC column ($5\mu m$, 150×4.6 mm; HiChrom, Reading, UK) was used in all analysis and a binary gradient method was developed which produced good polar metabolite separation. Solvent A was 0.1% v/v formic acid in HPLC grade water and solvent B was 0.1% v/v formic acid in acetonitrile. A flow rate of 0.3 $ml/min.$ was used and the injection volume was $10\mu l$. The gradient programme used was 80% B at 0 $min.$ to 50% B at 12 $min.$ to 20% B at 28 $min.$ to 80% B at 37 $min.$, with total run time of 45 minutes. The instrument was externally calibrated before analysis and internally calibrated using lock masses at m/z 83.06037 and m/z 195.08625. Samples were analysed sequentially and the vial tray temperature was set at a constant temperature of 4℃.

Data preparation and analysis

Determination of overt alterations in metabolite levels between experimental groups

The software programme Xcalibur (version 2.0) was used to acquire the LC-MS data. The raw Xcalibur data files from version 1.2 (Thermo Fisher, Hemel Hempstead, UK). SIEVE software (Thermo-Fisher Scientific) was used to identify all metabolites affected by drug treatment by calculating a p-value and ratio based on the difference in average intensities of individual peaks, which correspond to different metabolites, between wild-type and KO sample groups. A significant difference in the level of each metabolite between groups was set at p-value< 0.05 and/or ratio less than 0.5 for downregulated metabolites and greater than 2 for upregulated metabolites. The ratio is the fold change in average peak intensities from control and treatment groups. For metabolite identification the masses of the polar metabolites were compared to the exact masses of 6000 biomolecules using an inhouse developed macro (Excel, Microsoft 2007).

Hypergeometric probability testing

The hypergeometric probability test was used to calculate the probability of finding at least the observed number of metabolites of a given pre-defined metabolic pathway (as defined on the KEGG pathway database) in the clusters identified through the GSVD algorithm, with knowledge of the total number of metabolites present in that pathway detected by LC-MS in these samples. The hypergeometric probability test was used to identify whether any of the KEGG defined metabolic pathways were significantly over-represented in any of the GSVD identified clusters. In its general form hypergeometric probability allows the calculation of the probability of observing at least (k) metabolites from a given defined KEGG pathway in a defined cluster of metabolites (n) given the total number of metabolites (N) and the total number of metabolites from the pathway in question (m). The probability mass function of hypergeometric distribution is:

$$f(k; N, m, n) = P(X = k) = \frac{\begin{pmatrix} m \\ k \end{pmatrix} \begin{pmatrix} N-m \\ n-k \end{pmatrix}}{\begin{pmatrix} N \\ n \end{pmatrix}}. \tag{6.4}$$

So here the probability is calculated using the formula

$$P(X \geq k) = \sum_{i=k}^{m} \frac{\begin{pmatrix} m \\ i \end{pmatrix} \begin{pmatrix} N-m \\ n-i \end{pmatrix}}{\begin{pmatrix} N \\ n \end{pmatrix}}. \tag{6.5}$$

Significant over-representation of a given functional group in any GSVD defined significant cluster was set by a hypergeometric probability threshold of 0.05.

Table 6.2: Hypergeometric probability of KEGG defined metabolic pathways in the top cluster in Figure 6.3

KEGG Pathway		**Number of pathway metabolites in cluster(A)**	**Total number of pathway metabolites detected (B)**	**Hypergeometric Probability ($P(X) \geq k$)**
(a)	Phenylalanine, Tyrosine and Tryptophan biosynthesis	0	4	1.000
(b)	Thiamine metabolism	0	2	1.000
(c)	**Arginine and Proline metabolism**	**6**	**8**	**0.001**
(d)	Urea metabolism	3	7	0.186
(e)	Cysteine metabolism	3	5	0.073
(f)	Methionine metabolism	2	5	0.312
(g)	**Glycine, Serine and Threonine metabolism**	**4**	**7**	**0.043**
(h)	Taurine and Hypotaurine metabolism	1	3	0.538
(i)	Thiamine metabolism	0	2	1.000
(j)	Panthothenate and CoA biosynthesis	2	5	0.312
(k)	Butanoate metabolism	2	4	0.217
(l)	Pyrimidine metabolism	2	6	0.406
(m)	**Glutamate metabolism**	**2**	**2**	**0.049**
(n)	Alanine, Aspartate and Glutamate metabolism	1	6	0.792
(o)	Purine metabolism	3	13	0.598
(p)	Lysine metabolism	1	4	0.645
(q)	**Neuroactive ligands**	**4**	**7**	**0.043**
(r)	Histidine metabolism	2	5	0.312
(s)	**Nicotinamide metabolism**	**3**	**4**	**0.034**
(t)	Propanoate metabolism	1	1	0.224
(u)	Sulfur metabolism	1	3	0.538
(v)	Sphingolipid metabolism	1	3	0.538
(w)	Oxidative phosphorylation	1	1	0.224
(x)	Glutathione metabolism	1	4	0.645

Table 6.3: Metabolite identities and their relevant KEGG pathways in the bottom cluster of Figure 6.3

Formula	Metabolite Identity	KEGG Pathways	p-value	PCP/ Control Ratio	Direction of Change
$C_5H_4N_4O_2$	Xanthine	o	0.339	0.508	DECREASED
$C_{10}H_{16}N_2O_7$	Gamma Glutamylglutamic acid		0.143	0.54	DECREASED
$C_{14}H_{26}O_2$	Myristoleic acid		0.689	0.623	DECREASED
$C_5H_4N_4O$	Hypoxanthine	o	0.115	0.569	DECREASED
$C_{17}H_{37}NO_2$	Heptadecasphinganine	v	0.733	0.769	DECREASED
$C_{10}H_{13}N_4O_8P$	IMP	o	0.461	0.73	DECREASED
$C_{10}H_{17}N_3O_6$	Unknown ID		0.775	1.183	INCREASED
$C_6H_{15}NO_3$	Triethanolamine	y	0.691	1.207	INCREASED
$C_9H_{14}N_4O_3$	Carnosine	n, r, z	0.872	1.128	INCREASED
$C_{10}H_{12}N_4O_5$	Inosine	o	0.090	0.6	DECREASED
$C_{15}H_{12}O_5$	narigenin		0.196	0.862	DECREASED
$C_{10}H_{17}N_3O_6$	gamma-Glutamylglutamine		0.007	0.673	DECREASED
$C_{26}H_{42}N_7O_{20}P_3S$	2-Hydroxyglutaryl-CoA	k	0.179	0.715	DECREASED
$C_{31}H_{54}N_7O_{17}P_3S$	Decanoyl-CoA	aa	0.410	1.312	INCREASED
$C_{25}H_{44}NO_7P$	2-Aminoethylphosphocholate	z	0.243	0.662	DECREASED
$C_{22}H_{26}O_6$	Eudesmin		0.084	0.493	DECREASED
$C_3H_7NO_2S$	L-Cysteine	e, f, g, h, i, j, u, x	0.012	0.445	DECREASED
$C_3H_7O_6P)$	Glycerone phosphate	s, y, bb, cc, dd	0.063	0.381	DECREASED

Table 6.4: Hypergeometric probability of metabolic pathways in bottom cluster in Figure 6.3

KEGG Pathway		**Number of pathway metabolites in cluster(A)**	**Total number of pathway metabolites detected (B)**	**Hyper-geometric Probability ($P(X) \geq k$)**
(e)	Cysteine Metabolism	1	5	0.646
(f)	Methionine metabolism	1	5	0.646
(g)	Glycine, serine and threonine metabolism	1	7	0.770
(h)	Taurine and hypotaurine metabolism	1	3	0.460
(i)	Thiamine metabolism	1	2	0.335
(j)	Panthothenate and CoA biosynthesis	1	5	0.646
(k)	Butanoate metabolism	1	4	0.562
(n)	Alanine, aspartate and glutamate metabolism	1	6	0.715
(o)	Purine metabolism	4	13	0.191
(r)	Histidine metabolism	1	5	0.646
(s)	Nicotinamide metabolism	1	4	0.562
(u)	Sulphur metabolism	1	3	0.460
(v)	Sphingolipid metabolism	1	3	0.460
(x)	Glutathione metabolism	1	4	0.562
(y)	Glycerophospholipid metabolism	2	8	0.452
(z)	Beta-alanine metabolism	1	3	0.460
(aa)	Fatty acid metabolism	1	1	0.184
(bb)	Glycolysis and Gluconeogenesis	1	2	0.335
(cc)	Pyruvate metabolism	1	2	0.335
(dd)	Pentose phosphate metabolism	1	2	0.335

Chapter 7

Brain Networks Analysis

7.1 Background

7.1.1 Background in Brain Networks

The brain has a complex structure (anatomical) and functional organization that is yet to be fully elucidated. *In vivo* brain imaging techniques allow us to gain further insight into the structural and functional organization of the brain. Recently, it has been shown that structural and functional brain networks, as detected using *in vivo* brain imaging, display the features of complex networks that allow the brain to be modeled as networks or graphs [18, 24, 49, 70, 72, 114, 138, 141]. Properties of these networks can then be quantitatively defined through the application of graph theory. Many studies have characterized the properties of either functional [1, 9] or structural [52] brain networks independently, using functional magnetic resonance imaging (fMRI) or diffusion tensor imaging (dMRI), respectively. More recently, studies have been dedicated to further elucidating the relationship between structural and functional brain networks [18]. These studies have shown that both structural and functional brain networks have a small-world organization, that is to say they display a high clustering coefficient and short path length in comparison to networks with a random organization [1, 9, 10, 12, 18, 49, 67, 69, 71]. This small-world organization is likely to be optimal for efficient information transfer throughout the brain [9, 10, 11, 12].

In most studies brain networks are generated as weighted matrices of pairwise associations between brain regions (nodes). A threshold is then applied to these matrices resulting in the generation of a binary adjacency matrix (an undirected graph). Properties of brain networks are then characterized through the properties of these undirected graphs. However, this binarization procedure is not always applied and the properties of brain networks have also been investigated using real-value weighted [24] or directional graphs [70].

Structural brain networks describe the anatomical connectivity of the brain and can be represented as graphs comprised of nodes, representing different brain regions, with edges describing a physical relationship (anatomical connectivity) between these regions.

Functional brain networks describe another type of connectivity within the brain where the nodes of the graph represent anatomically defined brain regions and the edges represent the functional connectivity between those brain regions. Here functional connectivity quantifies how the activity in one brain region affects that of others within the network [1, 11, 12, 18, 31, 49, 114]. For example, this may be measured via correlations in electrical activity over time. While structural brain networks can help us to understand the fundamental architecture of the brain, functional brain networks reveal how this architecture supports neurophysiological dynamics in the brain. While the structural organization of brain networks is not altered by specific stimuli, the organization of functional brain networks is. Therefore, the coordinated study of the structural-functional relationship in complex brain networks is also an important theme [18, 66, 114, 138, 141].

In this dissertation we aim to characterize the properties of brain networks gained through different imaging methods. We have already considered the application of our algorithm to structural brain imaging data, from the macaque monkey (Chapter 2). In this chapter, we consider the usefulness of applying these algorithms to functional brain imaging data from the rat. Other important differences exist between the brain networks in these two studies. For example, the macaque brain networks, produced from anatomical tract tracing, are binary and not completely bidirectional. In contrast, the brain networks investigated in this chapter are real-valued weighted and undirected. As we stated above, a binary adjacency matrix can be produced from a weighted matrix by applying a threshold. However, the choice of threshold used to generate the binary matrix is crucial: different thresholds will generate graphs with different sparsity or connection density [18]. An uninformative threshold leads to a loss of information and the corresponding binary matrices produced are often heavily biased. For this reason, we chose to apply the algorithm to the weighted networks directly. In Chapter 4, we set up the algorithms to process a pair of real-valued weighted networks simultaneously. This enables us to explore the weighted, undirected brain networks generated from this brain imaging data. The work described in this chapter has evolved out of a collaboration with experimental neuroscientists, using previously unpublished data. We refer to [29] for further experimental details.

7.1.2 Motivation in NeuroScience

As we stated in section 6.1 in Chapter 6, NMDA receptor hypofunction contributes to pathophysiology of schizophrenia. Phencyclidine (PCP) is an NMDA receptor antagonist and treating rats with this drug provides a translational model for schizophrenia [23, 99]. Similar to Chapter 6, here we investigate the effects of PCP on functional brain networks in order to further understand the role of NMDA receptor hypofunction in contributing to altered brain functioning in schizophrenia. However, this chapter deals with a different type of data set involving direct correlations between brain regions.

In the work of Dawson et al. (2010) [27], brain network properties were calculated on the binary undirected graphs from PCP-treated and Control animals, separately. However, in this work, we are studying a pair of real-valued weighted brain networks simultaneously. We aim to find nodes that are strongly connected in one animal group but not the other. In other words, we want to find some brain regions or some functional groups which are significantly clustered in Control but not in PCP-treated animals, or vice versa. We also hope to test how alterations in hub brain regions in PCP-treated animals relate to the alterations we see in functional clustering.

In this work, we are anticipating that we can identify nodes (brain regions) that form a good cluster in the brain network of Control animals but not in PCP-treated animals, or vice versa. In Chapter 4, we derived algorithms to explore a pair of weighted networks simultaneously: we can find a group of nodes strongly connected in one graph that are not in the other. Applying these algorithms to functional brain imaging data from the translational PCP model has the potential to provide insight into the true nature of disrupted brain functioning in these animals. Furthermore, if we can also identify some specific functional groups that are significantly over-represented within the given clusters, we can gain further scientific insight into the specific functional subsystem disrupted in the brains of PCP-treated animals.

Section 7.2 briefly introduces what is our data. Section 7.3 presents the reordering results of the algorithm, showing the effect of subchronic PCP treatment on functional brain networks and discussing the results based on the identified significant clusters and hub regions. Finally, we give a summary of the work in this chapter in section 7.4. In addition, supplementary material is also provided at the end of the chapter, indicating how the brain data were collected and how the weighted brain data matrices were generated.

7.2 Material and Methods

In this work, we have two functional interaction data sets available: one dataset is obtained from control animals, the other is derived from a translational model relevant to schizophrenia, rats treated subchronically with phencyclidine (PCP). The materials and methods used to generate the object data are introduced in [26, 28, 29, 74]. Further information is provided at the end of this chapter. In summary, we have a pair of symmetric, real-valued weighted matrices A (Control) and B (PCP) in $\mathbb{R}^{64\times 64}$, representing two functional brain networks under the different experimental conditions (Control and PCP) over the same set of nodes (brain regions). The value of each entry a_{ij} or b_{ij} in these two correlation matrices can be regarded as the functional connection strength between the cooresponding nodes within each group. Here, the 64 brain regions can be associated with 10 different functional subsystems: Thalamus, Hippocampus, Frontal cortex, medial Prefrontal cortex, Cortex, Mesolimbic, Amygdala, Septum/DB, Basal Ganglia and Multimodal [27, 29] based on established anatomical connectivity or biological function. The 10 functional groups are shown in Table 7.1.

In section 7.1.1, we mentioned that brain networks are generally binarized. For example, in the work of Dawson et al. (2010) [27], they take the approach of converting the weighted correlation data to binary adjacency matrices by applying a range of thresholds. An element of the binary matrix can be understood to be denoting whether the two corresponding nodes have a functional connection, or not. The entry in the binary matrix is zero if the corresponding weight (correlation) a_{ij} or b_{ij} is lower than the defined threshold and unity if the coefficient was greater or equal to the defined threshold. The network properties are then defined by computing popular global or regional (local) statistical measures of these binary adjacency matrices [27]. In addition, hub brain regions are identified. In the network, a "hub" is a brain region with many connections, so it is important in governing the activity in the network. A illustration of a hub region group is shown in Figure 7.6. A brain region is considered to be an important hub region in the network when it has a high centrality relative to the average in the brain network (either the Control or PCP-treated brain network). Further details of the hub brain regions can be seen in [29]. The hub status of brain regions altered in PCP-treated animals was also investigated. 12 brain regions were identified as hub regions in Control that were lost in PCP-treated animals. On the other hand, 11 hub regions were gained in PCP-treated animals that were not important in Control, indicating an abnormal brain network organization in these animals. LC, dRT, vRT, VLthal, CLthal, MDthal, Re, RSC, CA2, NaCc, iHab and CA3 are the 12 hub regions in the Control network. FRA, PrL, LO, DLST, BST, AVthal, Rh, mHab, Piri, Sub and IP are the 11 hub regions in the PCP network. Recall that the full node name list of the 64 discrete brain regions was shown in Table 7.1.

7.3 Results and Discussions

7.3.1 Reordering the Data

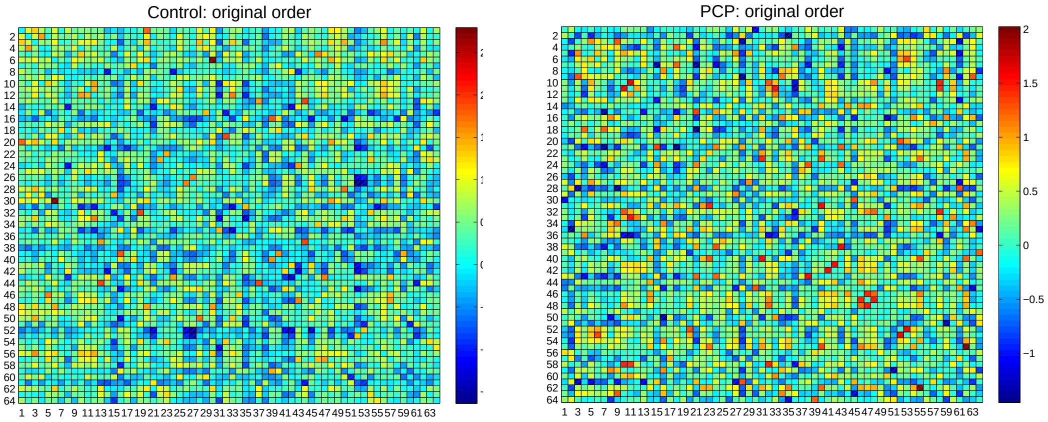


Figure 7.1: Original Control (A) and PCP (B) brain networks.

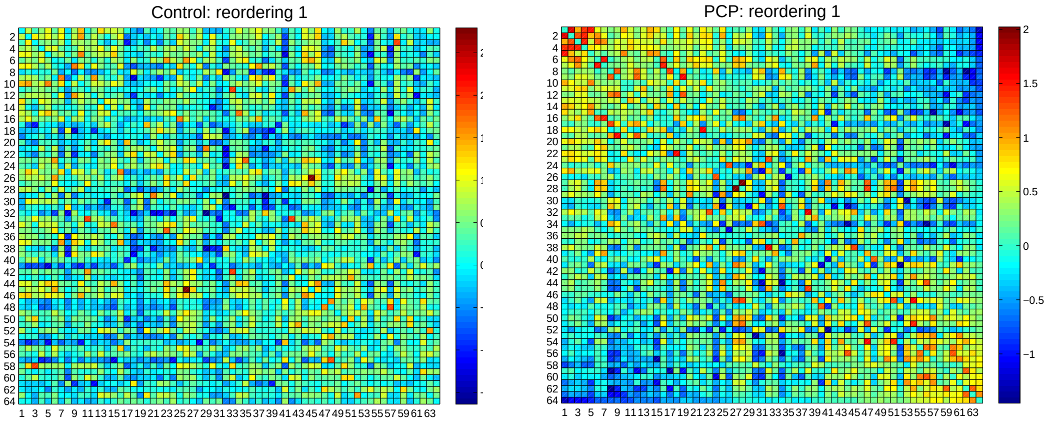


Figure 7.2: Reordering 1: Control (A) and PCP (B) brain networks reordered with the first column of X^{-T}.

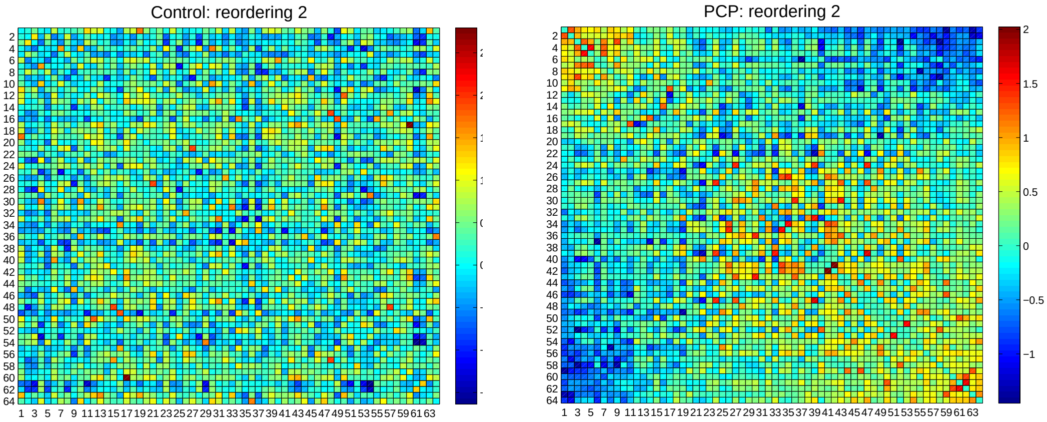


Figure 7.3: Reordering 2: Control (A) and PCP (B) brain networks reordered with the second column of X^{-T}.

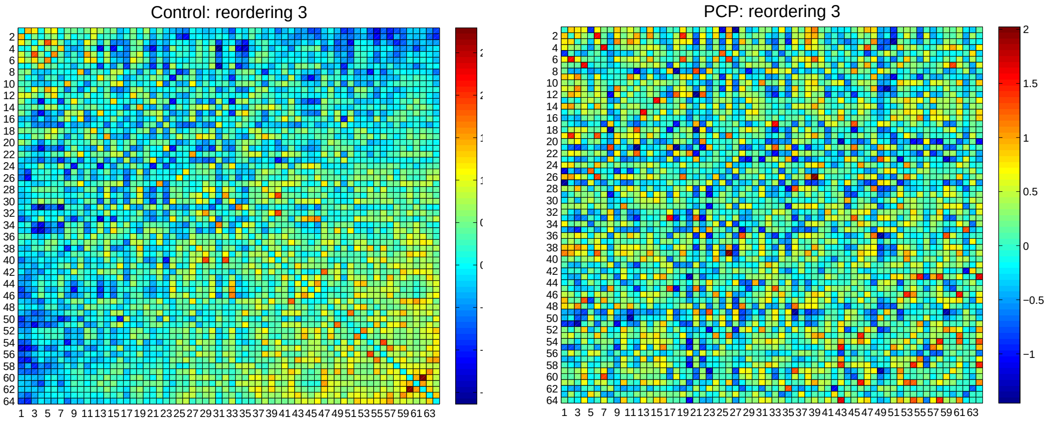


Figure 7.4: Reordering 3: Control (A) and PCP (B) brain networks reordered with the penultimate column of X^{-T}.

Figures 7.1 to 7.5 are heatmaps of the original and reordered brain data matrices. These allow the visual identification of the clustered nodes that are present in one experimental group but not the other. Figure 7.1 shows the original brain data matrices ordered alphabetically. Figure 7.2 shows the data reordered with the first column from X^{-T}, which we call reordering 1. Figure 7.3 shows the data reordered with the second column from X^{-T}, which we call

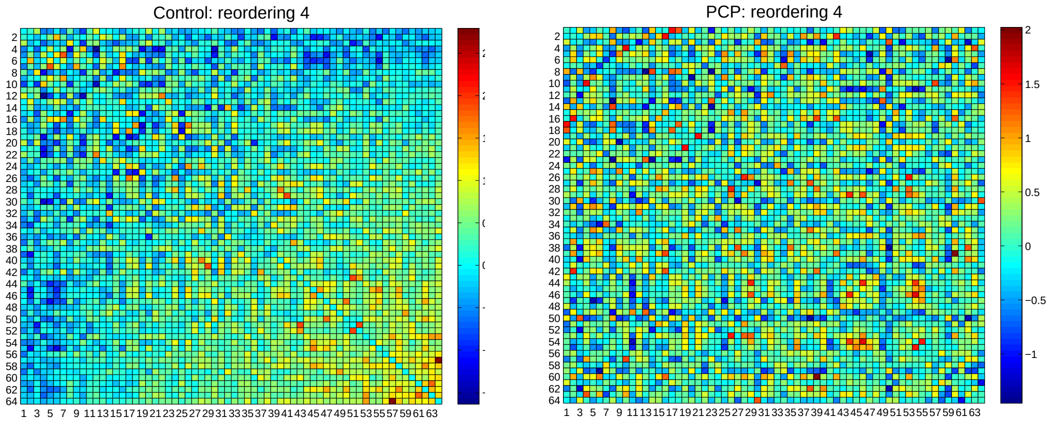


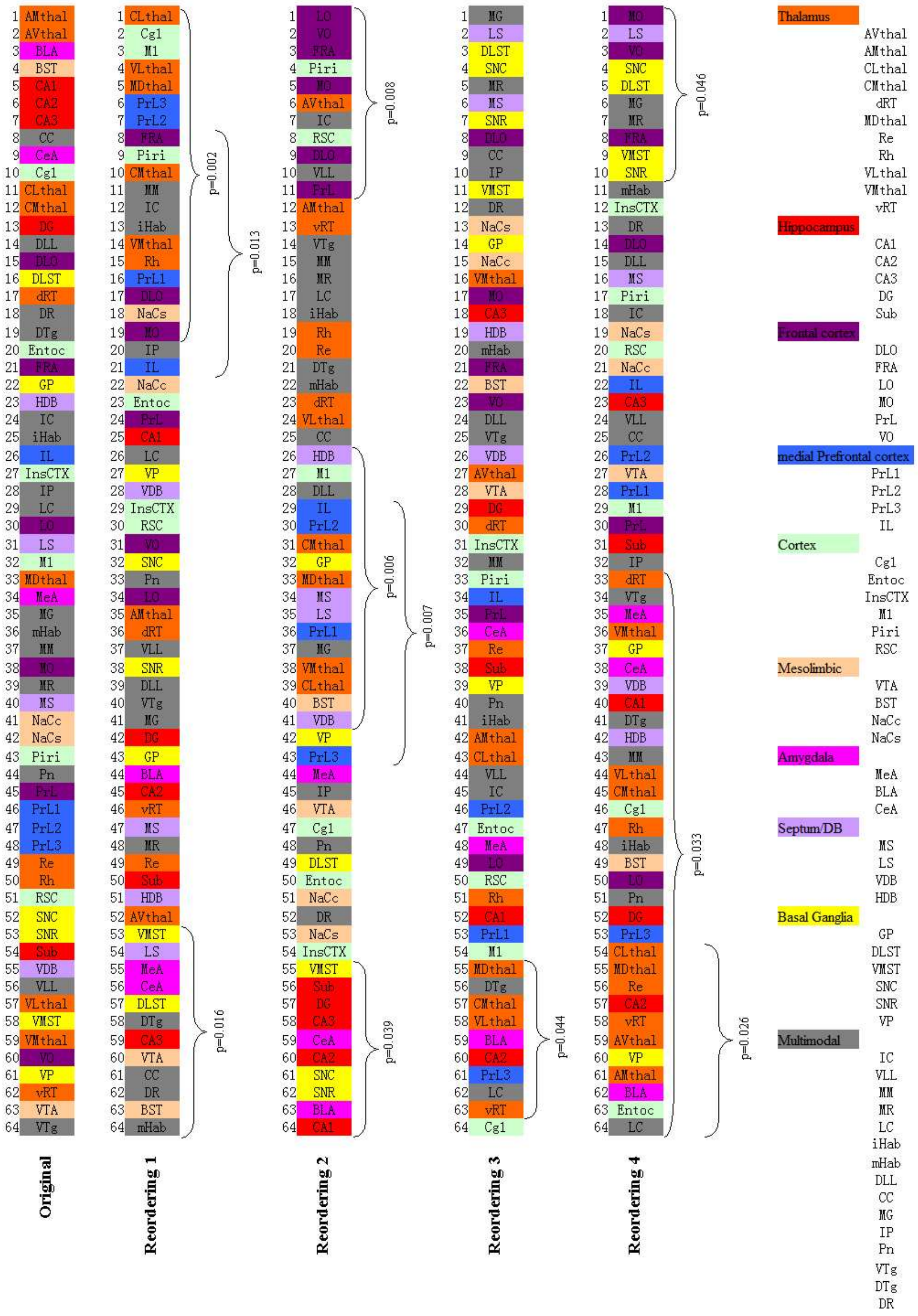
Figure 7.5: Reordering 4: Control (A) and PCP (B) brain networks reordered with the final column of X^{-T}.

reordering 2. Recalling the theory described in Chapter 4 on a pair of weighted matrices, we also reordered the networks in an attempt to reveal clusters in PCP that are not present in Control (Figure 7.2). Figure 7.3 is another attempt toward the same goal. Figure 7.2 gives visual evidence for two discrete node clusters (at the top left and bottom right hand side of the heatmaps) in PCP but not in Control. In Figure 7.3, there appear to be three discrete clusters (top left hand side, center and bottom right hand side of the heatmaps) in PCP but not in Control. On the other hand, Figures 7.2 and 7.3 show no obvious patterns of clustering for nodes in Control animals but not in PCP-treated animals. Both reordering 1 and 2 seem to reveal interesting differences in the form of clusters of nodes that are in PCP-treated animals but not present in Control animals. Perhaps ordering 2 is more informative than reordering 1. In Figure 7.4, we have reordered the networks with the penultimate column from X^{-T}, which we call reordering 3. Recalling the algorithms in Chapter 4, this is an attempt to reveal clusters in Control that are not present in PCP. Analogously, Figure 7.5 is another attempt to reveal clusters in Control but not in PCP, which are reordered with the final column of X^{-T} (reordering 4). Both reordering 3 and 4 seem to show differences in clustering between the experimental groups. Figure 7.4 gives visual evidence for one cluster of brain regions (at the bottom right hand side of the heatmaps) for Control animals that are not present in PCP-treated animals, and in Figure 7.5, there appear to be one cluster (at the bottom right hand side of the heatmaps) for Control animals that are not present in PCP-treated animals. We will quantify these visual observations by the cluster validation method introduced in section 6.3 of Chapter 6 to identify significant clusters in these reorderings (Table 7.1 in section 7.3.2).

7.3.2 Significant PCP-induced alternations in functional brain networks

Table 7.1 shows the rank order of brain regions in the original ordering and in reordering 1, reordering 2, reordering 3 and reordering 4, where color represents the distinct 10 functional groups that each brain regions belongs to. As we described in section 7.3.1, reordering 1 and 2 establish node clusters present in Control animals that are not in PCP-treated animals, whereas, reordering 3 and 4 establish node clusters present in PCP-treated animals that are not in present in Control animals. A small p-value (< 0.05) identifies a significant cluster in these reorderings. There are 3 significant clusters (1 discrete and 2 overlapping) identified in reordering 1, and 3 discrete significant clusters identified for PCP-treated animals in reordering 2. On the other hand, reordering 3 only reveals 1 significant cluster for Control animals, while reordering 4 reveals 3 significant clusters (1 discrete and 2 overlapping). The top cluster (nodes 1-10) is difficult to see on the printed page, but it was validated as a significant cluster ($p < 0.05$). Brain regions are grouped and color coded on the basis of their close anatomical connectivity or established functional role. On a visual basis it appears that discrete clusters of nodes identified through our algorithm may represent node clusters with known connectivity (e.g. cluster nodes 55-64 in reordering 2 appear to be mainly hippocampal). To test the significance of this possible functional segregation, the hypergeometric probability of observing at least the given number of brain regions in a defined functional cluster was calculated. The hypergeometric probability testing has already been introduced in the Supplementary Material at the end of Chapter 6. The significant over-abundance of a given functional group in any GSVD identified significant cluster was set at a hypergeometric probability of $p < 0.005$ and a Bonferroni type correction was applied to the probability value to account for the effect of multiple comparison in investigating 10 pre-defined functional groups ($p < 0.05/10$).

The corresponding probabilities and results are given in Table 7.2. For each significant cluster identified in the reordered matrices, the hypergeometric probability of observing functionally/anatomically-related nodes (as previously defined at the end of Chapter 6) within that cluster grouping was calculated. Anatomically/functionally related groups found to be significantly over-represented within a given functional cluster are highlighted in red. In reordering 4, which identifies nodes significantly clustered in Control animals that are not clustered in PCP-treated animals, the basal ganglia are shown to be over-abundant in the nodes 1-10 cluster of reordering 4. Thalamic regions are confirmed to be significantly over-abundant in both the nodes 33-64 and nodes 54-64 clusters of reordering 4. These results suggest that the basal ganglia and thalamus brain regions are functionally clustered in control animals but these functional clusters are lost following subchronic PCP-treatment. Interestingly, hub regions previously identified in Control animals [27, 29] also showed a significant overabundance in the nodes 54-64 cluster in reordering 4. These hub regions in control animals are known to lose their hub status in PCP-treated animals. Furthermore, as the control functional brain network displays the property of assortativity [27], where hub regions connect to one another within a network, this suggests that our algorithm is capable of identifying alterations in clustering previously identified through the application of other algorithms. Analogously, in reordering 2, which identifies nodes significantly clustered in PCP-treated animals that are not clustered in Control animals, the Frontal cortex regions are shown to be significantly over-represented in the nodes 1-11 cluster, the medial prefrontal cortex regions are shown to be significantly over-abundant in the nodes 29-43 cluster, septum/DB regions are shown to be significantly over-abundant in the nodes 26-41 cluster and hippocampus regions are confirmed to be significantly over-abundant in the nodes 55-64 cluster. This suggests that in PCP-treated animals these anatomically interconnected regions, the frontal cortex, medial prefrontal cortex, septum/DB and hippocampus regions, form discrete functional clusters that are not present under normal conditions, in control animals. From Table 7.2, we can see that reordering 4 and reordering 2 are more informative than reordering 3 and reordering 1 in showing more significantly over-abundant functional groups for one connectivity pattern that are not in the other pattern. This is consistent with the visual evidence from the reordering matrices shown in Figures 7.2 to 7.5. Finally, we mention that multimodal areas are involved in many different processes and each region has a very different role, hence we do not list the hypergeometric probabilities for this functional group in Table 7.2.


p=0.002
p=0.013
p=0.016
p=0.008
p=0.006
p=0.007
p=0.039
p=0.044
p=0.046
p=0.033
p=0.026

	Original	Reordering 1	Reordering 2	Reordering 3	Reordering 4
1	AMthal	CLthal	LO	MG	MO
2	AVthal	Cg1	VO	LS	LS
3	BLA	M1	FRA	DLST	VO
4	BST	VLthal	Piri	SNC	SNC
5	CA1	MDthal	MO	MR	DLST
6	CA2	PrL3	AVthal	MS	MG
7	CA3	PrL2	IC	SNR	MR
8	CC	FRA	RSC	DLO	FRA
9	CeA	Piri	DLO	CC	VMST
10	Cg1	CMthal	VLL	IP	SNR
11	CLthal	MM	PrL	VMST	mHab
12	CMthal	IC	AMthal	DR	InsCTX
13	DG	iHab	vRT	NaCs	DR
14	DLL	VMthal	VTg	GP	DLO
15	DLO	Rh	MM	NaCc	DLL
16	DLST	PrL1	MR	VMthal	MS
17	dRT	DLO	LC	MO	Piri
18	DR	NaCs	iHab	CA3	IC
19	DTg	MO	Rh	HDB	NaCs
20	Entoc	IP	Re	mHab	RSC
21	FRA	IL	DTg	FRA	NaCc
22	GP	NaCc	mHab	BST	IL
23	HDB	Entoc	dRT	VO	CA3
24	IC	PrL	VLthal	DLL	VLL
25	iHab	CA1	CC	VTg	CC
26	IL	LC	HDB	VDB	PrL2
27	InsCTX	VP	M1	AVthal	VTA
28	IP	VDB	DLL	VTA	PrL1
29	LC	InsCTX	IL	DG	M1
30	LO	RSC	PrL2	dRT	PrL
31	LS	VO	CMthal	InsCTX	Sub
32	M1	SNC	GP	MM	IP
33	MDthal	Pn	MDthal	Piri	dRT
34	MeA	LO	MS	IL	VTg
35	MG	AMthal	LS	PrL	MeA
36	mHab	dRT	PrL1	CeA	VMthal
37	MM	VLL	MG	Re	GP
38	MO	SNR	VMthal	Sub	CeA
39	MR	DLL	CLthal	VP	VDB
40	MS	VTg	BST	Pn	CA1
41	NaCc	MG	VDB	iHab	DTg
42	NaCs	DG	VP	AMthal	HDB
43	Piri	GP	PrL3	CLthal	MM
44	Pn	BLA	MeA	VLL	VLthal
45	PrL	CA2	IP	IC	CMthal
46	PrL1	vRT	VTA	PrL2	Cg1
47	PrL2	MS	Cg1	Entoc	Rh
48	PrL3	MR	Pn	MeA	iHab
49	Re	Re	DLST	LO	BST
50	Rh	Sub	Entoc	RSC	LO
51	RSC	HDB	NaCc	Rh	Pn
52	SNC	AVthal	DR	CA1	DG
53	SNR	VMST	NaCs	PrL1	PrL3
54	Sub	LS	InsCTX	M1	CLthal
55	VDB	MeA	VMST	MDthal	MDthal
56	VLL	CeA	Sub	DTg	Re
57	VLthal	DLST	DG	CMthal	CA2
58	VMST	DTg	CA3	VLthal	vRT
59	VMthal	CA3	CeA	BLA	AVthal
60	VO	VTA	CA2	CA2	VP
61	VP	CC	SNC	PrL3	AMthal
62	vRT	DR	SNR	LC	BLA
63	VTA	BST	BLA	vRT	Entoc
64	VTg	mHab	CA1	Cg1	LC

Thalamus	AVthal AMthal CLthal CMthal dRT MDthal Re Rh VLthal VMthal vRT
Hippocampus	CA1 CA2 CA3 DG Sub
Frontal cortex	DLO FRA LO MO PrL VO
medial Prefrontal cortex	PrL1 PrL2 PrL3 IL
Cortex	Cg1 Entoc InsCTX M1 Piri RSC
Mesolimbic	VTA BST NaCc NaCs
Amygdala	MeA BLA CeA
Septum/DB	MS LS VDB HDB
Basal Ganglia	GP DLST VMST SNC SNR VP
Multimodal	IC VLL MM MR LC iHab mHab DLL CC MG IP Pn VTg DTg DR

Table 7.1: All reordered brain region lists with identified significant clusters.

Table 7.2: All hypergeometric probability data for all identified significant clusters

Reordering	**Cluster**	$c(A,B)$	p**-value**	**Cluster size** (n)	**Functional group**	**Total number of RoI in grouping** (m)	**Number of RoI in cluster** (k)	**Hypergeometric probability** $(P(X) \geq k)$
1	nodes 1-19	7.448	0.002	19	Thalamus	11	6	0.056
					Cortex	6	3	0.242
					medial Prefrontal cortex	4	3	0.075
					Frontal cortex	6	3	0.242
					Mesolimbic	4	1	0.766
					PCP Hubs	11	3	0.701
	nodes 8-21	5.874	0.013	14	Thalamus	11	3	0.450
					Cortex	6	1	0.788
					medial Prefrontal cortex	4	2	0.206
					Frontal cortex	6	3	0.113
					Mesolimbic	4	1	0.638
					PCP Hubs	11	4	0.187
	nodes 53-64	7.503	0.016	12	Basal Ganglia	6	2	0.312
					Septum/DB	4	1	0.574
					Amygdala	3	2	0.088
					Hippocampus	5	1	0.659
					Mesolimbic	4	2	0.157
					PCP Hubs	11	3	0.337
2	nodes 1-11	13.863	0.008	11	Frontal cortex	6	6	6.162×10^{-6}
					Cortex	6	2	0.273

Continued on Next Page

Reordering	Cluster	$c(A, B)$	p-value	Cluster size (n)	Functional group	Total number of RoI in grouping (m)	Number of RoI in cluster (k)	Hypergeometric probability ($P(X) \geq k$)
2	nodes 1-11	13.863	0.008	11	Thalamus	11	1	0.897
					PCP Hubs	11	5	0.016
	nodes 29-43	6.836	0.007	15	medial Prefrontal cortex	4	4	2.148×10^{-3}
					Thalamus	11	4	0.229
					Septum/DB	4	3	0.037
					Mesolimbic	4	1	0.667
					Basal Ganglia	6	2	0.432
					PCP Hubs	11	1	0.961
	nodes 26-41	6.931	0.006	16	Septum/DB	4	4	2.864×10^{-3}
					Cortex	6	1	0.836
					medial Prefrontal cortex	4	3	0.045
					Thalamus	11	4	0.274
					Basal Ganglia	6	1	0.836
					Mesolimbic	4	1	0.694
					PCP Hubs	11	1	0.970
	nodes 55-64	6.202	0.039	10	Hippocampus	5	5	3.305×10^{-5}
					Basal Ganglia	6	3	0.044
					Amygdala	3	2	0.061
					PCP Hubs	11	1	0.871
3	nodes 55-63	5.202	0.044	9	Thalamus	11	4	0.040
					Amygdala	3	1	0.370
					Hippocampus	5	1	0.544

Continued on Next Page

Reordering	**Cluster**	$c(A,B)$	p**-value**	**Cluster size** (n)	**Functional group**	**Total number of RoI in grouping** (m)	**Number of RoI in cluster** (k)	**Hypergeometric probability** ($P(X)\geq k$)
3	nodes 55-63	5.202	0.044	9	medial Prefrontal cortex	4	1	0.463
					Control Hubs	12	5	8.566×10^{-3}
4	nodes 1-10	7.770	0.046	10	Frontal cortex	6	3	0.044
					Basal Ganglia	6	4	4.192×10^{-3}
					Septum/DB	4	1	0.502
					Control Hubs	12	0	1.000
	nodes 33-64	3.600	0.033	32	Thalamus	11	11	1.735×10^{-4}
					Amygdala	3	3	0.119
					Basal Ganglia	6	2	0.902
					Hippocampus	5	3	0.500
					Septum/DB	4	2	0.694
					Frontal cortex	6	1	0.988
					medial Prefrontal cortex	4	1	0.943
					Cortex	6	2	0.902
					Mesolimbic	4	1	0.943
					Control Hubs	12	9	0.053
	nodes 54-64	10.817	0.026	11	Thalamus	11	6	1.918×10^{-3}
					Hippocampus	5	1	0.624
					Amygdala	3	1	0.438
					Basal Ganglia	6	1	0.694
					Control Hubs	12	6	3.533×10^{-3}

7.3.3 Discussion

Figure 7.6 illustrates how different functional subsystems may exchange information through a center cluster (hub regions) in the brain. In Figure 7.6, yellow ellipse **A** represents a group of brain regions assigned to a defined functional subsystem. This functional group **A** consists of discrete brain regions, which are represented by blue circles inside. The other yellow ellipse **B** denotes another brain functional subsystem which is influenced by functional activity in **A**. The green circles in **B** represent another set of nodes (brain regions). The comparably smaller gray ellipse **C** is used to denote a center cluster which consists of the hub regions in brain. The black lines inside the ellipses are present in the brain network of PCP-treated animals and are not present in Controls. They represent nodes (brain regions) that are strongly connected within functional group **A** or **B** only in the PCP-treated brain network. The red lines, which are present in Control but lost in PCP-treated animals, denote the connections between the functional areas and the central cluster of hub brain regions. Therefore, here the black lines and the red lines can be used to represent two different types of functional interactions between the brain regions. Normally, different brain regions in functional subsystems are strongly connected to the central cluster (eg. Thalamus), which can transfer information between the different functional subsystems. Suppose **A** is the Frontal cortex functional group which controls behavioral flexibility, and **B** is the Hippocampus functional group which is involved in the memory. In Control animals, the red links are present. Therefore, information can be efficiently exchanged and integrated between these different functional subsystems (groups) **A** and **B** since **A** and **B** are both strongly connected to **C**. In this way, in Control animals, if an alteration in behavior (behavioral flexibility) is required, it can efficiently integrate with the previous experiences an animal remembers (memory). On the contrary, in PCP-treated animals, the red edges present in the Control network are lost. Therefore functional subsystems **A** and **B** can not exchange information efficiently. In this case, brain functions become abnormal and animals cannot modify their behavior based on knowledge of past experiences [99]. On the other hand, the internal black connections within **A** or **B**, which are relatively weak in Control animals, now become strong in PCP-treated animals. And this results in the abnormal functional clustering of these brain regions, as identified through GSVD analysis.

For example, in Figure 7.6, we may argue that the thalamic nodes, the center area **C**, are functionally clustered in Control animals but lost in PCP-treated animals. On the other hand, Frontal cortex and Hippocampus brain regions, functional areas **A** and **B**, are functionally clustered in PCP with the black edges but not in Control.

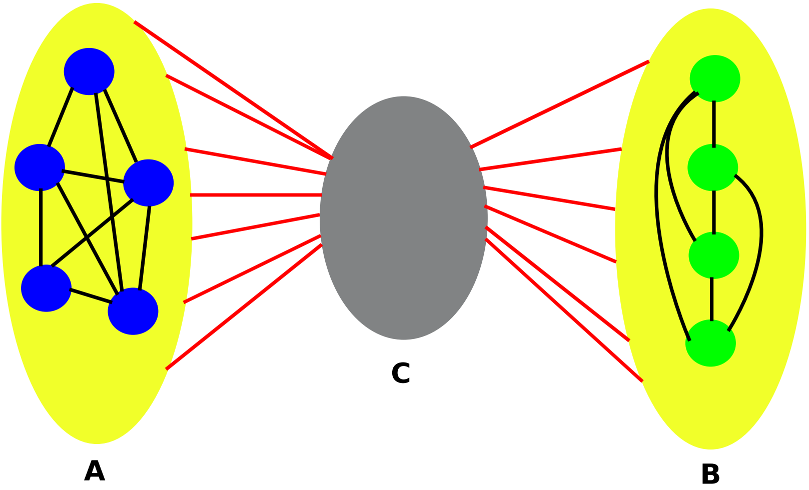


Figure 7.6: Information exchange between different functional groups through the hub regions.

7.4 Summary

In this chapter, we looked at the case of two functional brain networks over the same set of nodes (brain regions), and applied algorithms developed in Chapter 4 for processing of this pair of symmetric, real valued weighted brain networks. The target data, functional brain networks, are correlation matrices of the discrete brain regions. We found some exclusive good clusters present in one brain network that were not in the other. The clusters were found to consist of brain regions forming part of specific functional groups that are significantly over-abundant within the defined given clusters.

Through application to functional brain imaging data, we have shown that our algorithms can provide added insight into the true nature of brain dysfunction in a translational animal model relevant to schizophrenia. Taking such a biological systems approach to functional brain imaging data is currently of great interest to the neuroscience community.

Supplementary Material

The brain data are collected from the same background strain (Hooded Lister, Harlan-Olac, UK) of male rats. The Control animals are healthy animals that are injected with physiological saline. PCP-treated animals receive daily injections of $2.58 mg.kg^{-1}$ PCP.HCl that induces overt alternations in cerebral metabolism (function). We observe alterations of the amount of metabolism in each brain region over the 45 minute period for both Control and PCP-treated animals following tracer injection. Semi-quantitative ^{14}C-2-deoxyglucose (^{14}C-2-DG) Autoradiographic Imaging was applied to Control and PCP-treated rats to investigate metabolism in 64 anatomically distinct brain regions, defined with reference to a stereotactic rat brain atlas [97]. The rate of metabolism, Local cerebral glucose utilization (LCGU), in each region of interest (RoI) was determined as the ratio of ^{14}C present in that region relative to the average ^{14}C concentration in the whole brain of the same animal, and from hereon in will be referred to as the ^{14}C-2-DG uptake ratio. Whole brain average ^{14}C levels were determined from the average ^{14}C concentration across all sections in which a RoI was measured. The correlation between functional activity in the discrete brain regions is generated from the ^{14}C-2-DG uptake ratios for each region across all animals within the same experimental group.

Chapter 8

The Rectangular Case

8.1 Introduction

Generally, networks can be described in terms of graphs [30, 91]. In mathematics and computer science, a graph is usually represented by a matrix. In previous chapters, we developed an intuitive understanding of why the GSVD is useful for processing pairs of related data sets. Firstly, from an optimization viewpoint, we derived algorithms that attempt to shuffle nodes by exploiting the variational properties of the GSVD [21]. Second, in Chapter 4, we interpreted an algorithm for computing the GSVD as an iterative method in order to justify the approach further. However, the corresponding theory and experiments are all based on square matrices.

Recalling the definition of the standard GSVD in (2.4) of Chapter 2, the GSVD works on a pair of matrices A and B with the same column size, allowing different sizes for their rows. Hence the main goal of this chapter is to extend the application of the GSVD to a pair of nonsquare data matrices with the same column size. The nonsquare case corresponds to a bipartite graph where edges connect one set of nodes to another, distinct, set of nodes, so that a_{ij} represents the weight between the ith node in the first set and the jth node in the second set.

The chapter is organized as follows: Section 8.2 considers how an algorithm can be derived to reorder the rows and columns of a pair of nonsquare data sets. Section 8.3 gives an alternative algorithm to solve the same problem. Section 8.4 presents numerical test results of the most promosing algorithm on a synthetic data set. The chapter ends with a summary.

8.2 Algorithm Derivation

We recall that the GSVD expresses a pair of nonsquare matrices $A \in \mathbb{R}^{M\times N}$ with $M \geq N$ and $B \in \mathbb{R}^{P\times N}$ as

$$A = UCX^{-1} \qquad \text{and} \qquad B = VSX^{-1}, \tag{8.1}$$

where $C = \mathrm{diag}(c_i)$ with $0 \leq c_1 \leq c_2 \leq \cdots \leq c_N$, $S = \mathrm{diag}(s_i)$ with $s_1 \geq s_2 \geq \cdots \geq s_q \geq 0$ and $q = \min(P, N)$ [48].

Let the nonsingular matrix X be defined by the column partitioning $X = [x^{[1]},\ x^{[2]},\ \cdots,\ x^{[end]}]$, so that $x^{[i]}$ represents the ith column of X. Analogously, we use $u^{[i]}$ to represent the ith column for orthogonal matrix U and $v^{[i]}$ for the other orthogonal matrix V, and $e^{[i]}$ denotes the ith column of the identity matrix I.

Our aim is to motivate the use of $x^{[end]}$, the final column of X, to reorder the columns of A and B, and use $u^{[end]}$, the final column of orthogonal matrix U, to reorder the rows of A in order to find clusters in A that are not clusters in B. In this case, we do not necessarily have to think about ordering B with the final column vector $v^{[end]}$ from the other orthogonal matrix V.

Our approach is that we first think about columns of A in an attempt to justify the use of $x^{[end]}$ and then think about rows of A in order to justify the use of $u^{[end]}$. We will illustrate the approach in the case where A contains two distinct clusters.

Suppose the rows and columns of A can be divided into two sets, R_1, R_2, and C_1, C_2, respectively, such that if we order with respect to these sets we obtain the matrix as shown in Figure 8.1, where black areas represent large entries.

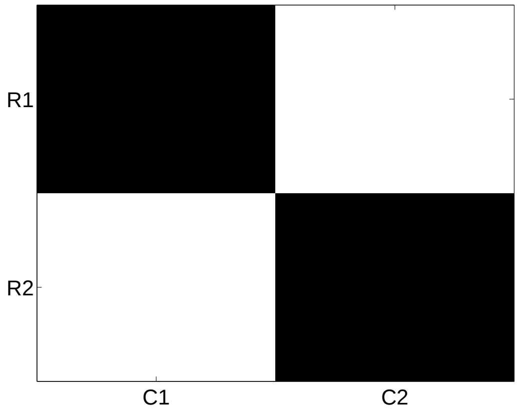


Figure 8.1: Reordered matrix.

We also suppose that B does not have these clusters. We introduce an indicator vector x with $x_i \in \{-1, 1\}$ for the columns. We want x to find the appropriate column structure, with $x_i = 1$ correspond to a column in C_1 and $x_i = -1$ corresponding to a column in C_2.

If x is a good indicator vector then

$$(Ax)_i := \sum_{j=1}^{N} a_{ij} x_j$$

will have large positive entries for rows in one group (say R_1) and large negative entries for rows in the other group. For B, the components

$$(Bx)_i := \sum_{j=1}^{N} b_{ij} x_j \tag{8.2}$$

will generally be small in modulus due to cancellation of terms involving large entries b_{ij}. In other words, maximizing (8.2) will not put large entries in the same cluster for B.

After relaxation to a real valued vector $x \in \mathbb{R}^N$, this motivates the optimization problem

$$\max_{x \neq 0} \frac{\|Ax\|_2^2}{\|Bx\|_2^2}. \tag{8.3}$$

Using the GSVD (8.1), we can solve (8.3) by writing it as

$$\max_{x \neq 0} \frac{\|UCX^{-1}x\|_2^2}{\|VSX^{-1}x\|_2^2}. \tag{8.4}$$

Let $y = X^{-1}x$, so we have

$$\max_{y \neq 0} \frac{\|Cy\|_2^2}{\|Sy\|_2^2}, \tag{8.5}$$

since the 2-norm is invariant under orthogonal transformation.

If we use a thin GSVD as introduced in Chapter 1, then the diagonal matrices C and S are both square. Then, the problem (8.5) can be solved by treating it as a very special case of the more general optimization problem (2.3) for a pair of square matrices in Chapter 2. We have the trivial GSVD

$$C = ICI \qquad \text{and} \qquad S = ISI.$$

Considering the order of the diagonal entries c_i and s_i, we see that $y = e^{[end]}$ is the solution of problem (8.5). Then problem (8.4), or equally (8.3), is solved by $x = Xy = Xe^{[end]} = x^{[end]}$. This justifies the final column of X for ordering columns of A.

Now, let $z = Ax^{[end]}$. Then

$$z_i := \sum_{j=1}^{N} a_{ij} {x_j}^{[end]}.$$

Therefore, we expect $z_i \geq 0$ for indices i corresponding to rows in R_1 and $z_i \leq 0$ for indices corresponding to rows in R_2. Hence, z should be a good indicator for rows in A. Now,

$$\begin{aligned} Ax^{[end]} &= UCX^{-1}x^{[end]} \\ &= UCe^{[end]} \\ &= c_{end}u^{[end]}, \end{aligned}$$

so Ax is a multiple of the final column of U. So the final column of U is a good choice for reordering rows of A.

Analogously, we can motivate the use of $x^{[1]}$, the first column of X, to reorder the columns of B and A, and $v^{[1]}$, the first column of orthogonal matrix V, to reorder the rows of B in order to find good clusters in B that are not present in A by applying similar arguments to the optimization problem

$$\max_{x \neq 0} \frac{\|Bx\|_2^2}{\|Ax\|_2^2}.$$

8.3 A Variant of the Algorithm

In this section, based on insights from the square case, we aim to develop a variant of the algorithm introduced in section 8.2 for reordering nonsquare matrices A and B simultaneously. The idea is derived from the corresponding optimization problem (2.6), $\max_{x \neq 0} \frac{\|B^{-1}x\|_2^2}{\|A^{-1}x\|_2^2}$, which is used to justify the use of columns from X^{-T} for finding the appropriate substructures. In Chapter 2, we used this as a basis for picking out nodes forming good clusters in A that are not in B.

Unfortunately, we can not apply it directly here because nonsquare matrices are not invertible. A possible solution is to use the $N \times N$ matrices A^TA and B^TB so that we may compute the inverse product for these square matrices. Thus we start here from an alternative optimization problem

$$\max_{x \neq 0} \frac{\|A(B^TB)^{-1}x\|_2^2}{\|B(A^TA)^{-1}x\|_2^2}. \tag{8.6}$$

Referring to Chapter 2, the product A^TA and B^TB can be interpreted as a single iteration from an algorithm that computes the dominant singular value of A and B, respectively. It is known from spectral graph theory that the dominant singular vectors give good directions in which to look for clusters [112, 117]. Inverting the weight matrix reverses their importance (the singular value σ becomes σ^{-1}) and hence a spectral clustering approach applied to $(A^TA)^{-1}$ will typically find the opposite of good clusters—poorly connected nodes will be grouped together [39]. So, intuitively, forming $A(B^TB)^{-1}$ in (8.6) should produce a data matrix for which the SVD approach finds good clusters for A and poor clusters for B. Analogously, the opposite holds for $B(A^TA)^{-1}$. Following the same reasoning behind the use of the optimization problem (2.6), $\max_{x \in \mathbb{R}^N, x \neq 0} \frac{\|B^{-1}x\|_2^2}{\|A^{-1}x\|_2^2}$, in Chapter 2, we may then interpret the new optimization problem (8.6) in a similar way. Making $A(B^TB)^{-1}x$ large encourages poor column structures for B, while making $B(A^TA)^{-1}x$ small encourages good column structures for A. In this case, we would base our algorithm on the GSVD of $B(A^TA)^{-1}$ and $A(B^TB)^{-1}$. These can be computed from the GSVD of A and B since, from (8.1), we have

$$A(B^TB)^{-1} = UCS^{-2}X^T = UCS^{-2}(X^{-T})^{-1}$$

and

$$B(A^TA)^{-1} = VSC^{-2}X^T = VSC^{-2}(X^{-T})^{-1}.$$

Then reverting to the arguments in section 8.2, we can use columns from X^{-T} as the basis for reordering. Although problem (8.6) was formed by using $A(B^TB)^{-1}$ and $B(A^TA)^{-1}$, in fact, we do not need to compute the GSVD on these two products. We can use the columns from X^{-T} by computing the GSVD on matrices A and B. That is, the algorithm also applies in the case where A^TA or B^TB are non-invertible.

Alternatively we could rewrite the initial problem (8.6) as

$$\max_{x\neq 0} \frac{\|UCS^{-2}X^Tx\|_2^2}{\|VSC^{-2}X^Tx\|_2^2}. \tag{8.7}$$

This is equivalent to

$$\max_{y\neq 0} \frac{\|CS^{-2}y\|_2^2}{\|SC^{-2}y\|_2^2} \tag{8.8}$$

by letting $y = X^Tx$. Re-applying the arguments in section 8.2, we could use the columns from the inverse of the third factor I in the GSVD of $CS^{-2} = I(CS^{-2})I$ and $SC^{-2} = I(SC^{-2})I$ as a basis for reordering A and B. Since $I^{-1} = I$, problem (8.8) is solved by $y = e^{[end]}$. So

$$x = X^{-T}y = X^{-T}e^{[end]} = x^{\star} \tag{8.9}$$

is the solution of problems (8.7) and (8.6), where $x^{\star}$ is the final column of X^{-T}.

Similarly, we could use the optimization problem

$$\max_{x\neq 0} \frac{\|B(A^TA)^{-1}x\|_2^2}{\|A(B^TB)^{-1}x\|_2^2}$$

to justify that the first column of X^{-T} is a good choice for finding the appropriate column structure for B but not for A.

To summarize, in terms of the GSVD (8.1), we will refer to the two reordering approaches as

Algorithm 3: reorder the columns of nonsquare matrix A via the final column of X and reorder the rows of A via the final column of U, reorder the columns of nonsquare matrix B via the first column of X and reorder the rows of B via the first column of V.

Algorithm 4: reorder the columns of nonsquare matrix A via the final column of X^{-T} and reorder the rows of A via the final column of U, reorder the columns of nonsquare matrix B via the first column of X^{-T} and reorder the rows of B via the first column of V.

8.4 Synthetic Test

In this section we present numerical results. All the results we have suggest that Algorithm 4 is more effective than Algorithm 3. This is consistent with our previous tests for the square case. Hence here we only present results for the X^{-T} reordering. We give a simple, controlled synthetic example where we know the "correct" answer. We generate two nonsquare matrices A and B, as shown in the left pictures in Figure 8.2 and Figure 8.3. There are 40 rows in A but 60 rows in B. Both matrices have the same column size, 20. In both networks, there is a block which consists of the elements a_{ij} and b_{ij} for $1 \leq i \leq 5$ and $1 \leq j \leq 5$, of relatively large entries which is clearly visible as red or yellow colour in the picture. In network A, there is an exclusive substructure consisting of the entries a_{ij} for $6 \leq i \leq 12$ and $6 \leq j \leq 10$ as shown in the same visible way. An exclusive cluster in B consists of the entries b_{ij} for $16 \leq i \leq 20$ and $14 \leq j \leq 20$. In practice, the rows and columns would be ordered arbitrarily, and these substructures would not be immediately apparent.

The GSVD approach is invariant under any reordering of the data. This can be understood by a similar argument to the one given for square case in section 2.4 of Chapter 2. Suppose we are given an arbitrary permutation matrix $P_a \in \mathbb{R}^{M\times M}$ for permuting the rows of A, an arbitrary permutation matrix $P_b \in \mathbb{R}^{P\times P}$ to shuffle the rows in B and another permutation matrix $P \in \mathbb{R}^{N\times N}$ to permute the columns of A and B. If we shuffle the original data from A and B to P_aAP^{-1} and P_bBP^{-1}, then the factorizations $A = UCX^{-1}$ and $B = VSX^{-1}$ are equivalent to $P_aAP = (P_aU)C(P^TX)^{-1}$ and $P_bBP = (P_bV)S(P^TX)^{-1}$. So, if we relabel the original data matrices and then compute the GSVD on the shuffled data, the algorithms still work, with columns of (P^TX) and $(P^TX)^{-T}$ playing the role that was played by those of X and X^{-T} in order to order the columns of the shuffled A and B, and appropriately permuted columns from (P_aU) and (P_bV) are used to reorder the rows of the shuffled data. Hence it does not matter what initial row and column ordering is supplied in our synthetic data.

We emphasize that any end column refers to an end column from a factor produced by computing a thin size GSVD of A and B. The algorithm still works when we compute a standard GSVD of A and B, but we would need be careful about which column is the appropriate $x^\star$ and $u^{[end]}$.

In Figure 8.2, the right picture shows the matrix A with rows and columns reordered according to the ordering of components in the final column of U and final column of X^{-T}. Similarly, the right picture in Figure 8.3 shows the matrix B with rows and columns reordered according to the ordering of components in the first column of V and first column of X^{-T}. We see that it is able to recover the blocks of large entries exclusive to A and B.

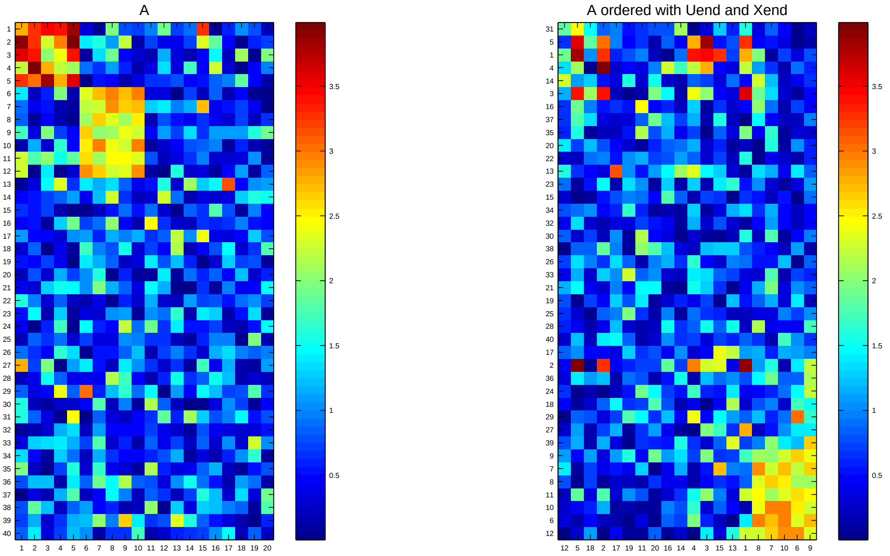


Figure 8.2: Original nonsquare matrix A and the reordering.

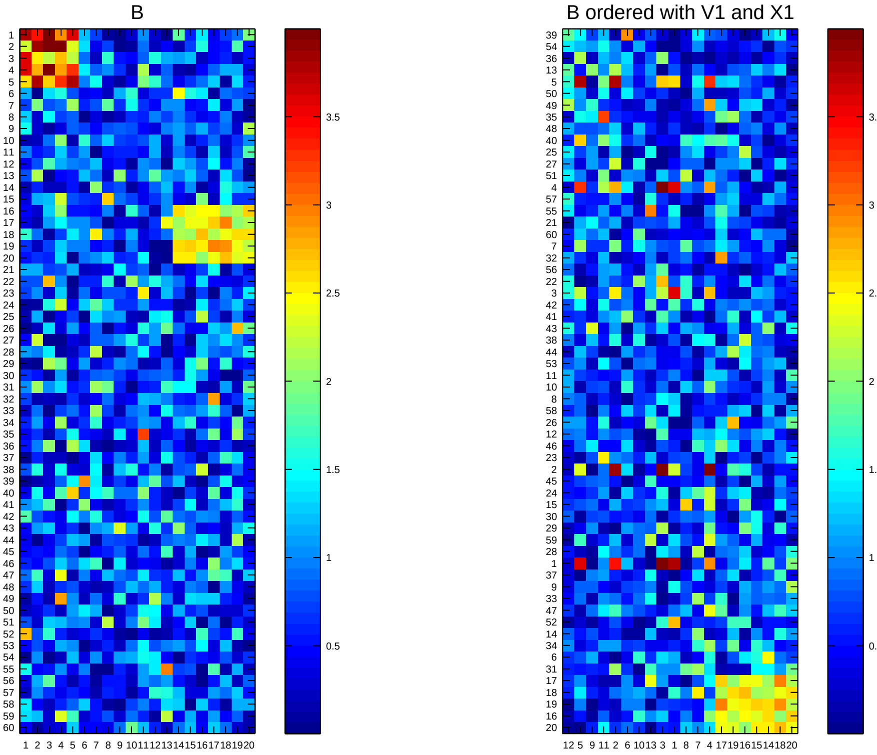


Figure 8.3: Original nonsquare matrix B and the reordering.

8.5 Summary

This chapter focuses on the situation where a pair of nonsquare matrices describe two different types of connection in the form of two bipartite graphs with one node set common to both. We argued from first principles that the Generalized Singular Value Decomposition (8.1) provides a simple and powerful computational tool to find patterns for one graph that are not present for the other. Our work shows that the common factor X^{-T} is useful for reordering the columns of A and B while columns of U and V contribute to picking out the appropriate row structures for A and B separately. We are currently seeking real world nonsquare data sets where the discovery of this type of substructure would be of interest.

Chapter 9

Conclusions

In this dissertation we have explored two different approaches towards an intuitive understanding of why the GSVD is useful for processing pairs of related data sets under the general title of "Complex Networks and the Generalized Singular Value Decomposition". We tested the algorithms with synthetic data and also applied them to real world data. In order to validate the results, we designed a cluster validation method. To examine the usefulness of these algorithms within life science, our biological applications focused on three areas: protein interaction networks, metabolic networks and brain networks. Finally, we also extended the theory and synthetic tests to the case of nonsquare interaction data sets.

In this chapter, we will summarize our work, providing an overview for each topic, accompanied by suggestions for future work.

9.1 Optimization Viewpoint

Based on the optimization view, we developed two different algorithms in order to interpret the principles of the GSVD working on a pair of square matrices A and B in $\mathbb{R}^{N\times N}$.

We first formed an optimization problem $\max_{x\in\mathbb{R}^N,\, x\neq 0} \frac{\|Ax\|_2^2}{\|Bx\|_2^2}$ and showed that we can regard the first few columns of X as candidates for finding good clusters in B that are not in A and, analogously, use the final few columns of X as candidates for picking out good clusters in A that are not present in B.

Second, we developed a variant of the algorithm. In this case, we used $\max_{x\in\mathbb{R}^N,\, x\neq 0} \frac{\|B^{-1}x\|_2^2}{\|A^{-1}x\|_2^2}$. We showed that this leads to use of columns from X^{-T}. Although this algorithm is derived from the GSVD of A^{-1} and B^{-1}, in practice the computation of the inverse products of A and B is not necessary.

We tested both of these heuristic algorithms with synthetic data and observed that X^{-T} gave the best results. Hence we focused on the use of this algorithm to test real world data sets from social science and neuroscience.

The following areas may turn out to be highly relevant future directions for this topic:

(i) **Exploring Common Patterns** In this dissertation, we focused on dissimilarity between two related networks in terms of clustering. In other words, our algorithms were designed to find clusters exclusive to one graph. An interesting alternative is to look for common clusters present in both networks. For real world biological data, the existence of this common pattern for the two graphs would indicate a biological process which is relevant to both types of interactions. A possible starting point for solving this problem based on the optimization view is to consider the optimization problem $\max_{x \in \mathbb{R}^N, x \neq 0} \frac{\|Ax\|_2^2}{\|B^{-1}x\|_2^2}$. Alternatively, we could re-consider the optimization problems (2.3) or (2.6) to use the middle columns instead of the first or final columns from X or X^{-T}.

(ii) **Data Normalization** Matrix decompositions are numerical computations, so the magnitudes of the values in the datasets must be comparable, otherwise the large magnitudes will have a greater influence on the result than the the smaller ones. In statistics, a common way to adjust data values is to use the standard score [113], that is the z-score or z-value. This makes the values roughly similar in magnitude, but the standard score normalization implicitly assumes that the raw data are normally distributed. Recently, the Sinkhorn-Knopp (SK) algorithm was introduced to normalize a large set of datasets from a matrix computation perspective [76, 77]. The SK algorithm can be used to balance the matrix, and is perhaps the simplest method for finding a diagonal scaling of a given square nonnegative matrix that balances the matrix to be doubly stochastic. In addition, a related normalization method is formed and used in the spectral clustering method [59, 60, 61, 62, 73], especially for the SVD. This normalization method also uses diagonal matrices derived from the rows or the columns of the original matrix to scale the data sets. In this dissertation, we did not use a normalization method in our tests since the numerical results were visually and statistically significant. However, a normalization technique would be vital in cases where the data is poorly calibrated. So normalization seems to be a good direction for future work.

(iii) **Comparing the Two Heuristic Algorithms** Although two algorithms are developed in this work, all computational tests suggest that the performance of Algorithm 2 is better than Algorithm 1. Mathematical arguments to explain this phenomenon would therefore be a valuable addition in this area.

(iv) **Optimal Vectors** In all the experiments, especially when tested on some real data, it is difficult for us to predict which column of the common factor X^{-T} will give us the best reordering of A and B and also to decide how to define a cluster from the reorderings produced by the columns. In order to develop a fully automated algorithm it would be necessary to combine the cluster validation procedures developed here with some sort of combinatorial searching.

9.2 Power Method Viewpoint

In Chapter 4, we interpreted how the GSVD works via an iterative method. We showed that the reordering algorithm can be regarded as the limit process arising from a process that reshuffles the nodes according to their relative strengths of connectivity in the two networks.

9.3 Cluster Validation

In Chapter 3, we designed a method to check if our test results are statistically significant. The null hypothesis H_0 of our test is that the cluster quality that we discovered could have arisen from the class of random networks produced in a randomization step. If the final p-value is less than 0.05, we reject H_0 so that it is very unlikely that this strength of clustering in the real data would arise if we take a random network.

We designed two different ways to define the cluster quality measure. We also used three different approaches to randomize the networks. We found that all approaches gave consistent results.

Future efforts within this topic might include attempting to make the validation method work for nonsquare matrices. In Chapter 8, we have not validated the clusters found in the nonsquare case. The main challenge is how to define the cluster quality for two nonsquare matrices with same column size but which have different row sizes. The current cluster quality defined in Chapter 3 can only work in the case of two matrices which are square and binary, so the data arises from the same group of nodes. In the nonsquare case, how can we compare an examined area (block) in one matrix with the other nonsquare matrix with different row size? In addition, we would also need to consider how to randomize the data in a reasonable way.

9.4 The Life Science Application

In Chapter 2, we applied our heuristic algorithms to some real world data including social networks and neural networks. Then, in Chapters 5, 6 and 7, we extended the application to real biological complex networks. These applications were protein interaction networks, metabolic networks and brain networks.

In Chapter 5, two types of interaction were available for the same large group of proteins: Protein-Protein Interaction (PPI) and Genetic Interaction (GI). Before running the GSVD, we applied some techniques to preprocess the data to remove the protein name inconsistency. For a fast computation, we also trimmed the data size by a common threshold for both matrices. Then we ran our algorithm on the data and identified some candidate node groups which are strongly connected under one type of interaction but are weakly connected in the other type. We then verified them by computing the p-values. We then recorded the protein name lists for the statistically consistent clusters. This project is still ongoing and our coworkers are now processing these name lists to verify their biological meaning.

Another important biological application is metabolic data tested in Chapter 6. In this chapter, we tested a pair of metabolic networks from Control and PCP-treated animals, separately. In the data, each node is a given molecule, and the weighted edges represent the metabolic connections between the corresponding two molecules. We first reordered the data based on an extension of our algorithm to the case of weighted edges. Then we compute the p-values to validate the observed clusters. To study the significance of the components within the given identified clusters, we introduced a hypergeometric probability test. We concluded that the algorithm is useful for exploring metabolic pathway disruption in the PCP model of schizophrenia.

In Chapter 7 we compared two types of functional interaction data over the same set of brain regions. They came from Control and PCP-treated animals, separately. Each node (brain region) was pre-assigned to a brain functional group. We were able to identify a significant cluster in Control animals that was destroyed in PCP-treated animals, and vice versa.

A promising and important future direction in life science is to develop and apply the techniques for the case of microarray data sets. Microarray data sets can be viewed as large nonsquare matrices $A \in \mathbb{R}^{M \times N}$ which record the behavior of a set of M genes across a set of N samples. "Samples" may correspond to tissues from different but related tumours, plus some normal tissues, or may be snapshots at different points in time for a single tissue. Correlations between samples, or between genes, then lead to square matrices. Some spectral clustering methods, such as the SVD, have been used for microarray data sets [50, 59, 60, 61, 73]. Previous notable work which has applied the GSVD to related gene expression data sets [5] also indicates that we might apply the GSVD to microarray data sets.

9.5 Nonsquare Case

In Chapter 8, we developed a theory for using a column of X or X^{-T} to reorder the columns of A and B while using the columns from U to shuffle the rows in A and columns from V to shuffle the rows in B in the case where A and B are nonsquare with the same column size. We proposed two different optimization problems to justify the columns of X and from X^{-T}.

We also expect that the power method could be used to justify the approach. The main difficulty in using the approach of Chapter 4 is that we cannot use the inverse of a nonsquare matrix. A possible solution is to re-explore the problem by using the pseudo inverse [13] of the nonsquare matrices. If this direction is useful, we might also use the pseudo inverse to form a more intuitive optimization problem, generalizing $\max_{x\neq 0} \frac{\|B^{-1}x\|_2^2}{\|A^{-1}x\|_2^2}$, to the nonsquare case.

A major opportunity for future work is to apply this novel algorithm to real data sets. Raw (uncorrelated) microarray data would be a very promising category.

Bibliography

[1] S. Achard, R. Salvador, B. Whitcher, J. Suckling, and E. Bullmore, *A resilient, low-frequency, small-world human brain functional network with highly connected association cortical hubs*, The Journal of Neuroscience, 26 (2006), pp. 63–72.

[2] A. T. Adai, S. V. Date, S. Wieland, and E. M. Marcotte, *LGL: creating a map of protein function with an algorithm for visualizing very large biological networks*, Journal of Molecular Biology, 340 (2004), pp. 179–190.

[3] R. Albert and A. L. Barabási, *Statistical mechanics of complex networks*, Reviews of Modern Physics, 74 (2002), pp. 47–97.

[4] O. Alter, P. O. Brown, and D. Botsein, *Singular value decomposition for genome-wide expression data processing and modeling*, PNAS, 97 (2000), pp. 10101–10106.

[5] O. Alter, P. O. Brown, and D. Botsein, *Generalized singular value decomposition for comparative analysis of genome-scale expression data sets of two different organisms*, PNAS, 100 (2003), pp. 3351–3356.

[6] G. D. Bader and C. W. Hogue, *Analyzing yeast protein-protein interaction data obtained from different sources*, Nature Biotechnology, 20 (2002), pp. 991–997.

[7] Z. Bai, *The CSD, GSVD, their applications and computations*, in University of Minnesota, 1992, pp. 3–5.

[8] Z. Bai and J. W. Demmel, *Computing the Generalized Singular Value Decomposition*, SIAM Journal on Scientific Computing, 14 (1993), pp. 1464–1486.

[9] D. S. Bassett and E. Bullmore, *Small-world brain networks*, Neuroscientist, 12 (2006), pp. 512–523.

[10] D. S. Bassett, E. Bullmore, B. A. Verchinski, V. S. Mattay, D. R. Weinberger, and A. Meyer-Lindenberg, *Hierarchical organization of human cortical networks in health and schizophrenia*, The Journal of Neuroscience, 28 (2008), pp. 9239–9248.

[11] D. S. Bassett, E. T. Bullmore, A. Meyer-Lindenberg, J. A. A. Apud, D. R. Weinberger, and R. Coppola, *Cognitive fitness of cost-efficient brain functional networks*, PNAS, 106 (2009), pp. 11747–11752.

[12] D. S. Bassett, A. Meyer-Lindenberg, S. Achard, T. Duke, and E. Bullmore, *Adaptive reconfiguration of fractal small-world human brain functional networks*, PNAS, 103 (2006), pp. 19518–19523.

[13] A. Ben-Israel and T. N. Greville, *Generalized inverses: Theory and applications*, Krieger, 1980.

[14] J. A. Berger, S. Hautaniemi, and S. K. Mitra, *Comparative analysis of gene expression and DNA copy number data for pancreatic and breast cancers using an orthogonal decomposition*, Proceedings of the 2004 IEEE Computational Systems Bioinformatics Conference (CSB 2004), 0 (2004), pp. 584–585.

[15] J. A. Berger, S. Hautaniemi, S. K. Mitra, and J. Astola, *Jointly analyzing gene expression and copy number data in breast cancer using data reduction models*, IEEE/ACM Transactions on Computational Biology and Bioinformatics, 3 (2006), pp. 2–16.

[16] S. Blackford, *LAPACK manual*, http://www.netlib.org/lapack/lug/node36.html.

[17] R. L. Breiger and P. E. Pattison, *Cumulated social roles: The duality of persons and their algebras*, Social Networks, 8 (1986), pp. 215–256.

[18] E. Bullmore and O. Sporns, *Complex brain networks: graph theoretical analysis of structural and functional systems*, Nature Reviews Neuroscience, 10 (2009), pp. 186–198.

[19] Y. Byun and K. Han, *Visualization of protein-protein interaction networks using force-directed layout*, in Computational Science - ICCS 2003: International Conference, Melbourne, Australia and St. Petersburg, Russia June 24, 2003 Proceedings, Part III, vol. 2659, 2003, pp. 190–199.

[20] S. Cho, S. G. Park, D. H. Lee, and B. C. Park, *Protein-protein interaction networks: from interactions to networks*, Journal of Biochemistry and Molecular Biology, 37 (2004), pp. 45–52.

[21] M. T. Chu, R. E. Funderlic, and G. H. Golub, *On a variational formulation of the generalized singular value decomposition*, SIAM Journal on Matrix Analysis and Applications, 18 (1997), pp. 1082–1092.

[22] W. S. Cleveland, *The elements of graphing data*, Wadsworth Publ. Co., Belmont, CA, USA, 1985.

[23] S. M. Cochran, M. Kennedy, C. E. McKerchar, L. J. Steward, J. A. Pratt, and B. J. Morris, *Induction of metabolic hypofunction and neurochemical deficits after chronic intermittent exposure to phencyclidine: Differential modulation by antipsychotic drugs*, Neuropsychopharmacology, 28 (2003), pp. 265–275.

[24] J. J. Crofts and D. J. Higham, *A weighted communicability measure applied to complex brain networks*, Journal of the Royal Society Interface, 6 (2009), pp. 411–414.

[25] L. L. Davidson and R. W. Heinrichs, *Quantification of frontal and temporal lobe brain-imaging findings in schizophrenia: a meta-analysis*, Psychiatry Research: Neuroimaging, 122 (2003), pp. 69–87.

[26] N. Dawson, L. Ferrington, H. J. Olverman, A. J. Harmar, and P. A. T. K. and, *Sex influences the effect of a lifelong increase in serotonin transporter function on cerebral metabolism*, Journal of Neuroscience Research, 87 (2009), pp. 2375–2385.

[27] N. Dawson, D. J. Higham, B. J. Morris, and J. A. Pratt, *Alterations in functional brain network structure induced by subchronic phencyclidine (PCP) treatment parallel those seen in schizophrenia.* Poster pre-

sented at the 2nd Biennial Schizophrenia International Research Society Conference, 10-14 April 2010. Florence, Italy.

[28] N. Dawson, R. J. Thompson, A. McVie, D. M. Thomson, B. J. Morris, and J. A. Pratt, *Modafinil reverses phencyclidine (PCP)-induced deficits in cognitive flexibility, cerebral metabolism and functional brain connectivity*, Schizophrenia Bulletin (In Press), (2010).

[29] N. Dawson, X. Xiao, D. J. Higham, B. J. Morris, and J. A. Pratt, *Sustained NMDA receptor hypofunction induces compromised neural systems integration and schizophrenia-like alterations in functional brain networks*, submitted (2011).

[30] E. de Silva and M. P. H. Stumpf, *Complex networks and simple models in biology*, Journal of the Royal Society Interface, 2 (2005), pp. 419–430.

[31] L. Deuker, E. T. Bullmore, M. Smith, S. Christensen, P. J. Nathan, B. Rockstroh, and D. S. Bassett, *Reproducibility of graph metrics of human brain functional networks*, NeuroImage, 47 (2009), pp. 1460–1468.

[32] B. Drees, V. Thorsson, G. Carter, A. Rives, M. Raymond, I. Avila-Campillo, P. Shannon, and T. Galitski, *Derivation of genetic interaction networks from quantitative phenotype data*, Genome Biology, 6 (2005), p. R38.

[33] R. Durrett, *Random Graph Dynamics (Cambridge Series in Statistical and Probabilistic Mathematics)*, Cambridge University Press, 2006.

[34] S. R. Eddy, *GENETICS: Total information awareness for worm genetics*, Science, 311 (2006), pp. 1381–1382.

[35] A. Egerton, L. Reid, S. McGregor, S. M. Cochran, B. J. Morris, and J. A. Pratt, *Subchronic and chronic PCP treatment produces temporally distinct deficits in attentional set shifting and prepulse inhibition in rats*, Psychopharmacology, 198 (2008), pp. 37–49.

[36] A. Egerton, L. Reid, C. E. McKerchar, B. J. Morris, and J. A. Pratt, *Impairment in perceptual attentional set-shifting following PCP administration: a rodent model of set-shifting deficits in schizophrenia*, Psychopharmacology, 179 (2005), pp. 77–84.

[37] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, *Cluster analysis and display of genome-wide expression patterns*, PNAS, 95 (1998), pp. 14863–14868.

[38] E. Estrada, *Protein bipartivity and essentiality in the yeast protein-protein interaction network*, Journal of Proteome Research, 5 (2006), pp. 2177–2184.

[39] E. Estrada, *Topological structural classes of complex networks*, Physical Review E, 75 (2007), p. 016103.

[40] E. Estrada, D. J. Higham, and N. Hatano, *Communicability and multipartite structures in complex networks at negative absolute temperatures*, Physical Review E, 78 (2008), p. 026102.

[41] E. Estrada, D. J. Higham, and N. Hatano, *Communicability betweenness in complex networks*, Physica A: Statistical Mechanics and its Applications, 388 (2009), pp. 764–774.

[42] S. R. A. Fisher, *Statistical Methods for Research Workers. Fourteenth Edition Revised*, Oliver & Boyd, 1925.

[43] C. Friedrich and F. Schreiber, *Visualisation and navigation methods for typed protein-protein interaction networks*, Applied Bioinformatics, 2 (2003), pp. S19–S24.

[44] J. Gerke, K. Lorenz, and B. Cohen, *Genetic interactions between transcription factors cause natural variation in yeast*, Science, 323 (2009), pp. 498–501.

[45] J. D. Gibbons and S. Chakraborti, *Nonparametric Statistical Inference (Statistics: a Series of Textbooks and Monogrphs)*, CRC, 2003.

[46] M. Girvan and M. E. J. Newman, *Community structure in social and biological networks*, PNAS, 99 (2002), pp. 7821–7826.

[47] R. Gnanadesikan, *Methods for Statistical Data Analysis of Multivariate Observations (Wiley series in probability & mathematical statistics)*, John Wiley & Sons Inc, 1977.

[48] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, third ed., 1996.

[49] I. J. Gomez Portillo and P. M. Gleiser, *An adaptive complex network model for brain functional networks*, PLoS ONE, 4 (2009), p. e6863.

[50] P. Grindrod, D. J. Higham, G. Kalna, J. K. Vass, A. Spence, and Z. Stoyanov, *DNA Meets the SVD*, Mathematics Today, (2008).

[51] P. Hage and F. Harary, *Structural Models in Anthropology*, Cambridge University Press, Cambridge, 1983.

[52] P. Hagmann, L. Cammoun, X. Gigandet, R. Meuli, C. J. Honey, V. J. Wedeen, and O. Sporns, *Mapping the structural core of human cerebral cortex*, PLoS Biology, 6 (2008), pp. 1479–1493.

[53] C.-G. Hahn, H.-Y. Wang, D.-S. Cho, K. Talbot, R. E. Gur, W. H. Berrettini, K. Bakshi, J. Kamins, K. E. Borgmann-Winter, S. J. Siegel, R. J. Gallop, and S. E. Arnold, *Altered neuregulin 1-erbB4 signaling contributes to NMDA receptor hypofunction in schizophrenia*, Nature Medicine, 12 (2006), pp. 824–828.

[54] K. Han and Y. Byun, *Three-dimensional visualization of protein interaction networks*, Computers in Biology and Medicine, 34 (2004), pp. 127–139.

[55] F. Harary, *On the notion of balance of a signed graph*, Michigan Mathematical Journal, 2 (1953), pp. 143–146.

[56] T. Hase, H. Tanaka, Y. Suzuki, S. Nakagawa, and H. Kitano, *Structure of protein interaction networks and their implications on drug design*, PLoS Computational Biology, 5 (2009), p. e1000550.

[57] D. J. Higham, *An Introduction to Financial Option Valuation: Mathematics, Stochastics and Computation*, Cambridge University Press, Cambridge, 2004.

[58] D. J. Higham, *Spectral reordering of a range-dependent weighted random graph*, IMA Journal of Numerical Analysis, 25 (2005), pp. 443–457.

[59] D. J. Higham, G. Kalna, and M. Kibble, *Spectral clustering and its use in bioinformatics*, Journal of Computational and Applied Mathematics, 204 (2007), pp. 25–37.

[60] D. J. Higham, G. Kalna, and J. K. Vass, *Analysis of the singular value decomposition as a tool for processing microarray expression data*, in Proceedings of ALGORITMY 2005,Conference on Scientific Computing, slovakia, 2005, Slovak University of Technology, pp. 250–259.

[61] D. J. Higham, G. Kalna, and J. K. Vass, *Spectral analysis of two-signed microarray expression data*, Mathematical Medicine and Biology, 24 (2007), pp. 131–148.

[62] D. J. Higham and M. Kibble, *A unified view of spectral clustering*, Mathematics Research Report 02, University of Strathclyde, Glasgow, UK, (2004), pp. 1–17.

[63] D. J. Higham, M. Rašajski, and N. Pržulj, *Fitting a geometric graph to a protein-protein interaction network*, Bioinformatics, 24 (2008), pp. 1093–1099.

[64] K. Hill, L. Mann, K. R. Laws, C. M. E. Stephenson, I. Nimmo-Smith, and P. J. McKenna, *Hypofrontality in schizophrenia: a meta-analysis of functional imaging studies*, Acta Psychiatrica Scandinavica, 110 (2004), pp. 243–256.

[65] P. Holme, F. Liljeros, C. R. Edling, and B. J. Kim, *Network bipartivity*, Physical Review E, 68 (2003), p. 056107.

[66] C. J. Honey, O. Sporns, L. Cammoun, X. Gigandet, J. P. Thiran, R. Meuli, and P. Hagmann, *Predicting human resting-state functional connectivity from structural connectivity*, PNAS, 106 (2009), pp. 2035–2040.

[67] M. Humphries, K. Gurney, and T. Prescott, *The brainstem reticular formation is a small-world, not scale-free, network*, Proceedings of the Royal Society B: Biological Sciences, 273 (2006), pp. 503–511.

[68] T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, and Y. Sakaki, *A comprehensive two-hybrid analysis to explore the yeast protein interactome*, PNAS, 98 (2001), pp. 4569–4574.

[69] M. Kaiser, *Mean clustering coefficients: the role of isolated nodes and leafs on clustering measures for small-world networks*, New Journal of Physics, 10 (2008), p. 083042.

[70] M. Kaiser and C. C. Hilgetag, *Modelling the development of cortical systems networks*, Neurocomputing, 58-60 (2004), pp. 297–302.

[71] M. Kaiser and C. C. Hilgetag, *Spatial growth of real-world networks*, Physical Review E, 69 (2004), p. 036103.

[72] M. Kaiser and C. C. Hilgetag, *Nonoptimal component placement, but short processing paths, due to long-distance projections in neural systems*, PLoS Computational Biology, 2 (2006), p. e95.

[73] G. Kalna, J. K. Vass, and D. J. Higham, *Multidimensional partitioning and bi-partitioning: analysis and application to gene expression datasets*, International Journal of Computer Mathematics, 85 (2008), pp. 475–485.

[74] S. Kelly, A. Bieneman, J. B. Uney, and J. McCulloch, *Cerebral glucose utilization in transgenic mice overexpressing heat shock protein 70 is altered by dizocilpine*, European Journal of Neuroscience, 15 (2002), pp. 945–952.

[75] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein, *Spectral biclustering of microarray data: Coclustering genes and conditions*, Genome Research, 13 (2003), pp. 703–716.

[76] P. A. Knight, *The Sinkhorn-Knopp algorithm: Convergence and applications*, SIAM Journal on Matrix Analysis and Applications, 30 (2008), pp. 261–275.

[77] P. A. Knight and D. Ruiz, *A fast algorithm for matrix balancing*, in Web Information Retrieval and Linear Algebra Algorithms, no. 07071 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2007.

[78] C. Kurtzman and J. Fell, *Yeast Systematics and Phylogeny-Implications of Molecular Identification Methods for Studies in Ecology*, Germany:Springer-Verlag Berlin Herdelberg, 2005, ch. Biodiversity and Ecophysiology of Yeasts (The Yeast Handbook), pp. 11–30.

[79] J. J. Leader, *Limit orbits of a power iteration for dominant eigenvalue problems*, Applied Mathematics Letters, 4 (1991), pp. 41–44.

[80] J.-L. Legras, D. Merdinoglu, J.-M. Cornuet, and F. Karst, *Bread, beer and wine: Saccharomyces cerevisiae diversity reflects human history*, Molecular Ecology, 16 (2007), pp. 2091–2102.

[81] B. Lehner, C. Crombie, J. Tischler, A. Fortunato, and A. G. Fraser1, *Systematic mapping of genetic interactions in caenorhabditis elegans identifies common modifiers of diverse signaling pathways*, Nature Genetics, 38 (2006), pp. 896–903.

[82] D. A. Lewis and B. Moghaddam, *Cognitive dysfunction in schizophrenia: Convergence of gamma-aminobutyric acid and glutamate alterations*, Arch Neurol, 63 (2006), pp. 1372–1376.

[83] E. Limpert, W. A. Stahel, and M. Abbt, *Log-normal distributions across the sciences: Keys and clues*, BioScience, 51 (2001), pp. 341–352.

[84] M. F. Lopez, *MS discovery-to-targeted SRM workflows incorporating ROC curve analysis of putative biomarker*.

[85] *MATLAB documentation*, http://www.mathworks.com/access/helpdesk/help/techdoc/r

[86] D. N. McCloskey and S. Ziliak, *The Cult of Statistical Significance: How the Standard Error Costs Us Jobs, Justice, and Lives (Economics, Cognition, and Society)*, University of Michigan Press, 2008.

[87] *MetWorks Metabolite Identification Software*, https://fscimage.fishersci.com/images/D14296 .pdf.

[88] T. Milenković, J. Lai, and N. Pržulj, *Graphcrunch: A tool for large network analyses*, BMC Bioinformatics, 9 (2008), p. 70.

[89] J. L. Morrison, R. Breitling, D. J. Higham, and D. R. Gilbert, *A lock-and-key model for protein-protein interactions*, Bioinformatics, 22 (2006), pp. 2012–2019.

[90] R. Murai, Y. Noda, K. Matsui, H. Kamei, A. Mouri, K. Matsuba, A. Nitta, H. Furukawa, and T. Nabeshima, *Hypofunctional glutamatergic neurotransmission in the prefrontal cortex is involved in the emotional deficit induced by repeated treatment with phencyclidine in mice: Implications for abnormalities of glutamate release and NMDA-CaMKII signaling*, Behavioural Brain Research, 180 (2007), pp. 152–160.

[91] M. E. J. Newman, *The structure and function of complex networks*, SIAM Review, 45 (2003), pp. 167–256.

[92] M. E. J. Newman, *A measure of betweenness centrality based on random walks*, Social Networks, 27 (2005), pp. 39–54.

[93] J. A. Nickoloff and M. F. Hoekstra, *DNA Damage and Repair: Volume I: DNA Repair in Prokaryotes and Lower Eukaryotes (Contemporary Cancer Research)*, Humana Press, 1998.

[94] J. Paananen and G. Wong, *FORG3D: Force-directed 3d graph editor for visualization of integrated genome scale data*, BMC Systems Biology, 3 (2009), p. 26.

[95] C. C. Paige and M. A. Saunders, *Towards a generalized singular value decomposition*, SIAM Journal on Numerical Analysis, 18 (1981), pp. 398–405.

[96] B. N. Parlett and W. G. Poole, *A geometric theory for the QR, LU and power iterations*, SIAM Journal on Numerical Analysis, 10 (1973), pp. 389–412.

[97] G. Paxinos and C. Watson, *The Rat Brain in Stereotaxic Coordinates*, Academic Press, 4th ed., January 1998.

[98] E. Phizicky and S. Fields, *Protein-protein interactions: methods for detection and analysis*, Microbiological Review, 59 (1995), pp. 94–123.

[99] J. Pratt, C. Winchester, A. Egerton, S. Cochran, and B. Morris, *Modelling prefrontal cortex deficits in schizophrenia: implications for treatment*, British Journal of Pharmacology, (2008), pp. S465–S470.

[100] N. Pržulj, D. G. Corneil, and I. Jurisica, *Modeling interactome: Scale-free or gemetric?*, Bioinformatics, 20 (2004), pp. 3508–3515.

[101] N. Pržulj and D. J. Higham, *Modelling protein-protein interaction netwroks via a stickiness index*, Journal of the Royal Society Interface, 3 (2006), pp. 711–716.

[102] N. Pržulj, D. A. Wigle, and I. Jurisica, *Functional topology in a network of protein interactions*, Bioinformatics, 20 (2004), pp. 340–348.

[103] K. E. Read, *Cultures of the Central Highlands, New Guinea*, Southwestern Journal of Anthropology, 10 (1954), pp. 1–43.

[104] A. Roguev, S. Bandyopadhyay, M. Zofall, K. Zhang, T. Fischer, S. R. Collins, H. Qu, M. Shales, H.-O. Park, J. Hayles,

K.-L. Hoe, D.-U. Kim, T. Ideker, S. I. Grewal, J. S. Weissman, and N. J. Krogan, *Conservation and Rewiring of Functional Modules Revealed by an Epistasis Map in Fission Yeast*, Science, 322 (2008), pp. 405–410.

[105] M. Rudemo, *Empirical choice of histograms and kernel density estimators*, Scandinavian Journal of Statistics, 9 (1982), pp. 65–78.

[106] Y. Saad, *Numerical Methods for Large Nonsymmetric Eigenvalue Problems (Algorithms & Architectures for Advanced Scientific Computing)*, Manchester University Press, 1992.

[107] D. Sarracino, B. Krastins, A. Prakash, and M. F. Lopez, *Quantitative proteomic workflow for discovery of early rejection kidney transplant peptide biomarkers and subsequent development of SRM assays in urine.*

[108] D. Sarracino, B. Krastins, A. Prakash, and M. F. Lopez, *A robust and sensitive workflow for label-free, quantitative identification of differentially expressed, endogenous peptides in human serum.*

[109] A. Schreiber, N. Shirley, R. Burton, and G. Fincher, *Combining transcriptional datasets using the generalized singular value decomposition*, BMC Bioinformatics, 9 (2008), p. 335.

[110] B. Schwikowski, P. Uetz, and S. Fields, *A network of proteinprotein interactions in yeast*, Nature Biotechnology, 18 (2000), pp. 1257–1261.

[111] M. SCIGELOVA, K. KLAGKOU, AND G. WOFFENDIN, *Analysis of beer using a high speed U-HPLC coupled to a linear ion trap hybrid mass spectrometer*. Thermo Fisher Scientific, Hemel Hempstead, UK.

[112] J. SHI AND J. MALIK, *Normalized cuts and image segmentation*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 22 (2000), pp. 888–905.

[113] D. SKILLICORN, *Understanding Complex Datasets: Data Mining with Matrix Decompositions (Chapman & Hall/CRC Data Mining and Knowledge Discovery Series)*, Chapman & Hall, 2007.

[114] O. SPORNS, D. CHIALVO, M. KAISER, AND C. HILGETAG, *Organization, development and function of complex brain networks*, Trends in Cognitive Sciences, 8 (2004), pp. 418–425.

[115] J. A. C. STERNE, G. D. SMITH, AND D. R. COX, *Sifting the evidence-what's wrong with significance tests?*, British Medical Journal, 322 (2001), pp. 226–231.

[116] L. J. STEWARD, M. D. KENNEDY, B. J. MORRIS, AND J. A. PRATT, *The atypical antipsychotic drug clozapine enhances chronic PCP-induced regulation of prefrontal cortex 5-ht2a receptors*, Neuropharmacology, 47 (2004), pp. 527–537.

[117] G. STRANG, *Computational Science and Engineering*, Wellesley-Cambridge Press, 2008.

[118] S. H. STROGATZ, *Exploring complex networks*, Nature, 410 (2001), pp. 268–276.

[119] J. N. Sutton, M. Athanas, A. Prakash, and M. Lopez, *A workflow to enable intelligent and targeted high throughput label free differential analysis experiments using SIEVE.*

[120] Y. Suzuki and H. F. Nijhout, *Evolution of a polyphenism by genetic accommodation*, Science, 311 (2006), pp. 650–652.

[121] K. Tarassov, V. Messier, C. R. Landry, S. Radinovic, M. M. S. Molina, I. Shames, Y. Malitskaya, J. Vogel, H. Bussey, and S. W. Michnick, *An in vivo map of the yeast protein interactome*, Science, 320 (2008), pp. 1465–1470.

[122] A. Taylor and D. J. Higham, *CONTEST: A Controllable Test Matrix Toolbox for MATLAB*, ACM Transactions on Mathematical Software, 35 (2009), pp. 1–17.

[123] H. C. J. Thode, *Testing for Normality (Statistics: a Series of Textbooks and Monogrphs)*, CRC, 2002.

[124] A. Thomas, R. Cannings, N. A. M. Monk, and C. Cannings, *On the structure of protein-protein interaction networks*, Biochemical Society Transactions, 31 (2003), pp. 1491–1496.

[125] J. Thomas, N. Ramakrishnan, and C. Bailey-Kellogg, *Protein design by sampling an undirected graphical model of residue constraints*, IEEE/ACM Transactions on Computational Biology and Bioinformatics, 6 (2009), pp. 506–516.

[126] A. H. Y. Tong, G. Lesage, G. D. Bader, H. Ding, H. Xu, X. Xin, J. Young, G. F. Berriz, R. L. Brost, M. Chang, Y. Chen,

X. Cheng, G. Chua, H. Friesen, D. S. Goldberg, J. Haynes, C. Humphries, G. He, S. Hussein, L. Ke, N. Krogan, Z. Li, J. N. Levinson, H. Lu, P. Menard, C. Munyana, A. B. Parsons, O. Ryan, R. Tonikian, T. Roberts, A.-M. Sdicu, J. Shapiro, B. Sheikh, B. Suter, S. L. Wong, L. V. Zhang, H. Zhu, C. G. Burd, S. Munro, C. Sander, J. Rine, J. Greenblatt, M. Peter, A. Bretscher, G. Bell, F. P. Roth, G. W. Brown, B. Andrews, H. Bussey, and C. Boone, *Global mapping of the yeast genetic interaction network*, Science, 303 (2004), pp. 808–813.

[127] *UCINET IV Version 1.0 DATASETS*, http://vlado.fmf.uni-lj.si/pub/networks/data/UciNet/UciData.htm#gama.

[128] P. Uetz, L. Giot, G. Cagney, T. A. Mansfield, R. S. Judson, J. R. Knight, D. Lockshon, V. Narayan, M. Srinivasan, P. Pochart, A. Qureshi-Emili, Y. Li, B. Godwin, D. Conover, T. Kalbfleisch, G. Vijayadamodar, M. Yang, M. Johnston, S. Fields, and J. M. Rothberg, *A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae*, Nature, 403 (2000), pp. 623–627.

[129] C. F. Van Loan, *Generalizing the singular value decomposition*, SIAM Journal on Numerical Analysis, 13 (1976), pp. 76–83.

[130] L. Wasserman, *All of Nonparametric Statistics (Springer Texts in Statistics)*, Springer, 2007.

[131] D. S. Watkins, *The Matrix Eigenvalue Problem: GR and Krylov Subspace Methods*, Society for Industrial Mathematics, 1 ed., November 2007.

[132] D. J. Watts and S. H. Strogatz, *Collective dynamics of 'small-world' networks*, Nature, 393 (1998), pp. 440–442.

[133] J. H. Wilkinson, *The Algebraic Eigenvalue Problem (Numerical Mathematics and Scientific Computation)*, Oxford University Press, USA, April 1988.

[134] L. Wilkinson and M. Friendly, *The history of the cluster heat map*, The American Statistician, 63 (2009), pp. 179–184.

[135] H. Wu, A. D. Southam, A. Hines, and M. R. Viant, *High-throughput tissue extraction protocol for NMR- and MS-based metabolomics*, Analytical Biochemistry, 372 (2008), pp. 204–212.

[136] X. Xiao, N. Dawson, L. McIntyre, B. J. Morris, J. A. Pratt, D. G. Watson, and D. J. Higham, *Exploring metabolic pathway disruption in the subchronic phencyclidine model of schizophrenia with the Generalized Singular Value Decomposition*, accepted for BMC Systems Biology (2011).

[137] H. Yu, P. Braun, M. A. Yildirim, I. Lemmens, K. Venkatesan, J. Sahalie, T. Hirozane-Kishikawa, F. Gebreab, N. Li, N. Simonis, T. Hao, J.-F. Rual, A. Dricot, A. Vazquez, R. R. Murray, C. Simon, L. Tardivo, S. Tam, N. Svrzikapa, C. Fan, A.-S. de Smet, A. Motyl, M. E. Hudson, J. Park, X. Xin, M. E. Cusick, T. Moore, C. Boone, M. Snyder, F. P. Roth, A.-L. Barabasi, J. Tavernier, D. E. Hill, and M. Vidal, *High-quality binary protein interaction map of the yeast interactome network*, Science, 322 (2008), pp. 104–110.

[138] L. Zemanova, C. Zhou, and J. Kurths, *Structural and functional clusters of complex brain networks*, Physica D: Nonlinear Phenomena, 224 (2006), pp. 202–212.

[139] J. Zhang, J. Huang, C. Loran, A. M. Zumwalt, A. Prakash, and T. D. McClure, *Metabolomics studies of drug containing urine samples by rapid ultra-high pressure chromatography and high resolution mass spectrometry with a LTQ orbitrap mass spectrometer.* Technical Poster, Thermo Electron Corporation, San Jose, CA 95134, USA.

[140] W. Zhong and P. W. Sternberg, *Genome-wide prediction of C. elegans genetic interactions*, Science, 311 (2006), pp. 1481–1484.

[141] C. Zhou, L. Zemanov, C. C. Hilgetag, and J. Kurths, *Structure-Function Relationship in Complex Brain Networks by Multilevel Modeling*, in Advances in Cognitive Neurodynamics ICCN 2007, Springer Netherlands, 2008, pp. 511–514.