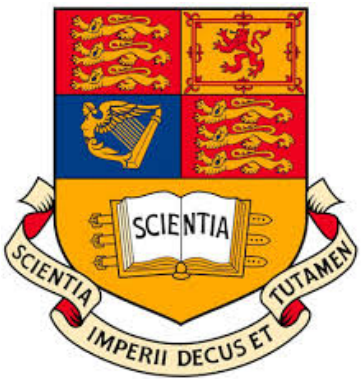


Imperial College London

MEng Individual Project

ON A NEW SIGNATURE THAT QUANTIFIES TOPOLOGICAL STRUCTURE IN BIOLOGICAL AND ECONOMIC NETWORKS

Author:
Răzvan Valentin Marinescu

Supervisors:
Dr. Nataša Pržulj
Prof. Marek Sergot

A thesis submitted in fulfilment of the requirements
for the degree of Master of Engineering
in the
Department of Computing

June 17, 2014

Abstract

Many interesting patterns can be uncovered from network data by studying the neighbourhood of nodes. For example, in protein-protein interaction (PPI) networks it has been shown that the function of a protein is strongly related to its interactions with other proteins [1]. Studying the neighbourhood of nodes can tell us important information not only about the nodes themselves, but also about the network as a whole. However, not much research has been done into studying these neighbourhoods of nodes. The clustering coefficient, one of the few signatures that quantify the topological structure in the neighbourhood of a node, is unable to capture complex patterns that arise in these sub-networks.

Our project defines a novel signature called the *Graphlet Cluster Vector* (GCV) that generalises the clustering coefficient of a node and quantifies the topological structure in the neighbourhood of a node. We apply the GCV signature to economic, protein interaction and metabolic networks and demonstrate its strength by uncovering interesting insights from the data and providing real-world interpretations of our results.

In the economic networks, we show that the structure of the economic network causes fluctuations in the change of crude oil price. Moreover, we also show that for a given country, a relatively sparse network of trading partners is beneficial for its economy. In the PPI networks, we show that the neighbourhood structure of a protein is influenced by the protein's involvement in RNA processing, translation, metabolism or Golgi endosome sorting. In Metabolic networks, we show that the network of interacting partners of an enzyme is affected by its involvement in several cellular processes, organismal systems or diseases. Moreover, we also quantitatively evaluate the novel GCV signature on clustering random networks and test its performance when dealing with noisy and incomplete data.

Acknowledgements

I would like to thank:

- Dr. Nataša Pržulj, for providing me the necessary guidance and motivation throughout the project and for offering me invaluable advice regarding a PhD position
- Vuk Janjić, for his constant support and for explaining me all the details about the PPI, Metabolic, World Trade networks and the corresponding scripts
- Prof. Marek Sergot, for important advice and feedback on how to structure this report
- Dr. Ömer Nebir Yaveroğlu, for explaining me how to conduct the experiments in the evaluation chapter and for providing me several of his `Python` scripts that preprocess network data
- Darren Davis, for explaining me how Canonical Correlation Analysis works and for providing me with his `R` CCA implementation
- My family and friends, for their support and encouragement throughout the project
- Petr Čermák, for his help with LaTeX and typesetting

Contents

List of Figures

Chapter 1

Introduction

In a complex network, studying the neighbourhood of a node is important for understanding the function of that node within the network. For example, vertex neighbourhoods have been used by Schwikowski et al. for predicting protein function in protein-protein Interaction (PPI) networks [1]. A. Hertz and D. de Werra have used node neighbourhoods for graph colouring using tabu search techniques [2]. For the World Wide Web network, query-dependent ranking algorithms calculate neighbourhood graphs of a given web page in order to measure its relevance and quality [3]. On the other hand, local properties of vertices in a network have also been studied by N. Pržulj using graphlets, which are small induced subgraphs of the original network. For each node in the network, a Graphlet Distribution Vector (GDV) can be constructed that captures the local topological structure around the node [4]. However, this GDV signature cannot capture the topological structure in the neighbourhood set of a vertex. This project addresses this issue by defining and analysing a novel signature called the *Graphlet Cluster Vector* (GCV), which will calculate the frequency of graphlets in the neighbourhood of a node. Therefore, the GCV signature will be able to bridge the gap between node neighbourhood analysis and graphlet analysis by combining both approaches.

1.1 Motivation

We developed the novel GCV signature in order to gather insights from two main types of networks: biological and economic. Over the past few decades, major advancements in Genomics and Molecular Research technologies have made available large biological networks of chemical interactions that can help us better understand molecular processes. On the other hand, economic networks have also been produced that track trade flows between countries or cities. Because of the sheer size of the networks, suitable algorithms need to be devised that capture important patterns in the data automatically.

We believe that studying the neighbourhood of nodes in biological or economic networks can yield very interesting insights and correlations that other classical methods cannot capture. For instance, it has been shown that in protein-protein interaction (PPI) networks, proteins that interact in a similar manner with their neighbours will probably have similar functions, even if the proteins are at a large distance from each other in the PPI network [1]. Therefore, studying the interactions of the neighbours of a node can tell us something about the properties of that node itself. We also believe that exploring new ways to analyse biological networks will help us shed a new light on complex biological processes. This could further lead to an increased understanding of diseases such as cancer or cardiovascular disorders that are leading causes of death worldwide [5, 6].

For each node in a given input network, our novel GCV signature will count the frequency of different graphlets in the neighbouring subgraph of the node. Unfortunately, hub nodes (i.e. nodes with a high degree) have a large neighbouring subgraph and computing the GCV

signature for these nodes can easily become infeasible. As a result, the computation needs to be parallelised for large networks with tens of thousands of nodes, such as the PPI networks.

Apart from getting insights from the network data, our GCV signature can also be used to align networks or assess which random model fits the real data best. The GCV signature can also be used in conjunction with other local and global properties of networks, such as the degree distribution, average diameter or node centralities. These network properties have so far been successfully used to identify enzymes or reactions that are crucial for the survival of organisms [7], model drug design trends [8] or modeling the world-wide airport network [9].

Our technique builds upon work done by the Pržulj group, which has developed a signature called *Graphlet Distribution Vector* (GDV) that counts the number of graphlets that a node touches [10]. This has been successfully used to fit random network models to real world networks [11], uncover biological network function [4] and topologically align networks [12]. Our novel *Graphlet Cluster Vector* signature can be seen as a generalisation of the GDV signature, by extending it on the neighbourhood set of a node. Given the different areas where the GDV signature has been successfully applied, we believe the new GCV signature can also be successful at uncovering insights and patterns from the networks we will apply it on.

1.2 Objectives

The project was concerned with exploring the properties of the novel GCV signature and using it for uncovering hidden patterns from the network data. Since the GCV signature is built on the older GDV signature developed by N. Pržulj, we applied previously used statistical techniques such as Canonical Correlation Analysis and Pearson's correlation matrices that have been successfully used with the older GDV signature. Our objectives were to:

1. Implement an algorithm that calculates the GCV signature for every node in a given network.
2. Calculate the GCV signatures for several biological and economic networks as well as random networks that have been generated from these. If the computation is taking too long, parallelise the processing.
3. Use statistical techniques to find out which graphlets from the GCV signature have a behaviour similar with each other.
4. Correlate the GCV signature with functional node annotations.
5. Implement a framework that automatically preprocesses a large number of networks and performs all the statistical experiments on each of them.
6. Identify which network dataset gives the best correlations with the GCV signature. Perform deeper experiments in the chosen dataset.
7. Interpret the results and report on the findings.

While objectives 1 and 2 represented the implementation of the core algorithms in this project, the rest of the objectives were focused on data analysis. In the data analysis part of the project, we tested our methods on a variety of networks, using different GCV normalisation procedures and functional annotations. The main network classes we applied this to are as follows:

- Real networks
    1. Protein-Protein Interaction (PPI) networks
    2. Metabolic networks

3. World Trade networks

- Random networks

1. Erdős-Rényi [13]
2. Erdős-Rényi (with preserved degree distribution)
3. Geometric networks [14]
4. Barabási-Albert (preferential attachment) [15]
5. Stickiness index-based [16]

Optionally, the following extra objectives have also been considered:

- Parallelising the computation for the GCV signature. This is required for large networks such as the PPI networks.
- Implementing a classifier that uses the GCV signature of proteins in PPI networks to predict protein function.
- Using the GCV signature to cluster random network models.

Finally, this work was a research project that had a certain amount of risk associated to it. The GCV is a novel signature that has not been studied before by the scientific community, so we could not predict beforehand how well it performs on our experiments. Throughout the project, we guided our experiments by the signals we got from initial experiments. Nevertheless, when analysing some network data such as the enzyme-based Metabolic networks we hit a dead end multiple times, suggesting that our signature is not suitable for analysing these types of networks.

1.3 Contributions

The main contributions of the project are summarised below:

- development of the mathematical model of the GCV signature followed by the implementation and parallelisation of the algorithm that computes it.
- implementation of algorithms that compute Pearson's correlation matrices and Canonical Correlation Analysis on several classes on networks
- interpretations of the results obtained by the initial experiments and the identification of the World Trade networks (WTNs) as the dataset which offered the best results.
- results in the WTN showing that:
  - Changes in the structure of the WTN are inversely correlated with the changes in the price of crude oil. Since the changes in the network structure happen one year before the changes in oil prices, we believe that the network structure causes the price of crude oil to change. This is one of the main results of the project (section 4.2.1).
  - For a certain country, sparse networks of trading partners are a sign of its economic well-being. On the other hand, dense networks of trading partners are detrimental for its economy (section 4.2.6)
  - The structure of trading partners of Saudi Arabia, a major oil exporter, is influenced by the change in oil price (section 4.2.9).
  - A clustered structure of the trading partners network of a country is correlated with the level of regional integration of the country (section 4.2.3).

  - A trading partner sparsity index score has been computed for a variety of countries for the time period 1962–2010. The index correlates with major economic events such as oil crises, political revolutions, economic reforms or changes in foreign policy (section 4.2.8).

- results in the PPI networks showing that the interaction neighbourhood of a protein is related to the protein's involvement in several processes (section 4.3.4):
  - Ribosome translation
  - RNA processing
  - Metabolism
  - Golgi endosome vacuole sorting
- results in the Metabolic network showing that the interaction neighbourhood of an enzyme is related to the enzyme's involvement in:
  - Cellular Processes (section 4.4.5)
    - Transport and Catabolism
    - Cell communication
    - Cell growth and death
  - Organismal Systems (section 4.4.6)
    - Environmental adaptation
    - Excretory system
    - Digestive system
    - Circulatory system
  - Human diseases (section 4.4.7)
    - Cardiovascular diseases
    - Substance dependence

Although more experiments are needed to confirm some of these results, they all have the potential to be published in a scientific journal. Unfortunately, the given time frame didn't allow us to perform supporting experiments to further certify the results.

1.4 Report Structure

The report structure can be summarised as follows:

- **Chapter 2** provides the background research on Graph theory, Global and Local Network Properties, Random Graphs, Pearson's and Spearman's correlation coefficients and Canonical Correlation Analysis. At the end of the chapter, section 2.7 provides detailed information about the networks analysed.
- **Chapter 3** describes the methodology used for developing the algorithms that compute the GCV signature, the Pearson's Correlation matrices and the Canonical Correlation Analysis.
- **Chapter 4** presents the results of the analysis on three different network classes: World Trade networks, PPI networks and Metabolic networks.
- **Chapter 5** describes the evaluation of the GCV signature on clustering random networks generated using different algorithms.
- **Chapter 6** outlines a summary of the project achievements, a critique of the current approach and future directions.

Chapter 2

Background

2.1 Graphs

Throughout the project we will be concerned with the study of networks represented by simple, undirected graphs. We therefore need to give the following basic definitions about graphs:

Definition 1 *A graph is a pair $G = (V, E)$ composed of a finite set of vertices or nodes V and a set of edges E.*

Definition 2 *An isomorphism f from a graph G to H is a bijective function $f : V(G) \to V(H)$ such that $\forall x, y \in V(G)$ there is an edge of G between x and y if and only if there is an edge of H between $f(x)$ and $f(y)$.*

Definition 3 *An automorphism of a graph G is an isomorphism from G to itself.*

An isomorphism is a function that maps vertices and edges from a graph G to a different graph H, while an automorphism is a function that maps a graph G to itself. Now that graphs have been introduced, we need to give the following definitions that will eventually introduce the concept of an *automorphism orbit*:

Definition 4 *Two graphs G and H are called isomorphic if and only if there exists an isomorphism from G to H.*

Definition 5 *A set of graphs S is called non-isomorphic if and only if there is no isomorphism between any two graphs from S.*

Definition 6 *The automorphisms of a graph G form a group $Aut(G)$ called the automorphism group of G.*

Definition 7 *For a node x of a graph G, the automorphism orbit of x is defined as $Orb(x) = \{y \in V(G) | y = f(x)$ for some $f \in Aut(G)\}$*

The automorphism orbits of a node x in graph G can be intuitively understood as the set of nodes similar to x that can be interchanged with it in an automorphism. This definition is needed later on for the definition of the *Graphlet Degree Vector* (see section 2.3.4).

Definition 8 *A subgraph $H = (V', E')$ of a graph $G = (V, E)$ is a graph such that $V' \subseteq V$ and $E' \subseteq E$*

Definition 9 *Let G be a graph and H be a subgraph of G. H is said to be induced (or full) if, for any pair of vertices x and y of H, xy is an edge of H if and only if xy is an edge of G.*

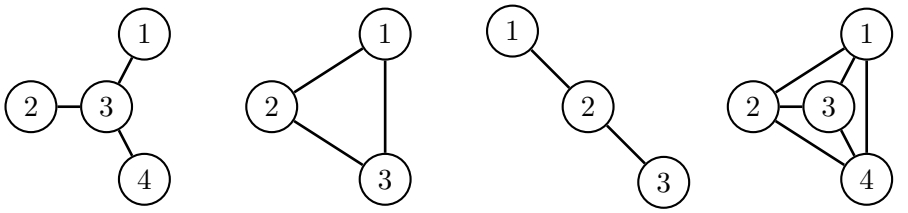
The definitions for non-isomorphic and induced subgraphs are needed later on in order to define what a graphlet is (see section 2.3.2).

2.1.1 Graph terminology

We shall now explain several commonly-used graph types and the terminology used to describe them. These will be used throughout the project when interpreting results from the Pearson GCV correlation matrices. The graph types are as follows:

- cycle S_n: a sequence of n vertices that starts and ends at the same vertex.
- path P_n: a sequence of n vertices.
- clique K_n: a graph of n vertices where every pair of vertices is connected by an edge.
- claw C_n: a graph of n vertices that has one central node and $n-1$ satellite nodes connected to it. The satellite nodes have no edges between them.
- bipartite-graph: a graph whose vertices can be split into two sets U and V such that every edge connects one node from U to a different node from V.

These structures will be used throughout the project in order to group graphlets[1] that have common properties. A few basic graph structures are show in figure 2.1.



(a) Claw (b) Cycle (c) Path (d) Clique

Figure 2.1: From left to right: A claw of 4 nodes (C_4), cycle of 3 nodes (S_3) (or triangle), a path of 3 nodes (P_3), a clique of 4 nodes (K_4).

2.2 Global Network properties

Global network properties give an overall picture of the network, but are unable to capture low-level patterns in the structure of the network. In the following sections we will present a few key global properties such as the *degree distribution*, *clustering coefficient* and the *average path length*.

2.2.1 Degree Distribution

Definition 10 *The degree of a node x in a graph G is the number of edges incident to the node, with loops counted twice.*

[1] small induced subgraphs; they will be defined later on

Definition 11 *The degree distribution $P(k)$ of a graph is the fraction of nodes in the network having degree k.*

Several classes of degree distributions exist, with some most commonly-used ones being:

- Binomial
- Poison
- Power-law
- Exponential

Definition 12 *A random variable X follows the Bionomial distribution with parameters n and p if its probability mass function is given by:*

$$f(k; n, p) = \binom{n}{k} p^k (1-p)^{n-k}$$

Definition 13 *A random variable X follows the Poisson distribution with parameter $\lambda > 0$ if its probability mass function is given by:*

$$f(k; \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Definition 14 *A random variable X follows the Power-law distribution with parameter γ if its probability mass function is given by:*

$$f(k; \lambda) = k^{-\gamma}$$

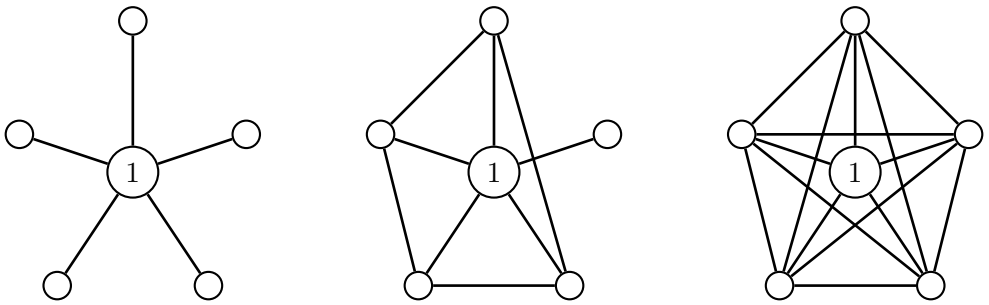
A Power-law degree distribution has a high number of nodes with low degree and a very small number of nodes with high degree, also called hub nodes.

Definition 15 *A random variable X follows the Exponential distribution with parameter $\lambda > 0$ if its probability mass function is given by:*

$$f(k; \lambda) = \lambda e^{-\lambda k}$$

Although many random graphs have a Poison degree distribution, it has been shown that many real networks actually have a Power-law degree distribution instead. Such networks include metabolic networks [17], the Internet [18] and social networks [19].

2.2.2 Clustering Coefficient



(a) (b) (c)

Figure 2.2: The clustering coefficient intuitively describes how densely connected the neighbours of a node are. In the above three scenarios, the clustering coefficient C of node 1 increases from $C = 0$ (image a) to $C = 1$ (image c) as more edges are added in its neighbourhood.

The clustering coefficient is another important property of a graph that is used for data analysis and comparisons. It measures the tendency of nodes to cluster together, which is commonly seen in social networks.

Watts and Strogatz gave the following definition to the clustering coefficient [20]:

Definition 16 *Let G be a graph and n a node that has k_n neighbours. The maximum number of edges between the neighbours of n is $\frac{k_n(k_n-1)}{2}$. The clustering coefficient of node n is then defined as the fraction C_n of these edges that are present in the set of neighbours of n. C_n can also be viewed as the probability of two neighbours of n being connected. This is then averaged against all the nodes in the graph and the final clustering coefficient C of graph G is obtained.*

It has been shown that real networks such as metabolic networks have a high clustering coefficient [21]. Later on we will see that some random networks such as the Erdős-Rényi graphs have a low clustering coefficient when the probability p of connecting two nodes is also low. This makes these models unsuitable for modeling real data.

2.2.3 Average path length

Definition 17 *Let $G = (V, E)$ be a graph and u and v two nodes in V. The distance between u and v is defined as the smallest number of links that have to be traversed to get from u to v.*

Definition 18 *Let $G = (V, E)$ be a graph. The average path length of G is the average distance between any pair of two nodes from V.*

Real networks have been shown to exhibit a small average path length. As a result, random network models that have been developed aimed at producing networks with a small average path length. Moreover, this property has several real-life applications. For example, in a real network such as the World Wide Web, a short average path length will facilitate the exchange of information and reduce operating costs. Similarly, a power grid will suffer less losses if its average path length is minimal.

2.2.4 Spectral Distribution

Spectral network theory explains the topology of a network in terms of the eigenvalues and eigenvectors of matrices associated with the network, such as the adjacency matrix or Laplacian matrix. In order to understand the spectral distribution of a network we first need to define its Laplacian matrix.

Definition 19 *Let $G = (V, E)$ be an unweighted graph and let A be its adjacency matrix. The diagonal degree matrix D of G is a matrix where the diagonal entries are equal to the node degrees, that is $D(x, x) = d_x$, where d_x is the degree of node x. The Laplacian matrix of G is defined as:*

$$L = D - A$$

Definition 20 *Let G be a graph and L its Laplacian matrix. The spectral distribution of G is defined as the ordered vector $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ of eigenvalues of L , where λ_1 is the largest eigenvalue and λ_n the smallest.*

The *Spectral distance* between two graphs G and H is then defined as the Euclidean distance of their spectral distributions. Wilson and Zhou have analysed the spectral distribution of various networks and showed that the spectral distance of two networks is the best measure for classification and clustering purposes [22]. Thorne and Strumpf have also used the spectral distribution for the analysis of the evolution of PPI networks [23].

2.3 Local Network properties

Local network properties capture detailed information about a local region in the network or even about one single node of interest in the network. However, local properties cannot give an overall description of the network in the way the global properties such as the degree distribution or the average path length do. In the next few sections we will present three main types of local network properties: *Node centralities*, *Graphlet Frequency Vector* and *Graphlet Degree Vector*

2.3.1 Node centralities

One commonly used property for measuring the importance of a node in a network is the centrality. Several types of centralities exist:

- *Degree centrality*: Measures the number of links that connect with the node. It is simply defined as the degree of the node.
- *Betweenness centrality*: Quantifies how many shortest paths pass through the node
- *Closeness centrality*: Measures how close the node is to the other nodes in the network.

Definition 21 *Let $G = (V, E)$ be a graph and v a node from V. The Degree centrality of v is defined as the degree of v.*

Definition 22 *Let $G = (V, E)$ be a graph and v a node from V. The Betweenness centrality of v is defined as:*

$$B(v) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

where $\sigma_{st}(v)$ is the number of shortest paths from s to t that pass through node v, while σ_{st} is the total number of shortest paths from s to t.

Definition 23 *Let $G = (V, E)$ be a graph, v a node from V and d_v be the sum of the distances from v to all the other nodes in the network. The Closeness centrality C_v is defined as $C_v = d_v^{-1}$.*

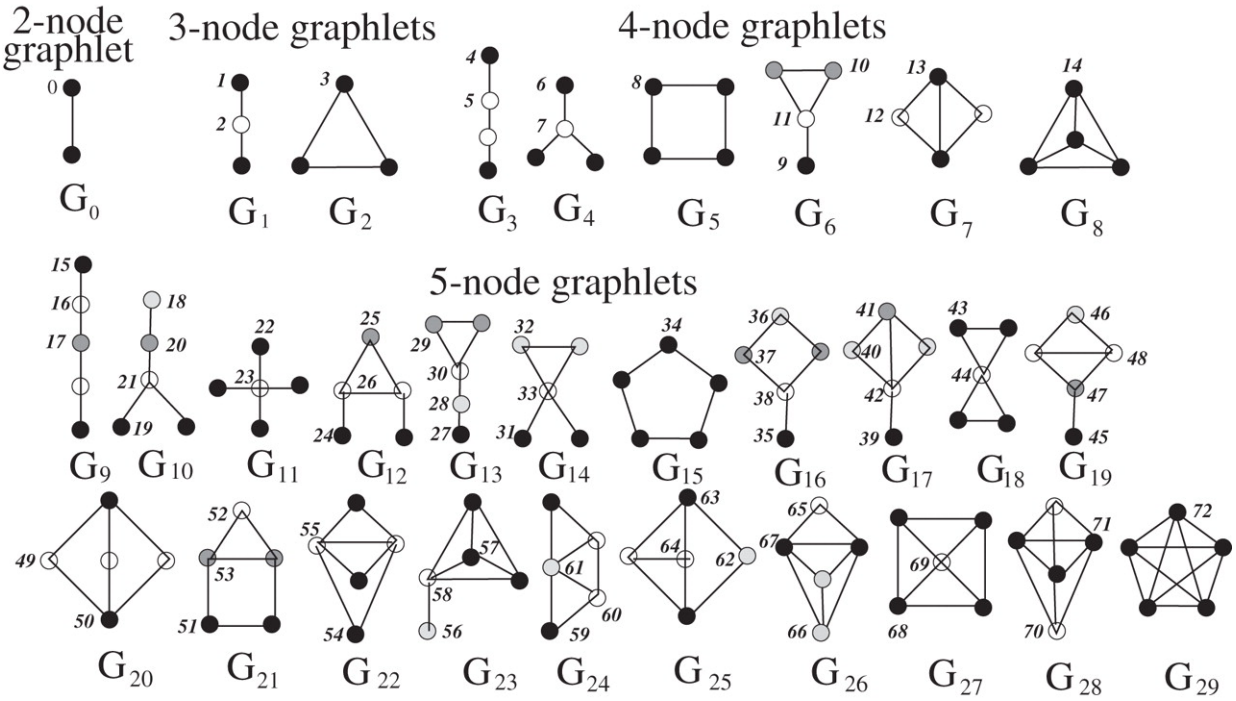


Figure 2.3: Graphlets for sizes of 2, 3, 4 and 5 nodes. They are ordered in groups according to the number of nodes they contain. These are the graphlets that are counted when computing the GDV and GCV metrics. The node labels represent unique automorphism orbits. Source: [4]

2.3.2 Graphlets

Graphlets are small connected non-isomorphic[2] induced[3] subgraphs of a graph. See definitions 5 and 9 for what non-isomorphic and induced graphs are. Figure 2.3 shows all the graphlets of 2,3,4 and 5 nodes. They have been previously used by Nataša et al. [4, 10] for developing signatures such as the Graphlet Degree Vector (GDV) that quantify the local topological structure around a node.

For a given graph G, the *Graphlet Frequency Vector* (GFV) can be calculated by counting the number of distinct graphlets of each type found in G. From here, we normalise the GFV against the total number of graphlets in G to calculate the *Relative Graphlet Frequency Vector* (RGFV).

Definition 24 *Let G_i be the total number of graphlet of type i in graph G. Then the Relative Graphlet Frequency Vector (RGFV) is defined as:*

$$GFV(G) = (F_1(G), F_2(G), ...F_{29}(G)) \tag{2.1}$$

where

$$F_i(G) = -\log\left(\frac{G_i}{\sum_{i=1}^{n} G_i}\right)$$

[2]No two graphs from the set are the same.

[3]An induced subgraph is a subset of the vertices of a graph G together with any edges whose endpoints are in the subset.

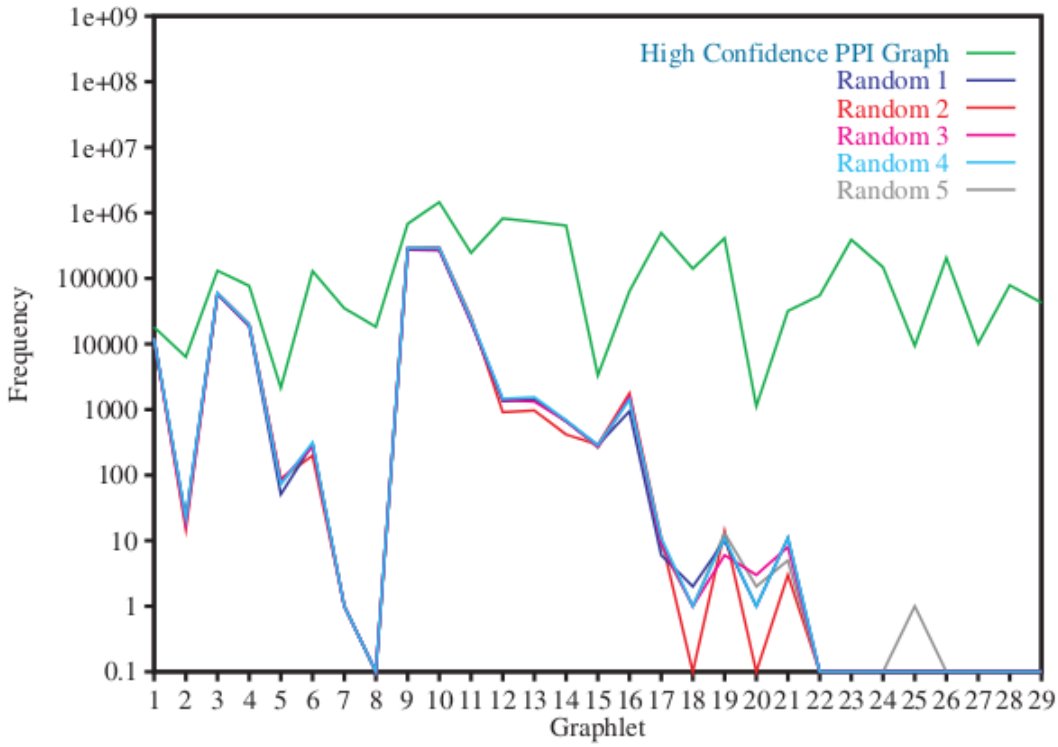


Figure 2.4: Example of *Graphlet Frequency Vectors* for several random graphlets of a PPI network of *S. cerevisiae* (baker's yeast). As one can see from the plot, the GFV signature of the real network is considerably different from the ones of the random networks. The random networks have been generated using the Erdős-Rényi method. Source: [11]

2.3.3 Relative Graphlet Frequency Distance

Using the Graphlet frequencies that have been previously defined, we can now compute a measure of disparity between two graphs by taking pairs of graphlet frequencies for each type and then summing their absolute difference. This is called the *Relative Graphlet Frequency Distance* (RGFD). It is formally defined as follows:

Definition 25 *Let G and H be two graphs and let $F_i(G)$ and $F_i(H)$ be the frequency of the i^{th} graphlet in G and H respectively. The Relative Graphlet Frequency Distance is then defined as:*

$$D = \sum_{i=1}^{n} |F_i(G) - F_i(H)|$$

2.3.4 Graphlet Degree Vectors

In order to explain what a *Graphlet Degree Vector* is, we first need to come back to automorphism orbits. For a node x, its automorphism orbit is the set of nodes similar to x in the graph that could be interchanged with it in an automorphism operation. See definition 7 on page 12 for a formal definition of the automorphism orbits.

Figure 2.3 shows the automorphism orbits for all the nodes in each of the 29 different graphlets. The different automorphism orbits are labelled with numbers ranging from 0 to 72, and the nodes in each graphlet are coloured according to which automorphism orbit it belongs to. Now that we have defined automorphism orbits, we can give a full definition of the *Graphlet Degree Vector* of a node:

Definition 26 *For a node x in a graph G, its Graphlet Degree Vector or GDV is a vector of 73 coordinates, where each coordinate i measures the number of graphlets that touch node x at automorphism orbit i.*

The GDV generalises the degree of a node, which counts the number of edges it touches, into the vector of graphlet degrees, which counts the number of graphlets that the node touches at a particular automorphism orbit. The resulting signature describes the local topology of the node neighbourhood up until a distance of 4 [4].

For a given node x in a graph G, we denote by x_i the i^{th} coordinate of the GDV vector of x. That is, x_i is the number of times x is touched at orbit i.

Definition 27 *The distance $D_i(x, y)$ between the i^{th} automorphism orbits of nodes x and y is defined as:*

$$D_i(x, y) = w_i \frac{|\log(x_i + 1) - \log(y_i + 1)|}{\log(\max(x_i, y_i) + 2)}$$

where $w_i \in [0, 1]$ are weights that normalise orbit dependency [4].

The logarithm function is used because the i^{th} coordinates of the signature vectors of two nodes can differ by several order of magnitude and we do not want the distance measure to be dominated by the larger values [4]. We also add 1 to u_i and v_i in order to prevent the logarithm from going to $-\infty$. We add 2 in the denominator of the formula in order to prevent it from being infinite or 0 [4].

Definition 28 *Given two GDVs of nodes x and y, the distance $D(x, y)$ between the GDVs of x and y is defined as:*

$$D(x, y) = \frac{\sum_{i=0}^{72} D_i}{\sum_{i=0}^{72} w_i}$$

The distance measure given in definition 28 is in the $[0, 1]$ range, where a distance of 0 means that the two GDVs are identical.

Definition 29 *The signature similarity between nodes x and y is defined as:*

$$S(x, y) = 1 - D(x, y)$$

The signature similarity gives a measure of how similar the topological structure around two nodes is. This is very useful because it can be easily applied to practical problems. For instance, it has been shown that the function of a protein can be predicted from its interactions [24]. Therefore, if a protein x is known to have a particular function and one would like to annotate a different, unknown protein y, one can transfer the function from x to y if their GDV signature similarity is high.

2.3.5 Graphlet Degree Distributions

The Degree Distribution of a network calculates the number of nodes touching k edges for each value of k. However, we can generalise this concept by looking at the 73 automorphism orbits (see figure 2.3) and counting the number of nodes that touch a particular graphlet at a particular orbit. Finally, we get a spectrum of 73 *Graphlet Degree Distributions (GDDs)* measuring local properties of a network.

We are now trying to compare the spectrum of 73 Graphlet Degree Distributions belonging to a graph G to the ones corresponding to another graph H. There might be several ways to perform this, but we will present the method used by N. Pržulj et al. in 2006 [10].

Definition 30 *Let G be a graph and let $d_G^j(k)$ be a sample distribution of the number of nodes in G touching orbit j ($j = 1 - 73$) k times. d_G^j represents the j^{th} graphlet degree distribution (GDD). The scaled j^{th} graphlet degree distribution $S_G^j(k)$ of G is then defined as:*

$$S_G^j(k) = \frac{d_G^j(k)}{k}$$

The reason for scaling d_G^j is because most of the information is retained in the lower degrees, whereas the high degrees mostly contain noise [10]. Afterwards, the distribution is normalised against its total area:

$$T_G^j = \sum_{k=1}^{\infty} S_G^j(k)$$

giving the normalised distribution:

$$N_G^j(k) = \frac{S_G^j(k)}{T_G^j}.$$

The reason why we are normalising the distribution is because a large network would have a lot of nodes that potentially touch orbit j and therefore a large area under the curve. Normalising it would make large and small biological networks comparable in terms of their GDD.

Definition 31 *For two graphs G and H and an orbit j, we define the distance $D^j(G, H)$ between their normalised j^{th} distributions as:*

$$D^j(G, H) = \sqrt{\sum_{k=1}^{\infty} [N_G^j(k) - N_H^j(k)]^2}$$

The distance $D^j(G, H)$ is between 0 and 1, where 0 means that G and H have the same GDD for automorphism orbit j. Now that we have a measure of distance between two graphs G and H, we need to invert the this distance in order to get the j^{th} *GDD agreement*:

Definition 32 *For two graphs G and H and an orbit j, we define the j^{th} GDD agreement between their normalised j^{th} distributions as:*

$$A^j(G, H) = 1 - D^j(G, H)$$

Moreover, the overall GDD agreement between the two networks G and H is defined as the arithmetic mean of $A^j(G, H)$ over all j:

$$A(G, H) = \frac{1}{73} \sum_{j=0}^{72} A^j(G, H)$$

The GDD agreement is like the GDV signature similarity of two nodes x and y, but this time for the overall graphs G and H. This measure can be used to compare different networks or even evaluate which random graphs best model the real data.

2.4 Random graphs

Random graphs are graphs that are usually generated using a random process. They are used in data analysis for comparing or aligning them against real networks. They can model the behaviour of real-world networks, such as the World Wide Web or PPI networks. Random graph models have been successfully used in various biological settings, such as: Network motifs [25], De-noising of protein-protein interaction network data [26] or guiding biological experiments [27]. In the next sections we will present several types of random graphs along with their properties.

2.4.1 Erdős-Rényi graphs

The work on random graph models started from the influential publications of Erdős and Rényi in the 1950s and 1960s. Edgar Gilbert also published a similar model later on. Erdős and Rényi described the $G_{n,m}$ model [13], while Gilbert described the $G_{n,p}$ model [28]. These two methods can be described as follows:

$\mathbf{G_{n,p}}$: We start with n disconnected nodes and a given probability p. We then go through every pair of nodes and connect them with probability p.

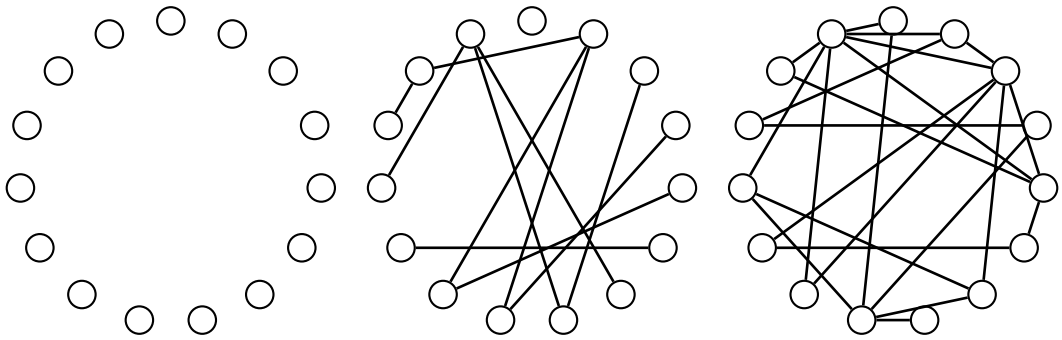
$\mathbf{G_{n,m}}$: We start with n disconnected nodes and a target number of edges m. Afterwards, we randomly select m pairs of nodes and connect them.

Although these networks are very easy to generate, it was later found that real networks have a structure that is different from the Erdős-Rényi graphs. More precisely, they have a different degree distribution and a low clustering coefficient. On the other hand, some real networks have a power-law degree distribution. For the Erdős-Rényi $G_{n,p}$ graph, the degree distribution is binomial:

$$P(k) = \binom{n-1}{k} p^k (1-p)^{n-1-k} \tag{2.2}$$

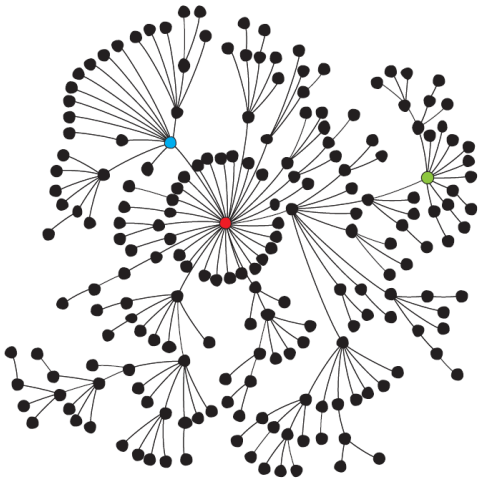
which can be approximated with a Poisson distribution for a large n:

$$P(k) = \frac{z^k * e^{-z}}{k!} \tag{2.3}$$

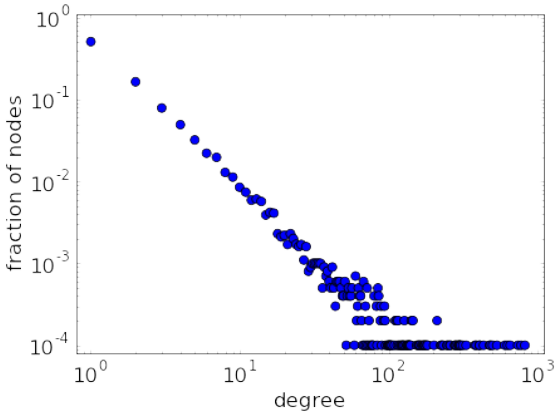


(a) $p = 0$ (b) $p = 0.1$ (c) $p = 0.2$

Figure 2.5: Example of three Erdős-Rényi random graphs generated using the $G_{n,p}$ method. The three different graphs differ with respect to the probability p of connecting one pair of nodes: (a) $p = 0$ – the graph is completely disconnected. (b) $p = 0.1$ – the graph is sparsely connected because the probability p is low (c) $p = 0.2$ – the graph becomes more dense because of the increase of p .



(a) Example of a scale-free network. Note the large number of nodes of small degree at the periphery of the network, while the number of hub nodes is very small.



(b) The power-law degree distribution of a scale-free network. As the degree of the nodes gets larger, the fraction of nodes decreases exponentially. Notice the logarithmic scale on the Y axis. It has been observed that many real networks exhibit a power-law degree distribution [17, 18, 19].

Figure 2.6: Scale-free network (a) and power-law degree distribution (b)

2.4.2 Erdős-Rényi with preserved degree distribution

As we have previously seen, the degree distribution of an Erdős-Rényi graph does not match the real data. We will now present a method for constructing an Erdős-Rényi network that preserves the degree distribution of a real network.

We start with n disconnected nodes. Each node is assigned a number of stubs according to the degree distribution of the real network that is being modelled. A stub is simply a slot belonging to a particular node from where an edge can be connected. Afterwards, edges are created only between random pairs of nodes with available stubs. After an edge is created, the number of stubs left available at the nodes that were just connected is decreased by one. Moreover, edges between one node and itself are not allowed.

This "stubs method" allows us to create Erdős-Rényi networks that have a power-law degree distribution and a small average path length. Unfortunately, they still have a low clustering coefficient.

2.4.3 Scale-free networks

Scale-free networks are networks that normally exhibit a power-law degree distribution (see fig 2.6). That is, $P(k) = k^{-\gamma}$, where $P(k)$ is the fraction of nodes having degree k. It is currently believed that many networks such as the World Wide Web, social networks or biological networks exhibit scale-free properties with a power-law degree distribution.

Barabási-Albert model

There are several proposed ways in which scale-free networks can be generated. The Barabási-Albert model is one such technique that uses the *preferential attachment* mechanism, with which nodes of high degree have a high probability of receiving even more connections.

In order to construct a network using the Barabási-Albert method, we start with an initial connected network of m_0 nodes. New nodes are consecutively added to the network one at a time. Each one of them is connected to $m \leq m_0$ target nodes with a probability that is

proportional to the degree of the target nodes. Formally, the probability p_i that the new node is connected to node i is:

$$p_i = \frac{k_i}{\sum_j k_j} \tag{2.4}$$

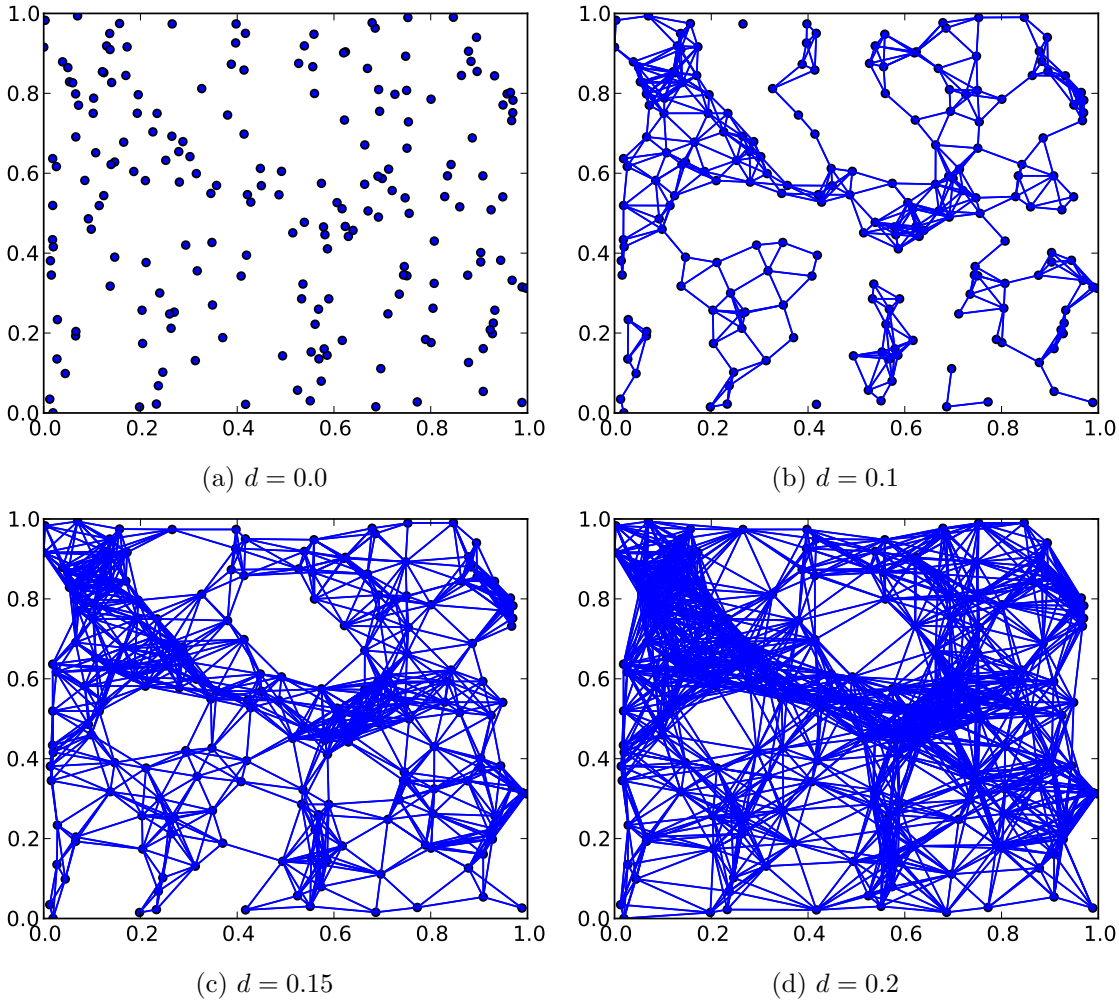
where k_i is the degree of node i, while the sum is over all the nodes j that already existed in the network when the new node is added. Because of the preferential mechanism, heavily linked nodes (also called hub nodes) tend to quickly accumulate links, whereas nodes with a low degree are unlikely to be chosen. It has also been shown that the starting network heavily influences the properties of the resulting network [29].

2.4.4 Geometric graphs

Geometric graphs are generated by fixing a certain metric space and using metrics such as geometric distance or radius to connect edges together. A metric space is a space that has a distance norm associated to it such as: the Euclidean distance, Chessboard distance or Manhattan distance.

Such a network is generated in the following manner:

1. Choose a metric space and place nodes within the space using a uniform random distribution.
2. If any nodes are within distance d from each other, then connect them with an edge.
3. d needs to be chosen so that the end number of edges matches the network that is modelled.



(a) $d = 0.0$ (b) $d = 0.1$

(c) $d = 0.15$ (d) $d = 0.2$

Figure 2.7: Geometric random networks built using different values for the distance parameter d, starting from $d = 0$. Initially, nodes are distributed randomly across a metric space. When $d = 0$ in graph (a), the nodes are all disconnected from each other. As d increases in graphs (b – d), the number of connections in the network also increases proportionally. When using a Geometric network to model a real network, one would normally use a value of d that would yield a similar number of edges as the real network.

2.4.5 Stickiness index-based graphs

Pržulj et al. have proposed in 2006 a simple random graph model that inserts a connection according to the degree or *stickiness* of the nodes involved [16]. This model has been inspired from analysing protein-protein interactions and is based on two assumptions:

1. A node with a high degree or *stickiness* represents a protein that has many binding domains and/or its binding domains are commonly involved in interactions.

2. A pair of proteins is more likely to interact, or share complementary binding domains, if they both have a high *stickiness*. On the other hand, if one or both of them have a low *stickiness* index, they are less likely to interact. Thus, the product of their stickiness values can be used as the probability of connecting the nodes.

Considering the above assumptions, a stickiness based random graph can be constructed as follows:

Comparison of real networks versus randomly generated networks

Model	Degree Distribution	Clustering coefficient	Average path length
Real networks	Power-law	High	Small
Erdős-Rényi	Poisson	Low	Large (for small p)
Erdős-Rényi - DD	Power-law	Low	Small
Barabási-Albert	Power-law	Low	Small
Geometric (uniform)	Poisson	High	Small
Stickiness based	Power-law	High	Small

Table 2.1: As one can observe from the table above, some of the models such as Erdős-Rényi are not suitable for modeling real networks according to these metrics. On the other hand, the Stickiness based random graph satisfies all the three criteria. Nevertheless, other network properties might exist for which the Stickiness based random network does not match the corresponding real network.

1. We start with a network of n nodes each having a degree deg_i sampled from a degree distribution of our choice
2. For each node i, we compute the stickiness index $\theta_i = deg_i / \sqrt{\sum_{j=1}^{N} deg_j}$. Note that $0 \leq \theta_i \leq 1$
3. For each pair of nodes (i, j), we connect them with probability $\theta_i \theta_j$

2.4.6 Random graph Comparisons

Now that we have presented a few commonly used random graph generating methods, we would like to compare them in terms of their underlying properties. As can be clearly seen in table 2.1, real networks normally have a power-law degree distribution, high clustering coefficient and a small average path length. In terms of degree distribution, only Erdős-Rényi (with a preserved degree distribution), Barabási-Albert and Stickiness-based random networks have a power-law degree distribution, which is found in real networks. However, it must be noted that although most of the real networks have a power-law degree distribution, this subject is still a matter of research. For example, it has been shown that the Interactome network can be better modelled with a Geometric network that has a Poisson degree distribution [11].

On the other hand, only the Geometric and the Stickiness based models have a high clustering coefficient. This is again something which has been observed in most of the real networks such as social networks or biological networks. Finally, most of the networks have a small average path length. It can therefore be noted that the Stickiness-based network is the most successful at modeling real-world phenomena with respect to these three properties. However, there might be other network properties, such as various node centralities [30] or *Relative Graphlet Frequency Agreement*, with can also be employed to assess the suitability of the random networks. Identifying which of these properties can best compare various types of networks is still an open problem in Network Analysis.

2.5 Measuring Correlation

In sections 2.3.3 and 2.3.5 we presented two main methods for calculating how closely two GDV vectors match: *Relative Graphlet Frequency Distance* and *Graphlet Degree Distribution Agreement*. However, other methods also exist that use correlation coefficients such as *Pearson's product-movement correlation coefficient* or *Spearman's rank correlation coefficient*. This section presents correlation techniques that can be used for GDV comparisons.

2.5.1 Pearson's product-movement correlation coefficient

Given two random variables X and Y from a population, *Pearson's correlation coefficient* or sometimes called *Pearson's population correlation coefficient* is defined as the ratio between the covariance of X and Y and the product of their standard deviation. It was introduced by Karl Pearson and it is based on a similar idea by Francis Galton in 1880 [31, 32].

Definition 33 *The Pearson's product-movement correlation coefficient $\rho_{X,Y}$ between random variables X and Y is defined as:*

$$\rho_{X,Y} = \frac{\sigma_{XY}}{\sigma_X \sigma_Y} = \frac{E[X - \mu_X]E[Y - \mu_Y]}{\sigma_X \sigma_Y}$$

where σ_{XY} is the covariance of X and Y, while σ_X, μ_X and σ_Y, μ_Y are the standard deviation and the expectation of X and Y respectively.

Pearson's correlation coefficient can also be applied to a sample from a given population, in which case it is called the *sample Pearson's correlation coefficient* and is commonly denoted by r. This can be calculated by using sample estimators for the covariance and standard deviation in the formula above.

Definition 34 *The sample Pearson's product-movement correlation coefficient $r_{X,Y}$ between population samples X and Y is defined as:*

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2}\sqrt{\sum_{i=1}^{n}\left(Y_i - \bar{Y}\right)^2}} \tag{2.5}$$

where σ_{XY} is the covariance of X and Y, while σ_X, μ_X and σ_Y, μ_Y are the standard deviation and the expectation of X and Y respectively.

The values of both the sample and the population variants of Pearson's correlation coefficients are between -1 and 1. Sample data points that have exactly 1 or -1 as their correlation coefficient will lie on a straight line. Moreover, Pearson's correlation coefficient is symmetric because:

$$\rho(X, Y) = \rho(Y, X)$$

where $\rho(X, Y)$ is defined as the correlation between random variables X and Y.

Pearson's distance

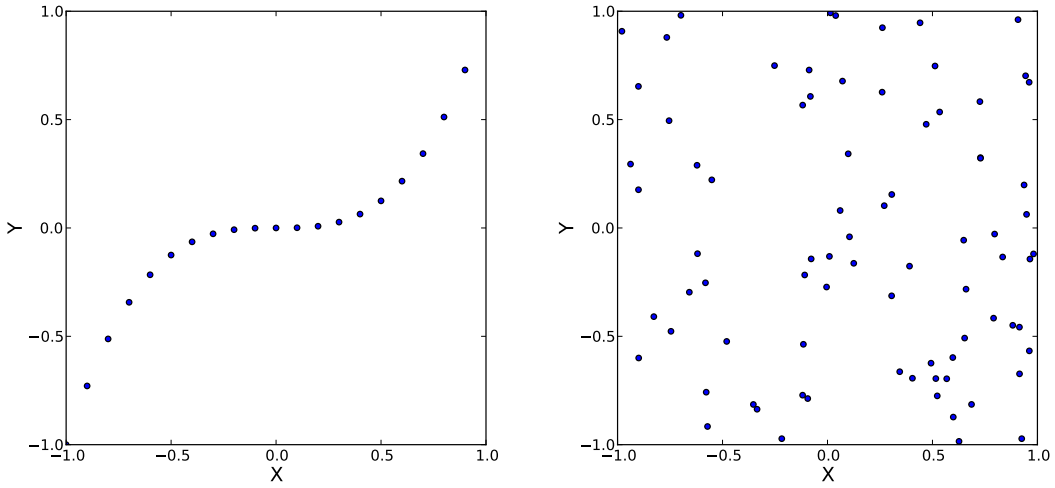
Given a correlation coefficient $\rho_{X,Y}$, a distance metric called the *Pearson's distance* can be derived as follows [33]:

$$d_{X,Y} = 1 - \rho_{X,Y}$$

It should be noted that because Pearson's correlation coefficient lies between -1 and 1, then the Pearson's distance will have a value between 0 and 2.

Interpretation

Several researchers have provided guidelines into how to interpret the size of the correlation coefficient [34] (equation 2.5). However, interpretation is highly dependent on the context of the problem. For example, a correlation of 0.8 might be low if one verifies physical laws using measurements made with high-precision instruments, but it might be considered high when applied to the analysis of social networks, because of underlying hidden factors.



(a)

Pearson correlation:	0.91
p-value:	0.0
Spearman correlation:	1.0
p-value:	0.0

(b)

Pearson correlation:	0.08
p-value:	0.44
Spearman correlation:	0.11
p-value:	0.31

Figure 2.8: Spearman's correlation coefficient is measuring how well the dependence between two variables X and Y can be modelled using a monotonic function. A Spearman's correlation of 1 can result even when the data points are not linear (subfigure a), as long as they are monotonically related. For the same dataset, Pearson's correlation coefficient is 0.91. When the data points are evenly spread (subfigure b), both the Pearson's and the Spearman's correlation coefficients will be low (0.08 and 0.11 respectively) and their p-value will be high (0.44 and 0.31), suggesting that the correlation is not statistically significant.

2.5.2 Spearman's rank correlation coefficient

The Spearman's rank correlation coefficient or Spearman's rho, named after Charles Spearman, is a non-parametric estimator of the statistical dependence of two random variables [35]. It intuitively measures how well the dependence between two variables can be measured using a monotonic function. It is normally defined as follows:

Definition 35 *Let X and Y be two population samples and let x_i and y_i be the ranks of each of the data points in X and Y. The Spearman's rank correlation coefficient is defined as the Pearson's correlation coefficient of the ranks x_i, y_i of the data points* [36].

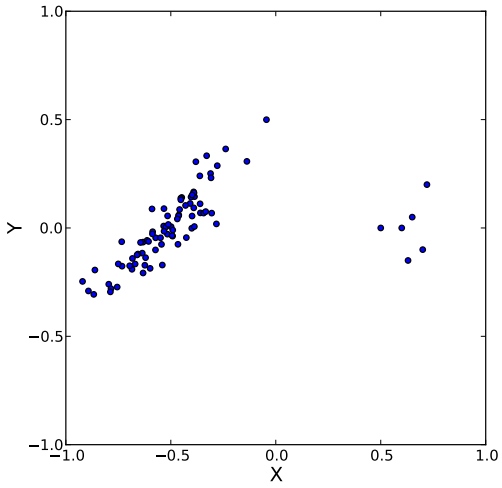
The calculation of the ranks is best illustrated in table 2.2. After the rank x_i, y_i of each data point is calculated, the Spearman's correlation coefficient is computed using the formula for the Pearson's correlation coefficient.

Spearman's correlation coefficient is considered non-parametric in the sense that one does not need to know any prior on the X and Y random variables, as it does not require knowledge (i.e. the parameters) of the joint probability distribution of X and Y.

2.5.3 Computing the GDV correlation matrix of a network

We can use the Pearson's or Spearman's correlation coefficient previously described to compute the *GDV correlation matrix* for a given network in the following manner:

1. We compute the 73-element Graphlet Degree Vector (GDV) for every node in the input network



(a)

Pearson correlation:	0.55
p-value:	0.0
Spearman correlation:	0.81
p-value:	0.0

Figure 2.9: Spearman's rank correlation coefficient is less sensitive to outliers than Pearson's correlation coefficient, because each data point is first projected to its rank. For the above dataset, the Pearson's correlation coefficient is only 0.55, while Spearman's correlation coefficient is 0.81. Both correlations are statistically significant, since their p-values are below 0.05.

2. We then construct samples $S_i, i \in 1, 2, 3, .., 73$ containing all the frequencies of the orbits of type i found in the GDVs of the nodes.

3. We compute the Pearson's (or Spearman's) correlation coefficient of each pair of samples (S_i, S_j) and we put them in the 73x73 correlation matrix C_{ij}.

The newly obtained graphlet correlation matrix will be symmetric with respect to the main diagonal, as Pearson's correlation coefficient is also symmetric. In order to display such a matrix, we use a heat map, with blue representing a correlation of -1 and red representing a correlation of 1.

Given two matrices from two different networks G and H, we then calculate the *Pearson's correlation matrix distance* between them by performing pairwise-subtractions of the elements.

Variable X_i	Position	Rank
0.3	1	1
0.6	2	2
1.2	4	$\frac{4+5}{2}$
1.2	5	$\frac{4+5}{2}$
0.8	3	3
1.9	6	1

Table 2.2: Computation of Spearman's ranks for a dataset of 6 samples. The data is initially sorted in ascending order. If the data point is unique, then the rank is simply the position in the ordered list. Otherwise, the rank is computed as the average of the positions of all the data points with the same value.

The *Pearson's correlation matrix distance* is also referred to as the *Graphlet correlation distance* in the literature [37].

Definition 36 *Let G and H be two graphs and G', H' their Pearson's correlation matrices. The Pearson's correlation matrix distance or Graphlet correlation distance between G and H is then defined as:*

$$D(G,H) = \sum_{i,j} \left(G'(i,j) - H'(i,j)\right)^2 \tag{2.6}$$

Note that the computed distance $D(G,H)$ is always greater than 0. We are now able to define the *Graphlet Correlation Matrix Agreement*, which measures how similar two graphs are with respect to the GDV signatures.

Definition 37 *The* Graphlet Correlation Matrix Agreement *between two graphs G and H is defined as:*

$$Agreement = 1 - D(G,H) \tag{2.7}$$

One advantage of using the *Graphlet Correlation Matrix Agreement* instead of the *GDD agreement* previously defined is that it has been shown to be more robust to noise in the network data [38].

2.5.4 Hierarchical clustering

When analysing the GDV correlation matrix of a network, we are interested to find out how graphlets group together with the rest of the graphlets according to their correlation. These groups of graphlets can be easily identified if we use *hierarchical clustering*, which is a clustering method that builds a dendogram of the graphlets according to a distance function.

The two main types of strategies for hierarchical clustering are:

- Agglomerative (bottom-up): Each observation starts in its own cluster. At each step, the clusters with the smallest distance between each other are merged.
- Divisive (top-down): All observations start in one cluster. At each step, the clusters are split recursively.

Distance metric

In order to perform *hierarchical clustering*, a distance metric has to be defined. Some commonly-used metrics are:

- Euclidean distance: $\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
- generalised p-norm: $\|a-b\|_p = (\sum_i |a_i - b_i|^p)^{\frac{1}{p}}$
- Mahalanobis distance: $\|a-b\| = \sqrt{(a-b)S^{-1}(a-b)}$, where S is the covariance matrix

Linkage criteria

The linkage criteria defines the distance between the sets of data points as a function of the distances between the data points themselves.

Some commonly used criteria for the linkage between two sets A and B are [39]:

- Complete linkage clustering: $max\{d(a,b)|a \in A, b \in B\}$
- Single linkage clustering: $min\{d(a,b)|a \in A, b \in B\}$
- Average linkage clustering [40]: $max\{d(a,b)|a \in A, b \in B\}$
- Centroid linkage clustering: $||c_A - c_B||$ where c_A and c_B are the centroids of clusters A and B.

2.6 Canonical Correlation Analysis

Canonical Correlation Analysis is a statistical method of analysing interdependence between two random variables X and Y. The method was first introduced in 1936 by Harold Hotelling [41] and it has been used for analysing and interpreting data in various fields including Psychology [42], Marketing [43] and Operations Research [44].

Given two random variables X and Y and a set of vector weights a_1 and b_1, let $u_1 = Xb_1$ and $t_1 = Ya_1$. Canonical Correlation Analysis (CCA) aims to find the weights a_1 and b_1 such that the correlation $\rho = r(t_1, u_1)$ is maximised. In this case, u_1 and t_1 are called the first canonical variates.

The CCA process can be repeated again in order to find a second pair of canonical variates u_2 and t_2, with the additional condition that they are orthogonal to the first set of canonical variates u_1 and t_1. Thus, the second stage of the canonical correlation problem can be stated as follows:

Choose a_2,b_2 to maximise

$$r(t_2, u_2) = r(Ya_2, Xb_2) \tag{2.8}$$

such that

$$r(t_1, t_2) = 0 \text{ and } r(u_1, u_2) = 0$$

This procedure can be repeatedly applied, although at each iteration the amount of correlation that we can achieve is decreasing. The reason for this is because each subsequent problem contains one extra orthogonality constraint compared to the previous one. The number of "stages" to the canonical correlation problem depends on the number of variables. If p is the number of X variables and q is the number of Y variables, then the maximum number of canonical variates that can be computed is $min(p, q)$.

2.6.1 Derivation of Canonical Correlation Analysis

We first define the correlation matrix R as:

$$R = \begin{pmatrix} R_{YY} & R_{YX} \\ R_{XY} & R_{XX} \end{pmatrix}$$

where R_{XY} is the correlation matrix between X and Y, while R_{XX} and R_{YY} are the correlation matrices of X and Y.

We find that solving the problem in matrix form will in fact give the solution to all stages of the problem. Dropping the subscripts on variates $u = Xb$ and $t = Ya$, we restate the problem as follows:

Choose a, b to maximise

$$r(t, u) = \frac{cov(t, u)}{\sqrt{var(t)var(u)}} \tag{2.9}$$

The numerator of the objective function in equation (2.9) is simply given by:

$$Cov(t, u) = \frac{[t'u]}{n-1} = \frac{a'Y'Xb}{n-1} = a'R_{YX}b$$

By standardising t and u, we effectively eliminate the denominator from the objective function in equation (2.9). Note that setting $var(t) = 1$ is equivalent to the following:

$$var(t) = 1$$
$$\implies \frac{[t't]}{n-1} = 1$$

$$\implies \frac{[a'Y'YA]}{n-1} = 1$$
$$\implies a'R_{YY}a = 1$$

Similarly, setting $var(u) = 1$ is the same as setting $b'R_{XX}b = 1$. Imposing these constraints, the problem becomes:

Choose a, b to maximise

$$a'R_{XX}b$$

subject to

$$a'R_{YY}a = 1 \text{ and } b'R_{XX}b = 1 \tag{2.10}$$

This constrained maximisation problem can be solved by using Lagrange multipliers and solving the first-order conditions. Using $\alpha/2$ and $\beta/2$ as Lagrange multipliers, the Lagrangian function is then given by:

$$L = a'R_{YX}b - \frac{\alpha}{2}(a'R_{YY}a - 1) - \frac{\beta}{2}(b'R_{XX}b - 1) \tag{2.11}$$

Differentiating with respect to a and b and setting the results equal to zero gives the first-order necessary conditions:

$$\frac{\partial L}{\partial a} = 0 \implies R_{YX}b - \alpha R_{YY}a = 0 \tag{2.12}$$
$$\frac{\partial L}{\partial b} = 0 \implies R_{XY}a - \beta R_{XX}b = 0 \tag{2.13}$$

Taking the expression in equation (2.12) and premultiplying by a' yields:

$$a'R_{YX}b - \alpha(a'R_{YY}a) = 0$$

which implies that $\alpha = r(t, u)$, the canonical correlation, because $a'R_{YY}a = 1$ under the scaling constraints we have imposed for this problem. Similarly, taking equation (2.13) and premultiplying by b' yields $\beta = r(t, u)$, which means that $\alpha = \beta$.

Now that the values of α and β are known, we can substitute into equations (2.12) and (2.13) and solve the expressions for either a or b.

Suppose we choose to solve for b. We use equation (2.12) to write a as a function of b as follows:

$$a = \frac{1}{r(t, u)}R_{YY}^{-1}R_{YX}b \tag{2.14}$$

We then substitute the right-hand side of equation (2.14) above for a in equation (2.13) and solve for b. The result is:

$$R_{XY}\left(\frac{1}{r(t, u)}R_{YY}^{-1}R_{YX}b\right) = r(t, u)R_{XX}b \tag{2.15}$$

Premultiplying by R_{XX}^{-1} and mutiplying through by $r(t, u)$ gives:

$$[R_{XX}^{-1}R_{XY}R_{YY}^{-1}R_{YX}]b = r^2(t, u)b \tag{2.16}$$

Equation (2.16) is an eigenvector-eigenvalue problem. The vector b is the first eigenvector of the matrix $R_{XX}^{-1}R_{XY}R_{YY}^{-1}R_{YX}$. The proportionality constant, which is the eigenvalue corresponding to b, is the squared canonical correlation $r^2(t, u)$. Although we will not prove this in the report, the structure of the canonical correlation problem ensures that the eigenvalues are both real and non-negative [45].

We can now find a by substituting b into equation (2.14). We also find that a is the first eigenvector of the matrix $R_{YY}^{-1}R_{YX}R_{XX}^{-1}R_{XY}$. The first eigenvalue is again the squared canonical correlation.

2.6.2 Canonical Loadings

To facilitate interpretation, it is helpful to look at canonical loadings, which are correlations between original variables and their corresponding canonical variates. The correlations between X and u, which we denote f, are given by:

$$f = \frac{1}{n-1}X'u = \frac{1}{n-1}X'(Xb) = R_{XX}b \tag{2.17}$$

Similarly, the correlations between Y and t, denoted g are given by:

$$g = \frac{1}{n-1}Y't = \frac{1}{n-1}Y'(Ya) = R_{YY}a \tag{2.18}$$

2.6.3 Canonical Cross-Loadings

A slightly different concept is given by canonical cross-loadings, which are the correlations between original variables and the opposite canonical variates. The correlations between X and t, which we denote h, are given by:

$$h = \frac{1}{n-1}X't = \frac{1}{n-1}X'(Ya) = R_{XY}a \tag{2.19}$$

The cross-loadings between Y and u, denoted j are given by:

$$j = \frac{1}{n-1}Y'u = \frac{1}{n-1}Y'(Xb) = R_{YX}b \tag{2.20}$$

In our project, when we present CCA results we only use canonical cross-loadings, because we are interested to find out how each element from one variate correlates with the elements from the opposite variate.

2.6.4 Interpretation of Canonical Correlation Results

The weights a and b that maximise the correlation $\rho = corr(Xa, Yb)$ can be easily interpreted in the following manner:

- If two values a_i, b_j have the same sign it means that variables X_i, Y_j are positively correlated. Similarly, if the values of a_i, b_j have different signs then it means that the variables are negatively correlated.
- A higher absolute value of a_i and b_j means that variables X_i, Y_j show a stronger correlation. Similarly, if the absolute value of a_i and b_j is close to zero then it shows that variables X_i, Y_j show a weak and insignificant correlation.
- If the weight vectors a and b are multiplied by scalars α and β respectively, then the resulting correlation $\rho' = corr(\alpha aX, \beta bY)$ is still the same as the original correlation between vectors a and b, that is $\rho = corr(Xa, Yb)$.

Note that in this report, when we say that two elements x_i and y_i of vectors X and Y correlate positively or have a positive correlation with respect to each other, it means that they have the same sign. Similarly, x_i and y_i will correlate negatively if they have opposite signs.

2.7 Networks analysed

Throughout the project we will be analysing several classes of networks:

- Protein-Protein Interaction (PPI) networks
- Metabolic networks
- World Trade networks

In order to perform Canonical Correlation Analysis, we have also used annotations, which are labels that offer information about each node in the graphs. For a country in the World Trade network (WTN), the annotations are financial indicators such as GDP per capita. On the other hand, for a protein in the PPI network these are properties such as "RNA transcription" or "energy production" that describe the function of the protein. Each of these networks and their annotations are described in detail in the following sections.

2.7.1 Protein-Protein Interaction networks

The Protein-Protein Interaction networks, or PPI networks are mainly represented by a graph where the nodes are proteins and the edges are interactions between proteins. These interactions are normally captured using technologies such as *Yeast two-hybrid screening* [46] or *affinity purification mass spectrometry* (AP-MS) [47, 48].

The source of our PPI networks is the *Biological General Repository for Interaction Datasets* (BioGRID). Throughout the project we have mainly focused on the Human PPI network, although some of our experiments have also been performed on the PPI networks of other model organisms such as C. elegans(worm), D. melanogaster(fruit fly), E. coli(bacteria), M. musculus(mouse) and S. cerevisiae(baker's yeast). We have done this in order to find out whether our results are consistent across a spectrum of networks from different species.

Annotations

In order to assign functional information to each protein in the PPI network, we have used *Gene Ontology terms*, commonly called *GO terms*. These are part of a large project called *Gene Ontology* that aims to unify the representation of gene attributes across all species [49]. Moreover, we have also used two smaller annotation sets that only contained 13 and 14 functional terms respectively. The first one belongs to Christian von Mering et al. [50] and contains the following annotations:

- Energy production
- Amino acid metabolism
- Other metabolism
- Translation
- Transcription
- Transcriptional control
- Protein fate
- Cellular organisation
- Transport and sensing
- Stress and defence
- Genome maintenance
- Cellular fate / organisation
- Uncharacterized

The second annotation file is from Charlie Boone [51] and contains a slightly different annotation set:

Basic statistics of the Human PPI network

Clustering coefficient	0.125	Number of nodes	11099
Connected components	77	Density	0.001
Network diameter	13	Heterogeneity	1.945
Average number of neighbours	10.236	Isolated nodes	0
Network centralisation	0.046	Number of self-loops	0
Shortest paths	119,301,268 (96%)	Multi-edge node pairs	0
Characteristic path length	3.963	Edges	56806

Table 2.3: Basic statistics for the Human PPI network. The Human PPI network has a large number of nodes (11099), a small clustering coefficient (0.125) and a large network diameter (13).

- Golgi endosome vacuole sorting
- Metabolism - mitochondria
- DNA replication
- Chromatin transcription
- Cell polarity morphogenesis
- Signalling stress response
- Chromatin segmentation
- Protein folding
- ER Golgi traffic
- Nuclear cytoplasmic transport
- Cell cycle progression meiosis
- Protein degradation proteosome
- RNA processing
- Ribosome translation

Nevertheless, both annotation files label each proteins according to their function. Since these annotation sets are more compact[4] than GO terms, we have found it easier to work with Boone's and von Mering's annotation files.

2.7.2 Metabolic networks

A metabolic network is a set of chemical and metabolic processes that regulate physiological and biochemical properties of a cell. Therefore, these networks contain metabolic pathways and regulatory interactions that guide these processes. The source of our metabolic network data and annotations is the *Kyoto Encyclopedia of Genes and Genomes* (KEGG) [52]. Other sources where metabolic networks are available include *EcoCyc* [53] and *BioCyc* [54].

Throughout the project we have analysed two main types of metabolic networks:

- Enzyme-based metabolic networks: each node in the network graph corresponds to an enzyme, protein, metabolite or other chemical. An edge is constructed whenever two chemicals participate in the same reaction.
- Compound-based metabolic networks: each node in the network graph is a compound, which is a set of enzymes that usually take part in one reaction of the metabolic process. In the compound-based networks, edges are formed between compounds as opposed to individual enzymes.

Annotations

The two types of metabolic networks can be annotated with functional information about the enzymes or compounds respectively. One annotation set we used is the Enzyme Commission

[4]GO terms are in the order of thousands. A more compact version called GO Slim exists, which has around 100 different functional annotations.

Basic statistics of the Human Metabolic network

Clust coeff	0.251	Nr of nodes	1343
Connected components	2	Density	0.005
Network diameter	9	Heterogeneity	2.702
Avg. nr of neighbours	6.774	Isolated nodes	0
Network centralisation	0.322	Number of self-loops	26
Shortest paths	1,796,942 (99%)	Multi-edge node pairs	1127
Characteristic path length	3.362	Edges	8610

Table 2.4: Basic statistics for the Human Metabolic network. In terms of node size, the Human Metabolic network lies somewhere in between the PPI network and the World Trade network. It also has a medium clustering coefficient (0.251) and small density (0.005).

number, which is a numerical classification for enzymes that is based on the chemical reactions they catalyse [55]. Every enzyme code consists of four numbers separated by periods. When annotating the enzymes from our metabolic networks we have only used the top-level EC numbers:

1. **Oxidoreductases**: enzymes that catalyse the transfer of electrons from one molecule to another.
2. **Transferases**: enzymes that enact the transfer of specific functional groups (e.g. a methyl or glycosyl group) from one molecule to another.
3. **Hydrolases**: enzymes that catalyse the hydrolysis[5] of a chemical bond.
4. **Lyases**: enzymes that catalyse the breaking of various chemical bonds by means other than hydrolysis.
5. **Isomerases**: enzymes that convert a molecule from one isomer to another.
6. **Ligases**[6]: enzymes that catalyse the joining of two large molecules by forming a new chemical bond.

2.7.3 World Trade Networks

The *World Trade network* (WTN), commonly called the trade or economic network in this report, contains a set of countries and the corresponding trade volume in commodities between them in a particular year. The volume of trade is expressed in international dollars ($). The data has been taken from the *United Nations Commodity Trade* website (Comtrade) [56]. The data that is available on the website is given as an undirected edge-list file, where each edge is weighted by the volume of trade between those two countries. Moreover, the edge list is sorted by the weight, with the countries that traded most with each other at the top of the list.

Since most of the countries trade with each other at least in small or negligible amounts, the original network graph is very dense. In order to reduce the density of the network and analyse only the important economic links between countries, the network has been thresholded to an 85% level. This means that only the highest-weighted edges that made up 85% of the total trade were finally kept, with the rest being discarded. As a result of this thresholding operation, we only kept the countries that trade significantly with each other.

We have analysed data from 49 different WTNs for all years between 1962 and 2010. Having this time series data allowed us to find patterns in the changes of world trade as it evolves over

[5]Hydrolysis is a chemical process in which chemical bonds are broken by the addition of water.
[6]from the Latin verb ligare: "to bind" or "to glue together"

Basic statistics of the 2010 World Trade Network

Clust coeff	0.583	Nr of nodes	119
Connected components	1	Density	0.110
Network diameter	4	Heterogeneity	1.255
Avg. nr of neighbours	12.992	Isolated nodes	0
Network centralisation	0.578	Number of self-loops	0
Shortest paths	14,042 (100%)	Multi-edge node pairs	0
Characteristic path length	2.137	Edges	773

Table 2.5: Basic statistics for the 2010 World Trade network that has been thresholded at the 85% level. The network has a small number of nodes(119), large clustering coefficient(0.583) and small network diameter(4).

time. Table 2.5 shows basic statistics for the 2010 trade network - thresholded at the 85% level. Moreover, apart from total trade network, we have also worked with the following commodity-specific trade networks:

- Minerals and fuels
- Food and live animals

These networks represent the trade in a specific commodity that was done throughout the world. Table 2.5 shows basic statistics for the 2010 WTN. One can notice that the diameter of the network is really small (0.4). This is slightly undesirable, because the smaller the diameter the stronger the GCV correlation will be between nodes, because the probability of two nodes sharing part of their neighbourhood is large. Therefore, we tried thresholding the network at levels lower than 85%, in order to make the network more disconnected and therefore increase its diameter. However, this attempt has not succeeded, because the network diameter has stayed at the same level. The reason for this might be because of the scale-free properties of the network, which ensure that when thresholding is applied, the isolated nodes are removed and only hub nodes are kept.

Annotations - Economic indicators

The basic economic indicators that we have used for the Canonical Correlation analysis are the following:

- Population (POP): The total population of the country. Data source: WEO [57]
- Level of Employment (LE): The number of people who performed some work during a specified period. Data source: WEO [57].
- Real GDP per capita (RGDPL): Purchasing Power Parity adjusted Gross Domestic Product (Laspeyres) which was derived from the growth rates of consumption share, government share and investment share. Data source: PENN [58]
- Real GDP per capita (RGDPL2) – Purchasing Power Parity adjusted Gross Domestic Product (Laspeyres) which was derived from the growth rates of domestic absorption. Data source: PENN [58]
- Real GDP per capita – Constant Prices Chain series (RGDPCH). Data source: PENN [58]
- Consumption Share of RGDPL (KC) – Data source: PENN [58]
- Government Share of RGDPL (KG) – Data source: PENN [58]

- Investment Share of RGDPL (KI) – Data source: PENN [58]
- Exchange Rate (XRAT) – Data source: PENN [58]
- Current Account Balance (BCA): The difference between a country's exports of goods and services and its imports. Financial transfers and investments are not taken into account. Data source: PENN [58]
- Trade Openness (OPENK) – Data source: PENN [58]

Moreover, this list of economic indicators has been augmented with composed indicators that are the products of several basic indicators. The full list is as follows:

- POP
- LE
- KI x RGDPL x POP
- RGDPCH x POP
- RGDPL x POP
- RGDPL2 x POP
- KG x RGDPL x POP
- KC x RGDPL x POP
- KC x RGDPL
- XRAT
- RGDPCH
- RGDPL
- RGDPL2
- KG x RGDPL
- KI x RGDPL
- KC
- KI
- BCA per RGDPL
- KG
- BCA
- OPENK

Chapter 3

Methodology

3.1 Mathematical model

We now introduce the novel signature that can be interpreted as a generalisation of the *Graphlet Distribution Vector* (GDV) described in section 2.3.4. This signature, which we shall call the *Graphlet Cluster Vector* (GCV) (for reasons that will soon become obvious), is a central concept of this project. Since the GCV is a novel signature, we would like to explore its properties and find out how to use it for getting insights from the network data. The idea for the novel GCV signatures came from Zoran Levnajić, one of Nataša Pržulj's collaborators. Before giving a full definition of the GCV, we first define what the *neighbouring subgraph* of a node n is:

Definition 38 *Let $G = (V, E)$ be a graph and n be a node in V. The neighbouring subgraph $S_n = (V_n, E_n)$ of node n is an induced subgraph of G where V_n is the set of all neighbouring vertices of n, with $n \notin V_n$.*

This implies that S_n will contain all the edges between the neighbours of n excluding those coming from the source node n itself. Now that we have defined the neighbouring subgraph of a node, we are ready to give the full definition of the new *Graphlet Cluster Vector*:

Definition 39 *Let G be a graph, n a node in G, S_n the neighbouring subgraph of n in G and let S_n^i be the number of graphlets of type i in S_n, $i \in \{1, 2, \dots 29\}$. The Graphlet Cluster Vector of node n is a vector of 29 elements defined as:*

$$GCV(n) = \left(S_n^1, S_n^2, \dots, S_n^{29}\right)$$

The GCV signature of a node n is therefore counting the number of graphlets of each type in n's neighbouring subgraph. One can also normalise it with respect to the total number of graphlets found in S_n to get the *normalised Graphlet Cluster Vector*. The formal definition is the following:

Definition 40 *Let G be a graph, n a node in G, S_n the neighbouring subgraph of n in G and let S_n^i be the number of graphlets of type i in S_n. The normalised Graphlet Cluster Vector of node n is defined as:*

$$GCV(n) = \left(F_n^1, F_n^2, ...F_n^{29}\right)$$

where

$$F_n^i = \frac{S_n^i}{\sum_{i=1}^{n} S_n^i}$$

There are several ways to interpret both variants of the GCV signature:

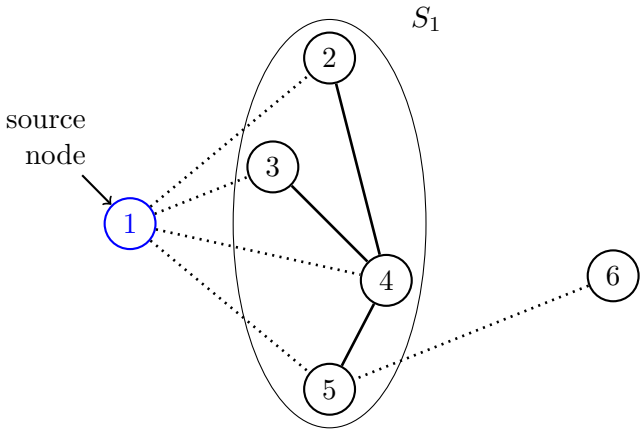


Figure 3.1: Illustration of the neighbouring subgraph S_1 of node 1. S_1 is made of 4 nodes: {2,3,4,5} and 3 edges: {(2, 4),(3, 4),(4, 5)}. In order to find the GCV signature we count the frequency of graphlets of each type in S_1. In this example we obtain: $(3, 3, 0, 1, 0, 0, 0, ...)$. Note that since there is no edge between nodes 1 and 6, node 6 is not used for calculating the GCV of node 1. Moreover, source node 1 and the edges linking it are also excluded from S_1.

- GCV generalises the GDV by capturing structural information in the neighbouring subgraph of a particular node. The GDV used to count the number of graphlets touching a node at a particular orbit.
- In the normalised version, if the GCV would have also recorded the frequency of graphlet G_0[1], that frequency would have represented the clustering coefficient of node n. Therefore, one can also interpret the GCV as a generalisation of the clustering coefficient of a node.
- The normalised version of the GCV of a node n can also be interpreted as an exponentiated[2] RGFV[3] of the neighbouring graph of node n.

The reason we don't include graphlet G_0 is because that simply gives us the clustering coefficient of the node. Moreover, G_0 correlates positively with all the other graphlets in the vector, since it is a subgraph of all the other graphlets. Therefore, that does not give us any useful information in Pearson's GCV correlation matrices or CCA.

As it was previously mentioned, the core idea of the GCV signature belongs to Dr. Zoran Levnajić, one of Nataša Pržulj's collaborators. Nevertheless, I also have some contributions to the mathematical model, because at the beginning of the project I researched a few normalisation methods and sizes of the neighbouring subgraph of a node where the GCV is calculated. After several normalisation procedures and neighbouring subgraph sizes have been discussed and analysed, we decided on the version presented in this paper. For an overview of a different normalisation method attempted, see section 3.1.1. For a study on the neighbouring subgraph size, see section 3.1.2.

3.1.1 GCV normalisation attempt

Our initial plan was to normalise each frequency in the GCV signature according to the maximum possible number of graphlets of that type in the neighbouring subgraph. However, this proved to be a very complicated mathematical problem, because both of the following subproblems are mathematically non-trivial:

[1] G_0 is simply an edge between two nodes.
[2] RGFV applies a logarithmic function to each of the frequencies, see definition 24 in section 2.3.2
[3] RGFV counts the frequency of graphlets in the whole graph, see definition 24 in section 2.3.2

1. Finding out which graphs contain the maximal number of graphlets of each type.

2. Once such a graph is found, finding the formula for the maximum number of graphlets.

Each of the above subproblems are unique for all the 30 different graphlets and they have to be solved separately. One remark we can make is that an upper limit for the maximum number of graphlets of type i is :

$$max(i) = \binom{n}{k} = \frac{n!}{k!(n-k)!}$$

where $i \in \{1, 2, \dots 29\}$ and k is the number of nodes in graphlet i.

For the first subproblem, one can hypothesise a graph structure that might have the maximal frequency of graphlet i for a fixed number of nodes n and mathematically prove that no other graph with the same number of nodes can yield a higher frequency. To illustrate the complexity of the problem, let us find the maximum number of graphlets of type 1 (G_1, path of 3 nodes) in a graph $H = (V, E)$, where $|V| = n$, for a given n. First of all, we need to identify which graph H gives a high frequency of graphlet G_1. Since G_1 is not a clique, it would not be convenient for H to be a clique either, otherwise the frequency of G_1 would be 0. Similarly, if H has no edges, then the frequency of G_1 is also 0.

One type of graph H that might give us a high frequency of G_1 is a bipartite graph. Moreover, let us also assume that the nodes of H are split into two sets of equal cardinality[4] H_1 and H_2. Since we would like H to be bipartite, let us assume edges exist between all nodes i and j, with $i \in H_1$ and $j \in H_2$. Now that we have a candidate graph H that might give us a high frequency of G_1 graphlets, one can count how many graphlets G_1 there are in H. The exact frequencies are given below:

$$max(G_1, H) = \begin{cases} n^2(n-2)/8 & \text{if } n \text{ is even} \\ x(x+1)(2x-1)/2 \text{ where } x = \lfloor \frac{n}{2} \rfloor & \text{if } n \text{ is odd} \end{cases}$$

Therefore, this is the hypothesised maximum number of graphlets of type G_1 in a graph H of n nodes. The formulae already look complicated and get even more complex as the graphlets increase in size and density. Therefore, the problem of finding the maximum theoretical number of graphlets of each type is infeasible, at least for the purposes of our project. We have therefore decided to only normalise the GCV with the sum of all the frequencies, as it is given in definition 40 in the previous section.

3.1.2 Study on neighbouring subgraph size

One other aspect that has been closely studied is the size of the neighbourhood subgraph. The current definition of the GCV uses a subgraph that excludes the source node and nodes that are at a distance of 2 or more from the source node. However, the subgraph can be extended in two different manners:

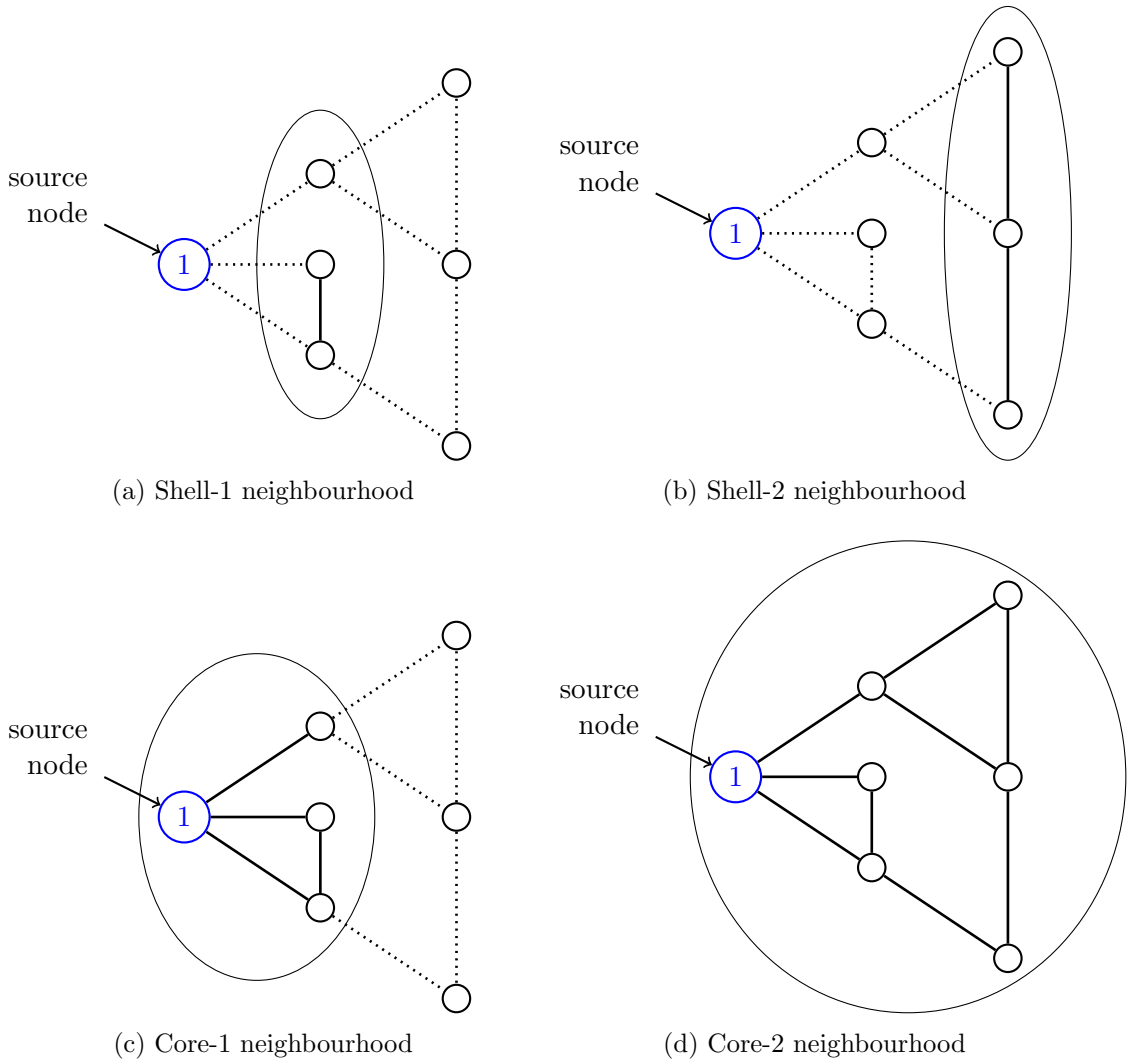
- **Shell extension**: For a given parameter d, the neighbouring subgraph S_n of a node n contains nodes that are at a distance d from n, where the distance between two nodes is defined as the minimum path length.

- **Core extension**: For a given parameter d, the neighbouring subgraph S_n of a node n contains all nodes that are at a distance d or smaller from n. Moreover, $n \in S_n$.

Figure 3.2 illustrates different shell and core neighbouring subgraphs for a source node. There are several problems associated with shell and core neighbourhoods that are larger than 1:

[4] If n is odd then one extra node is added in H_1.

- The GCV computation becomes intractable for large networks, because the neighbourhood of each node is bigger.
- Finding the actual neighbourhood requires graph searching in order to find the shortest path between the source node and every other node in the network. This places further computational demand on the algorithm.
- Some networks such as the World Trade Networks have a short diameter of approximately 5. Therefore, the core-5 neighbourhood will contain all the nodes in the network, while core-4 and core-3 will also contain a lot of nodes if the starting node is a hub node.



(a) Shell-1 neighbourhood

(b) Shell-2 neighbourhood

(c) Core-1 neighbourhood

(d) Core-2 neighbourhood

Figure 3.2: Shell and Core neighbourhoods for source node 1: (a) Shell-1 is the subgraph of nodes at distance 1 from the source node. (b) Shell-2 is the subgraph of nodes at distance 2 from the source node. (c) Core-1 is the subgraph of nodes at distance 1 or smaller from the source node (including the source node itself). (d) Core-2 is the subgraph of nodes at distance 2 or smaller from the source node (including the source node itself).

Because of these reasons we have decided against the use of core and shell neighbourhoods that have a size larger than 1. This left us only with core-1 and shell-1. The final decision was to use shell-1 because the resulting GCV signature would not count any graphlets that the older GCV signature would count as well. With core-1, there exist graphlets that touch the source node which are also counted by the other GDV signature.

3.1.3 Relative Cluster Frequency Distance

We now define the *Relative Cluster Frequency Distance* (RCFD), which measures the distance between two GCVs. It is the equivalent of RGFD (section 2.3.3), but instead uses the GCV instead of the GDV.

Definition 41 *Let G be a graph, p and q two nodes in G and let F_p^i and F_q^i be the frequency of the i^{th} graphlet in the GCVs of nodes p and q respectively. The Relative Cluster Frequency Distance (RCFD) between p and q is then defined as:*

$$RCFD(p,q) = \sqrt{\sum_{i=1}^{n} |F_p^i - F_q^i|^2} \tag{3.1}$$

Note that the RCFD formula uses the Euclidean-distance, while the RGDF uses the absolute value as the distance measure. In this project, we use the Euclidean distance version for computing the RCFD.

3.2 Implementation

After the mathematical model behind the Graphlet Cluster Vector has been formally defined, we implemented it in `C++`. The two main reasons for choosing `C++` are as follows:

1. Nataša Pržulj's research group already wrote a `C++` function that counted the number of graphlets in a given input graph. Therefore, we were able to leverage that code and add extra functionality on top of it.

2. `C++` is a compiled language that does not run in a virtual environment, and consequently it can run intensive computations very fast. A similar implementation in a language such as Java, which runs on a virtual machine, takes longer. Since our algorithm is required to execute intensive computations on large biological networks, we decided that `C++` was the most suitable programming language for this.

The `C++` file that I was given from N. Przulj's group (called ncount.cpp) was used to count both the RGFV (i.e. the number of graphlets of each type in a graph) and also the GDV signatures (i.e. the number of automorphism orbits that nodes touch). I was also given another script that was used to convert the given networks[5] to a file format called LEDA [59], that is easy to be read and processed by the graphlet counting function.

3.2.1 Node-based Graphlet Cluster Vector

We started writing the implementation by first modifying a function called `count()` that computed the GDV signature (from `ncount.cpp`) and removing the unnecessary code that was dealing with automorphism orbits. Afterwards, we realised that the function was still hard to work with, for it was very long[6] and resembled a 'God-function' that was responsible for everything: reading from the input file, parsing it, building an efficient data structure to store the input in, counting the graphlets and writing to the output file. We therefore decided to split it up into modules according to their responsibility. Some of these concepts have been introduced in the second-year Software Engineering course. We delegated the reading and parsing of the input file and writing the out files to separate functions and cleaned up the unnecessary computations.

[5] The given networks were represented as edge lists in a text files.
[6] 400 lines of code

After re-structuring the function that computes the GDV signature, we modified the main loop that was going over every single node and inserted some code that would extract its neighbouring subgraph (see definition 38) and store it in a data structure. However, this proved to be one of the hardest tasks because the data structures used to represent the graph were complex. The programmers who implemented the initial graphlet-counting function used a specially-optimised data structure that stored the network graph in two different structures at the same time: an adjacency matrix and an adjacency list. This turned out to be a good idea, because using both the list and the matrix forms allowed for many operations to be executed in constant time:

- The adjacency matrix allowed one to check whether two arbitrary nodes are connected in O(1) time.
- The adjacency list allowed one to get the list of all the neighbours of a node in O(1) time. This was especially useful for our extension, where we extracted the neighbourhood of a node in order to count the number of graphlets in it.

However, the complexity associated with this representation is that both structures had to be synchronised at every point in the execution of the algorithm. Fortunately, since our algorithm was only computing the number of graphlets in the given graph, there was no need to change the graph structure. Nevertheless, constructing the data structure was made even more complicated by the following facts:

- The representation used by the adjacency matrix and the adjacency list used low-level `C++` optimisations. For example, the adjacency matrix was using a `char` for storing 8 consecutive binary values. As a result, connecting two vertices i and j with an edge became as complicated as this: `adjmat[i][j / 8] = 1<<(j % 8)`
- The extensive use of macros in the code we were leveraging. For example, a for loop that iterated through all the nodes in the network was defined using the following declaration: `#define foreach_adj(x,y) for(x = edges_for[y]; x != edges_for[y+1]; x ++)`. Similar macros existed for operations like connecting two nodes, calculating the degree of a node or checking if two nodes are connected.

After the neighbourhood of the node has been successfully extracted and represented in this data structure, it was simply passed to the graphlet-counting function. The result obtained is a 29-element GCV containing the frequencies of each graphlet type. This process is then repeated for the rest of the nodes in the network graph. At the end the program outputs a list of nodes and their corresponding GCVs. The algorithm was tested for correctness on some small, manually-constructed networks. Afterwards, we ran it on four different real networks:

- Protein-Protein Interaction network (PPI)
- Metabolic network
- 2010 Full World Trade network (WTN)
- 2010 Thresholded World Trade network (WTN)

However, the computation was taking more than 10 hours for the 2010 Full WTN, so the next step was to parallelise the computation.

3.2.2 Parallelisation

Given a graph $G = (V, E)$, the easiest approach to parallelise the GCV computation is to split the vertex set V into n different chunks $(V_1, V_2, \ldots, V_n)$ and then distribute them to the working threads or processes. Each process i would receive its chunk V_i, calculate the GCV for each node in V_i and write the output to a file. After all the processes have finished their work, all the n files can be assembled together.

Although there exist several other approaches, this is the method we chose to implement. More precisely, we parallelised the code across multiple cores by creating child processes with the `C++ fork()` function. After being forked, each child process starts computing the GCV for its chunk of nodes and outputs the results to its own output file, suffixed with the process number. Meanwhile, the parent process waits until all the child processes finish their execution, at which point it assembles all the output files[7] together and cleans up the environment. A diagram of this process can be visualised in figure 3.3. Moreover, we added extra functionality to the parallelisation code by allowing a variable number of processes to be generated and have this number passed as a parameter to the program. This is useful especially because the ideal number of processes can vary from one machine to another, depending on the number of available cores. Given that we have access to machines that have at most 64 cores, this parallelisation can in theory offer us a maximum speedup of 64 if run on these machines. A pseudocode of the parallelisation logic is given in figure 3.4.

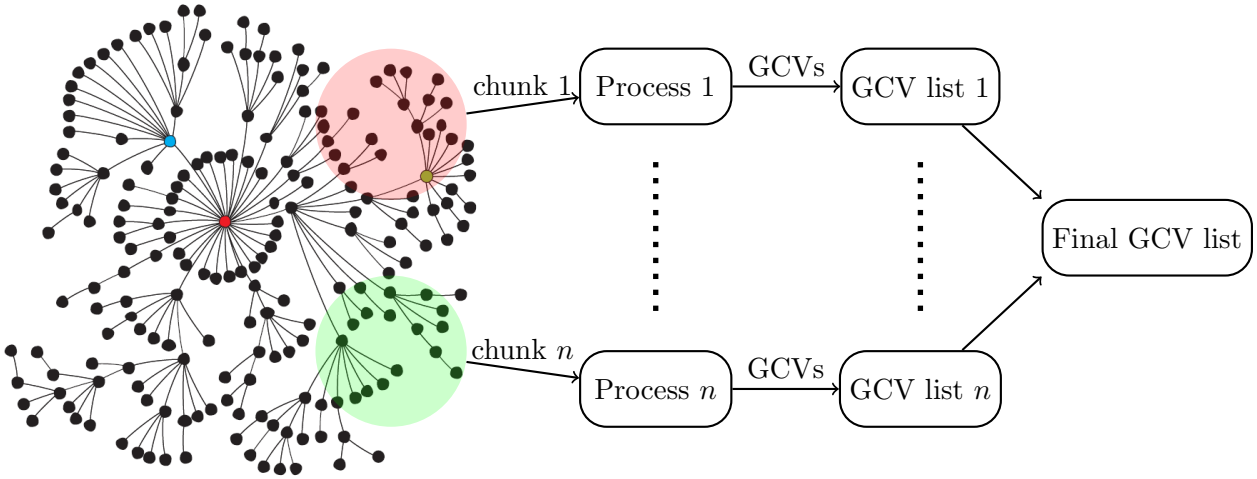


Figure 3.3: Illustration of the parallelisation process for the GCV computation. For an input network, we split the nodes into different chunks and assign each chunk to a process. Each process computes the GCVs only for his chunk of nodes and writes them to an output file. At the end, all the GCV lists from the output files are assembled together into one final list. During the assembly process, the final GCV list is also sorted by node entry in order to easily visualise all the GCVs and to simplify our consistency checks.

Because of the way we chose to implement parallelisation, some processes tend to finish earlier than other. Even if every process computes the GCV for the same number of nodes in the network, computing the GCV for hub nodes takes considerably longer because they have large neighbouring subgraphs. As a result, some processes finish early while others get stuck with the GCV computation for some hub nodes. However, this limitation tends to become less obvious as the size of the input network increases. One way to overcome this problem is to redistribute the computation to the processes that finish early.

[7] Each output file is tagged with an ID of the process that generated it. However, this ID is not the PID of the process.

```
for each child process
{
  /* spawn a new process and store its PID */
  pids[proc_index] = fork();
  if (current process is a child)
  {
    /* open file suffixed by process number */
    FILE* out_file = fopen(out_name + proc_index, "w");
    /* find the nodes that the child needs to process */
    nodes_to_process = [CHUNK_SIZE * proc_index, CHUNK_SIZE *
   proc_index + 1]
    /* compute the GCV list and write them to the output */
    compute_gcv(input_graph, out_file, nodes_to_process);
    /* the child process closes the file and terminates */
    fclose(fp_out);
    return 0;
  }
}

/* Parent process waits on all children to finish execution */
for each child process
{
  waitpid(pids[proc_index]);
}
```

Figure 3.4: Pseudocode for the parallelisation logic that is implemented in file `e_gdv.cpp`. Note that the actual GCV computation takes place in the `compute_gcv` function. The reason why the output file pointer is passed to this function is because we want each process to write the GCV signatures to the output files on the fly, as soon as they are computed. This helps us debug the software more easily and also avoid "out of memory" problems when processing large network files which have more than 11.000 nodes.

In order to evaluate the speedup from parallelisation, we repeatedly perform the GCV computation on the PPI, WTN and Metabolic networks using a variable number of processes and network size. Each experiment is also repeated 5 times and the average running time is reported. The machine we run the experiments is the `Bionets02`, having the following specifications[8]:

- cpu: 4 x AMD Opteron(tm) Processor 6282 SE @ 2600MHz
  - Number of cores: 16
  - Data width: 64 bit
  - Level 1 cache size: 8 x 64 KB 2-way associative shared instruction caches, 16 x 16 KB 4-way associative data caches
  - Level 2 cache size: 8 x 2 MB 16-way associative shared exclusive caches
- memory: 125GB

We therefore calculate on `Bionets02` the speedup obtained on the Human PPI, Human Metabolic and 2010 World Trade network as the number of processes increases from 1 to 64. The speedup s_n when using n processes is calculated as follows:

$$s_n = 100\left(\frac{T_1}{T_n} - 1\right)$$

where T_n is the wall-clock time of execution when using n processes, while T_1 is the wall-clock time of the serial execution. The final value is multiplied by 100 so that we can express it in percentage terms. Figure 3.5 shows the speedup for the PPI, Metabolic and WTN networks. Each experiment has been run 5 times and the average running times T_n have been used to compute the final speedup. When the number of processes is 2, we don't get any speedup in the execution, but as the number of processes increases, some speedup is clearly visible for the PPI and WTN networks. The Metabolic network only shows some speedup when 64 processes are running the computation. The PPI and the WTN networks also show a considerable speedup at the end, when executing the computation on 64 processes. It should be noted that the speedup of WTN becomes greater than the equivalent speedup of the PPI network when more than 32 processes are used.

[8]The CPU type and memory size are taken from the output of the `lshw` command. The specifications of the AMD Opteron 6282 SE processor are taken from the www.cpu-world.com website.

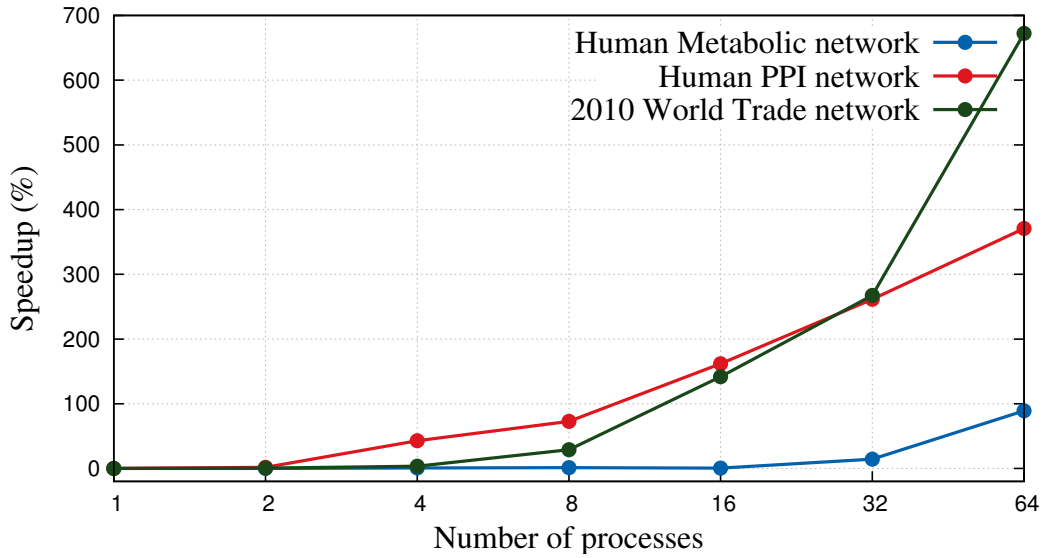


Figure 3.5: Speedup gained from parallelisation as the number of processes increases. The following three different networks have been tested: A human PPI network, a human metabolic network and a 2010 World Trade network (WTN). Although not immediately obvious from the graph because of the logarithmic X axis, the speedup trend is linear in the number of processes. A maximum speedup of 680% and 380% is obtained for the WTN and the PPI network respectively when 64 processes are used.

After evaluating the speedup of the parallelisation, we are now interested to see how the execution time changes when we increase/decrease the problem size. In order to test for different problem sizes, we take a large network and randomly remove edges from it. We therefore generate networks that contain 50%, 60%, ..., 100% of the edges of the initial network. We then compute the execution time (wall-clock time) on each of these incomplete networks for a different number of processes. For each process and network size, 5 trials are run and the average execution time is reported. Figure 3.6 shows the results obtained for this experiment. When the problem size is small, the execution time is fast regardless of the number of processes used. However, as the network size increases, the speedup gains from parallelisation become apparent, because the difference between lines widens. Eventually, when 64 processes are used, the execution time on the full network is approximately 32 seconds, which is 3-4 times faster than the equivalent execution time with 1 process. Note that the apparent inconsistencies between the PPI results in figure 3.5 and the last column from figure 3.6 might be because of the fact that the `Bionets02` might have had other services from users running in the meantime that could have affected the performance.

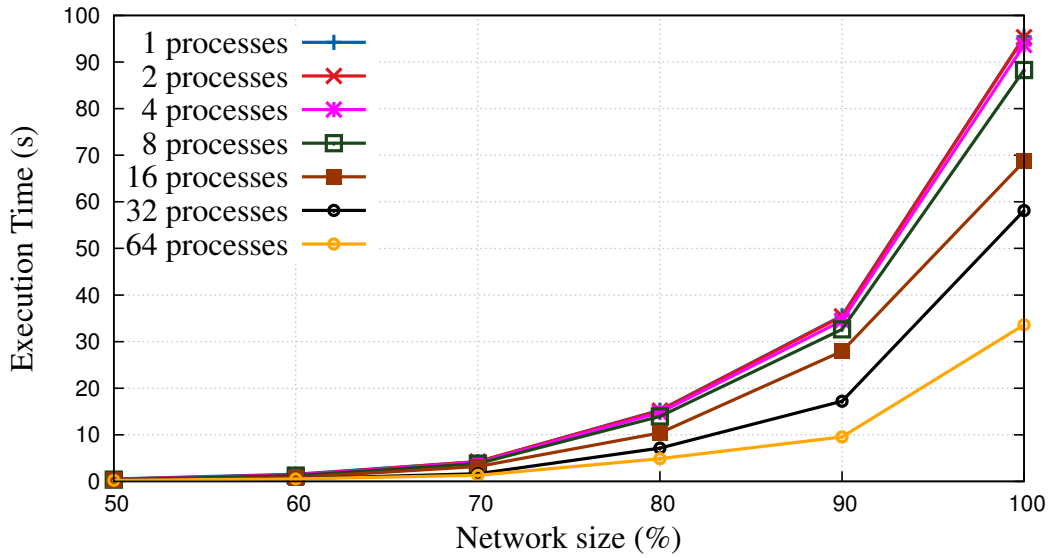


Figure 3.6: Execution time plotted for different number of processes as the network size increases. The input network used is the Human PPI network, and a network size of $p\%$ refers to the percentage of edges that were kept from the original network. For all network sizes, one can easily notice that the execution time gets faster as the number of processes increases. These results suggest that the gains in parallelisation are noticeable only for large networks.

The network that had the longest average runtime was the PPI network, with an execution time ranging from 95 seconds (1 process) to 32 seconds (64 processes). Although an execution time of 95 seconds is not problematic for our experiments, other PPI networks[9] we have experimented with have taken around 10 hours to finish using 64 parallel processes. Moreover, other networks such as the literature network[10] of the Bible have taken several days to finish using 64 parallel processes. The reason the GCV computation takes so long to finish on these networks is because some processes get stuck with computing GCVs for hub nodes, which have very large neighbouring graphs.

In conclusion, parallelisation of GCV computation was a key part of the project that enabled us to run more experiments faster and to exploit all the computational resources of our machines. Some of the experiments on the PPI and literature networks would not have been possible without parallel computation.

3.2.3 Pearsons's GCV correlation matrix

In the background section 2.5.3, we introduced the Pearson's GDV correlation matrix for a given graph. Similarly, the *Pearson's GCV correlation matrix* can also be computed in order to find out which graphlets cluster together. This is important because graphlets that cluster together have a similar behaviour and also correlate with the same functional annotations. We present to the reader the steps used for computing the *Pearson's GCV correlation matrix*, which uses the GCV (Graphlet Cluster Vector) instead of the GDV (Graphlet Degree Vector):

1. We compute the Graphlet Cluster Vector (GCV) for every node in the input network
2. We then construct samples $S_i, i \in 1, 2, 3, .., 29$ containing all the frequencies of the graphlet of type i found in the GCVs of the nodes. The length of S_i would be equal to N, the number of nodes in the network.

[9] the PPI networks from BioGRID - Full version, see section 4.3.3

[10] We have also experimented with literature networks, which are networks of characters from a book. However, the results were not significant so we have not included them in this report.

3. We compute the Pearson's correlation coefficient for each pair of samples (S_i, S_j) and we write them in the 29x29 correlation matrix C at position (i, j).

The program that computes the GCV correlation matrix has been written without using any library functions. Nevertheless, it took me a few of hours to identify some bugs and memory leaks. Using GDB and Valgrind has proven to be extremely helpful for this task. I also implemented my own function that computes the Pearson's coefficient for two samples X and Y and tested it using an excel spreadsheet that computed the correlation coefficient in parallel.

Unfortunately, the first heat maps that we get for the three main network classes (PPI, Metabolic and Trade) are not easy to interpret. See the initial image (top-left corner) from figure 3.8, which shows the original heat map obtained for the Human PPI network. Most of the graphlets display a high correlation (at least 0.5) and because of that we cannot distinguish clusters easily. Similar results are obtained for the other two networks: Human Metabolic and WTN. In order to identify which graphlets cluster together, we apply two main modifications to the matrices:

- Normalisation: We normalise the correlation values so that they lie more evenly in the (0,1) range.
- Hierarchical clustering: In order to better identify clusters of graphlets that are highly correlated with each other, we perform hierarchical clustering on the set of 29 graphlet signatures.

3.2.4 Normalisation

Two main normalisation steps have been performed in order to spread out the correlation values over the (0,1) range:

1. Feature scaling
2. Polynomial scaling

By feature scaling we denote a uniform scaling that makes the data fit on the (0,1) range. On the other hand, polynomial scaling applies a polynomial function to the input value. They are formally defined as follows:

Definition 42 *Let X be a population and $min(X)$, $max(X)$ be the minimal respectively maximal value in X. Feature scaling is a transformation that converts each element $x \in X$ into an element x' such that:*

$$x' = \frac{x - min(X)}{max(X) - min(X)} \tag{3.2}$$

Definition 43 *Let X be a population. Polynomial scaling is a transformation that converts each element $x \in X$ into an element x' such that:*

$$x' = x^n$$

For each matrix, feature scaling is first applied followed by polynomial scaling. For both feature scaling and polynomial scaling, the set of all entries in the 2D matrix are used as the population vector X. After applying feature scaling, all the entries in the correlation matrix are converted to the (0,1) range. This results in both the input and output of the polynomial scaling to also be in the (0,1) range, regardless of the parameter n that is used.

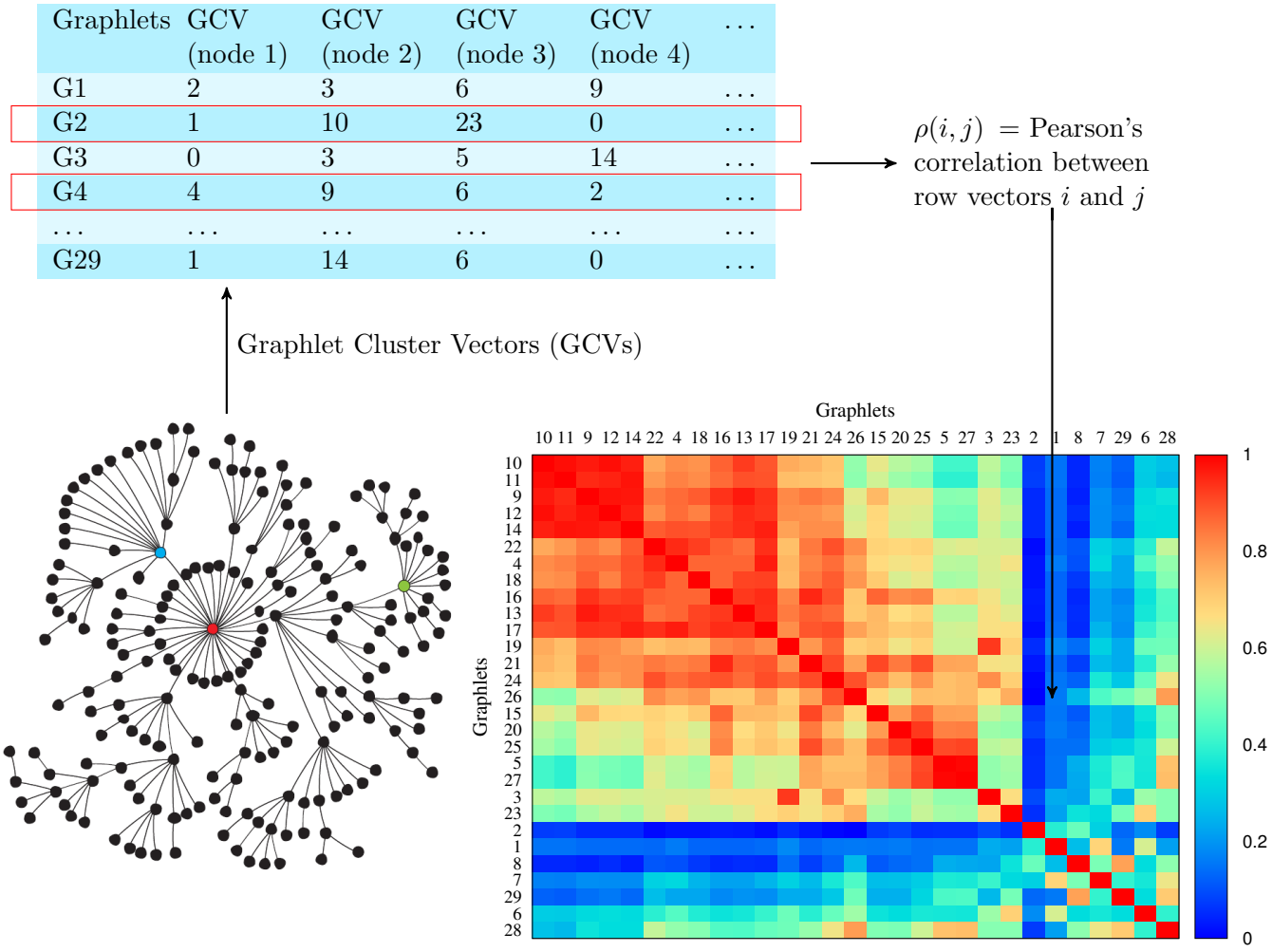


Figure 3.7: Computation process of the Pearson's GCV correlation matrix. For an input network, we compute a table of the GCV signatures for all the nodes in the input network. Afterwards, we compute the Pearson's correlation coefficient $\rho(i, j)$ between each pair of row vectors i and j and store it at position $M[i, j]$ in the correlation matrix M.

3.2.5 Hierarchical clustering

Hierarchical clustering is a method that clusters data points according to how similar they are. In our case the data points were the 29 GCV correlation vectors, and the similarity measure used was given by the Euclidean distance. See section 2.5.4 for background information on hierarchical clustering. The reason for clustering them is because we need to find out which graphlets are similar and which ones are different with each other. The graphlets that are similar probably have some common properties that we are able to identify and interpret.

We have used the python library `Scipy` to run hierarchical clustering. We first calculate a distance matrix using the `Scipy.spatial.distance`. This symmetric matrix stores the distances between every two data points as a 2D matrix. Afterwards, the hierarchical clustering is performed with the function call `scipy.cluster.hierarchy.linkage(dist_matrix, method ='complete')`. The method parameter refers to the type of hierarchical clustering that is performed. We have chosen to use Complete linkage[11] because it avoids the so-called *chaining phenomenon* of single linkage, where clusters are forced together due to outlier data points

[11] Complete linkage groups clusters together according to the shortest distance between the farthest points in the sets (see the definition for complete linkage in section 2.5.4).

being close to each other, even though the majority of the data points might actually be away from each other. It has also been shown that complete linkage tends to create clusters of similar diameter [60].

Each Pearson's correlation matrix is first normalised using both feature scaling and polynomial functions and then hierarchically clustered. We shall refer to this process as the *Pearson's GCV correlation matrix life cycle*. More details about its implementation aspects can be found in section 3.2.7. Figure 3.8 shows the *Pearson's GCV correlation matrix life cycle* for the Human PPI network. One can clearly see that the initial matrix is very hard to interpret, having all correlations very high. On the other hand, the final matrix is very easy to interpret and clearly shows graphlet clusters formed along the diagonal. In this example, we used a a 4^{th} degree polynomial, but for other networks we found that other polynomial functions offer better results. Chapter 4 presents the key results of the Pearson's matrices applied to different network classes.

3.2.6 Canonical Correlation Analysis

Graphlets only give us information about the topology of the network connections. However, in order to associate them with node functions or annotations, we need to correlate the GCV with a vector of node annotations. Canonical Correlation Analysis (CCA) is able to do exactly this and also give us a p-value, that can quantify the significance of the result. The theory behind CCA is given in background section 2.6. In this section, we present how we applied CCA in our experiments and discuss implementation details.

Initially, I experimented with CCA using three different implementations:

- `Python Scikit`: a Python implementation that is based on algorithms by Jacob A. Wegelin [61]. The problem with this implementation is the poor documentation available.
- `Matlab`: based on two books by Krzanowski, W. J. [62] and Seber, G. A [63]. The documentation is good, but the implementation does not provide canonical cross-loadings.
- Darren Davis's `R` script: Darren Davis is a collaborator of N. Pržulj who has previously applied CCA on GDV signatures. His `R` implementation also performs some preprocessing of the data points and covariance matrices, such as scaling, centring[12] and regularisation[13]. This implementation is also accompanied by some python scripts that can preprocess all the trade networks for a given range of years.

We have decided to use Darren Davis's script, mainly because it is able to calculate cross-loadings and it also had 5 accompanying python scripts that preprocess the economic networks. Therefore, we refactored the `R` script in order to allow several parameters to be passed from the command line. On the other hand, the preprocessing python scripts have been extensively modified to deal with the new GCV signature and for enabling them to process different annotation files, such as EC numbers (see section 2.7.2) or Boone's and von Mering's annotations (see section 2.7.1).

Darren Davis's `R` script gives us all the canonical correlation eigenvectors, with their associated correlation and p-values and writes them to a text file. The eigenvectors are sorted in descending order by their correlation strength, so the first eigenvector is the most significant. We therefore created a parser for this file that finds the first eigenvector and produces a LaTeX table with all the cross-loadings and the respective overall correlation and p-value. This script has been used for generating all the CCA tables in this report. We have also created another script that creates a vector graphics image containing the following:

[12] The data points are centred around the origin

[13] If the covariance matrix is singular, a weighted identity matrix is added to it. More formally, if the covariance matrix of the data points is S, then $S' = S + \lambda I$

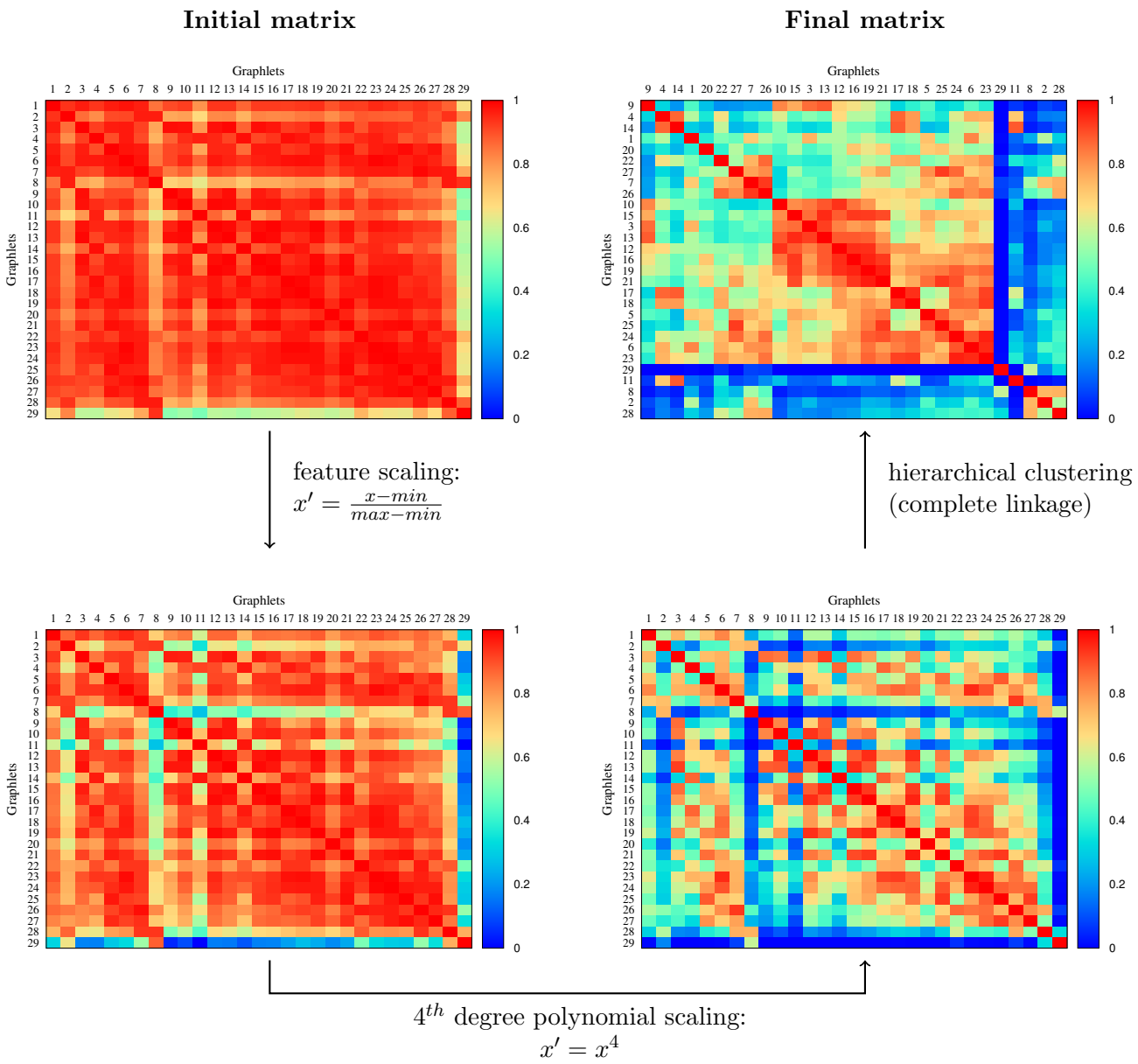


Figure 3.8: Pearson's GCV correlation matrix life cycle for the Human PPI network. The initial Pearson's GCV correlation matrix is on the top-left corner and the final matrix is on the top-right corner, after feature scaling, polynomial scaling and hierarchical clustering operations are applied. One can see that in the final matrix graphlet clusters are distinguished more easily compared to the initial matrix. The operation order is anti-clockwise. When feature scaling, the range of the correlations $[min, max]$ is scaled to $[0, 1]$. After performing polynomial scaling using a 4^{th} degree function, the correlations are lowered even more. Finally, after applying hierarchical clustering, similar graphlets cluster together along the diagonal. Hierarchical clustering uses complete linkage for grouping GCVs and the Euclidean distance to compute the difference between GCVs.

- an indicator list on the left-hand side where is element is coloured from green (1) to red (-1) according to their canonical cross-loading.
- a top-right panel containing the graphlets with the highest canonical cross-loading.
- a bottom-right panel containing the graphlets with the lowest canonical cross-loading.

- a gradient bar in the middle measuring correlation where the graphlets and indicators are connected.

This script is written in Python and generates fragments of `Tikz` code that can be used by LaTeXto generate the image. In order to get the images in their final state, further manual modifications are performed. The images are very intuitive to understand and will be used extensively during the presentation. They are also included in this report in figures 4.8, 4.12 and 4.20. However, we don't include them for all CCA results in the report because they don't give the exact correlations which are required for a careful analysis.

3.2.7 Network life cycle framework

In order to be able to run all our experiments in an automatic fashion and for a variety of networks, we implemented a few *network life cycle frameworks* that take a network and run all the statistical experiments automatically. The frameworks are defined by commands in a Makefile that chain a variety of scripts. We wrote two main classes of such frameworks:

1. Pearson's GCV Correlation matrix life cycle: used for computing all the correlation matrices and generating several types of heat maps.
2. Canonical Correlation life cycle: used for preprocessing a network and its annotation file and applying CCA on them.

Both of these are described in more detail in the following subsections.

Pearson's GCV correlation matrix life cycle

The *Pearson's GCV correlation matrix life cycle* takes an input network and computes several GCV correlation matrices and their corresponding heat maps. Several environment variables need to be set in order to run it, such as the network source folder, network file, generated folder[14] and the GCV normalisation type[15]. The steps that are performed in the life cycle are as follows:

1. Initial file handling: Several directories are created and input files are copied over.
2. GCV computation using `e_gdv.cpp`[16]
3. Average network GCV computation
4. Computation of the Pearson's GCV correlation matrix
5. Computation of 4 types of normalised correlation matrices[17]
6. Hierarchical clustering on all the correlation matrices
7. Heat map generation of all the correlation matrices using `gnuplot`

[14] The generated folder is used for storing all the program results
[15] A binary value that decides whether the GCV is normalised or not.
[16] The name of the script is derived from "extended gdv", because at the time we started writing the script we were not decided on the name for the new signature.
[17] The matrices are normalised with first, second, third and fourth degree polynomials. The higher the degree, the stronger the contrast between correlation values will become.

Canonical Correlation Analysis life cycle

A different framework was set up for Canonical Correlation Analysis. In order to run CCA on an input network, several preprocessing steps are required that transform the GCV and annotation files. For the World Trade network, the steps performed by the CCA life cycle are defined below:

1. Initial file handling: the output folders are created and input files are copied over.
2. Conversion of the GCV dump file to a CSV file for each of the 48 networks over the period 1962-2010.
3. Aggregation of the above per-year GCV CSV files into a single GCV file.
4. Aggregation of the economic indicator files into a single CSV file. Country/year entries with incomplete data are dropped.
5. Augmentation of the basic economic indicators with composed economic indicators (e.g. GDP per capita x Population to get the total GDP of a country).
6. Alignment of the GCV entries with the final economic indicators.
7. execution of the actual Canonical Correlation Analysis

The CCA life cycle described above is specific for the Trade networks. Similar frameworks were created for the other networks, which use different types of annotations. The steps performed are similar, but they use different parameters and one different preprocessing script that converts the indicators to a CSV file. It must be noted that the Python scripts performing the above steps are based on scripts created by Darren Davis for the GDV-based CCA.

3.2.8 Unit testing

Throughout the project we implemented a small suite of unit tests that checks the basic functionality of the GCV signature computation. We used the `BOOST` unit testing framework written in `C++` because of the following reasons:

- The core algorithm that computes the GCV signature has also been written in `C++`, which made integration easy.
- The `BOOST` testing framework has very useful features such as:
  - Grouping test cases into suites.
  - The ability of running multiple, independent tests in parallel.
  - The possibility of seeing the progress of long and complex tests.
- There exists a variety of online tutorials and learning materials on the `BOOST` testing framework

We wrote unit tests that check for the correctness of the GCV signature on PPI, Metabolic and World Trade networks. Moreover, we also tested the GCV on a small number of toy networks and compared the results with GCV signatures that were calculated by hand. Furthermore, we also tested the parallel computation in the following manner:

1. For a given input network, we compute the GCV signature list several times, using an increasing number of processes.
2. We compare each of the generated GCV lists for consistency.

This parallelisation test actually helped us discover a bug in the code caused by a null pointer exception, which only occured when 2 or more processes were used. This only affected a few of the processes that accessed the illegal area of the virtual memory and stopped their execution. As a result, the resulting GCV dump was incomplete, and this aspect was not immediately obvious without a close examination of the GCV list.

The unit tests in our suite can be run using the Makefile command: `make test`. This will execute all the tests in the test suite. One can also run individual tests or a subset of all the tests in the suite. Running two tests on the WTN network using 16 and 32 processes for the GCV computation produces an output similar to the one in figure 3.9.

```
Running 2 test cases...

Running: ./e_gdv trade_2010_thresholded.gw test_bank/
   trade_2010_thresholded 16
Finished parsing the LEDA file
Waiting on the children ...Children have finished processing.
   Assembling the files...Running: cat test_bank/
   trade_2010_thresholded.0* > test_bank/trade_2010_thresholded.
   ndump2;rm test_bank/trade_2010_thresholded.0*

Test successful

Running: ./e_gdv trade_2010_thresholded.gw test_bank/
   trade_2010_thresholded 32
Finished parsing the LEDA file
Waiting on the children ...Children have finished processing.
   Assembling the files...Running: cat test_bank/
   trade_2010_thresholded.0* > test_bank/trade_2010_thresholded.
   ndump2;rm test_bank/trade_2010_thresholded.0*
%
Test successful

*** No errors detected
```

Figure 3.9: Command-line output when running two unit tests on the 2010 World Trade network using 16 and 32 processes. For each test, the input network LEDA file is first parsed and then the parent waits until all children finish their processing. When the children processes finish their work, their output files are assembled together and checked for consistency.

Chapter 4

Applications

4.1 Initial Experiments

4.1.1 Average Network GCV

The first experiments we conduct with the novel GCV signature are comparisons of average network GCV signatures. For each node in the given network, the normalised GCV signature is calculated and then averaged over all the nodes in the network. Figure 4.1 shows the results for three different networks: A Human PPI network, a Human Metabolic network and a 2010 World Trade network.

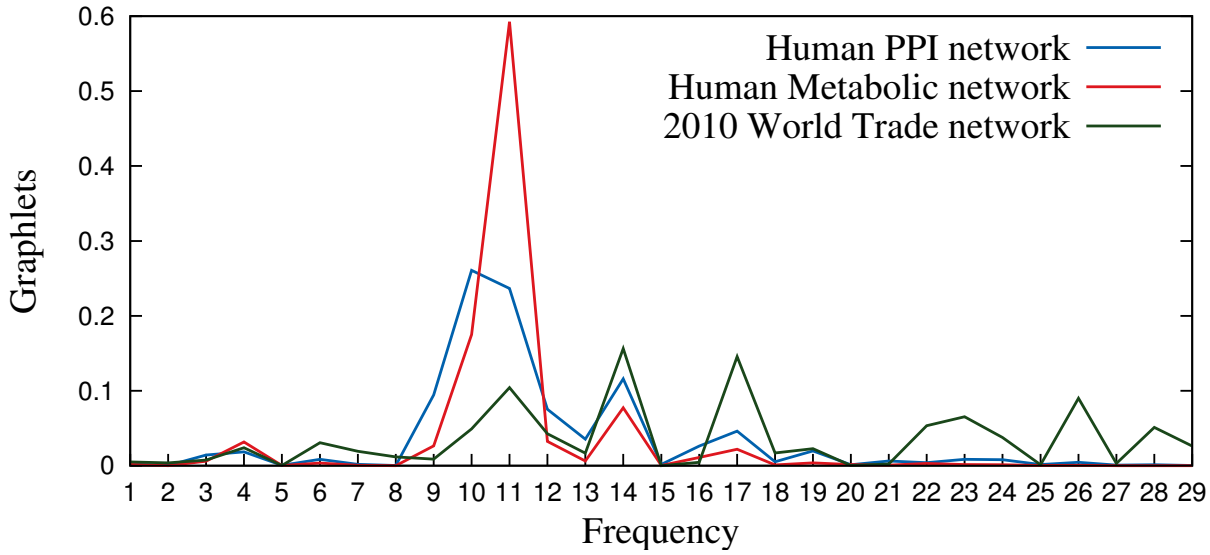


Figure 4.1: Comparison of the average GCV signatures for three different real networks: a PPI network, a Metabolic network and a 2010 World Trade network (WTN). There is a considerable discrepancy in the values of graphlets {9,10,11} across the network types. Moreover, the WTN is the only network in which graphlets {22,23,24,26,28,29} are represented.

We observe in figure 4.1 that there are slight differences between the normalised GCV of the three networks analysed. More precisely, Graphlets {9,10,11} seem to discriminate well between them, with the Metabolic network having the highest number of G_{11} graphlets and the WTN having the least. All these graphlets are sparse 5-node graphlets that have 4 edges each. The reason why the Metabolic network has a lot of graphlets G_{11} (a claw of 5 nodes) is because it is made of long metabolic paths that intersect with each other. This is best represented by graphlet G_{11} which is made of a central node and several satellite nodes. Moreover, the WTN also seems to have relatively more graphlets {22,26,28,29,22,23,24} compared to the other networks. The

reason for this is because the WTN is a dense network and all those graphlets are relatively dense compared to graphlets {9,10,11, ...} that have few connecting edges.

4.1.2 Random Networks

After we performed comparisons of the average GCV of real networks, our next step is to experiment with the following random network models:

1. Erdős-Rényi [13] (ER)
2. Erdős-Rényi (with preserved[1] degree distribution) (ER-DD)
3. Geometric networks [14] (GEO)
4. Scale-free Barabási-Albert – preferential attachment [15] (SF)
5. Stickiness index-based [16] (STICKY)

The corresponding labels (ER, ER-DD, GEO, SF, STICKY) will be used throughout this section to refer to each of these models. We generate 30 different models for every network (Metabolic, PPI, WTN) and random network generating algorithm (ER, ER-DD, etc ..) resulting in 150 total networks. Afterwards, in order to give a measure of precision to the GCV signature of random networks, we calculate the standard deviation for each of the values of the GCV.

The results obtained for the Human PPI network are shown in figure 4.2. For the Human PPI network, we notice that all the random models apart from ER-DD have very low standard deviations for all the graphlet frequencies. The graphlets where the ER-DD networks exhibit some degree of randomness are {10, 11}. On the other hand, the geometric random graphs are the only ones which contain some of the dense 5-node graphlets at the right-end of the spectrum {23,24,26,28,29}. Moreover, the ER random graphs only contain graphlet G_1, which is a $P3$. The reason for this is because ER is a rudimentary random graph model that is unable to capture the underlying complexity of the original graph. The random networks that seem to best approximate the original networks are the Scale-free[2] and the Stickiness-based. These results are confirmed by the *Relative Cluster Frequency Agreement* in section 4.1.3.

[1] The "stubs" method enables the Erdős-Rényi graph to preserve the degree distribution of the real network.
[2] Barabási-Albert Preferential Attachment

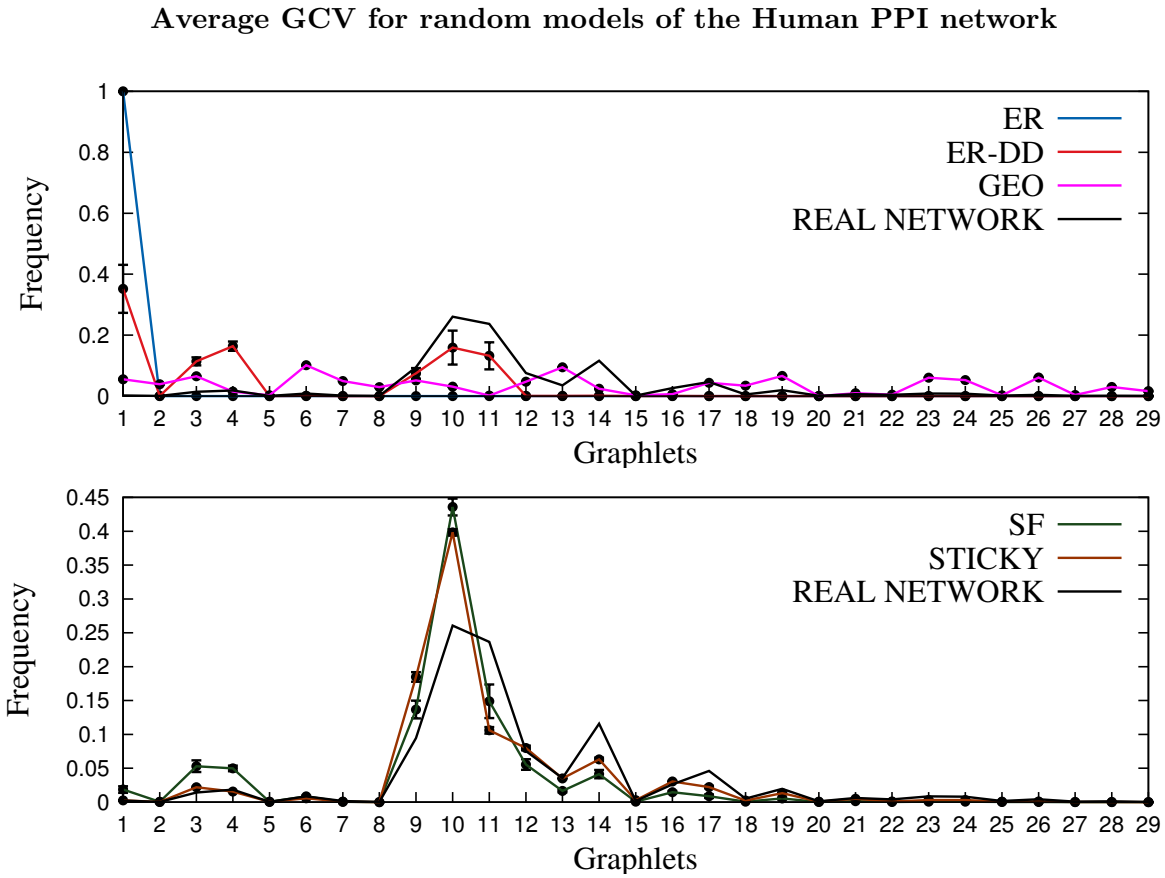


Figure 4.2: Average GCV for the Human PPI network, including the standard deviations displayed as vertical error bars. The random models analysed are: Erdős-Rényi (ER), Erdős-Rényi with preserved degree distribution (ER-DD), Geometric (GEO), Scale-free Barabási-Albert – Preferential Attachment (SF) and Stickiness-based (STICKY). The length of one vertical bar is equal to one standard deviation σ. We assume that the samples are normally distributed with mean μ and variance σ^2

Figure 4.3 shows the average GCVs for the Metabolic networks and the corresponding random networks. We notice that the metabolic networks have a slightly different signature compared to the PPI networks. First of all, they have more graphlets G_{11} but less graphlets G_{10}. Secondly, for this class of networks the ER-DD random network seems to be a better approximation according to the GCV signatures, especially for graphlet types {10,11,12}. The fact that ER-DD is the best approximation for the Metabolic network is again confirmed in section 4.1.3.

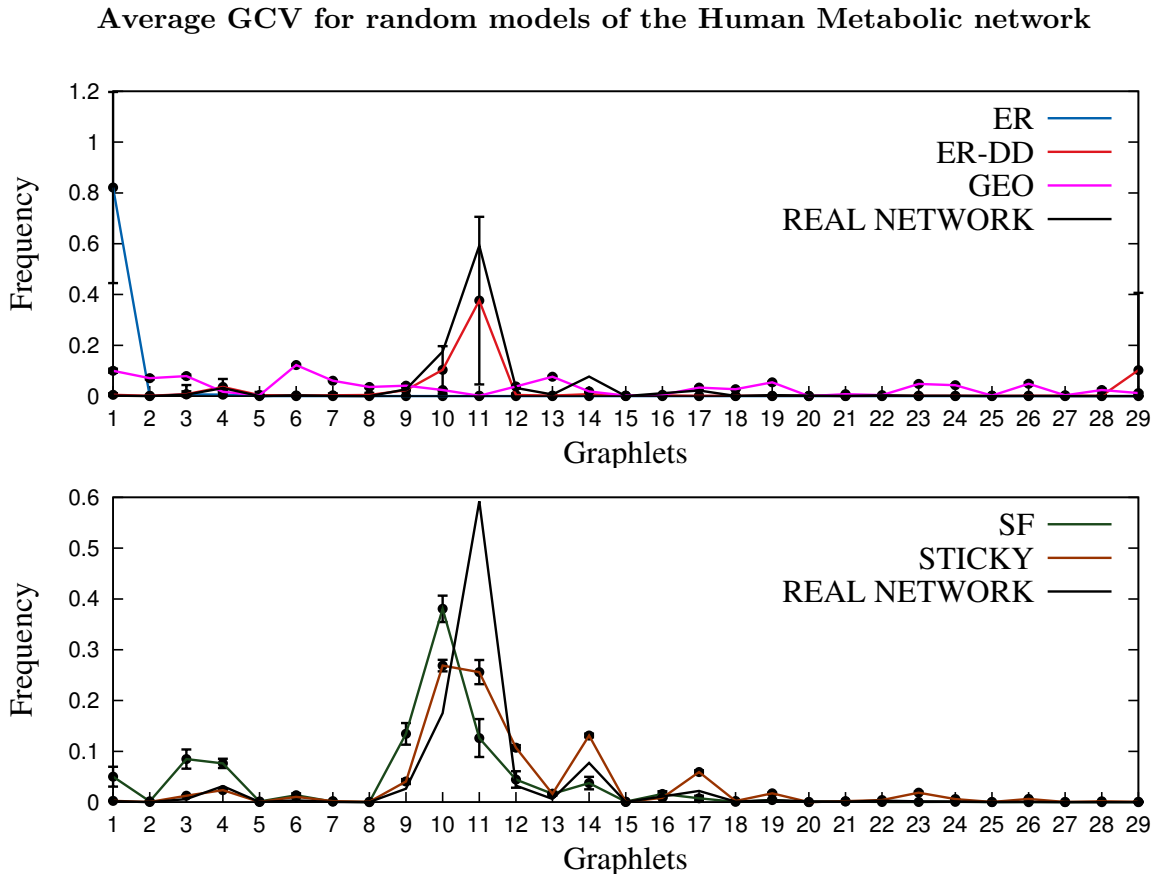


Figure 4.3: Average GCV for the Human Metabolic network, including the standard deviations displayed as vertical error bars. The random models analysed are: Erdős-Rényi (ER), Erdős-Rényi with preserved degree distribution (ER-DD), Geometric (GEO), Scale-free Barabási-Albert – Preferential Attachment (SF) and Stickiness-based (STICKY). The length of one vertical bar is equal to one standard deviation σ. We assume that the samples are normally distributed with mean μ and variance σ^2

Figure 4.4 shows the average GCVs for the WTNs and the corresponding random graphs. Surprisingly, for this type of networks we see a greater variety in the frequencies of graphlets, with graphlets in the 15–29 range now being much more represented than in the biological networks. The simple Erdős-Rényi model has a large variance for the frequency of graphlets {3,9,10}. On the other hand, the Erdős-Rényi graphs with preserved degree distribution offer a good GCV signature approximation, especially for graphlets in the range {9-18}, which are the 5-node graphlets at the sparse end of the spectrum. The Stickiness-based random graphs also offer a good approximation, a result that is confirmed by the *Relative Cluster Frequency Agreement* in section 4.1.3.

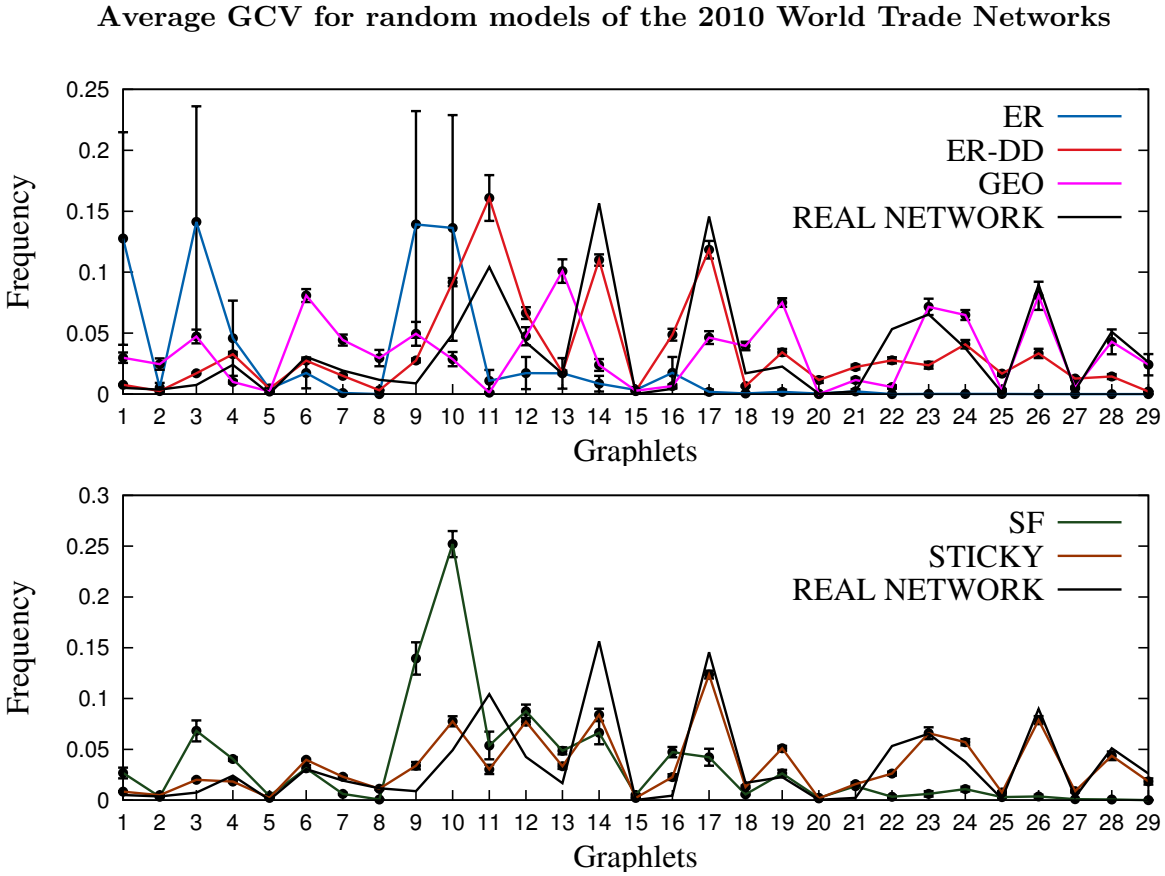


Figure 4.4: Average GCV for the 2010 WTN, including the standard deviations displayed as vertical error bars. The random models analysed are: Erdős-Rényi (ER), Erdős-Rényi with preserved degree distribution (ER-DD), Geometric (GEO), Scale-free Barabási-Albert – Preferential Attachment (SF) and Stickiness-based (STICKY). The length of one vertical bar is equal to one standard deviation σ. We assume that the samples are normally distributed with mean μ and variance σ^2

4.1.3 Relative Cluster Frequency Distance Results

The *Relative Cluster Frequency Distance* (RCFD) between two GCV vectors is a measure of how different they are with each other. It it is defined as the Euclidean norm of the difference between the two GCV vectors. A low RCFD value means that the signatures are similar to each other, while a high value means that the signatures are different. See section 3.1.3 for the exact definition of RCFD. When applied to the average GCV signature of two networks, RCFD can tell us how similar or different the two networks are. The question we are trying to answer here is: according to the average GCV signature, which random graph is best for modelling the real underlying network? The three tables from figure 4.5 show the RCFD distances between the real network and random models[3], applied to our three main classes of networks: PPI, Metabolic and World Trade.

For the Human PPI network (table (a) from fig 4.5), the random networks that best approximate the real network are the Stickiness-based (STICKY) random networks, having the smallest RCFD of 0.492, while the Scale-free Barabási-Albert (SF) graphs also offer a good approximation of the original network, having a RCFD of 0.607. The other random models perform much worse in this respect because they cannot capture all the underlying complexity in the dataset. Moreover, we also notice that the difference between the SF and STICKY GCV

[3]the real network has been used to generate these random models

signatures is really small (0.329), meaning that the generated networks are topologically similar to each other.

For the Human Metabolic network (table (b) from fig 4.5), the results are slightly different. The best-performing random networks are the ER-DD (RCFD to the real network is 0.557), built using the "stubs" method (see section 2.4.2). The second-best random network is the STICKY model which has an RCFD between itself and the Real network of 0.697. When analysing the 2010 WTN between countries(table (c) from fig 4.5), the random network with the best approximation to the real network is again the Stickiness-based network, followed closely by Erdős-Rényi with preserved degree distribution.

We therefore conclude that the STICKY random model is best at modelling PPI and WTN networks, while ER-DD is best at modelling Metabolic networks.

Model	ER	ER DD	GEO	SF	STICKY	REAL
ER	0.000	1.296	1.889	1.963	1.995	1.995
ER DD	1.296	0.000	1.554	1.018	1.233	1.191
GEO	1.889	1.554	0.000	1.413	1.406	1.311
SF	1.963	1.018	1.413	0.000	0.329	0.607
STICKY	1.995	1.233	1.406	0.329	0.000	**0.492**
REAL	1.995	1.191	1.311	0.607	**0.492**	0.000

(a) RCFD distances for the Human PPI network

Model	ER	ER DD	GEO	SF	STICKY	REAL
ER	0.000	1.499	1.610	1.709	1.804	1.807
ER DD	1.499	0.000	1.430	1.040	0.799	**0.557**
GEO	1.610	1.430	0.000	1.356	1.438	1.648
SF	1.709	1.040	1.356	0.000	0.784	1.054
STICKY	1.804	0.799	1.438	0.784	0.000	0.697
REAL	1.807	**0.557**	1.648	1.054	0.697	0.000

(b) RCFD distances for the Human Metabolic network

Model	ER	ER DD	GEO	SF	STICKY	REAL
ER	0.000	1.134	1.198	0.663	1.172	1.336
ER DD	1.134	0.000	1.065	0.819	0.490	0.572
GEO	1.198	1.065	0.000	1.099	0.609	0.873
SF	0.663	0.819	1.099	0.000	0.896	1.161
STICKY	1.172	0.490	0.609	0.896	0.000	**0.465**
REAL	1.336	0.572	0.873	1.161	**0.465**	0.000

(c) RCFD distances for the 2010 World Trade network

Figure 4.5: The RCFD distances for (a) Human PPI network (b) Human Metabolic network (c) 2010 WTN and five model networks: Erdős-Rényi (ER), Erdős-Rényi with preserved degree distribution (ER DD), Geometric (GEO), Scale-Free - Barabási-Albert Preferential Attachment (SF) and Stickiness-based (STICKY). For each network class, we have calculated not only the distance between every pair of random network models, but also the distance between every random network model and the real network which was used to generate the random models.

4.2 World Trade networks

After running initial experiments that study the average GCV of a network, we performed experiments that were specific to each of the network classes. In this section, we present the main results obtained for the World Trade Networks (WTNs). A brief summary of these networks is given in section 2.7.3.

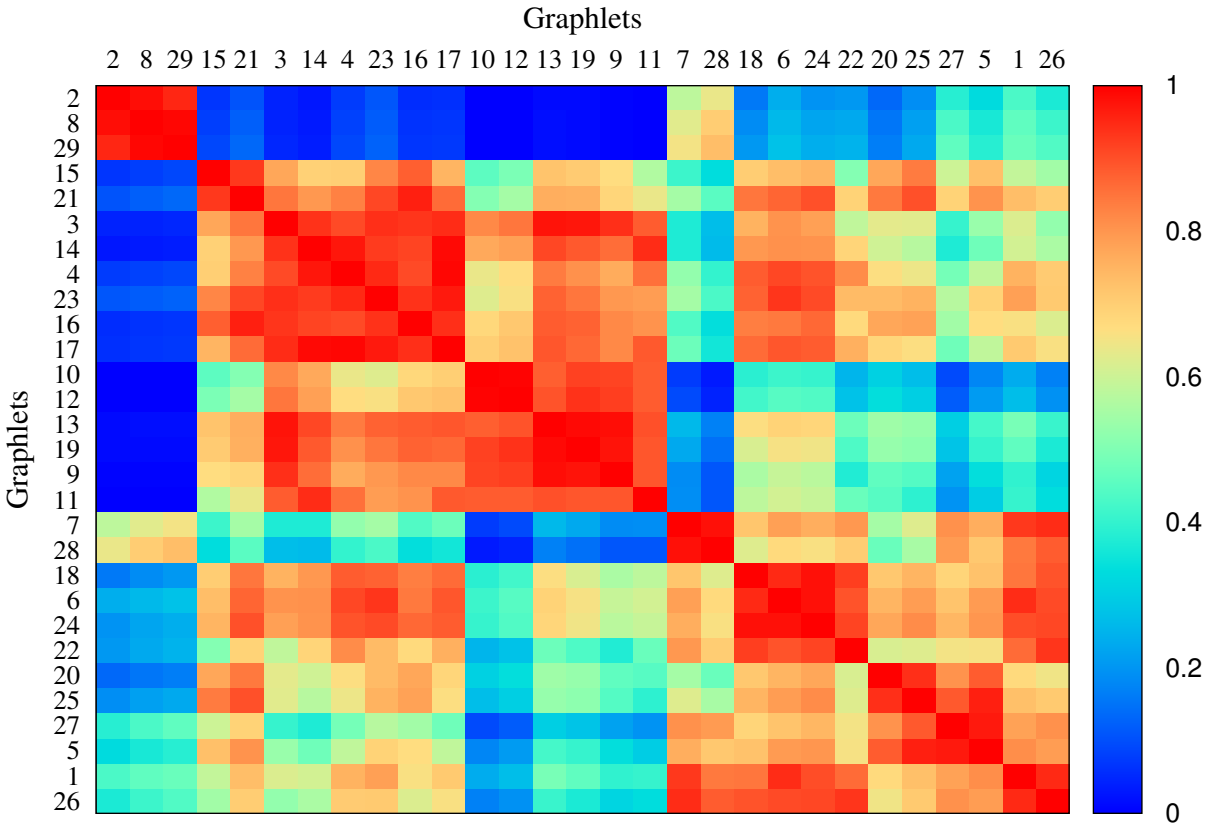


Figure 4.6: Pearson's GCV correlation matrix for the 2010 WTN. It has been normalised with feature scaling and a 3^{rd} degree polynomial, and then hierarchically clustered with complete linkage.

Figure 4.6 shows the *Pearson's GCV Correlation matrix* for the 2010 WTN. This correlation matrix was normalised with feature scaling and a 3^{rd} degree polynomial function. For details on how this has been calculated see the methodology section 3.2.3. Other polynomial functions have been tested, but the 3^{rd} degree polynomial was the most effective in emphasising the clusters of graphlets that are formed on the diagonal. These clusters of graphlets are as follows:

- Cliques cluster made of graphlets {2,8,29}.
- A cluster that is made of graphlets {15,21,3,14,4,23,16,17,10,12,13,19,9,11} which can be split into 2 further sub-clusters:
  - P_4 cluster made of graphlets {15,21,3,14,4,23,16,17}. These are all graphlets that contain a P_4 (path of 4 nodes, graphlet G_3).

  - Claw[4] cluster made of {10,12,13,19,9,11}. These graphlets all contain a C_3 (claw on 3 nodes, graphlet G_4)

- A cluster that is made of graphlets {20,25,27,5}. These graphlets all contain an S_4 (cycle of 4 nodes).

- Another set of graphlets that correlate is {18,6,24,22,1,26}. The reason why graphlets {1,26} have been added is because they also correlate with the other cluster, even if they are not right next to them in the heat map. These all contain at least one P_3 (path of 3 nodes).

Now that we know which graphlets cluster together, we will use these results in the subsequent CCA analysis in section 4.2.2.

4.2.1 Correlation matrix change during 1962–2010

The results from the previous section were concerned with the correlation matrix of the 2010 WTN. However, we are also interested to see how graphlets correlate in WTNs from other years. We therefore compute the Pearson's GCV correlation matrix for all the yearly WTNs in the period 1962–2010. Using the 49 different correlation matrices, we then compute the *change in the correlation matrix* during the respective time frame. In order to calculate the change in correlation matrix between year Y and $Y+1$, we simply subtract in a pairwise manner the two matrices and then return the sum of squares of all the elements in the matrix. For the exact formula used see equation 2.6 from section 2.5.3.

We then tried to find out if there is any correlation between the network topology and Crude Oil price. If one of these attributes changes, it might be possible that the other reacts. However, this might happen with a certain number of years delay. In order to account for this, we shift the vector of GCV correlation change by [-2,-1,0,1,2] years. For each of these 5 cases, we calculate the *Spearman's rank correlation coefficient* and the respective p-value for the following vectors:

- one 48-element vector containing the *change in Pearson's GCV correlation matrix*

- one 48-element vector containing the *change in the price of Crude Oil*

The best correlation is obtained when the vector of GCV correlation is shifted by -2 years. This scenario is plotted in figure 4.7. Surprisingly, the oil change in inversely correlated with the change in network topology: the Spearman's rank correlation coefficient is -0.49, having a p-value of 0.0004. Since the p-value is smaller than 0.05, the result is statistically significant. The explanation for this is as follows: high oil prices generally have a large negative impact on the global economic growth. Slower growth leads to diminished investment-related activity in the countries affected, which in turn deters the creation of new trading partners. This implies that the network structure remains mostly unchanged, a fact that results in a low GCV correlation change. Moreover, because the best correlation is obtained when the change in GCV is shifted by -2 years, this might suggest that changes in network structure cause the Crude Oil price to change. However, we did not have time to perform more supporting experiments in order to validate the causality aspect of this claim.

Furthermore, there are several major economic events for which we do not have a big change in the topology of our network, such as the 2007 sub-prime mortgage crisis or the 1997 Asian financial crisis. This implies that the unnormalised Pearson's GCV correlation matrix is not affected by these major events. Similar results that use a normalised version of the GCV are better correlated with global economic and social events (see section 4.2.7).

[4] A claw C_n is a graphlet that has a central node and $n-1$ satellite nodes connected to it. See section 2.1.1 for a full definition.

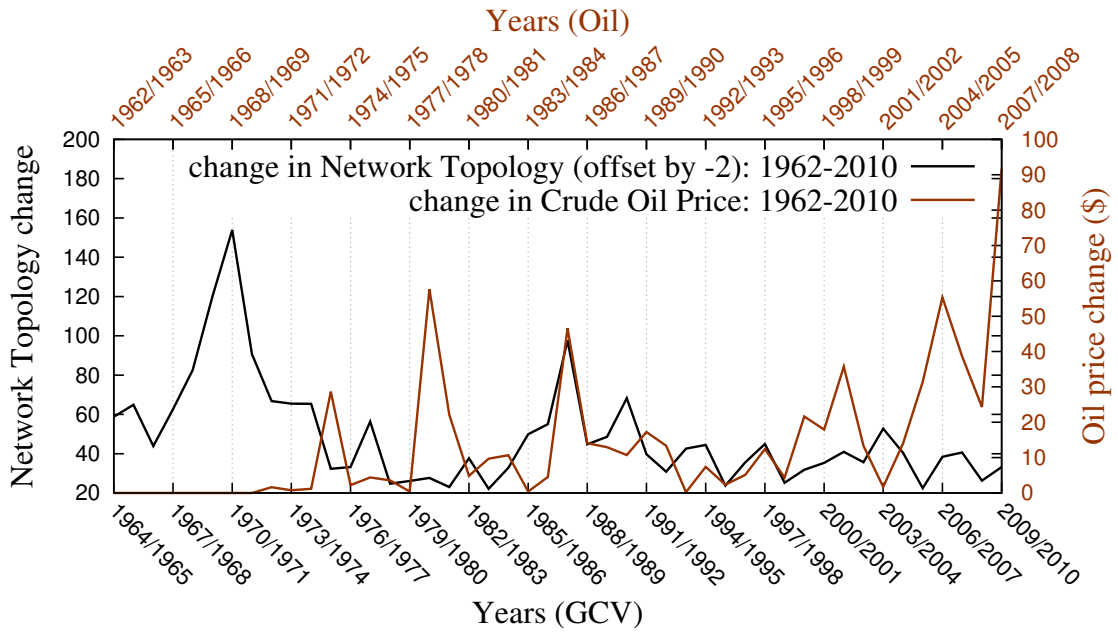


Figure 4.7: Evolution of WTN structure during 1962–2010 using the unnormalised GCV. Plotted in black is the change in GCV correlation that has been offset by -2 years, while the change in Crude Oil Price is plotted in brown. Spearman's rank coefficient between oil price change and change in network topology is -0.49 with a p-value of 0.0004. This suggests that when the change in GCV correlation between countries changes, then the oil price stays the same. The top and left axis tics correspond to the Oil curve, while the bottom and the left axis tics correspond to the network topology curve.

4.2.2 CCA - 1980–2010 World Trade networks

After correlating graphlets from the GCV vector with each other in order to see which one of them have a similar behaviour, the next step is to correlate the GCV vectors with the economic indicators of a country. This can be done using *Canonical Correlation Analysis* (CCA) which is described in section 2.6. The two variates we correlated are:

1. the X variate containing economic indicators (GDP per Capita, Level of Employment). See section 2.7.3 for details about all the economic indicators used.

2. the Y variate containing the unnormalised GCV vectors for each country.

The CCA analysis uses data for 119 countries over a period of 30 years (1980-2010). Each country-year pair represents one sample for which we have both economic indicators (X variate) and the GCV (Y variate).

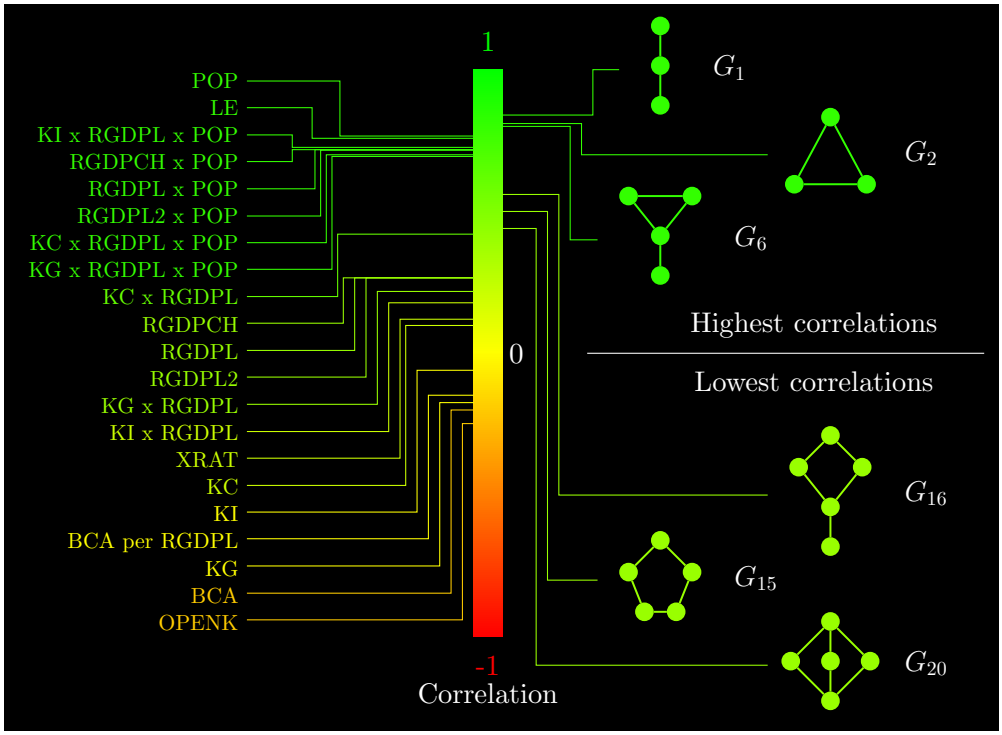


Figure 4.8: Canonical Correlation Analysis between economic indicators and the GCV signature. Only the graphlets that have the highest respectively lowest cross-loadings are shown in the picture. However, all the graphlets have positive cross-loadings, with the lowest cross-loading having a value of 0.44. Openness (OPENK), Balance Current Account (BCA) and a few other indicators correlate negatively with all the graphlets, because their cross-loadings have different signs. On the other hand, the rest of the indicators such as Population (POP), Level of Employment (LE) and GDP per capita (RGPDL, RGDPCH) correlate positively with all the graphlets, since their cross-loadings have the same sign. The overall correlation is 0.89 with a p-value smaller than 0.0001. This result suggests that big and wealthy countries have a large network of trading partners that is rich in graphlets.

CCA results are presented in figure 4.8. A supplementary table with the list of all the cross-loadings can be found in figure B.1 in the appendix. This result shows that all the graphlets correlate positively[5] with some indicators such as Population, Level of Employment or GDP per capita and negatively with Trade Openness and Balance of Current Account. This means that big and rich countries that have a high population and GDP per capita have a neighbourhood rich in graphlets, while small and poor countries with account deficits have a neighbourhood sparse in graphlets. The population of the country seems to be quite an important factor for determining whether it will have a rich neighbourhood because of the following two reasons:

- In the X variate, population has the weight with the highest magnitude: 0.766
- Most of the other economic indicators that have a high weight are obtained by multiplying population with other indicators. This is also the case in a similar CCA Analysis of Yaveroğlu et al. using graphlet orbits [37].

4.2.3 Economic Integration

We now try to find out if the level of *Economic Integration* of a country is positively correlated with dense graphlets and negatively correlated with sparse graphlets. This is something to be

[5] An element from the X variate correlates positively with another element from the Y variate if and only if their cross-loadings have the same sign

expected, because when a country is part of a strong trading bloc, it's neighbours have a higher probability of doing heavy trade with one another. This is because there is incentive for the country to trade more with the partners from the same bloc, that are already trading a lot with each other. This would in turn result in denser graphlets in the neighbourhood of that country. The idea of correlating the GCV with the integration level of a country was entirely mine.

For a given country, there exist several stages of economic integration. One possible classification is the following:

1. no economic integration
2. Multilateral Free Trade Area (e.g. AFTA, CEFTA, CISFTA)
3. Customs union (e.g. EAC, EUCU, MERCOSUR)
4. Common market (e.g. EEA, EFTA)
5. Customs and Monetary Union (e.g. CEMAC/franc, UEMOA/franc)
6. Economic union (e.g. CSME, EU)
7. Economic and monetary union (e.g. CSME + EC dollar, EU + euro)

We found some preliminary data on the Internet which labels each country using the most advanced integration agreement it signed. Using this data, we computed an integration index (1-7) for each country and correlated it with the GCV signature using CCA.

Figure 4.9 presents these preliminary CCA results. They confirmed our initial expectations, with dense graphlets correlating most with the integration index, while the sparse graphlets correlating least. However, since the source of the data that was used for this experiment could not be verified, we searched for an official index that quantifies political integration for each country around the world. Although we haven't found an index that uses the 6-level scale that we previously mentioned, we found some indices on the *World Trade Organisation* website that measure the number of *Regional Trade Agreements* (RTAs) of a country [64]. These RTAs are defined as trade agreements that are concluded between countries that are geographically close to each other[6]. For a given country, the number of RTAs gives us a measure of economic and political integration, since these agreements are mainly signed within trading blocks. The RTAs facilitate trade on a regional basis and can be of several types:

- A Free Trade Agreement (FTA)
- A Customs Union (CU)
- Economic Integration Agreement (EIA)
- Partial Scope Agreement[7] (PS)

The *World Trade Organisation* provides indices for each of the following classes of RTAs:

- Goods RTAs: agreements that facilitate trade in goods.
- Services RTAs: agreements that facilitate liberalisation of the services market.
- Physical RTAs: actual agreements signed that cover both goods and services.[8]

[6]However, the countries do not strictly have to be geographically close in order to sign an RTA.
[7]Covers only certain types of products
[8]An RTA that covers both goods and services is also counted for Goods RTAs and Services RTAs.

Canonical Correlation		0.61882	
p-value		0.01280	
X variate		Y variate	
Integration	0.61882	G29	0.28704
		G8	0.28349
		G2	0.27862
		G22	0.26820
		G28	0.26806
		G7	0.26021
		G26	0.26011
		G18	0.24661
		G1	0.23837
		G24	0.23133
		G6	0.22969
		G4	0.22021
		G27	0.22013
		G17	0.21021
		G14	0.20631
		G23	0.20420
		G5	0.20149
		G20	0.19391
		G25	0.19122
		G21	0.19076
		G16	0.19026
		G11	0.18330
		G3	0.17730
		G15	0.16814
		G13	0.16764
		G19	0.16051
		G9	0.15087
		G12	0.14886
		G10	0.14638

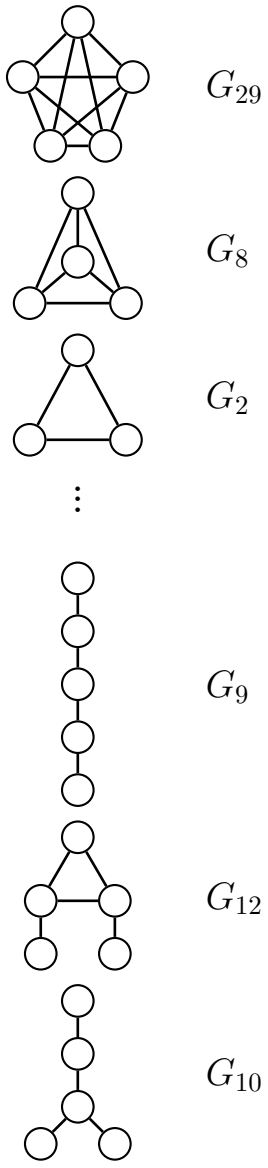


Figure 4.9: CCA results between the Integration index (X variate) and the GCV (Y variate). The Integration index of a country is positively correlated with all the graphlets. However, the strongest correlation is with dense graphlets such as cliques {29,8,2} because they have the highest weight, while sparser graphlets {10,12,9} have the lowest weight. The overall correlation is 0.61, with a p-value of 0.01, suggesting the correlation is statistically significant.

The results of the Canonical Correlation Analysis are shown in figure 4.10. As we expected, the Goods and Physical RTAs are correlating positively with dense graphlets such as cliques {2,8,29} and negatively with sparse graphlets such as {10,11,9,12}. This suggests that once a country is acceding to a trading block, its entire trade shifts towards its partners within the block, which trade mainly with each other, hence the dense graphlets in the neighbourhood structure. Surprisingly, the services EIAs are not showing this correlation, having a small but positive weight of 0.00187. This implies that when a country negotiates services EIAs, that doesn't result in the total trade getting redirected towards the signatories of the EIAs. Further research needs to be done in order to explain why this is the case.

4.2.4 Revision of GCV - normalisation

The results presented in previous sections used the un-normalised GCV vector which contained the total number of graphlets of each type found in the neighbourhood of a node. However, we also tried running all the experiments with the normalised GCV. See definitions 39 and 40 from section 3.1 for the un-normalised and normalised GCV respectively. The normalised GCV contains the proportion of each graphlet in the neighbourhood of a node.

All the experiments performed in this project have been run with both the un-normalised and normalised GCV versions. However, the only insightful results with the normalised GCV have been obtained for the WTN. The next two sections present the Pearson's Correlation matrix and the Canonical Correlation Analysis results using the normalised GCV signature.

Canonical Correlation		0.81460	
p-value		0.00000	
X variate		Y variate	
Services EIAs	0.00187	G10	0.04910
Physical RTAs	-0.14733	G11	0.04673
Goods RTAs	-0.15447	G9	0.04035
		G12	0.03890
		G20	0.03736
		G14	0.03384
		G13	0.03298
		G16	0.03295
		G15	0.02992
		G19	0.02690
		G17	0.01933
		G18	0.01594
		G21	0.01429
		G3	0.01405
		G4	0.01362
		G25	0.00856
		G22	0.00484
		G24	0.00225
		G23	-0.00702
		G27	-0.01388
		G5	-0.01887
		G6	-0.02347
		G26	-0.03082
		G1	-0.06278
		G7	-0.07024
		G28	-0.07331
		G29	-0.14967
		G8	-0.15671
		G2	-0.16970

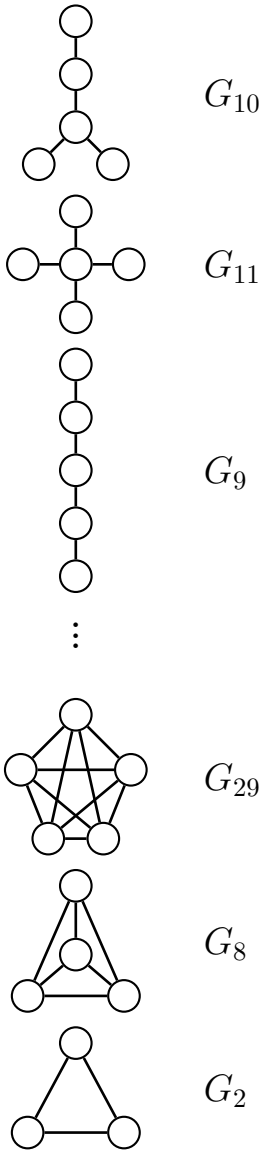


Figure 4.10: Canonical Correlation Analysis on Trade Integration using the number of Regional Trade Agreements as an indicator of trade integration. The Goods and Physical RTAs correlate positively with dense graphlets such as {2,8,29} because the weights have the same signs. At the other end, sparse graphlets such as {10,11,9,12} correlate negatively with Goods and Physical RTAs. The canonical correlation is 0.81, having a p-value of 0.

4.2.5 Pearson's normalised GCV correlation matrix

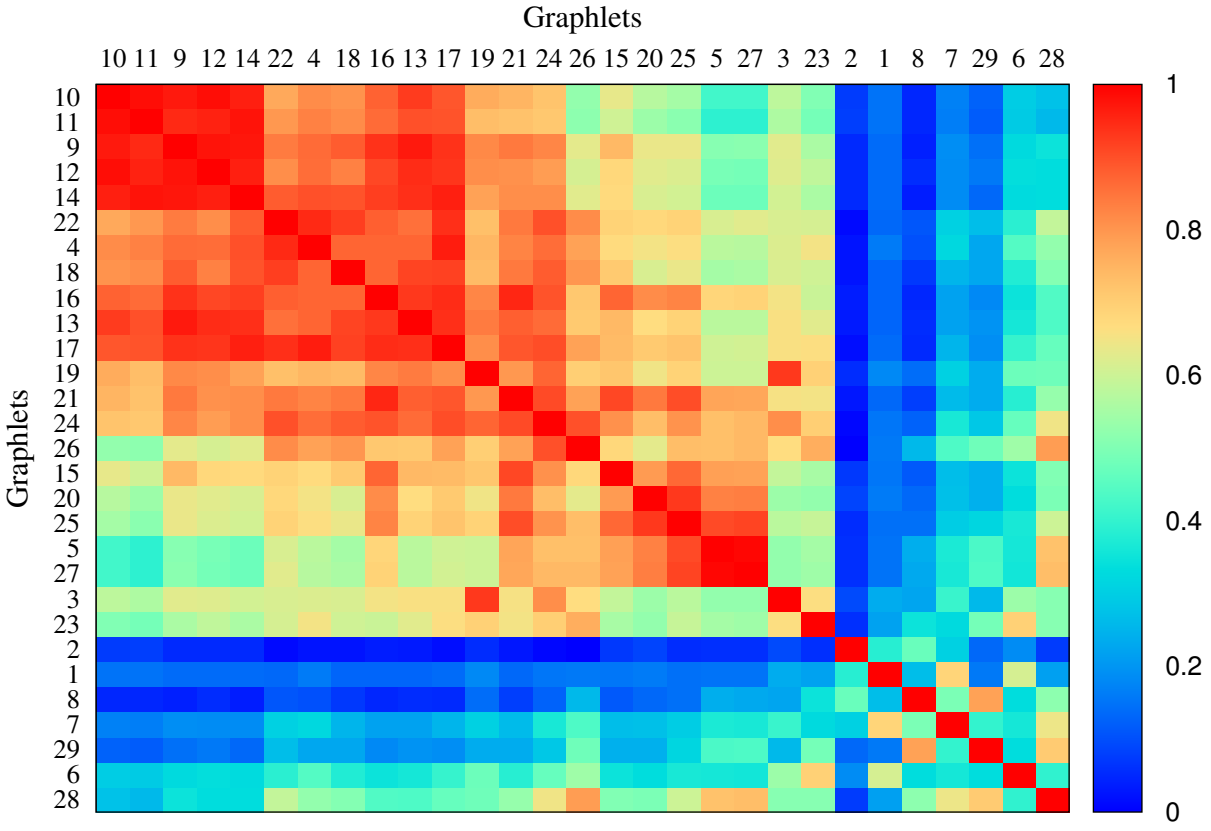


Figure 4.11: Pearson's GCV correlation matrix for the 2010 WTN using the normalised GCV. The heat map is normalised only with feature scaling.

Figure 4.11 shows the Pearson's correlation matrix on the 2010 WTN that is calculated using the normalised GCV signature. We can observe several clusters of graphlets that have been formed along the diagonal:

- **A**: Cluster made of graphlets {10,11,9,12,14}. These are all sparse graphlets that have 4 or 5 nodes.
- **B**: A slightly similar cluster that is also correlated with the one above is {22,4,18,16,13,17}. These graphlets all contain a C_4 as a subgraph.
- **C**: Another cluster is formed by graphlets {5,25,27}. These graphlets all contain a cycle of length 4 (S_4).

However, we also notice that this time the cliques {2,8,29} do not cluster together. Cliques used to cluster together when using the un-normalised GCV signature (see figure 4.6). We do not have a clear explanation for this behaviour and further research needs to be done into this.

4.2.6 Normalised GCV - Canonical Correlation Analysis

After finding out which graphlets cluster together, we run Canonical Correlation Analysis using the same methodology described in section 4.2.2, this time using the normalised GCV signature.

Figure 4.12 shows the results of the CCA, while the supplementary table with all the correlations can be found in the appendix figure B.2. The correlation is high $\rho = 0.94$ and the p-value is 0.0, suggesting that the result is statistically significant.

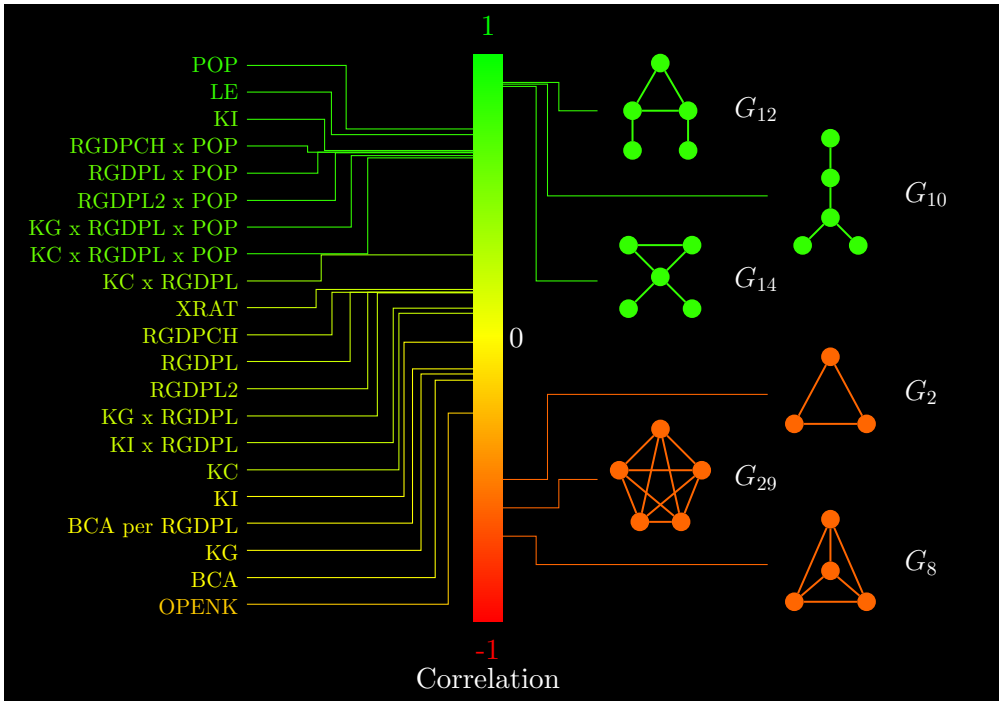


Figure 4.12: CCA between economic indicators and the normalised GCV signature. Only the graphlets that show the strongest respectively weakest correlations are shown. One can notice that graphlets {12,10,14} are relatively sparse, while graphlets {2,29,8} are dense, being cliques. The sparse graphlets are correlated with the good economic indicators (in green) such as Population (POP), Level of Employment (LE) and GDP per Capita (RGDPL), while dense graphlets are correlated with bad indicators (in red) such as the Balance of Current Account (BCA). The canonical correlation $\rho = 0.94$ and the p-value is smaller than 0.0001, suggesting that the result is statistically significant.

The good indicators such as population (POP), level of employment (LE) and GDP per capita (RGDPL) are positively correlated with the graphlets {12,10,14,17,9, ...}[9]. On the other hand, the bad indicators such as the balance of current account (BCA) correlate positively with graphlets {8,29,2,7,1,28}. Graphlets {10,12,14,9} on the positive side of the spectrum have also clustered together in the Pearson's correlation matrix (section 4.2.5). We first notice that graphlets {8,29,2,7,1,28} represent cliques {8,29,2} or almost cliques {7,1,28}. Since these graphlets are very densely connected, this suggests that the trading partners of small and poor countries are trading heavily with each other or form highly connected clusters. As a result, we deduce that *the majority of the trading partners of small and poor countries are the big and rich countries that are always trading heavily with each other.*

This theory seems to be confirmed by taking a few small and poor countries and looking at their trading partners. Note that since the network has only 119 countries, the poorest countries from Africa or South Asia have already been filtered out.[10] Therefore, let us consider Morocco a small and poor country relative to the others, although in reality it considered to have a medium level of development. Morocco's main trading partners are: Saudi Arabia, China, France, USA,

[9]The CCA figure 4.12 only shows the graphlets that have the strongest and weakest cross-loadings. See figure B.2 in the appendix for a list of cross-loadings for all the graphlets and economic indicators.

[10]This is the case because the network has been thresholded at an 85% level. See section 2.7.3 for more details.

Spain, Germany and Italy. These countries are big and rich and every single pair of them clearly trade with each other. Similar results have been observed for other countries such as Uruguay. Moreover, all of Morocco's trading partners are part of G20, a club of big and wealthy countries that collaborate with each other on economic matters. This leads us to a second theory: *since the trading partners of a small and poor country form highly connected clusters, these clusters represent big and rich economic groups such as G8, G20, OECD*[11] *or Paris-club*[12]. This theory can be validated by selecting a few countries and looking at their neighbours. For example, the biggest trading partners of Tunisia are Germany, France and Italy, all part of G8, G20, OECD and Paris-club.

Regarding the first group of graphlets (i.e. {12,10,14, ...}), we notice that all of them are sparse graphlets that contain relatively few edges. Having the sparse graphlets at one end of the spectrum and the dense graphlets at the other suggests that the graphlets are roughly ordered according to their density. Therefore, CCA shows that the sparse graphlets correlate with the good indicators such as population (POP), level of employment (LE) and GDP per capita (RGDPL) while dense graphlets correlate with bad indicators such as the balance of current account (BCA).

Now that we now know to interpret the positively weighted part of the graphlet vector as sparse graphlets, canonical correlation tells us that the trading partners of big and wealthy countries have a lot of sparse graphlets in their neighbourhood. The economic reason for this is because *big and rich countries like USA, China, Russia are trading with a lot of small, isolated countries which in turn do not trade with each other.* This theory is supported by a closer analysis with Cytoscape[13]. Using this software we found that the clustering coefficient of a country is inversely correlated with the wealth and size of that country, suggesting that big and rich countries indeed have a relatively sparse neighbourhood.

4.2.7 Normalised GCV - Correlation matrix change during 1962–2010

We also calculated the changes in Pearson's correlation matrix using the normalised GCV for the WTNs over the period 1962–2010. The results are plotted in figure 4.13 along with the changes in Crude Oil price. For this experiment we follow the same methodology as in section 4.2.1. We find that for the normalised GCV, the best results are obtained when the GCV vector is shifted with -1 year and yields a positive correlation $\rho = 0.34$ and a p-value of $p = 0.01$. These results are in contrast to the ones obtained using the original GCV signature in section 4.2.1 and at the moment we cannot give a reason why this is happening. Since the best correlation is obtained when the GCV vector is shifted with -1, this again suggests that the changes in the network structure might cause the changes in the price of Crude Oil.

[11] Organisation for Economic Co-operation and Development
[12] A group of countries that provide debt relief and debt restructuring to indebted countries and their creditors.
[13] A network analysis software that can provide useful statistics of the network data.

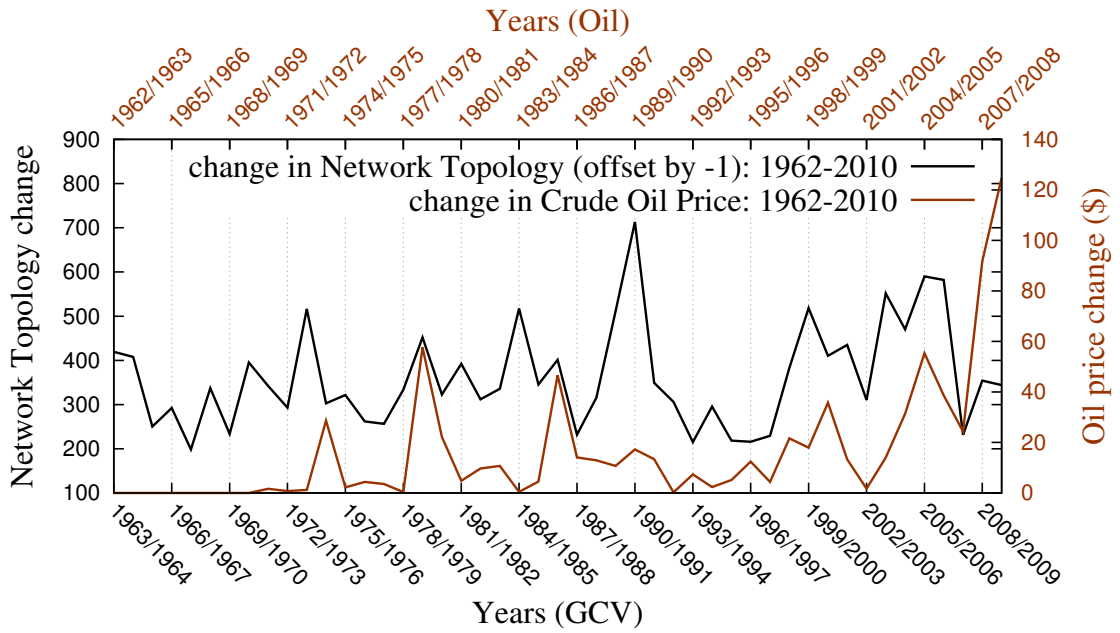


Figure 4.13: Change in WTN topology (as measured by the normalised GCV correlation matrix) versus change in crude oil price. The two plots are positively correlated, having a Spearman's rank correlation coefficient of 0.34 and p-value of 0.01. The best correlation coefficient is obtained when the change in network topology is shifted by -1 years. The top and left axis tics correspond to the Oil curve, while the bottom and the left axis tics correspond to the network topology curve.

There are several major economic and social events that have clearly affected the WTN structure. The 1970s were marked by two energy crises (1973 and 1979) that explain the two small peaks in both the topology change but also in the oil price change. Afterwards, the 1983/1984 peak in network topology change might have been caused by the early 1980s recession, which affected most of the developed world. A revival of neoliberalist economic policies around the world occurred in this period which led to reduced government intervention, lower taxes and deregulation. The peak in 1989 might be explained by the fall of communist/socialist governments in Russia, Eastern Europe and around the world accompanied by a fall in heavy industries and increased trade openness. These events have been accompanied by changes in government for some former left-wing or right-wing countries such as Russia, Poland, Chile and South Africa.

The early 1990s appear as a period of relatively low changes in oil and network topology, which reflects the overall economic stability at that time. However, bigger changes are noticed in the late 1990s, possibly started by the 1997 Asian financial crisis. By the 2000s, even bigger changes can be observed in the network topology plot that were caused by the commodities boom and rising oil prices and inflation.

4.2.8 Trade partners sparsity index

Using a combination of graphlet frequencies that are part of the GCV, we are now interested to create an index that is positively correlated with the good indicators from section 4.2.6 such as GDP per Capita (RGDPL) or Level of Employment (LE). Therefore, we take the three graphlets that have the highest correlation with the economic indicators variate {12,10,14} and the three that have the lowest correlation {8,29,2} (see figure B.2). Multiplying each of these by their respective CCA cross-loading and summing up the results gives us a *trading partner sparsity index*. The index T can formally be defined as:

$$T = w_{12}F_{12} + w_{10}F_{10} + w_{14}F_{14} + w_8F_8 + w_{29}F_{29} + w_2F_2$$

where F_i, w_i are the frequency respectively the canonical cross-loading of G_i. In order to compute the index, we use the cross-loadings obtained from CCA in figure B.2.

This index can be calculated for every country and for every year and can have both positive and negative values. It gives a measure of the sparsity of the network of the trading partners: the higher the value the sparser the neighbourhood, because the sparse graphlets have positive weights while the dense graphlets have negative weights. CCA has shown us that for a certain country a network of trading partners that has sparse graphlets indicates a healthy economy, so we expect the *trading partner sparsity index* to be high for big and wealthy countries and low for small and poor countries. We also expect the index to fluctuate during periods of economic uncertainty.

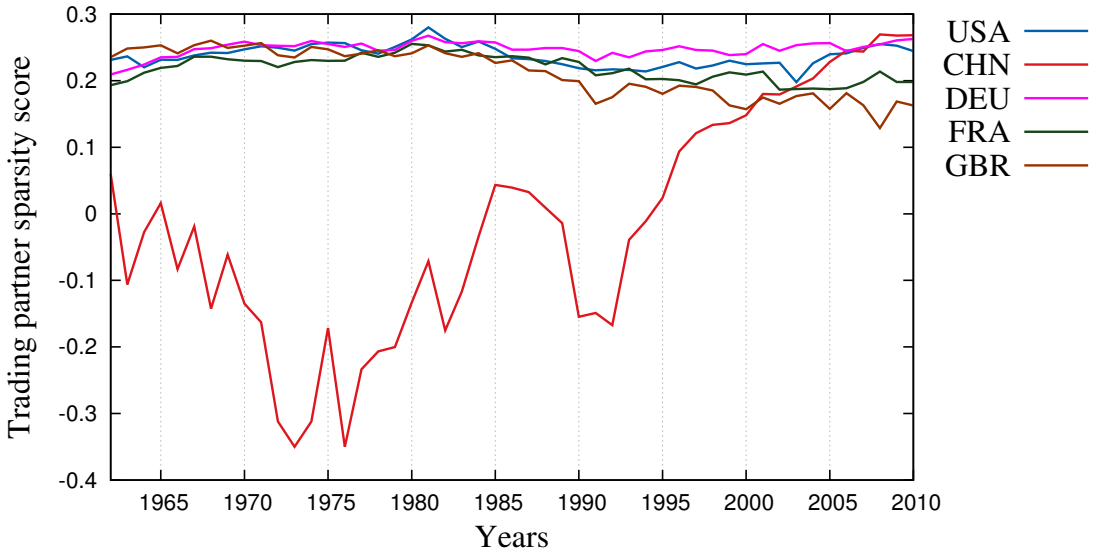


Figure 4.14: Trading partners sparsity index measured for 5 big economies: United States (USA), China (CHN), Germany (DEU), France(FRA) and the United Kingdom (GBR). The index for US, Germany, France and the UK is approximately flat over the 49-year period. However, the index of China has a downward trend over the time period 1963–1973 due to Mao Zedong's policies that harmed the economy of the country. However, the period after 1975 shows a surge that was boosted by economic reforms and growth. There is one exception in 1990–1992 right after the Fall of Communism in Eastern Europe, a global event that affected a socialist country such as China.

Figure 4.14 shows the *trading partners sparsity index* for several influential countries: United States (USA), China (CHN), Germany (DEU), France(FRA) and the United Kingdom (GBR). Throughout the 1965–2010 period, the corresponding index for the United States, Germany, France and the United Kingdom has been approximately flat, having a value of 0.2. Some small variation can be seen starting from 1990, with Germany and the United States having a slightly bigger index than France and the United Kingdom. Furthermore, for these four countries we don't observe any shocks during economic crises. On the other hand, China suffers a decrease in the trading partners sparsity index during 1965–1976, due to Mao Zedong's Cultural Revolution that resulted in a period of economic decline. However, the index increases again during 1976–1985, probably due to economic reforms that were initiated by Deng Xiaoping which helped revive the economy. Another low point is noticed in 1990–1992 right at the Fall of Communism in USSR and Eastern Europe, a global event that deeply affected a socialist state such as China.

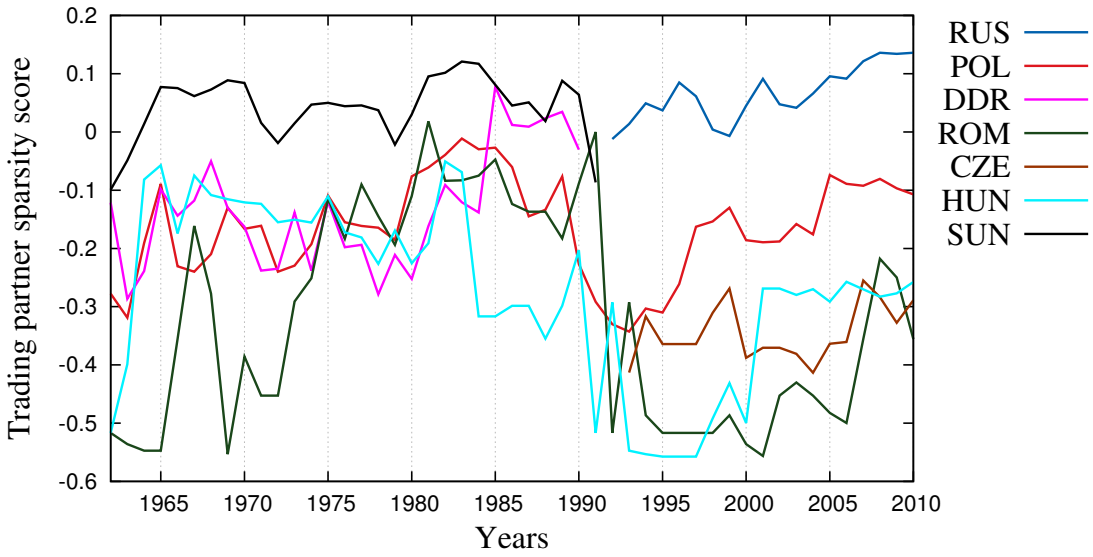


Figure 4.15: Trading partners sparsity index measured for countries from Eastern Europe: Russia (RUS), Poland (POL), East Germany (DDR), Romania (ROM), Czech Republic (CZE), Hungary (HUN) and the USSR (SUN). Most of the countries show a drop in the index after 1990 because of the Fall of Communism and the economic restructuring that took place at that time.

Figure 4.15 shows the *trading partners sparsity index* for several countries in Eastern Europe: Russia (RUS), Poland (POL), East Germany (DDR), Romania (ROM), Czech Republic (CZE), Hungary (HUN) and the USSR (SUN). In the period leading to 1990, the USSR had the highest index since it was a world superpower, while it's satellite states had a lower index. However, the Revolutions in December 1989 in Eastern Europe led to a large drop in the *trading partners sparsity index* for all these countries, a fact that is reflected by the economic situation at that time: unemployment skyrocketed and living standards fell considerably. It took some countries such as Poland of Hungary around approximately 10–15 years to reach the pre-revolutions level in the *trading partners sparsity index*.

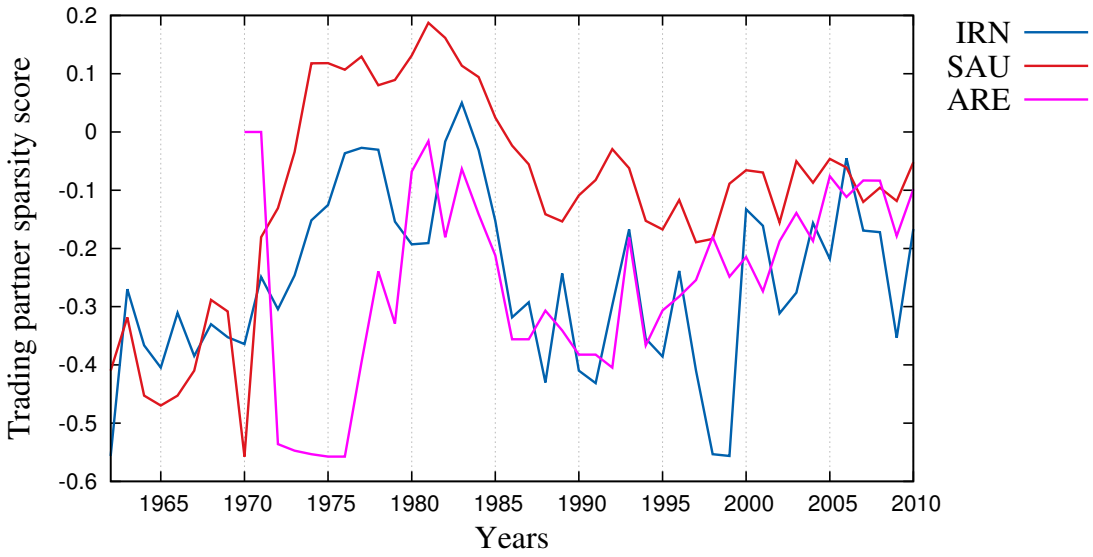


Figure 4.16: Trading partners sparsity index measured for 3 OPEC members: Iran (IRN), Saudi Arabia (SAU) and United Arab Emirates (ARE). The rise in petroleum prices and the Oil crisis in 1973 has led to a surge in the index for Saudi Arabia and Iran. Moreover, the Oil crisis of 1979 has also led to an increase in United Arab Emirate's index. However, the 1980s Oil glut that was caused by a serious surplus of oil had detrimental effects on all OPEC members, which is reflected in the drop of their trading partners density index.

Figure 4.16 shows the *trading partners sparsity index* for three main OPEC members: Iran, Saudi Arabia and United Arab Emirates. For Saudi Arabia and Iran, the rise in petroleum prices in 1970s led to a surge in it's index. However, during the 1980s the oil glut that was caused by a serious surplus of crude oil and a drop in demand had detrimental effects on all OPEC members, which are heavily dependent on the price of oil. It can also be noticed that the 1973 Oil Crisis has led to an increase in the index only for Saudi Arabia and Iran, while the 1979 Oil crisis has led to an increase in the index only for the United Arab Emirates.

4.2.9 Case study: Saudi Arabia

As we have seen in previous sections, the GCV signature can indeed capture the changes in Crude Oil prices and correlate with key economic and social events around the world. In this section we are trying to apply the same analysis but on a smaller scale, at a country level. We have selected Saudi Arabia as a major oil-exporting country, whose economy is heavily dependent on the price of oil. We are trying to find the answer to the following questions:

- Are the partners of Saudi Arabia affected by changes in Crude Oil price?
- Is the GCV of Saudi Arabia positively or negatively correlated with the Crude Oil Price?

Saudi Arabia is the world's largest oil-exporting economy and has the largest proven petroleum reserves. It is also a very influential member of the *Organisation of the Petroleum Exporting Countries* (OPEC). It's main export partners are the United States, China and Japan, while it's main import partners are China, United States and South Korea. Around 90% of it's exports consist of petroleum and related products.

We therefore calculate the normalised GCV of Saudi Arabia for each year in the period 1962–2009. Afterwards, the change in GCV between every two consecutive years is calculated using the Euclidean distance between the two vectors. Results of the GCV change along with the Crude Oil price are plotted in figure 4.17. The two plots are negatively correlated, having

a Spearman's rank correlation coefficient of -0.32 with a p-value of 0.026, which resembles the results we got for the original GCV change for the overall trade network in section 4.2.1. First of all, it must be noted that since Saudi Arabia is an oil-exporting country, it benefits massively from a rise in oil prices. However, high oil prices on the energy markets lead to less demand for petrol and provides other oil-poor countries an incentive for developing alternative sources of energy. The fact that Saudi Arabia benefits from high oil prices might explain why the change in it's trading partner network topology is inversely correlated with oil price: when the price of oil is low, Saudi Arabia always looks for new export markets and thus has a move volatile network of trading partners. On the other hand, when the price of oil is high, it means that the demand is much higher than the supply available, so Saudi Arabian oil companies prefer to export to their old trading partners, since there is no need for extra contracts, negotiations and bureaucracy.

Figure 4.17 shows that big changes in the trading partners of Saudi Arabia occurred between 1968/1969 and 1969/1970, which subsumed shortly afterwards. These might be explained as a consequence of the 1967 Oil Embargo, when Saudi Arabia and several Middle Eastern countries limited or completely stopped their oil supplies to Western countries such as the USA, UK and other European states. The result was that Saudi Arabia had to look for different export partners and that led to a change in its trading partner structure.

This experiment has also been run using the un-normalised GCV change, but it hasn't yielded a good correlation between the GCV change and the change in crude oil price. The associated p-value was also high, meaning that the result was not statistically significant. A plot and the equivalent results are given in figure A.1 in the appendix.

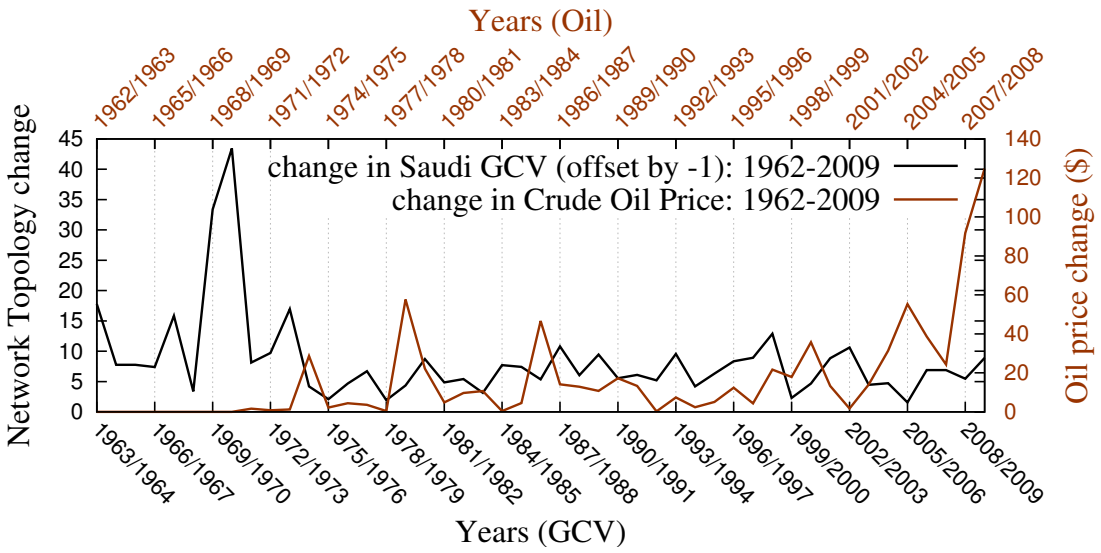


Figure 4.17: The change in the GCV of Saudi Arabia along with the change in Crude Oil price. The two plots are negatively correlated, having a Spearman's rank correlation coefficient of -0.32 with a p-value of 0.026. This correlation is obtained when the vector of changes in Saudi GCV is shifted by -1. Top and left axis tics correspond to the Oil curve, while the bottom and the right axis tics correspond to the Saudi GCV curve.

In order to find out how each of the individual elements of the GCV vector are influenced by the oil price, we apply Canonical Correlation between the GCV of Saudi Arabia and the Crude Oil price index. However, this proves to be problematic since we only have 49 samples to run the CCA on, one for every year during 1962–2010[14]. On the other hand, there are

[14]In previous CCA experiments, we used all country-year pairs that gave us in total around $119 * 29 = 3451$ samples, where 119 is the average number of countries in the network and 29 is the number of years CCA was

$29(\text{GCV}) + 1(\text{Oil Price}) = 30$ parameters that need to be estimated. This could easily overfit or yield singularities in our algorithm. Therefore, we trim down the GCV vector to only contain the essential graphlets 1-8, discarding all the 5-node graphlets. The final CCA variates are as follows:

X - short 8-element GCV of Saudi Arabia that only contains the essential graphlets (i.e. $G_1 - G_8$)

Y - a single-element vector containing the Oil price

Results for the CCA analysis are shown in figure 4.18. It is shown that graphlet G_3 correlates positively with the increase in Oil price, while graphlets {1,2,8} correlate negatively. One property that separates the two ends of the graphlet spectrum is their density. Graphlet G_3 is a sparse graphlet, while graphlets {1,2,8} are dense graphlets having a density of at least 0.66.

Using the results we got earlier from section 4.2.6, we know that sparse graphlets correlate with good economic indicators such as GDP per Capita (RGDPL), while dense graphlets correlate with bad economic indicators such as Balance of Current Account (BCA). Using this observation and the fact that sparse graphlets correlate positively with the oil price and dense graphlets vice versa, we can conclude that for Saudi Arabia the good economic indicators such as GDP per Capita, a result of a healthy economy, must correlate with the Oil price[15]. This is confirmed by the fact that Saudi Arabia is an Oil-exporting economy, and it's GDP per Capita has been shown to strongly correlate with the Oil price [66]. We expect similar behaviour for other oil-exporting economies such as Libya, Venezuela, Qatar or Russia.

Canonical Correlation		0.82353	
p-value		0.00000	
X variate		Y variate	
G3	0.49265	Crude Oil price	0.83032
G6	0.09838		
G4	0.05294		
G5	0.03942		
G7	-0.23884		
G8	-0.46603		
G2	-0.50725		
G1	-0.52241		

Figure 4.18: Canonical Correlation Analysis between the short GCV vector of Saudi Arabia and the price of Crude Oil. Only the short GCV-8 vector has been used because of the lack of samples. The results show that graphlet G_3 is has a strong positive correlation with the price of Crude Oil, while graphlets {1,2,8,7} have a negative correlation. This suggests that when the price of Oil is high, the trading partners of Saudi Arabia tend to form paths of 4 nodes (P_4). On the other hand, when the price of Oil is low, the trading partner network of Saudi Arabia tends to cluster ({1,2,8,7} are dense graphlets with a density of at least 0.66). This might be explained by the fact that when the price of Oil is high, Saudi Arabia starts new trading partnerships with isolated countries that are not part of a clustered network.

run on (period 1980-2010). The reason CCA was run from 1980 is because we did not have data for the economic indicators prior to 1980.

[15] if the correlation of XY is strictly positive and the correlation of YZ is likewise, then the correlation of X and Z is not necessarily strictly positive. This is however the case if the correlations of XY respectively YZ are close to 1 [65].

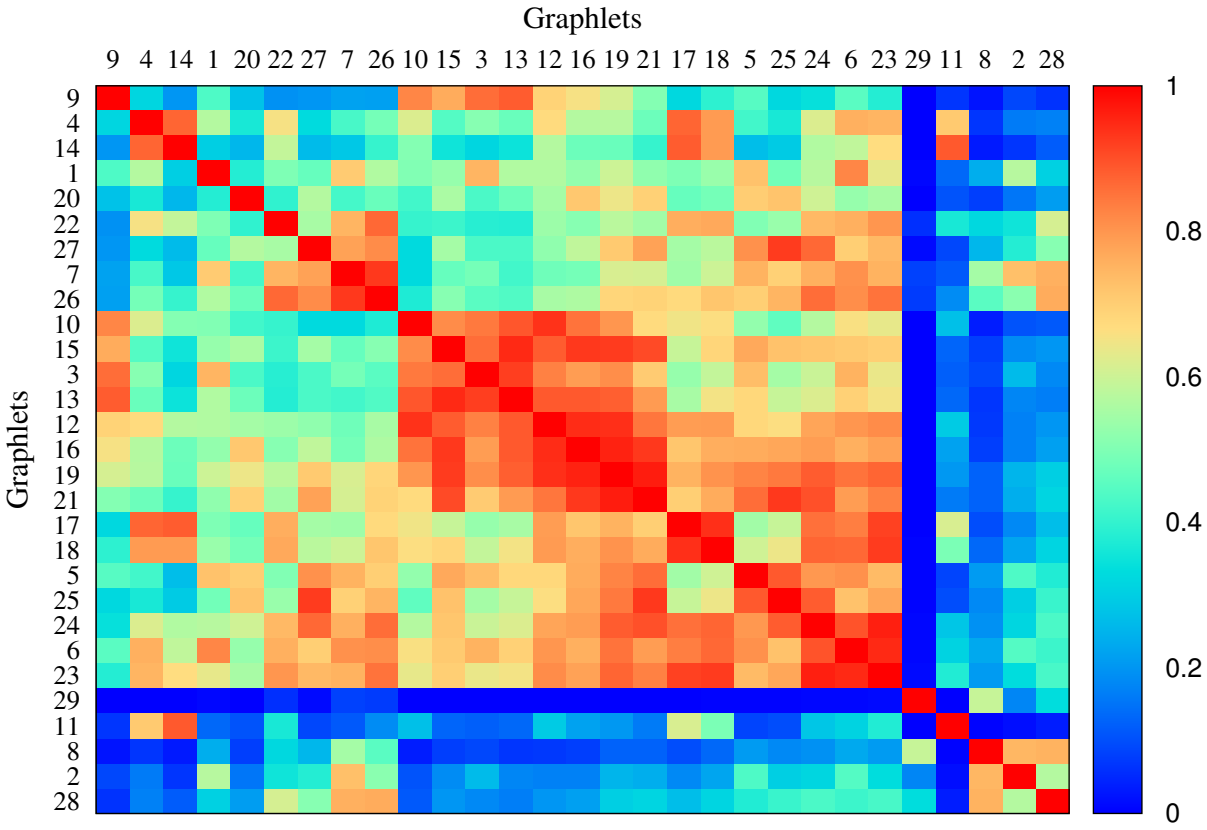


Figure 4.19: Heat map for the Pearson's GCV correlation matrix of the Human PPI network. The heat map has been first normalised with feature scaling and a 4^{th} degree polynomial and then hierarchically clustered.

4.3 Protein-protein Interaction Networks

In this section we apply our methodology for various PPI Networks. For more background information about how these networks are built and their properties, see section 2.7.1. We now present the Pearson's correlation matrix for a Human PPI network and Canonical Correlation Analysis results for six different Human and Yeast PPI networks using two annotation files: Boone's and von Mering's (see annotation descriptions in section 2.7.1). In short, the heat map of the Pearson's GCV correlation matrix did not give us any useful information, since graphlets formed faint clusters. However, the CCA results have helped us get some interesting insights into the interactions of the proteins present in these networks.

4.3.1 Analysis of Pearson's GCV Correlation Matrix

The heat map from figure 4.19 represents the Pearsons's correlation heat map for the Human PPI network. It was first normalised with a simple feature scaling and then with a 4^{th} degree polynomial[16], because the original correlation matrix yielded correlations that were too strong[17]. There are a few faint clusters formed on the diagonal:

[16]Other polynomial functions have also been tested, but the 4^{th} degree polynomial offers the best results.
[17]Having all correlations close to 1 made the identification of clusters impossible

- {10,15,3,13,12,16,19 and 21}. These graphlets all contain a P_4[18].
- {7,26} contain a G_7.
- {4,14} contain a G_4.
- {17,18} contain 2 G_2's (triangles).
- {5,25} contain a G_5.
- {24,6,23} contain a G_6.

The lack of clear graphlet clusters in the Human PPI is something that we cannot explain at the current time. Because of this, it has not been possible for us to get any actual insights from the Human PPI correlation matrix. Other human and yeast PPI networks have yielded similar results. Further research needs to be done into this area in order to explain the lack of graphlet clustering.

4.3.2 Canonical Correlation Analysis

The next step after the Pearson's GCV correlation matrix is to run CCA on the PPI network. We set the X variate to be the GCV and the Y variate to be a vector of values of Boone's annotation. For setting up the Y variate, we label each protein with a vector of binary entries, where the i^{th} entry is as follows:

$$Y_i = \begin{cases} 1, & \text{if the protein is annotated with the } i^{th} \text{ annotation} \\ 0, & \text{otherwise} \end{cases}$$

Since each protein had only one annotation, each sample from Y only contained one non-null entry. The results of the CCA on this network were unfortunately not good, since the correlation is low and the p-value is above 0.05, suggesting that the correlation is not statistically significant. In the next section we will explain the subsequent experiments that have been performed on other PPI networks.

4.3.3 Results for other PPI networks

The 17 experiments

Since the CCA applied to the Human PPI network didn't give us any meaningful information, we thought of exhaustively running it on several types of Human and Yeast PPI networks. We ran the same process on 5 other Human PPI networks with Boone's annotation file and on 6 Yeast networks using the two different annotation files: von Mering's and Boone's (see section 2.7.1). For these experiments we have also used high-confidence networks, which contain only protein interactions that have been confirmed by two independent sources. The networks analysed are as follows:

- 5 Human networks
  - A high-quality Human PPI network determined by Stitch-seq protocol [67], CCA results are not statistically significant.[19]
  - Two networks from I2D, a database of PPI networks maintained by Jurisca lab [68] at Ontario Cancer Institute:

[18] Path on 4 nodes, graphlet G_3

[19] In this subsection by statistically significant we mean that either the p-value was above 0.05 or the total correlation was below 0.2

    - ∗ Full version, CCA results are not statistically significant.
    - ∗ High-confidence version, CCA results are not statistically significant.
  - Two networks from BioGRID:
    - ∗ Full version, CCA results are not statistically significant.
    - ∗ High-confidence version, CCA results are not statistically significant.

- 6 Yeast networks x 2 annotation files
  - A network obtained through *affinity-purification mass spectroscopy* (AP-MS) by Collin's et al [69] - Co-complex membership associations, CCA results in figures B.3, B.7
  - A genetic network from BioGRID, CCA results in figures B.4, B.8
  - Literature-curated PPI network by Reguly et al. [70], CCA results are not statistically significant.
  - Yeast two-hybrid network made from the union of CCSB-YI1, Ito-core and Uetz-screen [71], CCA results are not statistically significant.
  - Two PPI networks from BioGRID:
    - ∗ Full version, CCA results in figures B.5, B.9
    - ∗ High-confidence version, CCA results in figures B.6, B.10

The best results have been obtained for the following Yeast networks, for both von Mering's and Boone's annotation files:

1. Collin's AP-MS network
2. BioGRID Full
3. BioGRID High-confidence.

Detailed interpretations of these results are given in the following section. The overall CCA correlations for these networks have been around 0.45-0.5, all having p-values smaller than 0.05. The other combinations of networks and annotation files have yielded much weaker correlations (only approx 0.2) and high p-values above 0.5. Therefore we could not get any insights from the human PPI networks or the other Yeast networks. One of the reason for this might be the amount of noise present in the PPI data. In the next section we present the key Yeast PPI results and provide biological interpretations for the observed phenomena. The other CCA results for all the 17 experiments are shown in the Appendix section 2.

4.3.4 Summary of the CCA Results from the 17 experiments

Ribosome translation

Figure 4.20 shows the CCA results for Collin's AP-MS[20] PPI network. A full list of all the cross-loadings is given in appendix figure B.3. The results mainly show that Ribosome Translation is correlated with all the graphlets, since their cross-loadings have the same sign. The spectrum of graphlets runs from the most dense graphlets {2,8,29} on top, having the highest cross-loading magnitude of around 1 to the sparser {9,10,13,11,12} graphlets at the bottom, having cross-loading magnitudes of approximately 0.46. The observation we can make is the following: *proteins involved in Ribosome translation generally interact more with clusters of other proteins and less with individual proteins.* This result is also confirmed by the same experiment that was run using Von Mering's annotation, with *Translation* also correlating positively with all

[20] affinity-purification mass spectroscopy

the graphlets (see figure B.7). The explanation for this is that these clusters are found in the *Ribosome complex*, a molecular machine that serves as the site for protein synthesis. It is usually made up of dozens of distinct proteins that interact with each other.

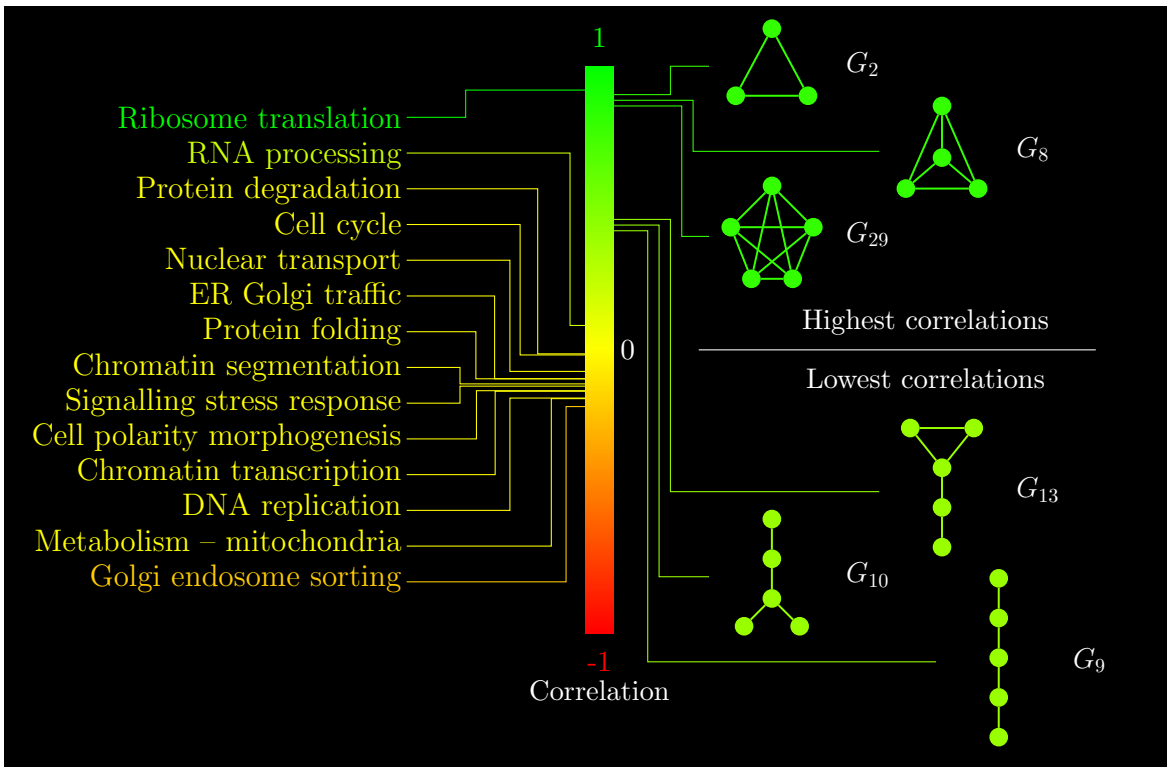


Figure 4.20: CCA Analysis on Collin's AP-MS Yeast PPI network using Boone's protein annotations (see section 2.7.1) and the GCV signature. The correlation value is 0.53 and the p-value is 0. Ribosome translation and RNA processing correlate positively with all the graphlets, while the rest of the protein annotations correlate negatively. On the annotation side, the correlation is dominated by Ribosome translation, which has the largest correlation by far. This suggests that proteins that are involved in Ribosome translation have a neighbourhood full of cliques and other graphlets. The explanation for this is that these clusters are part of the Ribosome complex. Other experiments have also confirmed the correlations of Ribosome translation, RNA processing , Metabolism – mitochondria and Golgi endosome sorting (figures B.5, B.6 and B.7). However, correlations for rest of the annotations were not consistent in results from other experiments, so we conclude that they are not statistically significant.

RNA processing

RNA processing, formally known as *Post-transcriptional modification* is a biological process in which primary transcript RNA is converted into mature RNA. CCA results also show that RNA processing is correlated with dense graphlets such as cliques {2,8,29}. Although the magnitude of the cross-loading for RNA processing is not extremely high (-0.08), other experiments (see figures B.5 and B.6) have actually yielded a higher-magnitude cross-loading of around -0.2, which means that the correlation cannot be attributed to chance or noise. If we try to understand the RNA processing a bit further, we find out that there are three main tasks that occur in the cell nucleus before the RNA is translated [72]:

- 5' capping
- 3' polyadenylation

- RNA splicing

The second step in RNA processing, 3' polyadenylation, is a process in which a segment of the newly made pre-mRNA is first cleaved off by a *set of proteins*. This protein complex then synthesises the poly(A) tail at the RNA's 3' end. We believe that this protein complex might be one of the reasons why cliques correlate highly with proteins involved in the polyadenylation step of RNA processing. The third step of the RNA processing, referred to as RNA splicing, is a process in which regions of the RNA that do not code for protein (i.e. introns) are removed and the remaining nucleotide sequence (i.e. exon) is re-connected to form a single continuous molecule. This splicing reaction is also catalysed by a large protein complex called the *Spliceosome* that is assembled from several smaller protein complexes and small nuclear RNA molecules. The presence of these protein complexes in RNA processing results in proteins interacting with dense clusters of other proteins that are part of these complexes.

Golgi Endosome vacuole sorting

At the other end of the Y variate we have Golgi Endosome vacuole sorting with a weight of -0.2. Golgi endosome vacuole sorting is an environment where material is sorted before it reaches the degradative state. CCA analysis shows that proteins involved in the Golgi endosome have a sparse environment, since all the graphlets correlate negatively with the Golgi endosome index[21]. The explanation for this is that proteins involved in Golgi endosome sorting mainly interact with the proteins that need to be sorted, but these don't interact with each other. This result is also confirmed by similar experiments run on the Yeast Biogrid networks, both full and high-confidence versions (see figures B.5 and B.6 in the appendix).

Metabolism - mitochondria

Figure B.3 shows that the Metabolism/mitochondria index is negatively correlated with all the graphlets. This suggests that the proteins present in mitochondria interact with other proteins which in turn don't interact much with each other. This could be explained by the fact that the proteins present in mitochondria each have a variety of different functions and therefore their partner proteins are unlikely to interact because they have different functions. The main functions of the proteins found in mitochondria are related to:

- Energy production and cellular metabolism - the main function of a large number of mitochondria proteins is the production of Adenosine triphosphate (ATP), commonly referred to as the energy currency of the cell. [73]
- Pyruvate and the citric acid cycle [73]
- Electron transport chain [73]
- Heat production [73]
- Storage of calcium ions [74]
- Signalling through mitochondrial reactive oxygen species [75]
- Regulation of the membrane potential [73]
- Apoptosis (programmed cell death) [76]
- Calcium signalling (including calcium-evoked apoptosis) [77]

[21] they correlate negatively since their weights have different signs: Golgi endosome has a weight of 0.2, while all the graphlets have negative weights

- Regulation of cellular metabolism [78]
- Certain heme synthesis reactions [79]
- Steroid synthesis [80]

We can illustrate our last argument using a small, simple example. *Cytochrome c* is a small protein found in the inner membrane of the mitochondrion. It is an essential protein in the Electron transport chain, where it carries one electron. Apart from electron transportation, it is also involved in the initiation of apoptosis, that is the programmed cell death. However, the interacting partners of *Cytochrome c* are less likely to interact with each other, since they are split in two different functional groups: electron transportation and apoptosis. Now, from a topological point of view, that is why the network of partners of *Cytochrome c* is more likely to form sparser graphlets such as {9,10,13,11,12} as opposed to dense graphlets such as {29,28}.

Ribosome translation, RNA processing, Golgi endoscope sorting and Metabolism/mitochondria are the annotations that have consistently shown up with strong correlations in all our relevant[22] experiments. The other annotations varied in their correlation, so their weights are not reliable. We conclude that our GCV signature coupled with Canonical Correlation Analysis cannot capture any patterns in proteins that are part of those processes.

4.4 Metabolic networks

We computed the Pearson's correlation matrix and CCA for metabolic networks belonging to several different organisms: Homo sapiens (human), C. elegans (worm), D. melanogaster (fruit fly), E. coli (bacteria), M. musculus (mouse) and S. cerevisiae (baker's yeast). Most of the experiments showed consistent results consistent across the spectrum of organisms, so only the heat maps and CCA figures/tables for the human metabolic network are presented. For background information on metabolic networks see section 2.7.2.

4.4.1 Analysis of Pearson's Correlation Matrix

Figure 4.21 illustrates the Pearson's GCV correlation matrix for the compound-based Human metabolic network, normalised with feature scaling and a 3^{rd} degree polynomial. We clearly distinguish several clusters of graphlets that formed along the main diagonal. Section 2.1.1 describes the graphlet terminology in detail. The main clusters are as follows:

A **Claw** cluster made of graphlets {4,16,5,25,1,17,14,22}. These graphlets all have a C_4 (claw of 4 nodes) as a subgraph.

B **Paths** cluster made of graphlets {9,13,21,10,15,12,3,19}. These graphlets all have a P_4 (path of 4 nodes) as a subgraph.

C **Triangles** cluster made of graphlets {2,26,24,18,23,27,6,7}. These graphlets all have triangles (graphlet G_2) as subgraphs

D **Dense graphlets** cluster made of graphlets {29,8,28}. Graphlets {8,29} are cliques, while G_{28} is almost a clique because it has one missing edge. Note that the 3-node clique (graphlet G_2) is missing, because it is more correlated with the triangle group above.

[22] i.e. experiments with the Biogrid and Collin's yeast networks, since they have a p-value below 0.05 and relatively high canonical correlations

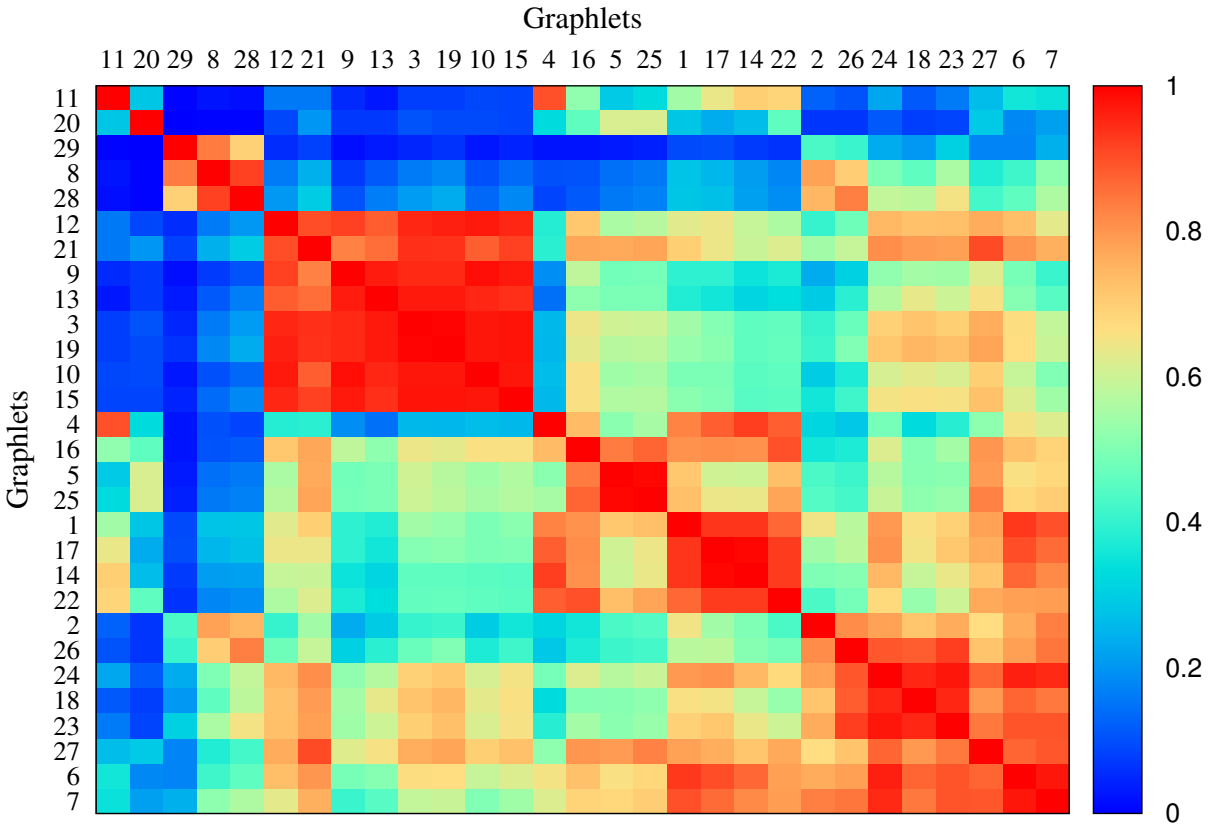


Figure 4.21: Pearson's GCV correlation matrix heat map for the compound-based Human Metabolic network. The heat map has been normalised with feature scaling and a 3^{rd} degree polynomial and hierarchically clustered.

Furthermore, we notice that graphlets from clusters A,B and C also have a certain amount of inter-correlation between them, with inter-correlation values of at least 0.5. However, this is not the case for cluster D, which is made of cliques. The cliques only bear some correlation with cluster C made of triangle-like graphlets, which is not surprising for the following reasons:

- Cliques contain a lot of triangles
- Cliques do not contain claws C_n or paths P_n, which miss several edges.

It should also be noted that graphlets G_{11} and G_{20} have been left outside, as they don't strongly correlate with any of the other groups. The cluster closest to these 2 graphlets is the claw cluster. To sum up, we conclude that graphlets cluster together according to what basic shapes they contain.

4.4.2 Canonical Correlation Analysis

In order to run Canonical Correlation Analysis on the metabolic networks we used *Enzyme Commission* (EC) numbers as annotations for the network nodes. More information about EC numbers can be found in background section 2.7.2. Basically, EC numbers are used to annotate each enzyme in the metabolic network according to the type of reaction it catalyses. The results of the CCA analysis using EC numbers is presented in figure 4.22.

There is some degree of correlation between the Graphlets and the EC numbers ($\rho = 0.517$), with a p-value smaller than 0.05. All the cross-loadings from both the graphlets and the EC

Canonical Correlation		0.51769	
p-value		0.00000	
X variate		Y variate	
EC5	-0.11422	G20	-0.13839
EC2	-0.12601	G11	-0.17646
EC4	-0.16144	G9	-0.17733
EC1	-0.16156	G10	-0.18420
EC3	-0.21057	G13	-0.18757
EC6	-0.40615	G16	-0.20243
		G12	-0.20399
		G4	-0.20521
		G15	-0.20553
		G3	-0.20671
		G19	-0.20956
		G22	-0.21329
		G5	-0.21897
		G25	-0.22033
		G14	-0.22256
		G17	-0.23125
		G21	-0.23144
		G18	-0.24860
		G1	-0.25408
		G27	-0.25641
		G6	-0.26110
		G24	-0.26365
		G23	-0.26813
		G7	-0.28239
		G26	-0.28362
		G28	-0.32754
		G29	-0.34583
		G2	-0.36785
		G8	-0.37442

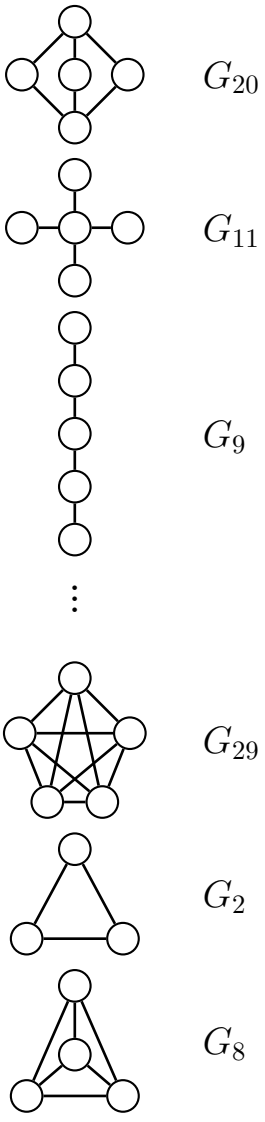


Figure 4.22: CCA analysis on the compound-based Human Metabolic network. The CCA is 0.51, with a p-value smaller than 0.0001. We notice that all EC numbers correlate positively with all the graphlets because their cross-loadings have the same sign. In the X variate EC6 shows the highest correlation while in the Y variate cliques {8,2,29} show the highest correlation. Note that the p-value is not exactly zero, but it was truncated to zero because of floating point approximations.

numbers have the same sign, which suggests that they are positively correlated. Cliques {8,2,29} have the highest magnitude in their weights, while EC6 (ligands) have the highest magnitude in the EC vector.

EC6 refers to Ligases, which are enzymes that can catalyse the joining of two large molecules by forming a new chemical bond. The reason why the magnitude of EC6 is quite high (0.4) compared to the other indicators is because the two large molecules catalysed by EC6 enzymes are represented in the metabolic network by cliques or dense clusters which have a lot of interactions and feedback loops between them. This is why cliques {8,2,29} or dense graphlets such as G_{28} have the cross-loadings with the highest magnitude. However, this doesn't exclude other sparser protein groups to be part of the two molecules catalysed by the Ligase, since graphlets such as {9,10,11 or 12} also correlate positively with EC6. Regarding the other functional groups, we cannot say much about them because the magnitude of their cross-loading is relatively smaller compared to the cross-loading of EC6.

4.4.3 CCA Results for other model organisms

We have analysed other compound-based metabolic networks that belong to the following organisms: C. elegans, D.melanogaster, E.coli, M.musculus, S.cerevisiae. These experiments confirm the results obtained for the human metabolic network. Average CCA correlation is around 0.5, EC6 has the highest magnitude at around 0.4 and cliques {2,8,29} are the graphlets most correlated with EC6 ($\rho = 0.35$).

The same methodology has also been applied to enzyme-based metabolic networks for all the 6 different organisms. However, these display a much lower CCA correlation (around 0.25), having p-values that are above 0.05, suggesting that the results are not statistically significant. This is the case for all the organisms, including humans. The graphlet signatures have very low signatures, while EC numbers don't have magnitudes above 0.22. These results have not been included in the report, but are available in the source code, under the `code/final_results/` folder[23].

4.4.4 CCA on the KEGG categories

We have also tried to use the KEGG categories as annotations for the enzymes in the metabolic network. We have initially annotated the enzymes with the following high-level categories:

- Metabolism
- Genetic Information Processing
- Environmental Information Processing
- Cellular Processes
- Organismal Systems
- Human Diseases

The CCA correlation obtained was only around 0.6, so we tried running CCA on the lower-level categories. That is, for each of those 6 high-level categories, we ran CCA on its subcategories. The best results were obtained for Human Diseases, Cellular Processes and Organismal Systems and are presented in the following subsections.

[23]The relevant folders are: `hsa_meta_ec_omer`, `cel_metabolic`, `dme_metabolic`, `eco_metabolic`, `mmu_metabolic`, `sce_metabolic`,

4.4.5 Cellular Processes

The overall CCA correlation for Cellular Processes is 0.98, which is quite high compared to previous CCAs, and the p-value is smaller than 0.00001. Figure 4.23 shows the CCA for Cellular Processes. We observe that graphlet G_9 correlates positively with *Transport and Catabolism*. The reason for this is because in Catabolism, large molecules such as polysaccharides, lipids and nucleic acids are broken down into smaller units such as monosaccharides, fatty acids or nucleotides. Since molecules such as polysaccharides are made up of long chains of small monomer units, graphlets that are made of long paths such as G_9 will be overly represented in these processes. Similarly, enzymes involved in transport are transporting nutrients from one chemical to another, so their interactions will be characterised by long "transportation" paths that are best represented by graphlet G_9. At the other end of the spectrum, *Cell growth and death* and *Cell communication* are correlated with graphlets {1,2,7,8}. The reason for this is because in Cell Communication, if a cell is stimulated, it's needs to send signals to its neighbours through the use of molecules. First of all, in order to ensure that a signal is successfully transmitted, several molecules carrying the same message could be transmitted and there must be different possible paths to reach the destination. If this is not the case, then the message will get lost when the path is disrupted or the messenger molecule stops functioning. This is why a graphlet like G_9 correlates negatively with these, because G_9 is made of a long path of 5 nodes and if one of the nodes fails, then the whole signal is lost. Graphlets {2,7,8} correlate positively because these are highly connected ({2,8} are cliques) or because they contain several alternative paths for message transmission (G_7). However, the reason why graphlet G_1 correlates with Cell Communication is still a matter or research.

4.4.6 Organismal Systems

Figure 4.24 shows the CCA for *Organismal Systems*. The CCA correlation is also very high (0.96) and the p-value smaller than 0.0001. These cross-loadings suggest that enzymes involved in Environmental Adaptation and Excretory systems are usually rich in interactions and their neighbours are also highly clustered, since all the graphlets correlate positively with these systems. On the other hand, enzymes involved in Circulatory and Digestive metabolic pathways have sparse neighbourhoods that would ideally contain few graphlets. One explanation for this is because in these systems enzymes, proteins and metabolites have to circulate throughout the whole body and interact with distant enzymes, which don't cluster together. Enzymes at the other end of the spectrum (Environmental Adaptation and Excretory system) are much more localised in the body. For instance, excretory system enzymes are mainly active in the kidney or liver. Moreover, the enzymes involved in the Circulatory and Digestive systems will probably have less interactions compared to their counterparts in the Environmental Adaptation and Excretory systems, because a neighbourhood sparse in graphlets is usually an indication of it being small.

4.4.7 Human Diseases

Figure 4.25 shows the CCA for various Human Diseases such as Cancers, Immune diseases, Neurodegenerative diseases or Cardiovascular diseases. The result that is most striking here is that Cardiovascular diseases and Substance dependence correlate negatively with almost all the graphlets (apart from {2,8}). This implies that the enzymes and proteins involved in these Human Diseases have a low number of interactions and when they do have interactions, their neighbourhood only contains small clusters of 3–4 nodes maximum. The explanation for this might be the same as for the Organismal Systems: the enzymes involved in Cardiovascular diseases and Substance dependence travel across long pathways throughout the body and end up interacting with distant chemicals that do not interact with each other because of their

Cellular Processes CCA

Canonical Correlation		0.98633	
p-value		0.00000	
X variate		Y variate	
Transport and catabolism	0.52121	G9	0.04828
Cell motility	0.20502	G21	0.01960
Cell communication	-0.40751	G25	0.01441
Cell growth and death	-0.69712	G5	0.01434
		G16	0.00969
		G13	0.00199
		G12	-0.00048
		G27	-0.00134
		G20	-0.00256
		G3	-0.00412
		G24	-0.01287
		G19	-0.01528
		G10	-0.01623
		G18	-0.01681
		G14	-0.02579
		G11	-0.02667
		G23	-0.02851
		G15	-0.03092
		G17	-0.03201
		G29	-0.04271
		G6	-0.04386
		G28	-0.04750
		G4	-0.05059
		G26	-0.05235
		G22	-0.05877
		G8	-0.05881
		G7	-0.07069
		G2	-0.07388
		G1	-0.07463

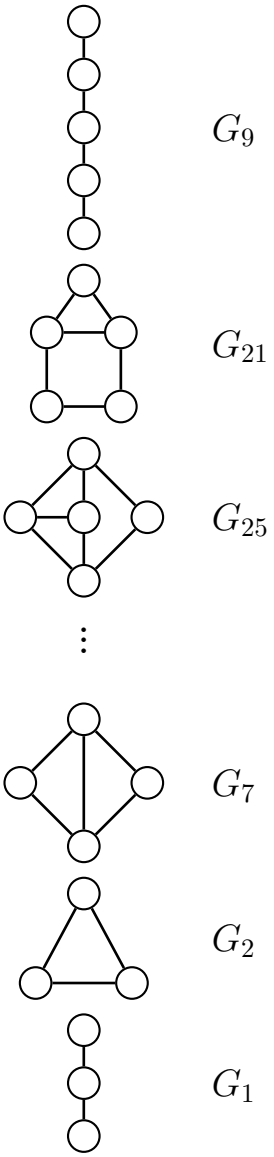


Figure 4.23: CCA on the Human Metabolic network using Cellular Processes from KEGG. The correlation value is high ($\rho = 0.98$) and the p-value is smaller than 0.00001, suggesting a very strong correlation. Transport and catabolism and cell motility correlate with the upper part of the graphlet spectrum: {9,21,25,5, ...} because their cross-loadings have the same sign. Similarly, Cell Communication and Cell growth and death correlate with the lower end of the graphlet spectrum: {1,2,7,8, ...}.

Organismal Systems CCA

Canonical Correlation		0.96925	
p-value		0.00000	
X variate		Y variate	
Environmental adaptation	0.20426	G26	0.30978
Excretory system	0.19729	G24	0.29818
Development	0.07461	G23	0.29308
Endocrine system	0.04192	G18	0.28901
Nervous system	-0.01315	G6	0.27857
Sensory system	-0.06276	G12	0.26520
Immune system	-0.15192	G19	0.25419
Digestive system	-0.23211	G3	0.24988
Circulatory system	-0.37659	G14	0.23274
		G13	0.23155
		G1	0.22611
		G17	0.22546
		G7	0.21225
		G27	0.20440
		G10	0.19976
		G9	0.19547
		G25	0.19422
		G16	0.19135
		G28	0.18874
		G4	0.18421
		G5	0.18381
		G20	0.17359
		G11	0.15537
		G21	0.14453
		G29	0.11208
		G8	0.10681
		G2	0.10369
		G22	0.08192
		G15	0.01276

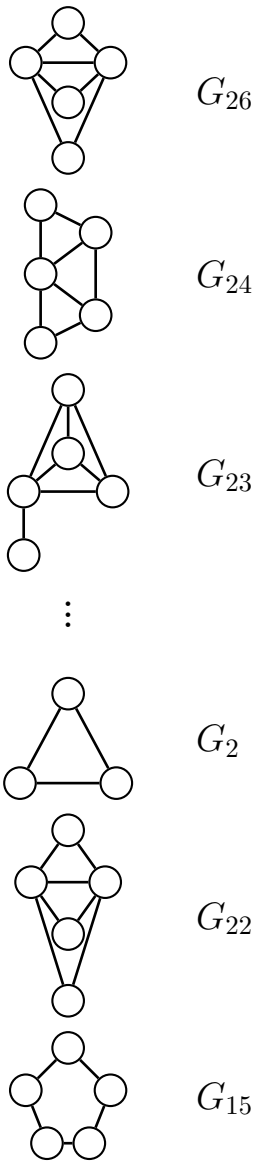


Figure 4.24: CCA on the Human Metabolic network between different Organismal Systems and the GCV signature of enzymes. The correlation value is again high ($\rho = 0.96$) and the p-value is small ($p < 0.00001$). Environmental adaptation, Excretory system, Development and the Endocrine system correlate positively with all the graphlets, while the Circulatory, Digestive, Immune, Sensory and Nervous systems correlate negatively. The actual p-value is not exactly zero but it is shown as zero because of floating point approximations.

distant location in the body. However, the small clusters of interactions might occur because of locality, that is all chemicals involved will be in the same area and therefore inevitably interact with each other. In the case of the Cardiovascular diseases, the enzymes travel long distances because they are transported through the blood vessels, while the ones involved in Substance dependence are again transported through the blood vessels or other channels. We cannot say anything about the rest of the diseases (Cancers, Immune Diseases, etc ...) because their relative cross-loadings have a very low magnitude relative to the others.

Human Diseases CCA

Canonical Correlation			0.99479
p-value			0.00000
X variate		Y variate	
Cardiovascular diseases	0.99171	G2	0.01681
Substance dependence	0.57989	G8	0.00462
Infectious diseases	0.21844	G29	-0.00737
Neurodegenerative diseases	0.00366	G7	-0.00812
Endocrine and metabolic diseases	-0.02545	G1	-0.00989
Immune diseases	-0.03029	G26	-0.01077
Cancers	-0.08682	G24	-0.01274
		G6	-0.01321
		G28	-0.01321
		G15	-0.01342
		G23	-0.01369
		G22	-0.01420
		G21	-0.01438
		G14	-0.01449
		G12	-0.01465
		G17	-0.01516
		G16	-0.01521
		G18	-0.01526
		G13	-0.01528
		G19	-0.01562
		G9	-0.01565
		G10	-0.01641
		G4	-0.01727
		G3	-0.01742
		G11	-0.01781
		G20	-0.01936
		G25	-0.02017
		G5	-0.02438
		G27	-0.02731

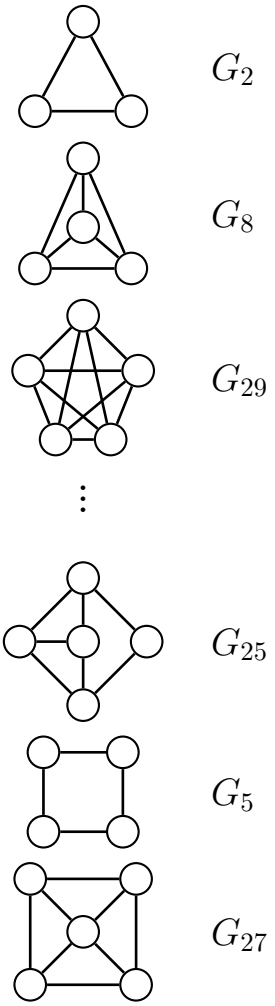


Figure 4.25: CCA Analysis on the Human Metabolic network between Human Diseases and the GCV signature. The correlation value is high ($\rho = 0.96$) while the p-value is very small ($p < 0.00001$). The actual p-value outputted by the program is 0.0 due to floating-point approximations. Cardiovascular diseases, Substance dependence, Infectious and Neurodegenerative diseases correlate positively with all the graphlets, while the Endocrine and Metabolic diseases, Immune diseases and Cancers correlate negatively.

Chapter 5

Evaluation

5.1 Strengths and weaknesses

The GCV signature has been successfully used on some of the networks and it helped us get valuable insights. The best results have been obtained on the WTN networks, followed by the PPI and metabolic networks. Overall, the main achievements of this project are as follows:

- The development of the mathematical model of the GCV signature followed by the implementation and parallelisation of the algorithm that computes it.
- The use of a rigorous methodology for the analysis of GCV correlations that helped us uncover insights from the network data.
- The results and interpretations presented for the economic, protein interaction and metabolic networks.
- The quantitative evaluation of the GCV signature (sections 5.2 and 5.6).

However, the GCV signature has inherent limitations and weaknesses. The main deficiencies with our methodology are:

- A more effective normalisation method of the GCV can be designed. Such a normalisation method can take into account the size of the neighbourhood subgraph.
- A redundancy analysis of each GCV frequency has not been made. This could tell us whether elements in the GCV vector are redundant and eliminating these will improve the GCV performance and remove noise.
- The GCV signature is only able to quantify the structure between the immediate neighbours of a node. It cannot capture the structure between nodes that are further away from the source node, at distances of 2 or more.
- The implementation of the GCV computation is not parallelisable on a cluster of computers. The program is only able to spawn processes that run on multiple cores. Moreover, when using multiple processes for parallel GCV computation, most of the processes finish their share early while a few processes get stuck with computing GCVs for hub nodes. This problem could be overcome by redistributing the workload to the processes that finish early.
- The GCV signature is not able to capture any information in some of the networks, such as the enzyme-based metabolic network. Further research needs to be done in order to understand why that is the case.
- The results we got for the WTN, PPI and Metabolic networks need more supporting experiments in order to validate the interpretations.

5.2 Evaluation of network clustering

Although the focus of this project is on using the GCV signature for uncovering hidden structures in the data analysed, we are also interested to find out whether the GCV can be used for clustering networks of different types. In this section, we evaluate the performance of the GCV signature on clustering the following types of random graphs:

- Erdős-Rényi graphs (ER)
- Erdős-Rényi graphs with preserved degree distribution (ER-DD)
- Geometric graphs (GEO)
- Scale-free Barabási-Albert graphs (preferential attachment) (SF-BA)
- Stickiness index-based graphs (STICKY)

For each model, we generate 30 different networks that are modelled from the 2010 World Trade network. These random networks have also been used in section 4.1.2 for computing the average network GCVs. For each one of the 150 generated networks, we compute 6 different signatures:

1. Graphlet Cluster Vector (GCV)
2. Degree Distribution
3. Average clustering coefficient
4. Spectral distribution
5. Graphlet Frequency Vector (GFV).
6. Graphlet Distribution Vector (GDV).

Calculating each of the 6 signatures requires a considerable amount of computation. This is the reason why we have chosen to generate the random networks from the WTNs, because these networks are small in comparison to the metabolic or PPI networks. Other networks such as the PPI networks are much larger and computation of all the signatures on 150 of these networks is very intense.

After all signatures for the 150 networks have been calculated, the distance between each pair of networks is computed. All these entries are placed in a 150x150 distance matrix and 6 distance matrices are finally obtained, one for each signature. The distance matrices can be used for visualising the distances using *Multi-dimensional scaling* or for performing Precision-Recall curve analysis or Receiver-Operating Characteristic (ROC) curve analysis. These results are presented in the next sections.

The Relative Graphlet Frequency distance (RGFD) defined in section 2.3.3 has been used as the distance metric between two Graphlet Frequency Vectors. For the distance metric between two Graphlet Distribution Vectors, we have used the Graphlet Correlation Distance defined in section 2.5.3. This will be denoted as GCD73, because it uses information from all 73 orbits and in order to distinguish it from a similar metric called GCD-11 that has been developed by Yaveroğlu et al [37]. For the degree and spectral distributions, we have used as the Euclidean distance between the first 60 elements.[1]

[1]These distributions are in theory infinite so we decided to cap them at 60, since very little information is retained after this threshold.

5.3 Multi-dimensional scaling results

Multi-dimensional scaling (MDS) refers to a series of visualisation techniques which attempts to represent n-dimensional data points into a 2D or 3D space such that the distances between them are preserved as much as possible. We computed 3D MDS plots for each of the 6 signatures using the `Python Scikit` library which provides the function `sklearn.manifold.MDS` that can perform the MDS transformation.

Figures 5.1 and 5.2 provide the 3D MDS plots for the GCV and the Clustering coefficient. For the GCV MDS plot, the ER networks are more spread compared to the other random graphs and clearly distanced from them. On the other hand, the distances between the ER-DD, STICKY and SF-BA graphs are really small, suggesting that the GCV signature cannot easily distinguish between these random networks. We also notice that the SF-BA random graphs have formed two different clusters. This phenomena might be explained by the fact that the SF-BA random graphs are very sensitive to the initial starting graph. We therefore conclude that the GCV signature can only distinguish the ER networks from the rest.

For the Clustering coefficient MDS, the data points are positioned in a nearly collinear fashion, because the clustering coefficient is a 1-dimensional signature. The ER-DD and Stickiness graphs show some degree of overlap[2], while the rest of the random graphs are clearly separated from each other. This means that the clustering coefficient is able to distinguish any two pairs of random graphs apart from a STICKY, ER-DD network pair.

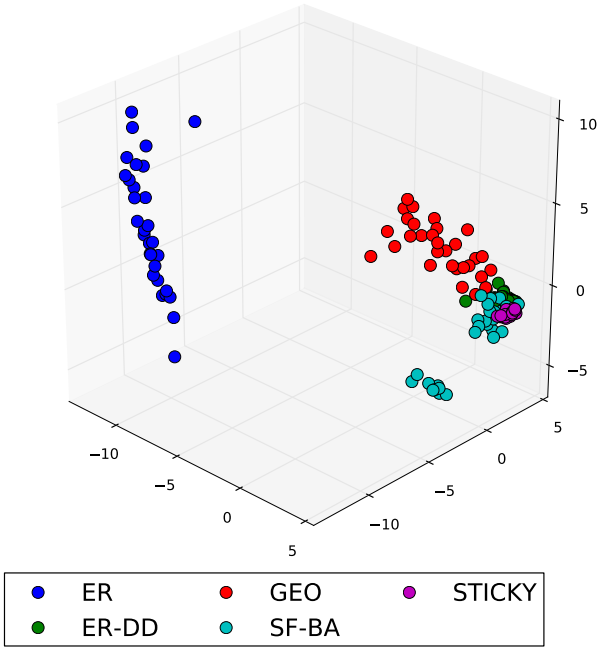


Figure 5.1: GCV MDS: The GCV signature cannot distinguish between ER-DD, GEO, SF and STICKY random graphs. The intra-cluster variance for ER networks is high.

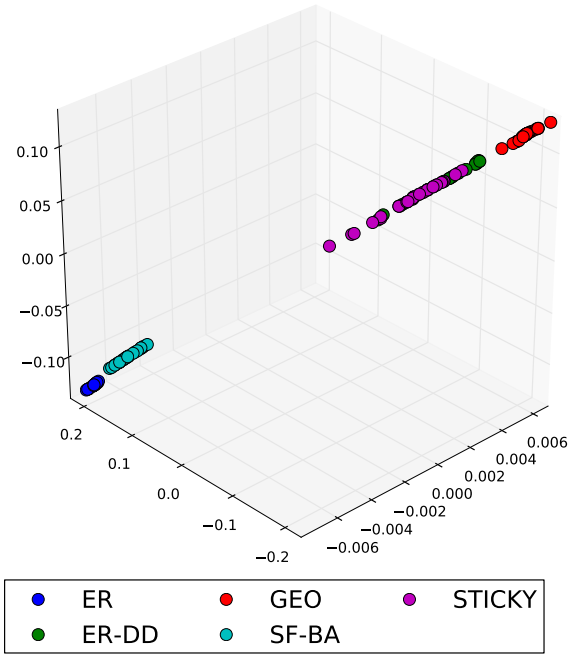


Figure 5.2: Clustering Coefficient MDS: The clustering coefficient cannot distinguish between ER-DD and STICKY random graphs (ER-DD points are hidden behind the STICKY points).

The RGFD and GCD73 MDS plots are shown in figures 5.3 and 5.4 respectively. The RGFD MDS shows that each of the clusters is clearly separated from the other, suggesting that RGFD

[2]this fact is not completely obvious from the graph because the STICKY data points are covering the corresponding ER-DD data points.

is highly suitable for separating these types of networks. The GCD-73 metric is also suitable for clustering random networks, but the clusters display a higher variance.

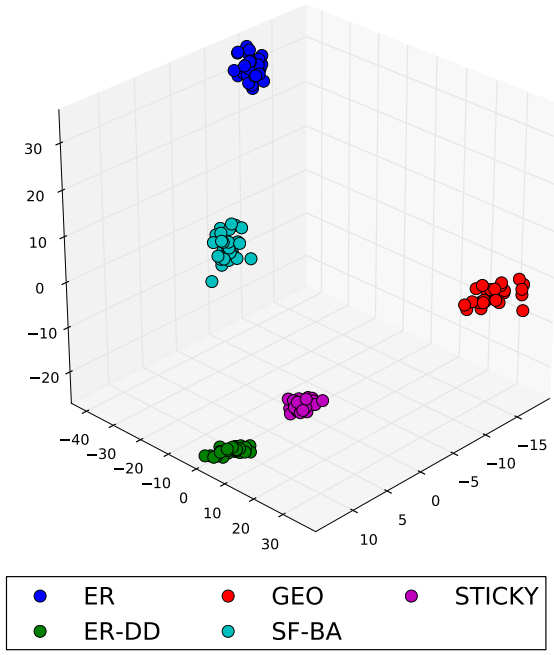


Figure 5.3: RGFD MDS: The RGFD is clearly able to separate all the random network models. The intra-cluster variance is low.

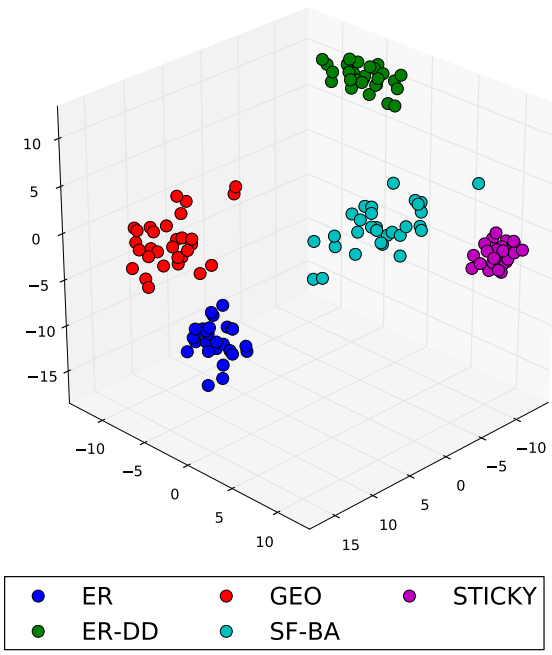


Figure 5.4: GCD73 MDS: The GCD73 metric is also efficient at clustering random networks although the clusters are more spread around.

Figures 5.5 and 5.6 provide the 3D MDS plots for the Degree distribution and the Spectral Distribution signatures. The GEO and ER clusters in the Degree distribution MDS show a certain degree of overlap, although in reality there is much less overlap because the viewing angle is unsuitable[3]. In the Spectral Distribution MDS, we notice that the ER and GEO clusters are very close to each other, suggesting that the Spectral Distribution cannot easily distinguish between these two types of random networks. However, the other clusters are clearly separated from each other.

[3]We tried to capture the image from other angles but that resulted in other clusters colliding.

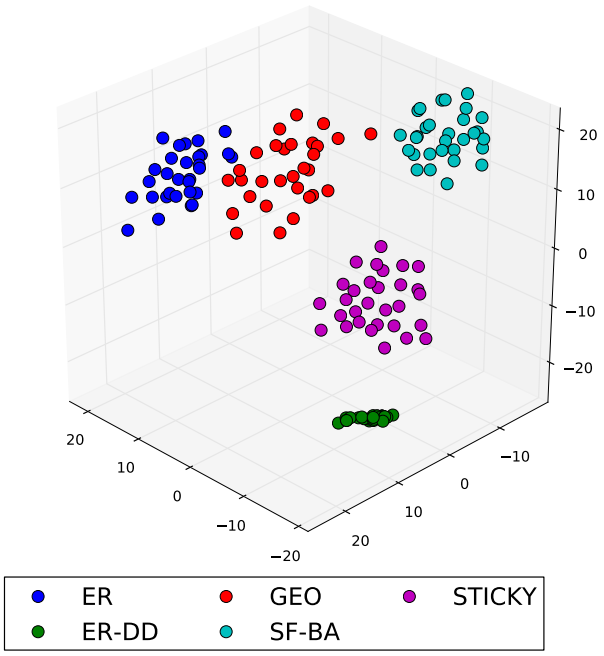


Figure 5.5: Degree Distribution MDS: Most of the clusters are clearly separated although the intra-cluster variance for ER, GEO, SF-BA and STICKY is high.

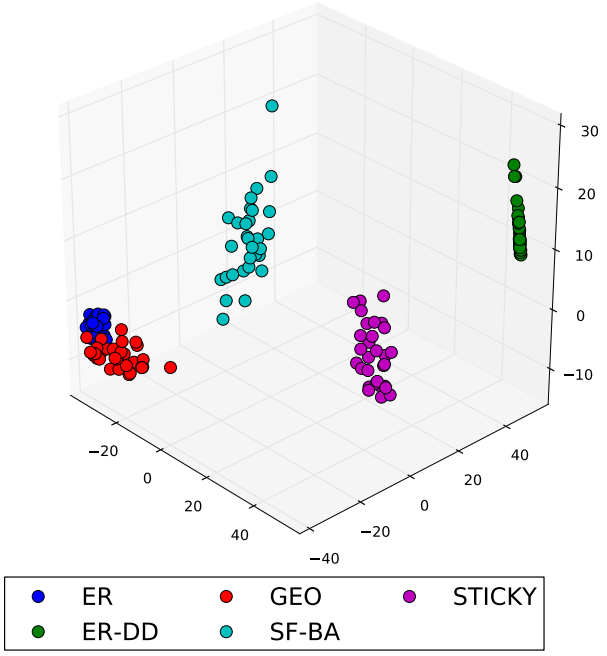


Figure 5.6: Spectral Distribution MDS: The spectral distribution cannot distinguish between ER and GEO random graphs. The other pairs of clusters are clearly separated.

5.4 Precision-Recall curve

MDS plots are only useful for visualising the distance matrices. However, one can test how well a distance measure groups networks of the same type by using the Precision-Recall curve. Starting from the 150x150 distance matrix, a Precision-Recall curve analysis can be performed in the following manner:

1. one searches for the minimum and maximum distance in the distance matrix.
2. for small increments of parameter ϵ such that $min \leq \epsilon \leq max$, if the distance between two networks is smaller than ϵ then the pair of networks is retrieved
   (a) Precision is calculated as the fraction of the correctly retrieved pairs (i.e. grouping networks from the same model)
   (b) Recall is calculated as the fraction of the correctly retrieved pairs over all the correct ones.
3. The Precision-Recall curve is plotted using the values calculated so far.
4. The Area under Precision-Recall (AUPR) can be calculated using the following formula:

$$AUPR = AUPR + 0.5 * (REC[k] - REC[k-1]) * (PREC[k] + PREC[k-1])$$

We chose to perform a Precision-Recall curve analysis because it is known to be more robust to large numbers of negatives than Receiver Operator Characteristic (ROC) curve analysis [81]. In our case negatives are pairs of networks that are grouped together although they belong to different random models.

Figure 5.7 shows the precision-recall curve for the six signatures calculated from their distance matrices. Our novel GCV signature has a generally low precision compared to the other signatures, as the precision decreases a lot in the recall range [0.2 – 0.5]. This result was expected from our signature, since the MDS plots showed that it cannot easily distinguish between ER-DD, STICKY, SF-BA and GEO random graphs (figure 5.1). The best-performing signature is actually the RGFD, which has a precision of 1 for any recall value in range [0 – 1]. Note that this is only faintly seen on the plot, because the GCD73 line overwrites it.

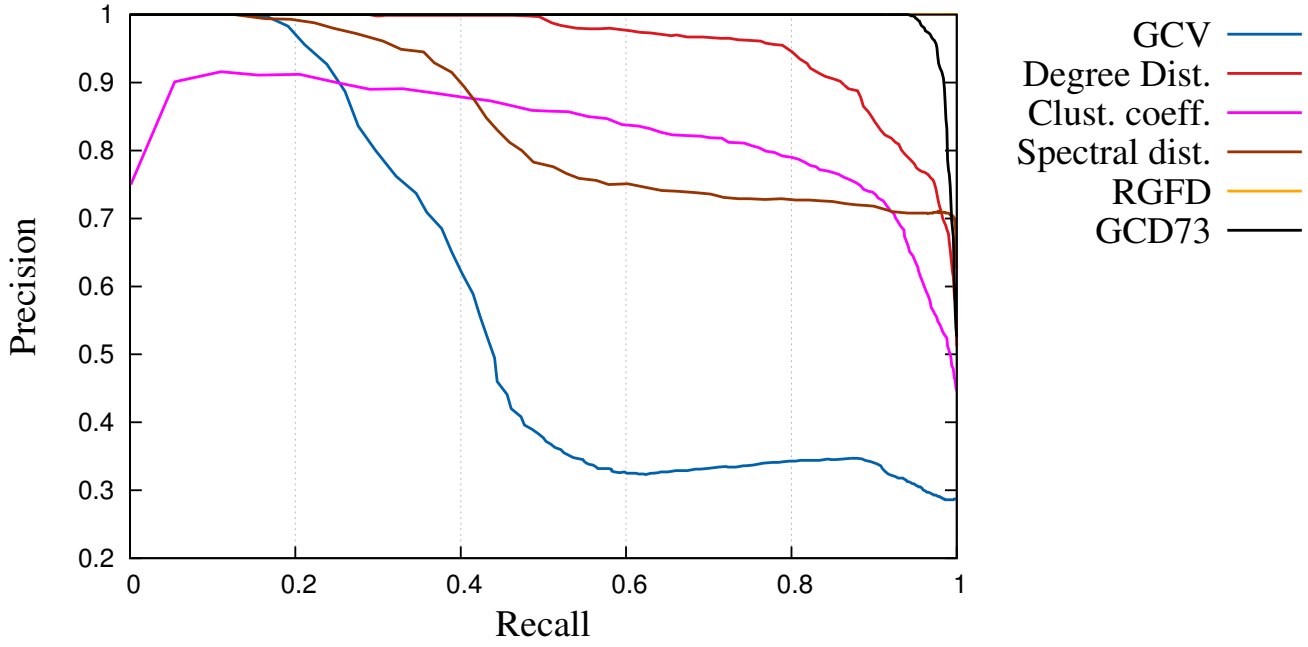


Figure 5.7: Precision-Recall curves for 6 different signatures: Graphlet Cluster Vector (GCV), Degree Distribution, Clustering Coefficient, Spectral Distribution, the Relative Graphlet Frequency distance (RGFD) and the Graphlet Correlation distance which uses the 73 automorphism orbits (GCD73). The best performing signature is the RGFD which has a value of 1 for any recall value. However, this is not clearly seen in the plot because the GCD73 line overwrites it. The GCV signature has the worst performance particularly in the range [0.25–1].

Table 5.1 shows the table of AUPR values for each of the signatures. The higher the AUPR, the better the signature is at distinguishing between different clusters. The best-performing distance measure is the RGFD that uses the *Graphlet Frequency Vector* of the random network. It has a perfect AUPR of 1.0, which is expected because the RGFD MDS plot showed it can clearly distinguish all the random graphs generated. On the opposite end, our GCV signature has the worst AUPR of only 0.575. This suggests that the GCV signature is not suitable for clustering random networks generated from the WTN.

GCV	0.575
Degree Distribution	0.949
Clustering Coefficient	0.829
Spectral Distribution	0.840
RGFD	**1.0**
GCD73	0.994

Table 5.1: AUPR table for the GCV and other signatures. The best AUPR has been obtained using the GCD73 signature, which has an AUPR of 0.994. On the other hand, our GCV signature performed worst with an AUPR of 0.575.

5.5 Robustness testing

This section evaluates the robustness of the six signatures. The same Precision-Recall curve analysis is performed, this time with data that is noisy, incomplete or when the signatures are approximated. However, because of the sheer number of experiments performed, only the final AUPR values are plotted. The methodology is similar to that performed by Yaveroğlu et al. on the short GCD-11 signature [37].

5.5.1 Network Rewiring

In most real-life scenarios the data we have to work with is noisy. In order to evaluate the GCV robustness to noise, we take each of our initial 150 generated random networks and rewire the edges with a probability p, for different values of p between 0 and 1. When rewiring an edge (i, j), we find a target node k such that there is no edge between nodes i and k. For each rewiring probability p we get 150 different networks that have been rewired. Afterwards, we calculate the AUPR for this set of networks. Figure 5.8 shows the AUPR for each signature as p increases from 0% to 90%. All signatures apart from GCV and clustering coefficient show a general downward trend. The GCV reaches a low point in AUPR for a rewiring rate of 50%, but it increases again shortly afterwards. On the other hand, the clustering coefficient reaches a maximal AUPR when $p = 0.3$, followed by a sharp drop afterwards. When the networks are almost random ($p = 0.9$), the values of the AUPR converge to the range [0.5,0.7] for all signatures.

We therefore conclude that the GCV signature is not robust to noisy data either, with other signatures such as RGFD always having an AUPR that is higher, for all rewiring rates. The best performing signature is again the RGFD which always has an AUPR above 0.6.

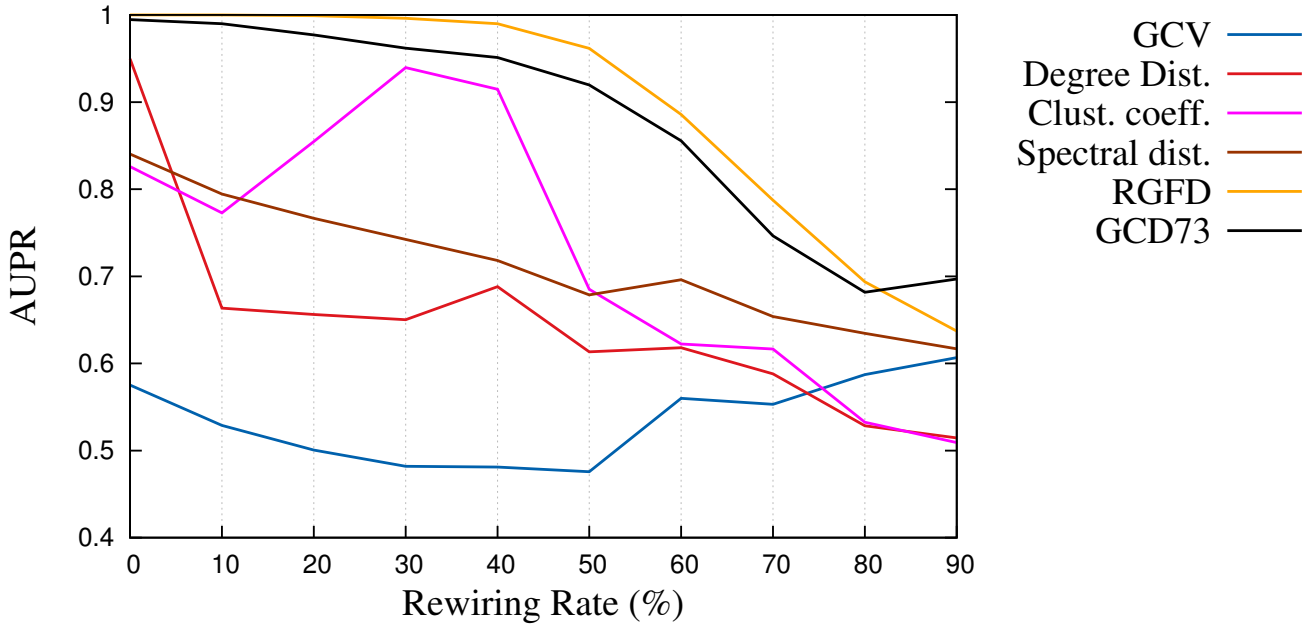


Figure 5.8: The AUPR for different percentages of noise in the model networks. The rewiring probability increases from 0 to 90%. The GCV signature has a poor performance when dealing with noisy data, having the lowest AUPR when the rewiring parameter is in the [0–70] range.

5.5.2 Edge completeness

In real-life situations, one also has to deal with incomplete data. In order to simulate incomplete data in our networks, we remove q% of the edges from the networks, where q varies from 100% (full network) to 10% (incomplete network). Moreover, in order to simulate both noisy and incomplete data, we choose the 40% rewired networks as the starting point and then start removing edges from these networks. We evaluate the performance of the signatures on these noisy and incomplete networks.

Figure 5.9 shows the AUPR of the networks as the edge completeness parameter varies from 100% to 10%. The initial networks have been rewired with a 40% probability. All the signatures display a general downward trend. The GCV signature performs poorly also in this experiment, always having an AUPR that is smaller than the AUPR of the other signatures. Some signatures such as the RGFD have a sharper drop in their AUPR than other signatures such as the Spectral Distribution. This suggests that RGFD is not as robust to incomplete data as the Spectral Distribution is. We conclude that the GCV signature is unable to deal with incomplete data either.

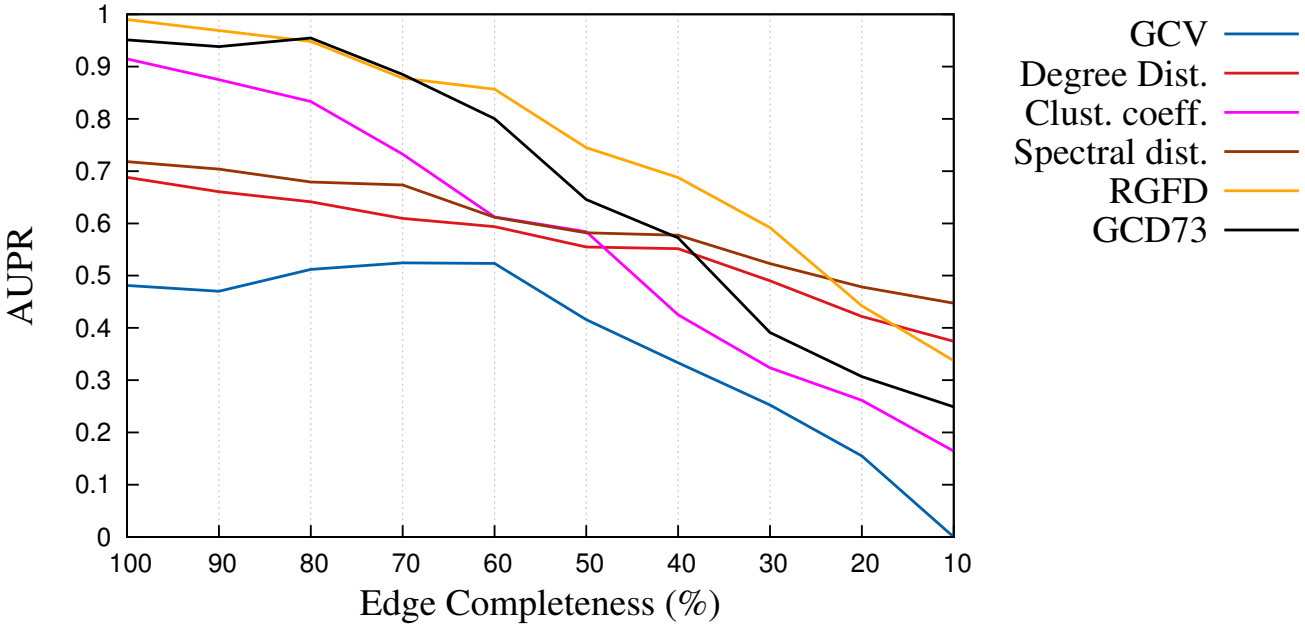


Figure 5.9: The AUPR for different percentages of edge completeness in the model networks. The GCV signature also has a poor performance when dealing with incomplete data, always having the lowest AUPR compared to the other signatures.

5.5.3 Signature approximation

In order to speed up computation, sometimes we have to approximate the signatures that are computed for all the random networks. In this section we try to evaluate the robustness of each signature to approximation. For each network, we only use a percentage p% of nodes to calculate the signatures. This is done for each signature/metric in the following manner:

1. Graphlet Cluster Vector (GCV): We compute the Pearson's GCV correlation matrix using the GCV signatures of only p% of the nodes.

2. Degree Distribution: We calculate the degree distribution from p% of the nodes in the graph.

3. Average clustering coefficient: We average only the clustering coefficient of $p\%$ of the nodes.

4. Spectral distribution: We compute the Laplacian matrix L of the original network, then randomly sample $p\%$ nodes and take the submatrix L' of L that corresponds to the sampled nodes. We compute the spectral distribution from the submatrix L'.

5. Graphlet Frequency Vector (used for computing RGFD): We randomly sample $p\%$ of the nodes and take the induced subgraph S on these nodes. We then compute the GFV in S.

6. GCD73: We compute the Graphlet correlation matrices using the GDV signatures of only $p\%$ of the nodes.

The experiments for signature approximation have also been run using the 40% rewired networks, which simulate noisy data. The results are presented in figure 5.10. For the GCV signature, we notice that it is actually robust to signature approximation, showing a very small but steady drop in the AUPR as less nodes are sampled. Other metrics such as the RGFD show a sharp drop in AUPR from 1.0 to 0.2, suggesting that RGFD is not robust to approximations. The Spectral distribution can also be considered robust, showing a peak AUPR of 0.7 when 50% of the nodes are sampled.

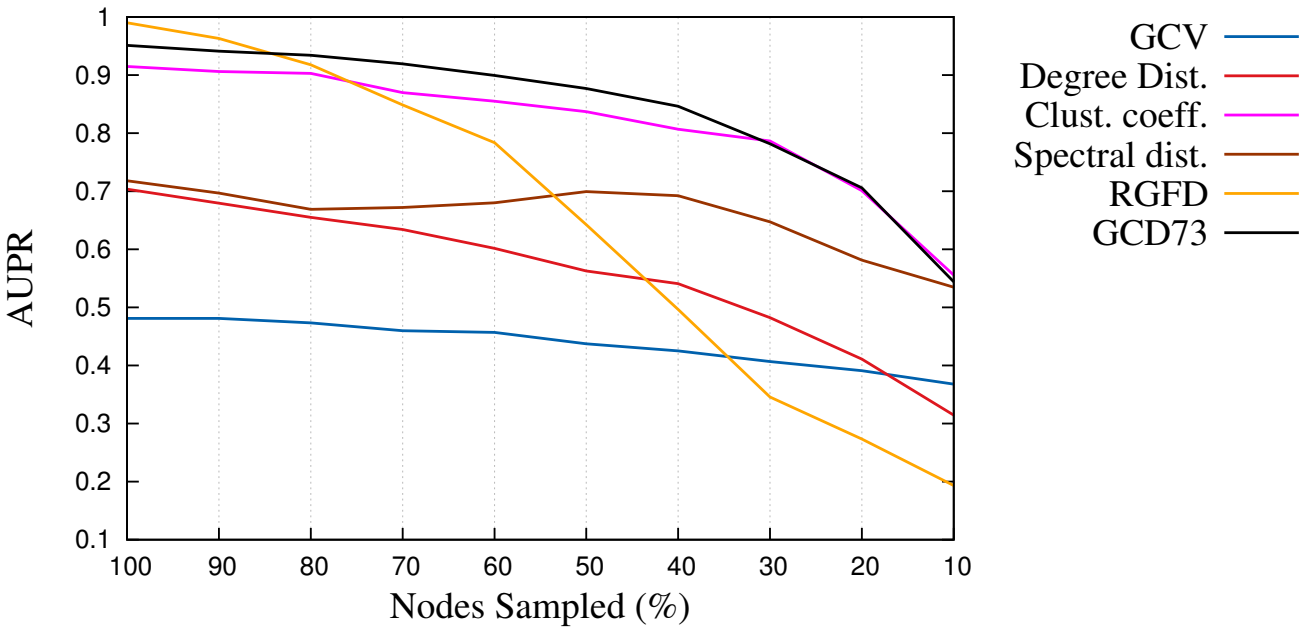


Figure 5.10: The AUPR for different percentages of nodes sampled in the model networks. The GCV signature is robust to approximation, showing only a slight but steady drop in AUPR as the percentage of nodes sampled varies from 100% to 10%. On the other hand, the RGFD shows a sharp drop, suggesting that it is not robust to signature approximation.

In conclusion, the novel GCV signature is not robust to noisy or incomplete data, but it is robust to signature approximation. The signatures that performed best on our tests are the RGFD and GCD73, which are mostly robust to noisy and incomplete data.

5.6 GCV-based Classifier

In this section we evaluate the performance of the GCV signature at classifying proteins into functional classes. We use Collin's Yeast AP-MS PPI network for computing the GCV of each

protein.[4] Separately, we label each protein using Boone's annotation that comprises 14 different classes. The classifier we wrote uses a K-nearest neighbours (K-NN) method for predicting the function of a protein in the following manner:

- Compute the GCV signature for all the proteins in the input network.
- For predicting the function of a given protein, compute the Euclidean distances between the GCV of the protein and the GCV of all the other proteins in the training data set. Store the distances in an array and sort it.
- Find the closest K data points to the input protein according to the computed distances.
- Perform majority voting[5] on the classes of the K nearest neighbours and return the result as the predicted class.

This process is run inside a Cross-validation framework, where the protein dataset is split into two groups:

- training data: this is stored in a data structure and is used for predicting the class of proteins using K-NN
- test data: this dataset is used for the actual prediction.

We split the dataset into 10 different chunks and run 10-fold cross-validation. We also choose $N = 5$ as the number of neighbours on which majority voting is performed. At each fold, 90% of the data is used for training and 10% for testing. For each fold during cross-validation, we compute a confusion matrix M for all the classes, where entry $M(i, j)$ corresponds to the number of data points that have actual label i, but the classifier predicted them as having label j. The confusion matrices for each fold are added together and a final confusion matrix is obtained at the end of the cross-validation process. From the final confusion matrix, we then count for each class C the following types of data points:

- **True Positives** (TP) are the data points that belong to C and have also been correctly predicted as belonging to class C.
- **False Positives** (FP) the data points that do not belong to C but have been incorrectly predicted as belonging to class C.
- **True Negatives** (TN) are the data points that not belong to C and have also been correctly predicted as not belonging to class C.
- **False Negatives** (FP) the data points that do not belong to C but have been incorrectly predicted as belonging to class C.

After we compute the number of TP, FP, TN and FN data points, we can calculate for each class C the following 3 statistics:

- Precision: the percentage of data points that have been correctly classified in C out of all the data points that have been classified in C. The exact formula for precision is:

$$Precision = \frac{TP}{TP + FP}$$

[4] The reason we run it on Collin's AP-MS network is because CCA analysis has given a high correlation on this dataset (see section 4.3.4).

[5] In majority voting, the class that has the highest frequency is the one that is returned. If two or more classes have the same highest frequency, one of them is chosen at random.

- Recall: the percentage of data points that have been correctly classified in C out of all the data points are in C. It is formally defined as:

$$Recall = \frac{TP}{TP + FN}$$

- F_1 score: it is a measure of the test's accuracy that combines both precision and recall. The formula for F_1 is as follows:

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

5.6.1 Classifier Results

Figure 5.11 shows the confusion matrix obtained for Collin's AP-MS Yeast PPI network using Boone's annotations. Note that only part of the classes have been tested in our classifier. The reason for this is because there were not enough data points for some classes to allow our algorithm to run properly. For example, there was only one sample belonging to the class *Cell cycle progression/meiosis* in our dataset. As a result, we ran our classifier only on 9 classes that did not originally give an F_1 score of zero.[6]

We observe that although our classifier correctly classified some data points (diagonal entries), there are considerable errors, especially in the first column (Nuclear transport). A considerable number of data entries have been incorrectly predicted as belonging to Nuclear transport, and we attribute this to the following bias in our methodology: during the majority voting phase, if two or more frequencies have the same highest score, the class with the lowest index is ultimately selected. As a result, the first class (Nuclear transport) is more likely to be selected as opposed to others. A closer look at the data further explains why some of the proteins are wrongly classified: there are many proteins that have the exact same GCV signature but different functional annotations. Most of these proteins tend to have a sparse neighbourhood, a fact that is easily noticed in the GCV signature, with many frequencies set to zero.

On the other hand, we also notice that there is a large number of True Positives for RNA processing (RNA proc.), Chromatin transcription (Chrom. transc.) and Ribosome translation (Rib. transl.). Ribosome Translation and Chromatin transcription also had a large cross-loading magnitude in the CCA analysis (see figure B.3). We can therefore conclude that the GCV signature is particularly suitable for analysing proteins that are involved in Ribosome translation and Chromatin transcription.

Finally, the classifier has not correctly classified any protein labelled with DNA replication, as there are no True Positives for this class. The reason for this might be because we only use $N = 5$ nearest neighbours, and if the classifier finds at least 3 proteins from Nuclear Transport (Nucl. trans.) that have the same GCV signature as a protein that belongs to DNA replication, it will instead be assigned a Nuclear Transport label instead. We have tried using a value of K that is bigger than 5, but that did not result in a better classification, having the final F_1 score approximately the same or lower.

[6] We first ran our classifier using all the classes and removed the classes for which the final F_1 score was zero.

actual \ pred.	Nucl. trans.	Chrom. seg.	RNA proc.	Chrom. transc.	DNA repl.	Prot. deg.	Golgi sort.	Metab.	Rib. transl.
Nucl. trans.	24	3	5	2	0	0	4	1	1
Chrom. seg.	41	23	5	3	0	3	1	0	1
RNA proc.	42	11	99	26	0	5	7	1	18
Chrom. transc.	62	21	20	95	0	6	6	1	11
DNA repl.	67	10	1	3	0	0	1	1	1
Prot. deg.	19	5	3	6	0	18	0	0	0
Golgi sort.	63	21	2	13	0	0	23	0	0
Metab.	72	4	4	2	0	0	2	7	4
Rib. transl.	35	10	31	23	0	0	0	1	60

Figure 5.11: Confusion matrix obtained on Collins AP-MS Yeast PPI network after 10-fold cross-validation, using Boone's annotations as classes. The rows represent actual classes, while columns represent predicted classes. The classes used were: Nuclear transport (Nucl. trans.), Chromatin segmentation (Chrom. seg.), RNA processing (RNA proc.), Chromatin transcription (Chrom. transc.), DNA replication, repair, HR cohesion (DNA repl.), Protein degradation (Prot. deg.), Golgi endosome vacuole sorting (Golgi sort.), Metabolism - mitochondria (Metab.) and Ribosome translation (Rib. transl).

Table 5.2 shows the Precision, Recall and F_1 rates for each of the 9 classes used by our classifier. At the bottom of the table, the average precision, recall and F_1 rates across all classes are shown. The results in the table show that the GCV-based classifier is not efficient at classifying proteins according to their function. The average precision and recall rates are only 0.41 and 0.31 respectively, while the average F_1 rate is 0.29. However, there is considerable variance in precision, recall and F_1 rates across the classes. For example, the classifier has a relatively high precision for classes such as Ribosome translation, Protein degradation, Golgi Endosome sorting and RNA processing. On the other hand, a class such as DNA replication has zero precision, meaning that no protein has been correctly classified to this class.

Overall, we conclude that the GCV-based K-NN classifier is not suitable for labelling proteins from PPI networks according to their function. Nevertheless, it still performs 3-4 times better than a random classifier, which would have average precision, recall and F_1 rates of around $\frac{1}{9} = 0.11$, when 9 classes are used. Last but not least, we also tried running the same classifier with different parameters N – nearest neighbours and F – fold numbers. When varying N, we got the best results when N was in the range [5,10], although the performance decreases only slightly for N greater than 10.

Class	Precision	Recall	F1
Nuclear-cytoplasmic transport	0.056	0.600	0.103
Chromatin segmentation	0.213	0.299	0.249
RNA processing	0.582	0.474	0.522
Chromatin transcription	0.549	0.428	0.481
DNA replication, repair, HR cohesion	0.000	0.000	0.000
Protein degradation proteosome	0.562	0.353	0.434
Golgi endosome vacuole sorting	0.523	0.189	0.277
Metabolism - mitochondria	0.583	0.074	0.131
Ribosome translation	0.625	0.375	0.469
Average	0.410	0.310	0.296

Table 5.2: Precision, recall and F_1 rates for each class used by the protein annotation classifier. At the bottom of the table, the overall average precision, recall and F_1 rates are given. The low scores in average precision (0.41), recall (0.31) and F_1 (0.29) suggest that our GCV-based classifier is not suitable for classifying proteins according to their function.

5.7 Evaluation Summary

We conclude that when clustering random networks generated with different algorithms, the GCV signature is not as efficient as other signatures or metrics. Nevertheless, the GCV can still be successfully used for data analysis and it helped us uncover interesting insights from the economic and biological networks. Moreover, the GCV might still be successfully used for clustering, if combined with other signatures such as the GDV.

Furthermore, our results do not precisely resemble those obtained by Yaveroğlu et al. in 2014 [37]. The reason for this is because we have not used the same source network to generate the random networks. Moreover, Yaveroğlu et al. have also done the precision-recall curve analysis on a larger number of networks. In our case, we considered that 150 generated networks are sufficient to perform the analysis and draw our conclusions.

Section 5.6 also showed that the GCV signature cannot be directly used as a classifier without further modifications. The K-NN classifier we built for Collin's AP-MS Yeast PPI network using Boone's annotations is not precise and has a low F_1 score of 0.29. The reason for this is because there are several proteins that have the exact same GCV signature but different functional labels. We therefore suggest an improved GCV signature that captures more information about the protein's neighbourhood. One possibility is the combination of GDV and GCV signatures into one vector of frequencies.

Chapter 6

Conclusion

6.1 Summary

At the beginning of the project we explored previous work that was done on studying node neighbourhoods in a network graph. We also studied other network analysis techniques that are based on graphlets and also correlation methods such as Pearson's and Spearman's correlation coefficients and Canonical Correlation Analysis. Next, we defined the mathematical model for our novel GCV signature that quantifies the neighbourhood structure around a particular node in a network graph. We then attempted to normalise each graphlet frequency in the GCV according to the theoretical maximum frequency of that graphlet in the neighbourhood graph. However, this turned out to be infeasible because of mathematical complexities, so we decided to normalise it only by dividing each frequency by the sum of all frequencies in the GCV.

The next step in our project was to implement an algorithm that computes the GCV signature for all the nodes in an input network. We learnt that using both an adjacency matrix and an adjacency list for representing the network allows us to perform a variety of graph operations much faster. However, after implementing the algorithm for computing the GCV signature we found out that the computation was taking between 5–10 hours for some large input networks such as PPI networks or un-thresholded World Trade networks. We therefore decided to parallelise the computation across multiple processes, which provided a speedup of order 5 for some networks that have a large number of nodes.

The GCV signature was then applied to three main classes of networks: World Trade networks, Protein-Protein Interaction networks and Metabolic networks. For each of these networks we computed the Pearson's GCV correlation matrices and normalised and hierarchically clustered them. We also computed Canonical Correlation analysis between the GCV signature and various node annotations. We found out that the best correlations and results are obtained for the WTNs, so we decided to focus more on these networks. We therefore calculated the change in normalised and un-normalised GCV correlation matrix over the period 1962-2010 and we found out that this yielded a correlation with the change in Crude Oil price (see sections 4.2.1 and 4.2.7). We also performed two CCA experiments on Economic integration, which showed that a country that is integrated in a trading bloc has a network of trading partners that is very clustered (section 4.2.3). Using the GCV cross-loadings obtained from CCA, we computed a *trading partners sparsity index* for a variety of countries. This index correlated with major economic and social events that affected those countries (section 4.2.8).

On the other hand, the results obtained for the yeast Protein-Protein Interaction networks (section 4.3.4) showed that the neighbourhood structure of a protein is influenced by its involvement in:

- Ribosome translation
- RNA processing

- Metabolism - mitochondria
- Golgi Endosome vacuole sorting

The CCA analysis on the human Metabolic networks showed that the neighbourhood structure of a protein is influenced by its involvement in:

- Cellular processes (section 4.4.5): Transport and Catabolism, Cell communication and Cell growth and death.
- Organismal systems (section 4.4.6): Environmental adaptation, Excretory systems, Digestive system and Circulatory system.
- Human Diseases (section 4.4.7): Cardiovascular diseases and Substance dependence.

In chapter 5, we have evaluated our novel GCV signature against 5 other comparable signatures:

1. Degree Distribution
2. Clustering Coefficient
3. Spectral distribution
4. Relative Graphlet Frequency distance (RGFD) (see definition 25 in section 2.3.3)
5. Graphlet correlation distance (GCD-73) (see definition 36 in section 2.5.3)

We used each of the signatures to cluster random networks generated using 5 different algorithms: Erdős-Rényi, Erdős-Rényi with the degree distribution of the real network, Geometric, Scale-free Barabási-Albert and Stickiness-based. We found out that the GCV signature performed worst of them all, meaning that it is unsuitable for being used in a classifier. Its performance was also relatively poor in robustness testing, when applied to noisy and incomplete data. The GCV also had a poor performance when used to classify proteins according to their function (section 5.6). Nevertheless, the project did not focus on classification but on implementation and data analysis, where the novel GCV signature helped us get important insights from the networks we analysed.

6.2 Critique

The novel GCV signature we have developed has several deficiencies we were aware of from the very beginning. First of all, it is only able to quantify the topological structure in the immediate vicinity of the node. As a result, it cannot capture the structure in the neighbours of a node that are at distances 2,3, ... away from it. Another deficiency of the GCV signature is that it doesn't assign a weight to each of the vector frequencies that would quantify how important a frequency is. A closer analysis might find that some of the frequencies are redundant or contain little information, in which case a low weight would be suitable for these frequencies. A similar analysis has been done on the GDV signature by Yaveroğlu et al. [37] and found that only 11 orbits out of 73 contained non-redundant information. This has resulted in an 11-element signature called GCD-11 that outperformed all other signatures in random graph clustering [37]. However, the timescale of our project did not allow us to study redundancies in the GCV signature.

In the evaluation section, we evaluated the performance of the GCV and several other signatures on random network clustering. One problem with our methodology is that we only ran the experiments on 5 random networks and using only 150 generated networks (excluding

the rewired and incomplete networks). The reason for this is because computing the signatures on all these networks takes around three hours and a total of 14GB of hard drive space, so scaling is not straightforward. These problems can be overcome by more efficient parallelisation techniques, such as running our experiments on a cluster of machines or using a Map-Reduce framework such as Hadoop that performs sharding.

Other problems we have experienced in this project have been related to inconsistencies between results when using the unnormalised GCV and the normalised GCV respectively. These inconsistencies have occurred for example when correlating the change in GCV correlation matrix with the changes in Crude Oil price (sections 4.2.1 and 4.2.7). We do not yet have an explanation for these inconsistencies and have commented on the results as they are. Moreover, for some of the results we also could not find a reason why several graphlets correlate with each other. More research needs to be done into these areas.

6.3 Future work

The GCV signature is one possible method to quantify the neighbourhood structure around a particular node but it is by no means the only signature one could develop for such purposes. As future work, one could try to derive several related signatures using different normalisation procedures or even combine the GCV with the older GDV signature into one composite signature. This allows for efficient use of both signatures at the same time. These newly developed signatures could be evaluated and applied on different networks in order to find out what hidden structures they can uncover.

Another idea that was suggested by Zoran Levnajić, one of Nataša Pržulj's collaborators, is to find out how important each of the elements from the *Graphlet Cluster Vector* is and assigning a weight to each of them. Redundant elements could get a low weight, while non-redundant elements could get a high weight. Using machine learning techniques or linear regression, optimal weights could be derived which make the signature more efficient for network clustering or classification. As it was previously mentioned, this kind of analysis has already been done on the other GDV signature by Yaveroğlu et al. [37], which identified a set of 11 non-redundant orbits and created a short signature made of these frequencies. This signature outperformed its counterparts and was then successfully applied to World Trade networks.

Another avenue for continuing research is to perform more experiments on each of the three main classes of networks in order to confirm the results obtained in this project and find potentially better interpretations for the observed phenomena. The timespan of the project did not allow us to run more experiments and tests on our data. For instance, one can do more case studies on the economic networks or correlate the GCV with other economic indices. Moreover, one could also apply the GCV signature for data analysis of other classes of networks, such as social networks (Facebook), hyperlink networks (World Wide Web), telecommunication networks or other types of biological networks such as gene regulatory networks, neuronal networks or signalling networks.

Finally, we hope that our work will help the scientific community better understand local properties of complex networks that can be used for data analysis. Ultimately, network analysis is a never-ending task: one can always find better ways to explain phenomena or behaviour. As networks change over time or become more complex, new models need to be developed that reproduce them as closely as possible.

Chapter 7

Bibliography

[1] Benno Schwikowski, Peter Uetz, and Stanley Fields. A network of protein–protein interactions in yeast. *Nature biotechnology*, 18(12):1257–1261, 2000.

[2] Alain Hertz and Dominique de Werra. Using tabu search techniques for graph coloring. *Computing*, 39(4):345–351, 1987.

[3] Monika R Henzinger. Hyperlink analysis for the web. *Internet Computing, IEEE*, 5(1):45–50, 2001.

[4] Tijana Milenkoviæ and Nataša Pržulj. Uncovering biological network function via graphlet degree signatures. *Cancer informatics*, 6:257, 2008.

[5] Ahmedin Jemal, Rebecca Siegel, Elizabeth Ward, Yongping Hao, Jiaquan Xu, Taylor Murray, and Michael J Thun. Cancer statistics, 2008. *CA: a cancer journal for clinicians*, 58(2):71–96, 2008.

[6] World Health Organization et al. Annex table 2: Deaths by cause, sex and mortality stratum in who regions, estimates for 2002. *The world health report*, 2004.

[7] Syed Asad Rahman and Dietmar Schomburg. Observing local and global properties of metabolic pathways:load points and choke points in the metabolic networks. *Bioinformatics*, 22(14):1767–1774, 2006.

[8] Muhammed A Yıldırım, Kwang-Il Goh, Michael E Cusick, Albert-László Barabási, and Marc Vidal. Drugtarget network. *Nature biotechnology*, 25(10):1119, 2007.

[9] Roger Guimera and Luis A Nunes Amaral. Modeling the world-wide airport network. *The European Physical Journal B-Condensed Matter and Complex Systems*, 38(2):381–385, 2004.

[10] Nataša Pržulj. Biological network comparison using graphlet degree distribution. *Bioinformatics*, 23(2):e177–e183, 2007.

[11] N Pržulj, Derek G Corneil, and Igor Jurisica. Modeling interactome: scale-free or geometric? *Bioinformatics*, 20(18):3508–3515, 2004.

[12] Oleksii Kuchaiev, Tijana Milenković, Vesna Memišević, Wayne Hayes, and Nataša Pržulj. Topological network alignment uncovers biological function and phylogeny. *Journal of the Royal Society Interface*, 7(50):1341–1354, 2010.

[13] Paul Erdős and Alfréd Rényi. On random graphs. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

[14] Mathew Penrose. *Random geometric graphs*, volume 5. Oxford University Press Oxford, 2003.

[15] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[16] Nataša Pržulj and Desmond J Higham. Modelling protein–protein interaction networks via a stickiness index. *Journal of the Royal Society Interface*, 3(10):711–716, 2006.

[17] Hawoong Jeong, Bálint Tombor, Réka Albert, Zoltan N Oltvai, and A-L Barabási. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–654, 2000.

[18] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *ACM SIGCOMM Computer Communication Review*, volume 29, pages 251–262. ACM, 1999.

[19] Lada A Adamic, Rajan M Lukose, Amit R Puniyani, and Bernardo A Huberman. Search in power-law networks. *Physical review E*, 64(4):046135, 2001.

[20] Duncan J Watts and Steven H Strogatz. Collective dynamics of small-worldnetworks. *nature*, 393(6684):440–442, 1998.

[21] Erzsébet Ravasz, Anna Lisa Somera, Dale A Mongru, Zoltán N Oltvai, and A-L Barabási. Hierarchical organization of modularity in metabolic networks. *science*, 297(5586):1551–1555, 2002.

[22] Richard C Wilson and Ping Zhu. A study of graph spectra for comparing graphs and trees. *Pattern Recognition*, 41(9):2833–2841, 2008.

[23] Thomas Thorne and Michael PH Stumpf. Graph spectral analysis of protein interaction network evolution. *Journal of The Royal Society Interface*, 9(75):2653–2666, 2012.

[24] Stanley Letovsky and Simon Kasif. Predicting protein function from protein/protein interaction data: a probabilistic approach. *Bioinformatics*, 19(suppl 1):i197–i204, 2003.

[25] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[26] Oleksii Kuchaiev, Marija Rašajski, Desmond J Higham, and Nataša Pržulj. Geometric denoising of protein-protein interaction networks. *PLoS computational biology*, 5(8):e1000454, 2009.

[27] Michael Lappe and Liisa Holm. Unraveling protein interaction networks with near-optimal efficiency. *Nature biotechnology*, 22(1):98–103, 2003.

[28] Edgar N Gilbert. Random graphs. *The Annals of Mathematical Statistics*, 30(4):1141–1144, 1959.

[29] Fereydoun Hormozdiari, Petra Berenbrink, Nataša Pržulj, and S Cenk Sahinalp. Not all scale-free networks are born equal: the role of the seed graph in ppi network evolution. *PLoS computational biology*, 3(7):e118, 2007.

[30] Mark Newman. *Networks: an introduction*. Oxford University Press, 2009.

[31] Stephen M Stigler. Francis galton's account of the invention of correlation. *Statistical Science*, 4(2):73–79, 1989.

[32] Joseph Lee Rodgers and W Alan Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[33] MH Fulekar. *Bioinformatics: Applications in life and environmental sciences*. Springer, 2009.

[34] Jack Cohen. *Statistical power analysis for the behavioral sciencies*. Routledge, 1988.

[35] Ann Lehman. *JMP for basic univariate and multivariate statistics: a step-by-step guide*. SAS Institute, 2005.

[36] Jerome L Myers, Arnold D Well, and Robert Frederick Lorch. *Research design and statistical analysis*. Routledge, 2010.

[37] Ömer Nebil Yaveroğlu, Noël Malod-Dognin, Darren Davis, Zoran Levnajic, Vuk Janjic, Rasa Karapandza, Aleksandar Stojmirovic, and Nataša Pržulj. Revealing the hidden language of complex networks. *Scientific reports*, 4, 2014.

[38] Oleksii Kuchaiev and Natasa Przulj. Learning the structure of protein-protein interaction networks. In *Pacific Symposium on Biocomputing*, volume 14, pages 39–50, 2009.

[39] Gabor J Szekely and Maria L Rizzo. Hierarchical clustering via joint between-within distances: Extending ward's minimum variance method. *Journal of classification*, 22(2):151–183, 2005.

[40] Robert R Sokal. A statistical method for evaluating systematic relationships. *Univ Kans Sci Bull*, 38:1409–1438, 1958.

[41] Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3-4):321–377, 1936.

[42] William W Cooley and Paul R Lohnes. *Multivariate data analysis*. J. Wiley, 1971.

[43] Peter S Fader and Leonard M Lodish. A cross-category analysis of category structure and promotional activity for grocery products. *Journal of Marketing*, 54(4), 1990.

[44] R Mohan Pisharodi and C John Langley. Interset association between measures of customer service and market response. *International journal of physical distribution & logistics management*, 21(2):32–44, 1991.

[45] J Douglas Carroll, Paul E Green, and Anil Chaturvedi. *Mathematical tools for applied multivariate analysis (rev.* Academic Press, 1997.

[46] KH Young. Yeast two-hybrid: so many interactions,(in) so little time... *Biology of reproduction*, 58(2):302–311, 1998.

[47] Leandra M Brettner and Joanna Masel. Protein stickiness, rather than number of functional protein-protein interactions, predicts expression noise and plasticity in yeast. *BMC systems biology*, 6(1):128, 2012.

[48] Shoshana J Wodak, James Vlasblom, Andrei L Turinsky, and Shuye Pu. Protein–protein interaction networks: the puzzling riches. *Current opinion in structural biology*, 23(6):941–953, 2013.

[49] Gene Ontology Consortium et al. The gene ontology project in 2008. *Nucleic acids research*, 36(suppl 1):D440–D444, 2008.

[50] Christian Von Mering, Roland Krause, Berend Snel, Michael Cornell, Stephen G Oliver, Stanley Fields, and Peer Bork. Comparative assessment of large-scale data sets of protein–protein interactions. *Nature*, 417(6887):399–403, 2002.

[51] Michael Costanzo, Anastasia Baryshnikova, Jeremy Bellay, Yungil Kim, Eric D Spear, Carolyn S Sevier, Huiming Ding, Judice LY Koh, Kiana Toufighi, Sara Mostafavi, et al. The genetic landscape of a cell. *science*, 327(5964):425–431, 2010.

[52] Kyoto Encyclopedia of Genes and Genomes (KEGG). http://www.genome.jp/kegg/, 2014.

[53] EcoCyc - a scientific database for the bacterium Escherichia coli K-12 MG1655. http://www.ecocyc.org/, 2014.

[54] BioCyc - a collection of 3530 Pathway/Genome Databases (PGDBs). http://biocyc.org/, 2014.

[55] Edwin C Webb et al. *Enzyme nomenclature 1992. Recommendations of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology on the Nomenclature and Classification of Enzymes.* Number Ed. 6. Academic Press, 1992.

[56] UN Comtrade. http://comtrade.un.org/, 2014.

[57] Imf - world economic outlook (weo) database. http://www.imf.org/external/pubs/ft/weo/2012/02/weodata/index.aspx, 2006.

[58] Alan Heston, Robert Summers, and Bettina Aten. Penn world table. *Center for International Comparisons at the University of Pennsylvania*, 2002.

[59] Leda graphs - file format description. http://www.algorithmic-solutions.info/leda_guide/graphs/leda_native_graph_fileformat.html, June 2014.

[60] Brian S Everitt, Sabine Landau, Morven Leese, and Daniel Stahl. Hierarchical clustering. *Cluster Analysis, 5th Edition*, pages 62–64, 2001.

[61] Jacob A Wegelin. A survey of partial least squares (pls) methods, with emphasis on the two-block case. 2000.

[62] Wojtek J Krzanowski and WJ Krzanowski. *Principles of multivariate analysis*. Clarendon, 2000.

[63] George AF Seber. *Multivariate observations*, volume 252. John Wiley & Sons, 2009.

[64] World trade organisation - regional trade agreements by country. http://rtais.wto.org/UI/publicPreDefRepByCountry.aspx, May 2014.

[65] Eric Langford, Neil Schwertman, and Margaret Owens. Is the property of being positively correlated transitive? *The American Statistician*, 55(4):322–325, 2001.

[66] François Lescaroux and Valérie Mignon. On the influence of oil prices on economic activity and other macroeconomic and financial variables*. *OPEC Energy Review*, 32(4):343–380, 2008.

[67] Haiyuan Yu, Leah Tardivo, Stanley Tam, Evan Weiner, Fana Gebreab, Changyu Fan, Nenad Svrzikapa, Tomoko Hirozane-Kishikawa, Edward Rietman, Xinping Yang, et al. Next-generation sequencing to generate interactome datasets. *Nature methods*, 8(6):478–480, 2011.

[68] Kevin R Brown and Igor Jurisica. Unequal evolutionary conservation of human protein interactions in interologous networks. *Genome biology*, 8(5):R95, 2007.

[69] Sean R Collins, Patrick Kemmeren, Xue-Chu Zhao, Jack F Greenblatt, Forrest Spencer, Frank CP Holstege, Jonathan S Weissman, and Nevan J Krogan. Toward a comprehensive atlas of the physical interactome of saccharomyces cerevisiae. *Molecular & Cellular Proteomics*, 6(3):439–450, 2007.

[70] Teresa Reguly, Ashton Breitkreutz, Lorrie Boucher, Bobby-Joe Breitkreutz, Gary C Hon, Chad L Myers, Ainslie Parsons, Helena Friesen, Rose Oughtred, Amy Tong, et al. Comprehensive curation and analysis of global interaction networks in saccharomyces cerevisiae. *Journal of biology*, 5(4):11, 2006.

[71] Haiyuan Yu, Pascal Braun, Muhammed A Yıldırım, Irma Lemmens, Kavitha Venkatesan, Julie Sahalie, Tomoko Hirozane-Kishikawa, Fana Gebreab, Na Li, Nicolas Simonis, et al. High-quality binary protein interaction map of the yeast interactome network. *Science*, 322(5898):104–110, 2008.

[72] Jeremy Mark Berg, Lubert Stryer, and John L Tymoczko. *Bioquímica*. Reverté, 2008.

[73] Donald Voet, Judith G Voet, and Charlotte W Pratt. Fundamentals of biochemistry. *Brisbane: John Willey and Sons*, 1999.

[74] George J Siegel, Bernard W Agranoff, R Wayne Albers, Stephen K Fisher, and Michael D Uhler. Basic neurochemistry. 1999.

[75] Xinyuan Li, Pu Fang, Jietang Mai, Eric T Choi, Hong Wang, XF Yang, et al. Targeting mitochondrial reactive oxygen species as novel therapy for inflammatory diseases and cancers. *J Hematol Oncol*, 6:19, 2013.

[76] Douglas R Green. Apoptotic pathways: the roads to ruin. *Cell*, 94(6):695–698, 1998.

[77] György Hajnóczky, György Csordás, Sudipto Das, Cecilia Garcia-Perez, Masao Saotome, Soumya Sinha Roy, and Muqing Yi. Mitochondrial calcium signalling and cell death: Approaches for assessing the role of mitochondrial calcium uptake in apoptosis. *Cell calcium*, 40(5):553–560, 2006.

[78] Heidi M McBride, Margaret Neuspiel, and Sylwia Wasiak. Mitochondria: more than just a powerhouse. *Current Biology*, 16(14):R551–R560, 2006.

[79] Tamiko Oh-Hama. Evolutionary consideration on 5-aminolevulinate synthase in nature. *Origins of Life and Evolution of the Biosphere*, 27(4):405–412, 1997.

[80] Michel F Rossier. T channels and steroid biosynthesis: in search of a link with mitochondria. *Cell calcium*, 40(2):155–164, 2006.

[81] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240. ACM, 2006.

[82] Nenad Mladenović and Pierre Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100, 1997.

[83] Jimeng Sun, Huiming Qu, Deepayan Chakrabarti, and Christos Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *Data Mining, Fifth IEEE International Conference on*, pages 8–pp. IEEE, 2005.

[84] Degree distributions - image source, May 2014.

[85] BioGRID - website. http://thebiogrid.org/, 2014.

[86] I2D PPI database - website. http://ophid.utoronto.ca/ophidv2.204/, 2014.

[87] Christian von Mering's lab page. http://www.isb-sib.ch/groups/zurich/bsb-von-mering.html, 2014.

[88] Yushi Jing and Shumeet Baluja. Pagerank for product image search. In *Proceedings of the 17th international conference on World Wide Web*, pages 307–316. ACM, 2008.

[89] Vesna Memišević and Nataša Pržulj. C-graal: Common-neighbors-based global graph alignment of biological networks. *Integrative Biology*, 4(7):734–743, 2012.

[90] James D Watson. The human genome project: past, present, and future. *Science*, 248(4951):44–49, 1990.

[91] Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

[92] Stephen F Altschul, Warren Gish, Webb Miller, Eugene W Myers, and David J Lipman. Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410, 1990.

Appendix A

Statistical results

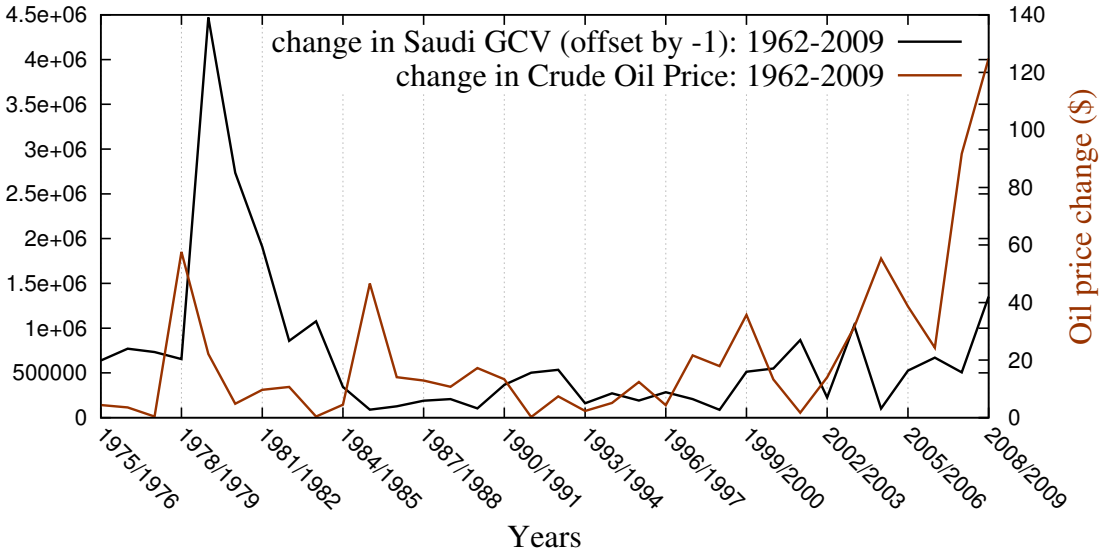


Figure A.1: The change in the un-normalised GCV of Saudi Arabia along with the change in Crude Oil price. The un-normalised GCV is not suitable for correlating the GCV of Saudi Arabia with the change in crude oil price, because Spearman's rank correlation coefficient is small (less than 0.13), while the p-value is large (bigger than 0.35). This is the case for all shifts in GCV in the range $[-2, 2]$

Appendix B

Canonical Correlation Tables

Canonical Correlation			0.89595
p-value			0.00000
X variate		Y variate	
POP	0.76667	G1	0.82332
LE	0.75818	G2	0.80569
KIxRGDPLxPOP	0.72552	G6	0.80315
RGDPCHxPOP	0.71889	G7	0.79365
RGDPLxPOP	0.71877	G3	0.78361
RGDPL2xPOP	0.71683	G4	0.78270
KCxRGDPLxPOP	0.69962	G17	0.77813
KGxRGDPLxPOP	0.69344	G22	0.76781
KCxRGDPL	0.42077	G23	0.76656
RGDPCH	0.26634	G24	0.76458
RGDPL	0.26629	G14	0.76293
RGDPL2	0.26616	G26	0.75734
KGxRGDPL	0.21884	G8	0.75020
KIxRGDPL	0.17869	G13	0.74857
XRAT	0.11999	G19	0.74581
KC	0.09780	G28	0.74525
KI	-0.05999	G18	0.74056
BCAperRGDPL	-0.14775	G12	0.74053
KG	-0.17422	G11	0.73369
BCA	-0.20019	G10	0.72915
OPENK	-0.24745	G9	0.72424
		G27	0.70899
		G29	0.68727
		G21	0.67142
		G5	0.66142
		G25	0.63754
		G16	0.62203
		G15	0.57546
		G20	0.44454

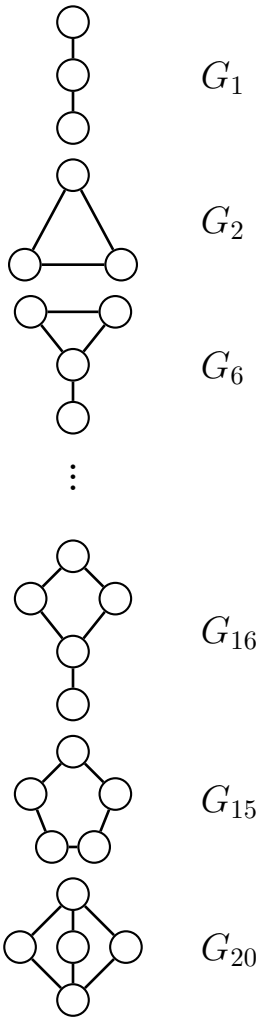


Figure B.1: Canonical Correlation Analysis between economic indicators (X variate) and the unnormalised GCV signature (Y variate) on the 2010 World Trade network. Openness (OPENK), Balance Current Account (BCA) a few other indicators correlate negatively with all the graphlets, because their cross-loadings have different signs. On the other hand, the rest of the indicators such as Population (POP), Level of Employment (LE) and GDP per capita (RGPDL, RGDPCH) correlate positively with all the graphlets, since their cross-loadings have the same sign. The overall correlation is 0.89 with a p-value of 0.0. In reality, the p-value is extremely small but it has been truncated to zero because of floating point approximations.

Canonical Correlation			0.94594
p-value			0.00000
X variate		Y variate	
POP	0.73628	G12	0.90456
LE	0.71650	G10	0.89337
KI x RGDPL x POP	0.66038	G14	0.87536
RGDPCH x POP	0.65383	G17	0.85955
RGDPL x POP	0.65376	G9	0.84692
RGDPL2 x POP	0.65226	G11	0.83708
KG x RGDPL x POP	0.64238	G19	0.70966
KC x RGDPL x POP	0.63303	G4	0.69198
KC x RGDPL	0.29252	G16	0.67490
XRAT	0.17083	G3	0.63019
RGDPCH	0.16079	G18	0.60564
RGDPL	0.16071	G24	0.59760
RGDPL2	0.16038	G13	0.59247
KG x RGDPL	0.15848	G22	0.54531
KI x RGDPL	0.10411	G23	0.47154
KC	0.08634	G15	0.43876
KI	-0.01620	G21	0.32221
BCA per RGDPL	-0.10953	G20	0.28966
KG	-0.12868	G26	0.27068
BCA	-0.14935	G6	0.23057
OPENK	-0.26502	G27	0.15386
		G25	0.14823
		G5	0.11232
		G28	-0.15016
		G1	-0.16367
		G7	-0.21277
		G2	-0.48656
		G29	-0.52462
		G8	-0.63741

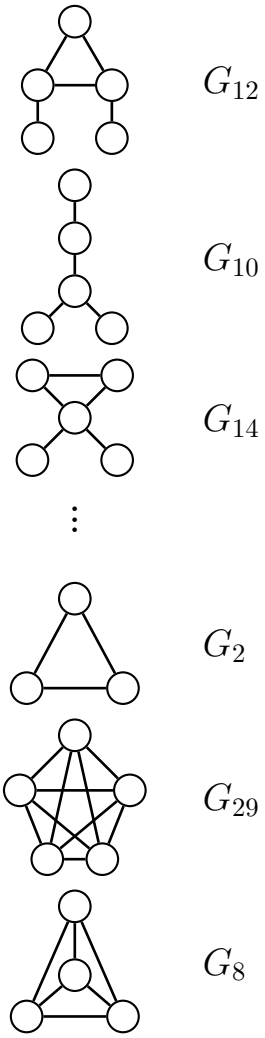


Figure B.2: CCA between the economic indicators and the normalised GCV signature on the 2010 World Trade network. Each graphlet has been colour-coded according to its density, from sparse graphlets in blue to dense graphlets and cliques in red. One can notice how the sparse graphlets have a positive cross-loading, while the dense graphlets have a negative cross-loading. Sparse graphlets are correlated with good economic indicators such as Population (POP), Level of Employment (LE) and GDP per Capita (RGDPL), while dense graphlets are correlated with bad indicators such as the Balance of Current Account (BCA).

B.1 The 17 experiments

In this section we only show the results that were statistically significant. All the other results can be found in the source code under `final_results/all_ppi/`

Canonical Correlation			0.53013
p-value			0.00000
X variate		Y variate	
Ribosome translation	0.91618	G2	0.89916
RNA processing	0.08561	G8	0.86246
Protein degredation	-0.01381	G29	0.83575
Cell cycle	-0.01819	G7	0.81776
Nuclear cytoplasmic transport	-0.07635	G1	0.81549
ER Golgi traffic	-0.10132	G28	0.79973
Protein folding	-0.10205	G26	0.76710
Chromatin segmentation	-0.12005	G27	0.75955
Signaling stress response	-0.12897	G5	0.74980
Cell polarity morphogenesis	-0.14394	G6	0.73719
Chromatin transcription	-0.14560	G22	0.72618
DNA replication	-0.17095	G24	0.71387
Metabolism mitochondria	-0.17109	G25	0.70796
Golgi endosome vacuole sorting	-0.20098	G23	0.67823
		G4	0.65612
		G20	0.65406
		G17	0.63899
		G21	0.61750
		G19	0.59378
		G3	0.59369
		G16	0.54884
		G14	0.54406
		G18	0.53288
		G15	0.52898
		G12	0.52683
		G11	0.49169
		G13	0.41908
		G10	0.41706
		G9	0.38194

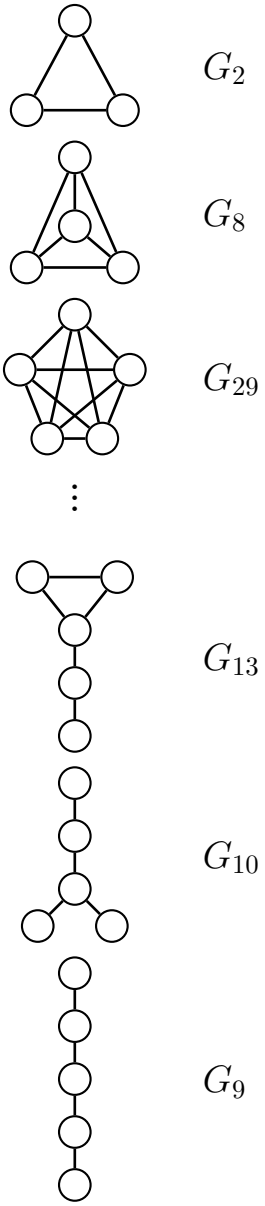


Figure B.3: CCA Analysis on Collin's AP-MS Yeast PPI network. The X variate is represented by Boone's protein annotations (see section 2.7.1), while the Y variate is represented by the GCV signature. The correlation value is 0.53 and the p-value is shown as 0.0 due to floating point approximations, although in reality it is very low but not exactly 0.0. RNA processing and Ribosome translation correlate positively with all the graphlets because their weights have the same sign, while the rest of the protein annotations correlate negatively with all the graphlets.

Canonical Correlation		0.34590	
p-value		0.00000	
X variate		Y variate	
Metabolism mitochondria	0.14787	G11	-0.05518
Ribosome translation	0.07535	G10	-0.08810
Cell polarity morphogenesis	0.05944	G9	-0.09217
RNA processing	0.05671	G14	-0.10260
Protein folding glycosylation cell wall	0.04637	G16	-0.11486
Signaling stress response	0.04030	G13	-0.11736
Cell cycle progression meiosis	0.03541	G12	-0.11781
Nuclear cytoplasmic transport	0.01806	G15	-0.11783
Golgi endosome vacuole sorting	0.01673	G4	-0.12001
Protein degredation proteosome	-0.00325	G3	-0.13304
ER Golgi traffic	-0.01553	G20	-0.13733
Chrom seg kinetoch spindle microtub	-0.03107	G18	-0.13930
DNA replication repair HR cohesion	-0.21915	G17	-0.13935
Chromatin transcription	-0.23242	G21	-0.14204
		G19	-0.14416
		G25	-0.16429
		G5	-0.16538
		G22	-0.16542
		G24	-0.16634
		G6	-0.16798
		G23	-0.16926
		G1	-0.17776
		G27	-0.18491
		G26	-0.19023
		G7	-0.20120
		G28	-0.20952
		G2	-0.23224
		G29	-0.23269
		G8	-0.23425

Figure B.4: CCA Analysis on the BioGRID Yeast genetic network. The X variate is represented by Boone's protein annotations (see section 2.7.1), while the Y variate is represented by the GCV signature. The correlation value is 0.34 and the p-value is shown as 0.0 due to floating point approximations, although in reality it is very low but not exactly 0.0. Chromatin transcription and DNA replication correlate positively with all the graphlets because their weights have the same sign while Metabolism mitochondria correlates negatively. This suggests that genes involved in Chromatin transcription and DNA replication have a relatively dense neighbourhood, while genes involved in Metabolism mitochondria have a relatively sparse neighbourhood.

Canonical Correlation		0.45880	
p-value		0.00000	
X variate		Y variate	
Metabolism mitochondria	0.15861	G11	-0.03350
Golgi endosome vacuole sorting	0.09172	G14	-0.03915
Protein folding glycosylation cell wall	0.08794	G10	-0.04096
Cell polarity morphogenesis	0.07806	G4	-0.04261
DNA replication repair HR cohesion	0.07679	G9	-0.04999
Signaling stress response	0.06513	G16	-0.05459
ER Golgi traffic	0.05000	G20	-0.05641
Chrom seg kinetoch spindle microtub	0.04864	G12	-0.05703
Cell cycle progression meiosis	0.03944	G17	-0.06682
Nuclear cytoplasmic transport	-0.01002	G3	-0.07415
Protein degredation proteosome	-0.01574	G13	-0.07585
Chromatin transcription	-0.02837	G22	-0.08361
RNA processing	-0.24003	G15	-0.08681
Ribosome translation	-0.35281	G18	-0.10240
		G19	-0.10562
		G1	-0.11901
		G21	-0.12887
		G6	-0.13115
		G23	-0.14507
		G5	-0.17207
		G24	-0.19117
		G25	-0.19158
		G26	-0.25346
		G27	-0.26491
		G7	-0.27551
		G28	-0.28974
		G29	-0.29937
		G8	-0.32430
		G2	-0.34523

Figure B.5: CCA on the BioGRID Yeast Full PPI network using Boone's annotations. Ribosome translation and RNA processing correlate positively with all the graphlets, while Metabolism mitochondria correlates negatively.

Canonical Correlation		0.48063	
p-value		0.00000	
X variate		Y variate	
Metabolism mitochondria	0.17069	G11	-0.01198
Cell polarity morphogenesis	0.10073	G4	-0.01502
Protein folding glycosylation cell wall	0.09465	G14	-0.01640
Golgi endosome vacuole sorting	0.08758	G10	-0.02782
Signaling stress response	0.08547	G17	-0.09114
DNA replication repair HR cohesion	0.08337	G1	-0.11310
ER Golgi traffic	0.05669	G23	-0.11908
Chrom seg kinetoch spindle microtub	0.05617	G12	-0.12373
Cell cycle progression meiosis	0.04399	G19	-0.12738
Nuclear cytoplasmic transport	0.01631	G16	-0.13678
Ribosome translation	0.00935	G9	-0.13785
Protein degredation proteosome	-0.05797	G18	-0.13982
Chromatin transcription	-0.24617	G15	-0.13997
RNA processing	-0.35707	G6	-0.14677
		G20	-0.14790
		G13	-0.15208
		G22	-0.15693
		G21	-0.16237
		G3	-0.16500
		G25	-0.16723
		G24	-0.17420
		G27	-0.17604
		G26	-0.18436
		G28	-0.18795
		G29	-0.19033
		G5	-0.19553
		G7	-0.23313
		G8	-0.25140
		G2	-0.32264

Figure B.6: CCA on the BioGRID Yeast high-confidence PPI network using Boone's annotations. Chromatin transcription and RNA processing correlate positively with all the graphlets, while Metabolism mitochondria and cell polarity morphogenesis correlate negatively.

Canonical Correlation		0.54489	
p-value		0.00000	
X variate		Y variate	
Cellular organisation	0.10410	G9	-0.21203
Uncharacterised	0.10038	G13	-0.23080
Genome maintenance	0.09481	G10	-0.23560
Other - metabolism	0.07261	G11	-0.27911
Cellular fate / organisation	0.06208	G15	-0.28931
Energy production	0.05791	G18	-0.29083
Protein fate	0.04958	G12	-0.29679
Amino acid metabolism	0.04792	G14	-0.30518
Transcriptional control	0.04232	G16	-0.30604
Stress and defence	0.03592	G3	-0.32479
Transport and sensing	0.03174	G19	-0.32981
Transcription	0.02139	G21	-0.34018
Translation	-0.54273	G17	-0.35616
		G4	-0.36201
		G20	-0.36383
		G23	-0.38103
		G25	-0.39269
		G24	-0.39272
		G22	-0.39991
		G6	-0.40577
		G5	-0.41118
		G27	-0.42175
		G26	-0.42604
		G1	-0.44139
		G28	-0.44763
		G7	-0.45248
		G29	-0.47622
		G8	-0.48968
		G2	-0.50543

Figure B.7: CCA Analysis on Collin's AP-MS Yeast PPI network [69]. The X variate is represented by von Mering's protein annotations (see section 2.7.1), while the Y variate is represented by the GCV signature. The correlation value is 0.54 and the p-value is shown as 0.0 due to floating point approximations, although in reality it is very low but not exactly 0.0. Translation has the strongest negative cross-loading and is the only annotation possitively correlated with all the graphlets from the Y variate. All other annotations are negatively correlated with all the graphlets.

Canonical Correlation			0.27203
p-value			0.00000
X variate		Y variate	
Uncharacterised	0.11238	G11	-0.00263
Translation	0.07793	G14	-0.04931
Other - metabolism	0.06314	G4	-0.05327
Transport and sensing	0.06161	G10	-0.06615
Energy production	0.04926	G9	-0.07092
Amino acid metabolism	0.04349	G16	-0.08171
Stress and defence	0.02417	G13	-0.08263
Cellular fate / organisation	-0.02093	G12	-0.08266
Transcription	-0.03456	G15	-0.08365
Protein fate	-0.04871	G17	-0.08782
Cellular organisation	-0.05765	G18	-0.08831
Transcriptional control	-0.15356	G20	-0.09333
Genome maintenance	-0.17739	G21	-0.09469
		G19	-0.09568
		G22	-0.09842
		G3	-0.09996
		G24	-0.10265
		G23	-0.10486
		G25	-0.10563
		G26	-0.11266
		G27	-0.11286
		G6	-0.11476
		G5	-0.11659
		G28	-0.12217
		G7	-0.13098
		G29	-0.13446
		G1	-0.13664
		G8	-0.14762
		G2	-0.16769

Figure B.8: CCA Analysis on the BioGRID Yeast genetic network. The X variate is represented by von Mering's protein annotations (see section 2.7.1), while the Y variate is represented by the GCV signature. The correlation value is 0.27 and the p-value is shown as 0.0 due to floating point approximations, although in reality it is very low but not exactly 0.0. Genome maintenance and Transcriptional have a strong positive correlation with all the graphlets, while Uncharacterised, Translation and Other - metabolism have the strongest negative correlation with all the graphlets.

Canonical Correlation		0.45779	
p-value		0.00000	
X variate		Y variate	
Uncharacterised	0.10778	G10	-0.03270
Other - metabolism	0.07615	G11	-0.03591
Transport and sensing	0.05434	G14	-0.03844
Energy production	0.04007	G4	-0.03870
Amino acid metabolism	0.03615	G9	-0.03897
Stress and defence	0.03498	G12	-0.05272
Cellular organisation	0.03431	G16	-0.05320
Cellular fate / organisation	0.03385	G20	-0.06397
Genome maintenance	0.02838	G3	-0.06528
Protein fate	0.02763	G17	-0.06662
Transcriptional control	0.01393	G13	-0.07353
Transcription	-0.11721	G22	-0.08133
Translation	-0.44068	G15	-0.08314
		G18	-0.10240
		G1	-0.11299
		G19	-0.12026
		G6	-0.13576
		G21	-0.13844
		G23	-0.17370
		G5	-0.18434
		G24	-0.21368
		G25	-0.22380
		G26	-0.29604
		G7	-0.30569
		G27	-0.31089
		G28	-0.34844
		G2	-0.37043
		G29	-0.37099
		G8	-0.37849

Figure B.9: CCA on the BioGRID Yeast Full PPI network using von Mering's annotations. The p-value is smaller than 0.05, suggesting that the correlation is statistically significant. Transcription and Translation correlate positively with all the graphlets, while Uncharacterised and Other - metabolism have the strongest negative correlation with the Y variate.

Canonical Correlation		0.42449	
p-value		0.00000	
X variate		Y variate	
Other - metabolism	0.09993	G11	0.00615
Transport and sensing	0.07195	G4	0.00505
Energy production	0.06707	G14	0.00443
Uncharacterised	0.06638	G10	0.00385
Cellular fate / organisation	0.05957	G9	0.00132
Amino acid metabolism	0.05531	G16	0.00025
Stress and defence	0.04321	G20	-0.00029
Cellular organisation	0.03877	G12	-0.01000
Protein fate	0.03514	G3	-0.01132
Genome maintenance	0.00376	G15	-0.01285
Translation	0.00362	G13	-0.01787
Transcriptional control	-0.08091	G17	-0.02307
Transcription	-0.40724	G1	-0.03294
		G23	-0.04304
		G19	-0.04477
		G6	-0.05256
		G21	-0.05642
		G5	-0.05950
		G25	-0.06383
		G18	-0.07042
		G27	-0.07725
		G22	-0.07798
		G24	-0.09178
		G26	-0.10001
		G28	-0.10303
		G29	-0.11274
		G7	-0.13958
		G8	-0.16503
		G2	-0.22352

Figure B.10: CCA on the BioGRID Yeast high-confidence PPI network using von Mering's annotations. The p-value is smaller than 0.05, suggesting that the correlation is statistically significant. Transcription and Transcriptional control correlate positively with all the graphlets, while Other - metabolism and Transport and sensing have the strongest negative correlation with the Y variate.