The Application of Spectral Clustering in Drug Discovery

A study submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

at

The University Of Sheffield.

by

Sonny Gan

This work was sponsored by

AstraZeneca

and

EPSRC

Information School

September 2013

Acknowledgements

First and foremost I would like to thank my Supervisors Val Gillet and Eleanor Gardiner of The University of Sheffield, and David Cosgrove of AstraZeneca, for not only giving me this opportunity but for their support and guidance from that point onwards. Special thanks must also go to Barry Pickup of The University of Sheffield and the late Andy Grant of AstraZeneca for their help and explanations of certain aspects of linear algebra.

Next, I would like to thank members of the CISRG, both past and present, for making my time in the group so enjoyable; so my thanks go to Mohammad Alkarouri, Sheerna Arif, Sorin Avram, Chia-Wei Chu, Edmund Duesbury, John Holliday, Nurul Malim, Iain Mott, George Papadatos, Nor Sani, Hua Xiang and Peter Willet. Special thanks must also go to Ben Allen, Richard Martin, Christoph “das Machine” Müller and Richard Sherhod for helping me get over the initial bumps of learning to program and making me feel so welcome in the initial months of this PhD and to Jorge Valencia and James “the Thesis fixer” Wallace for their constant willingness to engage in random discussions and for being sounding boards when I was struggling to fit together ideas.

For their willingness to provide a needed distraction from this research, my humble thanks go to my friends: Ben Corrie, Katie Norman, Adam Morgan, Sevcan Tanya Mehmet, Gary Marsh, Sultan and Shafia Ahmed.

I would also like to thank my Parents for their support and belief over the years, without which I am sure none of this would be possible. I guess I should also thank my younger Brother Teddy who I have spent too much time talking about matrix maths too, it is his fault though as he decided to take the Maths degree.

My final thanks goes to my Wife Katherine, who has shown the patience of a Saint in living with me through this period, and on a more serious note, has done an amazing job of keeping me sane.

I would like to finish by dedicating this work to my Nan, who passed away during the course of this research. I know she would be extremely proud to see the culmination of this project.

This work was made possible by funding from the EPSRC and AstraZeneca.

Abstract

The application of clustering algorithms to chemical datasets is well established and has been reviewed extensively. Recently, a number of 'modern' clustering algorithms have been reported in other fields. One example is spectral clustering, which has yielded promising results in areas such as protein library analysis. The term spectral clustering is used to describe any clustering algorithm that utilises the eigenpairs of a matrix as the basis for partitioning a dataset.

This thesis describes the development and optimisation of a non-overlapping spectral clustering method that is based upon a study by Brewer. The initial version of the spectral clustering algorithm was closely related to Brewer's method and used a full matrix diagonalisation procedure to identify the eigenpairs of an input matrix. This spectral clustering method was compared to the k-means and Ward's algorithms, producing encouraging results, for example, when coupled with extended connectivity fingerprints, this method outperformed the other clustering algorithms according to the QCI measure.

Although the spectral clustering algorithm showed promising results, its operational costs restricted its application to small datasets. Hence, the method was optimised in successive studies. Firstly, the effect of matrix sparsity on the spectral clustering was examined and showed that spectral clustering with sparse input matrices can lead to an improvement in the results. Despite this improvement, the costs of spectral clustering remained prohibitive, so the full matrix diagonalisation procedure was replaced with the Lanczos algorithm that has lower associated costs, as suggested by Brewer. This method led to a significant decrease in the computational costs when identifying a small number of clusters, however a number of issues remained; leading to the adoption of a SVD-based eigendecomposition method. The SVD-based algorithm was shown to be highly efficient, accurate and scalable through a number of studies.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

The University Of Sheffield.

Chapter 1

Introduction

Since the start of the twentieth century the average lifespan of a human being has almost doubled, going from around 45 in the early 1900s to over 80 in 2012 (Oregon, 2012). A number of factors have contributed to this rise, including improvements in sanitation and agriculture, but arguably the most crucial of these factors has been the significant improvement in our medical knowledge and the development of novel and innovative drugs, such as statins used to lower cholesterol; painkillers like aspirin; and antibiotics that are used to treat bacterial diseases (Kola and Landis, 2004).

Until the early 1930s drugs were typically discovered through the study and isolation of active ingredients from medicinal plants, for example, F. W. Sertürner isolated morphine from opium extract in 1815, or through serendipitous circumstances, such as the discovery of penicillin by the Nobel Laureate Alexander Fleming in 1928. Discoveries like that of penicillin, along with technological advances in analytical chemistry sparked a wave of pharmacies and chemical companies forming new pharmaceutical divisions aimed at capitalising on these newly

identified molecules by devising standardised procedures for their mass production (Drews, 2000). As technology advanced, particularly in the multidisciplinary fields of pharmacology and biochemistry, the focus within the pharmaceutical companies shifted to include research into new therapeutic compounds that could be used against newly discovered targets. This was the beginning of research and development within the new pharmaceutical industry and the earliest iteration of the modern drug discovery procedure. Over the course of the next 50 years, the drug discovery process was refined, incorporating new technologies and using new discoveries to shape the way drugs were designed. By the end of this period a medicinal chemist could typically synthesise and test around 100 - 200 compounds per year, however this dramatically changed with the introduction of combinatorial chemistry and high throughput screening (HTS) technologies in the early 1990s (Drews, 2000).

Combinatorial chemistry is the name given to a method of synthesis where a number of small compounds are reacted together in parallel to form a library of molecules. In a traditional synthesis two compounds A and B are reacted to form a new compound C, however in combinatorial synthesis this is expanded, such that a set of molecules A are reacted with a set B, producing a set of molecules C, which contains all possible combinations of A and B. So, if both sets A and B contain 10 molecules, the library C that is produced will contain 100 different compounds. This method allowed a medicinal chemist to move from producing a single molecule during a synthesis to producing libraries containing thousands of molecules, which could be analysed using HTS assays. The term HTS refers to a set of technologies utilised to screen large volumes of compounds for activity against a target in parallel within an assay. This technology revolutionised the process of drug discovery and is now a vital tool for medicinal chemists.

Nowadays, the term drug discovery is used to describe the process by which drugs, often known as new molecular entities (NME), are discovered or designed in the modern pharmaceutical industry. This process encompasses numerous stages including: the identification and selection of a biological target (typically a set of genes or a particular protein); the design of compound libraries and their screening for activity against the target; the selection of lead compounds and their optimisation; clinical trials; and finally the delivery of these drugs to market (Drews, 2000; Hann et al., 2001; Bleicher et al., 2003; Lipinski and Hopkins, 2004; Wermuth, 2004; Macarron et al., 2011). A typical drug discovery pipeline is provided in ***Figure 1.1***.

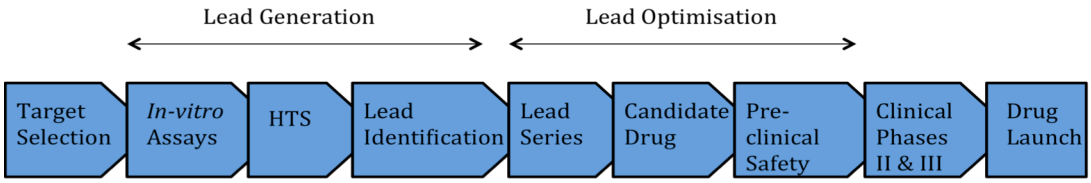


Figure 1-1: *The established drug discovery paradigm. Diagram based on figures from Lipinski and Hopkins (2004).*

Despite significant advances in all areas of modern drug discovery, it remains both financially costly and time consuming, due mainly to the comparatively low rate of new therapeutic drug identification and high attrition rate of potential candidate molecules. In fact, attrition rates are so high that in 2002 only 17 NMEs were approved by the Food and Drugs Administration, FDA, (pharmainfo.net, 2012), the lowest number of approvals since the early 1990s (Kola and Landis, 2004). The low number of NMEs brought to market around this time period led to the pharmaceutical industry exploring a number of new approaches to drug discovery (Sams-Dodd, 2005) including fragment-based drug discovery (Congreve et al., 2008) and genome-led personalised medicine (Ginsburg and McCarthy, 2001). These approaches, coupled with refinements of the general drug discovery process, have begun to show signs of reversing this trend, with the number of NME approvals rising to 30 in 2011 and 39 in 2012 (FDA, 2013). However, this improvement in the number of NMEs coming to market has done little to arrest the spiralling costs of drug discovery with it being estimated that the research and development phase for each new therapeutic compound cost approximately $1.8 billion dollars (Paul et al., 2010). As part of a concerted effort to combat the continually rising financial costs of bringing new therapeutic entities to market, there has been an ever-increasing use of in-silico methods throughout the various stages of drug discovery (Krejsa et al., 2003; Hert et al., 2005; Harper and Pickett, 2006; Ivanenkov et al., 2009). The rise in the use of computational methods within drug discovery has led to development of several fields of research that are concerned with aiding each of the various stages of this process, amongst those fields is Chemoinformatics.

Chemoinformatics is a field of Chemistry aimed at addressing chemical problems through the use of computational methods (Leach and Gillet, 2007) and it is within this field that this research sits. In many cases, effective solutions for a variety of chemical problems can be achieved through the use of data mining techniques, which aim to identify the underlying relationships within a dataset and to use this information to segment or group compounds. Whether a problem is selecting a representative subset of molecules from a large dataset; identifying potential scaffold hops; grouping molecules together in order to predict the

properties they are likely to exhibit; or separating a dataset into active and inactive molecules based upon the structure and activity of compounds, the data mining technique of choice is usually cluster analysis. Cluster analysis, or clustering, is the term used to describe a family of data mining techniques that aim to group data points, such that all data points within a cluster are similar, whilst data points in different clusters are dissimilar. There are a number of different approaches to clustering, one such method is spectral clustering, which is the focus of this research.

The aim of this thesis was to investigate the viability of using spectral clustering techniques – which have enjoyed widespread application in other scientific fields (Shi and Malik, 2000; Ng et al., 2002; Paccanaro et al., 2006) – with chemical data for a variety of tasks, in particular the separation of active and inactive molecules within an activity class. Our desire to investigate spectral clustering stems from two key features of these methods. Firstly, these algorithms have been shown to outperform all other clustering techniques within several fields and as a result are the method of choice within fields such as image segmentation and protein library analysis. Secondly, the ease at which these methods can be switched between forming overlapping and non-overlapping clusters from a dataset is of some interest, as not only does it allow the user to gain two different perspectives on the dataset being analysed, this step also comes at no additional computational cost.

This thesis begins with an introduction to the techniques utilised within this research. ***Chapter 2*** starts by providing a discussion of the central aspects of Chemoinformatics, for example, the similar property principle (Johnson and Maggiora, 1990) and molecular similarity; before continuing to describe some of the various data mining techniques used with chemical data, placing a particular emphasis on a group of unsupervised data mining methods called clustering algorithms. ***Chapter 3*** introduces the reader to a subset of clustering methods known as spectral clustering algorithms, which base the clustering of data points on the eigenpairs formed from an input matrix. This chapter includes an introduction to the mathematical concepts that underpin the implementation of spectral clustering, along with a review of the most commonly used methods and a detailed discussion of Brewer's novel application of spectral clustering to chemical data.

The experimental section of the thesis begins in ***Chapter 4*** with the development of a non-overlapping spectral clustering algorithm based upon the work of Sarkar and Boyer (1998) and a later study by Brewer (2007). This spectral clustering method was applied to the problem of separating active and inactive compounds within a number of well-known datasets. A robust

parameterisation of the spectral clustering method was carried out and the results of each application of spectral clustering quantified and compared to the leading clustering algorithms in the field. Although the algorithm provided interesting results, it was based on a full matrix diagonalisation procedure that has high associated time and storage costs, and hence was limited to datasets of 4000 molecules or fewer.

In ***Chapter 5*** the NOSC method was modified to allow the use of sparsely populated input matrices and a user-defined parameter for selecting the number of eigenpairs found (Golub and Van Loan, 1996; Parlett, 1998). The impact of using a sparsely populated input matrix was investigated and the effect it has on the performance of clustering was quantified. Although it was shown that sparse input matrices can be used with minimal impact on the results, spectral clustering remained computationally costly. Therefore, to overcome these issues the full matrix diagonalisation procedure was replaced with the Lanczos algorithm (Lanczos, 1950) in ***Chapter 6***, as suggested by Shi and Malik (2000), which has been used to speed up clustering in a variety of fields. The Lanczos-based spectral clustering method was compared to the spectral clustering algorithm from ***Chapter 5***, to determine the accuracy of its eigenpairs approximations, along with the costs and the comparative performance in clustering the activity classes used in ***Chapter 4***. The Lanczos-based spectral clustering algorithm was finally applied to datasets of increasing size to identify the maximum size of dataset to which this method could be applied.

Chapter 7 describes the development of an updated version of the spectral clustering algorithm that bases the identification of eigenpairs on a form of a singular value decomposition (SVD) algorithm. The SVD algorithm incorporates a number of preconditioning techniques and efficient input formats to maximise the computational efficiency of the algorithm and therefore minimise both the time and computational costs of clustering. The accuracy of the approximated eigenpairs was compared to those obtained using the modified spectral clustering algorithm from ***Chapter 5***. Scaling experiments were then undertaken, aimed at identifying the maximum size dataset that could be clustered. Finally the SVD-based spectral clustering algorithm was applied to two fragment-based drug discovery problems and the results compared to those obtained using the k-means method.

Chapter 8 discusses the development of an open source all-in-one spectral clustering tool that incorporates the SVD-based spectral clustering algorithm provided in ***Chapter 7***. The thesis concludes in ***Chapter 9*** by summarising the findings of this research and detailing possible future work.

The University Of Sheffield.

Chapter 2

Data Mining for Drug Discovery

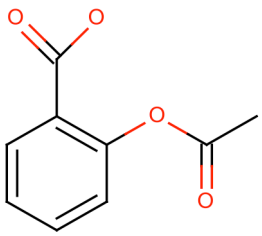
2.1 Introduction

The use of computational (or in-silico) methods to support medicinal chemist and biologists at various stages within the drug discovery process has been commonplace for a number of years now. These in-silico methods include both data mining and virtual screening techniques, where the term virtual screening (VS) is used to describe the computational analogue of biological screening and aims to rank, score and prioritise a set of structures using one or more computational tools (Leach and Gillet, 2007). For example, VS methods may be used to predict if a structure is likely to bind strongly to a protein's active site. The concept that underpins the utilisation of many in-silico methods is molecular similarity. This chapter initially provides an introduction to the concepts associated with calculating molecular similarity including molecular representation methods. The discussion then moves to the two major constituents of molecular similarity: molecular descriptors and similarity measures. Finally, a selection of data mining methods that have gained prominent use within the pharmaceutical industry are

discussed, placing a particular focus on the technique of cluster analysis, which is the basis of this research.

2.2 Molecular Representation

Since the discovery of chemical structures there has always been a requirement for this information to be represented in a form that can be easily communicated between chemists. Conventional representations include 2D structures and structural formulae, examples of which can be found in ***Figure 2.1***.



2D Structure

$C_9H_8O_4$

Chemical Formula

Figure 2-1: *Depicts the 2D structure and chemical formula of acetylsalicylic acid more commonly known as aspirin.*

Although these conventional methods of representation allow for the easy understanding and communication of information on chemical structures, they were not created with the consideration of computational representation in mind, and as a consequence they are not ideally suited for this purpose. This led to the development of further representations of chemical structures based upon graph theory.

Graphs are abstract structures containing nodes that are interconnected via edges and can be used to represent the topology of chemical structures. In a molecular graph, each node represents an atom and the edges are the bonds. It should be noted that hydrogen atoms are usually omitted from molecular graphs as their presence is considered implicit, with the number being deduced from the normal valence of the atoms they are bonded to. Properties can be associated with the nodes and edges, for example, atomic numbers may be associated with nodes and bond orders with edges (Leach and Gillet, 2007). As these graphical representations of structures can be generated in a number of ways, through the use of different numberings of the atoms and bonds, it is necessary to have a method that can identify if two graphs are identical. In graph theory, this is referred to as graph isomorphism and a number of well developed algorithms for identifying identical graphs have been

described in the computer science literature (Read and Corneil, 1977; Trinajstic, 1992). Possibly the most famous of these algorithms is the Morgan algorithm (Morgan, 1965). In the Morgan algorithm, numeric identifiers called connectivity values are assigned to each atom in an iterative process. The initial connectivity values are based upon the number of connections each atom has. Using the identifiers from the previous iterative step, new connectivity values for each atom are calculated as the sum of the connectivity values of the neighbours, until all atoms have been maximally disambiguated. A canonical numbering scheme for the atoms is then proposed using the final identifiers. The canonical numbering scheme of any molecule is unique, which greatly simplifies the problem of graph isomorphism.

The connection table (CT) provides a method for representing a molecular graph in the form of a table, and is generally used for the storage of molecules and transfer between programs. Simple connection tables are comprised of two sections: first is a list of the atoms contained in the molecule and second is a list of the bonds. More complex versions of connection tables may include more information such as hybridisation states of the atoms and bond orders. A simple connection table for aspirin can be found in ***Figure 2.2***.

```
  Marvin  12131014092D

 13 13  0  0  0  0            999 V2000
   -1.8268   -3.9187    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -2.5413   -4.3312    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -2.5413   -5.1563    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.8268   -5.5688    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.1123   -5.1563    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.1123   -4.3312    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -1.8268   -3.0937    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   -0.3978   -3.9187    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
   -2.4101   -2.5104    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
   -1.2434   -2.5104    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
    0.3166   -4.3312    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    1.0311   -3.9187    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    0.3166   -5.1562    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
  1  2  1  0  0  0  0
  1  6  2  0  0  0  0
  2  3  2  0  0  0  0
  3  4  1  0  0  0  0
  4  5  2  0  0  0  0
  5  6  1  0  0  0  0
  1  7  1  0  0  0  0
  6  8  1  0  0  0  0
  7  9  2  0  0  0  0
  7 10  1  0  0  0  0
  8 11  1  0  0  0  0
 11 12  1  0  0  0  0
 11 13  2  0  0  0  0
M  END
```

Figure 2-2: *Shows an example connection table for aspirin in MDL format, generated using the Marvin Sketch Software (ChemAxon, 2010).*

An alternative way to represent molecular graphs is through the use of linear notations. Currently the most commonly used form of line notation is the SMILES string, which stands for Simplified Molecular Input Line Entry Specification (Weininger, 1988). The extensive use of SMILES strings can be attributed to the ease of both their implementation and intelligibility.

SMILES strings are constructed by taking a *walk* through the chemical structure visiting each atom once. In SMILES each atom is represented by its chemical symbol with upper case characters denoting aliphatic atoms and lower case characters showing atoms in aromatic rings. The presence of single and aromatic bonds is inferred unless a specific example dictates otherwise, with double bonds being expressed with a "=" and triple bonds using a "#". Branches are shown via the use of brackets that are written next to the branch point atom, for example, the SMILES string CC(C=CC=C)C represents a 5-methylhexa-1,3-diene molecule. Furthermore, rings are dealt with by breaking one bond in the ring and appending an integer to each atom in the broken bond to show the break point, for example, a benzene ring is given as c1ccccc1. Information on chirality and geometric isomerism can also be added to SMILES strings using "@" to denote chirality, for example, L-alanine is shown using the SMILES string N[C@@H](C)C(=O)O, and either "/ /" to show a *trans*-isomer, for example, F/C=C/F representing E-difluoroethene, or "/ \" to show a *cis*-isomer, for example, Z-difluoroethene being described as F/C=C\F. To ensure a unique SMILES string for the molecule is generated, a canonicalisation technique is used, these methods are analogous to those used with molecular graphs, such as the Morgan algorithm. At one point in time the most commonly used canonicalisation algorithm with SMILES strings was called CANGEN (Weininger et al., 1989), an example of a canonical SMILES string produced using this algorithm is given in ***Figure 2.3***. Nowadays, each software package uses its own custom-built canonicalisation algorithm, the design of which is usually a closely guarded secret for each individual software producer, with the obvious exception of those used by open-source software.

CC(=O)Oc1ccccc1C(=O)O

SMILES String

Figure 2-3: *An example of a canonical SMILES string for aspirin.*

These graph-type representations were designed to allow the representation and easy transfer of chemical structures between programs and were not designed to compare the degree of chemical "likeness" between molecules. The discussion will now move to other types of molecular representations that were designed for use in quantifying the similarity between molecules, called molecular descriptors and the measure by which any two can be compared.

2.3 Molecular Similarity

The rationale that underpins many chemoinformatics techniques, such as virtual screening, is the similar property principle (Johnson and Maggiora, 1990) which states that structurally similar compounds typically exhibit similar chemical properties. Therefore, one would expect that close structural analogues of a biologically active molecule are also likely to show activity to the same target. It should be noted that this characteristic is also commonly referred to as neighbourhood behaviour (Patterson et al., 1996).

There have been several exceptions to the similar property principle noted within the literature, however these exception can be explained using the notion of a similarity cliff. To accurately describe the term similarity cliff, one must first consider the similarity paradox, which states that minor chemical modifications to highly similar molecules can significantly alter a molecule's activity (Kubinyi, 1998; Bajorath, 2002; Martin et al., 2002). Building upon this paradox the term similarity cliff is used to describe the point at which a small structural change leads to a dramatic change in the molecule's activity. For example, a similarity cliff is reached when a methyl group is added to the nitrogen of a kinase hinge binder, leading to a significant decrease in the activity of the molecule. Nevertheless, the similar property principle crucially provides an effective rule of thumb for the identification of novel bioactive molecules. As a result the quantification of the molecular similarity shared between a pair of compounds has become a vital aspect in virtual screening. In order to quantify the similarity between two molecules at least two independent components are required: a uniform method of representation for the data and a coefficient by which to quantify the similarity.

2.3.1 Molecular Descriptors

Molecular descriptors provide a way of representing structural or physicochemical properties of a molecule as numerical values to allow for further analysis or manipulation (Leach and Gillet, 2007; Bender et al., 2009). Since their initial conception, many different types of molecular descriptor have been described in the literature, with each varying in its complexity and the time required for its generation. It should be noted that the level of discrimination achieved by a molecular descriptor, is usually proportional to its complexity and the computational requirements for its creation (Brown and Martin, 1997). Molecular descriptors can be categorised into subsets according to the "dimensionality" of the data used to produce them. Hence, one-dimensional descriptors represent whole molecule properties that can be described by a single number. Descriptors that are based on two-dimensional structural

features, such as structural fragments and connectivity indices, are known as two-dimensional descriptors. Finally, three-dimensional descriptors are based on the three-dimensional information and features of molecules, for example, spatial relationships between atoms, hydrophobic centres and their molecular orbitals.

Some of the most commonly used molecular descriptors will now be discussed.

2.3.1.1 One-Dimensional Molecular Descriptors

One-dimensional molecular descriptors are numerical values that reflect global properties of molecules. The simplest one-dimensional descriptors are counts of molecular features such as hydrogen bond donors or acceptors, rotatable bonds and molecular weight. Other one-dimensional descriptors are based upon either experimentally determined or predicted physicochemical properties, for example, molar refractivity, which is the ratio of the speed of light through a vacuum compared to the speed of light through a sample of the compound, and LogP, which is the logarithm of the octanol/water partition coefficient (Atkins and Paula, 2009).

The representation of a molecule as a single number does not allow sufficient discrimination of molecules for most approaches. Consequently, multiple one-dimensional molecular descriptors are typically employed in unison to represent a molecule; for example, the use of numerical counts of hydrogen bond donors, hydrogen bond acceptors, molecular weight, experimentally determined LogP and molecular refractivity values together. When using multiple one-dimensional descriptors, standardisation of the descriptor values is vital to ensure they are equally weighted (Bajorath, 2001; Leach and Gillet, 2007).

2.3.1.2 Two-Dimensional Molecular Descriptors

Two-dimensional molecular descriptors are derived from two-dimensional representations of molecules, for example, connection tables. Numerous two-dimensional descriptors are commonly used in chemoinformatics ranging from topological indices (Randic, 1975; Hall and Kier, 2001) to two-dimensional molecular fingerprints and kappa shape indices (Hall and Kier, 1991).

Originally designed for the purpose of substructure searching, two-dimensional molecular fingerprints, otherwise known as 2D screens, are commonly used within the field of chemoinformatics for a variety of tasks. Two-dimensional molecular fingerprints can be divided into three main categories: structural keys, linear path hashed fingerprints and

extended connectivity fingerprints, although all molecular fingerprints share some common features, for example, they are all composed of a number of bins that are representative of structural fragments, environments and features of the molecules they describe, see ***Figure 2.4***.

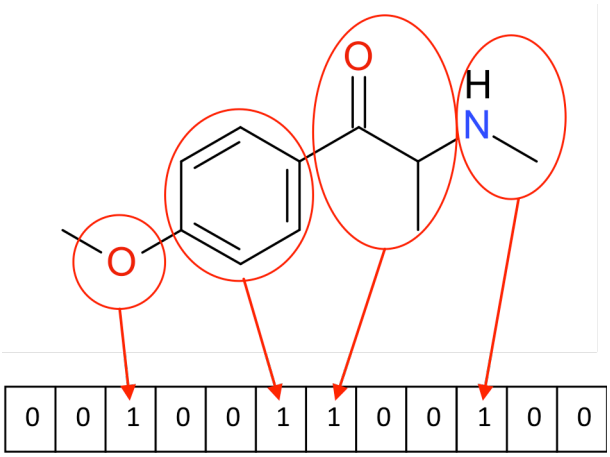


Figure 2-4: *Depicts an example of how bits are set in a molecular fingerprint for fragments of (RS)-1-(4-methoxyphenyl)-2-(methylamino)propan-1-one.*

Structural keys are two-dimensional molecular fingerprints based on the use of a fragment dictionary. A fragment dictionary is a pre-defined list of chemical fragments and features considered to be important for the recognition and separation of the molecules contained within a dataset. Each fingerprint is an array of binary values, with each bit representing the presence, 1, or absence, 0, of a particular structural fragment/feature in the molecule. Examples of structural keys are the BCI fingerprints (DigitalChemistry, 2005) and MDL public keys (Symyx, 2007).

Linear hashed fingerprints were designed to overcome the major disadvantage of structural keys, which is their lack of generality caused by the need for a fragment dictionary. Instead of using a fragment dictionary, linear hashed fingerprints encode all possible linear paths up to a specified number of bonds, typically 7. Each path is then input to a secondary procedure that uses a "hashing" algorithm to set a small number of bits to "1" in the fingerprint bitstring to represent the particular fragment. The drawback of these fingerprints is that each bit in the fingerprint may be set by more than a single fragment pattern, leading to collisions, and a lack of interpretability of individual bits. However the presence of these collisions does not produce falsely negative results in substructure searching. An example of linear hashed fingerprints are those used in the Daylight (Daylight, 2002) system. Furthermore, the SYBYL software (Tripos,

2007) utilises a hybridised method that combines the use of fragment definitions based on SYBYL line notation (SLN) to produce UNITY fingerprints.

Extended connectivity fingerprints (ECFP) are an example of circular molecular fingerprints. The term circular is used to describe fingerprints that are calculated through an iterative procedure similar to the Morgan algorithm (Morgan, 1965). In this procedure the substructures, features, paths and functional groups contained within concentric circles of increasing diameter, given as the number of bonds from each atom, are mapped and binned. The maximum diameter for the circles is user defined and is appended to the name of the circular fingerprint, such that ECFP_2 refers to a fingerprint containing circular substructures of up to a diameter of two bonds around each atom (Bender et al., 2004b; Hassan et al., 2006; Hu et al., 2009). An example of how part of an ECFP_6 fingerprint of a dimethyl-thiopyran-1-carboxamide molecule is generated is shown in ***Figure 2.5***. Where the features within 1, 2 and then 3 bonds of the central sulphur atom are mapped during three iterations.

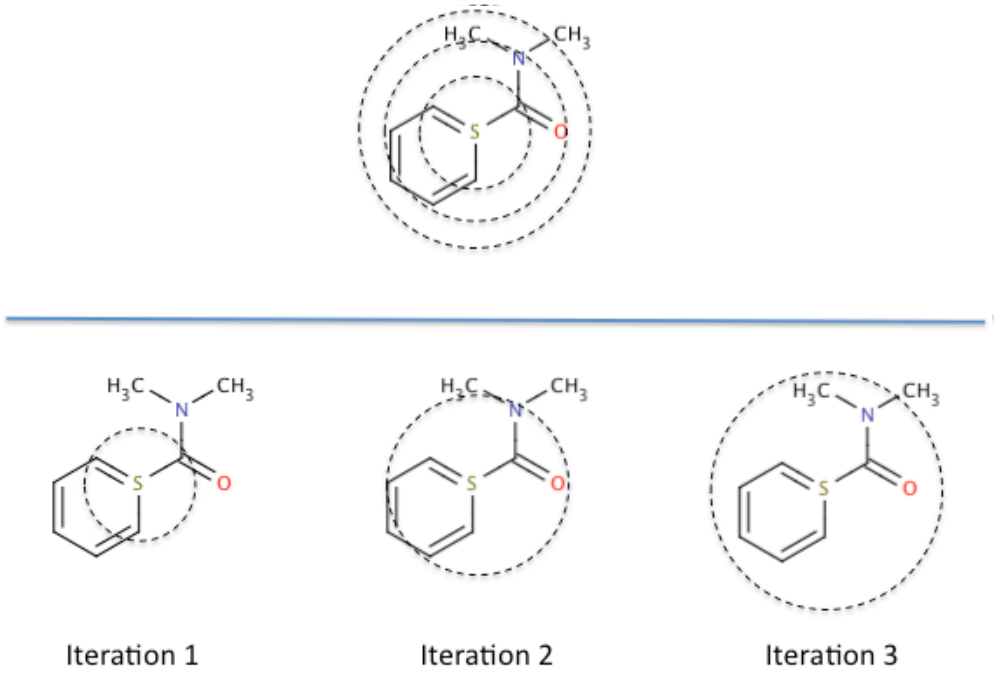


Figure 2-5: *Diagram depicting the construction of part of an ECFP_6 fingerprint for a dimethyl-thiopyran-1-carboxamide molecule.*

Circular fingerprints have become the descriptor of choice for the majority of similarity-based studies due to their ability to discriminate between even the most closely related compounds and the enhanced results this leads to. They can be generated simply within a number of software packages, for example, ECFPs within the Pipeline Pilot software (Accelrys, 2011) and circular fingerprints in the RDKit toolkit (Landrum, 2006).

2.3.1.3 Three-Dimensional Molecular Descriptors

Numerous three-dimensional descriptors have been reported in the literature for a range of uses in chemoinformatics. Here two types of three-dimensional descriptor are discussed; 3D screens and pharmacophore keys (Pickett et al., 1996).

Three-dimensional screens were initially developed for use in 3D substructure search systems, prior to a geometric search being carried out. These 3D screens usually encode the spatial relationships within a 3D substructure, such as the distances and angles between atoms and ring centroids. This information is encoded into a bitstring that contains a number of bins that represent angle or distance ranges. The UNITY system produces these bitstrings for 3D screens in two classes: rigid and flexible. Rigid screens encode the information from a single molecular conformation, whereas the flexible screens encode information for multiple conformations of a molecule by incrementally rotating each rotatable bond and recalculating the distances (Leach and Gillet, 2007).

Pharmacophore keys are an extension of the 3D fingerprints described above. A pharmacophore is defined as a set of structural features in a molecule that is recognised at a receptor site and is responsible for the molecule's biological activity (Gund, 1977). Thus, pharmacophore keys are based on the representation of atoms and substructures thought to be relevant in receptor binding. Each bit in a pharmacophore key represents a combination of three or four features and the distance between them. These features typically include hydrogen bond donors, hydrogen bond acceptors, hydrophobic atoms, hydrophobic centres and centres of aromaticity. In order to construct these 3 or 4-point binary pharmacophore keys, all possible combinations of pharmacophoric features and the distances between them must be enumerated so that they can be binned. For a more in depth review of these descriptors the reader is directed to a review by Willett (2009).

2.3.2 Similarity Coefficients

The need to determine a numerical measure of the similarity between two objects, each characterised by a common set of attributes, is common to a wide variety of disciplines including chemoinformatics, bibliographic information retrieval and psychology. This common requirement has led to the development of many coefficients for quantifying similarity; several of which have been re-invented for use within other disciplines, which has resulted in some coefficients being known by several names. The coefficients used in chemoinformatics can be

placed into one of three categories: association coefficients, which are commonly used with binary data and normalised to produce a value of between 0 and 1; correlation coefficients, which measure the degree of correlation between objects; and distance coefficients that quantify the dissimilarity between objects (Salim et al., 2002). A selection of some of the most commonly used similarity measures in chemoinformatics is provided in ***Table 2.1***.

The similarity measures in ***Table 2.1*** are given in both their continuous and dichotomous forms. The continuous forms of the similarity measures are used when the variables are integers or real numbers rather than binary values, whereas the dichotomous forms are applied when dealing with molecular fingerprints in which a bit can only have a value of either zero or one. Although each similarity measure is inherently different, they are implemented using an identical fundamental method, which is shown for binary data:

- Two molecular fingerprints are selected, a reference structure (A) and a query structure (B).
- The two molecular fingerprints are then compared bit by bit. If a bit is set to 1 in the reference structure and not in the query structure, then 1 is added to a counter a. Alternatively if a bit is not set in A, but is in B, a counter b is increased by one. When both bits are set to 1, c is incremented.
- The values of a, b and c are then fed into the dichotomous version of the desired similarity measure and a score produced.

When applying a similarity measure to a dataset, each of the N compounds is subjected to the method outlined above to calculate the respective similarity score for that pair of molecules. A typical application of similarity searching is to score and rank a database of compounds on their similarities to a single query compound. This is achieved by comparing each compound with the query in turn. The resulting scores can then be used to rank or segment the compounds via any chosen method, for example, a set of compounds may be ranked on their similarity to a target structure, with the top 10% chosen for further screening.

The most commonly implemented similarity coefficient used in chemical applications is the Tanimoto coefficient (Flower, 1998). The simplicity and the reliable performance of the Tanimoto coefficient has seen it applied in studies varying from the identification of Nearest Neighbours (Willett et al., 1986) to clustering (Downs and Barnard, 2002). Further discussions of similarity coefficients can be found in Willett et al. (1998) and Haranczyk & Holliday (2008).

An optional component in the calculation of similarity scores is the use of a weighting scheme, which provides a simple and effective way to emphasise either the presence or absence of a particular structural feature, through the application of a function. One example of a family of weighting schemes is those based on a Gaussian distribution; these methods can be used to place an increased/decreased emphasis on a set of similarity scores, by increasing or decreasing the distribution of the values. Further information on Gaussian functions can be found in ***Chapter 3***.

Name	Type	Formula for Continuous Variables	Formula for Dichotomous (Binary) Variables
Cosine Coefficient	Association	$S_{AB} = \frac{\Sigma x_{iA} \cdot x_{iB}}{[\Sigma(x_{iA})^2 \Sigma(x_{iB})^2]^{\frac{1}{2}}}$	$S_{AB} = \frac{c}{\sqrt{(a+c)(b+c)}}$
Dice Coefficient	Association	$S_{AB} = \frac{2\Sigma x_{iA} \cdot x_{iB}}{\Sigma(x_{iA})^2 \Sigma(x_{iB})^2}$	$S_{AB} = \frac{2c}{a+b+2c}$
Euclidean Distance	Distance	$D_{AB} = \left[\sum(x_{iA} - x_{iB})\right]^{\frac{1}{2}}$	$D_{AB} = \sqrt{a+b}$
Hamming Distance	Distance	$D_{AB} = \sum\lvert x_{iA} - x_{iB}\rvert$	$D_{AB} = a+b$
Soergel Distance	Distance	$D_{AB} = \frac{\sum\lvert x_{iA} - x_{iB}\rvert}{\sum \max(x_{iA}, x_{iB})}$	$D_{AB} = 1 - \frac{c}{a+b+c}$
Tanimoto Coefficient	Association	$S_{AB} = \frac{\Sigma(x_{iA} \cdot x_{iB})}{[\Sigma(x_{iA}) + \Sigma(x_{iB}) + \Sigma(x_{iA} \cdot x_{iB})]}$	$S_{AB} = \frac{c}{a+b+c}$
Tversky Coefficient	Association		$S_{AB} = \frac{c}{\alpha(a) + \beta(b) + c}$

Table 2-1: *Similarity, correlation and distance coefficients commonly used in comparing two molecules (A & B). For binary data "a" is defined as the number of bits unique to molecule A, "b" is the number of bits unique to molecule B, "c" is the number of bits set to "1" in both molecules A & B and "d" is the number bits set off in both A & B. Table based on (Willett et al., 1998; Leach and Gillet, 2007).*

2.4 Data Mining

Prior to the introduction of high-throughput technologies to the pharmaceutical industry in the mid 1990s, an experienced medicinal chemist would have typically been able to synthesise and test a maximum of around 100 molecules per year (Walters et al., 1998). The introduction

of combinatorial synthesis and high-throughput screening, HTS, dramatically changed this, with chemists being able to synthesise and test thousands or millions of compounds simultaneously.

As combinatorial and HTS methods have improved, there has been a dramatic increase in the sheer volume of data produced. Thus, the importance of effective analysis of this information to ensure it has maximal impact on the drug discovery process has led to the ever-increasing application of data mining techniques. Data mining has been described as *"the exploration and analysis, by automatic or semiautomatic means, of large quantities of data to discover meaningful patterns and rules"* (Berry and Linoff, 1997) and is now routinely implemented to aid in the identification of relationships within large multidimensional datasets. When applied to HTS data, the typical aim is to identify relationships between chemical structures and their corresponding activities, otherwise known as structure activity relationships or SARs (Harper and Pickett, 2006). Data mining methods are, however, not limited to this purpose and are used for a variety of tasks including the selection of diverse subsets of compounds in combinatorial library design.

An outline of some of the many different approaches to data mining used within drug discovery is given below. Data mining techniques can be characterised in a number of ways, however, here they will be discussed in two categories; supervised and unsupervised methods with a particular emphasis being placed on cluster analysis.

2.4.1 Supervised Data Mining Techniques

The term supervised learning refers to the machine-learning task of inferring a function or model from training data. This training data is comprised of training *examples* that contain input objects and their corresponding output values. Supervised learning methods analyse the training data and use it to establish and refine a model of the relationship between the input data and the output values. Once this training step has been completed the supervised learning method can be used to predict the output value for any valid input, for example, predicting if an untested molecule is likely to be active against a particular biological target.

Brief overviews of several supervised learning methods are presented below.

2.4.1.1 Substructural Analysis

First proposed by Cramer et al. (1974), substructural analysis (SSA) is based on the assumption that each individual substructural fragment in a molecule makes a constant contribution to the activity that is independent of other fragments in the molecule. By using this premise, SSA assigns a weight to a substructural fragment in accordance with the frequency in which it is found in active molecules compared to inactive molecules in a training set. The primary use for this method is to score and rank untested molecules to determine their respective merits for further research. However, SSA can also be used to identify fragments of interest, for example, those that are particularly prevalent in active molecules.

By assigning each fragment a weight, one can sum the weights of the fragments contained in a test compound to provide a score that can be used to indicate the likelihood of the molecule having activity against a particular biological target. The scores also allow for molecules in a dataset to be ranked according to their probabilities of activity, aiding in the selection of compounds for further testing (Leach and Gillet, 2007).

A multitude of different weighting schemes are available to assign the weights to the fragments. One of the most basic functions is shown in ***Equation 1***.

Equation 1: *A simple weighting scheme for substructural analysis.*

$$\omega_i = \frac{act_i}{act_i + inact_i}$$

Where, ω_i is the weighting of the i^{th} fragment; act_i is the number of active molecules that contain the i^{th} fragment; and $inact_i$ is the number of inactive molecules that contain the i^{th} fragment. It should also be noted that naïve Bayesian classifiers (Goldman and Walters, 2006; Hert et al., 2006) are closely related to SSA and have also been applied in the analysis of HTS datasets (Bender et al., 2004a; Xia et al., 2004; Rogers et al., 2005).

2.4.1.2 Feed Forward Neural Networks

Based on computational modelling of the brain, neural networks can be either supervised or unsupervised learning methods. Two different types of neural networks have found common application in chemistry; feed-forward neural networks and Kohonen networks (Gasteiger and Zupan, 1993; Schneider and Wrede, 1998). Kohonen networks are unsupervised learning techniques and hence will be discussed further in ***Section 2.4.2.1***.

Like the human brain, a feed-forward neural network consists of nodes, or neurons, connected to form a network. These nodes are generally organised into layers, with each node forming connections to all of the nodes in the adjacent layers. The structure of a feed-forward neural network is illustrated in ***Figure 2.6***.

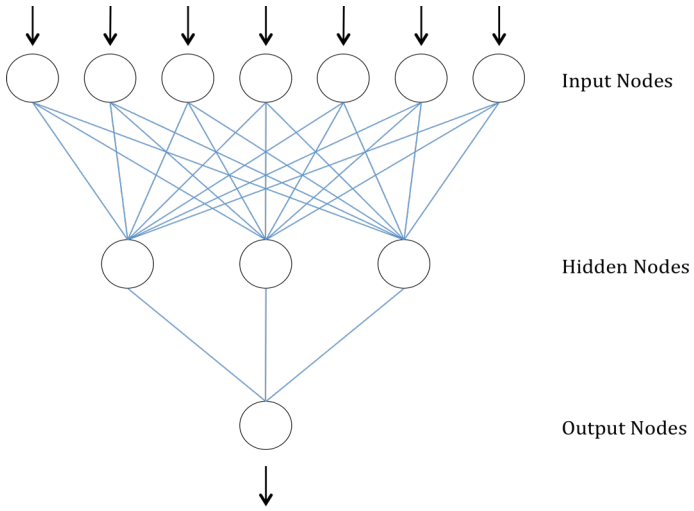


Figure 2-6: *Diagram shows the structure of a basic feed-forward neural network, which is split into three layers: an input layer, hidden layer and output layer. Diagram based on figure from (Leach and Gillet, 2007).*

Each node in the network exists in a state between 0 - 1, which is determined by the states and weights placed on the connections to the nodes in the previous layer. A feed-forward neural network can be split into three types of layers: the initial layer that is comprised of input nodes; the final layer that is comprised of output nodes and the hidden layer(s). The successful application of feed forward neural networks to a variety of problems has been enabled by the use of hidden layers and the back propagation algorithm (Rumelhart et al., 2002).

As feed-forward neural networks are supervised learning methods they require training before they can be used accurately. In the training step, the network is fed with inputs and corresponding outputs taken from a training dataset. The weights and other parameter values are initially set randomly and during training they are iteratively refined using a back-propagation algorithm until the correct outputs are reached. Once the feed-forward neural network has been trained it can be used to make predictions (Leach and Gillet, 2007).

2.4.1.3 Decision Trees

Decision tree learning is a commonly implemented method for data mining. When applied to chemical data, a decision tree can be described as a group of *rules*, which are used to associate specific molecular features with an activity or property.

The construction of a decision tree is a logical process, where at each stage a separation of the training data is carried out based upon the application of a rule, for example, is the molecular weight of the molecule above 80? Each rule leads to a single segmentation based upon whether the molecules do or do not comply with the rule. Several rules are examined and the one that produces the best separation of the data, into two groups, is selected. Each segment produced by the partition in the previous iteration is then subjected to further rules; this continues until no further partitions of the data are required, for example, the decision tree reaches a point at which the last partitioning of the data has produced a segment consisting of only active or inactive molecules. This process is commonly referred to as the decision tree reaching a terminal node. An example of a decision tree is provided in ***Figure 2.7***. In this example, 50 active and 12 inactive sumazole and isomazole analogues were introduced to the decision tree. Unseen molecules were subsequently classified by using their property values to traverse the tree until a terminal node was reached.

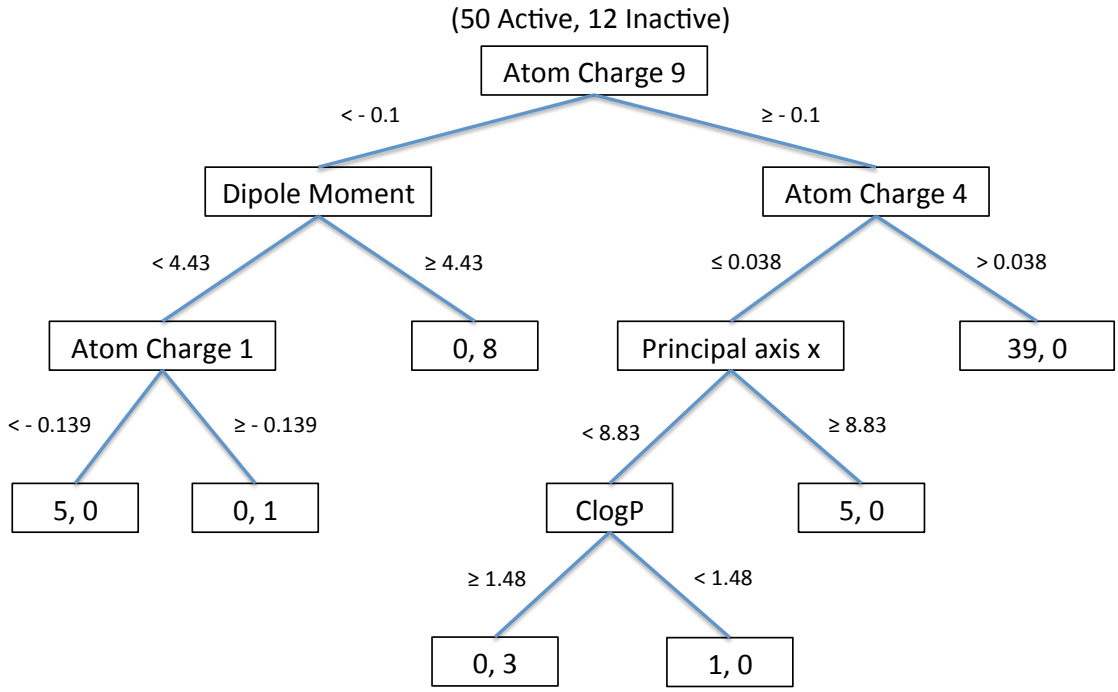


Figure 2-7: *Example of a decision tree, based on figure from (A-Razzak and Glen, 1992; Leach and Gillet, 2007), where a set of 50 active and 12 inactive sumazole and isomazole analogues are segmented into classes. The two numbers given in the terminal nodes provides the volume of both actives and inactives that are classified in each segment using the rules of the decision trees.*

For simple problems, decision trees produce data separations that are easy to interpret. However, for more complex problems, decision tree learning can be far less effective due to the number of rules required to separate intricate data that may lead to a loss of interpretability and overtraining. The term overtraining refers to the situation when the criteria for segmentation have become too specific to the training set and the tree is unable to classify unseen data correctly. This problem is not specific to decision tree learning and is common to all supervised machine-learning techniques, and as such, results in each method employing different steps to attempt to avert the issue.

2.4.1.4 Support Vector Machines

A support vector machine, or SVM, is a supervised learning technique that aims to identify a hyperplane that bisects a training set into two classes (Leach and Gillet, 2007). The position of the hyperplane is calculated through the use of training examples, which are referred to as support vectors. To avoid the problem of overtraining these support vectors are only taken from a subset of the training data. In the case of problems where a training set cannot be separated by a linear hyperplane, a kernel function is used to transform the data to a higher dimensionality, where it does become linearly separable. After a hyperplane has been placed, the test molecules are mapped to the feature space and their respective activities are predicted according to which side of the partition they are found on. The distance from the partition provides the user with a confidence level for each prediction such that, the confidence in the prediction increases with distance from the hyperplane (Leach and Gillet, 2007).

2.4.2 Unsupervised Data Mining Techniques

Unsupervised learning techniques do not use training data, and instead they identify the latent relationships that are present in a dataset, for example, patterns and relationships that underlie the data being analysed. A brief discussion of Kohonen neural networks is given below before ***Section 2.5*** introduces the unsupervised learning method called cluster analysis, which is more commonly known as clustering.

2.4.2.1 Kohonen Neural Networks

Kohonen neural networks are also known as self-organising maps or SOMs. These artificial neural networks typically consist of a rectangular array of nodes that are each associated with a vector corresponding to the input data, for example, the molecular descriptors. Each vector

is initialised to contain a few small random values. The input vectors are subsequently introduced to the network, one-by-one, and the distance between the input vector and each node vector is then determined with ***Equation 2***.

Equation 2: *Distance Metric for Kohonen neural network.*

$$d = \sum_{i=1}^{n} (x_i - v_i)^2$$

Where v_i is the value of the i^{th} component in the node vector and x_i is the value of the i^{th} component in the input vector and there are n molecular descriptors.

The node that is determined to have the minimum distance from the input vector is referred to as the winning node and its vector is updated according to the ***Equation 3***.

Equation 3: *Winning node update formula.*

$$v_i' = v_i + \eta(x_i - v_i)$$

Where η is the gain term, x_i corresponds to the input vector value, v_i is the node value and v'_i the updated node value.

In addition to updating the vector of the winning node, the neighbouring nodes are also updated via the same formula. This leads to the vector of the winning node and its neighbours becoming more similar to the input. In order to eliminate edge effects Kohonen networks can be constructed so that opposite sides of the network are joined together, which is demonstrated in ***Figure 2.8***. As each input is added to the network the gain term is gradually decreased along with the neighbourhood radius until the network is complete.

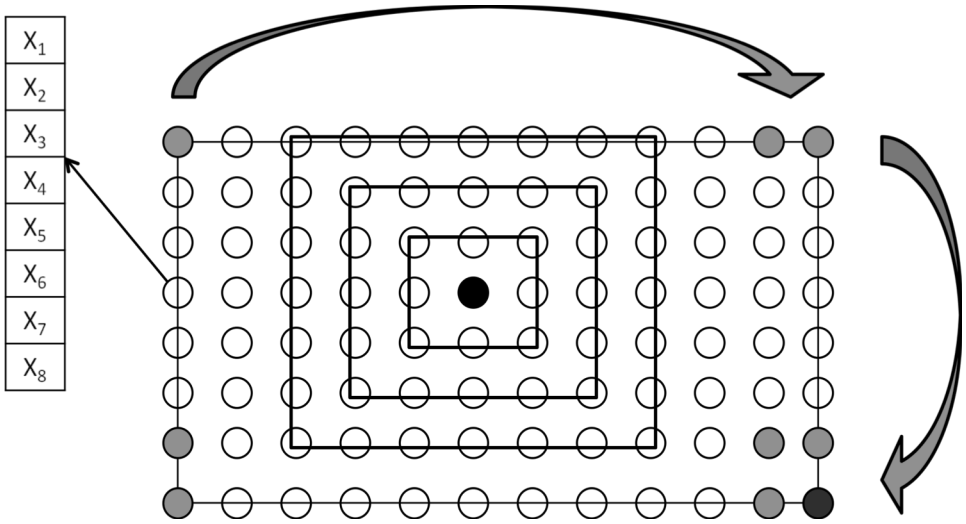


Figure 2-8: *Depicts a Kohonen network showing neighbourhood behaviour. The opposite sides of the network are joined together, as indicated by the arrows. Thus the immediate neighbours of the bottom right node would be those shaded in grey. Diagram based on figure from (Leach and Gillet, 2007).*

2.5 Cluster Analysis

The ability to analyse large heterogeneous datasets is a common requirement in a variety of scientific disciplines (Downs and Barnard, 2002; Bajorath et al., 2009). Clustering is a method of data mining that seeks to identify homogeneous subsets from within a heterogeneous dataset, based upon a similarity measure (Downs and Barnard, 2002). An ideal clustering of a dataset would hence produce several clusters such that all data points within a cluster are similar, according to the chosen similarity measure, and all points within different clusters are dissimilar (Willett, 1987).

Clustering techniques are categorised as unsupervised learning methods, as they do not take account of dependent variables during the analysis (Willett, 1987; Downs and Barnard, 2002). Clustering techniques have found widespread application in mathematics and science for use in diversity analysis, compound selection and data reduction, and they have generated particular pertinence in the area of chemoinformatics due to their use with high throughput screening data.

With the advent of combinatorial chemistry and high throughput screening technologies, pharmaceutical companies are able to produce and screen many thousands of compounds in a short period of time. This has led to the formation of both commercial and in-house databases containing millions of compounds. Although it is feasible to screen all of these compounds for

every new biological target that is investigated, opinions on the merits of such an approach within the pharmaceutical industry tend to oscillate over time. It should be noted that in certain circumstances some assays could never be developed at a high enough throughput to allow the testing of all compounds. Therefore, the use of clustering methods to assist in the selection of a small subset of molecules that are largely representative of all compounds available for screening is very appealing. Furthermore, clustering techniques have found use as a preliminary method of exploratory data analysis for any large chemical dataset in applications such as activity/property prediction and scaffold hopping (Böhm et al., 2004).

The overall process of clustering chemical data can be generalised into the following four steps:

- Generation of suitable molecular descriptors for each compound in the dataset.
- Selection of a similarity coefficient for quantifying similarity based on the descriptors.
- Selection of a suitable clustering algorithm.
- Analysis of the results from the clustering.

The selection of appropriate molecular descriptors, similarity coefficient and clustering method are pivotal to producing clusters that best represent the dataset in question. To be considered suitable, the selection should ideally produce a data separation where all similar compounds are clustered together; and when activity data is available, active and inactive compounds should appear in different clusters.

This section presents an overview of cluster analysis; including a discussion of commonly used, otherwise described as traditional, clustering algorithms and the studies that have assessed their results.

2.5.1 Non-Overlapping Clustering Algorithms

The term non-overlapping clustering applies to techniques in which each object in the dataset is added to a single cluster only during the analysis. These methods constitute the most commonly used approaches for clustering chemical datasets, due to their ease of implementation and the interpretability of the resultant clusters they produce. Thus, the discussion of clustering will be confined to these non-overlapping, or crisp, clustering methods in this chapter.

Non-overlapping clustering methods can be further divided into two sub-categories, hierarchical and non-hierarchical clustering methods (Willett, 1987; Downs and Barnard, 2002). A summary of each category will now be presented, focusing on their major features and their most commonly encountered implementations.

2.5.1.1 Hierarchical Clustering Methods

Hierarchical clustering techniques analyse datasets in an iterative manner, such that at each step a pair of similar clusters are merged or a larger cluster divided. This leads to a parent-child relationship between clusters at each adjacent level of the process. If clusters are merged together at each iterative stage of the process the hierarchical clustering is agglomerative. Alternatively, if the starting point is a single cluster containing all the data points that is iteratively divided into smaller clusters, the process is said to be divisive. Both of these methods can be visualised with the use of a dendrogram, an example of which is provided in ***Figure 2.9***, which shows how a dendrogram can be formed for a dataset of eight molecules. In divisive clustering the starting point is a single large cluster containing all eight molecules, which is divided iteratively to form the eight singletons as shown at the bottom of the diagram. Conversely agglomerative methods start at the bottom and fuse the singletons in an iterative manner until a final large cluster containing all compounds is reached.

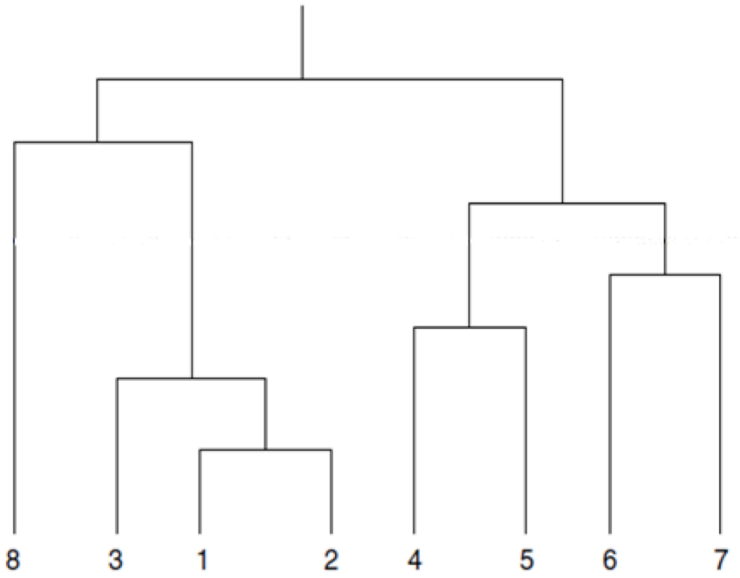


Figure 2-9*: An example of a dendrogram generated for eight clusters. Divisive methods begin at the top of this diagram and make multiple segmentations to reach the final eight clusters. Conversely in agglomerative clustering, clusters are joined at each clustering level until the combined cluster, shown at the top of the diagram, is reached.*

2.5.1.2 Hierarchical Divisive Clustering Methods

Most hierarchical divisive techniques are monothetic, meaning that each division is determined by a single descriptor. Each monothetic method differs by how the descriptor for the division is chosen, an example is the selection of the descriptor which maximises the distance between the resulting clusters (Downs and Barnard, 2002). Monothetic divisive methods are usually fast to execute but the results are often poor for chemical data. Some polythetic divisive methods have been described, however, they have not found wide application due to the computational resources required to carry them out. Barnard and Downs (2002) highlight one exception to this, called the minimum-diameter method (Guénoche et al., 1991). This method involves dividing the clusters at each iterative step such that the resultant cluster diameters are minimised, where the term cluster diameter refers to the maximum dissimilarity between any pair of data points found in a cluster. This task is accomplished by the minimum-diameter algorithm, which can be described in four steps:

- Generation of a sorted list of all pair-wise dissimilarities for the dataset, with the most dissimilar pairs listed first.
- The most dissimilar pair of points is selected and an initial division is performed in which each other data point is assigned to its closest point in the pair.
- The largest cluster is then selected and a further division is carried out as in the previous step.
- These divisions continue for a maximum of N-1 divisions, where N is the number of molecules in the dataset.

2.5.1.3 Hierarchical Agglomerative Clustering Methods

The most commonly implemented hierarchical agglomerative methods are those that belong to the family of sequential agglomerative hierarchical non-overlapping methods, otherwise known as SAHN methods. In SAHN methods each data point initially forms a singleton cluster and at each iterative step clusters are merged in order to group together those data points that are similar. In order to carry out these agglomerative methods a stored matrix algorithm is employed, the steps of which are as follows:

- An initial proximity matrix is calculated, which contains the proximities between all pairs of singletons in the dataset. Where the term proximity is used to define a

distance measure, for example, a proximity matrix may provide the Euclidean distance between two points in space.

- The most similar pair of clusters is highlighted in the matrix and merged to form a single cluster, and subsequently the matrix is updated with the new pairwise proximity values that are calculated between each singleton and the newly formed cluster.
- This continues until only a single cluster is produced.

For a dataset of N compounds, the stored matrix algorithm has a storage cost of $O(N^2)$ and a $O(N^3)$ time requirement for clustering, where O is defined as the order. This means that these hierarchical clustering techniques require a large number of computational operations to complete, increasing their time costs. However, one should note that these requirements can be decreased to O(N) and $O(N^2)$ in some cases, for example, when coupled with reciprocal nearest neighbours.

The methods in the SAHN family differ by how the proximity values are defined. The proximity calculation commonly uses the Lance-Williams matrix update formula (Lance and Williams, 1967), which is given below:

Equation 4: *Lance-Williams matrix update formula.*

$$d[k,(i,j)] = \alpha_i d[k,i] + \alpha_j d[k,j] + \beta d[i,j] + \gamma\,|d[k,i] - d[k,j]|$$

Where, *d[k, (i, j)]* is the proximity between cluster k and cluster formed from *i* and *j*; and α_i, α_j, β and γ are constants defined individually for each SAHN method (***Table 2.2*** for examples).

SAHN Method	α_i	α_j	β	γ
Complete Linkage	0.5	0.5	0	-0.5
Single Linkage	0.5	0.5	0	0.5
Group Average	$\frac{N_i}{N_i + N_j}$	$\frac{N_j}{N_i + N_j}$	$\frac{-N_i \times N_j}{(N_i + N_j)^2}$	0
Ward's	$\frac{N_i + N_k}{N_i + N_j + N_k}$	$\frac{N_j + N_k}{N_i + N_j + N_k}$	$\frac{-N_k}{N_i + N_j + N_k}$	0

Table 2-2: *Lance-Williams matrix update formula parameter values for a selection of commonly used SAHN methods. The parameters* $\boldsymbol{N_i}, \boldsymbol{N_j}$ *and* $\boldsymbol{N_k}$ *are the number of molecules in clusters i, j and k respectively. Table based on (Downs and Barnard, 2002).*

SAHN methods include single linkage clustering, where the proximity between two clusters is calculated as the minimum distance between any two points in different clusters, and the complete linkage method, where proximity is given as the maximum distance between any two data points in different clusters (Downs and Barnard, 2002). However, the most commonly used of the SAHN methods is the Ward's algorithm (also known as the minimum variance method), which seeks to form clusters with the smallest total variance. In Ward's method the proximity is defined as the variance between clusters, where variance is calculated as the sum of the squared deviations from the cluster mean. Therefore, at each iterative stage the Ward's method merges the clusters that give rise to the minimum change in square error. As each cluster produced by the Ward's method can be represented by its cluster-centroid. The Ward's algorithm is classified as a geometric method.

In 1983, Murtagh (1983) suggested adapting Ward's algorithm to include a reducibility property. The reducibility property states that *"for the merger of two clusters, a and b, to form cluster c, there cannot be another cluster, d, that is closer to c than to a or b"* (Downs and Barnard, 2002). As geometric methods satisfy the reducibility property, agglomerations can be performed on local areas of the proximity space, which are subsequently amalgamated to form the full hierarchy. This allows the stored matrix algorithm originally used in the Ward's method to be replaced with a reciprocal nearest neighbours algorithm (or RNN), which maps the proximities between data points with the aim of identifying all pairs of points that are reciprocally located nearest each other. The use of an RNN algorithm decreases the time constraints and storage costs for the Ward's clustering algorithm to $O(N^2)$ and $O(N)$ respectively. It should be noted that the implementation of a RNN algorithm, limits the Ward's method to using similarity measures that meet the criteria to be defined as a true metric. A true metric is defined as a metric which obeys the following rules (Pedoe, 1988):

- Non-negativity i.e. the *distance* (A, B) ≥ 0.
- Symmetric distances i.e. the *distance* (A, B) = *distance* (B, A).
- Triangular inequality, which states that the *distance (*A, C) ≤ *distance* (A, B) + *distance* (B, C).
- Identity of indiscernibles i.e. the *distance (*A, A*) = 0*.

An example of a true metric is Euclidean distance, which is used as the similarity metric for the Ward's algorithm to allow the use of reciprocal nearest neighbours, RNN.

2.5.1.4 Non-Hierarchical Clustering Methods

For non-hierarchical clustering methodologies, compounds are placed into clusters that have no predefined hierarchy. Although the many non-hierarchical clustering methods that have been reported in the literature can be further sub-categorised into a variety of families, the most widely implemented algorithms belong to just three:

- Single-pass and sphere exclusion.
- Nearest neighbour.
- Relocation.

2.5.1.5 Single-Pass and Sphere Exclusion Clustering Methods

Single-pass clustering methods cluster the data in a single scan, using a proximity threshold to either place a molecule into a pre-existing cluster or to place it in an entirely new cluster. The most commonly implemented single pass clustering algorithm is the leader algorithm, which produces clusters using the following approach:

- A proximity threshold, *t*, is defined.
- A molecule in the dataset is selected to act as a cluster centroid.
- Compounds that have proximity values that are greater than or equal to the threshold value *t*, are placed in the cluster with the centroid. Molecules with proximity values less than this threshold are used to start a new cluster.
- Molecules continue to be tested to identify if they fall within threshold of any of the centroids until all molecules have been placed into clusters.

Although these methods have the advantages of being fast and simple to implement, they do suffer from the disadvantage that the resulting clusters are dependent on the order the data is processed (Downs and Barnard, 2002). Other single pass clustering methodologies include sphere exclusion algorithms (Taylor, 1995; Butina, 1999).

2.5.1.6 Relocation Methods

The best-known relocation methods are those that belong to the k-means family, which contains a large number of variants (Pena et al., 1999; Holliday et al., 2004). The k-means algorithms all work by aiming to minimise the sum of the squared similarity scores between each molecule in a cluster and the cluster's centroid. The k-means approach that is most

commonly implemented for chemical applications can be broken down into a four-stage process:

- A set of k random seed molecules is selected to act as the centroid molecules of k clusters.
- Each compound is assigned to its nearest centroid.
- The cluster centroids are then recalculated and the molecules reassigned.
- This process continues until no further data movement occurs or until a specified number of iterations have been carried out.

Relocation clustering methods all suffer from similar limitations, in that selecting the correct number of seed molecules is difficult and the initial choice of seed compounds can lead to the method being adversely effected by outliers in the dataset, as these outliers will form singletons due to their dissimilarity to the molecules in the dataset. Another issue associated with k-means algorithms is that the iterative refinement of the centroid locations could lead to suboptimal centroids, as it would require the analysis of every combination of seed molecules to find the actual global optimum values for the centroids. Despite these issues, relocation clustering methods have found wide spread use in chemoinformatics due to their computational space efficiency, which is O(k), where k is the number of clusters, and time requirements, O(Nmk), where m is the number of iterations required to reach convergence, k is the number of clusters and N is the number of molecules in the dataset.

2.5.1.7 Nearest Neighbour Clustering Methodologies

Nearest neighbour clustering methodologies produce clusters by identifying the nearest neighbours of each molecule and using this information as the basis for assigning molecules to clusters. Despite a number of different nearest neighbour methods being devised for use with chemical data, Jarvis-Patrick clustering (Jarvis and Patrick, 1973) is used almost exclusively.

The implementation of Jarvis-Patrick clustering can be broken down into two stages. The first stage generates a list of the nearest neighbours, k, for each of the N compounds in a dataset. The proximity of the nearest neighbours is usually measured as either Euclidean distances or using the Tanimoto coefficient. The second stage scans the list of nearest neighbours and places molecules i and j in the same cluster if the following three conditions are met:

- Molecule i, is in the top k nearest neighbours of molecule j.
- Molecule j, is in the top k nearest neighbours of molecule i.

- Both molecules i and j have k_{min} of their top k nearest neighbours in common.

Where k and k_{min} are user defined (Downs and Barnard, 2002).

The Jarvis-Patrick method requires $O(N^2)$ time and has O(N) complexity, which are fairly low when compared to other clustering methods and is one of the main reasons why it originally gained prevalence for use with large datasets.

There is a major drawback to the use of Jarvis-Patrick clustering, which is that it produces clusters which can be largely dispersed, causing the formation of a number of singleton clusters and larger diverse clusters. However there have been a number of ways published to further improve the effectiveness of Jarvis-Patrick clustering (Leach and Gillet, 2007).

2.5.2 Comparing Clustering Methods

The efficiency and effectiveness of clustering techniques vary and they have their own characteristic merits and drawbacks. Therefore it is impossible to give a definitive answer to the question of which of the traditional clustering methods is the *best*. This is due to the fact that selection of a clustering technique is dependent on several factors including the molecular descriptors used, the chosen similarity coefficient and the size of the database.

When comparing clustering methods there are several different factors that must be evaluated such as the time required to carry out the analysis, the computational storage costs and how well the molecules are clustered. The first publication aimed at testing the quality of clusters produced by different methods was by Willett (1987), who carried out an extensive study of 30 hierarchical and non-hierarchical clustering techniques on 10 small datasets for which the properties were known. The comparison was based on property prediction and was carried out using the *leave-one-out approach*. The leave-one-out cross validation method is commonly used in the field of machine learning as a means to determine how accurately a learning algorithm is able to predict data that it was not trained on. In the leave-one-out method, all but one of the data points from a dataset are used to train the learning method, the model is then used to test the remaining data point and the error of its placement is noted. Willett assumed that a property value for each molecule, i, was unknown and the clusters were scanned to locate the one containing i. The predicted property value for i was then calculated by taking the mean of the property values for the other structures within the cluster. This was subsequently repeated for all molecules in the dataset. The correlation between the predicted and observed property values was then determined using the product-moment correlation

coefficient. The results of this study indicated that Ward's method performed the best overall, for a set number of clusters, however due to the time required for its implementation Ward's was deemed unsuitable for use with large datasets. Hence, Jarvis-Patrick clustering was considered to be a better compromise as it offered nearly as good results for property prediction yet had smaller time constraints attached.

In 1994, a study by Downs, Willett and Fisanick (1994) evaluated the use of three hierarchical clustering techniques along with standard Jarvis-Patrick clustering for property prediction. This study was based on a significantly larger dataset than those used in the first study by Willett, containing molecules described by 13 physicochemical properties. As in the earlier study, the predicted value of each property for a molecule was calculated as the mean of the property values of the other molecules in the same cluster. The results of the study showed that Ward's and the minimum-diameter method were particularly good at predicting properties. This study also highlighted that Jarvis-Patrick clustering performed the worst.

The results of the study by Downs, Willett and Fisanick (1994) are also in agreement with a subsequent study by Brown and Martin (1996). This study evaluated a variety of clustering methods alongside several molecular descriptors such as structural keys and hashed linear path fingerprints. The clustering methods were evaluated according to their ability to separate a set of biologically active and inactive molecules into different clusters. A finding of their study was that Ward's RNN clustering coupled with 2D descriptors was the most effective of a number of methods in the identification of a subset of bioactive compounds. Jarvis-Patrick clustering performed poorly, producing a far greater number of singleton clusters.

2.5.3 How Many Clusters to Select?

Many clustering methodologies, in particular hierarchical clustering and relocation methods, encounter a similar problem of how many clusters are required to best represent the dataset. If an example such as Ward's clustering method is considered, initially all the molecules in a dataset are found as singleton clusters and in each subsequent iterative step, two clusters are merged to produce a larger cluster. The clusters found prior to each iterative step provide the cluster level at which the method has reached i.e. at cluster level 1 all the molecules are singletons and at cluster level 2 the merging of the first two clusters takes place, etc. This continues until a final cluster level is reached in which the remaining two clusters are merged to form a single cluster that encompasses all the molecules in the dataset. This leaves the question of which cluster level best represents the molecules in the dataset.

In other disciplines many methods have been proposed to evaluate the *goodness* or *quality* of a particular clustering level, with the main body of work in this area coming from the psychology literature (Milligan, 1981; Milligan and Cooper, 1985). In 2000, Wild and Blankley (Wild and Blankley, 2000) compared nine methods on their effectiveness at selecting an ideal cluster level for Ward's clustering of chemical data using four different 2D descriptors. They first established the ideal set of clusters using their own chemical intuition such that molecules were clustered together if they believed they were chemically related. To assess the effectiveness of the cluster level selection methods, the clusterings produced were compared to the ideal clusters, using the Jaccard statistic that is identical to the Tanimoto coefficient.

Equation 5: *Jaccard Score formula.*

$$\text{Jaccard } (C_1 C_2) = \frac{a}{a + b + c}$$

Where:

$\text{Jaccard } (C_1 C_2)$ is the similarity between cluster method C_1 and cluster method C_2

a is the number of pairs of molecules that are clustered together in both $C_1$and C_2

b is the number of pairs of molecules that are clustered together in C_1but not in C_2

c is the number of pairs of molecules that are clustered together in C_2but not in C_1.

The measures of cluster level selection they evaluated included the Kelley (Kelley et al., 1996), C-index (Hubert and Levin, 1976), Point Biseral (Milligan, 1981) and Variance Ratio Criterion (Calinski and Harabasz, 1974) measures. They found that although no measure performed consistently well using all types of data; the Kelley measure performed consistently well with all fingerprint types and henceforth was the most appropriate for use when complexity, mean and worst case performance were considered.

The Kelley measure (Kelley et al., 1996) was initially described for use in selecting optimal clusters of protein NMR ensembles, and is implemented in BCI's OPTCLUS (DigitalChemistry, 2005) program. It uses the following equation:

Equation 6: *The Kelley measure.*

$$KELLEY_l = (n-2)\left(\frac{\bar{d}_{wl} - \min(\bar{d}_w)}{\max(\bar{d}_w) - \min(\bar{d}_w)}\right) + 1 + k_l$$

Where:

$\bar{d}_{wl}$ is the mean of distances between points in the same cluster at level l

$\max(\bar{d}_w)$ is the maximum distance value across all cluster levels

$\min(\bar{d}_w)$ is the minimum distance value across all cluster levels

k_l is a user defined value which is used to penalise cluster levels which contain a large number of singletons.

To identify the optimal clustering level using the Kelley measure requires the calculation of the Kelley score for each clustering level and subsequent identification of the level which the maximum Kelley score is associated with.

2.6 Summary

This chapter has introduced just some of the vast array of computational methods that are involved in the modern drug discovery process, placing an emphasis on the data mining techniques that are now routinely used within a variety of tasks. Data mining has played an integral role in allowing medicinal chemists to improve the design of combinatorial libraries and in the analysis of data generated via HTS technologies, with the use of traditional clustering algorithms becoming widespread. Despite their widespread use for a variety of applications, new algorithmic advances in clustering have been minimal within the field of Chemistry. The following chapters will aim at exploring the new advances in clustering from other scientific disciplines, placing a particular emphasis on those methods that cluster data based on the eigenpairs of a matrix. The scope for adaptation and implementation of these algorithms with chemical data will be considered and explored experimentally.

Chapter 3

Spectral Clustering

3.1 Introduction

A number of new clustering methodologies have appeared in the recent literature with promising results reported in many scientific fields. However, there has been little effort made towards researching their suitability for use with chemical data. Examples include: the application of spectral clustering to group molecular scaffolds (Brewer, 2007); the use of hypercliques to find hierarchical relationships within a distance matrix (Xiong et al., 2006); using sub-space clustering to identify areas of chemical spaces where different attributes are relevant for clustering (Parsons et al., 2004); and the implementation of multi-objective clustering to, for example, choose an appropriate number of clusters (Schuffenhauer et al., 2007).

This chapter presents an overview of spectral clustering algorithms: including a step-by-step introduction to the mathematical concepts which underpin spectral clustering algorithms, a

review of the well established algorithms implemented within other scientific disciplines and an in-depth discussion of Brewer's application of spectral clustering to chemical data (Brewer, 2007), which has been used as the basis for further studies in the field (Neres et al., 2009; Davenport et al., 2010; Heifetz et al., 2013).

3.2 An Introduction to Spectral Clustering

The term spectral clustering is used to describe any clustering algorithm that utilises the eigenvectors of a matrix as the basis for partitioning a dataset (Ng et al., 2002). The approach can vary in a number of ways, including the type of matrix that is formed from the dataset and the way in which the eigenvectors are used as the basis for the clustering (Weiss, 1999). Mathematically, the application of any spectral clustering method is relatively simple, providing the user has a basic knowledge of linear algebra and graph theory (von Luxburg, 2007).

The key step involved in the application of spectral clustering to any data is the recognition that the input matrix, produced from the dataset, is equal to a weighted adjacency matrix, W, of a graph of N nodes, where the N nodes relate directly to the data points that are being analysed. By recognising this key step and using the theory of graph spectra, it can be shown that the eigenvectors of the matrix (which can be a similarity, Laplacian or any other input matrix) constitute a set of weights, which can be used to cluster similar nodes (Sarkar and Boyer, 1998; Brewer, 2007).

This section of the chapter will build upon the initial discussion of graph theory given in ***Chapter 2***, before introducing the concept of generating matrix representations for similarity graphs, and finally providing an explanation of the basic mathematical operations that underpin spectral methods.

3.2.1 Graph Theory

Originally theorised by the mathematician Leonard Euler (1736), as a means of solving a mathematical puzzle, Graph Theory, GT, is a mathematical discipline that allows problems to be represented as abstract graphs, permitting the use of graph-theoretic algorithms to solve them (Biggs et al., 1986). In these terms, a graph is an abstract mathematical construct that is comprised of two key features: the vertex set, V, which contains the vertices (or nodes), and the edge set, E, which is used to represent the relationship between the vertices of the graph.

Thus, an undirected graph, G = (V, E), can be defined as containing a disjoint set of nodes, V, and edges, E, in which a pair of nodes can be connected by an edge and each edge in the edge set is a 2-element subset of V (Diestel, 2000). Additional information can be encoded into graphs by the weighting of edges and associating information with the nodes. Also it should be noted that in graph theory, two nodes are termed adjacent if they are connected by at least one edge and the number of adjacent nodes defines the degree of any vertex.

3.2.2 Similarity Graphs and Weighted Adjacency Matrices

The representation of molecules as graphs was discussed in ***Chapter 2***. Here this notion is extended so that an entire dataset is represented by what is known as a similarity graph. A similarity graph can be defined such that G = (V, E) is an undirected graph, where each vertex is associated with a molecule and each edge carries a non-negative weight equal to the similarity value between the two molecules represented by the vertices connected by the edge. Several different types of input matrices can be used to represent a similarity graph, in a similar manner to how a molecule can be represented by numerous different molecular descriptors. Nevertheless, the inherent goal of clustering is always to divide a dataset into several groups, so that the data points in each group are similar and dissimilar to data points in different groups. Clustering can now be reformulated in terms of the similarity graph, where the intention is to segment the graph so that data points within a cluster have high edge weights and data points in different clusters have low weighted edges (von Luxburg, 2007).

3.2.2.1 Matrix Representations

One of the most interesting properties of a graph is that it can be described mathematically through the use of matrices, and conversely, a graph can be formed from a matrix. This property allows a graph to be analysed through a branch of mathematics called linear algebra (see ***Section 3.2.1***). Once an input matrix A has been selected to represent a similarity graph, it can be segmented using linear algebra by recognising that this matrix A can be used as a weighted adjacency matrix in the calculation of eigenpairs. The term weighted adjacency matrix of a graph can be defined as a matrix W with weights (w_{ij}) where I and j = 1, 2, ..., n, where n is the number of data points and each weight is assigned to an edge between two data points. It should be noted that any pair of nodes that share no relationship are not joined by an edge (alternatively this can also be considered as the nodes sharing an edge with a weight equal to zero). A description of some of the most commonly used input matrices is now provided.

The most basic matrix representation of a similarity graph is a similarity matrix, or proximity matrix. Each element of a similarity/proximity matrix gives the similarity/proximity score between two objects according to a predefined similarity or distance measure, for example, the Tanimoto coefficient.

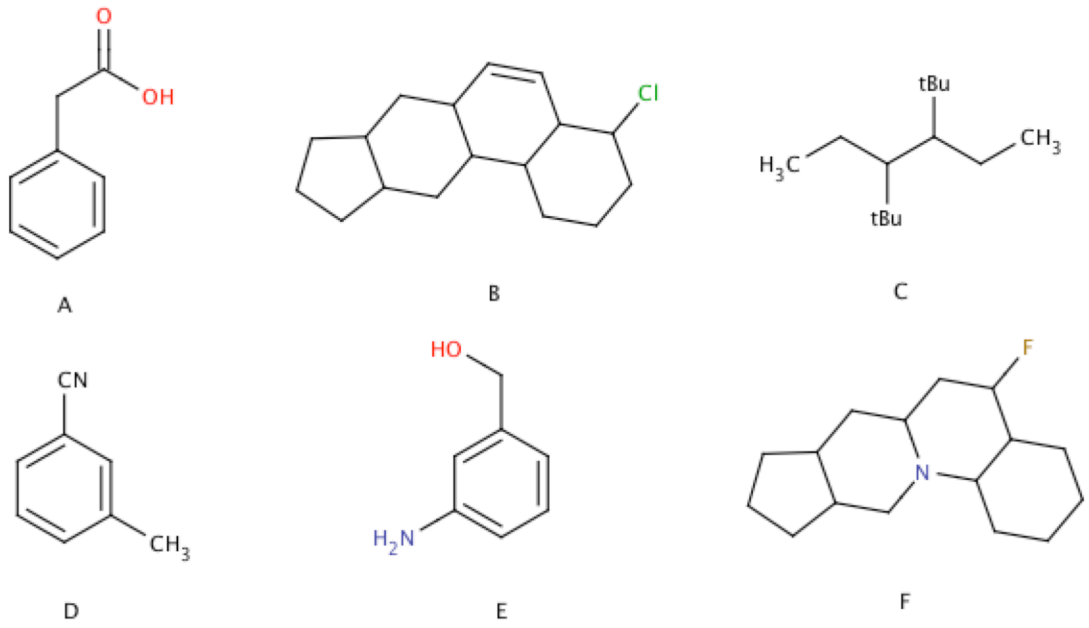


Figure 3-1: *Depicts an idealised dataset containing six molecules, A - F. In this dataset the molecules with a similarity score above a threshold value of 0.8 are given a similarity score of 1, with molecules with a similarity score below the threshold are given the value of 0.*

Similarity matrices are easy to construct and are routinely used in many data mining techniques. By using the contrived dataset shown in ***Figure 3.1*** a similarity matrix can be formed where, for the sake of simplicity, a pair of molecules which have a similarity value above a threshold value of 0.8 are given a similarity score of 1, with molecules with a similarity score below the threshold given the value of 0.

Similarity matrix, A =

1	0	0	1	1	0
0	1	0	0	0	1
0	0	1	0	0	0
1	0	0	1	1	0
1	0	0	1	1	0
0	1	0	0	0	1

In a similarity matrix, the self-similarity of a molecule is equal to unity, which is shown in the leading diagonal of the similarity matrix shown above. Affinity matrices are closely related to similarity and proximity matrices, and differ in that the self-similarity value of any object is given a value of zero.

For a weighted graph, G = (V, E), the degree of any vertex is defined by the number of its connections with a weight > 0. By identifying the degree of each node, a degree matrix, D, can be formed. A degree matrix is analogous to the identity matrix with the exception of providing the degrees, d_1, ..., d_n, across the leading diagonal instead of values of unity (von Luxburg, 2007). In graph theory, the Laplacian matrix is routinely used as a matrix representation of a graph and is commonly used in spectral graph theory when finding graphical properties, e.g., the eigenvectors. When the graph G = (V, E) is an undirected, unweighted graph without any graph loops or multiple edges from one node to another, the Laplacian matrix, L, of the graph G is given by the equation (Chung, 1997):

Equation 7: *Laplacian Matrix Formula.*

$$L = D - P$$

Where:

L is the Laplacian matrix

P is the proximity matrix

D is the degree matrix.

A Laplacian matrix can also be defined by the equation:

Equation 8: *Alternate equation to define Laplacian matrix*

$$l_{i,j} = \begin{cases} d(v_i) & \text{if } i = j. \\ -1 & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j. \\ 0 & \text{otherwise.} \end{cases}$$

Where $d(v_i)$ denotes the degree of the vertex, v, and both i and j are nodes in the graph G. An example of the Laplacian matrix for the dataset shown in ***Figure 3.1***, is given below:

$$L = \begin{matrix} 2 & 0 & 0 & -1 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 2 & -1 & 0 \\ -1 & 0 & 0 & -1 & 2 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 \end{matrix}$$

When a Laplacian matrix is being formed from a highly connected dataset, the values across the diagonal of the Laplacian matrix may become sufficiently large that it may be necessary to use a normalised Laplacian matrix. The normalised Laplacian matrix is defined as:

Equation 9: *Normalised Laplacian matrix formula.*

$$l_{i,j} = \begin{cases} 1 & \text{if } i = j. \\ -\frac{1}{\sqrt{d(v_i)\, d(v_j)}} & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j. \\ 0 & \text{otherwise.} \end{cases}$$

The normalised Laplacian matrix sets the values down the diagonal to 1 and normalises the magnitudes of the other entries of the matrix by dividing - 1 by the square root of the degrees of two nodes that the value relates too. An example of a normalised Laplacian matrix for the dataset shown in ***Figure 3.1*** is given below.

$$L_{norm} = \begin{matrix} 1 & 0 & 0 & -0.5 & -0.5 & 0 \\ 0 & 1 & 0 & 0 & 0 & -0.707 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ -0.5 & 0 & 0 & 1 & -0.5 & 0 \\ -0.5 & 0 & 0 & -0.5 & 1 & 0 \\ 0 & -0.707 & 0 & 0 & 0 & 1 \end{matrix}$$

Each of the matrices discussed above can be used as the weighted adjacency matrix of a graph in the approaches to spectral clustering discussed within this chapter. However, the ultimate decision of which is the best type of matrix to be constructed is dictated by the choice of spectral clustering algorithm.

3.2.3 Linear Algebra and Spectral Graph Theory

Linear algebra is the name given to a branch of mathematics in which vectors are studied, or more specifically vector spaces and the matrices that act upon them. In elementary algebra

the application of a function to an object is denoted by the equation f(x). The objects, x, used within elementary algebra are constants and variables that usually take the form of real numbers, and functions are often simple procedures such as multiplications or additions. However, in linear algebra the application of a function to an object is written as M(v), where M is a linear algebra function, represented by a matrix, and v is a vector.

A vector is a geometric object that is often referred to as the movement required to carry a point A to point B in multidimensional space and is one of the two major objects used within the field of linear algebra. Mathematically speaking, a vector is defined as a member of *vector space*, where the term vector space refers to a multidimensional construction occupied by a collection of vectors, and is comprised of two components, a magnitude and a direction. The second type of object used in linear algebra is a scalar, R, which is typically given in the form of a real or complex number. Each scalar is associated with an individual vector and is used to scale the magnitude of the vector and/or switch its direction.

In linear algebra the most important group of functions is called linear transformations. A linear transformation acts upon a vector, typically altering both the associated magnitude and direction of the vector. There are a number of linear transformations including rotations, reflections, stretches, compressions and shears, all of which are represented by matrices (Strang, 2003).

3.2.3.1 Eigenvalues, Eigenvectors and Eigendecompositions

In general, a matrix acts on a vector by changing both its magnitude and direction; however this is not always the case. In some cases, a matrix can act on a vector and alter its magnitude only, leaving the direction of the vector unchanged or in some cases inverted. Vectors whose directions are left unaltered or inverted by a linear algebra function are known as the eigenvectors of the matrix.

By definition the magnitude of a vector is always multiplied by a factor during a linear operation, the scalars that quantify the magnitude of these factors are called eigenvalues, λ. Every eigenvalue obtained from a matrix is associated with an individual eigenvector. A positive eigenvalue indicates that there has been no change in the orientation of the vector and conversely a negative eigenvalue demonstrates that the eigenvector has been inverted. Eigenvectors can be more formally defined using the eigenvalue equation, which states:

If A is a linear transformation, a non-null vector, x, is an eigenvector of A if there is a scalar λ such that:

Equation 10: *Eigenvector equation.*

$$Ax = \lambda x$$

Where the scalar λ, is more formally known as the eigenvalue associated with the eigenvector x (Blyth and Robertson, 2002; Strang, 2003).

The eigenvalues and their associated eigenvectors, otherwise known as the eigenpairs, can be identified using an eigendecomposition algorithm, which is the term given to a procedure for identifying eigenpairs from an input matrix. When describing eigendecomposition methods, it should be noted that this term encompasses a variety of methods ranging from those based on a full matrix diagonalisation (that is discussed further in ***Section 3.5*** and in ***Appendix A***, which contains a worked example of the application of a full matrix diagonalisation procedure to the dataset given in ***Figure 3.1***) to those that are often referred to as eigensolvers, which approximate the eigenpairs from an incomplete diagonalisation of a matrix.

Each of the eigendecomposition algorithms used within this thesis are outlined within their respective experimental chapters where the mathematical concepts that underpin each method, how they are implemented and their respective advantages and disadvantages are explored. Alternatively, for further information on eigendecomposition algorithms the reader is directed to textbooks by Strang (2003), Parlett (1998) and Golub (1996).

Now that the basic mathematical concepts that underpin the implementation of spectral clustering have been outlined, the next section will aim to relate the abstract concepts of eigenpairs to clustering data. To achieve this goal an informal discussion of the role of eigenvalues and eigenvectors within the context of spectral clustering will be undertaken, focussing on how eigenpairs can be visualised in the most basic spectral clustering algorithms.

3.3 The Role of Eigenpairs in Spectral Clustering

Typically in spectral clustering, an eigendecomposition algorithm is used to decompose an input matrix in order to produce a matrix of eigenvectors, C, in which each column is an eigenvector and each row is an object, or more correctly, a list of contributions that an object makes to each eigenvector. Furthermore, each of these eigenvectors (and their corresponding eigenvalues) can be associated with an eigencluster, where an eigencluster is defined as any

eigenvector that is used by a spectral clustering algorithm in segmenting a dataset. Each component in the eigenvector provides a measure of the contribution that an object makes to that particular eigencluster. The decision of which and how many of the eigenvectors act as eigenclusters is dictated by the choice of spectral clustering algorithm. An example of how the eigenvalues, eigenvectors and possible eigenclusters of a similarity matrix can be found is provided in ***Appendix A***.

This leaves the question of how and why an eigenvector allows one to gain any knowledge of the relationships between chemical compounds? An informal example is used to aid in the comprehension of this abstract concept. Let us consider a theoretical chemical space of N dimensions that contains each of the molecules within a dataset. An eigenvector can be used to describe a movement through this space, between two points. If one was to look down the vector they would gain a view of each compound from the perspective of the eigenvector, with molecules that provide the largest eigenvector components being located closest to the vector and molecules that make the smallest contribution being located the furthest away. Looking down each individual eigenvector allows the data to be viewed from different perspectives and the molecules that are closest to a vector to be identified.

Moreover, the eigenvalue associated with an eigencluster provides a means of quantifying the cluster's cohesiveness. Traditionally, the term cluster cohesiveness is defined as the degree to which the molecules in a cluster are similar to one another and is usually measured as the mean pairwise similarity of the compounds. However, in spectral clustering cluster cohesiveness defines the number of connections between molecules and their respective weights such that a set of identical data points would produce an eigenvalue of N-1 reflecting the presence of a maximum number of connection between the data points that were each weighted with the maximum value of one. Although this definition is easily visualised for similarity graphs where N is low, when dealing with large datasets like chemical activity classes, visualisation of this concept is made very difficult. Thus, an alternative way of considering this abstract concept is to visualise an eigenvalue, λ_n, as providing a measure for cluster cohesiveness by indicating the location where the associated eigenvector passes through the N dimensional chemical space, and the positioning of the compounds in relation to this vector. To enhance this description let us consider the example contained in ***Figure 3.2*** in which each black point represents a molecule in chemical space and the red point indicates a location where the eigenvector passes through the set of molecules. In cluster A, the eigenvector passes close to the centre of the cluster. In this case a large eigenvalue would be

produced, as the point where the vector passes through the molecules leads to a minimisation in the mean distance between the vector and the molecules. Conversely, in cluster B the eigenvector passes through the edge of the cluster, producing a cluster with a significantly lower eigenvalue than was produced for cluster A. This is due to the increase in the mean distance between the vector and the majority of the molecules. Therefore, it can be concluded that a cluster associated with a small eigenvalue indicates that the eigenvector travels through the cluster in a position that is far away from the bulk of the data points and that for clusters with large eigenvalues, the eigenvector traverses the cluster in a position that minimises the distances between eigenvector and the data points.

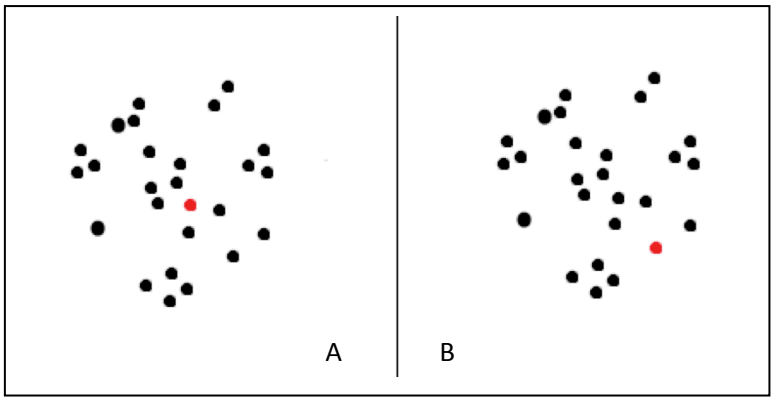


Figure 3-2: *A simple diagram showing the distribution of a theoretical set of molecules (shown by the black data points) within two clusters in relation to an eigenvector indicated by the red datapoint.*

3.4 Review of Spectral Clustering Algorithms

Spectral clustering methods based upon the eigenvectors of a matrix have found prominent use in many scientific fields, such as computer vision and geology. These approaches to clustering are extremely attractive as they are based upon simple eigendecomposition algorithms that are well understood.

Although each spectral clustering algorithm differs according to the choice of initial input matrix and how the eigenpairs are used in the segmentation of a dataset; all spectral clustering can be broken down into three generic steps:

- An initial input matrix is formed from the data.
- The eigenpairs of the input matrix are identified using an eigendecomposition algorithm.
- The dataset is partitioned using the retrieved eigenpairs as a basis for the clusters.

A review of several spectral clustering algorithms applied within a variety of different subject areas follows.

3.4.1 Scott & Longuet-Higgins (1995)

Whilst studying various problems in computer vision, Scott and Longuet-Higgins hit upon a novel clustering algorithm related to techniques widely used in molecular physics (Scott and Longuet-Higgins, 1990), which they entitled relocation spectral clustering. The goal of this algorithm is to partition a dataset into a user-defined number of clusters, based upon the truncated eigenvectors of a square matrix and a relocation step.

Initially a pairwise proximity matrix, *P*, is formed using the Euclidean distances between objects and the scaling equation:

Equation 11: *Scaling equation used to generate an input matrix for the relocation spectral clustering algorithm.*

$$P = e^{\left(\frac{-d^2}{2\sigma^2}\right)}$$

Where *d* is the Euclidean distance between the objects and *σ* is a user-defined scaling constant.

The next step is to elucidate the eigenvalues, λ, and eigenvectors of *P*, by harnessing an eigendecomposition algorithm. An eigenvector matrix, *C*, is formed by stacking the eigenvectors as columns in a matrix, moving from the first eigenvector to the last (herein the term the first eigenvector of a matrix is used to describe the eigenvector related to the largest eigenvalue).

Next, the number of columns of the matrix *C* is truncated using a user-defined parameter M that selects the number of eigenvectors, whose components will be used to provide the basis for clustering the data. Therefore, the value of M gives the number of clusters that are produced. The components of the eigenvectors outside the top M eigenvectors of the eigenvalue-ordered sequence are set to zero and each row of the truncated matrix *C* is relabelled as a vector T_i (***see Figure 3.3***). Each truncated vector, T_i, is then normalised to unity, producing a vector R_i.

λ =	**3**	**2**	**1**	**0**	**0**	**0**
T_1 =	0.5774	0	0	0	0	0
T_2 =	0	0.7071	0	0	0	0
T_3 =	0	0	-1	0	0	0
T_4 =	0.5774	0	0	0	0	0
T_5 =	0.5774	0	0	0	0	0
T_6 =	0	0.7071	0	0	0	0

Figure 3-3: *An example of the vectors T_1 to T_6 when the value of M is set to 3.*

Rather than directly using the eigenvectors to cluster the dataset, they are transformed into what the authors refer to as a bond order matrix. Each element of the bond order matrix, B, contains the scalar product of two vectors R_i and R_j that is given by ***Equation 12***.

Equation 12: *Scalar product equation used to form Bond Order Matrix.*

$$B_{ij} = R_i R_j \cos\theta$$

Where B_{ij} is the scalar product of two vectors R_i and R_j.

The cosine of the angle between two vectors R_i and R_j is calculated using ***Equation 12***.

Equation 13: *Used to calculate the cosine of the angle between two vectors.*

$$Q_{ij} = \frac{\sum R_i R_j}{\sqrt{\sum R_i^2}\sqrt{\sum R_j^2}}$$

Where R_i and R_j are vectors and Q_{ij} is the cosine of the angle between vectors.

It should be noted that the *Q* values are typically stored in an association matrix, where each element provides the *Q* value between two vectors. Finally, each object is classified into one of the M clusters based upon their highest elemental contribution across a row in the bond order matrix.

This spectral clustering algorithm has shown promising results in the grouping of complex feature space and has a number of advantages, such as the ability for the user to define the number of clusters desired. However, a major drawback of the algorithm is the associated

computational cost of solving the eigenvalue problem, which is in the order of N^3, where N is the number of objects in a dataset. A possible optimisation of this algorithm, which could offer a significant decrease in the computation costs, would be to calculate only the first M eigenvectors of the matrix, using a different eigendecomposition algorithm that scales more favourably with N.

Relocation algorithm key features:

- Uses a proximity matrix as input matrix.
- Defines the number of eigenvectors used as the basis for spectral clustering with a user-defined parameter M.
- M non-overlapping clusters formed.
- Forms intermediate renormalised vectors $R_{1....N}$.
- Calculates a bond order matrix using the *R* vectors.
- Objects assigned to clusters based upon their modified eigenvector components in bond order matrix.

3.4.2 Sarkar and Boyer (1998)

Sarkar and Boyer (1998) developed their approach to spectral clustering whilst looking for an effective way to provide a quantitative measure of change, based on features. This algorithm is different to many other spectral clustering algorithms as it produces an overlapping clustering of a dataset, by using the information gleaned from multiple eigenvectors.

The Sarkar and Boyer method of spectral clustering uses a multi-step algorithm to classify objects:

- An affinity matrix, A, is formed from the dataset, using a user-defined measure of affinity.
- An eigendecomposition algorithm is applied to matrix A, to find the eigenvectors of the matrix and their associated eigenvalues.
- Each eigenpair is then subjected to the 95% positive eigenvector rule. This rules states that an eigenvector can only be considered a meaningful eigencluster, if:

  - The associated $\lambda > 0$.

  - 95% of the eigenvector's magnitude is contributed by either the squared values of the negative or positive components only.

- If an eigenvector satisfies this rule it is deemed to be a viable eigencluster. Whereas, eigenvectors that fail to fulfil the requirements of this rule are disregarded.

The authors argue that whether an eigenvector obeys the 95% positive eigenvector rule is of vital importance to this algorithm for two reasons. Firstly, eigenvalues with negative magnitudes should be discounted, as the sum of the total weighted connections within a graph cannot be less than zero, so the information contained in the eigenvectors associated with these eigenvalues is redundant to the partitioning of the dataset. Secondly, a negative component within an eigenvector implies that a node is negatively associated with a cluster, which is not plausible. The term negative component is used here to infer a component that has the opposite sign to the majority of other components, for example, a component with the value 0.01 would be considered a negative component if the rest of the eigenvector was comprised of values such as -0.1, -0.7 and -.0.049, as both the eigenvectors **x** and **–x** can be associated with an eigenvalue. In practice Sarkar and Boyer relax the requirements for all nodes to make a positive contribution to the 95% level to ensure that valuable information about the natural portioning of the data is not lost if a large eigenvector has a minimal number of negatively contributing nodes.

The final spectral clustering algorithm produces overlapping clusters, in which an object can be a member of multiple clusters, as the contribution an object makes to all positive eigenclusters is considered during the evaluation. As N eigenvectors are evaluated during the algorithm, where N is the number of objects, the time costs and computational complexity is fairly high. However, the results obtained are favourable in the case of detecting feature change, i.e., the algorithm was able to group nodes corresponding to locations where feature change occurred across several images of an area that was being built upon.

Sarkar and Boyer algorithm key features:

- Uses an affinity matrix as the input matrix.
- The number of eigenvectors used as eigenclusters is selected based on the 95% positive eigenvector rule.
- Produces up to N overlapping clusters.

3.4.3 Perona and Freeman (1998)

In 1998, Perona and Freeman (1998) presented a spectral clustering algorithm based on placing a threshold on the first eigenvector of an affinity matrix (Weiss, 1999). This method is

called the affinity factorisation algorithm and can be used to group a set of objects into two clusters by applying a threshold to the approximation of the pairwise affinity between objects of the first eigenvector.

The basic steps of the affinity factorisation method are as follows:

- An affinity matrix, *A*, is constructed by comparing pairs of objects. Two points are considered similar if they have an affinity score below a threshold distance, d_0. The affinity between two molecules is calculated according to the affinity equation:

Equation 14: *Affinity equation for the Perona and Freeman spectral clustering implementation.*

$$A_{ij} = e^{-d^2(i,j)}$$

Where d is the Euclidean distance and A_{ij} is the affinity between two objects *i* and *j*,

- The first eigenvector, C_i, of the matrix *A* is calculated using an eigendecomposition algorithm.
- Objects are clustered into one of two clusters based upon their elemental contribution, C_i, to the eigenvector C_i. If an object *i* has a $C_i > 0$, it is placed into the foreground cluster, whereas an object *j* whose corresponding $C_i \leq 0$ is placed into the background cluster.

Perona and Freeman were able to prove that for affinity matrices, the first eigenvector has non-zero components corresponding to the objects in the dominant cluster (Weiss, 1999). This algorithm can also be implemented efficiently as only the first eigenvector is required to cluster a dataset. A major disadvantage of this algorithm is that objects are only classified into one of two clusters.

Affinity factorisation algorithm key features:

- Uses an affinity matrix as the input matrix, where the affinity is defined via ***Equation 14***.
- Only uses the first eigenvector as the basis for segmentation.
- Produces two non-overlapping clusters called the foreground and background clusters.

3.4.4 Shi and Malik (2000)

Shi and Malik (2000) proposed an approach to spectral clustering that is closely related to the works of Perona and Freeman (1998), called the normalised cuts algorithm, in reference to a global criterion used in the segmentation of a graph. In graph theory, a graph G = (V, E) can be partitioned into two disjoint groups, A and B, by removing the edges which connect the two parts of the graph. A cut is the mathematical term that describes the degree of dissimilarity between these two segments, and the cost of the cut can be computed as the sum of edge weights that have been removed from the graph. The term normalised cut describes a measure for the cost of a cut as a fraction of the total edge weight of the graph.

The normalised cuts algorithm differs from other approaches to spectral clustering as the partitioning of the dataset is based upon the use of the generalised eigenvectors of a matrix. Generalised eigenvectors are used to ensure that the algorithm remains applicable even in the case where an input is not fully diagonalisable. The generalised eigenvectors of a matrix are calculated as part of a multi-stage methodology. The first step is to form a weighted adjacency matrix, *W*, from the data, where an edge between any two nodes is weighted by their similarity. The degree matrix, *D*, of *W* is generated, such that:

Equation 15: *Equation for generating a degree matrix from a weighted adjacency matrix.*

$$D(i,i) = \sum_{j} W\ (i,j)$$

Where *i* and *j* are graph nodes/objects.

The next step is to identify the generalised eigenvectors of the matrix *W*. A generalised eigenvector must satisfy the equation:

Equation 16: *Generalised eigenvector formula.*

$$(D - W)y_i = \lambda_i D y_i$$

Where:

D = the degree matrix

W = the weighted adjacency matrix

y_i = a generalised eigenvector

λ_i = the associated eigenvalue.

Using mathematical operations, the generalised eigensystem can be reformulated into an eigenvector problem, such that:

Equation 17: *Formula for a generalised eigensystem.*

$$D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}x_i = \lambda_i x_i$$

This equation can be readily solved using an eigendecomposition algorithm and linear algebra.

The dataset is then bi-partitioned based on the eigenvector with the second smallest eigenvalue, which the authors describe as producing the optimal cut for a dataset based on their normalised cuts measure. Finally further segmentations can be made by recursively repartitioning the dataset.

This algorithm has provided very encouraging results in the field of image segmentation, where a normalised cut provides an unbiased measure of the disassociation between graph subgroups. Like most spectral clustering algorithms, the implementation of this method is hindered by the $O(N^3)$ operations required to apply the eigendecomposition algorithm. However, Shi and Malik (2000) do suggest using the Lanczos eigensolver method (Lanczos, 1950), which can reduce the number of operations required to find the eigenvectors of a matrix from $O(N^3)$ to $O(N)$. The use of the Lanczos algorithm as an optimisation method is described in ***Chapter 6***.

Normalised cuts algorithm key features:

- Uses a similarity/proximity matrix as the input matrix.
- Bases segmentation of data on the second smallest generalised eigenvector.
- Produces two non-overlapping clusters, but can be recursively applied to the dataset.

3.4.5 Ng, Jordan and Weiss (2002)

Ng, Jordan and Weiss (Ng et al., 2002) describes a hybrid approach to spectral clustering based upon the use of the k largest eigenvectors of a Laplacian matrix. This novel approach to spectral clustering utilises a k-means clustering algorithm to cluster the rows of a matrix to produce a clustering of the dataset. The outline of this algorithm is described below.

For a set of N objects, denoted as $H = (h_1, \ldots, h_N)$, an affinity matrix, A, is formed, where the affinity between two objects is given by the equation:

Equation 18: *Affinity measure for the Ng, Jordan and Weiss spectral clustering procedure.*

$$A_{ij} = e^{\left(-\frac{\|h_i - h_j\|^2}{2\sigma^2}\right)}$$

Where:

A_{ij} = the affinity between two objects h_i and h_j

σ = a scaling parameter.

The scaling parameter is a user-defined constant that controls how rapidly the affinity score decreases as the distance between a pair of objects is increased. Subsequently, the degree matrix, D, is defined and a Laplacian matrix, L, constructed, such that:

Equation 19: *General Formula to produce a Laplacian matrix from an affinity matrix.*

$$L = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$$

An eigendecomposition algorithm is applied to L, and the k largest orthogonal eigenvectors as given by the values of λ were selected, where k is a user-defined parameter. Furthermore in the case of repeated eigenvalues only orthogonal eigenvectors are used to ensure that the selected eigenvectors are unique. Two eigenvectors are orthogonal if their dot product is equal to zero.

The k largest eigenvectors are then stacked into a matrix, X, which is N x k in size. Each row of X is then normalised to unity to form matrix Y. Treating each row of the matrix Y as an individual data point, a clustering algorithm is applied to the matrix Y to cluster the data points into k clusters. For example, the Euclidean distances between the data points are then calculated and k seeds are selected at random, a k-means algorithm is then applied to minimise the Euclidean distance between the seed object and the other objects. The objects are relocated until the best clustering of the dataset is found.

Like all spectral methods this algorithm requires a significant time and computational effort due to the number of operations required to carry out the eigendecomposition step. Despite these constraints, this approach has shown promising results when applied to segmentation

problems in a number of fields, including the analysis of protein sequences (Paccanaro et al., 2006; Nepusz et al., 2010).

Ng, Jordan and Weiss spectral clustering algorithm key features:

- Uses a Laplacian matrix as the input matrix.
- The k largest eigenvectors are used to form a new matrix, which is renormalised to form matrix Y.
- A k-means clustering algorithm is applied to matrix Y to cluster the dataset.

3.5 The Application of Spectral Clustering to Chemical Data

In 2007, Brewer and his colleagues built upon the previous works of Sarkar and Boyer (1998) to produce a spectral clustering method for classifying chemical data, aimed at classifying molecules based on their molecular scaffolds (Brewer, 2007). This spectral clustering method generated overlapping clusters from the eigenvectors of a similarity matrix, where the term overlapping refers to a partitioning of the data in which each molecule may contribute to more than one cluster by a differing degree.

The starting point for the algorithm was to generate a similarity matrix using a similarity measure and molecular fingerprints. The next step was to decompose the similarity matrix using a diagonalisation procedure. The term diagonalisation refers to the application of an eigendecomposition algorithm that elucidates the eigenpairs from an input matrix based upon a full matrix diagonalisation procedure or FMD. This involves finding the characteristic polynomial of the matrix, before solving to identify the eigenvalues. These eigenvalues are subsequently used to calculate the associated eigenvectors. Eigenvectors retrieved by the eigendecomposition algorithm are pre-normalised, such that each eigenvector obeys the normalisation function:

Equation 20: *The normalisation function that all eigenvectors generated using eigendecomposition algorithms obey.*

$$1 = \sum_{j=1}^{N} c_{ij}^2$$

Where:

c_{ij} = individual contribution from each element *j* to eigenvector *i*.

N = number of contributing elements in vector.

The eigenvectors are stacked side by side into a square matrix of eigenvectors, *C*, where each column in this matrix is an eigenvector. Each column of C represents a possible eigencluster, in which the magnitudes of the eigenvector elements, c_{ij}, quantify the degree to which features of molecule *j* contribute to cluster *i*. Thus, each row of the eigenvector matrix is representative of a single molecule's contribution to each of the possible eigenclusters. A FMD is applied using ***Equation 21***.

Equation 21: *Full matrix diagonalisation formula.*

$$S = C^T \lambda_i C$$

Where λ_i is the diagonal matrix of eigenvalue, *C* is the matrix of eigenvectors and C^T is the transpose of matrix *C* (Press et al., 1992; Blyth and Robertson, 2002).

It should be noted that a FMD algorithm accurately identifies all of the N eigenpairs from an input matrix, and is typically implemented in a procedure that requires $O(N^3)$ operations to complete.

Brewer surmises that the cohesiveness of a cluster is denoted by the magnitude of the eigenvalue associated with the eigenvector, λ_i where $i = 1, \ldots, N$, and that the completeness of cluster *i*, can be determined by the extent to which the normalisation function is satisfied when summed over the individual contributions of the subset of molecules that contribute to it (Brewer, 2007), i.e.

Equation 22: *Equation for calculating the completeness of a cluster in Brewer's spectral clustering implementation.*

$$\frac{\sum c_{ij}^2 \text{ for subset}}{\sum_{j=1}^{N} c_{ij}^2} = \text{Cluster Completeness}$$

In this definition of cluster completeness, the subset of molecules for any cluster is given by the largest components of the eigenvector.

Preliminary studies by Brewer on similarity matrices produced from datasets comprised of molecules represented by two-dimensional Unity fingerprints, indicated that all molecules

from the dataset contributed to each of the clusters by a comparable amount, despite there being differences in the molecular scaffolds of the compounds. This issue can be attributed to the number of bits that each molecule has in common with the other molecules in the dataset leading a narrow range of similarity values being produced. Thus a filtered similarity matrix, S^F, was employed to produce modified eigenpairs to assist in discriminating between molecules with different molecular scaffolds. A filtered similarity matrix is a representation of the original similarity matrix with a decreased level of background similarity. A stronger emphasis is placed on pairs of similar molecules and the emphasis placed on pairs of dissimilar molecules is decreased.

The first method used to produce a filtered similarity matrix was the introduction of a discretised representation of S, where an exclusion similarity function S_{cut} was applied so that $S^F_{ij} = 1$ if $S_{ij} \geq S_{cut}$ and $S^F_{ij} = 0$ if $S_{ij} < S_{cut}$. Although this discretised function did lead to some improvement and was simple to implement, the improvement in the clustering was not sufficient to justify the loss of information that this function led to. Therefore, a continuous filtering function was introduced in place of the discretised function, which is given in ***Equation 23***.

Equation 23: *Gaussian filtering function with a tuneable parameter γ.*

$$S_{ij}^F = e^{-\gamma(S_{ij}-1)^2}$$

This continuous filtering function, is a Gaussian type function that was applied to the elements of similarity matrix, *S*, in order to remove background similarity whilst preserving the information contained within the similarity matrix, *S*. Brewer highlights that in practice the use of the tuneable parameter γ provided a convenient and intuitive way to modify the cluster cohesiveness and cluster contributions of individual molecules, by adjusting the eigenvector and eigenvalues of the filtered similarity matrix S^F.

3.5.1 The Application of Brewer's Spectral Clustering Method

This spectral clustering algorithm was applied by Brewer to a dataset of 125 cyclooxygenase-2 (COX-2) inhibitors taken from a study by Stahl, Laval et al. (2000) downloaded from the cheminformatics.org website (Cheminformatics.org, 2010). The application of the spectral clustering method was as follows:

1. Standard Unity 992 bit fingerprints were generated using the SYBYL software (Tripos, 2007) for the COX-2 inhibitor set (Laval et al., 2000).
2. The similarities between the compounds were quantified using the Tanimoto coefficient and the scores added to a similarity matrix, S.
3. The matrix S was filtered using the Gaussian filtering function shown in ***Equation 23*** to produce a filtered similarity matrix, S^F.
4. The matrix S^F was diagonalised to find the eigenpairs.
5. The cluster contributions were read into the MOE software and the clusters revealed by sorting the database by decreasing cluster contributions.

3.5.2 The Results of Brewer's Study

In his investigation, Brewer initially studied the effect that varying γ has on the cluster contributions of eight molecules (shown in ***Figure 3.4***) that were clustered together using his spectral clustering approach. For example, he found that at lower values of γ the contributions that molecules 1a – 1g made to the cluster associated with the largest eigenvalue were all of similar magnitude. Also in this example molecule 1h provides a contribution that Brewer deemed notable even though it was 2-3 orders of magnitude smaller. As the value of γ is increased, the contributions of 1c-h decrease, with compounds 1a and 1b emerging as the dominant molecules in the cluster. This shows that varying the value of γ in the Gaussian filtering function provides an intuitive method for emphasising the contribution of the dominant molecules to their respective clusters. The contributions of molecules 1a – 1h for several values of γ are shown in ***Table 3.1***.

	γ					
	12.5	25	50	100	200	400
λ_1 =	6.367	5.586	4.752	3.806	2.917	2.272
1a	0.149	0.181	0.208	0.243	0.286	0.348
1b	0.149	0.181	0.208	0.243	0.286	0.348
1c	0.132	0.144	0.142	0.133	0.115	0.075
1d	0.131	0.141	0.139	0.134	0.119	0.097
1e	0.131	0.141	0.139	0.134	0.119	0.097
1f	0.127	0.131	0.118	0.098	0.073	0.034
1g	0.097	0.082	0.046	0.016	0.002	4×10^{-5}
1h	0.004	6×10^{-5}	3×10^{-8}	0	0	0

Table 3-1: *Shows the effect of varying γ has on the size of λ and the squared eigenvector components of 8 molecules, 1a - 1h. Table adapted from (Brewer, 2007).*

Brewer determined that the γ value that produced the best set of clusters was 25. So using this value of γ, Brewer highlighted the five most cohesive clusters obtained from the dataset, i.e., those with the five largest eigenvalues. The cluster for the first eigenvalue, λ_1, contained eight molecules, of which the largest contributing molecules are based upon a 5,6-diphenyllimidazol[2,1-b]thiazole core. These molecules and their relative contributions are provided in ***Figure 3.4***. Viewing the molecules it is apparent that molecule 1h is based on a different ring scaffold than the other molecules in the cluster. This is reflected in significantly smaller contribution molecule 1h makes to the eigencluster compared to molecules 1a – 1g. This result shows the ability of the spectral clustering algorithm to differentiate between molecules with differing scaffolds, showing promise for future applications.

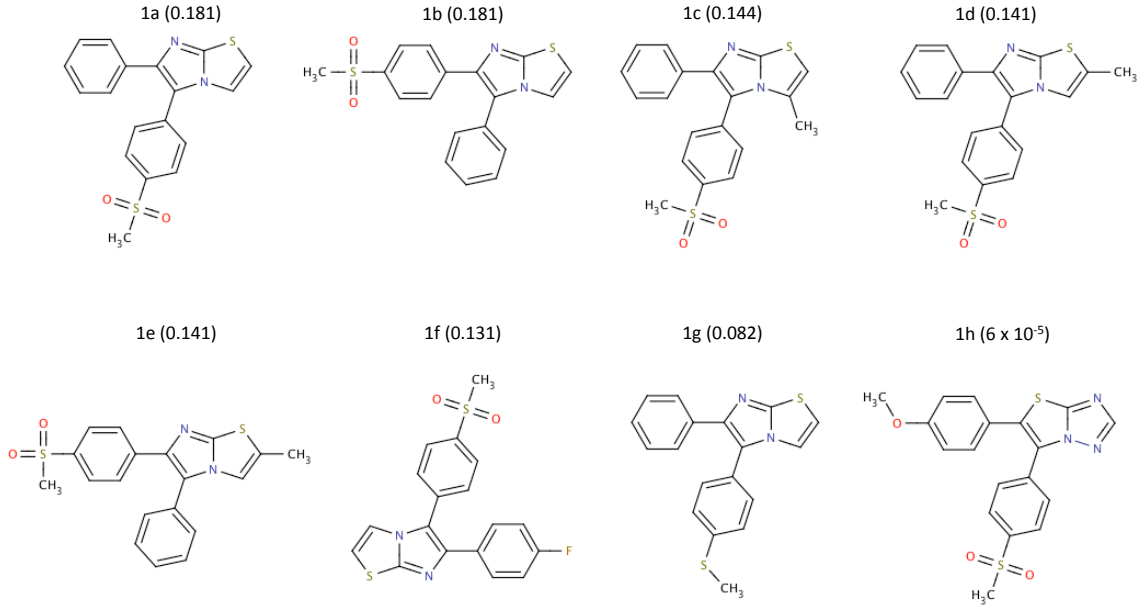


Figure 3-4: *Diagram shows the structures and their squared eigenvector components for cluster 1 when λ_1 =5.586 and γ = 25. Figure adapted from (Brewer, 2007).*

The cluster represented by the second largest eigenvalue contained seven molecules and was mainly dominated by 2-pyridyl-3-phenyl-5-chloropyridines. The third cluster was comprised of eight molecules based on a 1,5-diarylpyrazole scaffold. The cluster for the fourth eigenvalue, λ_4, also contained eight molecules and was dominated by diaryl cyclobutenones. In cluster five, five of the six molecules are 2,4,5-triarylthiazole derivatives (see ***Figure 3.5***). By looking at the molecules that comprise cluster 5, again we can see the ability of spectral clustering to differentiate between differing scaffolds; as molecule 5f has a different scaffold to the other molecules in the cluster and therefore makes a lower contribution in the eigenvector.

Brewer concludes that this method provides an intuitive and easy to implement method for clustering chemical data, which shows promise for use in separating compounds according to their chemical core, or Murcko scaffold. Brewer also highlights that the same molecule is

found in both clusters 1 and 5, molecules 1h and 5f respectively, and suggests that this indicates a possible scaffold hop.

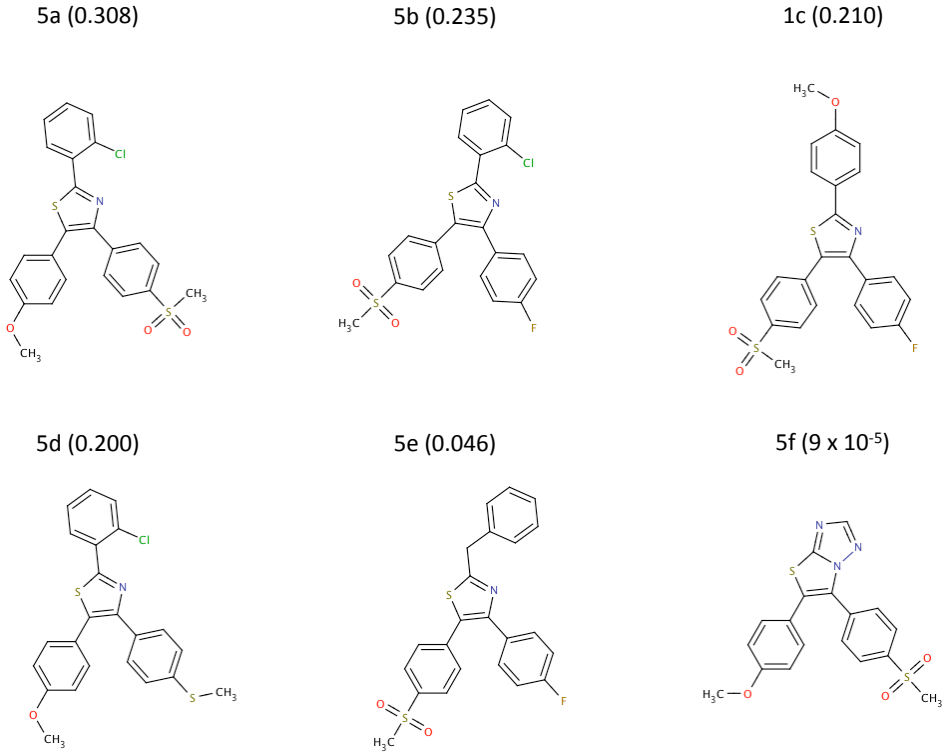


Figure 3-5: *Diagram shows the structures and their squared eigenvector components for cluster 5 when λ_5 =2.836 and γ = 25. Figure adapted from (Brewer, 2007).*

The spectral clustering algorithm discussed in this study has since been used within several other investigations, including:

- In the work of Neres et al. (2009) who used the spectral clustering algorithm to select representative molecules from 690 distinct clusters formed from a set of 1819 unique compounds that displayed activity as trypanosome cruzi trans-sialidase inhibitors (TcTS).
- A study by Davenport et al. (2010) where Brewer's approach to spectral clustering was applied as part of a 3D QSAR modelling protocol that was used to distinguish between the pharmacophores of histamine H3 receptors (H3R) and human ether-a-go-go-related gene (hERG) channel blockers.
- Within the GPCR (G protein coupled receptor) modelling stage of an investigation by Heifetz et al. (2013), aimed at providing structural insights into the melanin-concentrating hormone-1 receptor (MCH-1R) binding site and identifying novel chemotypes of potent MCH-1R antagonists.

3.6 Summary

Thus far the application of spectral clustering to chemical data has only been briefly explored in the literature (Brewer, 2007; Neres et al., 2009; Davenport et al., 2010; Heifetz et al., 2013). However, the promising results obtained in other fields, in particular bioinformatics where spectral clustering has shown significantly improved results in the clustering of protein sequences when compared to other clustering methods (Paccanaro et al., 2006) and genomics where an efficient method for inferring protein sequences has been developed (Nepusz et al., 2010), suggest that spectral clustering approaches may provide a useful tool in the modern drug discovery process.

The following chapters of this thesis aim to further establish the viability of using spectral clustering algorithms with pharmaceutical data. To achieve this goal spectral clustering algorithms will be both adapted and optimised, to allow their performances to be quantified using different measures and evaluated against the leading traditionally used clustering methods.

Chapter 4

Non-Overlapping Spectral Clustering

4.1 Introduction

The aim of this chapter is to provide a more in depth study of the viability of implementing spectral clustering with chemical data. To achieve this goal a non-overlapping spectral clustering technique, NOSC, based upon the method set out by Brewer (2007) was developed. The NOSC method utilises a full matrix diagonalisation algorithm to find the eigenvalues and eigenvectors of a similarity matrix. These eigenvectors are subjected to the "95% positive eigenvector" rule described by Sarkar and Boyer (1998), and the molecules placed into clusters based upon their largest component contribution to an eigenvector.

The performance of the NOSC in clustering datasets containing both actives and inactives was assessed for a variety of parameters, datasets and fingerprints. The ability of the NOSC technique to provide "meaningful" clusters was evaluated using the quality clustering index measure and the results compared to the leading traditional clustering algorithms.

4.2 A Non-Overlapping Spectral Clustering Algorithm

The NOSC method is based upon an adaptation of the spectral clustering algorithm set out by Brewer (2007). Implementation of the algorithm follows Brewer's method of generating a filtered similarity matrix, S^F, which was produced by applying a Gaussian type filtering function, given below, to the elements of the original similarity matrix.

$$S_{ij}^{F} = e^{-\gamma(S_{ij}-1)^2}$$

Subsequently a full matrix diagonalisation algorithm (see ***Section 3.5*** and ***Appendix A***) is applied to the matrix S^F to find the eigenvalues, λ_n, and eigenvectors, c_n. The eigenvectors are stacked as columns into an eigenvector matrix, *C*, such that each of the *n* columns in the matrix represents a potential eigencluster (eigenvetor) with a related eigenvalue, λ_n. Each row of the matrix *C* corresponds to a molecule, with each component providing a measure of how much that molecule contributes to a particular eigencluster. The columns of the eigenvector matrix are then tested to see if they are valid clusters based upon the 95% positive eigenvector rule outlined by Sarkar and Boyer (Sarkar and Boyer, 1998). Sarkar and Boyer proposed that a cluster should only be deemed meaningful if it fulfils a relaxed definition of a positive eigenvector. A positive eigenvector is described by two criteria:

- The eigenvector has an associated eigenvalue with a positive magnitude.
- The eigenvector elements are either all positive or all negative.

In the relaxed definition of a positive eigenvector, a positive associated eigenvalue is still required; however only 95% of the total contribution of an eigenvector must be provided by its dominant components, for example, 95% of the eigenvectors magnitude is provided by either the sum of the squared positive or negative components only. If a cluster is deemed valid, the contributions made by all molecules to the eigenvector are left unaltered. Eigenvectors that do not meet the criteria for a relaxed positive eigenvector are ignored, by setting all of the eigenvector's components to zero.

A threshold value is applied to the eigenvector contributions to filter out the components that make very small contributions to the eigenvector, such that if the size of the eigenvector element is less than the threshold, the component's value is set to zero. A consequence of this threshold is that any compound that contributes zero to each of eigenvectors then becomes a singleton cluster. In Brewer's method, each molecule makes a contribution to several clusters,

however, in the non-overlapping spectral clustering method described here each molecule is assigned to a single cluster (column) according to its largest contribution to an eigencluster.

4.3 Datasets

The four activity classes studied during this investigation (5HT1A antagonists, Matrix Metalloprotease inhibitors, Renin inhibitors and Substance P antagonists) were extracted from the ChEMBL database using the Pipeline Pilot software (Accelrys, 2011). During the extraction, each dataset was "cleaned" by deleting any duplicate molecules, removing all counter ions from salts and neutralising the remaining cations/anions using Pipeline Pilot. A molecule was classified as active if the corresponding $IC_{50} \leq 10000$ nM or $-\log(IC_{50}) \geq 5$, otherwise it was classed as inactive. The homogeneity of each of these activity classes was characterised using the mean pairwise similarity amongst the molecules calculated using Unity fingerprints and the Tanimoto similarity measure. Datasets that have a high mean pairwise similarity (larger than 0.5) are described as homogenous, whereas a mean pairwise similarity of less than 0.5 indicates a dataset is heterogeneous. ***Table 4.1*** provides further information on the datasets.

Activity Class	Abbreviations	Number of Molecules	Number of Actives	Mean Pairwise Similarity
Matrix Metalloprotease Inhibitors	MMP-1	3482	1764	0.381
5HT1A antagonists	5-HT1A	2784	299	0.354
Renin Inhibitors	Renin	2166	1746	0.520
Substance P Antagonists	NK1 / SubP	2760	1504	0.411

Table 4-1: *Information on the activity classes used within this investigation.*

4.4 Molecular Descriptors

The five fingerprints examined were BCI, Daylight, ECFP_4, MDL public keys and Unity fingerprints. Further information on two-dimensional molecular descriptors can be found in ***Chapter 2***.

BCI fingerprints were generated within the BCI software and are represented by a bitstring of 1052 binary characters derived using a fragment dictionary. Each bit represents a different

structural fragment selected for its ability to discriminate between molecules. The presence of a fragment within the molecule is denoted with a 1 and its absence with a 0.

Daylight fingerprints are an example of a hashed fingerprint based on generating all possible linear sequences of atoms and bonds up to a specified length that are present in the molecule. The default settings were used to calculate all sequences to a maximum path length of 7. These atom and bond sequences are represented by sets of bits designated by a hashing function. The Daylight fingerprints were generated with the Daylight toolkit and were encoded into fingerprints containing 2048 binary characters.

Extended connectivity fingerprints, otherwise known as ECFPs, are generated using a circular substructure approach to encoding molecules. Each atom in a molecule is assigned an integer value according to a set of features, including: the atom type, the charge, the number of connections to the atom and the atomic mass. These integer values are subsequently operated on using an algorithm analogous to the Morgan algorithm for a number of iterations. ECFP_2 fingerprints are the product of a single iteration, ECFP_4 are the product of two iterations, etc. The number that is appended to the name refers to the number of bonds that the circular substructures span. The ECFP fingerprints were generated using the Pipeline Pilot software (Accelrys, 2011).

MDL public keys are small two-dimensional fingerprints based upon the use of a structural key. The public keys were generated within the Pipeline Pilot software and are represented by a 166 bitstring.

The Unity fingerprint system uses a combination of both a structural key and a hashed fingerprint system. Unity fingerprints are represented by a bitstring of 992 binary characters and were generated within the Sybyl software.

Figures 4.1 to 4.4 show the distribution of pairwise similarity values generated using the Tanimoto coefficient and the different fingerprint types, for each of the four datasets in ***Table 4.1***, in the form of histograms produced by binning the elements of the similarity matrices into twenty bins of equal size.

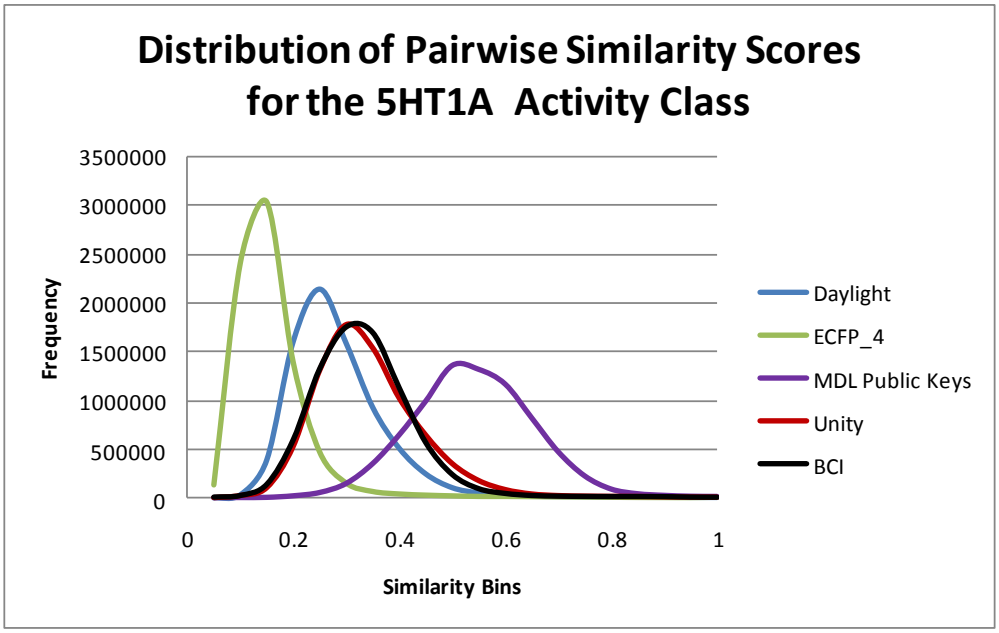


Figure 4-1: *The distribution of pairwise similarity scores for the 5HT1A activity class, generated using the Tanimoto coefficient and five different molecular fingerprints.*

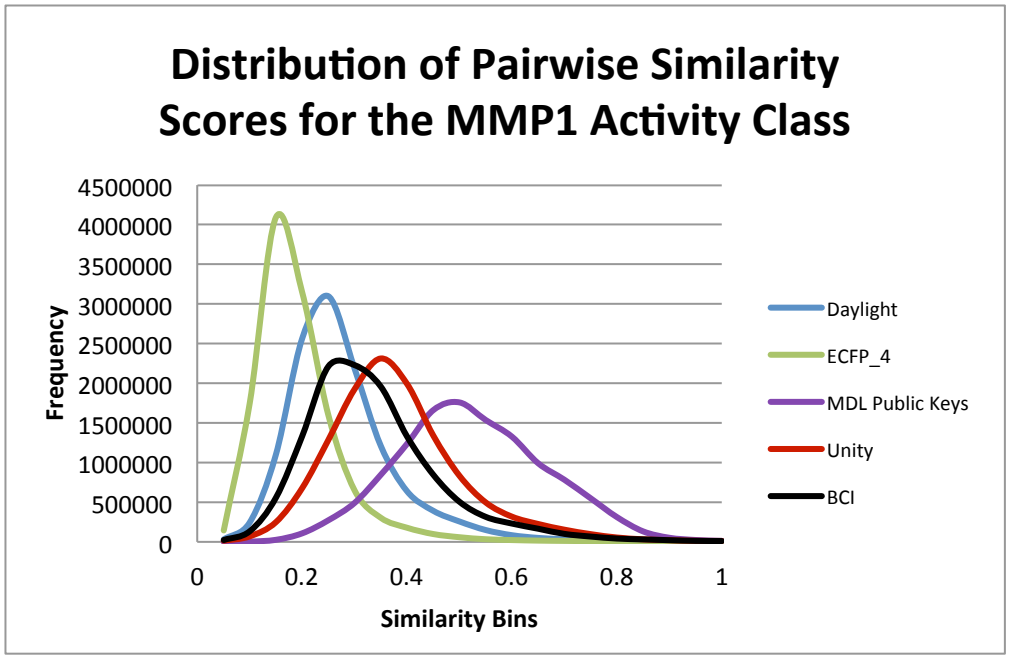


Figure 4-2: *The distribution of pairwise similarity scores for the MMP1 activity class, generated using the Tanimoto coefficient and five different molecular fingerprints.*

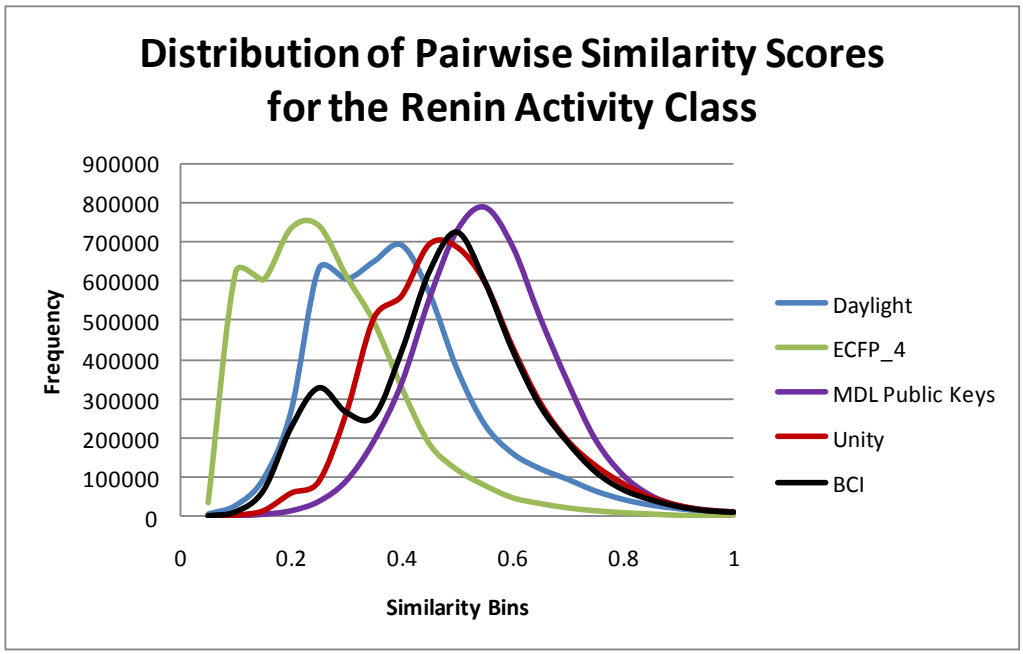


Figure 4-3: *The distribution of pairwise similarity scores for the Renin activity class, generated using the Tanimoto coefficient and five different molecular fingerprints.*

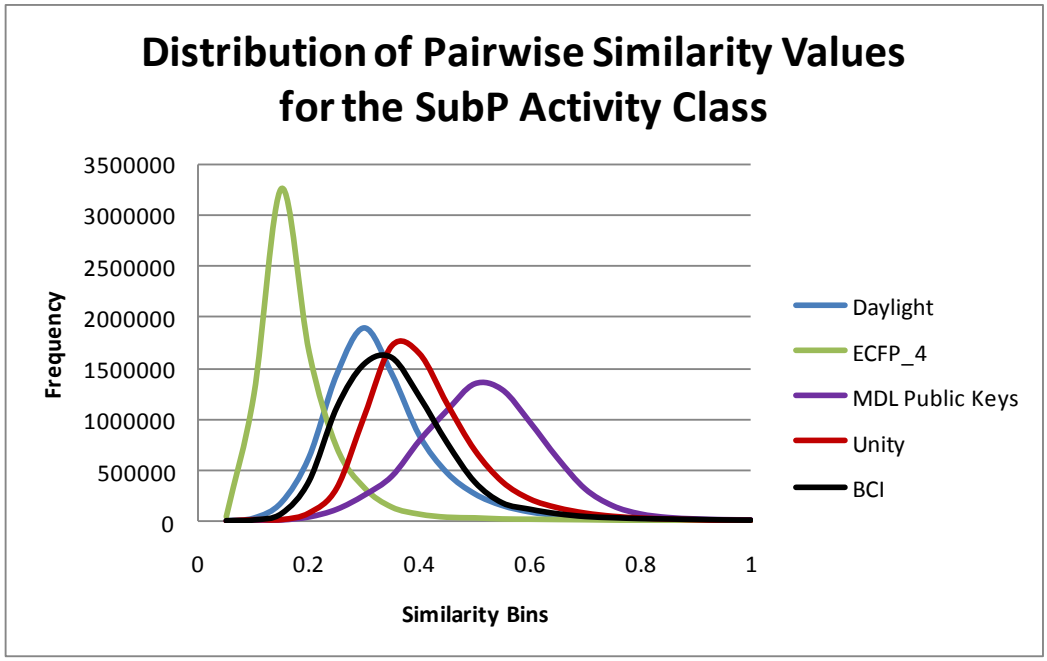


Figure 4-4: *The distribution of pairwise similarity scores for the SubP activity class, generated using the Tanimoto coefficient and five different molecular fingerprints.*

4.5 Investigating the Parameters that affect the NOSC Algorithm

Given a similarity matrix, the assignment of molecules to clusters is dependent on two variables: the value of γ in the Gaussian filtering function that is used to filter the similarity matrix prior to the application of the eigendecomposition algorithm and the threshold value applied to remove components of the eigenvector matrix. The effect of these variables on the numbers of clusters formed are explored for the five fingerprint types, by varying the value of γ from 25 - 400 at incrementing values of 25 and using differing eigenvector thresholds values (0.1, 0.01, 0.001, ..., 1 x 10^{-6}).

All experiments in this and subsequent chapters have been carried out on a Macbook pro with a 2 GHz Intel Core i7 processor, 8 GB of RAM and running OSX 10.7 Lion.

4.5.1 Results

The results are presented in ***Tables 4.2 - 4.5***. Where the number of clusters (given as the total number of both singleton and non-singleton clusters) and singletons generated for each dataset described by the different molecular fingerprint types are provided along with the number of molecules left unclassified. Molecules were left unclassified if all of their contributions to the eigenclusters were below a threshold value of 1 x 10^{-6}.

γ	BCI			Daylight			ECFP_4			MDI Public Keys			Unity		
	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules
25	147	2	0	203	3	0	353	20	0	5	0	0	111	0	0
50	308	38	0	390	42	0	604	82	0	45	1	0	311	20	0
75	391	67	0	460	86	0	731	113	1	97	5	0	401	61	0
100	450	103	0	501	100	0	803	128	8	181	15	0	454	100	0
125	485	117	0	539	106	0	844	134	6	249	22	0	508	111	0
150	520	122	0	556	127	0	873	141	5	314	50	0	550	118	0
175	561	131	0	570	142	1	882	146	41	357	44	0	564	124	7
200	586	139	0	584	143	1	876	146	49	400	61	0	583	123	9
225	602	155	0	600	151	3	858	140	69	446	74	0	598	137	10
250	626	156	0	610	157	3	845	135	115	470	83	0	609	136	18
275	631	162	28	616	164	6	832	126	125	489	90	0	622	142	18
300	649	161	34	626	172	5	821	115	158	508	97	0	637	148	21
325	655	165	38	634	174	38	800	117	160	526	100	0	644	158	22
350	663	170	42	647	171	38	767	105	204	546	104	0	660	166	30
375	672	170	45	641	176	41	731	101	223	553	106	0	668	167	31
400	691	173	52	650	170	44	710	95	274	562	116	0	674	164	29

Table 4-2: *The effect of varying γ has on the clusters for 5HT1A dataset (using a constant eigenvector threshold of 1 x 10^{-6}).*

γ	BCI			Daylight			ECFP_4			MDI Public Keys			Unity		
	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules
25	93	6	0	142	4	0	279	12	0	11	0	0	67	3	0
50	210	27	0	369	49	0	631	76	0	46	2	0	181	16	0
75	308	51	0	490	98	0	827	143	0	90	8	0	280	43	0
100	387	72	0	564	126	0	951	189	1	143	14	0	362	61	0
125	437	90	0	624	132	0	1020	216	0	208	17	0	424	84	0
150	496	96	0	676	149	0	1070	243	0	263	24	0	490	101	0
175	542	112	0	716	173	2	1104	246	3	324	37	0	526	132	0
200	581	129	0	754	184	1	1117	250	0	355	46	0	558	142	0
225	604	146	0	773	199	2	1112	244	2	395	55	0	590	151	0
250	642	148	0	794	192	2	1104	262	6	426	67	0	615	161	0
275	665	165	0	810	202	3	1088	246	0	461	76	0	640	163	0
300	688	178	0	821	208	5	1092	230	4	479	77	0	667	170	1
325	698	178	0	828	211	0	1073	228	0	515	76	0	689	171	2
350	731	181	1	832	218	0	1066	215	0	548	91	0	722	167	3
375	737	193	1	861	206	5	1059	193	8	565	96	0	736	171	0
400	748	198	3	856	221	7	1026	175	13	574	104	0	753	189	4

Table 4-3: *The effect of varying γ has on the clusters for MMP1 dataset (using a constant eigenvector threshold of 1 x 10^{-6}).*

γ	BCI			Daylight			ECFP_4			MDI Public Keys			Unity		
	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules
25	25	1	0	44	1	0	81	3	0	4	0	0	20	0	0
50	56	4	0	116	7	0	230	13	0	17	1	0	51	1	0
75	99	6	0	195	21	0	360	40	0	30	2	0	95	6	0
100	121	8	0	238	36	0	465	63	0	45	7	0	133	14	0
125	161	11	0	273	47	0	542	87	0	61	6	0	165	26	0
150	193	21	0	300	56	0	612	112	0	83	11	0	199	29	0
175	228	24	0	325	69	0	646	122	0	117	13	0	220	36	0
200	254	34	0	361	77	2	678	144	0	131	16	0	245	57	0
225	283	46	0	370	82	0	697	157	0	163	17	0	282	54	0
250	290	51	0	381	95	0	711	162	0	187	17	0	298	58	0
275	308	55	0	407	100	0	725	156	0	201	24	0	322	62	0
300	328	61	0	409	105	1	728	176	0	219	22	0	333	66	0
325	342	71	0	430	102	2	724	174	0	235	27	0	348	63	0
350	356	76	0	437	110	0	731	166	0	251	22	0	351	73	1
375	380	75	0	437	117	1	734	164	0	268	33	0	368	80	1
400	381	85	0	437	119	0	734	159	0	288	34	0	376	81	1

Table 4-4: *The effect of varying γ has on the clusters for Renin dataset (using a constant eigenvector threshold of 1 x 10^{-6}).*

γ	BCI			Daylight			ECFP_4			MDI Public Keys			Unity		
	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules	Clusters	Singletons	Unclassified Molecules
25	82	1	0	85	0	0	224	10	0	3	0	0	25	0	0
50	188	10	0	237	28	0	450	53	0	37	0	0	131	7	0
75	262	30	0	325	57	0	589	86	4	97	3	0	233	26	0
100	323	45	0	386	82	0	669	126	1	166	14	0	293	45	0
125	364	66	0	426	101	0	753	148	1	213	25	0	339	60	0
150	398	87	0	460	112	0	804	156	1	275	33	0	391	84	0
175	428	95	0	495	119	0	837	165	1	318	44	0	424	98	0
200	466	105	0	516	118	0	854	170	2	367	52	0	452	105	0
225	485	96	2	532	118	0	864	168	4	396	58	0	474	105	0
250	502	115	1	548	133	0	862	181	9	408	66	0	490	116	0
275	512	125	0	559	135	0	852	180	0	437	74	0	518	133	0
300	532	122	0	566	140	0	850	165	0	458	84	0	545	137	6
325	550	125	0	581	143	0	843	162	43	481	91	0	568	143	6
350	557	137	0	593	153	0	832	151	65	495	101	0	586	136	6
375	567	141	0	605	164	0	823	148	81	516	111	0	599	141	6
400	590	137	0	609	172	1	807	156	92	533	118	0	611	144	3

Table 4-5: *The effect of varying γ has on the clusters for SubP dataset (using a constant eigenvector threshold of 1 x 10^{-6}).*

A number of different trends and relationships within the data can be identified in ***Tables 4.2 - 4.5***. These trends include global trends, for example, those common to all datasets and fingerprint types, and trends related specifically to a particular fingerprint type or dataset. The most notable global trend is the increase in the number of non-singleton clusters produced for each dataset as the value of γ is raised. This trend continues until either a plateau is reached or the value of γ is large enough to lead to the formation of artificial singleton clusters, for example, the 5HT1A activity class described by ECFP_4 fingerprints. These artificial singletons occur in the case where the filtering function places an emphasis on the molecule with the largest contribution only, decreasing all other similarity values to the point that no other molecules are placed in the same cluster as the molecule with the largest similarity value regardless of their respective magnitudes, for example, two molecules with respective similarity values of 0.81 and 0.79 may be placed incorrectly into separate clusters at sufficiently high values of γ. Another trend common to all fingerprint types is the general increase in the number of singleton clusters produced via the NOSC method as the value of γ is increased. However, there are several instances which do not obey this trend and the number of singleton clusters decreases as the value of γ is increased, for example, between the values of γ = 200 and 225 for the SubP dataset described by BCI fingerprints. The presence of the anomalous results can be explained by either one of two factors:

- The Gaussian filtering function de-emphasising the eigenvector contribution of a molecule below the value of the eigenvector threshold, resulting in that molecule no longer being placed in the singleton cluster.
- Further molecules are placed into these clusters as the threshold value is decreased to allow the inclusion of molecules based on smaller eigenvector magnitudes.

Comparing the number of clusters produced for different values of γ indicates that ECFPs provided the best separation of the data at low values of γ, continually producing a significantly larger number of non-singleton clusters than any other fingerprint type analysed during this investigation. Conversely, the results produced when using MDL Public Keys show that a minimal number of clusters are formed at the lowest values of γ.

Next, a subset of the results obtained for the investigation into how the magnitude of the eigenvector threshold value affects the NOSC algorithm's ability to cluster a dataset are presented. In ***Tables 4.6 and 4.7*** the distribution of clusters, singletons and unclassified molecules produced by NOSC when using ECFP_4 fingerprints, two values of γ values (25 and 75) and an eigenvector threshold value between 0.1 and 1×10^{-6} are given. The results for the

other fingerprint types are present in ***Appendix B***, which contains full results Tables for the investigations carried out in this chapter.

Activity Class	Gamma	Eigenvector Threshold	# Clusters	# Singletons	# Unclassified
5HT1A	25	0.1	343	30	892
		0.01	353	20	65
		0.001	353	20	3
		0.0001	353	20	0
		0.00001	353	20	0
		0.000001	353	20	0
	75	0.1	724	120	356
		0.01	731	113	22
		0.001	731	113	3
		0.0001	731	113	2
		0.00001	731	113	1
		0.000001	731	113	1
MMP1	25	0.1	273	18	1365
		0.01	279	12	157
		0.001	279	12	2
		0.0001	279	12	0
		0.00001	279	12	0
		0.000001	279	12	0
	75	0.1	802	168	544
		0.01	825	145	43
		0.001	827	143	2
		0.0001	827	143	0
		0.00001	827	143	0
		0.000001	827	143	0

Table 4-6: *Shows the variations in the distribution of cluster types at a range of eigenvector threshold values for the 5HT1A and MMP1 datasets. In this table each dataset is described by ECFP_4 fingerprints at two values of gamma, 25 and 75.*

The results highlight how the number of molecules placed within non-singleton clusters increases as the value of the eigenvector threshold is decreased. This in turn leads to an increase in the number of non-singleton clusters coupled with a decrease (or the reaching of a plateau) in the number of molecules left unclassified or placed within a singleton cluster. Of particular interest is how the number of molecules unclassified dramatically decreases as the threshold value is reduced. Further discussion of these results is provided in ***Section 4.5.2***.

Activity Class	Gamma	Eigenvector Threshold	# Clusters	# Singletons	# Unclassified
Renin	25	0.1	81	3	1305
		0.01	81	3	269
		0.001	81	3	12
		0.0001	81	3	0
		0.00001	81	3	0
		0.000001	81	3	0
	75	0.1	344	56	637
		0.01	360	40	57
		0.001	360	40	1
		0.0001	360	40	0
		0.00001	360	40	0
		0.000001	360	40	0
SubP	25	0.1	217	17	990
		0.01	223	11	107
		0.001	224	10	11
		0.0001	224	10	0
		0.00001	224	10	0
		0.000001	224	10	0
	75	0.1	571	104	523
		0.01	587	88	53
		0.001	588	87	15
		0.0001	588	87	10
		0.00001	588	87	8
		0.000001	589	86	4

Table 4-7: *Shows the variations in the distribution of cluster types at a range of eigenvector threshold values for the Renin and SubP datasets. In this table each dataset is described by ECFP_4 fingerprints at two values of gamma, 25 and 75.*

4.5.2 Discussion

This section provides a discussion on the effect of varying molecular fingerprint (***Section 4.5.2.1***), value of γ in the Gaussian filtering function (***Section 4.5.2.2***) and magnitude of the eigenvector threshold, (***Section 4.5.2.3***) on the clustering of the four activity classes.

4.5.2.1 The Effect of Molecular Descriptors on Spectral Clustering

The key step in the application of spectral clustering to chemical data is the recognition that a similarity matrix is equivalent to a weighted adjacency matrix of a graph of N nodes. Thus, the composition of a similarity matrix plays a hugely important role in spectral clustering. ***Figures 4.1 to 4.4*** show how the similarity matrices vary for the different fingerprints, with these

differences providing the basis for explaining why certain characteristics occur within the clusters produced for the fingerprint types and datasets.

Previous studies have shown that ECFPs produce the most unique and specific two-dimensional molecular fingerprints, leading to the greatest differentiation between molecules (Hert et al., 2004; Bender et al., 2009). This characteristic of ECFPs means that the distribution of similarity values within an ECFP similarity matrix is naturally skewed towards low similarity scores, for example, each column in a similarity matrix will typically be comprised of a few large similarity values, with the rest of the similarity scores being low, due to the ability of ECFP fingerprints in differentiating between different compounds. As the bulk of the similarity values are so low, the largest similarity scores are naturally accentuated when an eigendecomposition algorithm is applied. This natural emphasis on the largest contributing molecules is further stressed by the filtering function and as a result the largest similarity scores in each column of the filtered similarity matrix are therefore reflected by the magnitude of their components in the corresponding eigenvectors. This leads to the production of numerous clusters containing only a few closely related chemical structures when using the NOSC algorithm. As would be expected there is a wider distribution of similarity values for the highly homogeneous renin activity class, which produces slightly fewer clusters that are comparatively larger than the clusters for the heterogeneous datasets, reflecting how the renin activity class contains fewer molecular scaffolds.

Daylight fingerprints are an intermediate in the specificity compared to ECFP and BCI/Unity fingerprints, which is shown by the relative position of the general distribution of similarity values. This feature leads to Daylight similarity matrices containing columns in which only the most similar compounds are represented via high similarity scores, with all other molecules being assigned very low similarity scores. Thus the clusters produced for Daylight fingerprints by the NOSC algorithm are both numerous and small, containing several highly related compounds.

The performance of both the Unity and BCI fingerprints varies based on the dataset they are applied to. Both fingerprint types are less specific than the ECFP and Daylight fingerprints, and therefore assign pairs of molecules higher similarity scores. This leads to the generation of two types of clusters by the NOSC algorithm: small clusters containing the most highly related compounds and larger globular clusters containing many molecules that share some molecular features.

MDL public keys produce similar distributions for both homogeneous and heterogeneous activity classes, and are characterised by the broad distributions of similarity values they produce. Within this distribution the majority of the similarities fall into one of the central bins, i.e., bins between 0.35 - 0.75. The increased similarity values obtained when using MDL public keys, and therefore the distribution of similarities, can be attributed to the size of the fingerprints. MDL public keys are only 166 binary characters in length. This small size means there is a limited amount of information that can be encoded into the fingerprints. Therefore their ability to discriminate between molecules that are structurally dissimilar yet have a similar chemical composition (or vica-versa) can be diminished, e.g., two molecules that are structurally different but consist of similar chemical components may yield a higher than anticipated similarity value when using MDL public keys. Clusters produced from MDL public keys by the NOSC algorithm are typically large (with large eigenvalues) that often include molecules that chemically and structurally, are not always similar.

Using the histograms shown in ***Figures 4.1 – 4.5***, the relative specificity of these fingerprints is outlined below:

ECFP_4 > Daylight > BCI > Unity > MDL Public Keys

This ranking of the fingerprint types can be used to explain which fingerprint type produces the most clusters (ECFP, which produces the largest number of clusters, to MDL public keys, which produce the least) and can also be used to relate the fingerprint type to the sizes of the clusters produced (ECFP, produces the largest number of small clusters, to MDL public keys, which produce the largest clusters).

4.5.2.2 The Effect of varying γ in the Gaussian Filtering Function

In 2007, Brewer carried out preliminary studies on the implementation of spectral clustering based upon similarity matrices derived from Unity fingerprints (Brewer, 2007). From this preliminary work it was concluded that when using an unmodified version of the similarity matrix, each molecule contributes to a cluster by a similar amount regardless of structural differences in the molecules. Hence, the use of the eigenvectors and eigenvalues of a modified version of the similarity matrix was required to obtain clusters with increased discrimination between structurally different molecules. This modified similarity matrix was formed using a Gaussian type filtering function, shown in ***Section 4.2***. The increase in discrimination between molecules that is gained through the use of the Gaussian filtering function can be simply explained. Consider a contrived example of how the molecules are situated in theoretical

chemical space (see ***Figure 4.5***). In this figure, the respective distances between a reference molecule, which all similarity scores are measured from, and the other molecules in the dataset is given by their similarity score. Molecules with the greatest similarity scores are located closest to the reference molecule and conversely molecules that are not similar to the reference structure are located furthest away.

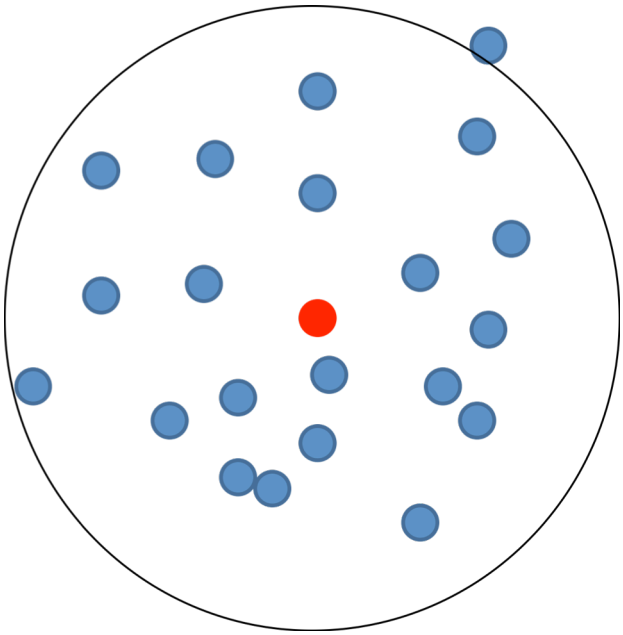


Figure 4-5: *A contrived example of the arrangement of molecules in theoretical chemical space, where the red node is the centroid (reference) molecule from which all similarities scores are measured; and the blue nodes are molecules whose distance from the centroid is given by their similarity scores. The circle is used to show the eigencluster formed from these molecules.*

The application of the Gaussian filtering function to the similarity values leads to each similarity score being reduced to varying degrees based upon its magnitude; such that the similarity value between two highly related molecules is left almost unaltered by the filtering function, whilst the score between two molecules that are highly dissimilar is decreased by a greater amount (see ***Figure 4.6***). In this figure the molecules closest to the centroid have only moved minutely, whereas the molecules with the lowest similarity scores have moved comparatively much further. The implementation of this filtering function therefore leads to the production of clusters that contain fewer compounds, with the molecules that have the lowest similarity values not being placed within the cluster.

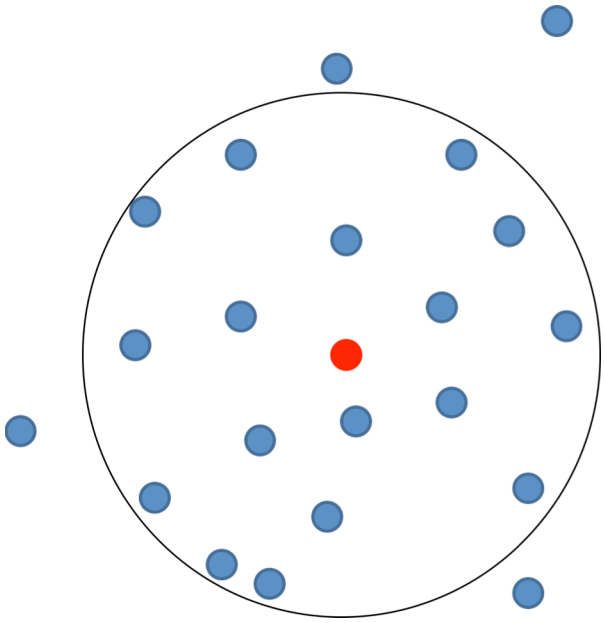


Figure 4-6: *The locations of the molecules after the application of the Gaussian filtering function.*

The effect that the Gaussian filtering function has on the distribution of similarity values within a dataset is highlighted in the example shown in ***Table 4.8***. The Gaussian filtering function has been applied to the distribution of similarities for the Substance P antagonist dataset represented by MDL public keys, at differing values of γ. The Gaussian filtering function decreases the emphasis placed on the lowest similarity scores resulting in a dramatic increase in the number of similarity scores that populate the lowest similarity bin, which goes from 206 entries in the original similarity matrix to 6728364 entries when the tuneable parameter γ is set to 25. As the value of γ is steadily increased the number of similarity values that a low emphasis is placed on also continues to rise, diminishing the effect of the background similarity contained within the fingerprints.

	Similarity Frequency			
Similarity Bin	**Original Similarity Matrix**	**γ = 25**	**γ = 50**	**γ = 75**
0 - 0.049	206	6728364	7432124	7519424
0.05 - 0.099	1244	430430	63540	25312
0.1 - 0.149	10544	160982	28422	13082
0.15 - 0.199	40608	84244	16544	7788
0.2 - 0.249	115484	50712	11706	6198
0.25 - 0.299	255242	32418	7472	4808
0.3 - 0.349	442364	24290	6772	4136
0.35 - 0.399	793400	16782	5454	3328
0.4 - 0.449	1090124	12976	4656	2896
0.45 - 0.499	1353280	11138	4058	2928
0.5 - 0.549	1295094	8156	3840	2800
0.55 - 0.599	975960	6464	3434	2366
0.6 - 0649	621508	7412	3316	1944
0.65 - 0.699	318166	5758	3038	2696
0.7 - 0.749	151192	5296	2634	1724
0.75 - 0.799	70510	5242	3294	2422
0.8 - 0.849	35266	5410	3012	1616
0.85 - 0.899	21310	4834	2416	2230
0.9 - 0.949	13986	5092	2682	2560
0.95 - 1	12112	11600	9186	7342

Table 4-8: *The effect that varying the value of γ has on the distribution of similarity values for the Substance P antagonist dataset represented by MDL public keys.*

Let us now consider the general features of the eigenvectors that correspond to the distribution of similarity values before and after the application of the Gaussian filtering function. First, recall that each eigenvector, c, is pre-normalised to unity according to the function:

$$1 = \sum_{j=1}^{N} c_{ij}^2$$

Where N is the number of molecules in the dataset and c_{ij} is the eigenvector component at row *i* for the j^{th} eigenvector.

Thus, the eigenvector produced from the distribution of similarity values prior to the application of the Gaussian filtering function would be comprised of N similarly sized components, indicating that each molecule contributes to a similar degree to the total magnitude of the eigenvector. This leads to the generation of a large eigencluster containing

each molecule from the dataset. After applying the Gaussian filtering function, the distribution of similarity values is greatly increased. In the filtered eigenvectors, the components that relate to the largest similarity scores are increased significantly by the filtering of the similarity scores, whereas the components associated with the lowest similarity scores are considerably decreased. In this eigenvector the majority of the components are substantially smaller than the largest, which allows for clusters containing only some of the compounds to be generated.

As the value of γ is increased, the number of eigenclusters, both singleton and non-singleton, increases significantly. The number of compounds contained in each of the non-singleton eigenclusters decreases as the value of γ is raised. This continues until γ reaches a value that is sufficiently large to lead to the production of artificial singleton clusters.

For the most specific types of molecular fingerprints, i.e., ECFP_4 and Daylight fingerprints, the application of the NOSC algorithm is able to produce a number of clusters even at the lowest values of γ used within this experiment. This is because the more specific fingerprint types are able to differentiate between molecules to a greater extent. Therefore, the large similarity scores are typically less affected by the level of background similarity and continue to produce sizeable eigenvector elements. As the specificity of the fingerprints decreases a larger value of γ is required to allow differentiation between structures.

4.5.2.3 The Effect of Varying the Eigenvector Threshold

In the NOSC method, an eigenvector threshold is applied to the elements of the matrix C, to set eigenvector components less than the threshold value to zero. Varying the threshold value can have a dramatic effect on the number of clusters and singletons produced by the NOSC algorithm. If the threshold value is set too high, then all molecules with a maximum eigenvector contribution below this threshold cannot be placed into clusters. In this case the NOSC algorithm will return a small number of clusters containing very few highly similar molecules. Of course, if this eigenvector threshold is set too low, some compounds that should have been placed into singletons will instead be assigned to clusters along with molecules that they are both structurally and chemically dissimilar to. As the eigenvector threshold value is decreased, the number of molecules left unclassified by the NOSC algorithm decreases. This is due to the molecules with the lowest eigenvector components being classified when the value of the threshold is relaxed (see ***Figure 4.7***).

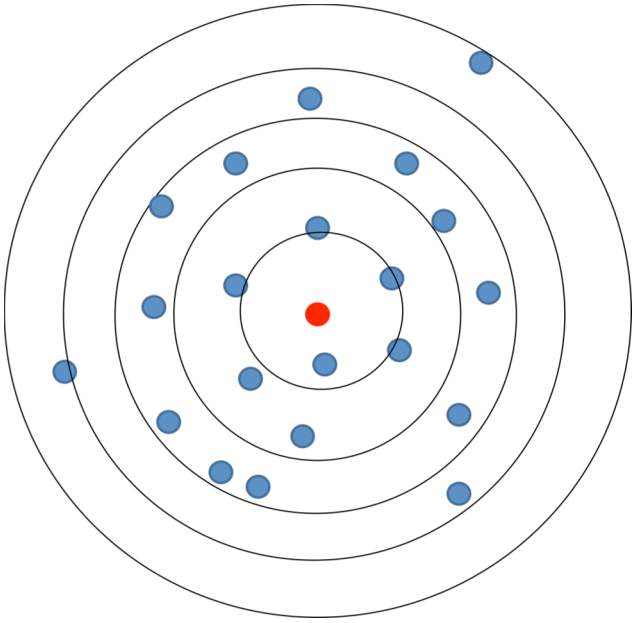


Figure 4-7: *Diagram depicting which molecules would be placed into an eigencluster at different threshold values. Each concentric circle is used to show a different threshold value, i.e., the largest ring shows the smallest threshold value that is able to classify all molecules into clusters whereas the smallest ring shows the largest eigenvector threshold in which only one molecule is classified into the eigencluster.*

As the homogeneity of an activity class increases, the value of the threshold should be decreased further to account for the smaller eigenvector contributions produced by the compounds that share largely similar scaffold/chemical features yet have their component scores penalised by the normalisation function applied to the eigenvectors identified using the full matrix decomposition algorithm.

4.5.2.4 Selecting Optimal Parameters

The selection of optimal parameters for any clustering algorithm is a key requirement in ensuring that "good" set of clusters is found. In this case, the term good is used to reflect a set of clusters that provides a balance between identifying enough clusters to allow for an effective partitioning of the dataset, whilst minimising the number of compounds that are left unclassified or misclassified (placed into clusters where they share minimal chemical and structural properties with the other members). For hierarchical clustering algorithms, there are several measures aimed at selecting a level of hierarchy that provides an apt set of clusters (see ***Chapter 2***). However, in spectral clustering, optimisation is a far more complex procedure due to number of factors that can have a significant effect on the results. These factors include the value of γ in the Gaussian filtering function and the eigenvector threshold value and

crucially how the homogeneity of the dataset and the choice of fingerprint can affect these first two parameters.

Optimal value ranges for different fingerprint types and homogeneities can be determined by studying the impact these factors have on the clusters produced using different parameter values. The rationale for the selection of parameter ranges for applying spectral clustering with datasets of varying homogeneity is provided below.

For heterogeneous datasets such as the MMP1 and SubP activity classes, different values of γ are required based on the specificity of the fingerprints used. For the most specific fingerprints, ECFP_4 and Daylight (in most cases), a γ value of 50 - 75 is sufficient to produce a good separation of the data, whereas this value must be increased to 75 for the BCI and Unity fingerprints, which are less able to differentiate between some molecules. As MDL public keys are far more generic than the four other fingerprint types, the value of γ required to produce a good separation of the data is in the range 100 - 150. For each of these fingerprints, an eigenvector threshold value of 0.001 is capable of assigning all but the smallest contributing molecules to clusters.

In homogeneous datasets, such as the Renin activity class, many molecules may share molecular scaffold features and as a result, lead to the production of clusters containing molecules based on several different yet highly similar scaffolds that ideally should be placed into separate clusters. In order to gain the level of discrimination that is required to produce a good clustering of these compounds, the value of γ used in the Gaussian filtering function is generally larger than the magnitude of the value used with heterogeneous datasets, e.g., 75 - 100 for Daylight and ECFP_4 fingerprints, 100 - 150 for Unity and BCI fingerprints and 200+ for MDL public keys. As many molecules are based on similar molecular scaffolds, the value of the eigenvector threshold should also be relaxed to around 1×10^{-6}, in order to account for the impact that the increase in average similarity score has on the pre-normalisation function applied to eigenvectors.

4.6 Clustering Activity Classes Using the NOSC Algorithm

The performance of the NOSC approach was compared to two of the leading traditional algorithms, the k-means and Ward's methods. To carry out this investigation, the NOSC algorithm was applied to each dataset using the (near optimal) parameters set out in ***Table 4.9***.

	5HT1A		MMP1		Renin		SubP	
	γ	**Threshold**	**γ**	**Threshold**	**γ**	**Threshold**	**γ**	**Threshold**
BCI	75	0.001	75	0.001	175	1×10^{-6}	75	0.001
Daylight	75	0.001	50	0.001	175	1×10^{-6}	50	0.001
ECFP_4	50	0.001	50	0.001	125	1×10^{-6}	50	0.001
MDL PK	100	0.001	100	0.001	200	1×10^{-6}	100	0.001
Unity	75	0.001	75	0.001	175	1×10^{-6}	75	0.001

Table 4-9: *Parameter values used for comparison between the NOSC algorithm and other clustering methods.*

The clustering of each dataset by the NOSC algorithm, and both traditional clustering methods, was evaluated using the quality clustering index measure, otherwise known by the acronym QCI (see ***Section 4.7.1***).

The Ward's and k-means clustering methods were applied to the four ChEMBL activity classes using the BCI software (DigitalChemistry, 2005). In each of these approaches the number of clusters was determined by a user-defined parameter, k. In this investigation both the k-means and Ward's methods were applied at two different clustering levels. Initially, the k-means and Ward's methods were implemented using k values equal to the number of clusters produced by the NOSC algorithm to ensure that any comparisons between clustering algorithms were not unduly affected by the number of clusters produced by each approach. Next, both the k-means and Ward's approaches were also implemented at an optimal clustering level determined using the BCI OPTCLUS program, which harnesses the Kelley measure in its calculation of what is an optimal level of clustering (see ***Section 2.5.3***). This second implementation was undertaken to allow us to draw a comparison between the performance of the NOSC and the optimal performances of each of the other methods.

4.6.1 The QCI Measure

The Quality Clustering Index, more commonly known as the QCI measure, was developed by Varin et al. (2008; 2009) and is used to evaluate the performance of a clustering algorithm in terms of its ability to separate active and inactive molecules within a dataset, using the equation:

Equation 24: *Equation for calculating the QCI measure.*

$$QCI = \frac{p}{p + q + r + s} \times 100$$

Where:

p is the number of active molecules in active clusters.

q is the number of inactive molecules in active clusters.

r is the number of active molecules in inactive clusters.

s is the number of active singleton clusters.

An active cluster is defined as a cluster containing a greater percentage of active molecules than the dataset as a whole.

4.6.2 Results and Discussion

Table 4.10 provides both the number of clusters and QCI score produced for each of the respective clustering algorithms. Comparing the different clustering approaches highlights how the performance of each algorithm varies according to the selection of molecular descriptor.

In general, each of the clustering methods produces their largest QCI scores when using ECFP_4 fingerprints, with spectral clustering outperforming both k-means and Ward's methods when using this descriptor in the majority of cases. In particular, the NOSC algorithm performs extremely well in the clustering of the 5HT1A class, where ECFP's ability to differentiate between the most closely related compounds leads to the production of multiple clusters containing a few closely related molecules, dramatically increasing the value of p within the QCI calculation. Conversely, the performance of each of the clustering methods when coupled with MDL Public Keys is comparatively poor, due to these fingerprints having a limited ability to differentiate between closely related chemical structures. This has a particularly large effect on the results of spectral clustering, leading to NOSC consistently producing smaller QCI scores than the other clustering algorithms.

Dataset	Fingerprint Type	NOSC		Ward's (NOSC)		k-means (NOSC)		Ward's (OPTCLUS)		k-means (OPTCLUS)	
		# Clusters	QCI	# Clusters	QCI	# Clusters	QCI	# Clusters	QCI	# Clusters	QCI
5HT1A	BCI	391	34.069	391	33.319	391	34.252	359	29.702	348	30.626
	Daylight	460	35.244	460	31.047	460	31.867	375	30.869	363	27.375
	ECFP_4	604	57.404	604	35.632	604	34.572	396	33.497	373	31.791
	MDL Public Keys	181	31.061	181	26.119	181	22.748	361	31.078	332	29.535
	Unity	401	32.294	401	32.169	401	30.815	396	31.678	388	29.103
MMP1	BCI	308	71.049	308	69.938	308	73.343	329	74.530	315	72.833
	Daylight	369	72.784	369	70.214	369	74.461	359	74.201	340	72.067
	ECFP_4	631	81.701	631	76.805	631	80.542	413	82.669	391	81.750
	MDL Public Keys	143	66.168	143	73.352	143	70.575	362	78.454	348	77.502
	Unity	280	73.384	280	76.893	280	75.424	362	79.329	337	77.280
Renin	BCI	228	60.675	228	60.429	228	60.033	194	59.033	182	57.464
	Daylight	325	61.023	325	60.821	325	61.628	224	63.875	209	59.604
	ECFP_4	542	83.125	542	69.753	542	66.835	255	64.793	249	65.717
	MDL Public Keys	131	72.747	131	60.258	131	62.667	208	64.553	200	62.983
	Unity	220	56.659	220	62.977	220	62.350	237	64.831	226	63.945
SubP	BCI	262	59.043	262	63.705	262	62.393	278	64.432	269	62.876
	Daylight	237	61.786	237	61.651	237	61.292	279	62.466	269	61.344
	ECFP_4	450	65.840	450	65.611	450	65.400	323	65.525	313	63.726
	MDL Public Keys	166	46.300	166	58.876	166	55.550	312	61.830	294	59.403
	Unity	233	56.959	233	59.978	233	60.000	269	61.798	261	60.580

Table 4-10: *Comparative performances of the three clustering algorithms using the QCI measure.*

When using other fingerprint types, the performance of the three clustering algorithms fluctuates, with the k-means method generally producing the largest QCI scores when using BCI fingerprints; the Ward's method producing the best performances with Unity fingerprints; and both the k-means and NOSC methods performing well in the clustering of activity classes using Daylight fingerprints.

The results of this study also show that the number of clusters selected by OTPCLUS do not always provide the best results according to the QCI measure, this is the result of OPTCLUS selecting higher levels of hierarchy as the optimal choices, which means that molecules are less likely to be found in clusters that meet the requirements to be classified as active clusters. This decreased the value of p and q in the QCI calculation and increases the value of r, leading to lower QCI scores.

The disparity between the QCI values obtained for the different datasets is due to a combination of the number of actives contained within the activity class and the number of non-singleton clusters produced. For the 5HT1A dataset, the low QCI values can be attributed to the combined effect of the low ratio of active to inactive molecules in the activity class and the relatively large number of clusters produced by the NOSC algorithm. This leads to the small number of active molecules being dispersed across the large number of clusters, which in turn results in only a small number of clusters meeting the requirement to be classified as active clusters according to Varin et al. (2009), thus producing low QCI scores. Conversely, the larger QCI values for the Renin activity class are the result of the large number of active molecules, generally being distributed across a smaller number of clusters. ***Appendix B*** provides full results tables for the QCI evaluation for the performance of the NOSC algorithm using a variety of parameters.

4.7 Time and Storage Requirements

As previously stated in ***Section 3.5***, the implementation of a full matrix diagonalisation procedure typically requires $O(N^3)$ operations to complete when applied to densely populated matrices, such as those used in this study. This leads to a FMD having large associated time and storage costs, and limits the application of the NOSC algorithm - in its current form - to datasets where these costs remain manageable. Preliminary investigations showed that the maximum size dataset that could be clustered using the NOSC approach was approximately 4000 molecules.

The average time taken to apply a FMD procedure to filtered similarity matrices generated for the ChEMBL activity classes when using the Tanimoto coefficient and five different molecular descriptors is provided in ***Table 4.11***. Each average was calculated from 5 implementations of NOSC at γ = 25.

Dataset	Fingerprint	Average Time / s
5HT1A	BCI	1627
	Daylight	1637
	ECFP_4	1525
	MDL Public Keys	1502
	Unity	1592
MMP1	BCI	3300
	Daylight	3385
	ECFP_4	3204
	MDL Public Keys	3305
	Unity	3392
Renin	BCI	687
	Daylight	651
	ECFP_4	634
	MDL Public Keys	653
	Unity	680
SubP	BCI	1514
	Daylight	1498
	ECFP_4	1531
	MDL Public Keys	1465
	Unity	1594

Table 4-11: *The average time required to implement a full matrix diagonalisation procedure on filtered similarity matrices produced for each of the four ChEMBL activity classes when using the Tanimoto coefficient and each of the molecular fingerprint shown in the table.*

The time taken to diagonalise a matrix increases significantly with the value of N, for example, the time taken to decompose the Renin matrices, where N = 2166, is less than quarter of the time required to diagonalise the larger MMP1 matrices, where N = 3482. The sizeable difference in the time needed to identify the eigenpairs for each matrix is the result of the dramatic increase in the number of operations required to decompose each matrix, which is ≈ 1.0619×10^{10} for the Renin similarity matrices and ≈ 4.2217×10^{10} for those related to the MMP1 activity class.

Along with the value of N, both the choice of molecular fingerprint and the value of γ also have a minor effect on the time required to diagonalise a similarity matrix. These effects are the result of how the similarity values are distributed within each input matrix. When using large

values of γ, a number of elements are set to zero by the Gaussian filtering function and therefore are not operated on during the FMD procedure, speeding up its implementation. Likewise, the specificity of the fingerprint also dictates how many elements are set to zero by the Gaussian filtering function, leading to changes in the time requirements for FMD. For example, ECFP are highly discriminating and hence provide a larger number of zero elements than the other fingerprint types, leading to a decrease in the time requirements for FMD.

4.8 Conclusion

The investigation into the parameters that have an effect on the clusters generated using the NOSC algorithm produced several interesting conclusions. Experiments demonstrated that the specificity of the fingerprint type has a large effect on the clustering. The most specific fingerprint types, in general, produced a large number of small clusters containing molecules that share a high proportion of structural commonalities. Whereas, the least specific fingerprint types produced a smaller number of large globular clusters, which encompassed large numbers of molecules with varying molecular scaffolds due to the fingerprint's inability to distinguish between certain chemical features.

The parameterisation experiments show a relationship between the value of γ required to produce a sufficient clustering of a dataset and the type of molecular descriptor employed. For highly specific fingerprint types, such as ECFP_4 fingerprints, a low value of γ is required, whilst the more generic fingerprint types, such as MDL public keys, require the use of a significantly larger γ value to produce a favourable clustering of the dataset. Examining the use of the eigenvector threshold in the NOSC algorithm showed how this threshold was necessary to stop molecules from being placed in clusters to which they did not belong. Other conclusions that can be drawn from the results of these experiments are based upon the relationship that exists between the homogeneity of a dataset and the required value of the eigenvector threshold. A general trend exists such that as the homogeneity of a dataset is increased there is a requirement to decrease the value of the threshold to account for the lower average magnitude of the eigenvector elements.

The performance of the NOSC algorithm, when compared to both the Ward's and k-means algorithms using the QCI measure, shows that for the most specific fingerprint types, for example, extended connectivity fingerprints, spectral clustering is capable of producing significantly improved results over traditional clustering methods. However, for other fingerprints types the results are more comparable, with each algorithm providing similar

results across each fingerprint and dataset combination. It should also be noted that the performance of spectral clustering with highly unspecific fingerprints such as MDL Public Keys is poor, as the eigenvectors produced from these fingerprints do not contain enough variation to allow for effective clustering of the data.

The operational cost of implementing a FMD procedure results in the NOSC method having large time and storage costs, limiting its application to activity classes of 4000 compounds or less. These large associated costs prevent the NOSC algorithm from being considered a viable alternative to the more traditional clustering algorithms. However these cost issues can be minimised through the exploitation of matrix properties and other eigendecomposition algorithms that scale more favourably with N.

Chapter 5

Sparsity in Spectral Clustering

5.1 Introduction

In the previous chapter a non-overlapping spectral clustering algorithm was presented that utilised a full matrix diagonalisation procedure to identify the eigenpairs from a densely populated input matrix. Although this NOSC algorithm produced comparable – or in some cases improved – results when compared to the k-means and Ward's methods, it was limited to use with datasets of less than 4000 compounds due to its associated time and storage costs. One method that is commonly used to decrease the operational cost of the eigendecomposition step is moving from the use of a densely populated input matrix to a sparser format, which can be decomposed at a lower computational cost.

In publications by both Shi and Malik (2000) and Brewer (2007), a method to reduce the operational costs of spectral clustering by replacing the full matrix diagonalisation procedure with an eigensolver, more precisely a variant of the Lanczos algorithm, was mooted.

Eigensolver, such as the Lanczos algorithm, provide the maximum efficiency savings when applied to sparse matrices. In the case of dealing with large datasets, a matrix that is at least 95% sparse is preferable (Parlett, 1998). Where sparsity is defined as the percentage of the matrix elements that are populated by zero entries. Thus, the aim of this chapter was to test the use of a similarity value threshold as a method to minimise the density of the input matrices, whilst maintaining enough information to allow the compounds in an activity class to be clustered effectively.

5.2 A Modified Non-Overlapping Spectral Clustering Method

In this study, the density of an input matrix was controlled through the use of a user-defined similarity threshold. This threshold was applied to the elements of a filtered similarity matrix, such that any similarity value below the threshold was set to zero, with values above the threshold being left unchanged. By using a threshold in this manner, the user is able to control both the sparsity of the matrix and any loss of information by varying its magnitude. We note that this similarity value threshold differs from that applied to the eigenvectors, as the similarity threshold is applied to an input matrix prior to clustering, whereas the eigenvector threshold is applied to a matrix of eigenvectors calculated by the eigendecomposition step.

Along with the addition of a similarity threshold, the 95% positive eigenvalue rule was replaced with a method for selecting the k eigenpairs associated with the largest positive eigenvalues – where k is user-defined – to form eigenclusters from. The switch from the 95% positive eigenvalue rule was made to provide the user with more freedom to probe several different clustering levels for a dataset, for example, when using the parameter k, the distribution of molecules across 100, 200, 300 and 400 clusters can simply be assessed, whereas the 95% positive eigenvalue rule limits the application to a single clustering level determined by the rule itself.

Thus, a modified version of the NOSC algorithm (m-NOSC) was implemented using the following steps:

- Generate a filtered similarity matrix, S^F, using the method outlined in ***Chapter 4***.
- Form a sparse matrix, A, by applying a similarity threshold to the elements of the matrix S^F.
- Diagonalise matrix A to find the complete set of eigenpairs for the matrix.
- Identify the eigenpairs related to the k largest eigenvalues.

- Apply the eigenvector threshold to the elements of the k eigenvectors.
- Assign molecules to the eigencluster to which they make the largest contribution.

5.3 The Effect of the Similarity Threshold on Matrix Sparsity

The first step in this study was to form filtered similarity matrices from four ChEMBL datasets (see ***Section 4.3*** for details), using the Tanimoto coefficient, five different types of molecular fingerprint (see ***Section 4.4***) and four values of γ (25, 50, 75 and 100). Six similarity thresholds were investigated (0.1, 0.01, 0.001, 0.0001, 0.00001 and 0.000001) and the sparsity of each resultant matrix was quantified as a percentage of the matrix elements. ***Tables 5.1 – 5.3*** and ***5.5*** show the results.

γ	Similarity Threshold	Sparsity / %				
		BCI	Daylight	ECFP	MDL Public Keys	Unity
25	0.1	99.08	99.11	99.78	94.06	98.87
	0.01	98.27	98.40	99.49	67.22	96.94
	0.001	95.43	96.76	99.16	33.16	89.64
	1.00E-04	84.93	91.18	98.65	12.11	71.84
	1.00E-05	59.74	77.28	97.84	3.37	41.68
	1.00E-06	29.55	50.98	95.84	0.79	13.08
50	0.1	99.42	99.44	99.91	98.54	99.32
	0.01	99.08	99.11	99.80	94.06	98.87
	0.001	98.75	98.79	99.66	83.28	98.25
	1.00E-04	98.27	98.40	99.49	67.22	96.94
	1.00E-05	97.31	97.79	99.33	49.55	94.34
	1.00E-06	95.43	96.76	99.16	33.16	89.64
75	0.1	99.56	99.56	99.93	99.13	99.49
	0.01	99.29	99.33	99.88	97.62	99.17
	0.001	99.08	99.11	99.80	94.06	98.87
	1.00E-04	98.87	98.90	99.71	87.63	98.50
	1.00E-05	98.63	98.67	99.60	78.44	97.92
	1.00E-06	98.27	98.40	99.49	67.22	96.94
100	0.1	99.64	99.63	99.94	99.35	99.59
	0.01	99.42	99.44	99.91	98.54	99.32
	0.001	99.24	99.28	99.86	96.98	99.09
	1.00E-04	99.08	99.11	99.80	94.06	98.87
	1.00E-05	98.92	98.95	99.73	89.33	98.60
	1.00E-06	98.75	98.79	99.66	83.28	98.25

Table 5-1: *The sparsity of the matrices produced for the 5HT1A activity class.*

γ	Similarity Threshold	Sparsity / %				
		BCI	Daylight	ECFP	MDL Public Keys	Unity
25	0.1	98.43	99.24	99.77	89.02	97.74
	0.01	95.27	98.07	99.41	65.48	92.53
	0.001	89.79	95.09	98.74	37.90	81.84
	1.00E-04	79.46	89.22	97.22	16.73	60.46
	1.00E-05	60.94	76.94	94.08	6.36	32.43
	1.00E-06	37.34	51.89	85.82	1.83	13.24
50	0.1	99.29	99.59	99.90	97.10	99.17
	0.01	98.43	99.24	99.77	89.02	97.74
	0.001	97.14	98.78	99.61	78.33	95.51
	1.00E-04	95.27	98.07	99.41	65.48	92.53
	1.00E-05	92.86	96.91	99.14	51.76	88.29
	1.00E-06	89.79	95.09	98.74	37.90	81.84
75	0.1	99.52	99.70	99.94	98.66	99.46
	0.01	99.04	99.47	99.86	94.78	98.80
	0.001	98.43	99.24	99.77	89.02	97.74
	1.00E-04	97.63	98.95	99.67	82.19	96.33
	1.00E-05	96.57	98.58	99.55	74.20	94.61
	1.00E-06	95.27	98.07	99.41	65.48	92.53
100	0.1	99.63	99.75	99.95	99.15	99.58
	0.01	99.29	99.59	99.90	97.10	99.17
	0.001	98.90	99.41	99.84	93.54	98.57
	1.00E-04	98.43	99.24	99.77	89.02	97.74
	1.00E-05	97.85	99.03	99.69	83.99	96.72
	1.00E-06	97.14	98.78	99.61	78.33	95.51

Table 5-2: *The sparsity of the matrices produced for the MMP1 activity class.*

The results of this study show that the sparsity of the matrices formed from the 5HT1A, SubP and MMP1 activity classes varies according to both the value of γ and the choice of molecular descriptor. For low values of γ the spread of the filtered similarity values means that the choice of magnitude for the similarity threshold has a large effect on the sparsity of the matrix. For example, when using BCI fingerprints to describe the 5HT1A at γ = 25, decreasing the threshold value by a factor of 10 decreases the sparsity of the matrix from 98.43% to 95.27% and the trend continues down to a sparsity value where the matrix could be considered densely populated when the threshold is 1×10^{6}. Alternatively at larger γ values the choice of magnitude for the similarity threshold has a much smaller effect on the sparsity of the matrices formed, such that relaxing the threshold by a factor of 10 decreases the sparsity of the matrix by less than 1% for most molecular descriptors. This effect can be better understood by considering ***Figure 5.1***, which shows the distribution of similarity values for the

SubP activity class along with the distribution of filtered similarity values generated when γ = 25. This figure shows that the distribution of similarity values for each fingerprint type is shifted towards the smaller values by the filtering function. When γ is small, for example, less than 50, the number of similarity values below 0.1 is significantly less than when γ is large, for example, 100. Therefore at low values of γ, decreasing the threshold, for example, from 0.1 to 0.01, increases the density of the input matrix by allowing a significant number of additional matrix elements to be considered. Conversely, when using large γ values, lowering the similarity threshold has a minimal effect on the number of additional matrix elements that can be considered within the eigendecomposition step, as the majority of the similarity values remain significantly below the threshold.

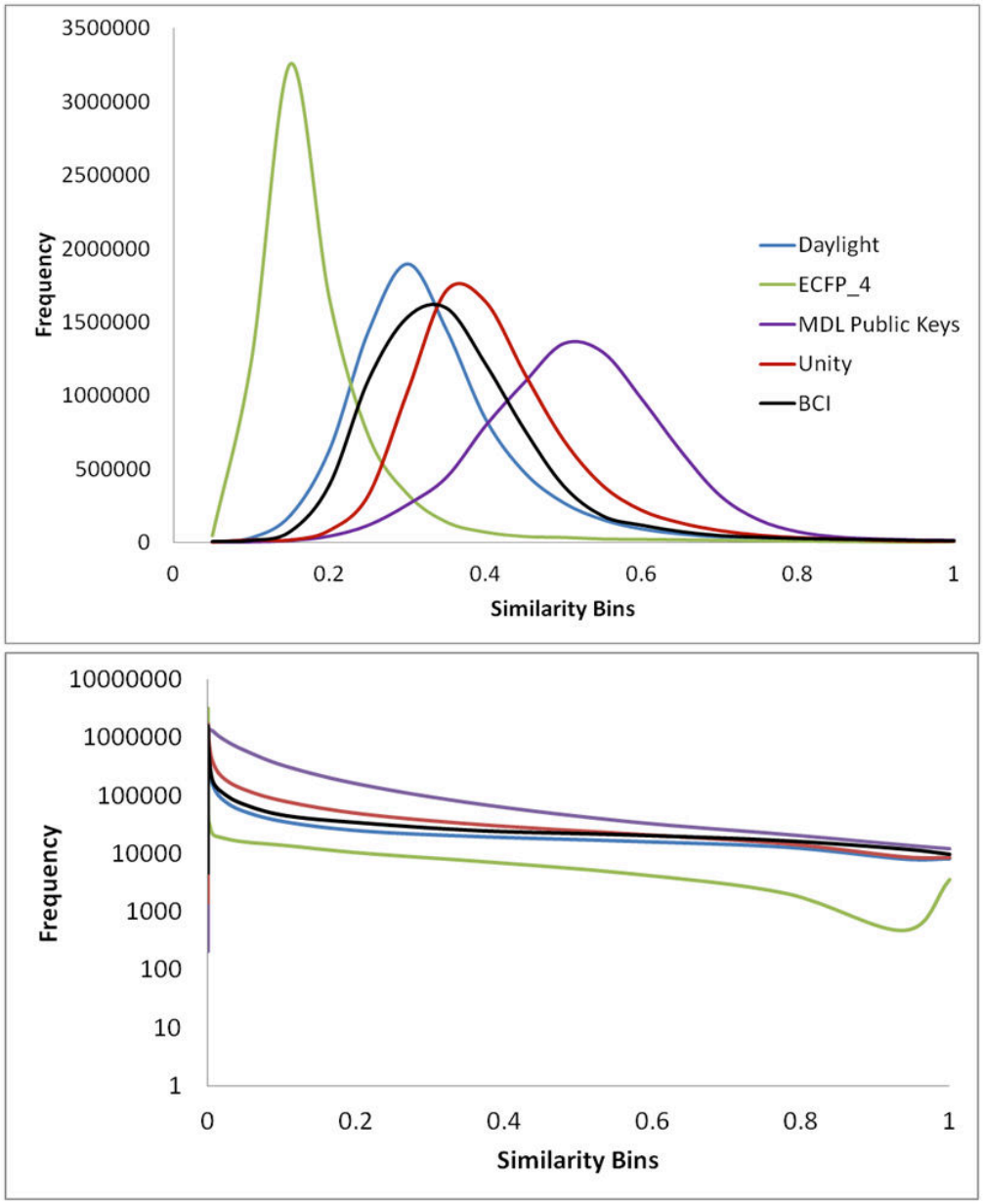


Figure 5-1: *Effect of the Gaussian filtering function. (a) Distribution of similarity scores in the SubP dataset. (b) Distribution of the similarity scores in the SubP dataset, when γ = 25.*

γ	Similarity Threshold	Sparsity / %				
		BCI	Daylight	ECFP	MDL Public Keys	Unity
25	0.1	98.40	98.74	99.62	93.98	97.92
	0.01	96.10	96.74	99.09	65.97	92.73
	0.001	90.79	91.81	98.49	30.25	78.78
	1.00E-04	76.20	80.24	97.67	10.46	49.60
	1.00E-05	50.30	55.85	96.01	2.99	16.32
	1.00E-06	23.09	23.39	91.44	0.60	2.92
50	0.1	99.13	99.31	99.83	98.40	99.09
	0.01	98.40	98.74	99.62	93.98	97.92
	0.001	97.50	97.95	99.36	83.19	95.94
	1.00E-04	96.10	96.74	99.09	65.97	92.73
	1.00E-05	94.08	94.81	98.82	47.09	87.43
	1.00E-06	90.79	91.81	98.49	30.25	78.78
75	0.1	99.36	99.49	99.89	99.04	99.39
	0.01	98.90	99.12	99.76	97.43	98.77
	0.001	98.40	98.74	99.62	93.98	97.92
	1.00E-04	97.84	98.25	99.45	87.62	96.71
	1.00E-05	97.09	97.60	99.27	78.08	95.04
	1.00E-06	96.10	96.74	99.09	65.97	92.73
100	0.1	99.49	99.59	99.92	99.27	99.53
	0.01	99.13	99.31	99.83	98.40	99.09
	0.001	98.78	99.03	99.73	96.80	98.59
	1.00E-04	98.40	98.74	99.62	93.98	97.92
	1.00E-05	97.99	98.38	99.49	89.38	97.05
	1.00E-06	97.50	97.95	99.36	83.19	95.94

Table 5-3: *The sparsity of the matrices produced for the SubP activity class.*

The relationship between the choice of similarity threshold and sparsity of the matrix produced for different fingerprints types are reflective of each fingerprint's ability to discriminate between molecules. For highly discriminating fingerprints such as ECFPs, varying the magnitude of the threshold has a minimal effect on the sparsity, as the distribution of filtered similarity values leads to the inclusion of a only a few of additional similarity values when the threshold is relaxed. For the less specific fingerprint types (Daylight, Unity and BCI) varying the value of similarity threshold only has a large effect on the sparsity of the matrices they produce when γ is low. In the case of MDL public keys, varying the threshold value continues to have a larger effect on the sparsity of the input matrices, for all values of γ, reflecting the difficulty that MDL public keys have in discriminating between molecules that share a large number of scaffold features. These observations are further illustrated by viewing the distribution of filtered similarity values for each fingerprint type (see ***Table 5.4***). The most

specific fingerprint types – ECFP and Daylight – produce a greater number of filtered similarity values under 0.001, representing pairs of molecules that share little similarity, whereas the more generic fingerprint types struggle to make this distinction and as a result produce significantly more filtered similarity scores above 0.01.

Bin	BCI	Daylight	ECFP_4	MDL Public Keys	Unity
< 0.001	10602474	11353354	11931138	3664998	9300144
0.0010	283888	175308	40324	929730	621868
0.0015	144906	87528	18924	620996	307858
0.0020	95880	55314	14388	443352	191284
0.0025	63906	39658	6512	244570	145748
0.0030	53888	30162	6404	305436	108334
0.0035	43552	23498	5252	224504	84156
0.0040	36830	19234	4038	184102	69348
0.0045	30798	16536	3928	176834	58440
0.0050	27324	13694	3304	157536	50342
0.0055	24598	11998	2618	134616	43010
0.0060	23080	10952	2934	130258	38526
0.0065	19540	9244	2186	109344	33692
0.0070	16468	8142	2142	120544	30072
0.0075	17922	7334	1792	86074	27806
0.0080	14476	6656	1424	98214	25602
0.0085	14118	6226	1470	85244	23000
0.0090	13098	5488	1604	82758	21866
0.0095	12260	5270	1428	85630	20198
0.0100	11390	4586	1152	54782	17610
> 0.01	573928	234142	71362	4184802	905420

Table 5-4: *Distribution of filtered similarity scores for the SubP activity class when γ = 25.*

The results for the Renin dataset (***Table 5.5***) are similar to those obtained for the other activity classes, with the exception that the high homogeneity of the Renin activity class leads to a smaller spread of filtered similarity values; therefore more similarity scores exceed the threshold value, leading to matrices that are more densely populated. This means that a larger similarity threshold is required to produce matrices that are at least 95% sparse.

γ	Similarity Threshold	Sparsity / %				
		BCI	Daylight	ECFP	MDL Public Keys	Unity
25	0.1	92.93	95.73	99.23	82.27	90.79
	0.01	75.28	88.25	97.51	38.75	68.55
	0.001	46.74	74.39	94.07	10.78	37.74
	1.00E-04	26.72	51.08	87.65	2.21	15.73
	1.00E-05	18.05	30.25	75.06	0.37	3.65
	1.00E-06	9.68	11.71	58.49	0.04	1.09
50	0.1	97.40	98.25	99.70	95.22	96.68
	0.01	92.93	95.73	99.23	82.27	90.79
	0.001	85.64	92.36	98.51	61.72	81.62
	1.00E-04	75.28	88.25	97.51	38.75	68.55
	1.00E-05	61.58	82.62	96.11	21.56	53.02
	1.00E-06	46.74	74.39	94.07	10.78	37.74
75	0.1	98.40	98.87	99.82	97.49	98.04
	0.01	96.22	97.54	99.57	91.90	94.98
	0.001	92.93	95.73	99.23	82.27	90.79
	1.00E-04	88.38	93.54	98.77	69.14	85.14
	1.00E-05	82.56	91.10	98.20	54.02	77.70
	1.00E-06	75.28	88.25	97.51	38.75	68.55
100	0.1	98.82	99.14	99.87	98.28	98.60
	0.01	97.40	98.25	99.70	95.22	96.68
	0.001	95.50	97.15	99.49	89.98	94.06
	1.00E-04	92.93	95.73	99.23	82.27	90.79
	1.00E-05	89.64	94.12	98.90	72.42	86.71
	1.00E-06	85.64	92.36	98.51	61.72	81.62

Table 5-5: *The sparsity of the matrices produced for the Renin activity class.*

This study has shown that, in general, by using a similarity threshold value between 0.01 and 0.001, similarity matrices that are at least 95% sparse can be formed from the four activity classes used in this study. MDL public keys provide an exception to this finding as their generic nature means that they require a significantly lower threshold in order to produce highly sparse matrices. This finding is key to minimising the storage costs of spectral clustering, as these highly sparse matrices are significantly smaller than their dense counterparts. This result is also vital to further studies in this thesis, as it allows the full matrix diagonalisation procedure to be replaced with an eigensolver in order to provide a more efficient eigendecomposition step in the spectral clustering approach.

5.4 Clustering using Sparse Similarity Matrices

After identifying an appropriate similarity threshold, the next step was to ensure that the sparse matrices do not have a detrimental effect on the performance of the m-NOSC algorithm in clustering activity classes. Thus, the m-NOSC algorithm was applied to the four ChEMBL datasets used in ***Chapter 4*** using the parameters outlined in ***Table 4.9***, a similarity threshold value of 0.001 and an identical number of clusters for each implementation as shown in ***Table 4.10***. The QCI scores produced were compared to the results obtained using the NOSC algorithm to identify how the performance varies between the two methods, see ***Table 5.6***

Dataset	Fingerprint Type	NOSC		m-NOSC	
		# Clusters	QCI	# Clusters	QCI
5HT1A	BCI	391	34.069	391	29.916
	Daylight	460	35.244	460	29.246
	ECFP_4	604	57.404	604	56.897
	MDL Public Keys	181	31.061	181	29.734
	Unity	401	32.294	401	29.797
MMP1	BCI	308	71.049	308	72.252
	Daylight	369	70.784	369	72.647
	ECFP_4	631	81.701	631	82.701
	MDL Public Keys	143	66.168	143	65.665
	Unity	280	73.384	280	74.821
Renin	BCI	228	60.675	228	64.713
	Daylight	325	61.023	325	62.630
	ECFP_4	542	83.125	542	83.578
	MDL Public Keys	131	72.747	131	71.212
	Unity	220	56.659	220	60.150
SubP	BCI	262	59.043	262	61.604
	Daylight	237	57.786	237	60.488
	ECFP_4	450	55.840	450	56.296
	MDL Public Keys	166	46.300	166	49.825
	Unity	233	56.959	233	57.203

Table 5-6: *A comparison of the respective performances of the NOSC and m-NOSC algorithms in clustering four ChEMBL activity classes.*

The results show that, in general, the use of the m-NOSC method leads to a small rise in the algorithm's performance according to the QCI measure. This general increase is the result of the sparsity threshold aiding in the removal of low similarity molecules from the clusters, which in-turn decreases the values of q (the number of inactive in active clusters) and r (the number of actives in inactive clusters) in the QCI measure, hence leading to greater QCI scores. However there are several exceptions to the general trend, for example, the m-NOSC method

provides a smaller QCI score in each application to the 5HT1A activity class. The decrease in QCI values for the 5HT1A dataset is the result of the small number of active molecules contained in the activity class. Therefore, the removal of a single active molecule from the clustering leads to a significant fall in the QCI score for the dataset. Other exceptions to this general trend are related to the use of MDL Public Keys, which produce smaller QCI scores as the similarity threshold leads to a number of molecules being excluded from the clustering, including several active molecules.

Another important consideration in this study to note, is if the switch from using the 95% eigenvalue rule has a negative impact on how molecules are assigned to clusters. For example, does the introduction of eigenpairs that would not be considered when using the 95% positive eigenvalue rule lead to the mis-assignment of molecules? To ascertain the answer to this question, a visual inspection of the clusters related to eigenpairs, that although related to the k eigenpairs with the largest eigenvalues, would fail to meet the requirements of the 95% positive eigenvalue rule, was undertaken in order to identify if the clusters remain chemically intuitive (see ***Figure 5.2***). This diagram shows a set of molecules based on the same basic chemical scaffold that were clustered together by the m-NOSC algorithm when k was set to 50. As the value of k is increased the molecules containing an additional naphthalene ring system were removed from the large cluster and placed into a new cluster, these clusters were broken down into 4 clusters based upon different chemical relationship as k was raised to 200. These results not only show that the use of clusters that are not considered by the 95% positive eigenvalue rule is not a problem, but that there is a pseudo-hierarchical relationship between some clusters.

Finally, comparing the time requirements of both the NOSC and m-NOSC methods shows that the two methods have nearly identical time costs. This means that although the use of sparse input matrices leads to a reduction in the storage costs it is unable to address the significant time costs of the full matrix diagonalisation procedure.

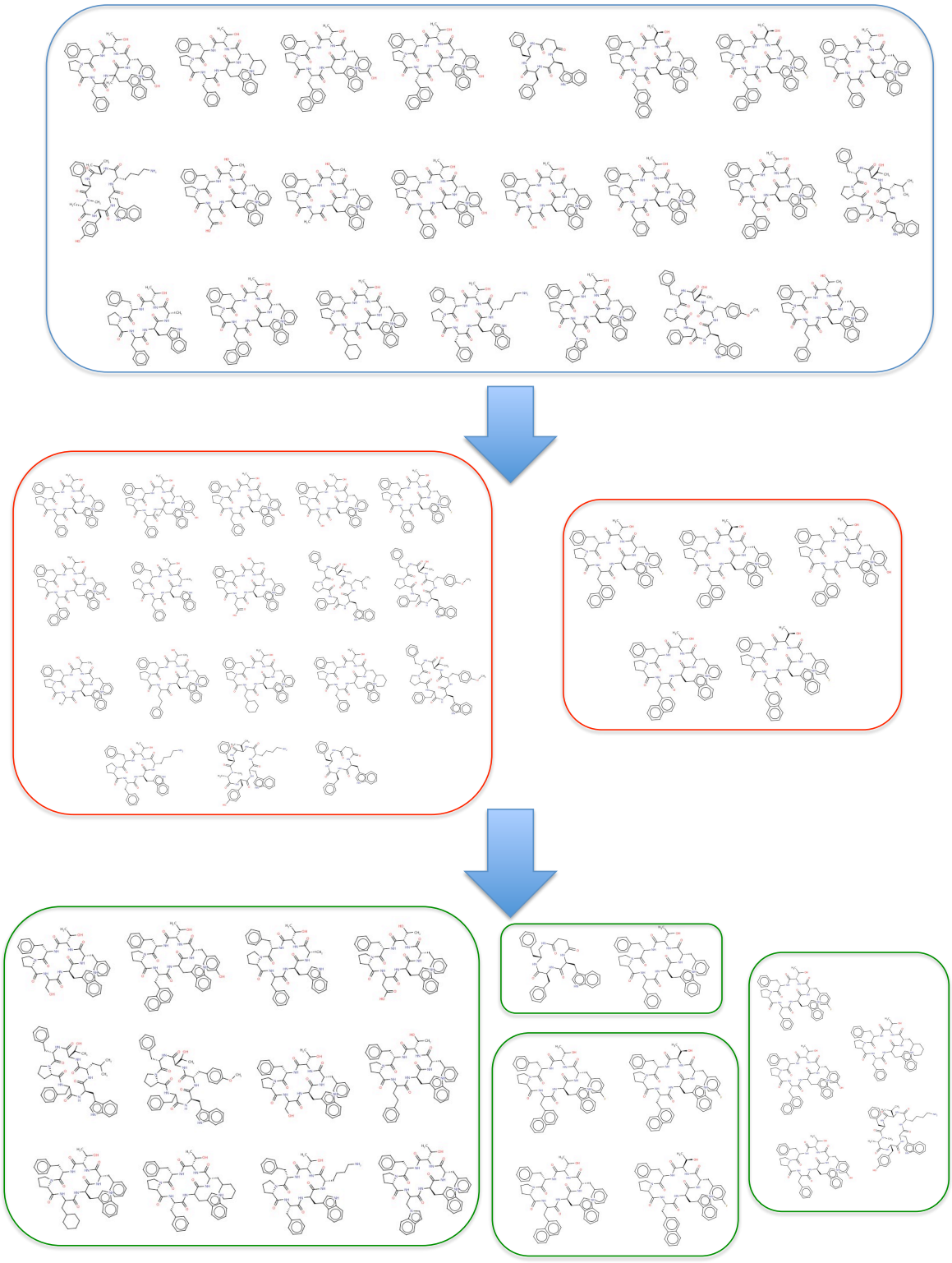


Figure 5-2: *A cluster related to an eigenpair that would not meet the requirements to be selected by 95% positive eigenvalue rule, but used in clustering by the m-NOSC algorithm. The blue box shows the cluster obtained when k = 50, the red boxes shows its breakdown when k = 100 and the green boxes when k = 200. This cluster was selected from the results obtained for the 5HT1A activity class described by ECFPs, when γ = 25.*

5.5 Conclusion

This study has shown that the sparsity of a matrix can be easily controlled through the application of a similarity threshold. In general, a threshold score of between 0.001 and 0.0001 is sufficient to produce matrices that are at least 95% sparse, decreasing the storage costs of spectral clustering and increasing the maximum sized dataset that it can be applied to. Furthermore, the ability to produce matrices that were highly sparse is also vitally important to ensuring that eigensolvers, such as the Lanczos algorithm, can be efficiently implemented within further studies. The comparison between the m-NOSC and NOSC algorithms shows that the switch to using sparsely populated input matrices has a minimal effect on the results of spectral clustering chemical data. Unfortunately, despite these results the time costs of this method still remained unchanged. Therefore, in the next chapter, the full matrix diagonalisation procedure (used in ***Chapters 4*** and ***5***) will be replaced with the more efficient Lanczos algorithm.

Chapter 6

Lanczos-based Spectral Clustering

6.1 Introduction

The requirement to quickly identify eigenpairs from large sparsely populated matrices is common to a number of fields (Parlett, 1998). This need has led to the investigation of several different algorithms for their ability to identify eigenpairs. In publications by Shi and Malik (2000) and Brewer (2007), one such method is recommended for use in decreasing the operational costs of spectral clustering, this method is called the Lanczos algorithm (Lanczos, 1950).

The Lanczos algorithm (see ***Section 6.3***) is a method of approximating the eigenpairs from a matrix by exploiting an intermediate form (see ***Section 6.2***). This method is routinely used to approximate a "few" eigenpairs for a variety of tasks, including web searching algorithms (Page et al., 1999) and image analysis (Nguyen et al., 2001). Although there appears to be little evidence to suggest what an empirical value for the term "few" is within most studies that

utilise this method. Furthermore, the Lanczos algorithm is afflicted by issues relating to its mathematical stability and as a result must be monitored to ensure it does not succumb to these issues.

The aim of this chapter is to present an alternative spectral clustering algorithm that utilises a modified version of the Lanczos procedure, rather than a full matrix diagonalisation, to produce an algorithm that is both faster than full matrix diagonalisation and more scalable; whilst maintaining a high level of accuracy in the results.

6.2 Tridiagonalisation

The most stable and easy to implement algorithms for identifying eigenpairs from a symmetric matrix are based on a full matrix diagonalisation procedure (see ***Section 3.1***); where the term stability refers to the extent that an algorithm is affected by the presence of roundoff errors (Golub and Van Loan, 1996; Strang, 2003). Unfortunately the full diagonalisation of a matrix is both computationally expensive and time consuming, typically requiring $O(N^3)$ operations, for an NxN matrix, which limits its scope. Although it should be noted that computational cost savings can be made by ensuring the input matrix is sparsely populated (as illustrated in ***Chapter 5***).

In practice the optimum strategy for computing eigenpairs is to reduce the matrix to a simple form, before applying an iterative procedure, sometimes known as an eigensolver, to identify the eigenpairs. When using symmetric matrices, either fully or sparsely populated, one of the preferred simple forms is a tridiagonal matrix (Press et al., 2007). A tridiagonal matrix is a reduced matrix form in which non-zero values are only contained within the elements of the main diagonal and the first off-diagonals (see below for an example). It should be noted that the tridiagonal matrices used in this study are always symmetric.

1	2	0	0	0	0
2	4	1	0	0	0
0	1	5	3	0	0
0	0	3	3	1	0
0	0	0	1	2	2
0	0	0	0	2	1

A tridiagonal matrix, T, is used as a simple representation of any symmetric matrix, A, as it provides a number of advantages, including:

- The eigenpairs of T can be elucidated in significantly fewer arithmetic operations than are required for A.
- Every A can be reduced to T in a finite number of elementary orthogonal transformations, whereas, in principle a full diagonalisation of A can require an infinite number of transformations (Golub and Van Loan, 1996; Parlett, 1998).

6.2.1 Reducing a Symmetric Matrix to a Tridiagonal Matrix

There are several different approaches that can be used to reduce a symmetric matrix to its tridiagonal form (see ***Figure 6.1***). The most commonly used tridiagonalisation procedures can be split into two categories:

- Direct methods, which are those procedures that harness a complex set of transformations and/or rotations to eliminate the non-zero off-diagonal elements of a matrix but must be run to completion in order to identify the eigenpairs of a matrix, for example, the Householder (1958) & Givens (1954) methods.
- Projection methods, which are algorithms that approximate the diagonal/first off-diagonal elements of a tridiagonal matrix based on the exploitation of Krylov subspaces (Parlett, 1998) these methods include the Lanczos algorithm (Lanczos, 1950) and the Power method (Parlett, 1998).

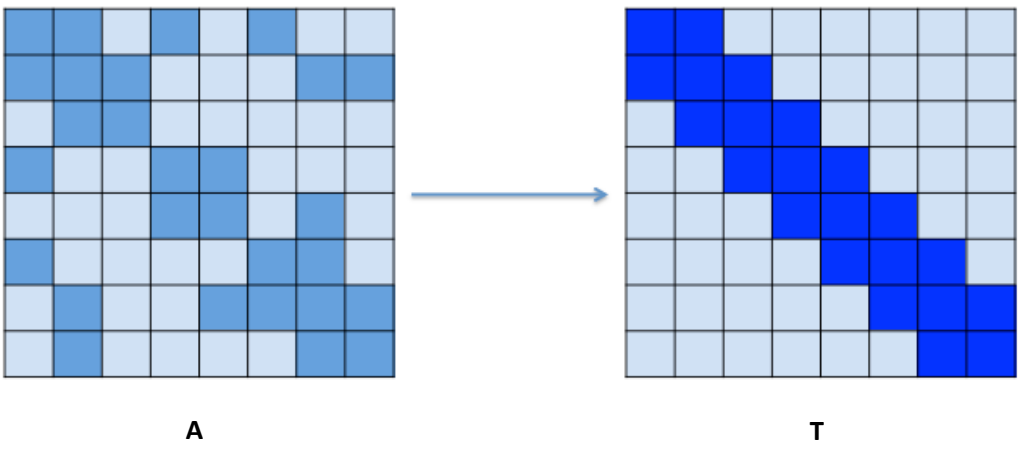


Figure 6-1: *Idealised tridiagonalisation of a sparse symmetric matrix. The light blue elements represent zero elements and the darker blue elements give the non-zero elements.*

For the purpose of this chapter, the discussion of tridiagonalisation procedures will be limited to the Lanczos algorithm, including its background, implementation and optimisation.

6.3 The Lanczos Algorithm

Since its initial proposal by Cornelius Lanczos in 1950 (Lanczos, 1950), the Lanczos algorithm has had a varied history. Initially the Lanczos algorithm was lauded as a simple way to reduce a whole matrix to its tridiagonal form, however it failed to fulfil its initial promise, due to its susceptibility to roundoff error and issues that occur when it is used in finite precision arithmetic problems, i.e., finding the eigenpairs from vectors comprised of a large number of decimal values (Parlett, 1998). Twenty years on, Paige showed that despite its mathematical instability, the simple Lanczos algorithm was still an effective tool for the computation of a low number of eigenpairs of a matrix (Paige, 1971); which led to a considerable amount of research being undertaken into improving the performance of the algorithm.

One of the most common implementations of the Lanczos algorithm uses an iterative procedure based on two major steps per iteration, to identify k of the eigenpairs from a matrix A. In the initial step, the Lanczos algorithm is applied to matrix A, identifying the diagonal elements, α, and the first off diagonal elements, β, associated with each of the k first-to-converge eigenvalues along with a set of Lanczos vectors. It is important to understand that the value of k that is input to the Lanczos method specifies the number of eigenpairs that is calculated by the algorithm and that each of the k eigenpairs can be related to either positive or negative eigenvalues depending on which eigenvalues are converged upon first. Hence, k can be divided into two sets; the p set, which are the elements associated with the positive eigenvalues, which form eigenclusters, and the neg set, that is related to the negative eigenvalues that do not form eigenclusters. The second step in this process is the use of an iterative solver to identify the k eigenpairs of A from the elements of T and the Lanczos vectors. ***Figure 6.2*** provides a diagrammatic overview of this process in which the iterative solver used is a variant of the QL decomposition, discussed further in ***Section 6.3.5***.

Sections 6.3.1 – 6.3.5 provide a brief explanation of the mathematical tools which underpin the application of the Lanczos algorithm; the steps of the Lanczos algorithm; the associated issues and their solutions; and a further discussion of both the QL and QR procedures.

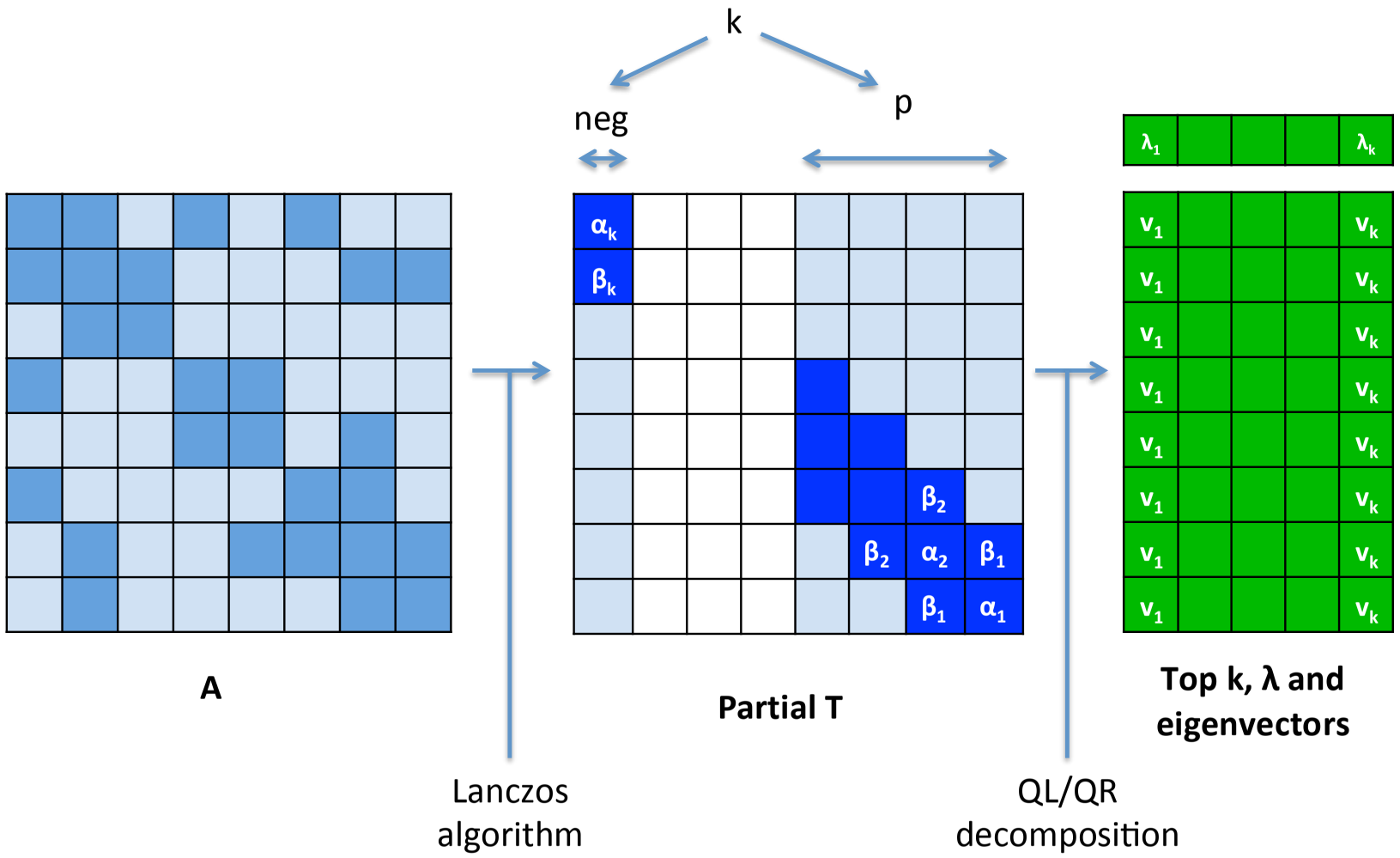


Figure 6-2: *Identifying the top k eigenpairs of matrix A, using the Lanczos algorithm coupled with a QL decomposition. The light blue elements represent zero elements, the elements shown in white are those for which eigenpairs will not be calculated and finally the darker blue elements represent the non-zero elements.*

6.3.1 Krylov Subspaces and the Ritz-Rayleigh Procedure

In linear algebra, a Krylov subspace is a theoretical space that is defined as the order-r subspace spanned by the projections of a vector, q, onto the first r columns of an n x n matrix, A, and is commonly given in the form (Parlett, 1998):

Equation 25: *Krylov subspace formula.*

$$\kappa_r(A, q) = (q, Aq, \dots A^{r-1}q)$$

Krylov subspaces are of considerable mathematical importance as they are central to the theory behind a number of methods used to compute some, or all, of the eigenpairs of A, including the Ritz-Rayleigh procedure.

The Ritz-Rayleigh procedure is a method of approximating the eigenpairs of A. In the initial steps of this procedure the columns of A are orthonormalised in order to determine a basis for a Krylov subspace. The subsequent steps generate a Rayleigh quotient matrix, Q, and a number of approximations for the eigenvalues (Ritz values) and eigenvectors (Ritz vectors). The best approximations for the eigenpairs, known as the Ritz pairs, are determined from the Krylov subspace and the accuracy is assessed via the orthogonalisation of the Ritz vectors against vectors from the quotient matrix, Q, producing a residual vector. Taking the norm of this residual vector gives the accuracy of the approximations (Parlett, 1998).

6.3.2 The Simple Lanczos Algorithm

The simple Lanczos algorithm, which is also known as the symmetric Lanczos algorithm, can be presented in a number of ways, but is most commonly presented as a simple and effective way of iteratively implementing the *Ritz-Rayleigh* procedure onto a sequence of *Krylov subspaces*, in a six step procedure (avoiding the costly factorisation procedure required in other implementations):

For each iteration;

1. $q_i = \begin{cases} \|r_i\| & i = 0 \\ \frac{r_{i-1}}{\beta_{i-1}} & i > 0 \end{cases}$

The initial step in this algorithm is effectively a normalisation of a random vector r, producing vector q. Where, if i = 0, then q is found by taking the square-root of vector

r's square (this is also known as taking the two norm of the vector), otherwise, if $i > 0$, q is found by using the equation given above where β_i is a first off-diagonal element.

2. $u_i = Aq_i$

In this next step, multiplying the input matrix A by the vector q_i, which is generated in the first step, generates a vector u_i.

3. $r_i = u_i - \beta_{i-1} * q_{i-1}$

The random vector is then updated using the above equation. It should be noted that if $i = 0$, the random vector remains unchanged.

4. $\alpha_i = q_i \, . \, r_i$

Next, the diagonal element α_i is approximated by taking the scalar product of the vectors $q_i$and r_i.

5. $r_i = r_i - q_i * \alpha_i$

Again, the random vector is again recalculated using equation 5.

6. $\beta_i = \|r_i\|$

Step 6, produces a final approximation for the first off diagonal element β_i by taking the two norm of the vector r_i.

The tridiagonal elements α and β, along with the Lanczos vectors, q, are subsequently passed forward into an iterative program that identifies the top k eigenpairs of the input matrix. It should also be noted that, according to the Kaniel-Saad theorems (Kaniel, 1966) the approximations for the outermost eigenpairs should improve as the number of iterations increases, as they are constantly refined during each iteration.

6.3.3 A Loss of Orthogonality

When the Lanczos algorithm is applied to a problem in finite precision mathematics, "*during its first few iterations, sometimes three, other times as many as thirty,"* (Parlett, 1998) the algorithm produces results that are indistinguishable to those calculated by the exact process.

This continues, until a new Lanczos vector, q, is calculated that is not orthogonal, to working precision, to its predecessors. After a few more steps, the roundoff errors are compounded such that each of the new Lanczos vectors generated are linearly dependent on those that precede it. This in turn leads to the approximation of new incorrect eigenvalues due to the linear dependence of the vectors (Golub and Van Loan, 1996). Furthermore, problems continue to occur as the algorithm begins to recalculate the largest eigenvalues, leading to the calculation of both degenerate eigenvalues and associated eigenvectors that are multiples of previous vectors. The end result is that the Lanczos algorithm in this form is not suitable for use in finding more than the outer most eigenpairs of the matrix A (Parlett, 1998).

6.3.4 Overcoming the Issues with Reorthogonalisation

In order to overcome the issues of a mutual loss of orthogonality between the Lanczos vectors, a reorthogonalisation procedure can be used. This reorthogonalisation of the vectors can be complete or partial (based on a selective parameter), with each solution having a notable algorithmic cost in terms of the number of operations required to carry out the reorthogonalisation.

6.3.4.1 Full Reorthogonalisation

It was Lanczos himself who proposed a full reorthogonalisation of the Lanczos vectors based on the Gram-Schmidt reorthogonalisation procedure as the original fix for this problem. The Gram-Schmidt process is a method for reorthogonalising a set of vectors to span the same n-dimensional subspace, using the equation:

Equation 26: *Gram-Schmidt reorthogonalisation procedure.*

$$u_n = v_n - \sum proj_{u_n}(v_n)$$

Where:

u is an orthogonal vector

v is an input vector

proj is the projection of v on to u and is given by $\frac{\langle v \,.\, u \rangle}{\langle u \,.\, u \rangle}$

n is the number of vectors.

The Gram-Schmidt reorthogonalisation can be applied to the Lanczos algorithm as an additional step using the following equation:

Equation 27: *Equation for applying the Gram-Schmidt reorthogonalisation to Lanczos algorithm.*

$$r_{j(orth)} = r_j - \sum q_i(q_i * r_j)$$

Where;

r is a modified vector from the Lanczos algorithm

j is the current Lanczos step

q is a modified vector generated during the Lanczos algorithm

i is an iterator going from 0 to j.

Using the equation given above the Gram-Schmidt process reorthogonalises each new vector against all previous vectors by subtracting their non-orthogonal components at a notable computational operation cost. For example, at the third iteration, the reorthogonalisation process is given by the equation:

Equation 28: *Example of the Gram-Schmidt equation for the third iteration of the Lanczos algorithm.*

$$r_{3(orth)} = r_3 - q_1(q_1 * r_3) - q_2(q_2 * r_3) - q_3(q_3 * r_3)$$

6.3.5 The QL & QR Algorithm

After a matrix has been reduced to its tridiagonal form, an iterative procedure is required to identify its associated eigenpairs. The basic premise behind the use of a QL (or QR) algorithm is that any real symmetric matrix, A, can be decomposed into the form:

Equation 29: *QL formula.*

$$A = Q \,.\, L$$

Where A is a general matrix, Q is an orthogonal rotation matrix and L is the left/lower triangle of a tridiagonal matrix. Using this premise, a QL/QR algorithm is applied in order to reduce the off-diagonal elements swiftly until they are negligible. This is achieved through the repeated application of a complex similarity transformation to the matrix, thereby computing a series of intermediate matrices, which eventually converge to a diagonal matrix. The general process is outlined below:

1. Identify both Q and L.
2. Generate A_1 = QL.

3. Identify Q_1 and L_1.
4. Generate $A_2 = Q_1L_1$.
5. Continue until completion.

This procedure has a workload of $O(n^3)$ per iteration when applied to a full matrix, however these computational costs are reduced to O(n) when dealing with a tridiagonal matrix. An implicit form of this algorithm, sometimes known as the QL/QR algorithm with implicit shifts, can be used to gain further computational advantages through the application of Householder transformations to each row of A (Press et al., 2007).

6.4 A Java Implementation of Lanczos-Based Spectral Clustering

The m-NOSC algorithm discussed in ***Chapter 5*** has been re-implemented here with two notable exceptions:

- A Lanczos decomposition algorithm is used in preference to a full matrix diagonalisation when finding the eigenpairs of A.
- In the Lanczos-based method, the number of eigenpairs that is identified is specified on input, rather than calculating a complete set of eigenpairs and then selecting a subset, as is the case when using a full diagonalisation-based method.

A Lanczos decomposition class was implemented in java based on the COLT matrix package (CERN, 2011) that was modified to generate reproducible results by setting the random number seed to a constant. This class was subsequently used within a spectral clustering program.

6.4.1 Applying the Spectral Clustering Algorithm

In our initial experimentations, the Lanczos-based spectral clustering approach was used to cluster the Stahl-COX2 inhibitor set of 125 molecules used within ***Chapter 4***. Unfortunately, the results produced by this study exhibited many of the issues described in ***Section 6.3.3***. As a result, it was concluded that a way of reorthogonalising the vectors was required.

6.4.2 Full Reorthogonalisation via Gram-Schmidt

A full Gram-Schmidt reorthogonalisation was added to the Lanczos algorithm as an additional stage between steps 5 and 6 of the algorithm given in ***Section 6.3.2***; the resulting algorithm

shall herein be referred to as the L-NOSC. The steps required to implement this reorthogonalisation step are set out in the pseudo-code presented in ***Figure 6.3***.

Finally, a test to identify if the Gram-Schmidt reorthogonalisation has been successful was added to the code. This test works by taking the dot product of any two vectors, which should be equal to zero (in practice the dot product must be approximately equal to 0, i.e., $< 10^{-10}$) if the vectors are orthogonal.

```
//Step 1: Read in similarity matrix
Read in matrix A

//Step 2: Apply Simple Lanczos Algorithm
while(k < number of eigenpairs sought)
	generate a random vector r(i)
	//Lanczos Step 1: q(i) = r(i-1)/b(i-1)
	if(j = 0)
		vector q(i) = vector r(i) / square root(norm2(r(i)))
	else if(j > 0)
		q(i) = r(i) / off-diagonal element b(i-1)
	//Lanczos Step 2: u(i) = Aq(i)
	r(i) = A * q(i)
	//Lanczos Step 3: r(i) = r(i) - (b(i-1) * q(i-1))
	if(i = 0)
		q(i-1) = 0, therefore r is unchanged
	else if(i > 0)
		vector t1 = r(i) - (q(i-1) * b(i-1))
	//Lanczos Step 4: a(i) = q(i) * r(i)
	generate diagonal element a(i) which is equal to q(i) * r(i)
	Store diagonal element a(i) in 1D matrix a
	//Lanczos Step 5: r(i) = r(i) - (a(i) * q(i))
	vector t2(i) = r(i) - (a(i) * q(i))
	//Additional Step: Full Gram Schmidt Reorthogonalisation
	if(i = 0)
		vector already orthogonal to itself
	else if(i > 0){
		for(j = i; j >= 0; j--)
			vector t3(j) = copy vector r(j)
			double vr = t3(j) * r(i);
			vector t4(j) = t3(j) * vr
		r(i) – the sum of vectors t4(j) = orthogonalised r(i)
	//Lanczos Step 6: b(i) = norm(r(i))
	calculate off-diagonal element b(i) = norm2(r(i))
	Store off-diagonal element b(i) in 1D matrix b

//Step 3: Use QL Routine to identify eigenpairs
Pass matrices a & b into a implicit QL decomposition to identify the
eigenpairs
```

Figure 6-3: *A pseudo-code implementation of the symmetric Lanczos algorithm with a full vector reorthogonalisation procedure through the Gram-Schmidt method.*

6.4.3 Assessing the Accuracy of the L-NOSC Method

The L-NOSC algorithm was used to cluster the Stahl COX2 inhibitor dataset into ten clusters and the results compared to those obtained using the original m-NOSC algorithm configured to identify the eigenclusters associated with the ten largest eigenvalues. The two sets of clusters are given below:

Clusters formed with L-NOSC

1.91439 - 1 2 3 4 5 12 13 18 20 24 25 26 27 28 29 30 31 43 46 47 54 62 65 66 67 68 69 70 71 72 73 74 75 76 77 81 82 84 85 86 87 88 89 90 93 104 105 106 107 117 118 119 122 125
1.94257 - 34 35 40 63 83 103 120 121
2.52212 - 56 57 58 59 60 61
2.54941 - 16 17 36 37 38 39 41 42 44 45 64 123
2.82780 - 78 79 80
2.83641 - 6 7 11 114 115 116
3.01242 - 91 92 100 101 102
3.85216 - 8 9 10 19 21 22 23 48 49 50 51 55 124
4.33198 - 32 33 94 95 96 97 98 99
5.58525 - 14 15 52 53 108 109 110 111 112 113

Clusters formed using m-NOSC

1.91439 - 18 46 47 107 122 125
1.94257 - 34 35 40 63 83 103 118 119 120 121
2.52212 - 2 4 20 24 25 26 27 28 29 43 54 56 57 58 59 60 61 62 65 67 68 69 70 71 75 85 86 87 89
2.54941 - 16 17 36 37 38 39 41 42 44 45 64 123
2.82780 - 3 30 31 66 72 73 74 76 77 78 79 80 81 82 84 90 105
2.83641 - 6 7 11 104 114 115 116 117
3.01242 - 91 92 100 101 102
3.85216 - 8 9 10 12 13 19 21 22 23 48 49 50 51 55 106 124
4.33198 - 32 33 88 93 94 95 96 97 98 99
5.58525 - 14 15 52 53 108 109 110 111 112 113

The molecular identifiers shown in red indicate molecules that are placed within the same clusters by both methods, with the black identifiers highlighting molecules that have been assigned to different clusters. The numbers in bold provide the eigenvalues that each cluster is related too. Molecule 1 is excluded from the clustering by the m-NOSC approach as it forms a true singleton, i.e., makes an eigenvector contribution of 0 to each eigencluster. It should be noted that the differences between the eigenvalues and eigenvectors presented in this section and those obtained by Brewer in his 2007 study (Brewer), are the result of the application of a similarity threshold value of 0.001.

By comparing the two outputs, it is apparent that the two algorithms place the majority of molecules into the same clusters, with structures that share similar chemical scaffolds being clustered together. Discrepancies in the assignment of molecules to clusters can be explained by the presence of low level roundoff error that is inherent in the eigenvectors calculated using the Lanczos-based approaches. This is due to the disproportionately large effect roundoff error has on the classification of molecules based on very low eigenvector elements; which in turn, leads to all molecules that would usually be placed into one of the other clusters based on low eigenvector element scores being assigned to the cluster with the smallest eigenvalue, for example, the cluster at λ = 1.91439. The removal of these low contributing molecules from their respective clusters is in fact desirable as these molecules tend to be based on significantly different scaffolds to the other molecules that form the eigencluster in m-NOSC. This is illustrated by inspecting the two eigenclusters related to the two largest eigenvalues produced by both algorithms. ***Figures 6.4 and 6.5*** depict the clusters at λ = 5.58525 and λ = 4.33198 respectively.

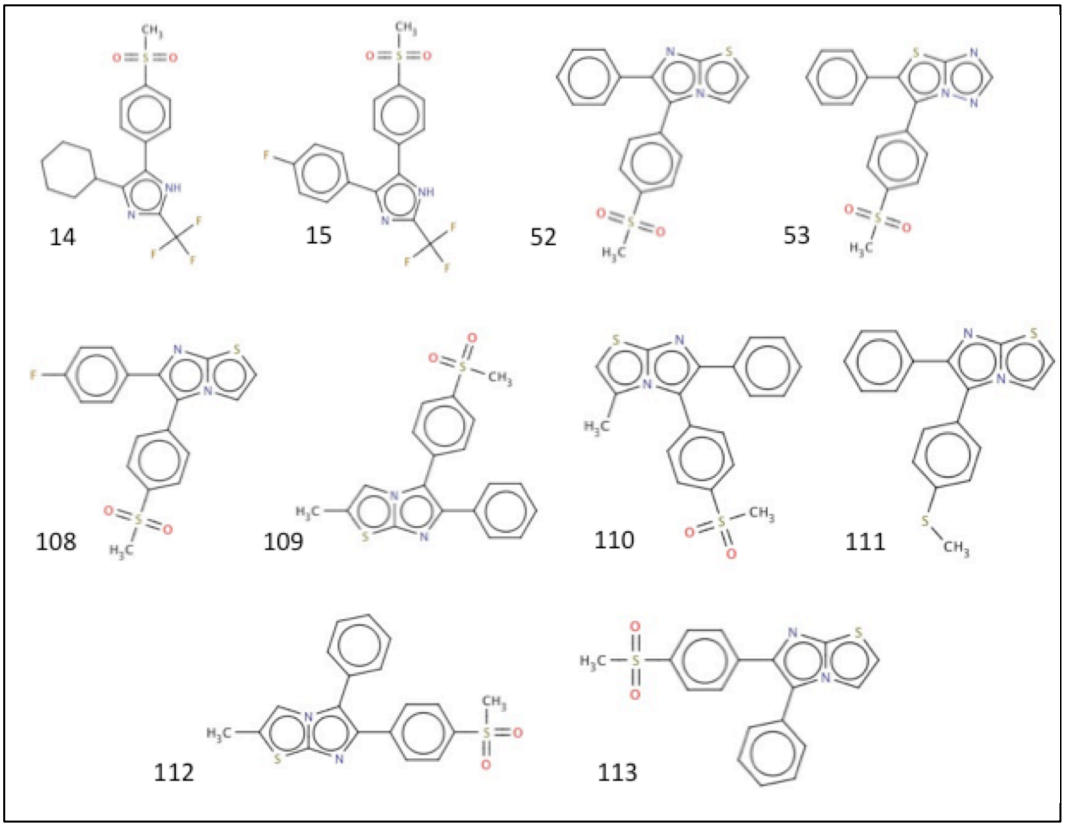


Figure 6-4: *Molecules placed in the cluster for λ = 5.58525, when using both spectral clustering method.*

The clusters produced for λ = 5.58525 are, mainly, based on a 5,6-diphenylimidazo[2,1-b][1,3]thiazole scaffold, with both clustering algorithms identifying an identical set of

molecules. Considering the structures within the cluster, molecules 14 and 15 are based on a significantly different scaffold to the other compounds, which is reflected by the considerably lower contribution to the eigencluster (see ***Table 6.1***). There are two structural changes in the fused ring system of molecule 53 compared to the majority of the other compounds: the presence of an additional nitrogen and switching of the placement of the sulphur and another nitrogen atom; this also leads to the compound providing a decreased contribution to the cluster. It is important to note that each molecule's contribution to the cluster formed by both methods is identical to 5 decimal places, indicating the roundoff error introduced by the Lanczos algorithm is less than 1×10^{-5}, see ***Table 6.1***.

Identifier	L-NOSC score	m-NOSC score
14	0.00011	0.00011
15	0.00210	0.00210
52	0.42504	0.42504
53	0.00712	0.00712
108	0.36100	0.36152
109	0.37500	0.37502
110	0.37900	0.37989
111	0.28679	0.28679
112	0.37502	0.37502
113	0.42500	0.42504

Table 6-1: *Eigenvector elements for cluster at λ = 5.58525, when using both the L-NOSC and m-NOSC algorithms.*

The cluster produced for λ = 4.33198 is based on a 3-(4-methanesulphonylphenyl)-2-(pyridine-3-yl)pyridine scaffold and differs for the two clustering algorithms, with molecules 88 and 93 being omitted from the cluster when using the Lanczos-based method. Molecule 88 is based on a significantly different scaffold to the other molecules in the cluster, which is reflected in the significantly lower contribution made to the cluster. Similarly, the omission of molecule 93 can be attributed to the replacement of the middle pyridine with a thiophene. In fact the scaffold difference between the molecules with low contributions and the other compounds is so great that one could argue that the m-NOSC algorithm has miscategorised them. Furthermore the individual contributions made to the cluster by the molecules are again identical for both methods to 5 decimal places, see ***Table 6.2***, showing that Lanczos

decomposition produces approximations that are sufficiently accurate for use in clustering chemical data.

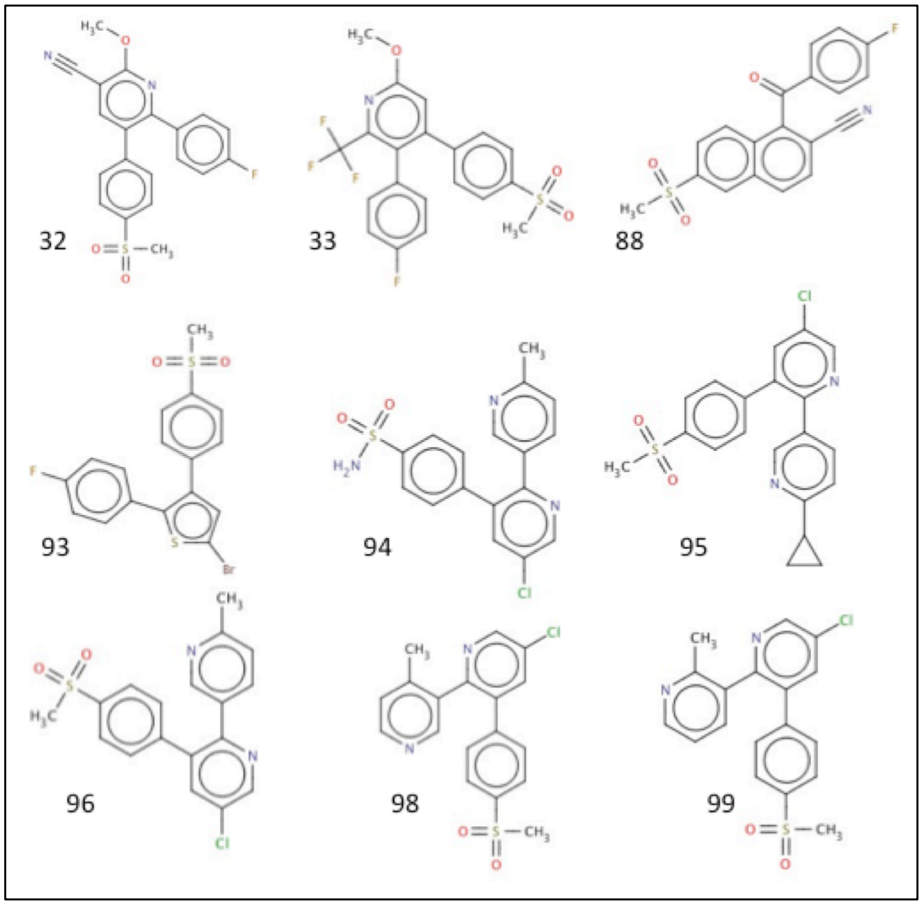


Figure 6-5: *Molecules placed in the cluster for λ = 4.33198, when using both spectral clustering methods.*

Identifier	L-NOSC Element	m-NOSC Element
32	0.00702	0.00702
33	0.0002	0.0002
88	-2.94E-17	-4.66E-16
93	-8.42E-18	5.04E-17
94	0.24253	0.24253
95	0.30549	0.30549
96	0.47722	0.47722
98	0.44083	0.44083
99	0.46536	0.46536

Table 6-2: *Eigenvector elements for cluster at λ = 4.33198, when using both the L-NOSC and m-NOSC algorithm.*

Although, the presence of low-level roundoff error associated with the L-NOSC leads to the removal of molecules that make small contributions to a cluster in the m-NOSC method, it unfortunately places them into the cluster associated with the smallest positive eigenvalue. This occurs, as the eigenvector associated with the smallest eigenvalue is the least well refined, due to the fact that it has not undergone as many iterations – and therefore iterative refinements – as the other k eigenvectors. A solution to this problem would be to introduce an eigenvector threshold that places molecules below the threshold into an artificial singleton cluster. Alternatively the algorithm could be forced to undergo several iterative refinements however, that would come at a significant additional time cost, which is undesirable.

6.4.4 Variation in the Number of Positive Eigenpairs with k

Before an in-depth comparison of L-NOSC and other clustering methods can be made, the relationship between the number of eigenpairs sought, k, and the number of eigenclusters formed must be understood. The eigenpairs that are calculated by L-NOSC can include both positive and negative eigenvalues. However, clusters are only generated from eigenvectors with positive eigenvalues, as both our preliminary investigations and previous studies by other groups have shown that the eigenvectors related to the negative eigenvalues produce poor clusters of the data (Sarkar and Boyer, 1998). Therefore, the relationship between k and the number of positive eigenvectors, p, was investigated. This relationship is not trivial, and the ratio of positive to negative eigenvalues varies based on the characteristics of the dataset being clustered. Thus the ratios of positive and negative eigenvalues found for various values of k for four different ChEMBL datasets, described by different molecular descriptors were investigated.

Filtered similarity matrices for each of the four ChEMBL datasets and the different descriptor types were generated, at γ = 25, 50, 75 and 100. These similarity matrices were then made sparse using a threshold value of 0.001 and the L-NOSC algorithm was applied for k = 100, 200, 300 and 400.

6.4.4.1 Results and Discussion

The results of this study can be split into three sections; to reflect the effect that each of the main parameters – dataset homogeneity, fingerprint type and value of γ - has on the ratio of k to p.

The results presented in ***Table 6.3***, show how the percentage of positive eigenvalues, p, varies between activity classes. The percentage of k that was positive exceeds 80% in almost all instances, with the only exception occurring for the Renin dataset when k = 400. An explanation for this anomaly, and moreover a possible reason why the Renin dataset yields lower percentages of p in all instances, is the homogeneity of the dataset. The high homogeneity of this dataset means many of the molecules have similarity scores that are of similar magnitudes; this leads to the production of eigenvalues that span a larger range of values, which results in the L-NOSC algorithm finding a greater number of negative eigenvalues that converge between the larger positive eigenvalues.

Dataset	k	p	% p
5HT1A	100	88	88.00
	200	167	83.50
	300	258	86.00
	400	333	83.25
MMP1	100	91	91.00
	200	176	88.00
	300	258	86.00
	400	341	85.25
Renin	100	84	84.00
	200	162	81.00
	300	243	81.00
	400	315	78.75
SubP	100	89	89.00
	200	169	84.50
	300	254	84.67
	400	342	85.50

Table 6-3: *Table showing how the percentage of positive eigenvalues, p, varies for different datasets, described by BCI fingerprints at γ = 25 and using a sparsity threshold of 0.001.*

Experiments aimed at identifying the effect that the choice of molecular descriptor has on the relationship between k and p were carried out for each of the four ChEMBL activity classes using γ values of 25, 50, 75 and 100. ***Table 6.4*** contains the results of the study for the MMP1 activity class at a γ value of 25. Further results can be found in ***Appendix C***.

The results highlight how the percentage of k that is positive is at least 80% in all instances; with the percentage of p decreasing for each fingerprint type as the value of k increases. This trend would be expected as when considering the large values of k, the difference between the smallest eigenvalues in the p set and the largest absolute eigenvalues in the negative set

can be small, therefore leading to the convergence of an increased number of negative eigenvalues within the k eigenpairs.

Fingerprint	k	p	% p
BCI	100	91	91.00
	200	176	88.00
	300	258	86.00
	400	341	85.25
Daylight	100	93	93.00
	200	175	87.50
	300	258	86.00
	400	348	87.00
ECFP_4	100	83	83.00
	200	165	82.50
	300	242	80.67
	400	336	84.00
MDL PK	100	91	91.00
	200	173	86.50
	300	253	84.33
	400	329	82.25
Unity	100	92	92.00
	200	177	88.50
	300	257	85.67
	400	339	84.75

Table 6-4: *Table showing how the percentage of positive eigenvalues, p, varies based on the choice of molecular descriptor for MMP1 activity class at γ = 25 and using a sparsity threshold of 0.001.*

The effect that the γ value has on the percentage p retrieved for the 5HT1A dataset is given in ***Table 6.5***. The results of this study show that when γ is less than 75, the percentage p score for the 5HT1A dataset is above 80% in all instances, which is also the case for the other datasets given in ***Appendix C***. Above γ = 75, this percentage p score begins to decrease below the 80% mark. This can be explained by the effect of the γ value in the Gaussian filtering function on the magnitude of the eigenvalues found using the Lanczos decomposition. As the value of γ is increased the span of the eigenvalue magnitudes is decreased, leading to a smaller number of positive eigenvalues converging before negative eigenvalues.

Gamma	k	p	% p
25	100	88	88.00
	200	167	83.50
	300	258	86.00
	400	333	83.25
50	100	82	82.00
	200	164	82.00
	300	251	83.67
	400	321	80.25
75	100	81	81.00
	200	161	80.50
	300	249	83.00
	400	324	81.00
100	100	81	81.00
	200	159	79.50
	300	249	83.00
	400	309	77.25

Table 6-5: *Results for how the percentage of positive eigenvalues, p, varies based on the choice of γ for 5HT1A activity class described by BCI fingerprints and using a sparsity threshold value of 0.001*

Therefore, it can be concluded that in order to guarantee that the number of eigenpairs identified is sufficient, the value of k must be set at a value 25% higher than value of k required, for example, if 100 eigenpairs are required k should be set to 125.

6.4.5 Choosing an appropriate value of p

Another important consideration is how to decide what is an appropriate value of p. Unfortunately there is no hard and fast rule for determining an appropriate value of p to use in the spectral clustering of data, and instead a balance must be struck between finding enough eigenpairs to give a good separation of the data and avoiding the identification of too many eigenpairs due to the additional computational costs this leads to. Thus, to answer this question, two considerations must be made:

- How the respective magnitudes of the eigenvalues vary across a dataset (see ***Figure 6.6***).
- If the distribution of cluster types changes as more eigenpairs are used in the clustering of a dataset (see ***Figure 6.7***).

Figure 6.6 shows the distribution of eigenvalues for the SubP activity class described by ECFP_4 fingerprints. In this figure, the eigenvalues are ranked according to their increasing magnitudes, leading to the right hand side of the distribution having a steep gradient that

reflects the rapid decrease in the size of the eigenvalues as more are found. When using the L-NOSC algorithm, the eigenvalues are generally found in decreasing order. This is a desirable feature, as the eigenpairs that contain the "most" information will be found in this subset, just like in principal component analysis where the first few principal components contain the greatest share of the required information. The large computational costs associated with identifying increasing numbers of eigenpairs dictates that a minimal number of eigenpairs should be sought. In particular, identifying eigenvalues that between the eigenvalue magnitudes of 0 and 1 for matrices where N ≥ 2000, should be avoided due to large number of iterations required to identify their respective eigenvectors.

This general distribution of the eigenvalues is seen for each of the activity class and descriptor combinations, with two points of variation. Firstly, the general magnitude of the largest eigenvalues changes as their magnitudes are affected by the size of the clusters, for example, MDL public keys produce larger maximum eigenvalues than Daylight, which produce larger than ECFP_4. Secondly, the number of eigenpairs between the maximum value and 1 will vary as this some fingerprints types will find multiple significant clusters whereas others will only find a few, for example, ECFP_4 fingerprints would be expected to find more eigenvalues with a magnitude greater than 1 than MDL public keys.

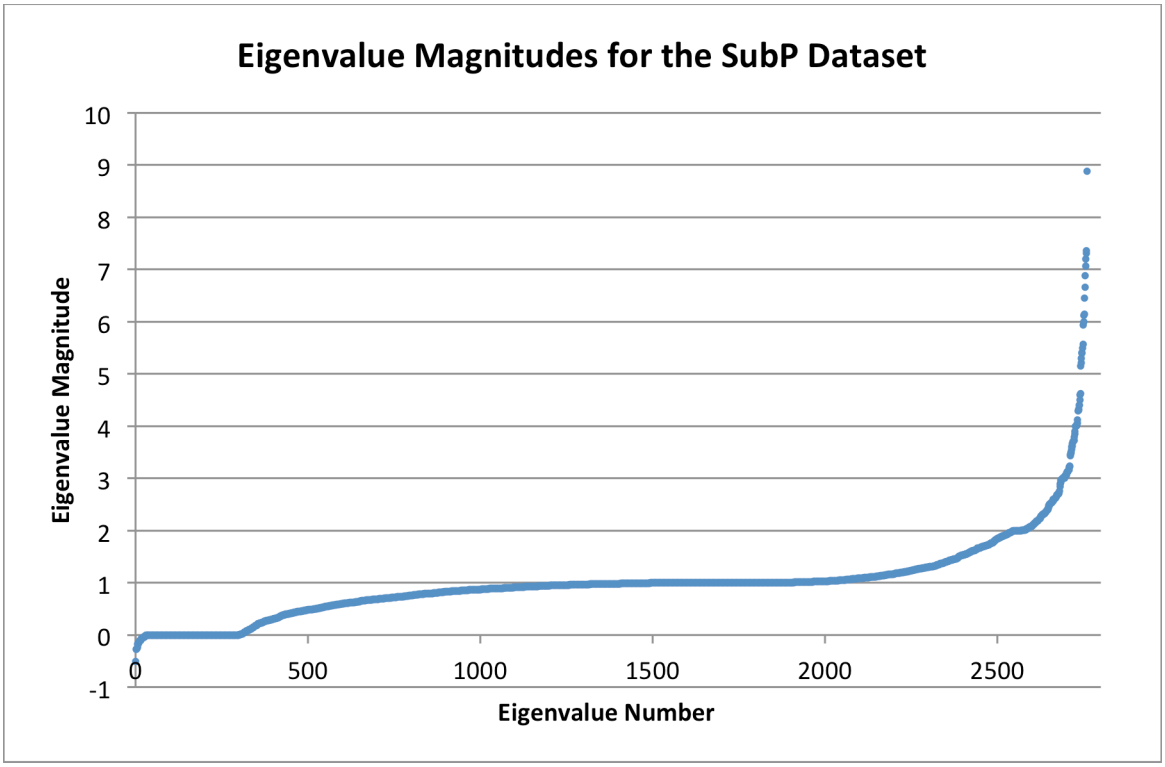


Figure 6-6: *Magnitudes of the eigenvalues for the SubP activity class described by ECFP_4 fingerprints and using a gamma value of 25.*

Figure 6.7 shows how the distribution of the different types of clusters formed changes when the p subset begins to encompass an increasing number of eigenvalues. Four different types of clusters can be identified during spectral clustering:

- Positive Clusters are eigenclusters that contain at least two compounds.
- Empty Clusters, which are formed when an eigenvector forms an eigencluster yet no compounds are assigned to this cluster based upon their eigenvector score.
- True Singletons, which are singleton clusters that are naturally formed, i.e., formed as a result of one compound being based on a significantly different chemical scaffold and hence, leading to the production of an eigenvector dominated by a single molecule.
- Forced Singletons/unclassified molecules, which are formed from molecules that do not make a contribution large enough to be assigned to an eigencluster and as a result are forced into a cluster on their own.

As the number of positive eigenvalues reaches 850 there is an obvious change in the gradient of the curve showing the number of positive clusters formed. This represents the point at which finding further eigenvalues becomes undesirable as quality of the clustering begins to decrease rapidly, which is shown by the decreasing number of eigenvalues forming new clusters containing more than a single molecule, and the rapid increase in the number of empty clusters and forced singletons formed. Upon further inspection, it can be noted that the eigenvalues associated with clusters beyond this point have magnitudes of 1.008 or less. By looking at other datasets we can see that this trend continues with eigenvalues below 1.014 for the Renin dataset and 1.009 for the MMP1 activity class, producing increasingly large numbers of non-clustered molecules and singletons clusters, when using a gamma value of 25 and ECFP_4 fingerprints.

Thus, it can be concluded that there is an optimum upper limit to p, beyond which the number of meaningful clusters decreases rapidly. This limit is associated with the calculation of eigenvalues below a certain threshold magnitude that varies from dataset to dataset, for example, the point at which there is a decline in the cluster quality for the ChEMBL datasets is with eigenvalues below 1.01.

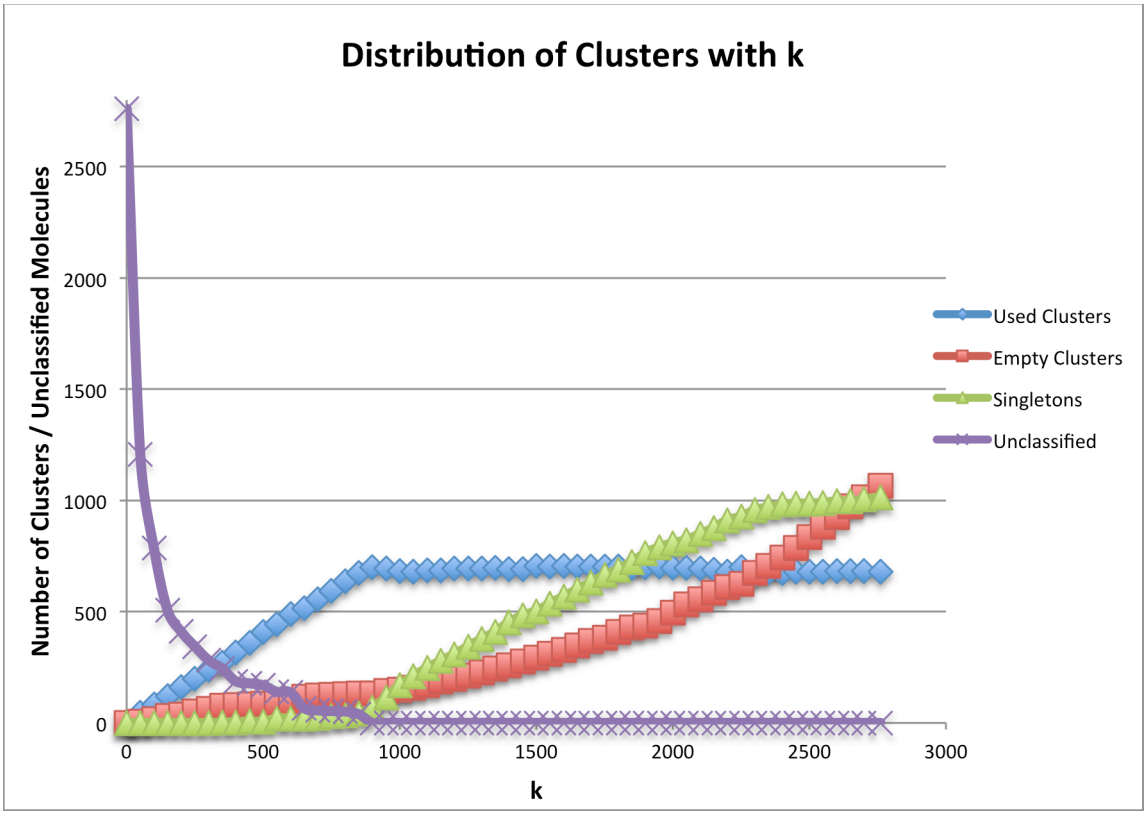


Figure 6-7: *Distribution of different cluster types as p is increased, for the SubP activity class described by ECFP_4 fingerprints and using a gamma value of 25.*

6.5 Comparing the L-NOSC Algorithm to other Clustering Methods

The effectiveness of the L-NOSC algorithm for clustering activity classes was assessed using the QCI measure, and the results compared to the m-NOSC, k-means and Ward's clustering methods. The performance of the L-NOSC algorithm was tested with four values of gamma, 25, 50, 75 and 100, and four different clustering levels, where p ≈ 100, 200, 300 and 400, identified using k values of 125, 250, 375 and 500 respectively. In these experiments an eigenvector threshold value was introduced to ensure that molecules were not erroneously classified into clusters based on eigenvector elements affected by the residual error inherent in Lanczos-approximated eigenpairs. The eigenvector threshold used in this section was 0.0001; with molecules that make a contribution less than this score being set to zero, and compounds related to an element that is equal or greater the threshold remaining unaffected.

6.5.1 Results and Discussion

Table 6.6 contains the QCI values obtained for the 5HT1A dataset described by standard 1052-bit BCI fingerprints. The two spectral clustering algorithms show identical performances. This assignment of the compounds by the two methods is the result of the eigenvector threshold

only allowing the classification of molecules based on their component values to five decimal places, i.e., the value of the eigenvector threshold was set to 0.00001 for all experiments. By limiting the assignment of molecules in this manner, the unwanted classification of compounds to clusters based on significantly different molecular scaffolds is prevented.

Gamma	*Approx.* p desired	L-NOSC	m-NOSC	k-means	Ward's
25	100	21.753	21.753	22.571	22.959
	200	26.708	26.708	27.636	29.063
	300	25.210	25.210	30.374	32.059
	400	26.816	26.816	31.923	32.447
50	100	25.458	25.458	22.571	22.959
	200	26.218	26.218	27.636	29.063
	300	27.001	27.001	30.374	32.059
	400	28.947	28.947	31.923	32.447
75	100	29.528	29.528	22.571	22.959
	200	32.107	32.107	27.636	29.063
	300	33.902	33.902	30.374	32.059
	400	34.889	34.889	31.923	32.447
100	100	23.691	23.691	22.571	22.959
	200	31.828	31.828	27.636	29.063
	300	31.993	31.993	30.374	32.059
	400	31.782	31.782	31.923	32.447

Table 6-6: *QCI score comparison for the clustering of the 5HT1A activity class described by BCI fingerprints. It should be noted that both spectral clustering approaches were applied using the gamma values contained in the table and a similarity threshold of 0.01.*

The effect that the eigenvector threshold has on the QCI scores obtained using both methods is illustrated in ***Table 6.7***, where the results for the same dataset are presented without using the eigenvector threshold. The differences between the respective QCI values that were produced for the two spectral clustering methods can be explained by the roundoff errors in the eigenpairs produced using the Lanczos decomposition. This residual error results in the methods differing in their classification of some molecules into the cluster associated with the smallest positive eigenvalue that was generated. The minimal fluctuation seen between QCI results at different clustering levels is due to many of the extra eigenpairs generated at higher clustering levels forming either clusters containing no actives or empty clusters, i.e., clusters containing no molecules. These observations are also consistent for each of the other activity classes, with other examples being presented in ***Table 6.8*** and ***Appendix C***.

The comparison of the performance of the L-NOSC with that of the two different traditional clustering algorithms shows that both algorithms provide similar performances in discriminating actives and inactives. This provides evidence that the L-NOSC can be considered a viable alternative to the traditional clustering methods for the task of activity class clustering.

Gamma	*Approx.* p desired	L-NOSC	m-NOSC
25	100	22.486	21.753
	200	25.998	26.708
	300	27.640	25.210
	400	30.963	26.816
50	100	20.892	25.458
	200	24.739	26.218
	300	26.336	27.001
	400	28.947	28.947
75	100	18.182	29.528
	200	22.459	32.107
	300	24.589	33.902
	400	27.453	34.889
100	100	24.288	23.691
	200	21.118	31.828
	300	23.409	31.993
	400	26.735	31.782

Table 6-7: *Results of clustering the 5HT1A activity class described by BCI fingerprints, without the use of the eigenvector threshold.*

Gamma	*Approx.* p desired	L-NOSC	m-NOSC	K-means	Ward's
25	100	46.923	46.923	56.317	57.243
	200	60.173	60.173	60.120	60.224
	300	58.683	58.683	62.110	60.728
	400	55.353	55.353	62.039	60.972
50	100	54.688	54.688	56.317	57.243
	200	65.508	65.508	60.120	60.224
	300	63.265	63.265	62.110	60.728
	400	58.781	58.781	62.039	60.972
75	100	52.927	52.927	56.317	57.243
	200	60.902	60.902	60.120	60.224
	300	64.032	64.032	62.110	60.728
	400	62.727	62.727	62.039	60.972
100	100	56.161	56.161	56.317	57.243
	200	62.413	62.413	60.120	60.224
	300	61.888	61.888	62.110	60.728
	400	61.746	61.746	62.039	60.972

Table 6-8: *A QCI score comparison for the spectral clustering of the Renin activity class described by Unity fingerprints. Using a similarity threshold of 0.01 and a gamma value as shown in the table for the spectral clustering implementations.*

6.5.2 Conclusion

The comparable QCI values obtained for the respective spectral clustering algorithms shows that the L-NOSC algorithm can be used in preference to the m-NOSC method without having an impact on the results that are produced for a wide range of parameters. Furthermore, the L-NOSC algorithm also provides comparable results to those obtained for the traditional clustering methods, indicating that this method can be used as an alternative approach to clustering molecules.

6.6 Parameterisation Experiments

After establishing that the use of a L-NOSC algorithm can yield identical results to those produced when using a full matrix diagonalisation, the next step was to carry out a robust parameterisation of the method. The effect several parameters have on the algorithm were investigated in order to identify the algorithm's scalability in terms of the maximal size of datasets that can be handled and also how the time requirements increase with the number of clusters sought.

6.6.1 Number of Clusters vs Time

The L-NOSC algorithm, was applied to the four ChEMBL datasets using the different descriptors, γ = 25, similarity and eigenvectors threshold of 0.0001 and nine different values of k. The time required to identify the top k first-to-converge eigenpairs in each experiment was then recorded. These times were compared against the time taken to carry out a full matrix diagonalisation of the activity classes with the aim of identifying the point at which the Lanczos based approach becomes more time consuming than the use of the full matrix diagonalisation procedure.

6.6.1.1 Results and Discussion

Table 6.9 shows the relationship between k and time for the L-NOSC algorithm and ***Figure 6.8*** plots the results for one dataset graphically.

Activity Class	Fingerprint	k						m-NOSC
		100	200	300	400	500	600	
5HT1A	BCI	17.42	66.76	410.95	1451.23	3355.19	5081.93	1685.05
	Daylight	16.97	77.23	421.23	1768.59	3244.95	5357.62	1698.65
	ECFP_4	17.21	175.91	1052.50	2177.52	4146.95	6005.34	1616.19
	MDL Public Keys	18.55	69.57	214.56	964.37	2114.61	4542.52	1602.65
	Unity	16.21	73.08	328.01	1588.75	3064.07	5337.22	1701.44
MMP1	BCI	24.22	75.82	272.30	999.38	2706.75	6379.30	3358.29
	Daylight	22.49	85.92	357.81	1112.06	3236.20	6963.14	3465.69
	ECFP_4	28.85	214.55	838.30	2125.15	4158.92	7524.92	3584.63
	MDL Public Keys	34.80	77.62	239.55	967.41	2402.76	5955.52	3518.52
	Unity	22.43	67.22	320.14	946.11	2471.65	6089.80	3563.07
Renin	BCI	13.41	40.74	142.86	658.57	1826.35	2858.94	706.15
	Daylight	11.48	38.91	203.21	865.29	2437.44	3798.93	706.38
	ECFP_4	9.5	103.2	644.9	1456.59	2844.05	3863.65	758.82
	MDL Public Keys	15.36	42.46	147.57	651.72	1364.27	2400.64	686.90
	Unity	15.13	37.95	166.95	737.82	2040.88	2754.65	733.03
SubP	BCI	16.39	49.3	198.05	1007.14	2401.32	4610.92	1592.47
	Daylight	16.54	65.76	275.66	1147.57	2521.81	5141.07	1548.26
	ECFP_4	14.68	126.41	497.83	1917.25	3474.35	5858.51	1645.95
	MDL Public Keys	19.71	68.59	198.61	926.66	2382.75	4224.71	1573.79
	Unity	16.66	58.15	216.11	1047.71	2759.92	5701.43	1653.70

Table 6-9: *The time required to identify k eigenpairs from each dataset/fingerprint combination using the LNOSC algorithm, compared to the time required to carry out a full matrix diagonalisation procedure.*

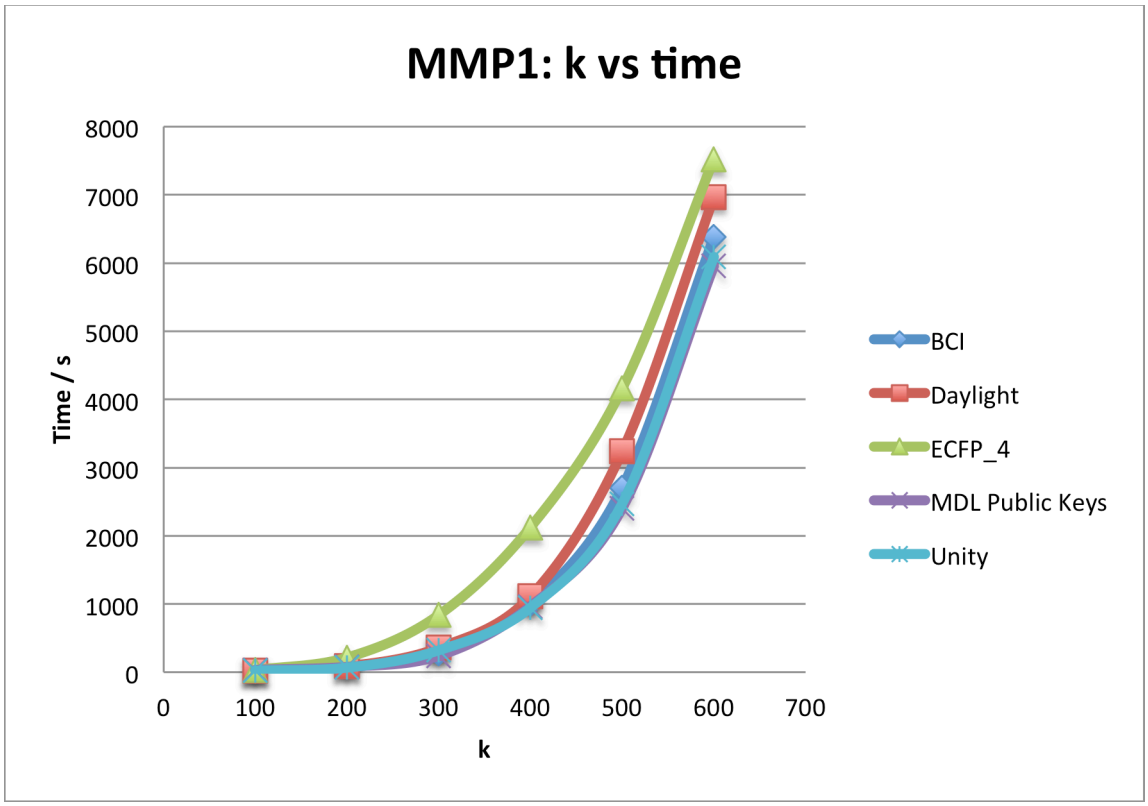


***Figure 6-8**: Graph comparing k v time for the MMP1 dataset, which contains 3482 molecules.*

Figure 6.8 shows that time increases according to an almost second order scaling with k, i.e., $O(k^2)$. This leads to the assumption that there will be a point at which the time costs, required to identify a significantly high value of k, will become too high to justify the use of the L-NOSC in preference to the m-NOSC.

An important observation is that, in general, there is an increase in the time required to identify the top k eigenpairs as the size of the dataset increases, which is observed when studying the time requirements for each of the four ChEMBL datasets (where the Renin, SubP, 5HT1A and MMP1 are 2166, 2760, 2785 and 3482 molecules in size respectively). Although this relationship is to be expected, it is interesting to see the significant difference between the average time taken to calculate the top 600 eigenpairs for MMP1 and Renin activity classes, which requires 5962 seconds and 3135 seconds respectively. This highlights how unfavourably the speed of the algorithm scales with the number of molecules in the dataset, as the MMP1 and Renin activity classes contain 3482 and 2160 compounds respectively.

Interestingly, the time required to calculate k eigenpairs for ECFP_4 fingerprints was found to be considerably longer than for any other fingerprint type. This result was unexpected as in general ECFP_4 fingerprint produce the most sparsely populated input matrices, and therefore

we would expect their decomposition to take significantly less time than other fingerprint types that produce more densely populated input matrices. An explanation for this observation is based around the issues encountered by the Lanczos algorithm when it is applied to finite precision mathematic problems. These issues mean that the generation of eigenvectors from ECFP_4 similarity matrices, which produce eigenvectors containing a large number of extremely small elements, leads to the need for more operations per iteration to be carried out in order to elucidate and optimise the calculated eigenvectors, significantly increasing the time costs of the operation. This result casts doubt upon the continued use of the Lanczos algorithm within the spectral clustering method as its use with ECFP_4 fingerprints, and other circular fingerprints, is vitally important as they consistently yield the best results.

The observed performance of MDL public keys was also unexpected, as their matrices, which were densely populated in some cases, continually required the least amount of time to decompose. These observations were the result of the Lanczos algorithm performing favourably with the smaller span of similarity values in the input matrices, which led to a minimisation in the number of operations needed to identify eigenvectors with this method. The only exception is in the case of the SubP dataset, where the MDL public key based matrices took longer to decompose than BCI and Unity based matrices. The other three fingerprint types used within this study are intermediates between these two extremes with the comparative time requirements using each fingerprint type varying based on the dataset being evaluated.

The results for the secondary study into the time required to apply a full matrix diagonalisation procedure to the activity classes and their descriptors, are presented in ***Table 6.10***. This table shows the average time taken to fully diagonalise the matrix S^F for each fingerprint type and largest value of k that can be identified using the L-NOSC algorithm before the time costs exceed those of the complete matrix diagonalisation method, m-NOSC. By plotting the results of this study on to a graph, the point at which the L-NOSC stops being more efficient, in terms of time, can be identified as the intercept between the two curves. An example is given in ***Figure 6.9***.

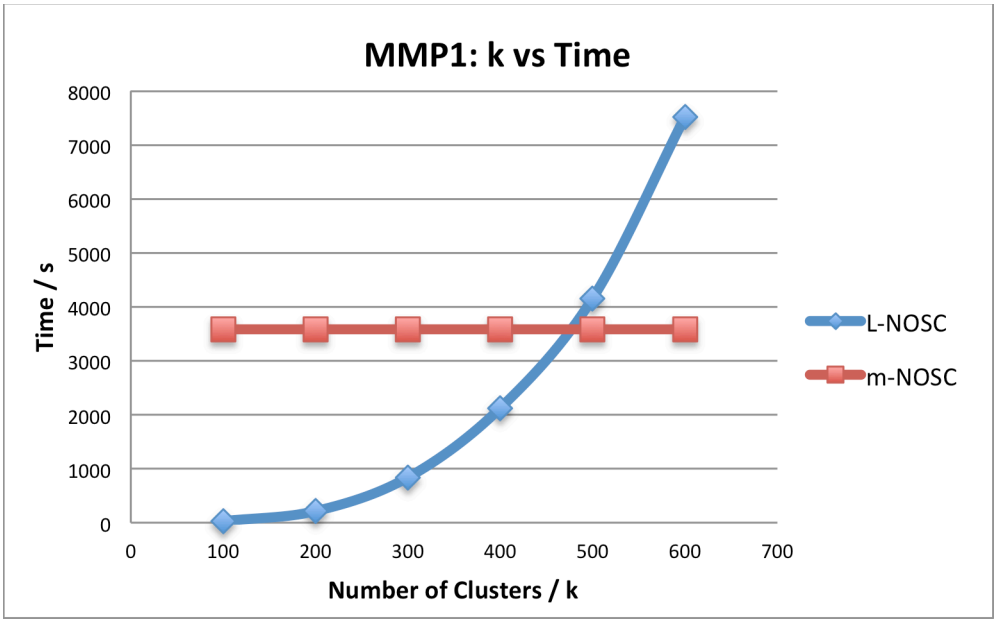


Figure 6-9: *Graph depicting the time taken to generate varying values of k using both the L-NOSC and m-NOSC algorithms.*

		Time Taken for diagonalisation with the m-NOSC/ s	Largest k value for which the L-NOSC is faster
5HT1A	BCI	1685	400
	Daylight	1699	350
	ECFP_4	1616	350
	MDL Public Keys	1603	450
	Unity	1701	400
MMP1	BCI	3358	500
	Daylight	3466	500
	ECFP_4	3585	450
	MDL Public Keys	3519	500
	Unity	3563	500
Renin	BCI	706	400
	Daylight	706	350
	ECFP_4	759	300
	MDL Public Keys	687	400
	Unity	733	350
SubP	BCI	1592	450
	Daylight	1548	450
	ECFP_4	1646	350
	MDL Public Keys	1574	450
	Unity	1654	450

Table 6-10: *The time required to carry out a full diagonalisation of each dataset, for each descriptor, and the maximum number of eigenpairs that can be calculated using the Lanczos decomposition before exceeding the time required for the diagonalisation.*

The value of k at which the Lanczos algorithm stops being faster than the full matrix diagonalisation procedure varies between k = 300 to k = 500 for the four ChEMBL activity classes. With the maximum value of k for ECFPs often being at least 50 less than the other fingerprint types. When this information is considered in addition to the results of the study in ***Section 6.4.5***, it can be concluded that the Lanczos algorithm is unable to identify all the eigenpairs related to useful clusters, from the four ChEMBL datasets, faster than the full matrix diagonalisation procedure. Therefore, more research is required to further decrease the time costs of this method, if it is to be considered as a completely viable alternative to the use of full matrix diagonalisation.

By examining the relationship between the maximal value of k and the size of the datasets presented in ***Table 6.10***, a hypothesis can be made that the use of the usefulness of the L-NOSC algorithm increases with the size of the dataset being clustered. As the cost savings this algorithm provides – when compared to the full matrix diagonalisation procedure – are more noticeable with larger datasets.

6.6.2 Testing the L-NOSC Algorithm's Ability to Cluster Large Datasets

The most significant drawback associated with the use of spectral clustering methods that utilise a full matrix diagonalisation procedure, is that the algorithm is realistically limited to use with datasets of 4000 molecules or less. Thus, identifying a different approach that increases the maximum size of dataset that can be handled is vital.

Hence, experiments were carried out aimed at identifying, firstly, if it was possible to cluster datasets up to 10000 molecules and secondly the respective times required to calculate the top k eigenpairs of each dataset, where k = 100, 200, 300, 400, 500 and 600. In order to generate data for these experiments each of the four ChEMBL datasets (5HT1A, MMP1, Renin and SubP datasets which contain 2784, 3482, 2166 & 2760 molecules respectively) used in the previous studies were agglomerated, and random selections of between 1000 - 10000 molecules identified. The results of these experiments are given in ***Table 6.11*** and are presented graphically in ***Figure 6.10.***

The results provided in ***Table 6.11*** and ***Figure 6.10*** show three significant trends. Firstly, there is a positive correlation between the number of clusters k and the time required to identify them, with larger values of k requiring a significantly increased time investment to calculate the eigenvectors associated with the smaller eigenvalues. Secondly the time required to

calculate k clusters increases rapidly in accordance with the size of the dataset, for example, it takes 509.44 seconds to identify 600 clusters from a dataset of 1000 molecules yet this increases to 27694.86 seconds for a dataset of 10000 compounds. Finally as the value of N rises the difference in the time taken to calculate an increasing number of clusters within a dataset increases significantly, for example, it takes an additional 111 seconds to go from calculating 300 to 400 eigenpairs for 1000 molecules, whereas it takes an additional 2778 for 5000 molecules. This is a result of the additional calculations required to identify the eigenpairs for larger matrices.

Dataset Size	**k**					
	100	**200**	**300**	**400**	**500**	**600**
1000	4	16	49	159	159	509
2000	11	70	135	579	1007	1742
3000	20	93	588	905	1743	3476
4000	35	207	811	1981	2964	4791
5000	59	335	853	3631	7101	8746
6000	73	261	1213	3275	6997	15181
7000	92	374	1305	4153	8448	15776
8000	124	430	1547	5532	12614	18518
9000	168	521	1875	6494	15425	23289
10000	221	548	2051	7013	18909	27695

Table 6-11: *Shows the time (in seconds) required to identify the top k eigenpairs of datasets varying in size between 1000-10000 molecules described by ECFP_4 fingerprints using a γ value of 25.*

Furthermore, as the size of the dataset increases the time saving that can be made through the use of the L-NOSC method grows rapidly, for example, using a full matrix diagonalisation to identify the eigenpairs of a matrix where N = 5000 takes approximately 9147 seconds, whereas finding 600 eigenpairs using the L-NOSC algorithm requires approximately 8745. When N is increased to 10000, a full matrix diagonalisation procedure requires approximately 45511 seconds to complete, yet identifying 600 eigenpairs requires only 27695 seconds.

Thus it can be concluded that there are two major determining factors in the time requirements for the L-NOSC algorithm, the dataset size and the number of eigenpairs that are identified; and as the value of N increases, so does the possible efficiency savings that can be made through the use of the L-NOSC method.

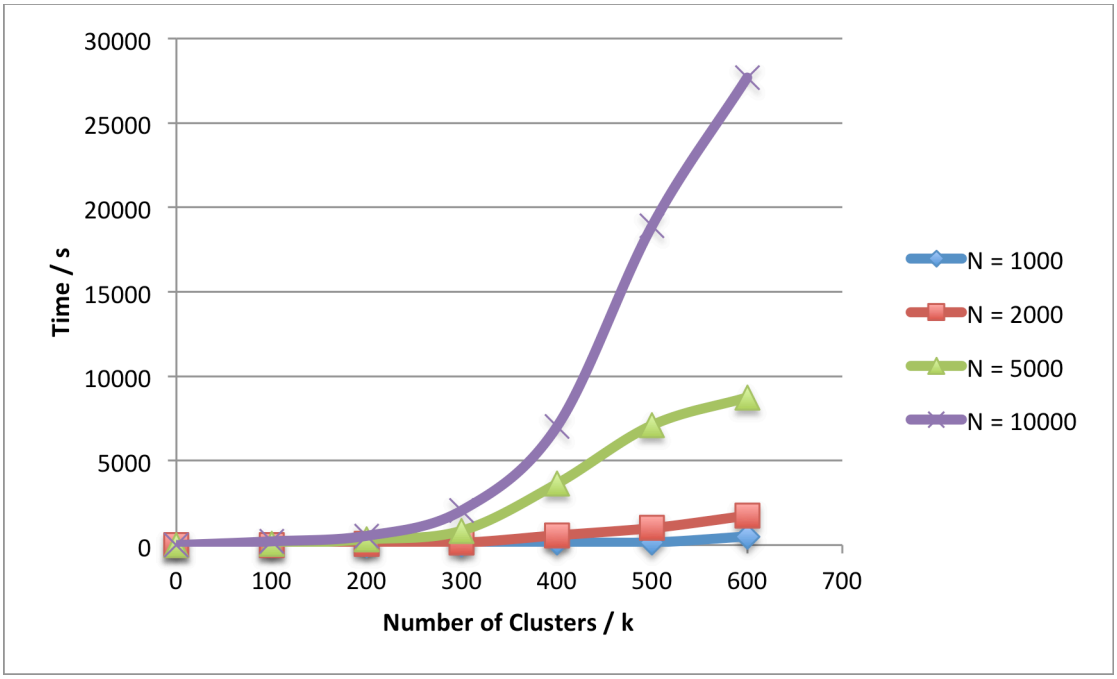


Figure 6-10: *Graph showing the time taken to cluster datasets of different sizes varies with k. Each dataset was formed by taking a random selection of molecules from an agglomerated dataset containing each of the four ChEMBL activity classes, described by ECFP_4 fingerprints and using a γ value of 50 in the Gaussian filtering function.*

6.7 Summary

This study has shown that the Lanczos algorithm provides a useable alternative to full matrix diagonalisation for the calculation of eigenpairs, at least up to the point at which the time required to approximate k eigenpairs exceeds that of full matrix diagonalisation procedure. Studies throughout this chapter have shown that despite issues with losing orthogonality, the Lanczos algorithm coupled with a Gram-Schmidt reorthogonalisation procedure was shown to produce eigenvector approximations that are accurate to at least 5 decimal places. Due to the accuracy of the approximations, the L-NOSC algorithm is able to generate results identical to those obtained with the m-NOSC method. A major disadvantage of this method is its poor performance with ECFPs, which is problematic due to the ECFP fingerprints continuously providing the best performances with the spectral clustering approach.

The application of the L-NOSC algorithm to larger datasets is more promising, as more eigenpairs can be found before it becomes slower than full matrix diagonalisation. However, this method is limited by the increasing time costs of converging all but the most extreme eigenvalues of the matrix and the infeasibility of calculating the “inner” eigenpairs of the matrix. Thus, to improve this spectral clustering algorithm further, the use of a number of

mathematical tools, more commonly known as preconditioning techniques are required, along with an improved approach to identifying the eigenpairs that is more suited for use with ECFPs.

The University Of Sheffield.

Chapter 7

SVD-based Spectral Clustering

7.1 Introduction

Experiments described in previous chapters have demonstrated that spectral clustering provides a viable alternative to traditional clustering methodologies in discriminating actives from inactives. Nevertheless, the biggest disadvantage that afflicts any spectral clustering approach is the large time and storage costs associated with the eigendecomposition procedure. In fact these costs can be so large that scaling the algorithm for use with large datasets can become simply not viable, for example, applying a full matrix diagonalisation procedure to a dataset that contains 100000 molecules. The aim of this chapter is to present an optimised spectral clustering approach based upon the use of a singular value decomposition algorithm, which is both faster and more scalable than the other eigendecomposition methods used within this project thus far.

This chapter begins with an introduction to singular value decomposition and its reformulation for use with eigenproblems, before moving to discuss the Tversky index and why it is of academic interest. The discussion then moves towards the library used to implement SVD-based spectral clustering and the optimisations it utilises to produce efficient and accurate eigenpairs approximations. Finally the experimental section of this chapter tests the accuracy and scalability of the SVD-NOSC method, before demonstrating its application to fragment-based drug discovery problems and comparing the results to those obtained using the k-means algorithm.

7.2 Singular Value Decomposition

An important innovation in linear algebra is the Singular Value Decomposition algorithm, or SVD, which is a commonly used method of matrix factorisation. SVD-based approaches are often used in solving a variety of mathematical problems, including linear least squares and matrix rank approximations, and are typically implemented according to the equation (Strang, 2003):

Equation 30: *Singular Value Decomposition Formula.*

$$A_{mn} = U_{mm} S_{mn} V_{nn}^{T}$$

Where A is a matrix of size m x n, U is a unitary matrix of size m x m, S is a matrix of size m x n containing the singular values and V^T is the conjugate transpose of an n x n unitary matrix V.

Our interest in SVD methods stems from its close association with eigendecomposition algorithms, such that for a symmetric matrix it can be shown that:

- The left singular vectors of A, i.e., the columns of U, are equal to the eigenvectors of matrix AA^T.
- The right singular vectors of A, i.e., the columns of V^T, are equal to the eigenvectors of matrix A^TA.
- The non-zero singular values of A, the diagonal elements of S, are the square roots of the non-zero eigenvalues of both AA^T and A^TA (De Lathauwer et al., 2000; Khademhosseini, 2002). As the eigenvalues are equal to the roots of the singular values, one can select either the positive or negative roots to represent the eigenvalues. In the case of SVD, the positive roots are always selected.

An example of how a basic SVD is implemented to identify the eigenpairs of an input matrix is provided in ***Appendix D***.

Interestingly, SVD methods can also be applied to non-symmetric eigenproblem through the use of an augmented matrix (Berry and Sameh, 1989; Berry et al., 2006), where:

$$A_{aug} = \begin{pmatrix} 0 & A^T \\ A & 0 \end{pmatrix}$$

and has eigenvalues $\pm\sigma_1, \ldots, \pm\sigma_n$ with corresponding eigenvectors to:

$$\frac{1}{\sqrt{2}} \begin{pmatrix} u_i \\ \pm v_i \end{pmatrix}$$

By using the augmented matrix, a solution that produces two distinct sets of eigenvectors linked through a common set of eigenvalues can be identified. This feature of using a SVD-based eigendecomposition can be exploited to allow the production of two sets of eigenvectors; providing the user with the ability to view the clustering from two perspectives by using an asymmetric measure, for example, the Tversky coefficient. It should be noted that in the case of symmetric matrices the two sets of eigenvectors, U and V^t, are equal.

7.2.1 The Tversky Index

The Tversky Index is a similarity measure calculated using the formula (which is also shown in ***Chapter 2***):

Equation 31: *Tversky Index.*

$$S_{AB} = \frac{c}{\alpha(a - c) + \beta(b - c) + c}$$

This measure contains two weighting functions, α and β, that are used to control the relationship between the three variables a, b and c. By setting α and β to be unequal, for example, 0.1 and 0.9 respectively, the Tversky index produces similarity scores where the similarity of molecule A to B does not equal that of B to A. This results in the production of a non-symmetric similarity matrix that can be used to calculate two distinct sets of eigenclusters when using SVD-based eigensolvers. Furthermore, other similarity measures can be probed since when both α and β are set to 0.5 the similarity scores are equal to those obtained when

using the Dice coefficient, and when both are set to 1, the similarity scores are equal to those of the Tanimoto coefficient (Leach and Gillet, 2007).

Our interest in the use of the Tversky index – and the asymmetric matrices it can be used to form – stems from the work of Senger, who showed that the Tversky coefficient could be used to provide interesting results in core hopping studies (Senger, 2009). Our aim is to identify if using the Tversky coefficient to identify asymmetric input matrices will allow us to probe two sets of clusters, one where a reference molecule, A, is treated as a superstructure (similarity of A to B) and another set of clusters where a reference molecule is treated as a substructure (similarity of B to A, where B is a query structure).

7.2.2 SVDLIBC

SVD algorithms are commonly used within a variety of mathematical operations due to their ease of optimisation, and as such, a number of programming libraries already exist for their implementation. In this chapter, we make use of the SVDLIBC library (Rohde, 2009) to carry out each eigendecomposition. A description of the SVDLIBC library, the SVD algorithm it contains and the each of the built-in optimisations of the algorithm are now provided.

SVDLIBC (Rohde, 2009) is a C/C++ library, based on SVDPACKC (Berry et al., 1993) and their predecessor FORTRAN libraries. Unlike these other libraries, SVDLIBC is designed solely for application in large sparse matrix problems and hence, employs a single SVD algorithm to carry out operations. This algorithm is called las2 and has consistently been shown to be the fastest algorithm available for the identification of singular values from large sparse matrices. The las2 algorithm is able to achieve this by incorporating a number of features aimed at providing a fast and stable solution to singular value problems, including:

- **Harwell-Boeing Input**

In previous spectral clustering implementations within this thesis, the input matrices have been very sparse, but remain held within a full matrix format. This leads to the algorithms incurring unnecessary storage costs, which decreases their scalability. In order to fully exploit the advantages of using highly sparse input matrices, the SVD-based spectral clustering algorithms harnesses a more computationally efficient storage form called the Harwell-Boeing – or HB – format. This is the most popular way of storing large sparse matrices in a computationally efficient way, in the form of a text file (Duff et al., 1992; MatrixMarket, 2011). Sparse matrices are converted to a file containing an 80-column fixed length format,

comprised of a 4 or 5 line header block. The information stored in each line is given in ***Figure 7.1***.

Line 1
Col. 1 - 72 Title
Col. 73 - 80 Key

Line 2
Col. 1 - 14 Total number of lines excluding header
Col. 15 - 28 Number of lines for pointers
Col. 29 - 42 Number of lines for row indices
Col. 43 - 56 Number of lines for numerical values
Col. 57 - 70 Number of lines for right-hand sides
(zero indicates no right-hand side data is present)

Line 3
Col. 1 - 3 Matrix type
Col. 15 - 28 Number of rows (NROW)
Col. 29 - 42 Number of columns (NCOL)
Col. 43 - 56 Number of row indices (NNZERO)
(equal to number of non-zero entries for matrices)
Col. 57 - 70 Number of elemental matrix entries (NELTVL)
(zero in the case of input matrices)

Line 4
Col. 1 - 16 Format for pointers
Col. 17 - 32 Format for row indices
Col. 33 - 52 Format for numerical values of coefficient matrix
Col. 53 - 72 Format for numerical values of right-hand sides

Line 5 (Only present if there is a right hand side matrix)
Col. 1 Right-hand side type:
F for full storage or M for same format as matrix
Col. 2 starting vector (if supplied).
Col. 3 exact solution vector (if supplied).
Col. 15 - 28 Number of right-hand sides
Col. 29 - 42 Number of row indices

Figure 7-1: *Shows the information contained in each line of the Harwell-Boeing Input Storage method (Duff et al., 1992; MatrixMarket, 2011).*

In this study all matrices have no right hand side information present, and hence line 5 is not applicable in these cases (Duff et al., 1992). This input format is used to minimise the volume of data held within the programs memory, improving the scope of the algorithm.

- **Lanczos Bi-Diagonalisation**

The Golub-Kahan-Lanczos bi-diagonalisation procedure, or LBD, is a method for solving sparse singular value problems through the use of a modified version of the Lanczos algorithm and an intermediate minimised matrix representation called a bi-diagonal matrix (Larsen, 2001). An example is shown in ***Figure 7.2***.

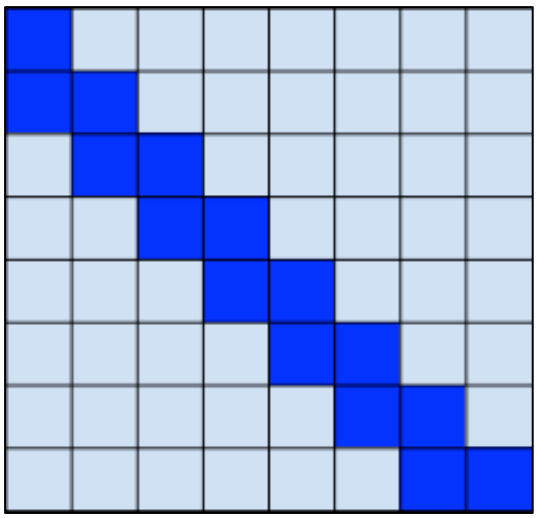


Figure 7-2: *Example of a bidiagonal matrix.*

LBD is implemented via an eight-step algorithm:

1. Select an initial vector p_0 and let $\beta_1 = \|p_0\|$, $u_1 = {}^{p_0}/_{\beta_1}$ and $v_0 \equiv 0$.

Now, for k = 0, 1, 2, …, n.

2. $r_k = A^T u_k - \beta_k v_{k-1}$
3. $\alpha_k = \|r_k\|$
4. $v_k = {}^{r_k}/_{\alpha_k}$
5. $p_k = A v_k - \alpha_k u_k$
6. $\beta_{k+1} = \|p_k\|$
7. $u_{k+1} = {}^{p_k}/_{\beta_{k+1}}$
8. Continue until k eigenvalues are identified.

Where A is an n x n input matrix, v is a vector from matrix V, u is a vector from the unitary matrix U, r and p are intermediate vectors, α is the diagonal element from the bi-diagonal matrix B and β is the first off diagonal element of B.

After k steps the decomposition is given by the equations:

$$AV_k = U_{k+1} B_k$$

$$A^T U_{k+1} = V_k B_{k+1}^T + \alpha_{k+1} v_{k+1} e_{k+1}^T$$

Where e is the residual error and V and U have orthonormal columns. From here the largest singular values of B_k converge rapidly to the largest singular values of A. For the sake of brevity, the reader is directed to SVDPACKC user's guide (Berry et al., 1993) and publications

by Jia (2003) and Larsen (1998) for more information and further discussion of the mathematics that underpin this algorithm.

- **Partial Reorthogonalisation**

Like all Lanczos implementations and adaptations, LBD is also plagued by the problems associated with a loss of orthogonality. To overcome these issues a reorthogonalisation step is required. In ***Chapter 6***, a full reorthogonalisation procedure was harnessed to ensure eigenvectors maintained a level of orthogonality equal to the roundoff error. Although this method provides accurate results, studies have shown that it is conservative, and in fact only a level of semi-orthogonality is required (Simon, 1984; Grimes et al., 1988). As such, there are faster methods for orthogonalisation, which sacrifice a level of accuracy in order to increase the speed of the method; one such procedure is called partial reorthogonalisation, PRO.

PRO is based on simple recurrence (Larsen, 1998) allowing the user to monitor the loss of orthogonality amongst Lanczos vectors directly by using the information from the recurrence. One should note that the term recurrence describes the relationship between values such that the second value is a function of the first. Thus, the vectors are reorthogonalised only when required, through the introduction of reorthogonalisation procedure after steps 2 and 5 of the LBD algorithm.

- **Implicit Restarts**

The final major optimisation used within SVDLIBC is an implicit restarting mechanism. Restarting mechanisms aim to minimise the time and storage costs of the algorithm by making use of one of the Lanczos algorithms biggest advantages, its ability to quickly and efficiently identify the first few eigenpairs of a matrix (Larsen, 2001). Usually this is carried out using an explicit mechanism where the method is stopped after an unspecified number of steps, a new start vector calculated and the algorithm subsequently restarted. In the case of the las2 algorithm (Berry et al., 1993), an implicit method is used. Here the term implicit denotes that the restarting method does not require the explicit recalculating of the start vector, but rather couples the LBD with an implicit QR/QL procedure directly, minimising the computational costs further.

7.3 Evaluating the Accuracy and Scalability of SVD-based Spectral Clustering

The aim of this section was to test both the accuracy and scope of the SVD-based spectral clustering approach. ***Section 7.3.1*** details the testing of the SVD-NOSC algorithm's ability to produce accurate eigenpair approximations. ***Section 7.3.2*** draws a comparison between the time requirements of the m-NOSC, L-NOSC and SVD-NOSC approaches, with ***Section 7.3.3***, showing the scalability of the algorithm through its application to datasets of increasing size.

7.3.1 Assessing the Accuracy of the SVD Eigenpair Approximations

Before SVD-based spectral clustering can be used for chemical problems, we must first be sure that the eigenpair approximations produced by this method are accurate. The Stahl COX2 dataset (Stahl and Rarey, 2001; Cheminformatics.org, 2010) was segmented into 10 clusters using the SVD-NOSC algorithm, the Tanimoto coefficient, standard Unity fingerprints, a γ value of 10 and a similarity threshold value of 0.01 applied to the elements of the filtered similarity matrix. After identifying the top 10 eigenpairs, molecules were placed into eigenclusters based on their largest eigenvector element. To ensure molecules were not placed into clusters due to extremely low eigenvector contributions, a threshold value of 0.00001 was applied. Eigenvector elements below this threshold were set to zero and not considered for the placement of molecules into eigenclusters. The clusters produced via the SVD-NOSC approach were compared to those obtained through the m-NOSC method implemented under identical parameters.

7.3.1.1 Results & Discussions

If the eigenpair approximations produced by the SVD-NOSC algorithm are accurate, one would expect both the SVD-NOSC and m-NOSC methods to assign molecules to identical clusters; providing that both methods are run using the same set of parameters. The clusters produced by both methods are given below:

Clusters formed using m-NOSC and SVD-NOSC

2.2408 - 5 12 13 14 15 20 24 25 28 29 48 49 54 81 82 84 104 105 106
2.3959 - 1 2 3 4 26 27 30 31 33 35 36 40 43 46 63 68 72 76 83 85 86 88 91 92 93 100 102 103 107 117 125
2.9347 - 62 73 77 78 79 80 87 118 119
3.5971 - 6 7 114 115 116
3.6205 - 11 90
3.7391 - 56 57 58 59 60 61 65 66 67 69 70 71 75
4.9041 - 23 94 95 96 97 98 99
5.2509 - 8 9 10 19 21 22 32 50 51 55 124
6.3735 - 16 17 18 34 37 38 39 41 42 44 45 47 64 89 101 120 121 122 123
6.9658 - 52 53 74 108 109 110 111 112 113

The identical clusters obtained for the two spectral clustering methods indicates that the SVD-based eigendecomposition algorithm produces accurate eigenpairs. The accuracy of these predictions can be further probed by looking at the contribution each molecule makes to the cluster it has been assigned to.

Tables 7.1 and ***7.2*** show the contribution of each molecule to the eigenclusters corresponding to the eigenvalues 6.96576 and 5.25085, respectively. Each contribution in these tables is given to 5 decimal places and was identical across both methods, which reinforces the conclusion that the eigenpairs generated using SVD-based spectral clustering are accurate.

Molecule ID	Contribution when using SVD-NOSC	Contribution when using m-NOSC
52	0.32445	0.32445
113	0.32445	0.32445
109	0.31154	0.31154
112	0.31154	0.31154
110	0.31118	0.31118
108	0.30847	0.30847
111	0.26325	0.26325
53	0.08335	0.08335
74	0.00629	0.00629

Table 7-1: *Table showing the contribution that each molecule makes to the eigencluster corresponding to the eigenvalue of 6.96576.*

Molecule ID	Contribution when using SVD-NOSC	Contribution when using m-NOSC
8	0.30189	0.30189
10	0.29335	0.29335
22	0.29331	0.29331
124	0.29129	0.29129
9	0.28077	0.28077
21	0.18030	0.18030
51	0.14531	0.14531
55	0.10848	0.10848
19	0.09816	0.09816
50	0.09750	0.09750
32	0.06515	0.06515

Table 7-2: *Table showing the contribution of each molecule to the eigencluster λ = 5.25085.*

Although not pertinent to this discussion, molecule 74 is based on a significantly different scaffold to the other compounds that populate the cluster λ = 6.96576, which is shown by its lower eigenvector contribution (an observation that was also made when testing the L-NOSC algorithm). By raising the eigenvector threshold value, for example to 0.01, molecule 74 can be removed from this cluster and instead will form a true singleton. This shows that careful consideration should be given to the magnitude of the eigenvector threshold to ensure molecules are not placed into clusters where their scaffolds differ significantly from the other molecules. The ideal magnitude of this eigenvector threshold will vary from dataset to dataset, so cannot be identified prior to clustering the data.

This study has shown that the SVD-based eigendecomposition algorithm used within our method is capable of providing accurate approximations of the eigenpairs. As a result, we can conclude that further research into the scalability and applications of SVD-NOSC are merited.

7.3.2 Comparing the Time Costs of the Spectral Clustering Methods

Although increasing the scalability of the algorithm in terms of the maximum size dataset that the method can be applied to is vital, an equally important consideration is minimising the associated time costs of the method. In this study the SVD-NOSC algorithm was applied to the four ChEMBL activity classes used throughout ***Chapters 4 – 6***, and the time required to identify varying number of clusters (k = 100, 200, 300, 400, 500 & 600) using each of the procedures was compared, when γ = 25, similarity threshold = 0.001 and eigenvector threshold = 0.001.

7.3.2.1 Results and Discussion

The respective times required to calculate k eigenpairs from each dataset and descriptor combination using the SVD-NOSC approach are presented in ***Table 7.3***. When these results are compared to those in ***Table 6.9***, it is very apparent that the SVD-NOSC method provides significantly lower time costs than those associated with both m-NOSC and L-NOSC methods. For example, the SVD-based spectral clustering method often identifies k eigenpairs in an order of magnitude faster than the other methods. This significant decrease in the time requirements can be attributed to the cumulative time savings gained through the use of the pre-conditioning techniques that are in-built in the SVD-based eigendecomposition.

Unlike the L-NOSC algorithm used in ***Chapter 6***, the SVD-NOSC method is able to utilise the high sparsity of the input matrices. This is highlighted by the faster times reported for ECFP_4 fingerprints which are considerably more sparse than MDL public keys and Unity fingerprints following the application of the Gaussian filtering function. The decrease in time for calculating the eigenpairs increases the viability of using spectral clustering as an alternative to other clustering algorithms. Crucially the algorithm's performance with ECFP_4 fingerprints is extremely encouraging, as its use with this descriptor has consistently produced impressive results.

Dataset	Fingerprint	k					
		100	200	300	400	500	600
5HT1A	BCI	9	15	21	29	47	64
	Daylight	10	17	23	31	51	68
	ECFP_4	8	16	20	26	34	43
	MDL Public Keys	9	16	23	33	53	73
	Unity	9	14	21	32	45	74
MMP1	BCI	18	28	38	56	88	119
	Daylight	17	26	35	44	61	89
	ECFP_4	13	23	35	41	49	65
	MDL Public Keys	18	25	38	46	69	94
	Unity	17	25	36	42	63	94
Renin	BCI	7	11	18	29	31	41
	Daylight	8	12	18	28	38	54
	ECFP_4	6	9	15	25	29	36
	MDL Public Keys	7	11	19	28	35	44
	Unity	7	11	18	30	46	51
SubP	BCI	10	15	21	34	55	66
	Daylight	11	17	24	34	48	74
	ECFP_4	8	14	21	30	36	46
	MDL Public Keys	9	16	23	36	56	76
	Unity	10	15	22	35	51	71

Table 7-3: *The time (in seconds) required to identify varying values of k eigenpairs using the SVD-NOSC method from four ChEMBL activity classes described by different molecular descriptors.*

7.3.3 Determining the Scalability of SVD-based Spectral Clustering

The next step in this investigation was to test the scalability of the method, by determining the maximum size dataset that can be clustered using SVD-based spectral clustering. The Pipeline Pilot software (Accelrys, 2011) was used to extract random subsets of N compounds from the MDL Drug Data Report, or MDDR, database (MDL, 2006), which contains 102513 biologically relevant molecules, meaning that the molecules have a known biological target or are a structural analogue of one of these aforementioned compounds. Note, the value of N in this study varies between 1000 – 102513 at regular intervals. Each dataset was clustered at two different values of k, 100 and 1000, using the SVD-NOSC approach, the Tanimoto coefficient and circular RDKit fingerprints, which are analogous to ECFP_4 fingerprints. The time required to cluster each dataset when γ = 100 and the similarity and eigenvector thresholds were both set to 1×10^{-6}, was recorded.

The time required for each value of k were compared and the effect that the γ parameter has on the results was considered. The results are presented in ***Table 7.4*** and graphically in ***Figure 7.3***.

Dataset Size / N	**k**	
	100	1000
	Time /s	
1000	2	2
2000	9	9
3000	21	22
4000	38	42
5000	58	71
7500	130	137
10000	237	417
20000	928	1340
30000	2192	2359
40000	3800	4187
50000	5932	6329
60000	8546	8976
70000	14421	15711
80000	24513	25679
90000	32784	33916
102513	40532	42765

Table 7-4: *Time taken to cluster random subsets of size N extracted from the MDDR database, when γ = 100, both the similarity and eigenvector threshold = 1 x 10^{-6} and k = 100 and 1000 respectively.*

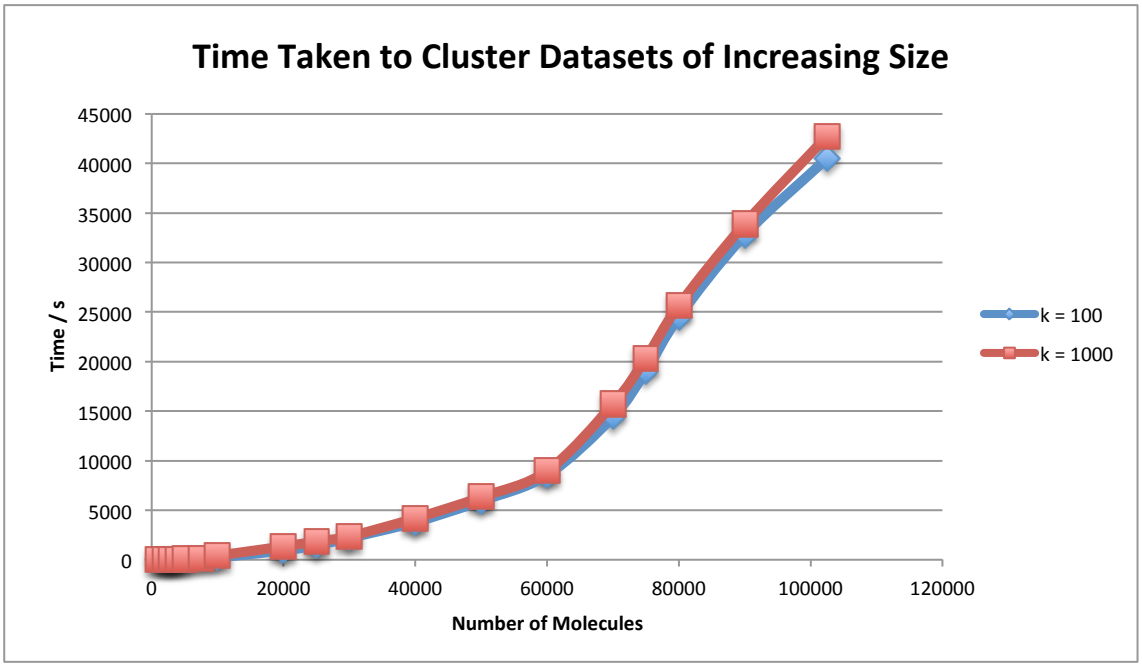


Figure 7-3: *Time taken to identify 100 and 1000 eigenpairs from random subsets extracted from the MDDR database.*

The results show that it is possible to apply the SVD-NOSC algorithm to chemical datasets where N exceeds 100000. This is a significant step forward in terms of the scalability of the spectral clustering approach presented in this work, which was previously limited to datasets where N ≈ 10000, due to the storage costs of using the L-NOSC and m-NOSC algorithms, see ***Chapters 5 - 6***.

When N and k are below 2000 and 200 respectively, the time requirement for identifying eigenclusters using the SVD-NOSC algorithm is comparable to those of the L-NOSC method. However, for identifying a large number of clusters, for example, 500 clusters or more, the SVD-NOSC algorithm is significantly faster than L-NOSC. For example, it takes 417 seconds to identify 1000 clusters from a dataset of 10000 molecules, see ***Table 7.4***, using SVD-NOSC, whereas it takes on average 27695 seconds to identify just 600 clusters using L-NOSC, see ***Table 6.11***. In fact the significantly lower time costs that are associated with the SVD-based eigendecomposition allows our spectral clustering approach to find a greater number of eigenpairs from a datasets of 80000 compounds faster than the L-NOSC algorithm can be applied to 10000 molecules.

The curves for both k = 100 and k = 1000, shown in ***Figure 7.3***, are very similar and highlight the rapid growth in the time taken to cluster a dataset as the value of N increases. The steep gradient is typical of any eigendecomposition algorithm, which typically require greater than N operations to complete. Although time requirements of spectral clustering rise quickly with

the value of N, the increase in the time with increasing k is small compared to other eigendecomposition algorithms used in this study. For example, the difference in the time required to find 100 and 1000 eigenpairs when N = 102513 is only 2223 seconds. This is encouraging as it shows that an apt number of eigenpairs can be found to cluster large datasets without the user incurring significantly increased time costs.

The results provided in ***Table 7.4*** illustrate the increased capacity of SVD-based spectral clustering along with the decreased time costs. However the scalability of the algorithm is not the only important factor that must be considered; for this research to be of practical use, the clustering algorithm must produce a set of clusters that are chemically sensible/interesting whilst maintaining the increased scalability.

In this study γ was set to 100 to ensure that timings were easily comparable to each other. However, the use of such a high γ value in the Gaussian filtering function has a significant effect on the clusters produced from the datasets. While dealing with larger datasets, for example, those where N > 10000, the use of a γ value of 100 provides an intuitive set of results where the mode of the cluster size is 13 for k =1000. When the value of N is decreased to below 10000, the quality of the clusters quickly decreases, such that the modal cluster size falls to 3 with many compounds left unclassified. By decreasing the value of γ to 10 for datasets where N < 10000, the quality of the clusters can be improved significantly. An example of this is given in ***Figures 7.4*** and ***7.5***, which show some of the clusters produced for a dataset of 5000, when γ = 100 and 10 respectively.

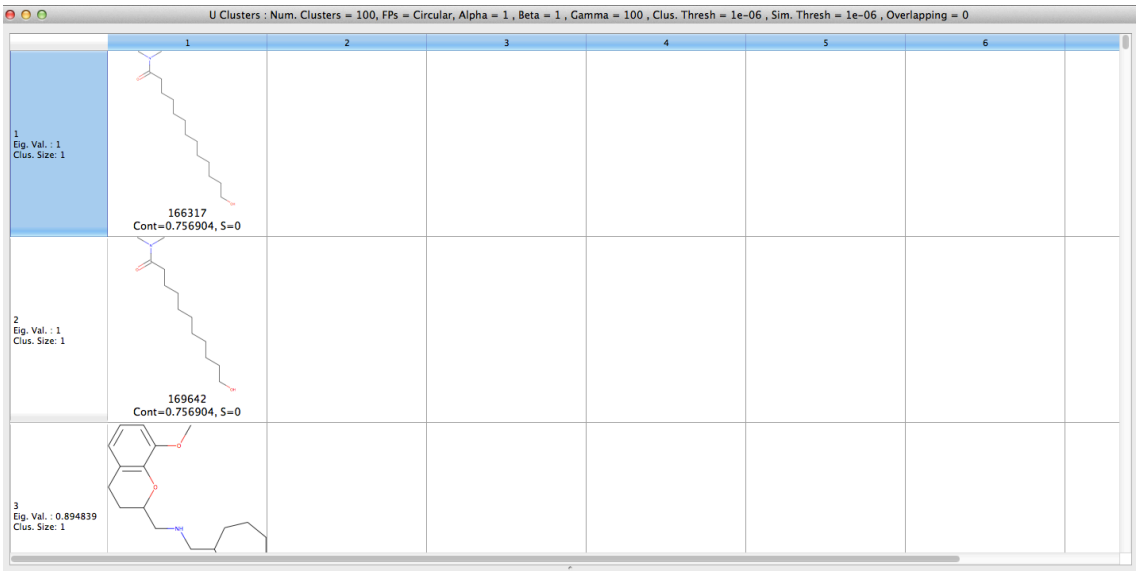


Figure 7-4: *Example of clusters produced for the dataset containing 5000 molecules when γ = 100.*

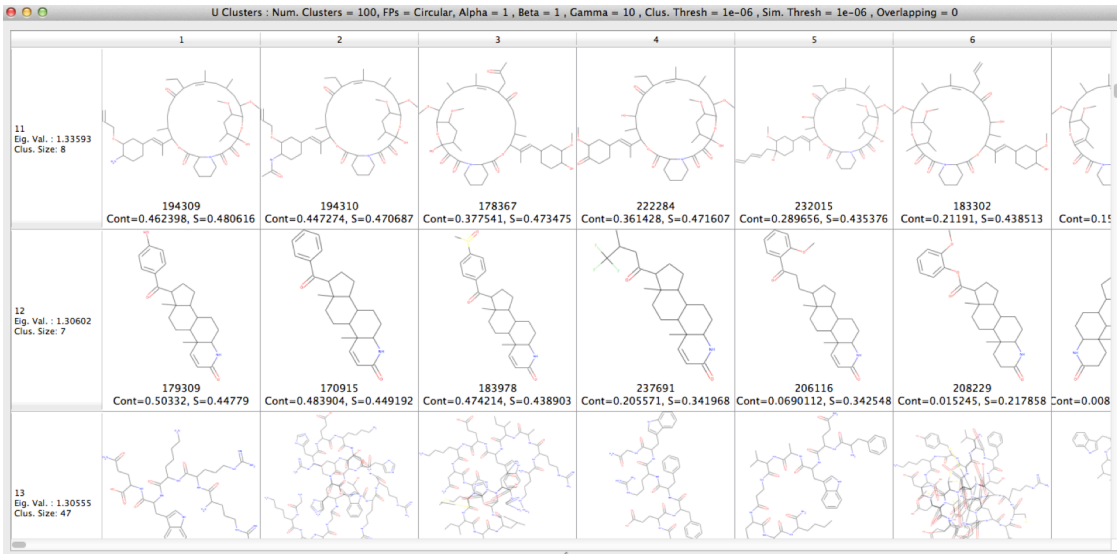


Figure 7-5: *Example of clusters produced for 5000 molecule MDDR dataset when γ = 10.*

When γ = 100 the molecules are distributed into clusters containing 5 molecules or less, with many molecules being placed into singleton clusters or being not classified at all (see ***Figure 7.4***). This is obviously an unsatisfactory set of clusters that are of no use from a practical perspective. Conversely, when γ = 10 a more intuitive set of clusters is produced, with molecules being grouped with those they share scaffold features with, see ***Figure 7.5***. This shows that the choice of γ remains an important consideration, and must be adjusted to reflect the size of the dataset.

7.3.4 Conclusion

The use of a SVD-based eigendecomposition method within the spectral clustering algorithm is a significant improvement over the previous iterations of this algorithm that harnessed a full matrix diagonalisation procedure and the Lanczos algorithm respectively. ***Section 7.3.1*** demonstrated that SVD-NOSC was able to approximate the eigenpairs with a high level of accuracy and ***Section 7.3.2*** showed that the eigenpairs could be calculated at a significantly faster rate than the previous methods. Furthermore, the results of the investigation in ***Section 7.3.3*** shows that by using the SVD-based eigendecomposition algorithm within the spectral clustering method, the scalability can be increased by an order of magnitude. The minimal difference in the time requirements for finding 100 and 1000 eigenpairs, points to this method scaling more favourably with k than the L-NOSC algorithm, increasing its possible applications. Finally, the choice of magnitude for γ in the Gaussian filtering function continues to have a significant effect on the performance on the clustering algorithm, and therefore must be monitored carefully to ensure that the clusters produced are useful.

7.4 Applying SVD-Based Spectral Clustering to FBDD Problems

The aim of this section was to demonstrate the use of the SVD-NOSC algorithm in two fragment-based drug discovery problems. ***Section 7.4.1*** begins by providing a brief overview of what is meant by the term fragment-based drug discovery, along with its uses and why it is a significant area for research within the pharmaceutical industry. Next, the first investigation in this section, examines the clusters produced using the SVD-NOSC method and compares them to both an ideal set of clusters and those produced using the k-means algorithm. Before, ***Section 7.4.3*** assesses SVD-NOSC's effectiveness in separating active and inactive fragments for a variety of parameters.

7.4.1 An Introduction to Fragment-Based Drug Discovery

Since the 1990s the pharmaceutical industry has predominantly followed a target-based approach to drug discovery, where biological targets, for example, a set of genes that are shown to have a causative role in the onset or progression of a particular disease, are initially identified (Congreve et al., 2008). Following the identification of a target, the current drug discovery process, outlined in ***Chapter 1***, aims to identify new therapeutic moieties through the screening of compound libraries to determine a set of compounds that show activity in modulating the desired target. Those lead compounds are then optimised to enhance certain favourable features, for example, potency and ADMET properties (Lipinski and Hopkins, 2004). Despite numerous scientific and technological developments that have improved this process, target-led drug discovery is yet to reverse the continuing fall in the number of novel medicines that are brought to market each year (Sams-Dodd, 2005). To arrest this trend, the pharmaceutical industry began to search for other approaches to drug discovery that may yield greater results. One approach that has gained particular interest is called fragment-based drug discovery, FBDD (Congreve et al., 2008).

In FBDD, the term fragment typically refers to a molecule that contains 12 or fewer non-hydrogen atoms (Congreve et al., 2008). Significantly, studies by both Jencks (1981) and Ariens (1982) showed that any drug-like moiety is comprised of at least two fragments, and it is this fundamental concept that underpins all research into fragment-based techniques.

FBDD involves identifying lead compounds through the screening of small molecules, or fragments, for activity against a particular target (Rees et al., 2004). The subsequent steps in this process are similar to those of a target or structure-based approach, including both

optimisation and testing of the molecules. The success of FBDD relies on two central tenets to distinguish it from target or structure led approaches:

1. The proportion of the relevant chemical space that can be screened for fragment sets is significantly larger than for compound libraries. This is because the estimated chemical space that can be formed from fragments is approximately 10^7 molecules in size, whereas drug-like compound space, i.e. the chemical space occupied by molecules containing 30 non-hydrogen atoms or fewer, is in the order of 10^{60} molecules. Hence, screening 10000 fragments allows a greater proportion of fragment-like chemical space to be probed, than the screening 10000 molecules, would allow from drug-like space (Rees et al., 2004).

2. Fragments typically exhibit lower binding affinities to their target proteins than larger molecules, while maintaining an equal or greater binding efficiency per atom (Congreve et al., 2008).

Due to these advantages, FBDD has seen a continual rise in its implementation in both pharmaceutical and academic institutions (Rees et al., 2004; Congreve et al., 2008). However, applying Chemoinformatics techniques to FBDD data is not without problems. One such issue occurs when using the Tanimoto coefficient. The characteristics of the Tanimoto coefficient mean that a similarity score produced using this measure is directly affected by the number of bits set. Thus, the similarity between two large molecules, which set a large number of bits, is disproportionately higher than the similarity between two small compounds that inherently set a smaller number of bits (Willett et al., 1998; Leach and Gillet, 2007). Hence, to ensure the experiments in this chapter are not affected by bias, other similarity measures will be considered.

7.4.2 Evaluating the use of SVD-Based Spectral Clustering for Clustering Fragments

The DC100 fragment set, which contains 100 chemical fragments extracted from FBDD screens carried out at AstraZeneca, was manually clustered into 14 clusters representing each of the scaffolds contained in the dataset (see ***Appendix E***). To ensure that the sets of clusters produced for the dataset by each algorithm and parameter set were comparable, a consistent value of k = 14 was used in further experiments.

The DC100 dataset was clustered with the SVD-NOSC algorithm using two different fingerprint types, RDKit Circular and RDKit Linear (Landrum, 2006), and several different similarity measures. To control the choice of similarity measure the magnitude of the two weighting values in the Tversky index, α and β, were altered. The different combinations of α and β, and the similarity measure produced by the respective combinations are given in ***Table 7.5***.

α	β	Similarity Measure
1	1	Tanimoto
0.9	0.1	Asymmetric Tversky
0.8	0.2	Asymmetric Tversky
0.7	0.3	Asymmetric Tversky
0.6	0.4	Asymmetric Tversky
0.5	0.5	Dice

Table 7-5: *α and β combinations used with the Tversky index in this study.*

By ensuring that different similarity measures are used, any possible bias caused by the use of the Tanimoto coefficient could be examined. When using a version of the asymmetric Tversky index, two sets of clusters were produced by SVD-NOSC, one corresponding to the matrix of eigenvectors U and the other to the matrix of eigenvectors V. Each set of eigenvectors provides a different set of clusters that are based on looking at the eigenpairs from different perspectives. In this study we expect one set of fragments to produce a set of clusters where the reference structure is viewed as a superstructure and another where the reference structure is viewed as a substructure.

All spectral clustering implementations in this experimental section use a γ value of 10, which was determined to produce the most chemically interesting clusters through pre-experiments. Each descriptor and similarity measure was tested at three values of both the similarity and eigenvector threshold (0.01, 0.001, 0.0001).

The clusters produced for each SVD-NOSC implementation were compared to those of the ideal clustering using the Jaccard coefficient in order to assess the success of the SVD-NOSC algorithm in clustering the fragment class. Further clustering of the DC100 fragment set was carried out for both fingerprint types using the Tanimoto coefficient coupled with the k-means method, for comparison.

7.4.2.1 Results and Discussion

Table 7.6 shows the results of the Jaccard comparison between the different clustering methods using RDKit Circular fingerprints. Due to the small effect that varying the magnitude of both thresholds has on the results of the Jaccard comparisons, only the results formed from the clusters when both the similarity and eigenvector threshold equal 0.0001 are presented. Full results tables can be found in ***Appendix F***.

Clustering 1	Clustering 2		Jaccard	
	α	β	U	V
ideal	1	1	0.604	0.604
	0.9	0.8	0.655	0.627
	0.8	0.2	0.627	0.627
	0.7	0.3	0.627	0.627
	0.6	0.4	0.627	0.612
	0.5	0.5	0.645	0.645
	k-means		0.612	0.612
k-means	1	1	0.630	0.630
	0.9	0.8	0.575	0.627
	0.8	0.2	0.601	0.627
	0.7	0.3	0.601	0.601
	0.6	0.4	0.627	0.586
	0.5	0.5	0.645	0.645

Table 7-6: *Results of the Jaccard comparison between the different SVD-NOSC implementations and the ideal/k-means clustering of the dataset, when using RDKit Circular fingerprints.*

When comparing a set of clusters to the ideal case, the performance of an algorithm can be measured by how closely the clusters it produces mirror the ideal set. Therefore, the Jaccard score produced from the comparison of a set of clusters to the ideal case can be used as a measure to quantify the relative quality of a set of clusters. Examining the Jaccard scores, given in ***Table 7.6***, indicates that there is a significant overlap between the clusters formed using the SVD-NOSC method and the ideal set. The clusters produced using both the asymmetric Tversky index and Dice coefficient provide an improvement over those calculated via the Tanimoto coefficient. This difference in the relative Jaccard scores indicates that the Tversky/Dice measures are able to highlight certain chemical relationships to a greater extent than the Tanimoto coefficient by altering the emphasis placed on the similarity scores.

Comparing the clusters visually shows that the Tanimoto coefficient struggles to form a set of clusters that accurately reflect the distribution of molecular scaffolds, as many molecules were placed either in small clusters containing 2 - 4 molecules or within an extremely large cluster

containing 64 of the compounds (see ***Figure 7.6***). Interestingly, the only similarity measure that leads to the production of singleton clusters is the Tanimoto coefficient, which produced two singleton clusters (see ***Figure 7.7***) for the two largest eigenvalues. This is unusual, as typically the largest eigenvalue is associated with a cluster containing several highly related molecules. The poor performance achieved when using the Tanimoto coefficient can be explained by its documented bias towards assigning lower than expected similarity scores to small fragments. Furthermore, these issues may be diminished when dealing with larger datasets, as the effect of the Gaussian filtering function is magnified, leading to a wider spread of the similarity values and hence, the production of a more chemically intuitive set of clusters.

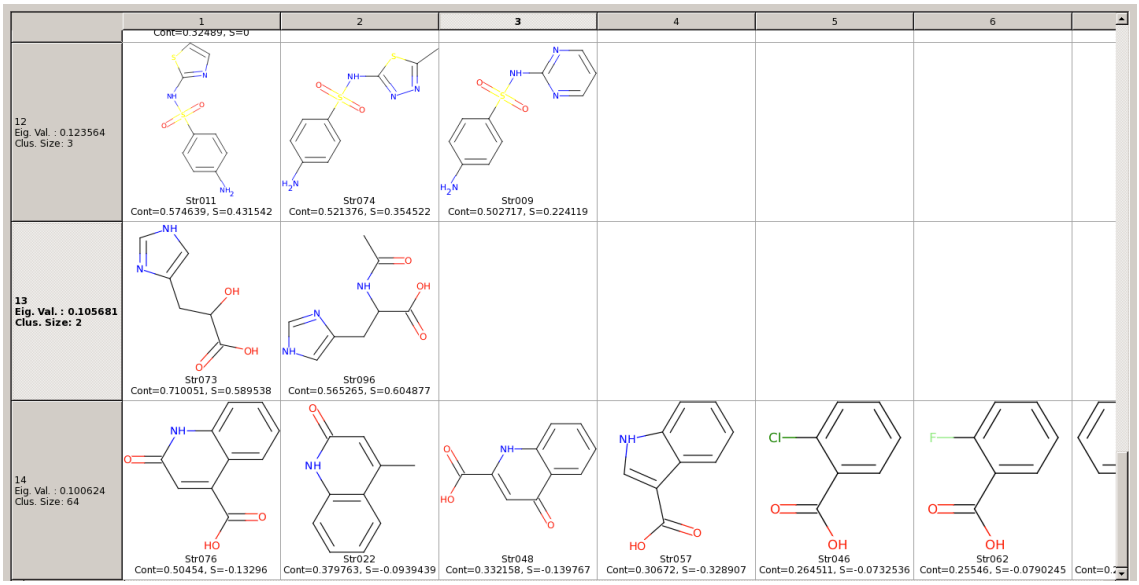


Figure 7-6: *Screen shot showing the large cluster containing 64 molecules produced by the application of the SVD-NOSC algorithm to the DC100 fragment set described by RDKit circular fingerprints*

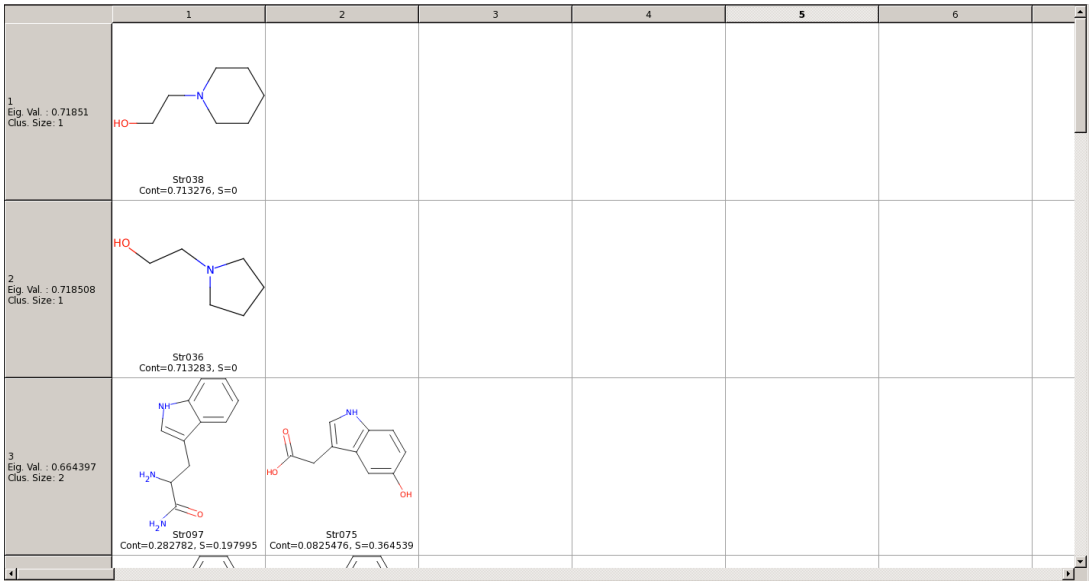


Figure 7-7: *Screen shot showing the two singleton clusters produced by the application of the SVD-NOSC algorithm to the DC100 fragment set described by RDKit circular fingerprints.*

Conversely the clusters produced for other similarity measures contain a greater distribution of compounds, with cluster sizes generally varying between 5 and 13 molecules, as shown in ***Figure 7.8***. Within these clusters, molecule assignments can be quite erratic, for example, molecule 94 in the clusters for the matrix U when α = 0.8 and β = 0.2 (as shown in ***Figure 7.8***), is assigned to a cluster where it shares little commonality with the other constituents of the cluster. These misclassifications also occur when using other clustering methods such as the k-means algorithm and highlight the difficulty of accurately clustering fragments.

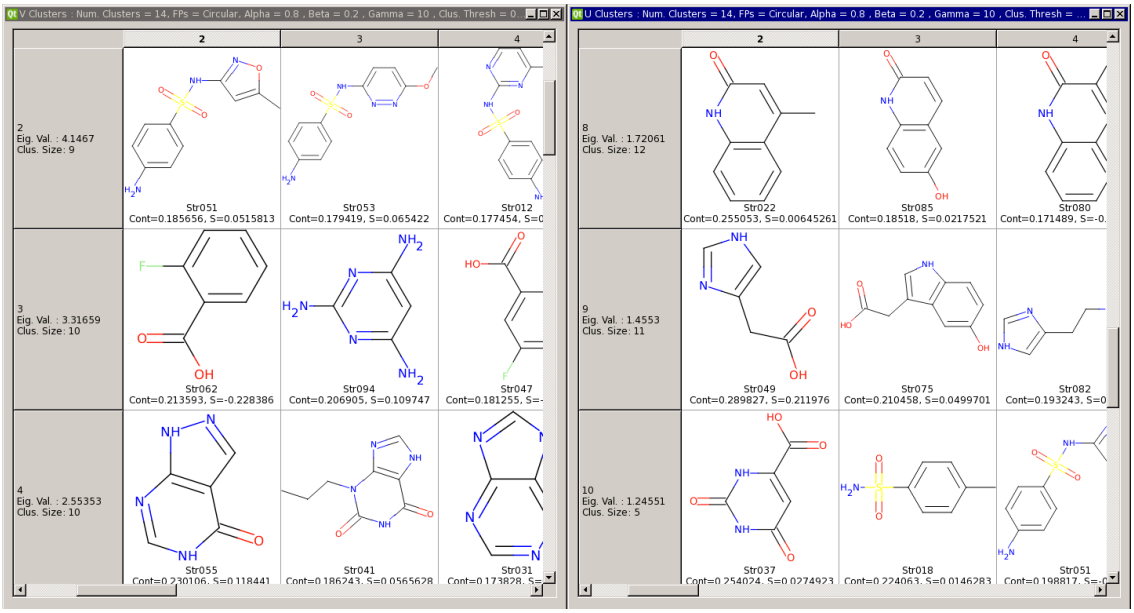


Figure 7-8: *Screen shot showing the two sets of clusters (U and V) produced when α = 0.8 and β = 0.2.*

The asymmetric Tversky coefficient was shown to produce two distinct sets of clusters when used with the SVD-NOSC algorithm. These two sets of clusters, although distinct, share a significant level of overlap that is often greater than the overlap shared between two non-related sets of clusters, yet is still low enough for the clusters to be considered complementary. For example, when α = 0.9 and β = 0.1, the comparison of the two sets of clusters produce a Jaccard score of 0.670, showing that these sets are both highly related yet still provide complementary perspectives on how the molecules should be grouped. The large Jaccard scores are the result of the both sets of clusters grouping the most highly related compounds together. However, examination of the contribution each compound makes to the two sets of eigenvectors shows that the compounds are grouped based on significantly different contributions. This highlights the differences between treating the reference structure as superstructures or substructures. These observations are also correct for the clusters produced using the other versions of the asymmetric measure, and emphasises how the use of the Tversky measure provides an interesting set of results.

Varying the eigenvector and similarity thresholds by an order of magnitude had little effect on the clusters produced (see ***Appendix F***), with a variation in the similarity threshold by two orders of magnitude to 0.01 only having a minimal impact on a few of the sets of clusters produced by the SVD-NOSC implementations.

Comparing the ideal set and the k-means clusters provided a Jaccard score of 0.612. Although this score is comparable to that obtained for SVD-NOSC using the Tanimoto coefficient, it is not as large as the scores generated when using the Tversky index or Dice coefficient in most instances. This indicates that a small improvement in the clustering of the dataset can be obtained through the coupled use of the SVD-NOSC algorithm and an asymmetric variant of the Tversky index. By comparing the SVD-NOSC clusters to those generated using k-means, see ***Table 7.6***, some degree of overlap between the two methods is apparent. This level of overlap is significant as it can be used to show if the methodologies are complementary. Typically one would expect to see a large similarity between the results of two algorithms, as both should, in theory, be producing clusters that are similar to the ideal case. However, for two methods to be considered complementary the level of similarity must be low enough to allow for both methods to give the chemist a better overview of the ideal case when they are considered in unison. Although these approaches show a high level of overlap, producing Jaccard scores between 0.575 and 0.645, indicates that the SVD-NOSC and k-means algorithm generates results that are complementary. When comparing the clusters visually, it can be seen that the different clustering algorithms place emphasis on differing features and hence produce clusters that are fundamentally different yet equally valid from a chemists point of view, see ***Figure 7.9***.

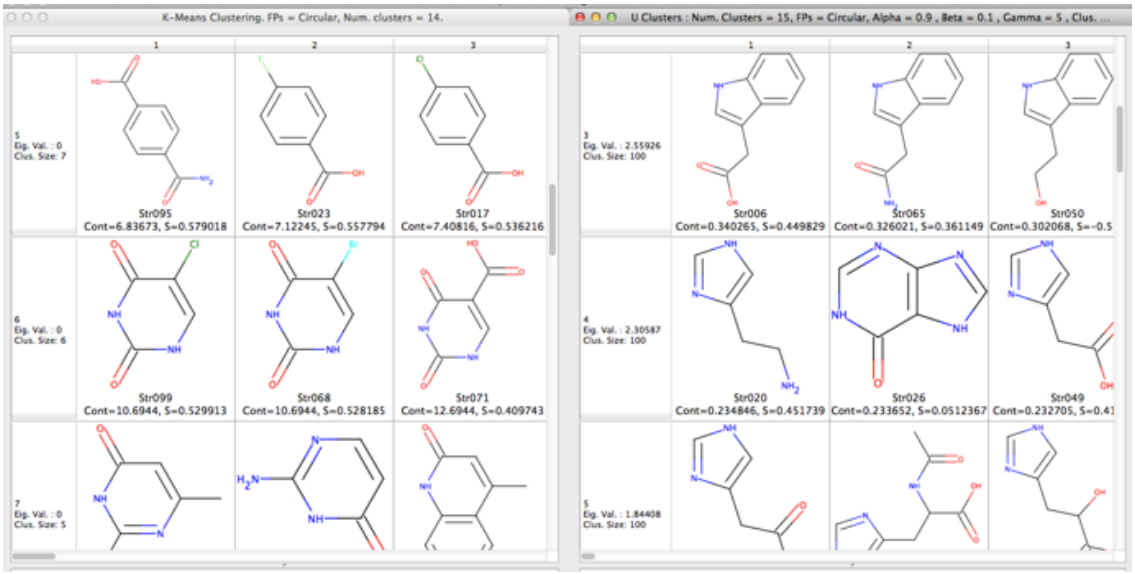


Figure 7-9: *Figure highlighting the difference in the scaffold that clusters are based upon for the k-means and SVD-NOSC approaches.*

The results produced using the RDKit Linear fingerprints, see ***Table 7.7***, highlights both similarities and differences between the respective performances of the two fingerprint types. Comparing the clusters produced using SVD-NOSC to those of the ideal clustering shows a relationship between the assignment of molecules in both approaches. However, the correlation for linear fingerprints is lower than that exhibited for RDKit Circular fingerprints. An explanation for the difference in the correlations is that circular fingerprints provide a greater ability to discern between molecules and therefore are able to segment molecules that share features with two or more scaffolds to a greater extent.

Clustering 1	**Clustering 2**		**Jaccard**	
	α	**β**	**U**	**V**
ideal	1	1	0.586	0.586
	0.9	0.8	0.575	0.612
	0.8	0.2	0.601	0.667
	0.7	0.3	0.627	0.601
	0.6	0.4	0.627	0.586
	0.5	0.5	0.586	0.586
	k-means		0.639	0.639
k-means	1	1	0.639	0.639
	0.9	0.8	0.575	0.612
	0.8	0.2	0.627	0.586
	0.7	0.3	0.655	0.601
	0.6	0.4	0.627	0.586
	0.5	0.5	0.667	0.667

Table 7-7: *Results of the Jaccard comparison between the different SVD-NOSC implementations and the ideal/k-means clustering of the dataset, when using RDKit Linear fingerprints.*

The clusters produced using the asymmetric Tversky index are again closer to those obtained for the ideal clustering, than the clusters generated for the two symmetric similarity measures. When using the Tanimoto and Dice coefficients with linear fingerprints, the relative successes of both algorithms in clustering the fragment class were equal. Whereas the Dice coefficient was shown to produce a minimally improved clustering of the data when Circular RDKit fingerprints were selected as the molecular descriptors. The two sets of clusters produced from the Tversky index continue to also be distinct and complementary with the two sets of clusters produced when $\alpha = 0.9$ and $\beta = 1$ producing a Jaccard score of 0.623.

Varying the magnitude of the similarity and eigenvector thresholds, between 1×10^{-3} and 1×10^{-4}, had no effect on the classification of molecules (see ***Appendix F***). This can be explained by considering how these values act upon the respective matrices. If we first consider the similarity threshold, this cut off value is applied to the filtered similarity scores in order to

make the input matrix sparse. When applied to these filtered similarity scores, the spread of the data caused by the Gaussian filtering function leads to the large similarity scores being represented by filtered similarity scores above 0.001 and low similarity scores being less than 0.0000001. Hence varying the similarity threshold between 1×10^{-3} and 1×10^{-4} leads to very few additional values being included. In the case of the eigenvector threshold, the size of the dataset leads to the production of eigenvectors where the molecules that contribute to a cluster have large eigenvector elements. Therefore the eigenvector threshold values are too high to include the extremely low contributing molecules and too small to significantly effect the molecules that are assigned to cluster based on large scores. Although these threshold values failed to have a noticeable effect on the clusters produced, it was important to analyse the effect they have on the clusters, as the molecules they act upon can provide interesting changes to the results.

The Jaccard value produced for the comparison of the k-means and ideal clusterings of the DC100 dataset was similar to the scores obtained for the various SVD-NOSC implementations. Furthermore, the comparison of the clusters produced using the k-means and SVD-NOSC approaches, presented into ***Table 7.8***, generated scores in the range of 0.575 to 0.655. This highlights the high level of overlap between the clusters produced by both methods, however these values remain low enough to allow the two methods to continue to be considered complementary.

This investigation has shown that the use of different similarity measures coupled with the SVD-NOSC algorithm is worth further consideration and investigation, as the use of the Tversky index and Dice coefficient consistently produced superior results than those obtained when using the Tanimoto. Furthermore, these experiments continue to provide evidence that spectral clustering can be used both as a valid alternative and also a complementary method to the k-means algorithm for a variety of purposes.

7.4.3 The Clustering of a Fragment Activity Class

The clustering of chemical fragments, or HTS data, is one of the most common applications for clustering algorithms, where the aim is to group active compounds together based on their chemical structure. In this investigation, the SVD-NOSC algorithm was applied to the problem of clustering a fragment dataset and its performance quantified using the QCI measure. The fragment class used contains 741 fragments that were screened as part of an assay against a

single biological target at AstraZeneca. For obvious reasons we cannot disclose this target and therefore shall refer to this dataset as *Target X* from herein.

The Target X dataset was clustered using the SVD-NOSC algorithm at three different values of k (25, 50 and 75), three different values of γ (5, 10 and 15) and several different values of both α and β in the Tversky index (outlined in ***Table 7.5***). For all experiments, similarity and eigenvector threshold values were set to 0.0001; these values were determined in the pre-experimental stage to ensure both the input and eigenvector matrices were dense enough to cluster the data effectively. The clusters calculated for each implementation were analysed using the QCI measure and compared to those obtained with the k-means method.

7.4.3.1 Results & Discussion

Table 7.8 provides the respective QCI scores produced by each application of the SVD-NOSC method to the Target X fragment class using RDKit Circular fingerprints. Also contained in these tables are the QCI values for the k-means clustering of the Target X fragment class using identical values of k as used in the SVD-NOSC implementation.

A key observation from ***Table 7.8*** is that as the number of clusters, k, increases there is a rise in the respective QCI scores. This is the result of larger clusters being broken down into multiple smaller clusters that meet the requirements to be classed as active, increasing the value of p in the QCI calculation, and hence the overall QCI score. A similar observation can be made about the increase in the value of γ, however careful consideration must be given to if the rise in the QCI score is the result of the γ value excluding molecules from the clusters.

k	α	β	γ	QCI		γ	QCI		γ	QCI	
				U	V		U	V		U	V
25	1	1	5	31.29		10	34.56		15	37.45	
	0.9	0.1		32.55	29.06		27.73	31.46		29.67	28.38
	0.8	0.2		33.78	31.61		30.43	31.94		30.16	29.40
	0.7	0.3		32.47	31.25		30.90	30.99		31.19	29.18
	0.6	0.4		32.52	30.91		31.03	29.51		33.20	29.59
	0.5	0.5		32.50			33.98			35.67	
	k-means			38.78			38.78			38.78	
50	1	1		37.26			35.91			40.35	
	0.9	0.1		33.40	32.79		32.26	34.13		34.15	33.42
	0.8	0.2		33.48	36.32		34.25	33.91		36.70	34.47
	0.7	0.3		34.86	36.99		33.16	35.15		36.26	34.73
	0.6	0.4		33.92	36.98		34.76	34.03		35.53	32.53
	0.5	0.5		35.39			38.46			39.90	
	k-means			41.26			41.26			41.26	
75	1	1		39.94			37.76			40.72	
	0.9	0.1		35.99	36.99		36.22	35.94		35.58	36.18
	0.8	0.2		36.45	38.30		36.96	34.98		36.43	37.11
	0.7	0.3		36.98	38.73		35.91	37.70		37.16	35.07
	0.6	0.4		37.47	37.78		34.96	36.36		37.22	34.16
	0.5	0.5		38.02			38.06			39.23	
	k-means			41.20			41.20			41.20	

Table 7-8: *QCI scores obtained for the clustering of the Target X dataset using RDKit Circular descriptors.*

The results contained in ***Table 7.8*** show that the choice of similarity measure has can have a sizeable effect on the QCI scores produced for the spectral clustering of the Target X fragment set. In contrast to the findings in ***Section 7.5.2***, the use of spectral clustering coupled with the Tanimoto coefficient was able to provide good results in clustering the Target X fragment class, out performing the asymmetric Tversky index in many instances. A visual inspection of the clusters produced using the Tanimoto coefficient shows that logical clusters appear to be formed from the Target X dataset, without the presence of any large clusters that encompass multiple scaffolds and skew the results, as seen in ***Section 7.5.2***. An example of the fragment assignments produced using the Tanimoto coefficient is provided in ***Figure 7.10***. In each figure the activity of a molecule is denoted by its colour: highly active compounds being shaded bright green and highly inactive molecules being coloured bright red, with the lighter shades of each colour providing a scale of how active a compound is.

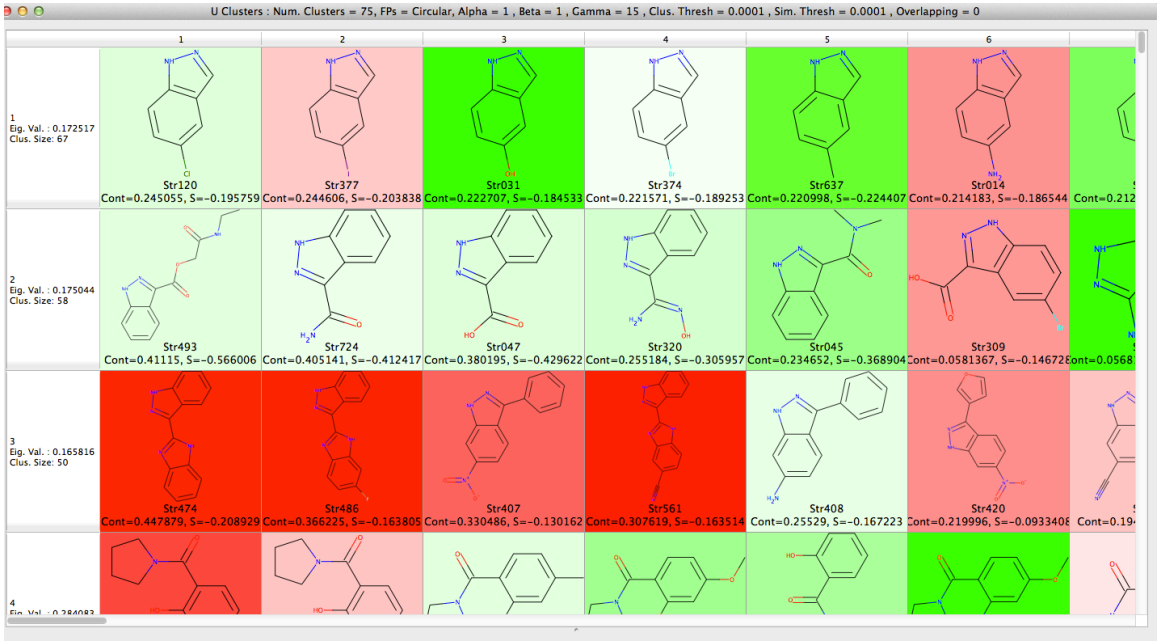


Figure 7-10: *An example of the clusters produced using the SVD-NOSC algorithm coupled with the Tanimoto coefficient, when k = 75 and γ = 10.*

In ***Figure 7.10***, four clusters produced using the Tanimoto coefficient and spectral clustering are shown. Each of the clusters can be described as chemically intuitive because of the obvious chemical relationships apparent within each cluster. For example, the first three clusters each incorporate an indazole ring system within their structures, which can lead to some clustering approaches agglomerating these clusters, as they share a significant amount of overlap. However, the SVD-NOSC approach bases its classification of molecules on more subtle changes to this indazole ring system, with cluster 1 containing indazole-based fragments that have a single branching point from their benzene ring; cluster 2 groups those compounds that contain

a ketone group branching from their pyrazole group; and finally cluster 3 groups indazoles that are attached to further ring/ring systems through a branching point on their pyrazole rings. Interestingly, the addition of the extra ring systems, the scaffold of which differs in the case of each molecule, decreases the activity of these fragments, suggesting that the increase in size of the compounds leads to a decrease in the activity. However, the application of the spectral clustering coupled with the Tanimoto coefficient is not without issue. When using a γ value 15, a number of molecules are left unclassified by the method, with 460 molecules clustered when k = 25, 606 when k = 50 and 664 when k =75. This leads to the conclusion that the Tanimoto coefficient is not suitable for use when γ > 10, as the decrease in the number of classified fragments is both undesirable and means that the QCI scores are not comparable to other QCI scores calculated from the clusters associated with using other similarity measures.

The application of spectral clustering coupled with the different versions of the asymmetric Tversky index continued to produce two distinct sets of clusters that despite showing a significant amount of overlap (producing Jaccard scores of around 0.7), provide two very different and complementary perspectives in the data. In general the performance of SVD-NOSC coupled with the Tversky index provided QCI scores lower than those achieved using symmetric measures. However the interlinked nature of the two sets of clusters produced, generated some interest from medicinal chemists at AstraZeneca, and suggests that further research on the use of the Tversky index is merited. ***Figure 7.11*** provides an example of the two distinct sets of clusters calculated from the application of the SVD-NOSC method using the asymmetric Tversky coefficient.

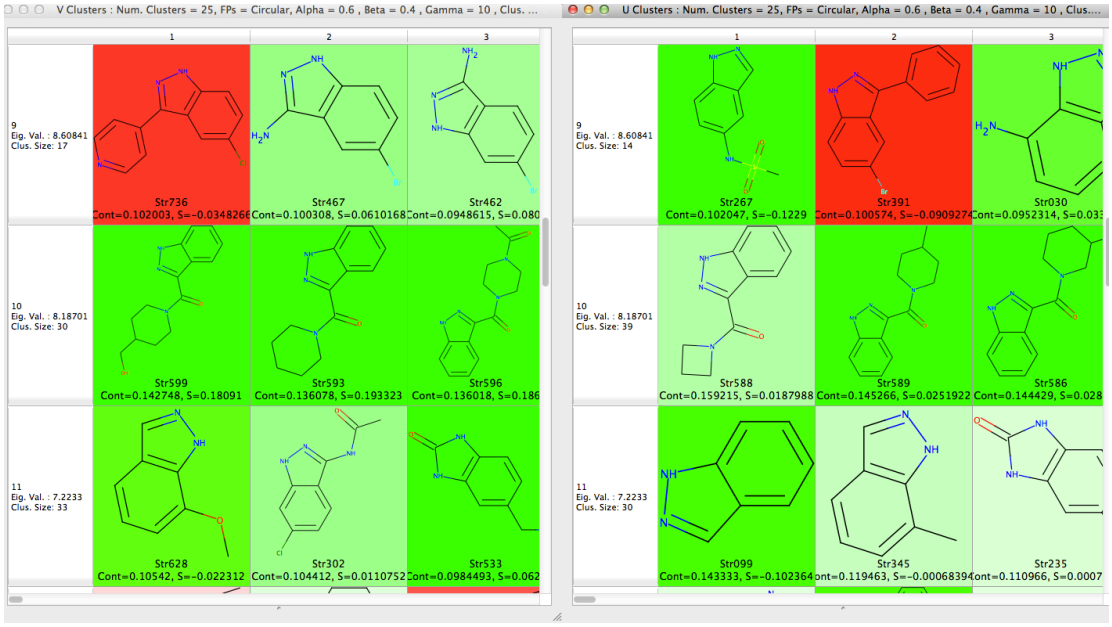


Figure 7-11: *An example of the two sets of clusters (U and V) generated using the SVD-NOSC method and the Tversky index, where α = 0.6, β = 0.4, γ = 10 and k = 25.*

In general, the results generated using the Dice coefficient continually provided the largest (or close to the largest) QCI scores for each spectral clustering implementation. Unlike the Tanimoto coefficient, the increase in the QCI scores with the Dice coefficient, when γ and k are increased, are not due to the algorithm leaving molecules unclassified or through the generation of a single large cluster, and instead are the results of more clusters meeting the requirements to be considered active under the metric set out by Varin et al. (2008), see ***Section 4.6.1***. Visually inspecting the clusters produced using the Dice coefficient and spectral clustering show that, in general, molecules are classified into clusters where they share a significant amount of scaffold features with the other constituents, for example see ***Figure 7.12***. The encouraging performance of the spectral clustering with the Dice coefficient, leads to the conclusion that the Dice coefficient merits further investigation for use within other chemical problems, such as the selection of representative molecules from a database.

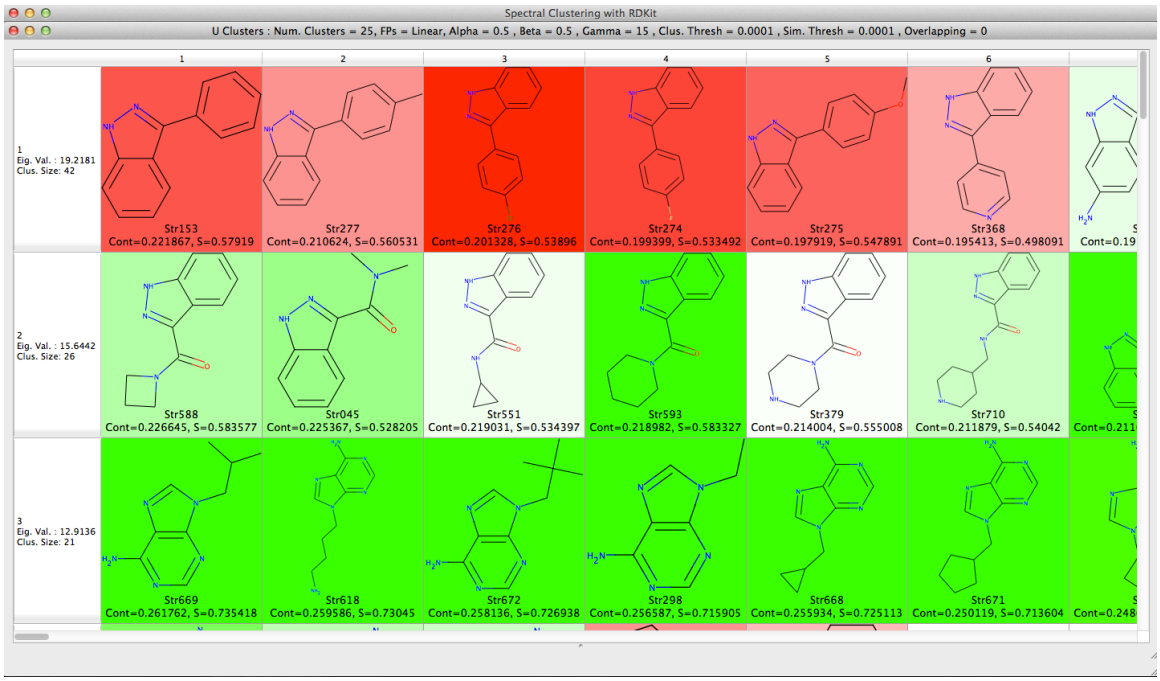


Figure 7-12: *An example of the clusters calculated using the SVD-NOSC algorithm and the Dice coefficient, where γ = 15 and k = 25.*

The comparison of the respective performances of the SVD-NOSC algorithm and the k-means method shows that the k-means method is consistently able to provide greater QCI scores than those produced using spectral clustering. However, examining the clusters produced by the k-means method highlights the issues with considering the QCI scores in isolation. Each set of k-means clusters produced during this investigation contains a single large cluster comprised of molecules based on several different scaffolds, an example is provided in ***Figure 7.13***.

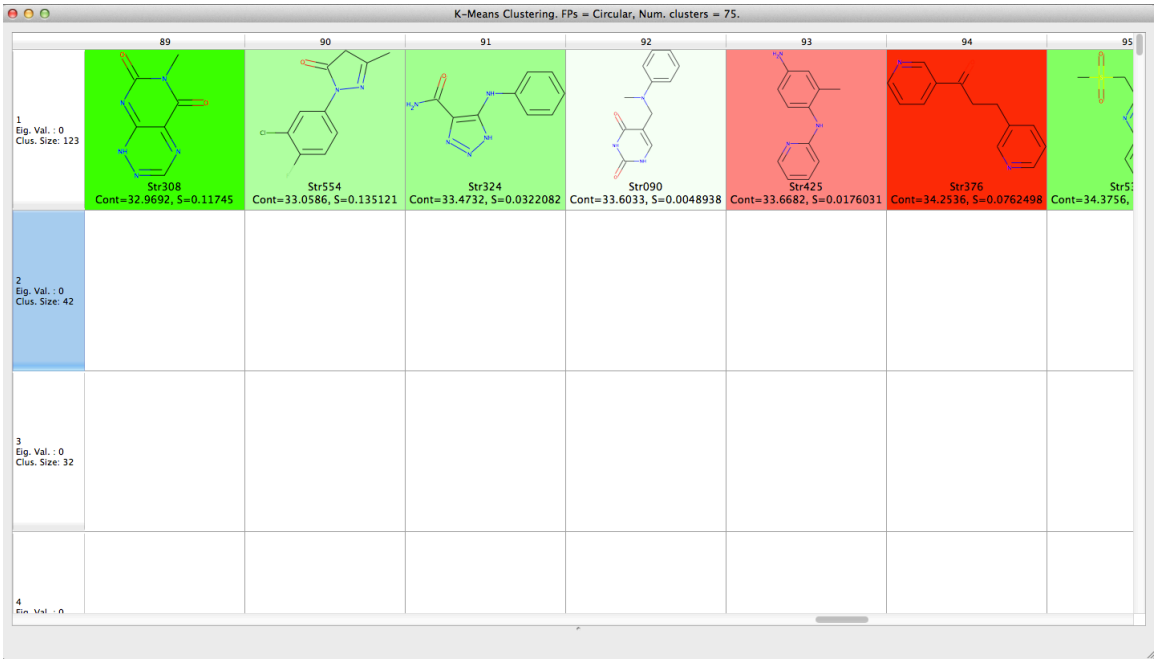


Figure 7-13: *Example of the different scaffolds placed within a single large cluster when using the k-means method, where k = 75.*

The presence of this multi-scaffold cluster leads to the production of higher than expected QCI scores, due to the number of active molecules placed within it making the like for like comparison of the methods using the QCI measure difficult. Determining which method produces the "best" set of clusters from the Target X fragment class is very difficult due to certain features that each algorithm favours, for example, the multi-scaffold cluster produced by k-means. However, the two methods can be discussed in terms of the complementary sets of clusters they produce.

By examining each set of clusters it is apparent that the different methods cluster compounds differently. Within the two sets of clusters there is a level of overlap showing that both methods cluster the most similar molecules identically, with variations in assignments being based around compounds that can be classified into different clusters based on the objectives of the clustering algorithm chosen. For example a 3-benzyl-1H-inadzole-5-sulphonic acid molecule as shown in ***Figure 7.14***, can be clustered with other sulphonic acids, indazoles or indazoles linked with other ring systems. It is in these situations where clustering algorithms can disagree, and produce complementary sets of clusters, as all 3 approaches would have merits and drawbacks. Which clustering approach produces the "best" set of clusters is then judged based upon the goal of the clustering.

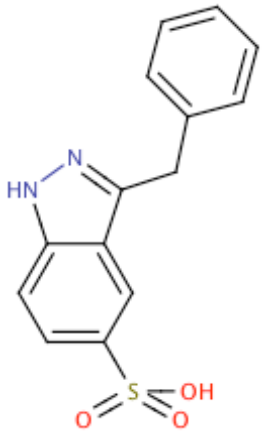


***Figure 7-14**: A 3-benzyl-1H-inadzole-5-sulphonic acid molecule.*

The two different sets of clusters can also be appraised through the use of heat maps to check the general distribution of the clusters and active compounds. ***Figure 7.15*** provides an example of the comparison between the SVD-NOSC and k-means approaches using heat maps.

***Figure 7-15:** Heat map comparison of SVD-NOSC and k-means methods at k = 75.*

The heat maps above show the different distributions of clusters produced by the two methods, with the k-means method having a signifcant number of molecules within a single large cluster, along with several clusters of around 25-30 molecules. On the other hand the SVD-NOSC algorithm places the majority of molecules within one of several clusters of around 35 – 45 molecules in size.

k	α	β	γ	QCI		γ	QCI		γ	QCI	
				U	V		U	V		U	V
25	1	1	5	33.24		10	37.79		15	38.85	
	0.9	0.1		34.26	29.85		32.91	32.80		30.67	30.65
	0.8	0.2		32.71	30.74		32.07	32.80		32.53	32.49
	0.7	0.3		33.84	30.88		31.86	31.27		30.92	31.85
	0.6	0.4		32.74	32.01		31.30	31.47		31.60	32.72
	0.5	0.5		33.06			36.43			35.68	
	k-means			36.72			36.72			36.72	
50	1	1		40.11			39.34			39.79	
	0.9	0.1		36.75	34.99		38.93	36.29		38.54	36.08
	0.8	0.2		35.47	34.89		35.33	38.19		35.95	35.57
	0.7	0.3		36.02	35.62		35.64	35.62		34.72	36.25
	0.6	0.4		35.71	36.25		36.90	36.34		36.36	34.55
	0.5	0.5		39.33			40.21			39.36	
	k-means			39.09			39.09			39.09	
75	1	1		41.09			43.27			41.90	
	0.9	0.1		38.36	39.47		39.55	37.09		39.45	38.67
	0.8	0.2		38.69	36.99		38.02	40.72		36.92	38.19
	0.7	0.3		37.38	37.22		38.16	36.51		37.92	40.27
	0.6	0.4		38.31	36.76		36.21	37.06		37.78	36.79
	0.5	0.5		42.53			44.53			43.17	
	k-means			42.98			42.98			42.98	

Table 7-9: *QCI scores obtained for the clustering of the Target X dataset using RDKit Linear descriptors.*

The results in ***Table 7.9*** provide the respective QCI scores produced by each application of the SVD-NOSC and k-means algorithms to the Target X fragment class using RDKit Linear fingerprints. The results shown in this table mirror those already discussed within this section. In particular, these results reinforce the argument that the use of the Dice coefficient with spectral clustering provides the best possible results when clustering fragments, and merits further research within other chemical problems.

7.4.4 Conclusion

This section has highlighted the potential of the SVD-NOSC procedure for clustering fragments. With the SVD-NOSC algorithm providing comparable and complementary results to the k-means method in both studies.

Section 7.5.1 highlighted the significant overlap in how clustering algorithms assign molecules/fragments to clusters, with comparisons of both SVD-NOSC/k-means to the ideal case yielding Jaccard scores of around 0.65 – 0.70. Although both methods shared a sizeable overlap to the ideal case, the clusters they produced were complementary, also shown by Jaccard scores. This investigation also highlighted how other similarity measures could be used to produce superior results to those generated using the Tanimoto coefficient, when working on small scale fragment-based problems. With both the Dice coefficient and asymmetric Tversky variations producing interesting results.

The second investigation in ***Section 7.5.2*** produced two key findings. Firstly, that the choice of similarity measure, used in conjunction with the SVD-NOSC algorithm, has a sizeable effect on the calculated QCI scores. With the Dice coefficient continually providing good results. In contrast to the previous study the application of spectral clustering coupled with the Tanimoto coefficient, was also shown to produce good results as the use of the Gaussian filtering function has a greater effect on this larger dataset. Secondly, the number of clusters, k, extracted from the fragment class has a significant effect on the magnitude of the QCI scores calculated. When k is low, a number of large clusters are formed, resulting in significant contributions to the p, q and r values in the QCI calculation. As the value of k increases, the larger clusters are split to form several smaller clusters, some of which will be active. The greater number of clusters meeting the criteria to be considered active, increases the contribution made to the p value and decreases both the values of q and r, hence leading to a rise in the overall QCI values.

7.5 Summary

This chapter has introduced an improved method for the spectral clustering of chemical data based upon the use of an SVD-based eigensolver. These SVD-based spectral clustering methodologies have significantly lower associated time costs and a greater scope for implementation, whilst maintaining a high level of accuracy in the eigenpair approximations. The ability to cluster larger datasets along with the improved time costs of the spectral clustering method further increases the feasibility of their use as an alternative to the more traditional clustering algorithms.

The results of the experiments aimed at identifying if SVD-based spectral clustering can be used in FBDD problems showed encouraging results with the SVD-NOSC algorithm providing comparable results to the k-means algorithm, whilst producing a complementary set of clusters. Furthermore, these studies have shown that using other similarity measures in preference to the Tanimoto coefficient provides an interesting avenue for further research, due to the promising results obtained when using both the Tversky Index and the Dice coefficient.

Chapter 8

SVDCLUS

8.1 SVDClus

Working in conjunction with David Cosgrove at AstraZeneca, we have developed an opensource program for clustering chemical data called SVDClus. The SVDClus application was written using C++ and incorporates the Qt (Trolltech, 2008), Boost (Dawes and Abrahams, 2002), RDKit (Landrum, 2006) and SVDLIBC (Rohde, 2009) libraries to allow the user to carry out a number of functions. SVDClus can be run from both the command line or by using the Qt Creator application to launch the custom-built graphical user interface, GUI, which was written with the Qt framework (Trolltech, 2008).

A discussion of the features and capabilities of SVDClus is now provided.

8.2 Input & Molecular Descriptors

Currently, SVDclus is limited to reading in datasets in the form of SMILES files that contain a SMILES string and a unique molecular identifier per line. SVDClus utilises the RDKit molecular drawing libraries to generate a 2D representation of each of the molecules, along with their respective identifier, see ***Figure 8.1***.

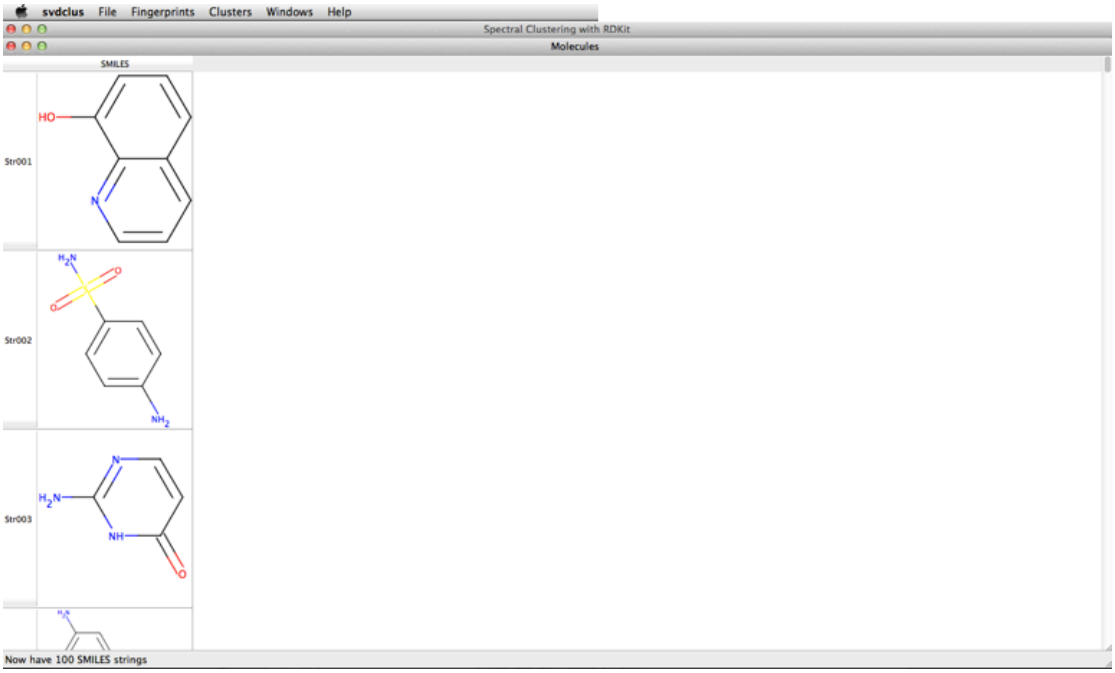


Figure 8-1: *Shows how the SVDClus program displays molecules and their identifiers after reading in a set of SMILES strings.*

Once a dataset has been imported into SVDClus, a molecular descriptor can be selected from the fingerprint tab. A molecular descriptor that is chosen will be used for calculating the similarity scores for each clustering implementation. SVDClus provides the user with three options for selecting a molecular descriptor. Firstly, the user can choose between two native 2D molecular fingerprint types:

- Circular/Morgan RDKit fingerprints, which are generated using the RDKit library and utilise an analogous method to that used in the calculation of ECFP fingerprints. In this approach an algorithm is used to map atoms that fall within a circle that has a radius of 4 bonds from a starter atom. The algorithm calculates indices from the atoms that fall within this circle and calculates a fingerprint from them. Circular RDKit fingerprints share a similar set of characteristics with other circular fingerprints, such as ECFPs.

- Linear Path RDKit fingerprints that are calculated using the RDKit library and produce fingerprints via the hashing of molecular features found within different paths into a fingerprint. These fingerprints share a similar set of features as Daylight fingerprints, which they are closely related too.

The third option that is to read in another set of fingerprints generated outside of the SVDClus tool. These fingerprints must be input within a text file that carries an identifier and fingerprint per line.

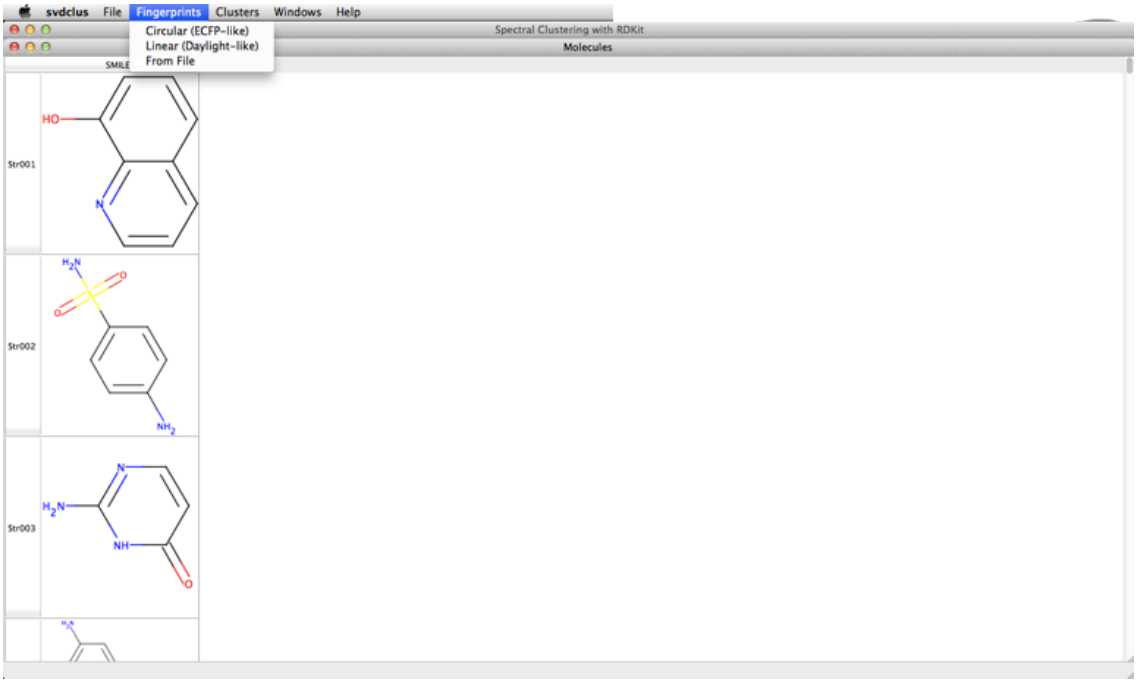


Figure 8-2: *Screenshot showing the different fingerprint options built into SVDClus.*

8.3 Clustering Algorithms

Currently SVDClus contains three different in-built clustering algorithms: SVD-based spectral clustering, fuzzy k-means and k-means. Each clustering algorithm has its own panel that allows the parameters of each clustering method to be easily adjusted. It should be noted that if these parameters are not altered, they are set to default values.

8.3.1 SVD-based Spectral Clustering

The implementation of spectral clustering in the SVDClus toolkit can be broken down into three main steps:

1. The calculation of the similarity matrix.
2. Identifying the eigenpairs.
3. Assigning molecules to the appropriate eigenclusters.

To calculate the similarity matrix, four parameters must be set within the SVD-based spectral clustering control panel. These parameters are the values of α and β in the Tversky index, the value of γ in the Gaussian filtering function and the similarity threshold value that is used to make the input matrix sparse. The values of α and β have two effects on the similarity matrix produced. Firstly, the magnitude of their values determines the similarity measure used in calculations, for example, selecting values of $\alpha = 1$ and $\beta = 1$ means that the Tanimoto coefficient is used for calculating the similarity matrix. Secondly, when using two differing values for α and β, a non-symmetric similarity matrix is produced, and hence two sets of clusters, U and V, are identified during the later steps.

After values for these parameters have been selected, the SVDClus program generates the similarity matrix by calculating each similarity score, applying the Gaussian filtering function directly to this value and comparing it to the similarity threshold value. If the filtered similarity score is greater or equal to the threshold, this value is stored directly in a sparse matrix held in the Harwell-Boeing format. By taking this approach, the computational costs can be minimised, producing both time and storage cost savings. It should also be noted that SVDClus automatically calculates and displays the sparsity of the input matrix.

Next, the las2 algorithm from the SVDLIBC library is applied to the HB format similarity matrix, calculating the k eigenpairs (see ***Chapter 7***). In this method, the k eigenpairs related to largest positive eigenvalues are identified, unlike in the Lanczos approach described in ***Chapter 6***. The magnitude of each eigenvector element was then compared to a user-defined eigenvector threshold to determine if an eigenvector element contributes at a level significant enough to be considered part of the final eigencluster produced from an eigenvector.

Finally, molecules are placed into eigenclusters according to either an overlapping or non-overlapping method, which is set as a parameter in the SVD clustering panel of the GUI or when calling this method from the command line. In the case of overlapping clustering,

molecules are placed into each eigencluster to which they make a contribution larger than the eigenvector threshold. This means that a molecule can belong to several clusters at once, sharing different levels of overlap to structures in each cluster. For example, a hypothetical molecule A, can belong to two clusters based on eigenvector element scores of 0.5 and 0.7 respectively, and we can say that molecule A is more closely related to the molecules in the second cluster. In the non-overlapping approach, or SVD-NOSC, clusters are formed based on the molecules largest contribution to an eigenvector, using the same method outlined in ***Chapters 4 - 7***. The parameter panel for SVD-based SC in the SVDClus GUI is shown in ***Figure 8.3***.

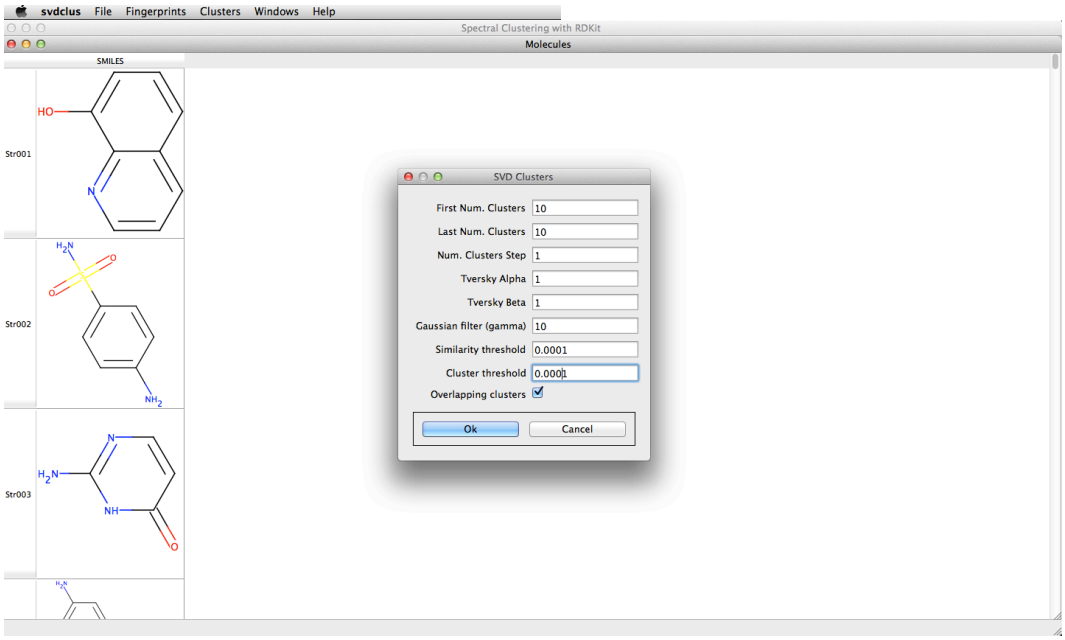


Figure 8-3: *SVD-based clustering control panel.*

After applying any of the clustering methods, a visual output of the clusters is generated. Each row in the output shows the molecules that comprise the cluster, ranked according to their numerical contributions. Molecules that make the largest contributions appear first in this output, with molecules that make smaller contribution being found further down the cluster. SVDClus also provides the user with an output of the number of clusters produced by the clustering, the time taken to cluster the molecules in seconds and either a Silhouette score if the clustering is non-overlapping or a Fuzzy Silhouette score for overlapping clusters (see ***Section 8.4***). An example of this output is shown in ***Figure 8.4***.

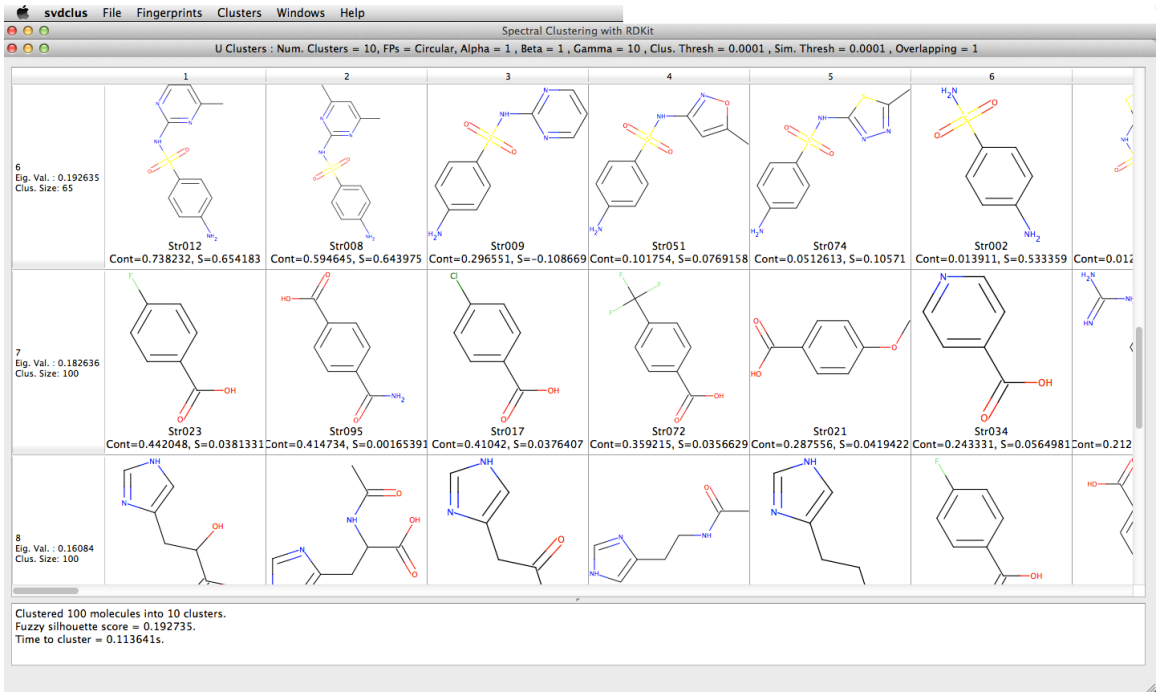


Figure 8-4: *Example clusters generated using the SVD-SC algorithm. The bottom window within the GUI displays the number of clusters created during the implementation, the fuzzy silhouette score for the set of clusters and the time taken to cluster the dataset.*

Along with the measures and the percentage sparsity of the input matrix, SVDClus also calculates and displays the time required to generates the clusters from a dataset.

8.3.2 K-means

A basic version of the k-means method is utilised within the SVDClus application for clustering data. This implementation is set up to run continuous updates of the seeds across a user-defined number of iterations. Although this method produces the most accurate set of clusters possible using this approach, it does lose some of the time advantages conveyed by updating the cluster centroids within a batch, i.e. all centroids are updated at the end of an iteration, as continuously updating the centroids comes at an operational cost. ***Figure 8.5*** shows the parameter panel for k-means clustering.

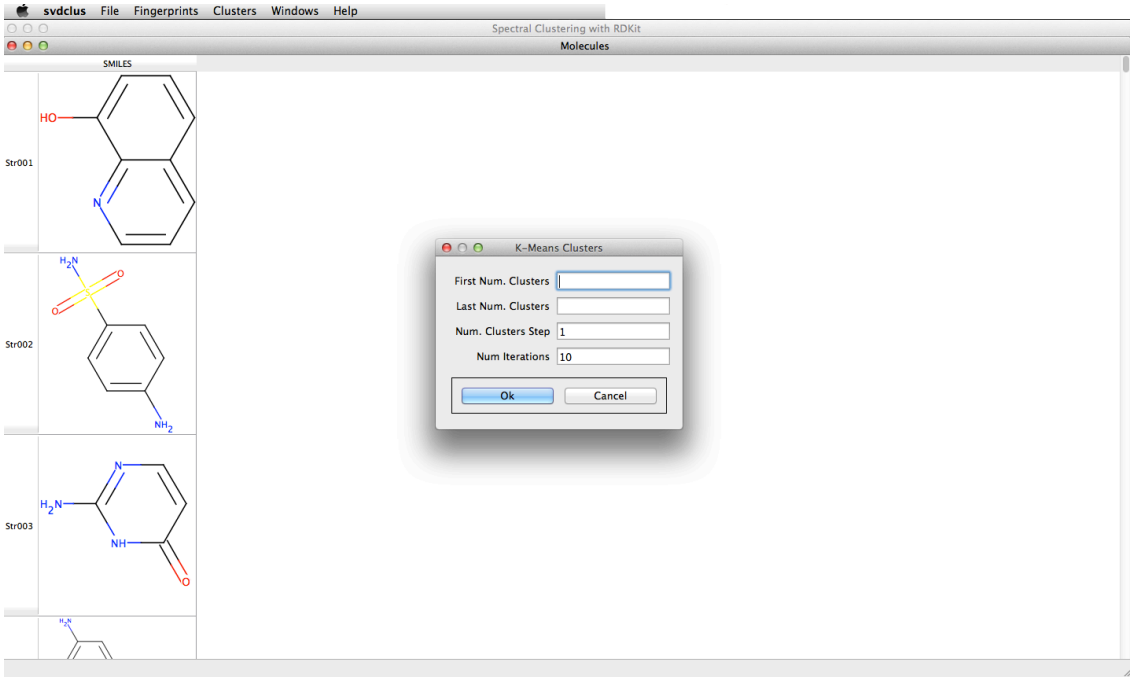


Figure 8-5: *The parameter panel for k-means clustering in SVDClus.*

8.3.3 Fuzzy K-means

Traditional non-overlapping clustering methods are afflicted with several common problems related to the non-overlapping clusters they produce. These issues include:

- The restrictiveness of conventional clusters i.e. that a molecule must be placed in a single cluster.
- The inability of clustering algorithms to effectively cluster "bridge" molecules, which bridge together two clusters.
- The desire to avoid the formation of singletons that can lead to outliers being placed in clusters that they should not be in.
- The occurrence of pairs of molecules placed in different classes that have a greater similarity score than some pairs of molecules that are placed in the same cluster.

Hence, to avoid the issues shared by non-overlapping clustering algorithms, fuzzy logic has been used to extend some of the traditional clustering algorithms. Fuzzy logic is the term used to describe a branch of mathematics that aims to model the ambiguity of real-world problems mathematically (Zadeh, 1965). The rationale behind incorporating fuzzy logic into existing clustering algorithms is that fuzzy logic is better able to describe data that traditionally forms poorly separated clusters when analysed with non-overlapping clustering methodologies (Chi et al., 1996). The use of fuzzy sets in cluster analysis allows objects that naturally share

common characteristics with more than a single set, to be placed into each of the sets simultaneously, more accurately reflecting real world situations.

Each cluster produced by a fuzzy clustering algorithm is a fuzzy subset of the set of objects; in which the membership function of each object denotes the degree to which it belongs to a cluster. Ergo, if a cluster is defined as “a group whose members share common properties” then the membership function of an object indicates the degree to which the object exhibits those properties. The sum of the membership functions for any object is equal to unity, for mathematical traceability. It should also be noted that, in the ideal or extreme case, where each molecule is either a member of a set or not, fuzzy clustering would produce the same clusters as the traditional clustering algorithm that it is an extension of. The use of Fuzzy clustering with chemical data has been presented by Holliday et al. (2004), who applied a fuzzy k-means algorithm to two-dimensional chemical structures using a multistep algorithm. This study showed that the fuzzy k-means method provided complementary results to other clustering approaches.

The fuzzy k-means algorithm has been included as an alternative to overlapping spectral clustering within the SVDClus program for comparative purposes. Like the k-means algorithm built into SVDClus, the fuzzy k-means approach used in this tool also utilises a method that continuously updated the cluster centroids and hence, is slower than the batch form used in some other programs. The parameter panel for this method (see ***Figure 8.6***) allows the user to adjust the fuzziness index, m, along with k and the threshold used to identify which clusters that a molecule contributes too.

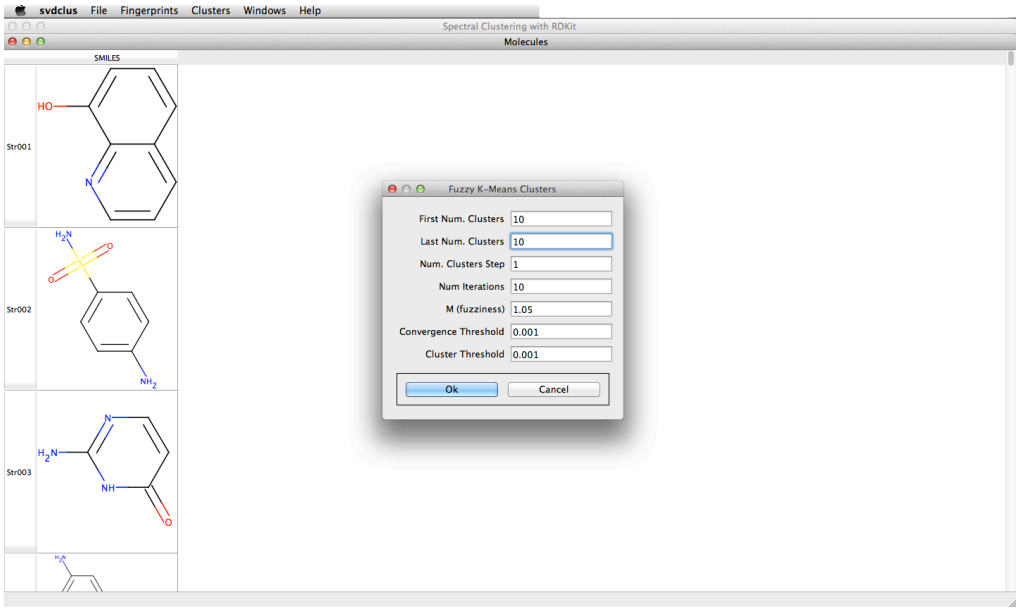


Figure 8-6: *The parameter panel for the fuzzy k-means algorithm in SVDClus.*

8.4 The QCI Measure and using Assay Values to Plot Activity

The activity values for a dataset can be read into the SVDClus toolkit in the form of a separate file, which can contain activity information for several different assays. Using this information the molecules are coloured according to their activity against a certain target, which is determined by a user-specified threshold. Molecules with activity values below this cut off are coloured green to signify their activity, with the most active molecules being the darkest. Alternatively, inactive molecules are coloured red, with the shade of red becoming darker as the molecules become less active. Molecules that fall close to the threshold are left uncoloured, leaving molecules without activity data yellow (see ***Figure 8.7*** for example).

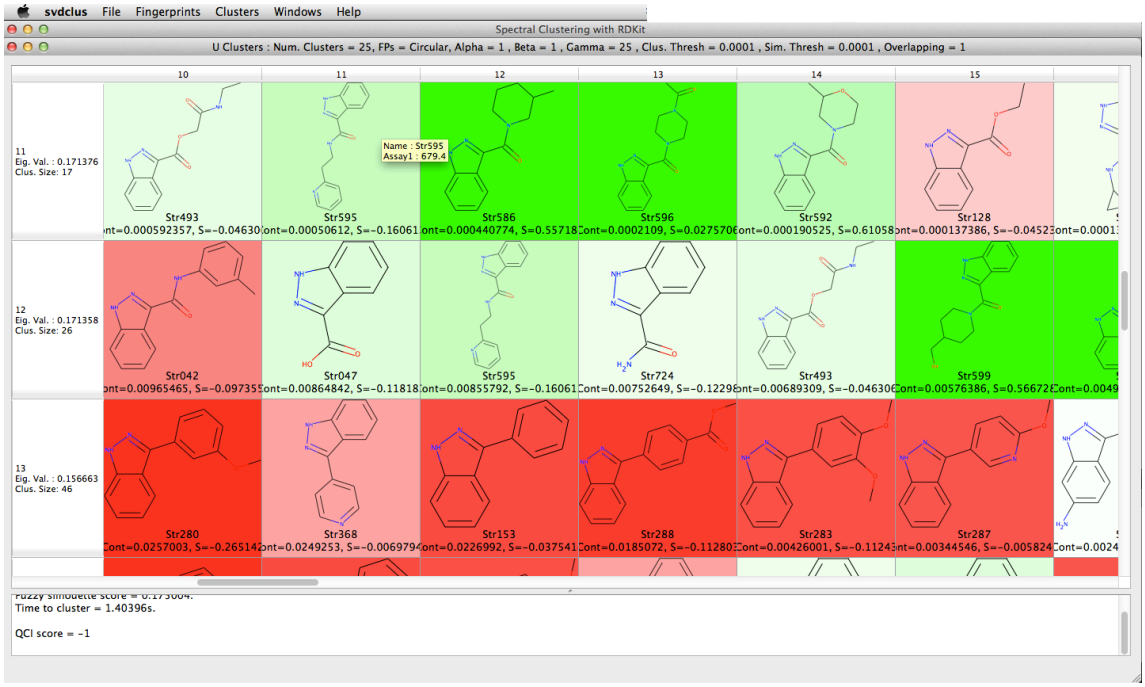


Figure 8-7: *Screenshot showing how molecules are coloured based upon their activity with SVDClus.*

For non-overlapping methods, the QCI score (see ***Section 4.6.1***) for the clustering of the dataset are also automatically calculated.

For the purpose of this measure an active cluster is defined as a cluster containing a greater percentage of active molecules than the dataset as a whole. With the QCI measure giving a measure of the quality of the placement of actives within clusters as a value between 0 and 1, where 1 signifies all actives are placed in active clusters and 0 indicates that all actives are placed into inactive clusters.

By zooming out from the molecules, heat maps are generated from this activity data to show the distribution of both inter and intra-cluster activities, see ***Figure 8.8***. The heat maps

produced from the application of SVD-based spectral clustering can be sorted based upon the magnitude of their eigenvectors or the respective size of each cluster.

Figure 8-8: *Heat map produced from the overlapping clustering of a dataset using SVD-based spectral clustering within the SVDClus tool.*

Two further measures are included in the SVDClus program, which are the Silhouettes (Rousseeuw, 1987) and fuzzy Silhouettes (Campello and Hruschka, 2006) measures and are automatically calculated for the non-overlapping and overlapping clustering algorithms respectively.

8.5 Summary

SVDClus provides an easy-to-use opensource method for the spectral clustering of chemical data. The graphical user interface of this program allows users unfamiliar with the command-line to cluster different datasets using a variety of clustering methods and quickly produce a visual set of clusters that can be easily interpreted and compared to other sets of clusters. The in built measures also provide the user with quantifiable values to compare the performance of different clustering algorithms. The SVDClus software is currently available for use at AstraZeneca R&D facility in England and is used for a number of different chemical problems, producing encouraging results and receiving positive reactions from those who use it.

Chapter 9

Conclusions and Further Work

9.1 Conclusions

This thesis has detailed the development and optimisation of a non-overlapping spectral clustering method for use with chemical problems, most notably grouping actives and inactives within a dataset. The problem of grouping active and inactive molecules was chosen as the primary application for this method as it requires a clustering algorithm to not only identify large scaffold changes, but also small structural changes that are often the difference between a compound being active or inctive. While other clustering algorithms are more established, this method of spectral clustering has consistently been shown to produce comparable or improved results when coupled with highly discriminative molecular fingerprints. Furthermore, comparing the clusters produced by the NOSC, k-means and Ward's algorithms has shown that, although there is a considerable overlap between these methods, each produces a complementary set of clusters. However, spectral clustering algorithms are usually plagued by a single major drawback: the cost of the eigendecomposition step, which in

some cases makes scaling the algorithm for use with larger datasets extremely difficult. As a result, this work focused on how these computational costs can be minimised in order to ensure that spectral clustering can be used as a viable alternative to the more established methods and can be scaled for use with large datasets that contain in excess of 100000 compounds.

Chapter 4 detailed the first experimental work of this study, beginning with the development of a non-overlapping spectral clustering algorithm based on two studies; one by Sarkar and Boyer (1998) and another by Brewer (2007). In the resulting approach a dataset of molecules is clustered by forming a filtered similarity matrix; decomposing the matrix to find the eigenpairs using a full matrix diagonalisation procedure; and finally classifying molecules into clusters based upon their contributions to a subset of eigenvectors that is selected using the 95% positive eigenvalue rule. The clusters produced by the NOSC algorithm can be altered using two parameters: γ and the eigenvector contribution threshold. Experiments aimed at identifying how these parameters effect the clusters produced when using different activity classes and molecular fingerprints, were undertaken in ***Section 4.5***.

These experiments showed the relationship between the value of γ required to produce an effective clustering of a dataset and the type of molecular descriptor employed. For highly specific fingerprint types, such as ECFP_4, a low value of γ is required due to the larger distribution of pairwise similarity scores, which in turn, leads to the calculation of eigenvectors that are dominated by a few large eigenvector contributions reflecting the most closely related molecules. On the other hand, the more generic fingerprint types, such as MDL Public Keys, require the use of a significantly larger γ value to produce a favourable clustering of the dataset. This is due to the fingerprints being unable to discriminate between closely related molecules, leading to the production of eigenvectors without any clearly dominant components and therefore requires the use of a larger γ value to increase their separation. It should be noted that the performance of other fingerprints varies between these two extremes. Furthermore, the homogeneity of the dataset was shown to have a large impact on the value of γ required to segment a dataset, with highly homogeneous data requiring the use of significantly larger γ values to separate clusters that are based on scaffolds that share large structural commonalities yet are distinct. The experiments in ***Section 4.5*** also highlighted the need for the use of an eigenvector threshold to avoid the erroneous classification of molecules. In general, the eigenvector threshold value of 0.0001 was deemed sufficiently low enough to ensure all molecules are placed into the correct cluster or are identified as outliers,

when dealing with heterogeneous data. Other findings of this investigation show that the choice of eigenvector threshold magnitude is not affected by the choice of fingerprint. A general trend was observed that as the homogeneity of a dataset is increased, there is a requirement to decrease the value of the threshold to account for the lower average magnitude of the eigenvector elements that are retrieved.

Section 4.6 contains a QCI comparison between the clusters produced when using the NOSC, Ward's and k-means algorithms. This comparative study showed that for BCI, Unity and Daylight fingerprints the performance of spectral clustering was broadly in-line with the other methods, with each clustering algorithm producing slightly improved results when applied with certain fingerprints. For example, the NOSC algorithm in particular favoured Daylight fingerprints that are more capable of discriminating between molecules and as a result allow spectral clustering to produce a better partitioning of the data and higher QCI scores, whereas the Ward's method produced a slightly better performance than the two other algorithms when applied to Unity fingerprints. Although the application of the NOSC approach produced encouraging results with Daylight, Unity and BCI fingerprints, the algorithm's performance with MDL Public Keys was extremely poor due to the generic nature of these descriptors and the problems this creates when using the NOSC algorithm. On the other hand, ECFPs produced very encouraging results when used in conjunction with the NOSC algorithm. The highly descriptive nature of these fingerprints leads to the production of multiple small clusters containing several highly related structures; resulting in the generation of large QCI scores with NOSC outperforming both the k-means and Ward's methods for all the datasets analysed.

After establishing that the NOSC algorithm could produce comparable - and in some cases improved - results when compared to the two traditional clustering methods, the next step in this study was to evaluate the computational costs of the algorithm, with ***Section 4.7*** providing an investigation into the respective time costs of applying the NOSC method to the four different sized activity classes. The results show that the algorithm scales unfavourably with N in terms of both time and storage costs and as a result is not applicable to datasets greater than around 4000 molecules.

Chapter 5 extended this work by presenting a modified version of the NOSC algorithm that aimed to address two major weaknesses of the original method. Firstly, the m-NOSC method aimed to provide the user with greater controllability by introducing a user-defined parameter k that is used to select the number of eigenvectors to act as clusters. Secondly, the m-NOSC algorithm aimed to reduce the computational cost of spectral clustering by making the input

matrices highly sparse and hence minimising the cost of the diagonalisation procedure using a similarity threshold, below which the similarity was set to zero. Experiments within this chapter were split into two sections: those aimed at identifying an appropriate value for the similarity threshold (***Section 5.3***) and those aimed at testing the effect the m-NOSC algorithm has on the QCI scores produced (***Section 5.4***).

In ***Section 5.3*** several different similarity thresholds were applied to each of the datasets and the percentage of zero elements quantified. These experiments showed that, in general, a similarity threshold value of between 0.001 and 0.0001 was able to consistently produce input matrices that were at least 90% sparse. This finding was fundamental to further investigations throughout this thesis, as the eigensolvers used in later chapters are able to exploit the sparsity of these input matrices to significantly decrease the computational costs of spectral clustering. ***Section 5.4*** provided further encouraging results showing that, in most instances, the use of the m-NOSC algorithm - in preference to the NOSC method - actually leads to an improvement in the QCI scores obtained. This is the result of the similarity threshold removing low similarity scores, decreasing the possible eigenvector contributions from some compounds and hence leading to their exclusion from the final set of clusters. However, there were several noted exceptions to this finding, which related mainly to the use of the 5HT1A activity class or the MDL Public Keys. The decrease in the QCI values for the 5HT1A dataset was caused by the small number of active molecules coupled with high heterogeneity. Therefore, the removal of a single active molecule, even one placed erroneously, led to a significant fall in the QCI score for the dataset. The generally low QCI scores related to the use of MDL Public Keys reflect the similarity threshold excluding a higher proportion of similarity scores from the eventual clustering, including several active molecules.

Despite the improvements in the performance of the method, the computational costs remained both high and restrictive. Thus, the full matrix diagonalisation procedure was replaced with a basic version of the Lanczos algorithm in ***Chapter 6***, with the aim of significantly minimising the operational costs. The basic Lanczos algorithm is able to minimise the costs of the eigendecomposition step by approximating a subset of k eigenpairs from an input matrix. Preliminary investigations into applying this algorithm showed that the Lanczos algorithm was succumbing to its well-documented issues with roundoff error and loss of orthogonality, leading us to introduce a Gram-Schmidt reorthogonalisation step. To ensure that these approximated eigenpairs were accurate both the m-NOSC and L-NOSC methods were applied to the set of 125 COX-2 inhibitors used in Brewer's original study and the clusters

they produced compared. Classification of compounds by the two methods were shown to differ when no eigenvector threshold was applied, as the roundoff error inherent in the approximated eigenvectors was large enough to lead to the reclassification of low contributing molecules (i.e. those molecules that make a contribution to a cluster that is lower than 0.000001) into the cluster with the smallest eigenvalue when using the Lanczos approach. Molecules were placed into the cluster with the smallest eigenvalue, as its vector had undergone the least number of refinements and hence had the largest associated error. By examining the contributions, it was determined that the approximations were accurate to at least six decimal places, and hence could be thresholded to ensure that the low level residual error could not affect the results. This experiment provided further evidence of the necessity of using an eigenvector threshold to ensure misclassification does not take place. Other preliminary studies, showed that the value of k did not retrieve the eigenpairs related to largest eigenvalues, and instead identified the eigenpairs that converged first. This distinction is crucial as in general, only ~ 80% of the eigenpairs calculated were related the largest eigenpairs. To combat this issue, eigenpairs were considered as being part of a p (or positive set) and neg (or negative set), with the neg set not being considered in the final clustering. Thus, the simplest solution to this issue was to set the value of k at a value 25% greater than desired and to remove any excess eigenpairs, although this came at an increased computational cost. For example, if 100 eigenpairs were needed for clustering, k would be set to 125 and only the largest 100 considered.

Comparisons between the m-NOSC, L-NOSC, k-means and Ward's algorithms were consistent with the previous findings. The spectral clustering algorithms continued to perform very well with ECFP and Daylight fingerprints, often outperforming the traditional methods. Comparing the time costs of m-NOSC and L-NOSC showed that there is a point at which the Lanczos algorithm stops providing an advantage in terms of time, making its use in preference to full matrix diagonalisation unnecessary. Unfortunately, the Lanczos algorithm struggles to approximate the eigenpairs from ECFPs, and therefore the point at which the Lanczos algorithm's use becomes redundant – i.e. no longer provides a time advantage – is reached much earlier with these fingerprints. This casts doubt over the merits of continued use of this particular eigensolver, as its use with ECFPs is of primary concern due to their use consistently yielding the best results thus far.

This chapter concludes with a scalability test, where an agglomerated set of the four ChEMBL activity classes were clustered using both the L-NOSC and m-NOSC methods. This study

showed that the effectiveness of the Lanczos algorithm is increased as the number of compounds increases, as the number of eigenpairs that can be found before the point at which the algorithm stops being faster is increased.

Due to the issues with stability and the unfavourable results obtained with ECFPs, the Lanczos algorithm was replaced with a SVD-based eigendecomposition algorithm - which used a number of pre-conditioning techniques, a form of the Lanczos algorithm and efficient storage methods - to produce a mathematically stable method with significantly decreased operational costs based on the SVDLIBC library (Berry et al., 1993; Rohde, 2009). Tests of the accuracy of the eigenvector approximations highlighted that the SVD-based eigendecomposition method was not afflicted by the same issues with roundoff error as the basic Lanczos algorithm, and instead was able to produce accurate approximations of the eigenpairs to a greater degree of accuracy. Importantly the value of k was also shown to be related to the eigenpairs with the largest eigenvalues and as a result did not require the calculation of further unnecessary eigenpairs.

The SVD-NOSC algorithm was used to cluster the four ChEMBL activity classes and the time requirements compared to the other spectral clustering approaches. This study showed that the SVD-NOSC algorithm was able to identify a large number of eigenpairs significantly faster than either of the other methods. Encouragingly, the use of the Harwell Boeing matrix format within the SVD-NOSC method was also shown to exploit matrix sparsity to a greater extent than the previous method, leading to SVD-NOSC's application with ECFPs being highly favourable. The scalability of this method was also significantly increased, with its use with datasets containing over 100000 molecules in size being demonstrated; in fact the SVD-NOSC algorithm was able to identify 1000 eigenpairs from a 100000 compound dataset in less than 12 hours. This step forward in terms of scalability increases the scope of the algorithm, allowing its use on larger scale problems such as separating out multiple activity classes from a database.

A key feature of the SVD algorithm that is yet to be discussed is its use with asymmetric measures, where it forms two distinct sets of eigenpairs linked through a common set of eigenvalues. In ***Section 7.6*** the SVD-NOSC algorithm was used in two FBDD problems, with the results being compared to the k-means algorithm. To avoid the problems associated with using the Tanimoto coefficient with small molecules, the Tversky index was used to allow several different, symmetric and asymmetric measures to be probed. The study in ***Section 7.5.1*** highlighted the significant overlap in how clustering algorithms assign molecules to clusters,

with comparisons of both SVD-NOSC and k-means methods to the ideal case yielding Jaccard scores of around 0.65 – 0.70. Although both methods shared a sizeable overlap to the ideal case, the clusters they produced were complementary with each method assigning some compounds to different clusters based on differing sets of chemical features. This investigation also highlighted how other similarity measures could be used to produce superior results to those generated using the Tanimoto coefficient, at least when working on small scale fragment-based problems; with both the Dice coefficient and asymmetric Tversky variations producing interesting results. In particular the asymmetric Tversky indexes provide an interesting approach to clustering with one set of clusters being based on viewing the reference structure as a superstructure and another where it is treated as a substructure. The second investigation in ***Section 7.5.2*** showed that the choice of similarity measure has a sizeable effect on the clusters produced by the SVD-NOSC algorithm, with the Dice coefficient in particular continually providing good results. The results of this section highlighted the potential of the SVD-NOSC procedure for clustering fragment classes, with spectral clustering providing comparable and complementary results to the k-means method within both studies.

Finally ***Chapter 8*** presents an open source tool for spectral clustering, called SVDClus that was developed in conjunction with David Cosgrove of AstraZeneca. This tool was used to carry out the experiments throughout ***Chapter 7*** and has shown good results. The graphical user interface allows this program to be used by a wider variety of chemists, who are unfamiliar with the command line and programming, allowing them to produce clusters using different methods, metrics and parameters, and assess them both visually and with the in-built measures.

9.2 Future Work

There are several areas where this work can be advanced or supplemented. Firstly, the inclusion of an overlapping version of the SVD-based spectral clustering algorithm in the SVDClus program, lays the foundation for a significant area of further research. Overlapping spectral clustering methods provide a move away from Boolean type classification, and instead allow molecules to be members of multiple different clusters at any one time. This not only provides an interesting new perspective of how compounds should have been grouped in the problems described within this thesis, but also increases the viability of using spectral clustering algorithms within problems that are more suited to the use of overlapping clustering methods such as, looking for possible scaffold hops to circumvent undesirable chemical features or in drug reclassification studies where the aim is to identify other targets that a drug

may be used against, for example, clustering a set of compounds Y, along with several molecules that are known to be active against a target to try to identify if any of the molecules in set Y show potential for use against this target.

Throughout this thesis, and in general within the clustering literature, one of the biggest challenges has been determining how many clusters (eigenpairs in the case of spectral clustering) should be considered when clustering a dataset. To answer this question the 95% positive eigenvalue rule was considered along with a user defined parameter k. Both of these methods provide positives and negatives, however neither provided an empirical answer to question. One method by which an answer to this question may be achieved is through the study of eigengaps, which are also known as spectral gaps within the spectral clustering literature. In the most basic terms, an eigengap is the name given to the difference between two successive eigenvalues (von Luxburg, 2007). In many applications, the size of this gap is monitored to identify which eigenpairs should be considered for further analysis; with large eigengaps indicating that eigenpairs should be considered and low eigengaps indicating the opposite. However, identifying what qualifies as a "large" eigengap is subjective and as a result each method determines this through the use of a different measure or set of bounds (Lin, 1991; Ng et al., 2001; Li and Liu, 2007; Kong et al., 2010). Therefore, a significant area of further study could be aimed at ascertaining how eigengaps can be used to provide a robust method for quantifying how many eigenpairs should form eigenclusters.

Although the use of SVD-based eigendecomposition methods has been shown to increase the scalability of spectral clustering significantly, there are other eigensolvers that also merit further investigation. One example is the Jacobi-Davidson method that has shown promise in finding accurate eigenpairs for large problems (Sleijpen et al., 1996; Sleijpen et al., 1998; Sleijpen and Van der Vorst, 2000; Geus, 2002; van den Eshof, 2002; Hochstenbach and Notay, 2006). For more information on the basics of this method the reader is directed to the publication by Golub and Van Loan (1996).

Other avenues of research that could be used to supplement this work are the investigation of other factors that affect the generation of the input matrix. In our investigations, the choice of molecular descriptor was limited to 2D molecular fingerprints, however spectral clustering approaches can be applied to any sort of data that a matrix can be formed from. Therefore the use of other descriptors such as various 1D counts, topological indices, 3D fingerprints and pharmacophore keys, could all provide interesting results and allow the application of spectral clustering to other chemical problems not mentioned within this research. Further

investigation into the use of similarity measures other than the Tanimoto coefficient, would also be interesting, in particular extending the use of the asymmetric Tversky and Dice coefficients to problems involving larger molecules. Another factor that requires further investigation is the choice of input matrix, which was limited to similarity matrices within our studies, but provides a large scope for further research. In other fields the use of Laplacian (Ng et al., 2002) and affinity (Perona and Freeman, 1998) matrices - where affinity is defined by measures unique to each clustering application - in particular have shown promising use. Their use is likely to have a fairly large impact on the clusters produced due to each matrix type providing a different view of a problem. The use of filtering functions other than those based on Gaussian functions may also be considered, however preliminary investigations and work by Brewer (2007) has failed to identify an improved method for filtering values. The performance of SVD-NOSC with other similarity measures may also provide an interesting area of further research that would supplement the finding of the thesis.

Finally the area that undoubtedly provides the largest scope for further research is the adaptation and testing of other pre-existing spectral clustering methods that have yielded promising results in other fields. In particular the algorithm by Ng, Jordan and Weiss (2002) – which identifies the k largest eigenpairs from a Laplacian matrix, places them into a new matrix Y, renormalises the rows and applies a k-means algorithm to the vectors to identify the clusters (see ***Section 3.4.5***) – has shown successful adaptation for use with proteins (Paccanaro et al., 2006; Nepusz et al., 2010), and therefore would be expected to perform well with chemical data. Also the algorithm by Shi and Malik (2000) – that calculates the eigenvector related to the smallest generalised eigenvalue, clusters datapoints based on the sign of their contributions to the vector and recursively implements these steps until enough clusters are identified (see ***Section 3.4.4***) - has been heavily cited within other works (Paccanaro et al., 2006; Brewer, 2007) and as result should also be considered for use with chemical data. Furthermore, the spectral clustering approach used in this study represents probably the most computationally costly example of spectral clustering, due to the large number of accurate eigenpairs that are used in the clustering step. Therefore, by identifying the SVD-based eigendecomposition method as the best approach to minimising these costs, we can also optimise other spectral clustering algorithms by using this method in their eigendecomposition steps too.

Appendix A

Identifying Eigenpairs using Full Matrix Diagonalisation

Let us again consider the ideal dataset out in ***Figure 3.1,*** which contains six molecules *A* - *F*. A similarity matrix, S, of size N x N, where N is equal to the number of molecules in the dataset can be formed. (***See Table***)

	a	***b***	***c***	***d***	***e***	***f***
a	1	0	0	1	1	0
b	0	1	0	0	0	1
c	0	0	1	0	0	0
d	1	0	0	1	1	0
e	1	0	0	1	1	0
f	0	1	0	0	0	1

$$S = \begin{matrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{matrix}$$

Next an eigendecomposition algorithm is applied to S. The first step in a basic eigendecomposition algorithm is to find the eigenvalues of the matrix using the equation:

$$det\,(S - \lambda I_n) = 0$$

Where;

S = similarity matrix

λ = eigenvalues

I_n = identity matrix.

To solve the eigenvalue equation, the matrix $(S - \lambda I_n)$ is formed:

$$(S - \lambda I_n) = \begin{matrix} 1-\lambda & 0 & 0 & 1 & 1 & 0 \\ 0 & 1-\lambda & 0 & 0 & 0 & 1 \\ 0 & 0 & 1-\lambda & 0 & 0 & 0 \\ 1 & 0 & 0 & 1-\lambda & 1 & 0 \\ 1 & 0 & 0 & 1 & 1-\lambda & 0 \\ 0 & 1 & 0 & 0 & 0 & 1-\lambda \end{matrix}$$

The determinant of any matrix is given by:

$$det = (\sum \text{forward diagonals}) - (\sum \text{backward diagonals})$$

Taking the determinant of $(S - \lambda I_n)$ produces the characteristic polynomial equation of:

$$x^6 - 6x^5 + 11x^4 - 6x^3 = 0$$

By solving the quadratic equation above; the eigenvalues for the similarity matrix, S, can be found. In this instance the eigenvalues of the matrix, S, are 0, 0, 0, 1, 2 and 3.

Next the eigenvectors of the matrix must be elucidated by substituting each eigenvalue into the basic eigenvector equation (***given below***) and solving to find the eigenvector x:

$$Sx = \lambda x$$

The eigenvectors of the matrix S are given below:

For the eigenvalues 0, the corresponding eigenvectors are

$$\begin{pmatrix} 0.4082 \\ 0 \\ 0 \\ -0.8165 \\ 0.4082 \\ 0 \end{pmatrix} \begin{pmatrix} 0.7071 \\ 0 \\ 0 \\ 0 \\ -0.7071 \\ 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0.7071 \\ 0 \\ 0 \\ 0 \\ -0.7071 \end{pmatrix}$$

For the eigenvalue 1, the corresponding eigenvector is

$$\begin{pmatrix} 0 \\ 0 \\ -1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

For the eigenvalue 2, the corresponding eigenvector is

$$\begin{pmatrix} 0 \\ 0.7071 \\ 0 \\ 0 \\ 0 \\ 0.7071 \end{pmatrix}$$

For the eigenvalue 3, the corresponding eigenvector is

$$\begin{pmatrix} 0.5774 \\ 0 \\ 0 \\ 0.5774 \\ 0.5774 \\ 0 \end{pmatrix}$$

A way of checking if the correct eigenvectors of matrix S have been found is carried out using the matrix diagonalisation equation:

$$S = C^T . \lambda . C$$

Where;

S = similarity matrix

λ = diagonal matrix of eigenvalues

C = matrix of eigenvectors

C^T = transposed matrix of eigenvectors.

The matrix of eigenvectors, C, is obtained by "stacking" the eigenvectors from the previous step into a square matrix; with the eigenvectors being stacked from left to right in ascending order, according to their associated eigenvalues. Also it should be noted that either the transpose or the inverse of the matrix could be used in the equation above, as they are analogous due to the matrix being *Hermitian*.

$$C = \begin{matrix} 0.4082 & 0.7071 & 0 & 0 & 0 & 0.5774 \\ 0 & 0 & 0.7071 & 0 & 0.7071 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 \\ -0.8165 & 0 & 0 & 0 & 0 & 0.5774 \\ 0.4082 & -0.7071 & 0 & 0 & 0 & 0.5774 \\ 0 & 0 & -0.7071 & 0 & 0.7071 & 0 \end{matrix}$$

In order to check that the correct eigenvalues and eigenvectors have been found the matrix diagonalisation is solved, to reform the similarity matrix. This can be done by multiplying the matrix of eigenvectors by the transpose matrix, C^T, and the diagonal matrix of eigenvalues, both of which are given below.

$$C^T = \begin{matrix} 0.4082 & 0 & 0 & -0.8165 & 0.4082 & 0 \\ 0.7071 & 0 & 0 & 0 & -0.7071 & 0 \\ 0 & 0.7071 & 0 & 0 & 0 & -0.7071 \\ 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0.7071 & 0 & 0 & 0 & 0.7071 \\ 0.5774 & 0 & 0 & 0.5774 & 0.5774 & 0 \end{matrix}$$

$$\lambda = \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 \end{matrix}$$

If the eigenvalues and eigenvectors that have been found are correct $C^T \lambda \ C = S$

$$C^T \lambda \ C = \begin{matrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \end{matrix} = S$$

The presence of three non-zero eigenvalues indicates the dataset may contain up to three eigenclusters, where the molecules which populate each cluster are given by the eigenvector components i.e.

for the eigenvalue 2, the associated eigenvector is

$$\begin{pmatrix} 0 \\ 0.7071 \\ 0 \\ 0 \\ 0 \\ 0.7071 \end{pmatrix}$$

where the non-zero eigenvector components show that molecules *b* and *f*, are members of the cluster with an eigenvalue of 2.

Appendix B

Chapter 4 Complete Results Tables

This appendix contains the full results tables generated by the application of the non-overlapping spectral cluster to the ChEMBL activity classes listed below:

- 5HT1A antagonists.
- Matrix Metalloprotease inhibitors.
- Renin inhibitors.
- Substance P antagonists.

Each of the datasets were described by five different types of 2D molecular fingerprint (BCI, Daylight, ECFP_4, MDL Public Keys and Unity) and were clustered using the parameters set out below:

- At sixteen different values of γ in the Gaussian filtering function, with values varying between 25 and 400 at intervals of 25.
- Using six different magnitudes for the eigenvector threshold value, ranging between 0.1 and 1 x 10^{-6}, with the size of the threshold decreasing by an order of magnitude in each subsequent application.

The results for each dataset and fingerprint combination are presented within two tables, in the first table the results for γ values between 25 and 200 are presented with the second table presenting the results obtained using γ values between 225 and 400.

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	144	5	1261	31.719	125	475	127	526	37.852
0.01		147	2	105	22.459		483	119	36	32.734
0.001		147	2	0	22.028		484	118	4	32.576
1.00E-04		147	2	0	22.028		485	117	0	32.416
1.00E-05		147	2	0	22.028		485	117	0	32.416
1.00E-06		147	2	0	22.028		485	117	0	32.416
0.1	50	302	44	714	34.416	150	507	135	566	39.706
0.01		308	38	38	32.710		517	125	66	34.316
0.001		308	38	1	32.629		520	122	6	33.898
1.00E-04		308	38	0	32.586		520	122	0	33.763
1.00E-05		308	38	0	32.586		520	122	0	33.763
1.00E-06		308	38	0	32.586		520	122	0	33.763
0.1	75	379	79	614	34.566	175	553	139	454	39.823
0.01		391	67	61	34.820		560	132	51	36.873
0.001		391	67	3	34.069		561	131	3	36.842
1.00E-04		391	67	0	34.022		561	131	0	36.788
1.00E-05		391	67	0	34.022		561	131	0	36.788
1.00E-06		391	67	0	34.022		561	131	0	36.788
0.1	100	439	114	539	35.140	200	580	145	412	39.716
0.01		448	105	54	31.425		585	140	48	35.994
0.001		450	103	5	30.615		586	139	2	36.134
1.00E-04		450	103	1	30.542		586	139	0	36.084
1.00E-05		450	103	0	30.542		586	139	0	36.084
1.00E-06		450	103	0	30.542		586	139	0	36.084

Table B-1: *The results of the application of the NOSC algorithm to the 5HT1A activity class described by BCI fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	595	162	401	40.545	325	641	179	413	39.818
0.01		602	155	42	36.415		650	170	126	38.675
0.001		602	155	6	35.501		653	167	45	36.479
1.00E-04		602	155	0	35.501		654	166	42	36.517
1.00E-05		602	155	0	35.501		654	166	39	36.364
1.00E-06		602	155	0	35.501		655	165	38	36.364
0.1	250	616	166	397	40.367	350	649	184	385	40.598
0.01		624	158	40	36.161		657	176	97	36.681
0.001		625	157	9	37.611		662	171	15	34.053
1.00E-04		625	157	3	35.792		663	170	3	33.877
1.00E-05		626	156	1	35.792		663	170	2	33.835
1.00E-06		626	156	0	35.792		663	170	2	33.835
0.1	275	621	172	430	40.402	375	659	183	388	42.096
0.01		627	166	84	39.596		666	176	62	38.973
0.001		630	163	38	38.554		669	173	15	36.835
1.00E-04		630	163	33	38.589		672	170	8	39.237
1.00E-05		630	163	33	38.589		672	170	6	39.237
1.00E-06		631	162	28	38.565		672	170	5	39.237
0.1	300	634	176	411	40.323	400	677	187	385	43.633
0.01		644	166	117	38.288		689	175	31	37.677
0.001		647	163	41	36.783		690	174	5	36.461
1.00E-04		647	163	38	36.783		691	173	3	36.412
1.00E-05		648	162	35	36.732		691	173	2	36.412
1.00E-06		649	161	34	36.732		691	173	2	36.412

Table B-2: *The results of the application of the NOSC algorithm to the 5HT1A activity class described by BCI fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	200	6	1048	32.225	125	524	121	415	38.488
0.01		203	3	96	25.162		534	111	107	35.369
0.001		203	3	1	24.004		538	107	14	34.812
1.00E-04		203	3	0	23.964		538	107	2	34.765
1.00E-05		203	3	0	23.964		539	106	0	34.765
1.00E-06		203	3	0	23.964		539	106	0	34.765
0.1	50	375	57	545	34.888	150	542	141	433	38.053
0.01		388	44	36	33.148		553	130	92	35.043
0.001		390	42	1	31.812		554	129	23	33.818
1.00E-04		390	42	0	31.812		556	127	12	33.291
1.00E-05		390	42	0	31.812		556	127	1	33.291
1.00E-06		390	42	0	31.812		556	127	0	33.291
0.1	75	453	93	469	36.869	175	559	153	429	38.645
0.01		459	87	36	35.209		567	145	120	35.080
0.001		460	86	0	35.244		569	143	23	32.946
1.00E-04		460	86	0	35.244		570	142	21	32.903
1.00E-05		460	86	0	35.244		570	142	20	32.861
1.00E-06		460	86	0	35.244		570	142	19	32.947
0.1	100	490	111	402	36.421	200	570	157	452	39.122
0.01		500	101	30	33.333		584	143	49	33.921
0.001		501	100	2	33.510		584	143	6	32.090
1.00E-04		501	100	0	33.510		584	143	3	32.218
1.00E-05		501	100	0	33.510		584	143	2	32.178
1.00E-06		501	100	0	33.510		584	143	1	32.262

Table B-3: *The results of the application of the NOSC algorithm to the 5HT1A activity class described by Daylight fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	591	160	439	39.522	325	625	183	475	41.667
0.01		600	151	71	33.464		632	176	111	37.246
0.001		600	151	15	32.342		633	175	45	34.734
1.00E-04		600	151	6	32.147		634	174	41	34.734
1.00E-05		600	151	3	32.152		634	174	38	34.734
1.00E-06		600	151	3	32.152		634	174	38	34.734
0.1	250	602	165	440	38.368	350	638	180	463	42.276
0.01		610	157	29	35.198		647	171	88	37.143
0.001		610	157	1	34.574		647	171	46	35.920
1.00E-04		610	157	0	34.574		647	171	44	35.920
1.00E-05		610	157	0	34.574		647	171	39	35.920
1.00E-06		610	157	0	34.574		647	171	38	35.920
0.1	275	609	171	473	40.909	375	632	185	480	43.451
0.01		615	165	135	38.278		639	178	134	38.312
0.001		616	164	19	37.633		640	177	53	36.794
1.00E-04		616	164	8	37.425		640	177	49	36.794
1.00E-05		616	164	6	37.313		641	176	45	36.794
1.00E-06		616	164	6	37.313		641	176	41	36.794
0.1	300	619	179	463	42.178	400	639	181	462	42.400
0.01		626	172	76	38.669		649	171	129	37.636
0.001		626	172	9	36.246		649	171	57	36.189
1.00E-04		626	172	6	36.143		649	171	52	37.422
1.00E-05		626	172	5	36.143		649	171	49	37.422
1.00E-06		626	172	5	36.143		650	170	44	37.364

Table B-4: *The results of the application of the NOSC algorithm to the 5HT1A activity class described by Daylight fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	343	30	892	68.350	125	831	147	339	68.767
0.01		353	20	65	52.731		843	135	56	59.603
0.001		353	20	3	51.648		844	134	11	57.023
1.00E-04		353	20	0	51.648		844	134	7	56.904
1.00E-05		353	20	0	51.648		844	134	6	56.994
1.00E-06		353	20	0	51.648		844	134	6	56.994
0.1	50	592	94	533	61.290	150	857	157	326	69.553
0.01		602	84	88	58.613		869	145	54	57.263
0.001		603	83	11	57.755		872	142	13	56.289
1.00E-04		604	82	1	57.404		873	141	5	56.379
1.00E-05		604	82	0	57.404		873	141	5	56.379
1.00E-06		604	82	0	57.404		873	141	5	56.379
0.1	75	724	120	356	64.286	175	868	160	349	69.653
0.01		731	113	22	56.855		880	148	98	54.303
0.001		731	113	3	54.064		881	147	56	56.317
1.00E-04		731	113	2	53.962		882	146	48	51.128
1.00E-05		731	113	1	53.861		882	146	45	51.128
1.00E-06		731	113	1	53.861		882	146	41	50.841
0.1	100	788	143	367	67.908	200	861	161	398	70.091
0.01		798	133	68	57.438		875	147	114	56.114
0.001		801	130	17	55.098		875	147	64	53.550
1.00E-04		802	129	9	54.127		876	146	58	53.629
1.00E-05		803	128	8	54.111		876	146	53	53.414
1.00E-06		803	128	8	54.111		876	146	49	53.307

Table B-5: *The results of the application of the NOSC algorithm to the 5HT1A activity class described by ECFP_4 fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	843	155	467	69.659	325	787	130	663	72.281
0.01		856	142	204	50.996		799	118	258	43.599
0.001		856	142	85	48.794		800	117	25	34.152
1.00E-04		856	142	80	48.799		800	117	4	34.108
1.00E-05		856	142	76	45.546		800	117	1	33.983
1.00E-06		858	140	69	45.470		800	117	0	34.063
0.1	250	834	146	507	68.807	350	758	114	747	71.528
0.01		845	135	220	52.321		764	108	413	49.480
0.001		845	135	35	46.642		765	107	240	42.500
1.00E-04		845	135	17	46.570		765	107	222	42.244
1.00E-05		845	135	16	46.570		767	105	212	42.036
1.00E-06		845	135	15	46.570		767	105	204	41.694
0.1	275	822	136	547	70.701	375	720	112	817	70.968
0.01		830	128	262	48.092		729	103	388	43.165
0.001		831	127	142	42.188		729	103	241	40.033
1.00E-04		831	127	136	42.122		729	103	233	40.000
1.00E-05		831	127	133	42.056		730	102	228	38.629
1.00E-06		832	126	125	42.105		731	101	223	38.760
0.1	300	805	131	612	71.622	400	698	107	878	68.953
0.01		817	119	287	44.308		708	97	506	45.580
0.001		820	116	181	48.708		709	96	291	40.917
1.00E-04		820	116	172	44.696		709	96	286	40.917
1.00E-05		821	115	163	44.905		710	95	279	40.878
1.00E-06		821	115	158	44.905		710	95	274	40.809

Table B-6: *The results of the application of the NOSC algorithm to the 5HT1A activity class described by ECFP_4 fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	4	0	2755	1.004	125	244	27	1131	57.927
0.01		5	0	1050	5.063		249	22	113	40.498
0.001		5	0	41	11.734		249	22	3	37.971
1.00E-04		5	0	0	11.667		249	22	0	37.971
1.00E-05		5	0	0	11.667		249	22	0	37.971
1.00E-06		5	0	0	11.667		249	22	0	37.971
0.1	50	43	3	2323	56.349	150	299	65	912	55.191
0.01		45	1	851	24.294		314	50	49	42.357
0.001		45	1	4	16.842		314	50	2	42.271
1.00E-04		45	1	0	16.807		314	50	0	42.271
1.00E-05		45	1	0	16.807		314	50	0	42.271
1.00E-06		45	1	0	16.807		314	50	0	42.271
0.1	75	93	9	1961	49.761	175	344	57	856	58.197
0.01		96	6	496	21.976		355	46	72	43.519
0.001		97	5	8	21.180		356	45	1	42.348
1.00E-04		97	5	0	21.193		357	44	0	42.348
1.00E-05		97	5	0	21.193		357	44	0	42.348
1.00E-06		97	5	0	21.193		357	44	0	42.348
0.1	100	180	16	1387	54.135	200	384	77	730	57.821
0.01		181	15	122	31.986		399	62	40	46.970
0.001		181	15	1	31.046		400	61	3	46.230
1.00E-04		181	15	0	31.046		400	61	1	46.230
1.00E-05		181	15	0	31.046		400	61	0	46.230
1.00E-06		181	15	0	31.046		400	61	0	46.230

***Table B-7**: The results of the application of the NOSC algorithm to the 5HT1A activity class described by MDL public keys, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	431	89	590	57.853	325	516	110	412	56.744
0.01		446	74	24	50.178		526	100	15	52.481
0.001		446	74	0	50.177		526	100	0	52.174
1.00E-04		446	74	0	50.177		526	100	0	52.174
1.00E-05		446	74	0	50.177		526	100	0	52.174
1.00E-06		446	74	0	50.177		526	100	0	52.174
0.1	250	451	102	536	56.032	350	534	116	382	60.750
0.01		470	83	24	50.823		545	105	19	54.314
0.001		470	83	1	50.179		546	104	0	54.086
1.00E-04		470	83	0	50.267		546	104	0	54.086
1.00E-05		470	83	0	50.267		546	104	0	54.086
1.00E-06		470	83	0	50.267		546	104	0	54.086
0.1	275	479	100	499	54.684	375	540	119	408	61.039
0.01		489	90	24	48.427		550	109	21	52.751
0.001		489	90	0	48.547		553	106	1	52.642
1.00E-04		489	90	0	48.547		553	106	0	52.642
1.00E-05		489	90	0	48.547		553	106	0	52.642
1.00E-06		489	90	0	48.547		553	106	0	52.642
0.1	300	495	110	469	57.801	400	551	127	401	58.680
0.01		507	98	27	51.321		561	117	26	53.801
0.001		508	97	2	49.554		562	116	1	53.786
1.00E-04		508	97	0	49.644		562	116	0	53.682
1.00E-05		508	97	0	49.644		562	116	0	53.682
1.00E-06		508	97	0	49.644		562	116	0	53.682

Table B-8: *The results of the application of the NOSC algorithm to the 5HT1A activity class described by MDL public keys, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	107	4	1628	28.198	125	492	127	475	36.264
0.01		111	0	215	20.020		506	113	25	35.021
0.001		111	0	3	20.693		507	112	4	34.724
1.00E-04		111	0	0	20.630		508	111	0	34.724
1.00E-05		111	0	0	20.630		508	111	0	34.724
1.00E-06		111	0	0	20.630		508	111	0	34.724
0.1	50	307	24	879	34.595	150	533	135	436	36.901
0.01		311	20	57	27.815		549	119	29	34.228
0.001		311	20	1	27.177		550	118	1	34.610
1.00E-04		311	20	0	27.177		550	118	0	34.610
1.00E-05		311	20	0	27.177		550	118	0	34.610
1.00E-06		311	20	0	27.177		550	118	0	34.610
0.1	75	387	75	605	34.410	175	552	136	449	38.758
0.01		399	63	46	32.986		561	127	70	35.897
0.001		401	61	4	32.294		564	124	16	35.157
1.00E-04		401	61	0	31.879		564	124	8	34.966
1.00E-05		401	61	0	31.879		564	124	7	34.918
1.00E-06		401	61	0	31.879		564	124	7	34.918
0.1	100	447	107	527	35.323	200	569	137	491	38.636
0.01		454	100	28	33.029		579	127	60	36.903
0.001		454	100	0	32.737		583	123	14	36.173
1.00E-04		454	100	0	32.737		583	123	12	36.072
1.00E-05		454	100	0	32.737		583	123	11	36.022
1.00E-06		454	100	0	32.737		583	123	9	36.022

Table B-9: *The results of the application of the NOSC algorithm to the 5HT1A activity class described by Unity fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	580	155	468	39.415	325	635	167	430	42.664
0.01		591	144	54	38.272		643	159	75	36.139
0.001		595	140	21	37.994		644	158	61	36.182
1.00E-04		597	138	15	36.111		644	158	52	36.182
1.00E-05		598	137	12	36.100		644	158	32	36.131
1.00E-06		598	137	10	36.050		644	158	22	36.222
0.1	250	596	149	498	39.469	350	648	178	431	42.524
0.01		605	140	117	38.272		658	168	127	38.042
0.001		608	137	29	36.888		660	166	75	35.139
1.00E-04		609	136	22	36.910		660	166	60	34.986
1.00E-05		609	136	20	36.805		660	166	30	34.986
1.00E-06		609	136	18	36.700		660	166	30	34.986
0.1	275	611	153	488	40.824	375	658	177	418	42.264
0.01		620	144	80	38.534		667	168	117	35.960
0.001		621	143	28	36.903		668	167	38	34.747
1.00E-04		621	143	24	36.744		668	167	34	34.747
1.00E-05		622	142	19	36.872		668	167	32	34.747
1.00E-06		622	142	18	36.819		668	167	31	34.699
0.1	300	629	156	450	43.141	400	661	177	436	43.056
0.01		634	151	86	36.471		674	164	71	35.596
0.001		636	149	33	35.624		674	164	39	35.254
1.00E-04		636	149	27	35.425		674	164	34	35.246
1.00E-05		637	148	22	33.292		674	164	31	35.334
1.00E-06		637	148	21	33.250		674	164	29	36.652

Table B-10: *The results of the application of the NOSC algorithm to the 5HT1A activity class described by Unity fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	91	7	2228	74.488	125	425	102	828	72.822
0.01		93	6	318	59.524		436	91	68	72.336
0.001		93	6	7	59.914		436	91	12	72.059
1.00E-04		93	6	0	59.804		437	90	2	72.035
1.00E-05		93	6	0	59.804		437	90	0	72.049
1.00E-06		93	6	0	59.804		437	90	0	72.049
0.1	50	204	33	1518	76.417	150	482	110	782	74.735
0.01		209	28	174	67.830		493	99	51	73.307
0.001		210	27	3	66.302		496	96	5	73.248
1.00E-04		210	27	0	66.208		496	96	1	73.141
1.00E-05		210	27	0	66.208		496	96	1	73.141
1.00E-06		210	27	0	66.208		496	96	0	73.141
0.1	75	295	64	1148	74.634	175	525	129	754	75.033
0.01		306	53	109	72.257		542	112	67	73.989
0.001		308	51	7	71.079		542	112	2	74.061
1.00E-04		308	51	0	70.977		542	112	0	73.987
1.00E-05		308	51	0	70.977		542	112	0	73.987
1.00E-06		308	51	0	70.977		542	112	0	73.987
0.1	100	375	84	968	76.180	200	565	145	672	73.811
0.01		384	75	59	71.919		580	130	53	72.380
0.001		387	72	7	72.125		581	129	2	72.464
1.00E-04		387	72	0	72.104		581	129	0	72.478
1.00E-05		387	72	0	72.104		581	129	0	72.478
1.00E-06		387	72	0	72.104		581	129	0	72.478

Table B-11: *The results of the application of the NOSC algorithm to the MMP1 activity class described by BCI fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	587	163	660	74.046	325	669	207	603	74.249
0.01		603	147	81	73.279		693	183	111	72.713
0.001		604	146	2	73.119		696	180	8	71.925
1.00E-04		604	146	0	73.119		697	179	3	71.917
1.00E-05		604	146	0	73.119		697	179	1	72.076
1.00E-06		604	146	0	73.119		698	178	0	72.076
0.1	250	626	164	616	74.404	350	707	205	554	74.166
0.01		640	150	79	73.469		728	184	108	73.439
0.001		642	148	3	72.968		730	182	10	72.151
1.00E-04		642	148	0	72.968		731	181	3	71.959
1.00E-05		642	148	0	72.968		731	181	1	72.117
1.00E-06		642	148	0	72.968		731	181	1	72.117
0.1	275	652	178	601	74.296	375	714	216	570	73.823
0.01		663	167	49	73.190		734	196	87	72.551
0.001		665	165	0	72.952		736	194	13	72.140
1.00E-04		665	165	0	72.952		737	193	3	72.000
1.00E-05		665	165	0	72.952		737	193	3	72.000
1.00E-06		665	165	0	72.952		737	193	1	72.000
0.1	300	668	198	582	74.364	400	725	221	551	74.513
0.01		685	181	77	72.646		744	202	103	72.399
0.001		687	179	1	72.070		747	199	12	71.791
1.00E-04		688	178	0	72.070		748	198	3	71.864
1.00E-05		688	178	0	72.070		748	198	3	71.864
1.00E-06		688	178	0	72.070		748	198	3	71.864

Table B-12: *The results of the application of the NOSC algorithm to the MMP1 activity class described by BCI fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	139	7	1871	74.070	125	600	156	609	73.188
0.01		142	4	317	65.846		621	135	54	73.242
0.001		142	4	5	64.200		624	132	2	73.410
1.00E-04		142	4	0	63.995		624	132	0	73.373
1.00E-05		142	4	0	63.995		624	132	0	73.373
1.00E-06		142	4	0	63.995		624	132	0	73.373
0.1	50	357	61	1034	74.381	150	659	166	534	73.153
0.01		369	49	101	70.657		674	151	56	72.915
0.001		369	49	2	70.784		676	149	3	73.108
1.00E-04		369	49	0	70.813		676	149	1	73.085
1.00E-05		369	49	0	70.813		676	149	0	73.085
1.00E-06		369	49	0	70.813		676	149	0	73.085
0.1	75	474	114	667	74.159	175	701	188	480	72.957
0.01		489	99	39	73.014		714	175	110	73.337
0.001		490	98	1	72.887		716	173	7	72.494
1.00E-04		490	98	0	72.901		716	173	2	72.522
1.00E-05		490	98	0	72.901		716	173	2	72.522
1.00E-06		490	98	0	72.901		716	173	2	72.522
0.1	100	548	142	612	73.252	200	732	206	489	72.606
0.01		563	127	47	72.834		752	186	61	72.211
0.001		564	126	2	73.025		754	184	5	72.139
1.00E-04		564	126	0	72.951		754	184	2	72.182
1.00E-05		564	126	0	72.951		754	184	1	72.145
1.00E-06		564	126	0	72.951		754	184	1	72.145

Table B-13: *The results of the application of the NOSC algorithm to the MMP1 activity class described by Daylight fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	754	218	480	73.305	325	810	229	495	72.800
0.01		771	201	52	71.895		826	213	63	70.509
0.001		773	199	6	71.881		828	211	0	69.989
1.00E-04		773	199	2	71.821		828	211	0	69.989
1.00E-05		773	199	2	71.821		828	211	0	69.989
1.00E-06		773	199	2	71.821		828	211	0	69.989
0.1	250	769	217	484	73.687	350	818	232	478	72.823
0.01		790	196	58	72.484		832	218	26	70.918
0.001		794	192	8	72.417		832	218	3	70.585
1.00E-04		794	192	3	72.314		832	218	1	70.585
1.00E-05		794	192	2	72.328		832	218	0	70.585
1.00E-06		794	192	2	72.328		832	218	0	70.585
0.1	275	783	229	485	73.272	375	829	238	491	73.573
0.01		808	204	78	71.683		856	211	102	71.421
0.001		809	203	8	71.622		858	209	13	70.450
1.00E-04		810	202	3	71.652		859	208	11	70.450
1.00E-05		810	202	3	71.652		861	206	6	70.518
1.00E-06		810	202	3	71.652		861	206	5	70.548
0.1	300	799	230	475	73.333	400	836	241	517	73.379
0.01		817	212	42	71.279		851	226	94	71.125
0.001		820	209	8	71.295		853	224	15	70.485
1.00E-04		821	208	5	71.362		855	222	12	70.485
1.00E-05		821	208	5	71.362		856	221	9	70.485
1.00E-06		821	208	5	71.362		856	221	7	70.500

Table B-14: *The results of the application of the NOSC algorithm to the MMP1 activity class described by Daylight fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	273	18	1365	80.367	125	995	241	388	79.988
0.01		279	12	157	78.295		1013	223	39	79.433
0.001		279	12	2	77.266		1019	217	6	79.623
1.00E-04		279	12	0	77.266		1020	216	0	79.698
1.00E-05		279	12	0	77.266		1020	216	0	79.698
1.00E-06		279	12	0	77.266		1020	216	0	79.698
0.1	50	615	92	734	82.865	150	1045	268	392	79.352
0.01		631	76	40	81.880		1069	244	39	77.961
0.001		631	76	2	81.701		1070	243	10	77.869
1.00E-04		631	76	0	81.710		1070	243	3	77.893
1.00E-05		631	76	0	81.710		1070	243	0	77.893
1.00E-06		631	76	0	81.710		1070	243	0	77.893
0.1	75	802	168	544	82.113	175	1082	268	390	78.877
0.01		825	145	43	81.722		1098	252	78	76.820
0.001		827	143	2	81.703		1100	250	19	76.310
1.00E-04		827	143	0	81.723		1101	249	12	76.362
1.00E-05		827	143	0	81.723		1103	247	7	76.387
1.00E-06		827	143	0	81.723		1104	246	3	76.359
0.1	100	933	207	425	80.574	200	1100	267	411	78.642
0.01		945	195	59	80.204		1115	252	67	75.587
0.001		950	190	5	79.957		1116	251	7	74.972
1.00E-04		951	189	1	79.957		1117	250	1	74.986
1.00E-05		951	189	1	79.957		1117	250	0	75.000
1.00E-06		951	189	1	79.957		1117	250	0	75.000

Table B-15: *The results of the application of the NOSC algorithm to the MMP1 activity class described by ECFP_4 fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	1094	262	444	78.957	325	1050	251	677	77.861
0.01		1105	251	112	74.901		1072	229	127	70.034
0.001		1108	248	37	73.786		1073	228	2	67.767
1.00E-04		1109	247	28	73.667		1073	228	1	67.767
1.00E-05		1111	245	24	73.681		1073	228	0	67.784
1.00E-06		1112	244	21	73.655		1073	228	0	67.784
0.1	250	1085	281	497	77.330	350	1043	238	739	77.345
0.01		1098	268	167	74.068		1061	220	316	72.242
0.001		1102	264	25	71.985		1064	217	34	67.383
1.00E-04		1102	264	16	71.921		1066	215	15	67.112
1.00E-05		1103	263	13	71.991		1066	215	1	66.915
1.00E-06		1104	262	6	72.007		1066	215	0	66.915
0.1	275	1070	264	574	77.951	375	1040	212	783	78.803
0.01		1084	250	233	74.190		1056	196	224	71.650
0.001		1088	246	6	71.148		1058	194	30	70.072
1.00E-04		1088	246	1	71.110		1058	194	19	70.031
1.00E-05		1088	246	0	71.125		1058	194	16	70.046
1.00E-06		1088	246	0	71.125		1059	193	8	69.974
0.1	300	1065	257	634	78.491	400	1005	196	879	78.376
0.01		1078	244	265	73.843		1019	182	423	73.128
0.001		1089	233	68	71.367		1022	179	174	70.698
1.00E-04		1091	231	61	71.336		1025	176	156	71.070
1.00E-05		1091	231	52	71.282		1026	175	150	70.931
1.00E-06		1092	230	46	71.282		1026	175	133	71.015

Table B-16: *The results of the application of the NOSC algorithm to the MMP1 activity class described by ECFP_4 fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	10	0	3398	66.667	125	202	23	1929	77.864
0.01		11	0	1645	64.350		208	17	342	68.812
0.001		11	0	4	42.712		208	17	1	66.514
1.00E-04		11	0	0	42.655		208	17	0	66.484
1.00E-05		11	0	0	42.655		208	17	0	66.484
1.00E-06		11	0	0	42.655		208	17	0	66.484
0.1	50	44	3	2963	78.916	150	257	30	1666	78.358
0.01		46	2	813	61.290		263	24	203	71.345
0.001		46	2	9	56.939		263	24	1	69.039
1.00E-04		46	2	0	56.836		263	24	0	69.039
1.00E-05		46	2	0	56.836		263	24	0	69.039
1.00E-06		46	2	0	56.836		263	24	0	69.039
0.1	75	88	9	2473	82.328	175	317	44	1431	77.810
0.01		90	8	601	67.493		324	37	217	73.705
0.001		90	8	12	61.171		324	37	0	71.105
1.00E-04		90	8	0	61.015		324	37	0	71.105
1.00E-05		90	8	0	61.015		324	37	0	71.105
1.00E-06		90	8	0	61.015		324	37	0	71.105
0.1	100	140	17	2162	80.573	200	344	57	1327	77.239
0.01		143	14	313	68.843		354	47	148	72.293
0.001		143	14	7	66.168		355	46	1	71.340
1.00E-04		143	14	0	66.079		355	46	0	71.306
1.00E-05		143	14	0	66.079		355	46	0	71.306
1.00E-06		143	14	0	66.079		355	46	0	71.306

Table B-17: *The results of the application of the NOSC algorithm to the MMP1 activity class described by MDL public keys, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	389	61	1186	75.904	325	504	87	937	77.607
0.01		395	55	156	74.226		515	76	113	76.970
0.001		395	55	2	73.172		515	76	1	76.995
1.00E-04		395	55	0	73.197		515	76	0	76.957
1.00E-05		395	55	0	73.197		515	76	0	76.957
1.00E-06		395	55	0	73.197		515	76	0	76.957
0.1	250	419	74	1082	77.254	350	535	104	862	77.819
0.01		426	67	85	75.713		547	92	79	77.270
0.001		426	67	4	75.334		548	91	0	77.745
1.00E-04		426	67	0	75.455		548	91	0	77.745
1.00E-05		426	67	0	75.455		548	91	0	77.745
1.00E-06		426	67	0	75.455		548	91	0	77.745
0.1	275	455	82	961	78.017	375	553	108	829	77.502
0.01		461	76	108	77.218		565	96	69	77.247
0.001		461	76	9	76.865		565	96	4	76.946
1.00E-04		461	76	0	76.896		565	96	0	76.981
1.00E-05		461	76	0	76.896		565	96	0	76.981
1.00E-06		461	76	0	76.896		565	96	0	76.981
0.1	300	474	82	971	77.906	400	557	121	831	77.826
0.01		478	78	96	76.805		573	105	93	77.496
0.001		479	77	1	76.824		574	104	7	77.439
1.00E-04		479	77	0	76.751		574	104	0	77.411
1.00E-05		479	77	0	76.751		574	104	0	77.411
1.00E-06		479	77	0	76.751		574	104	0	77.411

Table B-18: *The results of the application of the NOSC algorithm to the MMP1 activity class described by MDL public keys, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	64	5	2494	75.311	125	414	94	985	78.333
0.01		66	4	419	58.805		423	85	93	76.157
0.001		67	3	10	55.993		424	84	2	75.706
1.00E-04		67	3	0	55.945		424	84	0	75.706
1.00E-05		67	3	0	55.945		424	84	0	75.706
1.00E-06		67	3	0	55.945		424	84	0	75.706
0.1	50	177	20	1814	78.286	150	474	117	849	79.450
0.01		181	16	222	72.356		489	102	73	77.647
0.001		181	16	1	70.630		490	101	0	77.761
1.00E-04		181	16	0	70.599		490	101	0	77.761
1.00E-05		181	16	0	70.599		490	101	0	77.761
1.00E-06		181	16	0	70.599		490	101	0	77.761
0.1	75	274	49	1268	79.934	175	504	154	786	78.736
0.01		278	45	107	73.577		526	132	55	77.644
0.001		280	43	0	73.384		526	132	2	77.163
1.00E-04		280	43	0	73.384		526	132	0	77.174
1.00E-05		280	43	0	73.384		526	132	0	77.174
1.00E-06		280	43	0	73.384		526	132	0	77.174
0.1	100	352	71	1152	77.955	200	541	159	694	78.321
0.01		361	62	110	75.561		558	142	72	77.709
0.001		362	61	3	75.406		558	142	3	77.101
1.00E-04		362	61	0	75.511		558	142	0	77.124
1.00E-05		362	61	0	75.511		558	142	0	77.124
1.00E-06		362	61	0	75.511		558	142	0	77.124

Table B-19: *The results of the application of the NOSC algorithm to the MMP1 activity class described by Unity fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	572	169	711	78.316	325	665	195	583	78.214
0.01		589	152	67	78.032		684	176	57	78.231
0.001		590	151	2	77.846		689	171	13	77.934
1.00E-04		590	151	0	77.857		689	171	5	77.784
1.00E-05		590	151	0	77.857		689	171	2	77.784
1.00E-06		590	151	0	77.857		689	171	2	77.784
0.1	250	600	176	612	78.721	350	696	193	555	78.630
0.01		615	161	43	78.306		717	172	50	77.965
0.001		615	161	2	77.964		721	168	5	77.586
1.00E-04		615	161	0	77.987		722	167	3	77.598
1.00E-05		615	161	0	77.987		722	167	3	77.598
1.00E-06		615	161	0	77.987		722	167	3	77.598
0.1	275	616	187	629	77.785	375	717	190	545	79.087
0.01		636	167	95	77.732		736	171	40	77.460
0.001		640	163	3	77.312		736	171	2	77.144
1.00E-04		640	163	0	77.273		736	171	2	77.144
1.00E-05		640	163	0	77.273		736	171	0	77.168
1.00E-06		640	163	0	77.273		736	171	0	77.168
0.1	300	639	198	599	77.677	400	727	215	534	78.647
0.01		663	174	39	77.888		749	193	87	77.154
0.001		666	171	5	77.703		752	190	6	76.105
1.00E-04		666	171	2	77.738		753	189	5	76.170
1.00E-05		666	171	2	77.738		753	189	4	76.055
1.00E-06		667	170	1	77.738		753	189	4	76.055

Table B-20: *The results of the application of the NOSC algorithm to the MMP1 activity class described by Unity fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	23	2	1815	64.646	125	157	15	908	72.162
0.01		25	1	498	34.436		160	12	52	52.971
0.001		25	1	4	45.929		161	11	0	52.980
1.00E-04		25	1	0	46.044		161	11	0	52.980
1.00E-05		25	1	0	46.044		161	11	0	52.980
1.00E-06		25	1	0	46.044		161	11	0	52.980
0.1	50	54	6	1424	63.224	150	190	24	783	68.917
0.01		56	4	201	51.649		193	21	60	55.836
0.001		56	4	6	48.332		193	21	1	54.740
1.00E-04		56	4	0	47.688		193	21	0	54.765
1.00E-05		56	4	0	47.688		193	21	0	54.765
1.00E-06		56	4	0	47.688		193	21	0	54.765
0.1	75	97	8	1174	66.088	175	221	31	668	67.213
0.01		99	6	96	49.915		228	24	30	60.829
0.001		99	6	0	51.613		228	24	0	60.675
1.00E-04		99	6	0	51.613		228	24	0	60.675
1.00E-05		99	6	0	51.613		228	24	0	60.675
1.00E-06		99	6	0	51.613		228	24	0	60.675
0.1	100	118	11	1101	70.896	200	244	44	622	67.403
0.01		121	8	133	52.775		252	36	43	60.833
0.001		121	8	2	49.343		254	34	0	60.480
1.00E-04		121	8	0	49.344		254	34	0	60.480
1.00E-05		121	8	0	49.344		254	34	0	60.480
1.00E-06		121	8	0	49.344		254	34	0	60.480

Table B-21: *The results of the application of the NOSC algorithm to the Renin activity class described by BCI fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	275	54	553	66.691	325	332	81	482	64.668
0.01		282	47	22	62.313		341	72	39	59.356
0.001		283	46	0	61.931		342	71	7	61.305
1.00E-04		283	46	0	61.931		342	71	0	61.585
1.00E-05		283	46	0	61.931		342	71	0	61.585
1.00E-06		283	46	0	61.931		342	71	0	61.585
0.1	250	283	58	583	68.843	350	347	85	442	64.804
0.01		289	52	70	61.158		356	76	24	58.654
0.001		290	51	2	60.803		356	76	0	58.236
1.00E-04		290	51	0	60.791		356	76	0	58.236
1.00E-05		290	51	0	60.791		356	76	0	58.236
1.00E-06		290	51	0	60.791		356	76	0	58.236
0.1	275	299	64	549	65.632	375	370	85	397	64.183
0.01		308	55	58	60.938		377	78	27	59.841
0.001		308	55	5	60.539		380	75	0	62.056
1.00E-04		308	55	0	59.868		380	75	0	62.056
1.00E-05		308	55	0	59.868		380	75	0	62.056
1.00E-06		308	55	0	59.868		380	75	0	62.056
0.1	300	320	69	469	65.349	400	370	96	402	64.569
0.01		328	61	50	60.611		378	88	35	60.045
0.001		328	61	3	60.089		381	85	3	59.966
1.00E-04		328	61	0	60.100		381	85	0	59.933
1.00E-05		328	61	0	60.100		381	85	0	59.933
1.00E-06		328	61	0	60.100		381	85	0	59.933

Table B-22: *The results of the application of the NOSC algorithm to the Renin activity class described by BCI fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	42	2	1566	76.067	125	264	56	602	63.351
0.01		44	1	585	49.304		272	48	50	55.995
0.001		44	1	6	47.428		273	47	0	55.814
1.00E-04		44	1	0	47.354		273	47	0	55.814
1.00E-05		44	1	0	47.354		273	47	0	55.814
1.00E-06		44	1	0	47.354		273	47	0	55.814
0.1	50	111	12	1155	71.073	150	291	65	588	63.379
0.01		116	7	111	50.489		298	58	97	60.952
0.001		116	7	0	49.563		300	56	2	59.901
1.00E-04		116	7	0	49.563		300	56	0	59.868
1.00E-05		116	7	0	49.563		300	56	0	59.868
1.00E-06		116	7	0	49.563		300	56	0	59.868
0.1	75	187	29	899	62.663	175	318	76	570	64.470
0.01		195	21	82	57.094		324	70	71	61.977
0.001		195	21	1	54.192		325	69	2	61.023
1.00E-04		195	21	0	54.162		325	69	0	61.065
1.00E-05		195	21	0	54.162		325	69	0	61.065
1.00E-06		195	21	0	54.162		325	69	0	61.065
0.1	100	233	41	685	62.069	200	353	85	432	65.223
0.01		237	37	71	54.872		358	80	28	58.993
0.001		238	36	0	57.166		361	77	3	58.939
1.00E-04		238	36	0	57.166		361	77	2	58.961
1.00E-05		238	36	0	57.166		361	77	2	58.961
1.00E-06		238	36	0	57.166		361	77	2	58.961

Table B-23: *The results of the application of the NOSC algorithm to the Renin activity class described by Daylight fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	357	95	446	64.918	325	416	116	332	65.485
0.01		368	84	34	61.907		426	106	51	64.216
0.001		370	82	0	61.931		428	104	10	64.474
1.00E-04		370	82	0	61.931		430	102	2	64.745
1.00E-05		370	82	0	61.931		430	102	2	64.745
1.00E-06		370	82	0	61.931		430	102	2	64.745
0.1	250	369	107	384	65.827	350	426	121	296	65.727
0.01		381	95	32	62.570		435	112	20	64.918
0.001		381	95	4	62.646		436	111	5	64.518
1.00E-04		381	95	1	62.895		437	110	0	64.860
1.00E-05		381	95	0	62.916		437	110	0	64.860
1.00E-06		381	95	0	62.916		437	110	0	64.860
0.1	275	398	109	324	65.627	375	429	125	310	65.367
0.01		404	103	18	62.746		434	120	49	64.237
0.001		407	100	0	62.612		437	117	3	64.090
1.00E-04		407	100	0	62.612		437	117	1	64.018
1.00E-05		407	100	0	62.612		437	117	1	64.018
1.00E-06		407	100	0	62.612		437	117	1	64.018
0.1	300	401	113	352	64.551	400	428	128	317	64.811
0.01		408	106	21	62.802		434	122	26	63.729
0.001		409	105	4	63.657		437	119	2	63.667
1.00E-04		409	105	2	63.434		437	119	0	63.631
1.00E-05		409	105	1	63.454		437	119	0	63.631
1.00E-06		409	105	1	63.454		437	119	0	63.631

Table B-24: *The results of the application of the NOSC algorithm to the Renin activity class described by Daylight fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	81	3	1305	81.791	125	521	108	344	83.002
0.01		81	3	269	75.830		541	88	19	83.295
0.001		81	3	12	72.370		542	87	0	83.125
1.00E-04		81	3	0	72.207		542	87	0	83.125
1.00E-05		81	3	0	72.207		542	87	0	83.125
1.00E-06		81	3	0	72.207		542	87	0	83.125
0.1	50	224	19	904	83.063	150	601	123	243	82.803
0.01		230	13	94	76.987		612	112	7	83.248
0.001		230	13	4	76.941		612	112	0	83.257
1.00E-04		230	13	0	76.993		612	112	0	83.257
1.00E-05		230	13	0	76.993		612	112	0	83.257
1.00E-06		230	13	0	76.993		612	112	0	83.257
0.1	75	344	56	637	81.833	175	633	135	225	82.138
0.01		360	40	57	80.488		642	126	25	81.986
0.001		360	40	3	80.936		646	122	0	82.125
1.00E-04		360	40	0	80.845		646	122	0	82.125
1.00E-05		360	40	0	80.845		646	122	0	82.125
1.00E-06		360	40	0	80.845		646	122	0	82.125
0.1	100	448	80	413	81.479	200	666	156	180	81.098
0.01		464	64	11	82.299		677	145	19	81.409
0.001		465	63	1	82.276		678	144	2	81.511
1.00E-04		465	63	0	82.276		678	144	0	81.532
1.00E-05		465	63	0	82.276		678	144	0	81.532
1.00E-06		465	63	0	82.276		678	144	0	81.532

Table B-25: *The results of the application of the NOSC algorithm to the Renin activity class described by ECFP_4 fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	676	178	178	81.343	325	712	186	191	80.776
0.01		692	162	9	80.425		722	176	16	78.444
0.001		697	157	0	80.789		723	175	4	78.248
1.00E-04		697	157	0	80.789		724	174	0	78.343
1.00E-05		697	157	0	80.789		724	174	0	78.343
1.00E-06		697	157	0	80.789		724	174	0	78.343
0.1	250	695	178	169	81.801	350	720	177	201	81.014
0.01		706	167	14	81.139		727	170	29	77.546
0.001		711	162	0	81.314		730	167	5	77.549
1.00E-04		711	162	0	81.314		731	166	0	77.600
1.00E-05		711	162	0	81.314		731	166	0	77.600
1.00E-06		711	162	0	81.314		731	166	0	77.600
0.1	275	708	173	172	82.209	375	726	172	219	81.767
0.01		720	161	35	80.751		733	165	13	76.582
0.001		724	157	2	80.481		734	164	4	76.403
1.00E-04		725	156	0	80.560		734	164	0	76.444
1.00E-05		725	156	0	80.560		734	164	0	76.444
1.00E-06		725	156	0	80.560		734	164	0	76.444
0.1	300	716	188	169	81.741	400	723	170	227	81.739
0.01		726	178	18	78.966		732	161	13	78.141
0.001		727	177	1	78.935		734	159	3	77.911
1.00E-04		728	176	0	79.005		734	159	1	77.936
1.00E-05		728	176	0	79.005		734	159	0	77.949
1.00E-06		728	176	0	79.005		734	159	0	77.949

Table B-26: *The results of the application of the NOSC algorithm to the Renin activity class described by ECFP_4 fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	3	0	2127	93.333	125	60	7	1307	81.456
0.01		4	0	689	83.243		61	6	286	70.618
0.001		4	0	12	72.346		61	6	12	65.691
1.00E-04		4	0	0	72.081		61	6	0	65.529
1.00E-05		4	0	0	72.081		61	6	0	65.529
1.00E-06		4	0	0	72.081		61	6	0	65.529
0.1	50	15	2	1872	92.562	150	81	13	1142	79.472
0.01		17	1	624	82.019		82	12	174	70.501
0.001		17	1	16	70.776		83	11	4	67.923
1.00E-04		17	1	0	70.449		83	11	0	67.849
1.00E-05		17	1	0	70.449		83	11	0	67.849
1.00E-06		17	1	0	70.449		83	11	0	67.849
0.1	75	30	2	1671	87.871	175	114	16	943	78.235
0.01		30	2	514	66.465		117	13	83	72.826
0.001		30	2	1	40.497		117	13	1	73.052
1.00E-04		30	2	0	40.497		117	13	0	73.012
1.00E-05		30	2	0	40.497		117	13	0	73.012
1.00E-06		30	2	0	40.497		117	13	0	73.012
0.1	100	45	7	1463	77.273	200	127	20	991	79.008
0.01		45	7	284	60.238		130	17	132	73.870
0.001		45	7	2	54.898		131	16	4	72.747
1.00E-04		45	7	0	54.898		131	16	0	72.808
1.00E-05		45	7	0	54.898		131	16	0	72.808
1.00E-06		45	7	0	54.898		131	16	0	72.808

Table B-27: *The results of the application of the NOSC algorithm to the Renin activity class described by MDL public keys, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	159	21	876	78.412	325	229	33	680	75.618
0.01		163	17	102	73.936		234	28	52	76.481
0.001		163	17	0	72.500		235	27	1	76.800
1.00E-04		163	17	0	72.500		235	27	0	76.757
1.00E-05		163	17	0	72.500		235	27	0	76.757
1.00E-06		163	17	0	72.500		235	27	0	76.757
0.1	250	179	25	774	74.229	350	244	29	641	76.053
0.01		186	18	69	71.313		249	24	51	76.089
0.001		187	17	6	71.756		251	22	3	75.812
1.00E-04		187	17	0	71.739		251	22	0	75.741
1.00E-05		187	17	0	71.739		251	22	0	75.741
1.00E-06		187	17	0	71.739		251	22	0	75.741
0.1	275	193	32	782	74.839	375	260	41	604	75.926
0.01		200	25	77	73.399		266	35	61	75.116
0.001		201	24	1	73.643		268	33	2	75.056
1.00E-04		201	24	0	73.658		268	33	0	74.902
1.00E-05		201	24	0	73.658		268	33	0	74.902
1.00E-06		201	24	0	73.658		268	33	0	74.902
0.1	300	212	29	754	74.526	400	282	40	573	76.164
0.01		219	22	87	73.850		288	34	41	74.511
0.001		219	22	3	75.527		288	34	4	74.718
1.00E-04		219	22	0	75.554		288	34	0	73.928
1.00E-05		219	22	0	75.554		288	34	0	73.928
1.00E-06		219	22	0	75.554		288	34	0	73.928

Table B-28: *The results of the application of the NOSC algorithm to the Renin activity class described by MDL public keys, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	19	0	1858	47.699	125	162	29	960	64.327
0.01		20	0	455	32.349		165	26	106	55.340
0.001		20	0	0	45.474		165	26	2	54.357
1.00E-04		20	0	0	45.474		165	26	0	54.382
1.00E-05		20	0	0	45.474		165	26	0	54.382
1.00E-06		20	0	0	45.474		165	26	0	54.382
0.1	50	50	2	1542	68.089	150	194	34	834	65.794
0.01		51	1	411	52.457		199	29	36	53.344
0.001		51	1	5	53.227		199	29	1	52.996
1.00E-04		51	1	1	49.570		199	29	0	53.022
1.00E-05		51	1	0	49.597		199	29	0	53.022
1.00E-06		51	1	0	49.597		199	29	0	53.022
0.1	75	87	14	1272	69.515	175	215	41	718	64.117
0.01		92	9	164	50.824		220	36	66	56.652
0.001		95	6	0	51.653		220	36	2	56.659
1.00E-04		95	6	0	51.653		220	36	0	56.598
1.00E-05		95	6	0	51.653		220	36	0	56.598
1.00E-06		95	6	0	51.653		220	36	0	56.598
0.1	100	128	19	1075	64.187	200	239	63	644	63.043
0.01		132	15	125	50.290		245	57	33	57.063
0.001		133	14	3	50.682		245	57	0	56.247
1.00E-04		133	14	0	50.709		245	57	0	56.247
1.00E-05		133	14	0	50.709		245	57	0	56.247
1.00E-06		133	14	0	50.709		245	57	0	56.247

Table B-29: *The results of the application of the NOSC algorithm to the Renin activity class described by Unity fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	274	62	539	60.976	325	335	76	424	63.296
0.01		280	56	27	56.517		345	66	25	65.781
0.001		282	54	2	56.763		348	63	0	66.393
1.00E-04		282	54	0	56.731		348	63	0	66.393
1.00E-05		282	54	0	56.731		348	63	0	66.393
1.00E-06		282	54	0	56.731		348	63	0	66.393
0.1	250	290	66	495	59.841	350	341	83	423	64.172
0.01		298	58	34	58.315		351	73	30	62.703
0.001		298	58	3	58.573		351	73	1	62.852
1.00E-04		298	58	2	58.596		351	73	1	62.852
1.00E-05		298	58	0	58.586		351	73	1	62.852
1.00E-06		298	58	0	58.586		351	73	1	62.852
0.1	275	314	70	455	62.482	375	360	88	358	63.649
0.01		321	63	19	59.564		367	81	29	62.240
0.001		322	62	0	60.310		368	80	2	62.299
1.00E-04		322	62	0	60.310		368	80	1	62.320
1.00E-05		322	62	0	60.310		368	80	1	62.320
1.00E-06		322	62	0	60.310		368	80	1	62.320
0.1	300	326	73	455	64.532	400	370	87	362	65.346
0.01		332	67	23	60.894		376	81	33	65.098
0.001		333	66	0	61.500		376	81	3	65.213
1.00E-04		333	66	0	61.500		376	81	2	65.177
1.00E-05		333	66	0	61.500		376	81	1	65.196
1.00E-06		333	66	0	61.500		376	81	1	65.196

Table B-30: *The results of the application of the NOSC algorithm to the Renin activity class described by Unity fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	80	3	1415	57.895	125	353	77	631	63.501
0.01		82	1	92	53.175		363	67	40	62.472
0.001		82	1	0	53.925		363	67	5	62.610
1.00E-04		82	1	0	53.925		364	66	0	62.651
1.00E-05		82	1	0	53.925		364	66	0	62.651
1.00E-06		82	1	0	53.925		364	66	0	62.651
0.1	50	182	16	974	64.138	150	384	101	606	63.228
0.01		188	10	87	59.484		396	89	35	61.998
0.001		188	10	2	59.111		398	87	3	61.826
1.00E-04		188	10	0	59.101		398	87	0	61.792
1.00E-05		188	10	0	59.101		398	87	0	61.792
1.00E-06		188	10	0	59.101		398	87	0	61.792
0.1	75	255	37	828	62.818	175	417	106	532	63.895
0.01		262	30	55	58.873		428	95	30	63.175
0.001		262	30	0	59.043		428	95	0	63.415
1.00E-04		262	30	0	59.043		428	95	0	63.415
1.00E-05		262	30	0	59.043		428	95	0	63.415
1.00E-06		262	30	0	59.043		428	95	0	63.415
0.1	100	310	58	714	62.794	200	458	113	484	64.108
0.01		322	46	65	61.307		465	106	34	62.854
0.001		323	45	0	61.175		466	105	1	62.939
1.00E-04		323	45	0	61.175		466	105	1	62.939
1.00E-05		323	45	0	61.175		466	105	1	62.939
1.00E-06		323	45	0	61.175		466	105	0	62.939

Table B-31: *The results of the application of the NOSC algorithm to the SubP activity class described by BCI fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	476	105	503	65.376	325	534	141	450	64.428
0.01		484	97	48	64.011		549	126	40	63.233
0.001		485	96	4	63.277		550	125	0	63.146
1.00E-04		485	96	3	63.297		550	125	0	63.146
1.00E-05		485	96	3	63.297		550	125	0	63.146
1.00E-06		485	96	2	63.297		550	125	0	63.146
0.1	250	491	126	494	64.534	350	538	156	425	64.082
0.01		501	116	45	62.946		554	140	40	63.284
0.001		502	115	5	62.859		556	138	2	62.557
1.00E-04		502	115	1	62.920		557	137	0	62.557
1.00E-05		502	115	1	62.920		557	137	0	62.557
1.00E-06		502	115	1	62.920		557	137	0	62.557
0.1	275	503	134	465	65.051	375	551	157	411	64.742
0.01		510	127	46	62.973		564	144	56	64.028
0.001		512	125	2	63.234		566	142	15	63.776
1.00E-04		512	125	0	63.199		567	141	8	63.853
1.00E-05		512	125	0	63.199		567	141	3	63.842
1.00E-06		512	125	0	63.199		567	141	0	63.847
0.1	300	524	130	451	65.173	400	573	154	389	64.531
0.01		531	123	41	63.196		588	139	42	63.954
0.001		532	122	0	63.172		590	137	3	64.032
1.00E-04		532	122	0	63.172		590	137	0	64.032
1.00E-05		532	122	0	63.172		590	137	0	64.032
1.00E-06		532	122	0	63.172		590	137	0	64.032

Table B-32: *The results of the application of the NOSC algorithm to the SubP activity class described by BCI fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	84	1	1563	60.710	125	411	116	538	61.436
0.01		85	0	216	51.072		424	103	32	59.280
0.001		85	0	2	50.261		426	101	1	59.176
1.00E-04		85	0	0	50.234		426	101	0	59.142
1.00E-05		85	0	0	50.234		426	101	0	59.142
1.00E-06		85	0	0	50.234		426	101	0	59.142
0.1	50	230	35	872	61.134	150	450	122	467	61.081
0.01		237	28	50	58.154		460	112	37	60.573
0.001		237	28	0	57.786		460	112	3	60.272
1.00E-04		237	28	0	57.786		460	112	0	60.260
1.00E-05		237	28	0	57.786		460	112	0	60.260
1.00E-06		237	28	0	57.786		460	112	0	60.260
0.1	75	319	63	691	62.537	175	477	137	479	61.340
0.01		325	57	40	59.831		494	120	23	61.321
0.001		325	57	1	60.033		495	119	0	60.443
1.00E-04		325	57	0	60.000		495	119	0	60.443
1.00E-05		325	57	0	60.000		495	119	0	60.443
1.00E-06		325	57	0	60.000		495	119	0	60.443
0.1	100	373	95	560	62.526	200	502	132	468	61.576
0.01		386	82	29	60.918		515	119	45	61.203
0.001		386	82	0	60.677		516	118	4	60.729
1.00E-04		386	82	0	60.677		516	118	1	60.739
1.00E-05		386	82	0	60.677		516	118	0	60.704
1.00E-06		386	82	0	60.677		516	118	0	60.704

Table B-33: *The results of the application of the NOSC algorithm to the SubP activity class described by Daylight fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	513	137	479	61.609	325	567	157	427	61.571
0.01		530	120	42	61.322		580	144	36	59.707
0.001		532	118	3	61.617		581	143	4	59.827
1.00E-04		532	118	1	61.660		581	143	1	59.816
1.00E-05		532	118	0	61.682		581	143	1	59.816
1.00E-06		532	118	0	61.682		581	143	0	59.839
0.1	250	529	152	454	61.587	350	582	164	425	61.732
0.01		545	136	61	60.362		591	155	69	59.893
0.001		548	133	6	60.525		593	153	11	59.628
1.00E-04		548	133	1	60.501		593	153	1	59.862
1.00E-05		548	133	0	60.523		593	153	1	59.862
1.00E-06		548	133	0	60.523		593	153	0	59.885
0.1	275	544	150	446	62.021	375	593	176	438	62.057
0.01		554	140	65	60.392		603	166	55	60.454
0.001		559	135	18	60.491		604	165	5	59.494
1.00E-04		559	135	5	60.534		605	164	1	59.563
1.00E-05		559	135	0	60.567		605	164	1	59.563
1.00E-06		559	135	0	60.567		605	164	0	59.529
0.1	300	550	156	467	61.538	400	595	186	435	62.612
0.01		563	143	81	60.405		603	178	120	60.686
0.001		566	140	9	59.861		606	175	31	59.128
1.00E-04		566	140	3	59.791		607	174	25	59.224
1.00E-05		566	140	0	59.780		608	173	11	58.947
1.00E-06		566	140	0	59.780		609	172	1	59.351

Table B-34: *The results of the application of the NOSC algorithm to the SubP activity class described by Daylight fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	217	17	990	57.092	125	740	161	354	59.454
0.01		223	11	107	53.178		750	151	20	60.604
0.001		224	10	11	53.264		753	148	1	60.706
1.00E-04		224	10	0	53.257		753	148	1	60.706
1.00E-05		224	10	0	53.257		753	148	1	60.706
1.00E-06		224	10	0	53.257		753	148	1	60.706
0.1	50	439	64	689	61.968	150	788	172	292	59.891
0.01		450	53	33	55.993		803	157	25	59.964
0.001		450	53	1	55.840		804	156	1	59.916
1.00E-04		450	53	0	55.840		804	156	1	59.916
1.00E-05		450	53	0	55.840		804	156	1	59.916
1.00E-06		450	53	0	55.840		804	156	1	59.916
0.1	75	571	104	523	62.935	175	821	181	284	60.258
0.01		587	88	53	59.075		835	167	27	59.611
0.001		588	87	15	58.820		836	166	5	59.578
1.00E-04		588	87	10	58.824		837	165	1	59.520
1.00E-05		588	87	8	58.813		837	165	1	59.520
1.00E-06		589	86	4	58.712		837	165	1	59.520
0.1	100	653	142	429	60.745	200	836	188	291	59.351
0.01		665	130	40	60.344		851	173	27	59.988
0.001		669	126	7	60.152		852	172	12	60.256
1.00E-04		669	126	2	60.174		852	172	6	60.341
1.00E-05		669	126	1	60.198		854	170	2	60.413
1.00E-06		669	126	1	60.198		854	170	2	60.413

Table B-35: *The results of the application of the NOSC algorithm to the SubP activity class described by ECFP_4 fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	842	190	306	60.069	325	818	187	472	60.241
0.01		858	174	43	59.063		841	164	95	58.700
0.001		862	170	16	58.716		841	164	56	56.957
1.00E-04		862	170	7	58.917		842	163	48	57.116
1.00E-05		863	169	6	58.881		842	163	47	57.143
1.00E-06		864	168	4	58.991		843	162	43	57.125
0.1	250	835	208	339	60.169	350	809	174	527	60.355
0.01		855	188	58	59.133		824	159	342	60.918
0.001		860	183	20	59.044		830	153	80	58.478
1.00E-04		860	183	15	59.143		831	152	75	58.667
1.00E-05		860	183	14	59.168		831	152	69	58.636
1.00E-06		862	181	9	59.230		832	151	65	58.737
0.1	275	832	200	371	59.301	375	802	169	554	59.291
0.01		851	181	64	58.521		815	156	198	58.129
0.001		851	181	6	57.631		821	150	102	58.135
1.00E-04		852	180	0	57.517		822	149	92	58.500
1.00E-05		852	180	0	57.517		822	149	86	58.469
1.00E-06		852	180	0	57.517		823	148	81	58.629
0.1	300	828	187	427	60.147	400	791	172	583	59.505
0.01		850	165	76	59.429		800	163	193	58.557
0.001		850	165	3	59.239		804	159	117	57.979
1.00E-04		850	165	0	59.312		806	157	107	58.134
1.00E-05		850	165	0	59.312		807	156	99	58.313
1.00E-06		850	165	0	59.312		807	156	92	58.104

Table B-36: *The results of the application of the NOSC algorithm to the SubP activity class described by ECFP_4 fingerprints, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	2	0	2751	75.000	125	207	31	1128	57.026
0.01		3	0	1304	10.375		213	25	95	48.690
0.001		3	0	107	19.773		213	25	6	48.375
1.00E-04		3	0	3	21.008		213	25	0	48.350
1.00E-05		3	0	0	21.029		213	25	0	48.350
1.00E-06		3	0	0	21.029		213	25	0	48.350
0.1	50	36	1	2219	51.220	150	267	41	890	57.246
0.01		37	0	602	38.477		275	33	84	51.501
0.001		37	0	32	43.107		275	33	1	51.350
1.00E-04		37	0	0	42.921		275	33	0	51.377
1.00E-05		37	0	0	42.921		275	33	0	51.377
1.00E-06		37	0	0	42.921		275	33	0	51.377
0.1	75	93	7	1647	51.318	175	314	48	786	58.028
0.01		97	3	240	44.616		318	44	81	52.092
0.001		97	3	3	43.714		318	44	2	51.403
1.00E-04		97	3	0	43.669		318	44	0	51.401
1.00E-05		97	3	0	43.669		318	44	0	51.401
1.00E-06		97	3	0	43.669		318	44	0	51.401
0.1	100	157	23	1312	55.187	200	359	60	710	58.346
0.01		166	14	138	47.875		365	54	65	54.983
0.001		166	14	0	46.300		367	52	2	54.872
1.00E-04		166	14	0	46.300		367	52	0	54.867
1.00E-05		166	14	0	46.300		367	52	0	54.867
1.00E-06		166	14	0	46.300		367	52	0	54.867

Table B-37: *The results of the application of the NOSC algorithm to the SubP activity class described by MDL public keys, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	380	74	672	57.371	325	468	104	582	57.439
0.01		394	60	82	55.404		478	94	61	56.207
0.001		396	58	2	54.961		480	92	1	55.772
1.00E-04		396	58	0	55.055		481	91	0	55.772
1.00E-05		396	58	0	55.055		481	91	0	55.772
1.00E-06		396	58	0	55.055		481	91	0	55.772
0.1	250	392	82	657	58.069	350	482	114	504	57.660
0.01		406	68	67	55.524		493	103	44	56.644
0.001		408	66	3	55.069		495	101	2	57.103
1.00E-04		408	66	0	55.008		495	101	0	57.119
1.00E-05		408	66	0	55.008		495	101	0	57.119
1.00E-06		408	66	0	55.008		495	101	0	57.119
0.1	275	423	88	633	56.158	375	503	124	483	59.135
0.01		435	76	83	54.243		514	113	45	57.640
0.001		437	74	1	53.457		516	111	3	57.694
1.00E-04		437	74	0	53.457		516	111	0	57.709
1.00E-05		437	74	0	53.457		516	111	0	57.709
1.00E-06		437	74	0	53.457		516	111	0	57.709
0.1	300	444	98	609	56.909	400	521	130	474	58.799
0.01		456	86	82	55.403		532	119	62	57.473
0.001		458	84	1	55.022		533	118	6	57.523
1.00E-04		458	84	0	55.022		533	118	1	57.538
1.00E-05		458	84	0	55.022		533	118	0	57.562
1.00E-06		458	84	0	55.022		533	118	0	57.562

Table B-38: *The results of the application of the NOSC algorithm to the SubP activity class described by MDL public keys, when γ = between 225 and 400.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	25	24	0	2326	62.216	125	327	72	774	61.142
0.01		25	0	560	38.536		338	61	58	58.450
0.001		25	0	28	37.906		339	60	2	58.519
1.00E-04		25	0	0	37.922		339	60	0	58.456
1.00E-05		25	0	0	37.922		339	60	0	58.456
1.00E-06		25	0	0	37.922		339	60	0	58.456
0.1	50	129	9	1444	60.571	150	375	100	660	61.784
0.01		131	7	154	50.939		389	86	50	60.044
0.001		131	7	2	50.080		391	84	1	59.542
1.00E-04		131	7	0	50.027		391	84	0	59.564
1.00E-05		131	7	0	50.027		391	84	0	59.564
1.00E-06		131	7	0	50.027		391	84	0	59.564
0.1	75	221	38	1045	61.993	175	413	109	568	63.392
0.01		231	28	49	56.932		423	99	50	59.989
0.001		233	26	0	56.959		424	98	0	59.714
1.00E-04		233	26	0	56.959		424	98	0	59.714
1.00E-05		233	26	0	56.959		424	98	0	59.714
1.00E-06		233	26	0	56.959		424	98	0	59.714
0.1	100	284	54	871	61.636	200	438	119	518	62.238
0.01		293	45	48	57.975		451	106	41	60.840
0.001		293	45	0	58.001		452	105	4	60.518
1.00E-04		293	45	0	58.001		452	105	0	60.495
1.00E-05		293	45	0	58.001		452	105	0	60.495
1.00E-06		293	45	0	58.001		452	105	0	60.495

Table B-39: *The results of the application of the NOSC algorithm to the SubP activity class described by Unity fingerprints, when γ = between 25 and 200.*

Eigenvector Threshold	γ	Clusters	Singletons	Unclassified Molecules	QCI	γ	Clusters	Singletons	Unclassified Molecules	QCI
0.1	225	458	121	514	62.639	325	551	160	378	60.792
0.01		470	109	80	61.743		565	146	51	60.900
0.001		474	105	8	60.973		566	145	11	60.540
1.00E-04		474	105	1	60.740		568	143	7	60.745
1.00E-05		474	105	0	60.706		568	143	7	60.745
1.00E-06		474	105	0	60.706		568	143	6	60.767
0.1	250	479	127	502	62.976	350	565	157	363	62.685
0.01		488	118	48	60.034		583	139	85	62.189
0.001		490	116	7	59.765		585	137	13	60.708
1.00E-04		490	116	0	59.732		586	136	8	60.741
1.00E-05		490	116	0	59.732		586	136	7	60.741
1.00E-06		490	116	0	59.732		586	136	6	60.764
0.1	275	507	144	445	62.849	375	580	160	347	62.608
0.01		517	134	46	61.433		596	144	68	61.575
0.001		517	134	8	61.032		598	142	16	60.999
1.00E-04		517	134	6	61.054		599	141	11	61.067
1.00E-05		518	133	2	61.042		599	141	7	60.997
1.00E-06		518	133	0	61.007		599	141	6	61.019
0.1	300	536	146	387	61.430	400	595	160	335	62.341
0.01		543	139	53	60.789		610	145	36	61.260
0.001		544	138	9	60.046		610	145	12	60.922
1.00E-04		545	137	7	60.092		611	144	5	60.999
1.00E-05		545	137	6	60.115		611	144	4	60.256
1.00E-06		545	137	6	60.115		611	144	3	60.221

Table B-40: *The results of the application of the NOSC algorithm to the SubP activity class described by Unity fingerprints, when γ = between 225 and 400.*

Appendix C

Chapter 6 Extended Results Tables

In this section ***Tables C.1 – C.4*** show the extended results of the investigation into how many of the largest eigenpairs, p, are retrieved, using the L-NOSC algorithm for the four ChEMBL activity classes:

- 5HT1A antagonists.
- Matrix Metalloprotease inhibitors.
- Renin inhibitors.
- Substance P antagonists.

When γ is varied between 25 and 100, at intervals of 25.

Tables C.5 and ***C.6*** then show the QCI comparison between the m-NOSC, L-NOSC, Ward's and k-means methods. Where spectral clustering is implemented at fours of γ (25, 50, 75 and 100), four values of k (100, 200, 300 and 400), when the similarity threshold is 0.01 and eigenvector threshold is 0.00001.

Descriptor	k	BCI		Daylight		MDL Public Keys		ECFP_4		Unity	
		p	% p	p	% p	p	% p	p	% p	p	% p
25	100	88	88.00	88	88.00	81	81.00	87	87.00	89	89.00
	200	167	83.50	173	86.50	163	81.50	167	83.50	175	87.50
	300	258	86.00	254	84.67	249	83.00	252	84.00	253	84.33
	400	333	83.25	323	80.75	348	87.00	326	81.50	328	82.00
50	100	82	82.00	85	85.00	87	87.00	87	87.00	84	84.00
	200	164	82.00	168	84.00	183	91.50	166	83.00	160	80.00
	300	251	83.67	254	84.67	283	94.33	251	83.67	242	80.67
	400	321	80.25	317	79.25	380	95.00	325	81.25	325	81.25
75	100	81	81.00	85	85.00	91	91.00	87	87.00	84	84.00
	200	161	80.50	168	84.00	195	97.50	162	81.00	160	80.00
	300	249	83.00	248	82.67	288	96.00	252	84.00	244	81.33
	400	324	81.00	313	78.25	387	96.75	332	83.00	319	79.75
100	100	81	81.00	83	83.00	97	97.00	85	85.00	83	83.00
	200	159	79.50	169	84.50	195	97.50	165	82.50	162	81.00
	300	249	83.00	249	83.00	291	97.00	248	82.67	248	82.67
	400	309	77.25	318	79.50	389	97.25	326	81.50	307	76.75

Table C-1: *The percentage of the largest eigenvalues, p, that were approximated using the L-NOSC method when k was set to 100, 200, 300 and 400, for the 5HT1A activity class.*

Descriptor	k	BCI		Daylight		MDL Public Keys		ECFP_4		Unity	
		p	% p	p	% p	p	% p	p	% p	p	% p
25	100	91	91.00	93	93.00	83	83.00	91	91.00	92	92.00
	200	176	88.00	175	87.50	165	82.50	173	86.50	177	88.50
	300	258	86.00	258	86.00	242	80.67	253	84.33	257	85.67
	400	341	85.25	348	87.00	336	84.00	329	82.25	339	84.75
50	100	92	92.00	92	92.00	85	85.00	90	90.00	90	90.00
	200	172	86.00	174	87.00	179	89.50	174	87.00	174	87.00
	300	263	87.67	260	86.67	279	93.00	255	85.00	261	87.00
	400	336	84.00	341	85.25	374	93.50	335	83.75	346	86.50
75	100	91	91.00	91	91.00	90	90.00	89	89.00	89	89.00
	200	173	86.50	169	84.50	188	94.00	173	86.50	174	87.00
	300	255	85.00	261	87.00	287	95.67	257	85.67	252	84.00
	400	344	86.00	335	83.75	386	96.50	337	84.25	336	84.00
100	100	88	88.00	87	87.00	95	95.00	91	91.00	87	87.00
	200	171	85.50	173	86.50	196	98.00	171	85.50	171	85.50
	300	251	83.67	260	86.67	293	97.67	256	85.33	250	83.33
	400	337	84.25	332	83.00	393	98.25	333	83.25	341	85.25

Table C-2: *The percentage of the largest eigenvalues, p, that were approximated using the L-NOSC method when k was set to 100, 200, 300 and 400, for the MMP1 activity class.*

Descriptor	k	BCI		Daylight		MDL Public Keys		ECFP_4		Unity	
		p	% p	p	% p	p	% p	p	% p	p	% p
25	100	84	84.00	86	86.00	89	89.00	87	87.00	87	87.00
	200	162	81.00	167	83.50	172	86.00	166	83.00	166	83.00
	300	243	81.00	249	83.00	242	80.67	242	80.67	246	82.00
	400	315	78.75	326	81.50	314	78.50	314	78.50	320	80.00
50	100	87	87.00	84	84.00	78	78.00	89	89.00	88	88.00
	200	165	82.50	165	82.50	167	83.50	173	86.50	168	84.00
	300	236	78.67	244	81.33	266	88.67	248	82.67	247	82.33
	400	326	81.50	328	82.00	366	91.50	327	81.75	339	84.75
75	100	86	86.00	83	83.00	84	84.00	91	91.00	90	90.00
	200	168	84.00	168	84.00	182	91.00	169	84.50	172	86.00
	300	246	82.00	245	81.67	285	95.00	248	82.67	247	82.33
	400	319	79.75	330	82.50	381	95.25	328	82.00	328	82.00
100	100	84	84.00	83	83.00	91	91.00	89	89.00	89	89.00
	200	171	85.50	166	83.00	188	94.00	167	83.50	171	85.50
	300	248	82.67	248	82.67	289	96.33	246	82.00	245	81.67
	400	318	79.50	325	81.25	386	96.50	327	81.75	337	84.25

Table C-3: *The percentage of the largest eigenvalues, p, that were approximated using the L-NOSC method when k was set to 100, 200, 300 and 400, for the Renin activity class.*

Descriptor	k	BCI		Daylight		MDL Public Keys		ECFP_4		Unity	
		p	% p	p	% p	p	% p	p	% p	p	% p
25	100	89	89.00	89	89.00	84	84.00	87	87.00	88	88.00
	200	169	84.50	175	87.50	162	81.00	172	86.00	174	87.00
	300	254	84.67	251	83.67	238	79.33	249	83.00	252	84.00
	400	342	85.50	339	84.75	323	80.75	332	83.00	337	84.25
50	100	87	87.00	87	87.00	81	81.00	86	86.00	87	87.00
	200	169	84.50	168	84.00	167	83.50	169	84.50	172	86.00
	300	252	84.00	251	83.67	265	88.33	248	82.67	257	85.67
	400	333	83.25	328	82.00	365	91.25	332	83.00	332	83.00
75	100	86	86.00	88	88.00	88	88.00	86	86.00	86	86.00
	200	171	85.50	168	84.00	185	92.50	162	81.00	180	90.00
	300	253	84.33	245	81.67	285	95.00	246	82.00	258	86.00
	400	336	84.00	330	82.50	379	94.75	330	82.50	328	82.00
100	100	86	86.00	78	78.00	91	91.00	85	85.00	85	85.00
	200	171	85.50	169	84.50	191	95.50	166	83.00	174	87.00
	300	252	84.00	249	83.00	285	95.00	243	81.00	255	85.00
	400	325	81.25	326	81.50	387	96.75	324	81.00	332	83.00

Table C-4: *The percentage of the largest eigenvalues, p, that were approximated using the L-NOSC method when k was set to 100, 200, 300 and 400, for the SubP activity class.*

Descriptor	k	5HT1A						MMP1					
		L-NOSC / mNOSC				k-means	Ward's	L-NOSC / mNOSC				k-means	Ward's
		γ = 25	γ = 50	γ = 75	γ = 100			γ = 25	γ = 50	γ = 75	γ = 100		
BCI	100	20.870	25.408	26.279	24.490	22.571	22.959	62.089	66.594	68.251	69.252	63.238	67.098
	200	25.780	26.353	26.480	31.057	27.636	29.063	67.495	68.893	68.805	71.462	70.588	71.812
	300	24.948	26.836	27.745	31.771	30.374	32.059	69.512	69.380	70.511	71.405	71.368	73.938
	400	26.845	28.480	29.738	30.876	31.923	32.447	70.847	70.343	71.818	71.681	73.079	75.024
Daylight	100	21.362	25.820	25.714	25.405	23.011	24.084	63.819	66.302	72.289	71.735	66.771	67.494
	200	23.192	27.391	29.518	29.060	25.699	25.218	66.366	70.665	72.800	72.830	70.492	70.420
	300	24.752	29.766	31.171	30.849	28.142	28.901	69.130	72.216	73.714	74.564	72.212	72.662
	400	26.375	29.925	32.194	31.827	29.670	30.653	71.681	73.510	74.574	74.623	73.735	74.346
ECFP_4	100	31.307	63.095	69.492	61.290	20.380	21.530	74.105	81.429	78.877	81.319	69.977	72.856
	200	41.953	60.417	61.111	55.000	25.271	27.820	75.356	80.046	79.671	82.864	76.591	78.846
	300	41.570	59.677	62.069	53.271	28.353	31.807	78.397	81.255	83.210	86.271	78.216	81.376
	400	52.356	54.032	64.085	63.846	30.831	33.130	78.780	82.970	83.353	86.807	80.223	82.593
MDL Public Keys	100	22.954	25.109	26.098	26.768	22.079	21.764	60.690	63.438	62.773	59.616	68.997	70.588
	200	26.931	29.657	28.894	31.786	26.036	26.739	62.763	67.266	68.723	63.865	74.615	75.642
	300	29.175	34.316	30.247	39.144	28.045	29.651	63.624	67.981	69.559	68.353	76.190	77.355
	400	31.194	36.616	34.082	42.550	28.769	31.186	62.983	68.652	70.363	70.866	77.507	79.108
Unity	100	21.397	22.924	21.277	25.000	21.229	22.344	64.991	69.792	71.490	72.412	68.565	71.719
	200	22.967	24.738	23.165	27.338	25.130	26.829	68.465	73.067	72.109	75.568	75.024	75.109
	300	24.899	27.547	26.424	31.317	28.681	30.585	70.213	73.892	74.768	75.676	77.115	77.530
	400	26.847	28.944	28.769	31.006	29.856	32.169	71.605	75.123	76.290	75.528	78.770	80.553

Table C-5: *QCI comparison between the m-NOSC, L-NOSC, k-means and Ward's algorithms for the 5HT1A and MMP1 datasets.*

Descriptor	k	Renin						SubP					
		L-NOSC / mNOSC				k-means	Ward's	L-NOSC / mNOSC				k-means	Ward's
		γ = 25	γ = 50	γ = 75	γ = 100			γ = 25	γ = 50	γ = 75	γ = 100		
BCI	100	59.209	60.333	58.126	57.343	54.233	56.649	55.731	56.201	56.893	56.059	57.150	58.185
	200	59.298	58.021	56.773	59.163	54.983	58.769	60.990	60.480	60.641	59.485	60.511	63.399
	300	58.693	61.992	63.434	63.788	63.063	63.094	61.226	62.071	62.972	62.382	63.101	64.773
	400	59.534	64.143	63.176	64.322	61.444	63.328	62.237	62.693	63.620	62.492	63.210	65.156
Daylight	100	56.142	56.452	57.362	54.764	58.132	55.972	50.854	54.808	54.233	59.192	56.657	55.211
	200	58.403	60.034	63.003	60.829	60.230	62.882	55.810	59.510	57.025	57.863	59.969	61.659
	300	62.934	59.704	63.872	62.341	62.777	65.675	57.525	59.846	59.987	60.028	61.613	62.466
	400	62.061	61.626	63.284	66.029	63.692	66.242	59.245	61.592	61.865	61.462	63.286	63.436
ECFP_4	100	69.276	79.678	77.759	78.389	58.079	61.051	48.592	48.209	50.528	55.352	58.496	58.020
	200	77.922	80.757	83.119	80.176	65.345	64.094	54.853	51.335	51.617	55.165	60.982	63.083
	300	76.927	80.459	84.011	84.622	64.826	65.121	56.479	53.478	54.372	56.656	63.326	64.918
	400	77.469	81.682	85.621	84.351	66.260	68.629	58.035	55.611	55.544	55.435	64.699	65.275
MDL Public Keys	100	56.725	60.994	68.843	61.838	60.203	57.628	46.660	45.484	49.435	48.198	54.091	56.497
	200	63.889	68.929	69.630	67.773	61.664	62.575	47.457	51.072	48.628	50.795	57.551	59.851
	300	67.187	72.587	72.920	70.688	61.280	64.490	50.638	52.561	50.519	53.378	58.651	61.613
	400	68.245	72.717	72.478	71.283	64.875	68.427	50.676	53.185	51.433	55.712	61.973	62.778
Unity	100	47.010	56.818	53.001	56.106	56.317	59.243	52.865	54.309	52.271	50.269	56.463	56.324
	200	59.633	65.565	61.045	61.938	60.120	64.224	57.367	57.198	57.928	58.650	58.703	59.945
	300	58.651	63.230	64.077	61.530	65.110	64.728	59.914	58.160	60.988	59.879	61.969	61.928
	400	55.353	58.781	62.776	61.790	66.039	64.972	59.423	59.675	61.951	61.329	62.073	63.100

Table C-6: *QCI comparison between the m-NOSC, L-NOSC, k-means and Ward's algorithms for the Renin and SubP datasets.*

Appendix D

The Identification of Eigenpairs using Singular Value Decomposition

This worked example, adapted from Khademhosseini (2002), highlights how a simple SVD can be used to solve an eigenproblem, through the identification of two sets of eigenvectors U and V that share a common set of eigenvalues. In this case an asymmetric matrix example has been selected to exhibit the algorithm's possible application in both symmetric and asymmetric problems. Let us begin by considering the SVD equation:

$$A_{mn} = U_{mm} S_{mn} V^{T}_{nn}$$

Where,

m and n are integers

A is a general matrix of size m x n

U is an m x m unitary matrix of the left singular vectors

S is a diagonal matrix of size m x n containing the singular values

V^T is the conjugate transpose of the n x n matrix V containing the right singular values of A.

One can solve for U by recalling that the relationship between SVD and eigendecompositions is such that U is equal to the eigenvectors of the matrix AA^T.

Thus, for an input matrix:

$$A = \begin{bmatrix} 3 & 1 & 1 \\ -1 & 3 & 1 \end{bmatrix}$$

One can identify the matrix U using the following process:

Recall,

U = eigenvectors of the matrix AA^T

Therefore, if,

$$A = \begin{bmatrix} 3 & 1 & 1 \\ -1 & 3 & 1 \end{bmatrix} \qquad A^T = \begin{bmatrix} 3 & -1 \\ 1 & 3 \\ 1 & 1 \end{bmatrix}$$

$\therefore$

$$AA^T = \begin{bmatrix} 3 & 1 & 1 \\ -1 & 3 & 1 \end{bmatrix} . \begin{bmatrix} 3 & -1 \\ 1 & 3 \\ 1 & 1 \end{bmatrix}$$

Multiplying the matrices gives:

$$AA^T = \begin{bmatrix} 11 & 1 \\ 1 & 11 \end{bmatrix}$$

If we now substitute matrix AA^T into the eigenvalue equation:

$$M\,v = \lambda\,v$$

M = a general matrix

v = an eigenvector

λ = an eigenvalue.

We get:

$$AA^T\,v = \lambda\,v$$

$$\begin{bmatrix} 11 & 1 \\ 1 & 11 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

This can also be written as a set of simultaneous equations:

$$11\,x_1 + x_2 = \lambda\,x_1$$

$$x_1 + 11\,x_2 = \lambda\,x_2$$

Rearranging gives:

$$(11 - \lambda)\,x_1 + x_2 = 0$$

$$x_1 + (11 - \lambda)\, x_2 = 0$$

The next step is to form a coefficient matrix from the equations shown above, and to take its determinant.

$$\mathrm{Det}\left(\begin{vmatrix}(11 - \lambda) & 1\\ 1 & (11 - \lambda)\end{vmatrix}\right) = ((11 - \lambda)\,(11 - \lambda)) - (1 \times 1)$$

One can now solve to identify the eigenvalues, λ, by setting the determinant to equal 0.

$$\big((11 - \lambda)\,(11 - \lambda)\big) - (1 \times 1) = 0$$

Multiplying out the brackets gives:

$$121 - 22\lambda + \lambda^2 - 1 = 0$$

$$120 - 22\lambda + \lambda^2 = 0$$

Re-factorising this equation gives:

$$(\lambda - 12)(\lambda - 10) = 0$$

$$\therefore \qquad \lambda = 10 \text{ or } 12$$

One can now identify the eigenvectors that comprise U by substituting λ = 10 and λ = 12 into the co-efficient equations as follows:

When λ = 10 we get

$$(11 - 10)\, x_1 + x_2 = 0$$

$$x_1 = -x_1$$

Therefore $x_1 = 1$ and $x_2 = -1$ for simplicity.

For λ = 12 we get

$$(11 - 12)\, x_1 + x_2 = 0$$

$$x_1 = x_2$$

∴ x_1= 1 and x_2 = 1

This gives us the matrix:

$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Finally we have to orthogonalise the matrix using the Gram Schmidt orthonormalisation process. The first orthonormalised vector of U which we shall call u', is be calculated from the unorthogonalised vector u_1 as follows:

$$\mathrm{u}' = \frac{\mathrm{u}_1}{|\mathrm{u}_1|} = \frac{[1\,,1]}{\sqrt{1^2 + 1^2}} = \left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right]$$

The second orthonormal vector of U, u'', is identified removing a multiple of the u' from the vector and normalising. The following step is used to orthogonalise the vector:

$$u_2^{int} = u_2 - u' \,.\, u_2 \times u'$$

$$u_2^{int} = [1\,,-1] - \left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right] .[1\,,-1] \times \left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right]$$

$$u_2^{int} = ([1\,,-1] - 0\,) \times \left[\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right]$$

Next one must normalise the vector:

$$u'' = \frac{u_2^{int}}{\left|u_2^{int}\right|} = \left[\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}\right]$$

Therefore,

$$U = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{bmatrix}$$

Where vector u^{int} vectors represent intermediate vector generated during the calculations.

Now that U has been identified we can solve for V using the same procedure by recalling that V contains the eigenvectors of the matrix A^TA.

$$A^TA = \begin{bmatrix} 3 & -1 \\ 1 & 3 \\ 1 & 1 \end{bmatrix} . \begin{bmatrix} 3 & 1 & 1 \\ -1 & 3 & 1 \end{bmatrix}$$

Therefore:

$$A^TA = \begin{bmatrix} 10 & 0 & 2 \\ 0 & 10 & 4 \\ 2 & 4 & 2 \end{bmatrix}$$

The eigenvalues of A^TA can be found using the equation:

$$\begin{bmatrix} 10 & 0 & 2 \\ 0 & 10 & 4 \\ 2 & 4 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \lambda \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

This can also be written as a system of three simultaneous equations:

$$10\,x_1 + 2\,x_3 = \lambda\,x_1$$

$$10\,x_2 + 4\,x_3 = \lambda\,x_2$$

$$2\,x_1 + 4\,x_2 + 2\,x_3 = \lambda\,x_3$$

These equations can be written as:

$$(10 - \lambda)\,x_1 + 2\,x_3 = 0$$

$$(10 - \lambda)\,x_2 + 4\,x_3 = 0$$

$$2\,x_1 + 4\,x_2 + (2 - \lambda)\,x_3 = 0$$

To solve the equations given above, we reform the coefficient matrix and take the determinant:

$$\begin{vmatrix} (10-\lambda) & 0 & 2 \\ 0 & (10-\lambda) & 4 \\ 2 & 4 & (2-\lambda) \end{vmatrix} = 0$$

This can be factorised down to:

$$(10-\lambda)\begin{vmatrix} (10-\lambda) & 4 \\ 4 & (2-\lambda) \end{vmatrix} + 2\begin{vmatrix} 0 & (10-\lambda) \\ 2 & 4 \end{vmatrix}$$

Multiplying this out, gives us:

$$(10-\lambda)[(10-\lambda)(2-\lambda)-16] + 2[0-(20-2\lambda)] = 0$$

$$\lambda\,(\lambda-10)(\lambda-12) = 0$$

Therefore the eigenvalues of the matrix A^TA equals λ = 0, λ = 10 and λ = 12.

To identify the eigenvector associated to the eigenvalue, λ = 12, we substitute this value into the equation below:

$$(10-\lambda)\,x_1 + 2\,x_3 = 0$$

This gives:

$$(10-12)\,x_1 + 2\,x_3 = 0$$

$$-2\,x_1 + 2\,x_3 = 0$$

∴ x_1= 1 and x_3 = 1.

The value of x_2 is found by substituting λ = 12, x_1= 1 and x_3 = 1 into the equation below:

$$(10-\lambda)\,x_2 + 4\,x_3 = 0$$

To give:

$$(10-12)\,x_2 + 4 = 0$$

$$\therefore \quad -2x_2 = -4$$

$$\therefore \quad x_2 = 2$$

Thus, the eigenvector associated to λ = 12 is:

$$\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

When λ = 10, the eigenvector is:

$$\begin{bmatrix} 2 \\ -1 \\ 0 \end{bmatrix}$$

For λ = 0, the associated eigenvector equals:

$$\begin{bmatrix} 1 \\ 2 \\ -5 \end{bmatrix}$$

We can now form the matrix V, by stacking the eigenvectors, such that:

$$V = \begin{bmatrix} 1 & 2 & 1 \\ 2 & -1 & 2 \\ 1 & 0 & -5 \end{bmatrix}$$

The matrix V was orthogonalised using the Gram-Schmidt orthonormalisation process, as outlined for the eigenvectors of matrix AA^T earlier.

$$u' = \frac{u_1}{|u_1|} = \left[\frac{1}{\sqrt{6}}, \frac{2}{\sqrt{6}}, \frac{1}{\sqrt{6}}\right]$$

$$u_2^{int} = u_2 - u' \,.\, u_2 \times u' = [2, -1, 0]$$

$$u'' = \frac{u_2^{int}}{|u_2^{int}|} = \left[\frac{2}{\sqrt{5}}, \frac{-1}{\sqrt{5}}, 0\right]$$

$$u_3^{int} = u_3 - u' \,.\, u_3 \times u' - u''.\, u_3 \times u'' = \left[\frac{-2}{3}, \frac{-4}{3}, \frac{10}{3}\right]$$

$$u''' = \frac{u_3^{int}}{\left|u_3^{int}\right|} = \left[\frac{1}{\sqrt{30}}, \frac{2}{\sqrt{30}}, \frac{-5}{\sqrt{30}}\right]$$

Thus the orthogonal matrix of eigenvectors is:

$$V = \begin{bmatrix} \frac{1}{\sqrt{6}} & \frac{2}{\sqrt{5}} & \frac{1}{\sqrt{30}} \\ \frac{2}{\sqrt{6}} & \frac{-1}{\sqrt{5}} & \frac{2}{\sqrt{30}} \\ \frac{1}{\sqrt{6}} & 0 & \frac{-5}{\sqrt{30}} \end{bmatrix}$$

Therefore:

$$V^T = \begin{bmatrix} \frac{1}{\sqrt{6}} & \frac{2}{\sqrt{6}} & \frac{1}{\sqrt{6}} \\ \frac{2}{\sqrt{5}} & \frac{-1}{\sqrt{5}} & 0 \\ \frac{1}{\sqrt{30}} & \frac{2}{\sqrt{30}} & \frac{-5}{\sqrt{30}} \end{bmatrix}$$

Where each column is an eigenvector.

Appendix E

Ideal Clusters from DC100 Fragment Class

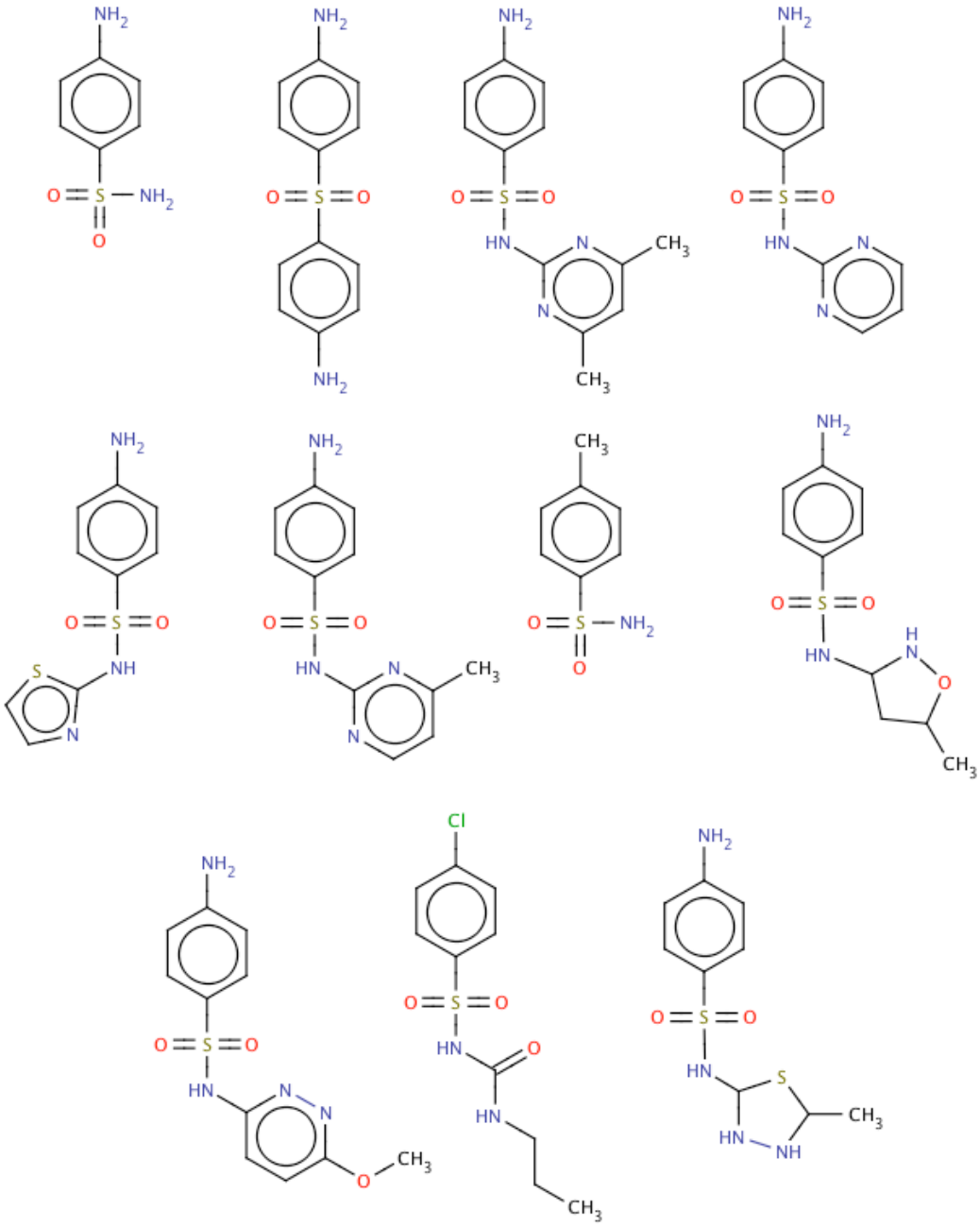


Figure E-1: *Cluster 1 from DC100 set.*

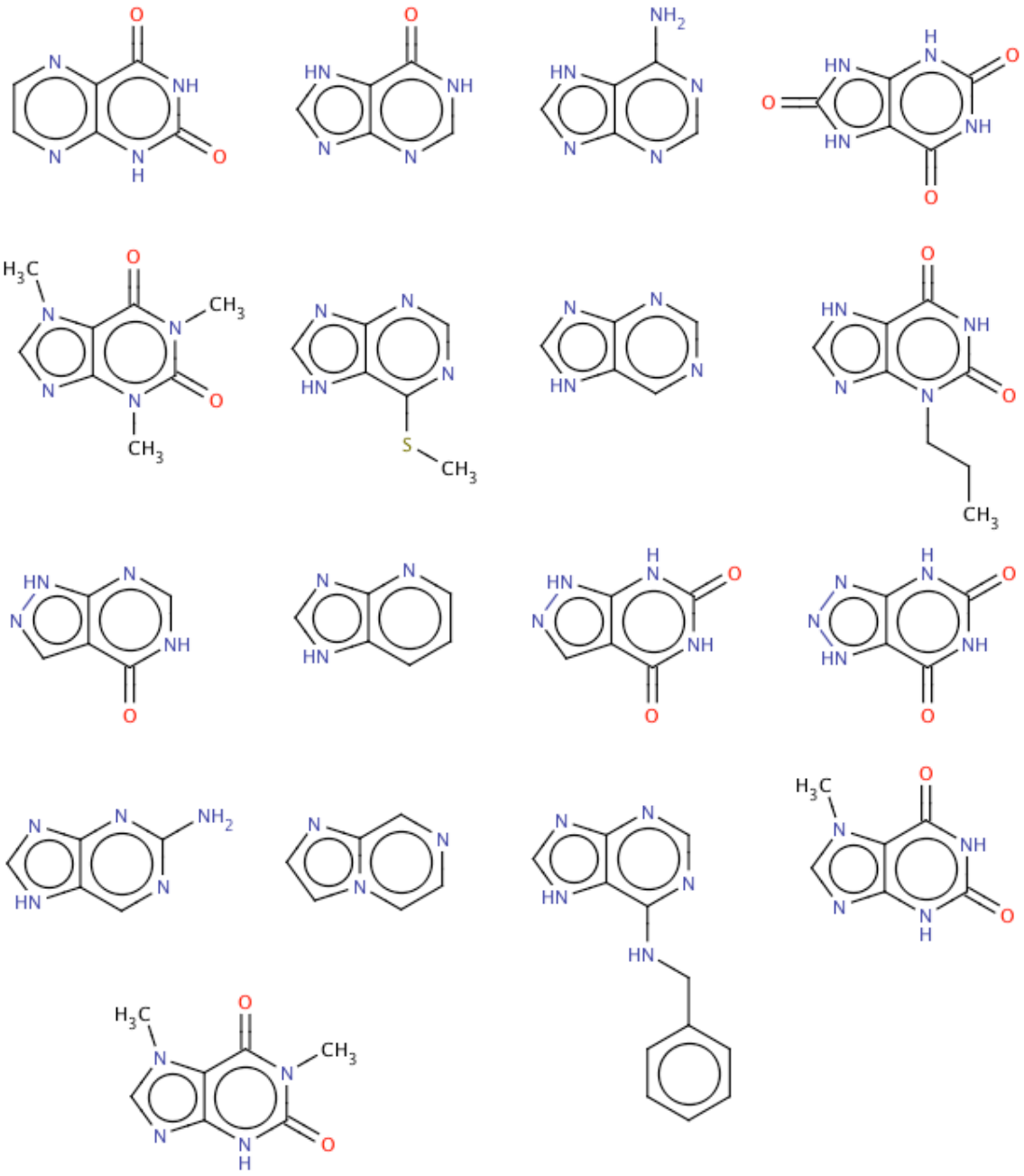


***Figure E-2**: Cluster 2 from DC100 fragment set.*

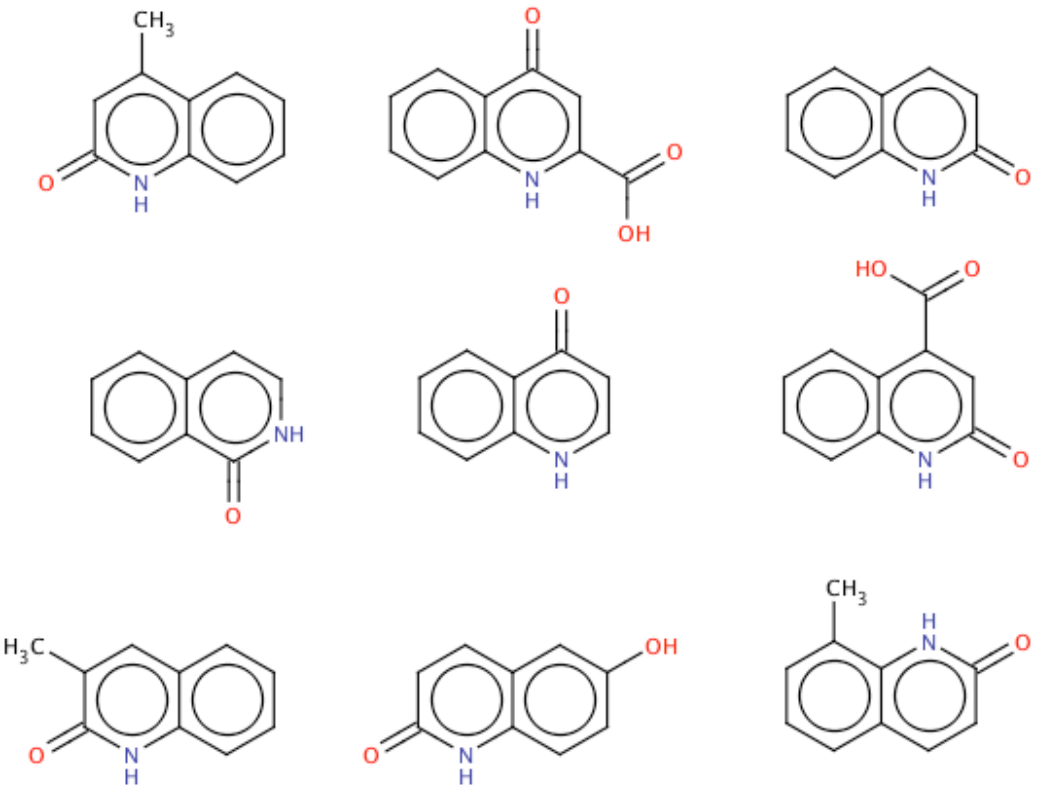


Figure E-3: *Cluster 3 from DC100 fragment class.*

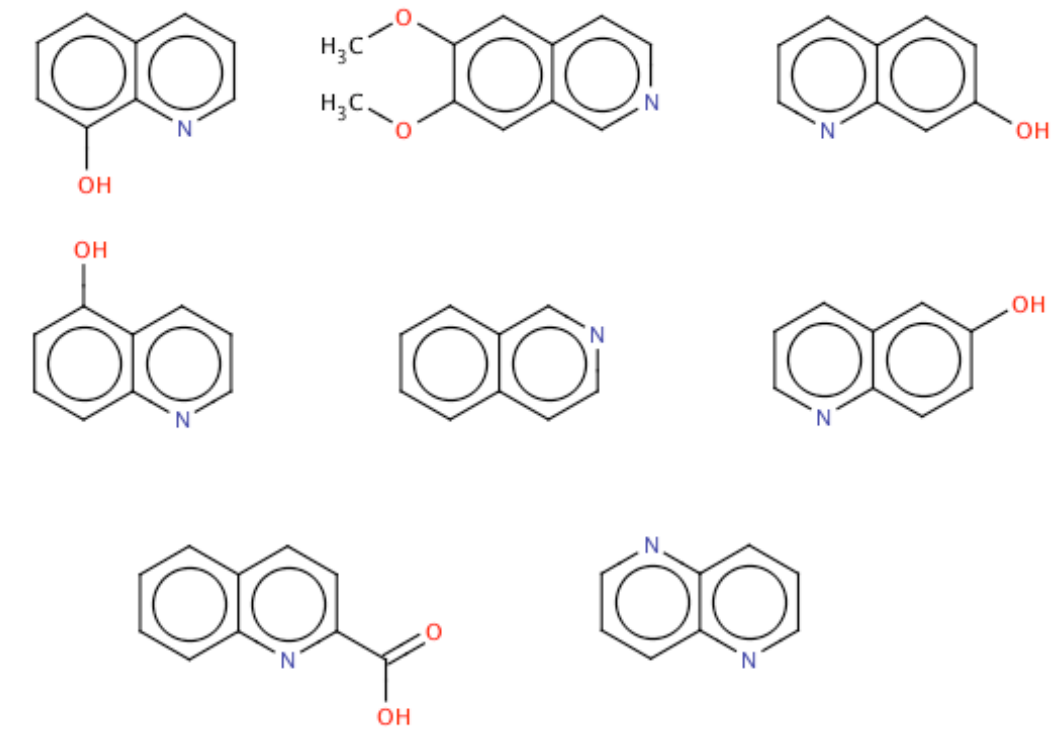


Figure E-4: *Cluster 4 from DC100 fragment class.*

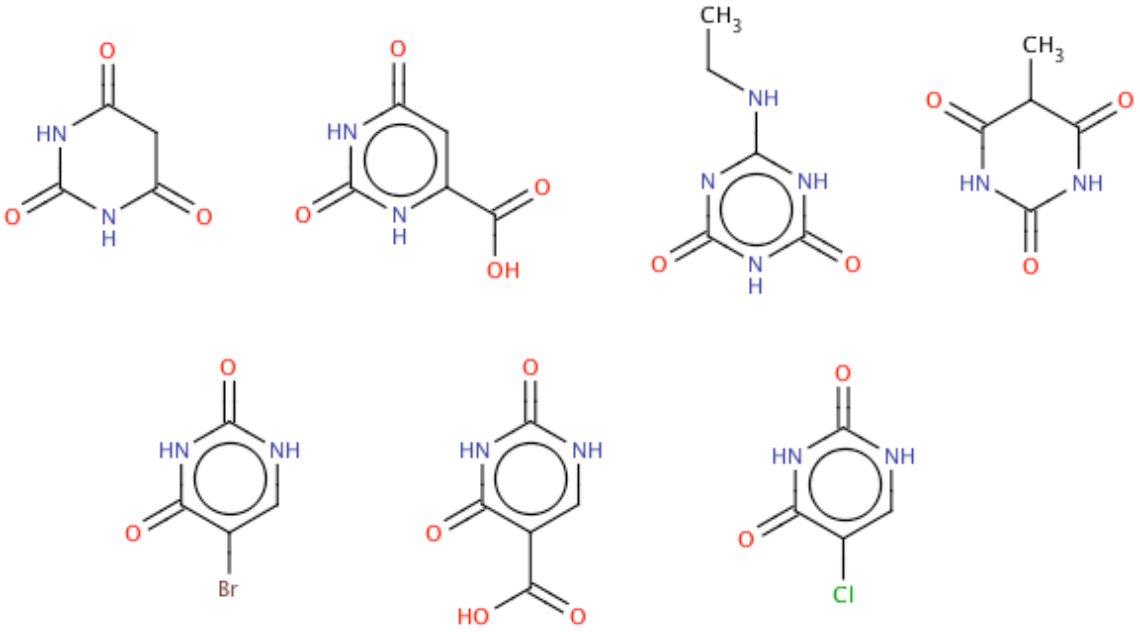


Figure E-5: *Cluster 5 from the DC100 fragment class.*

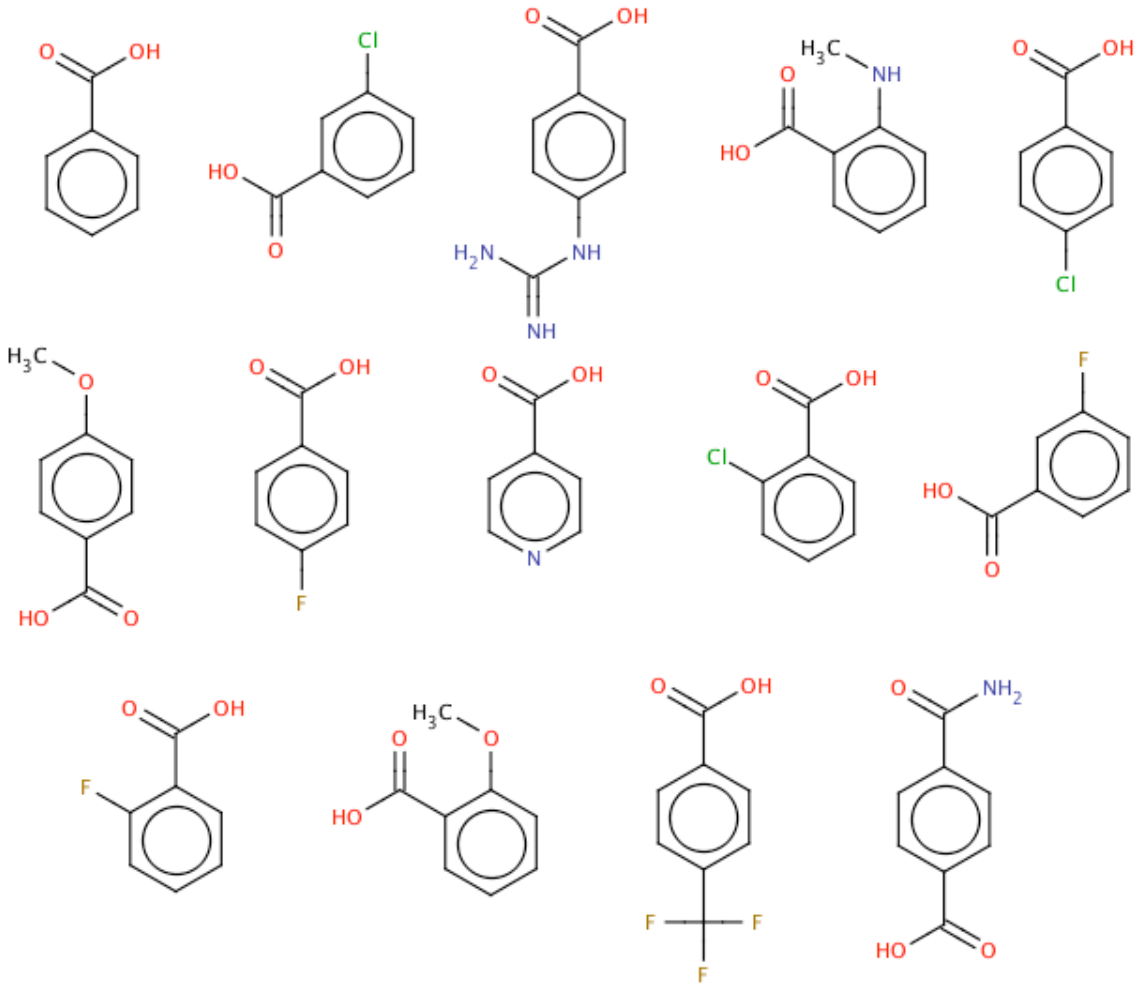


Figure E-6: *Cluster 6 from the DC100 dataset.*

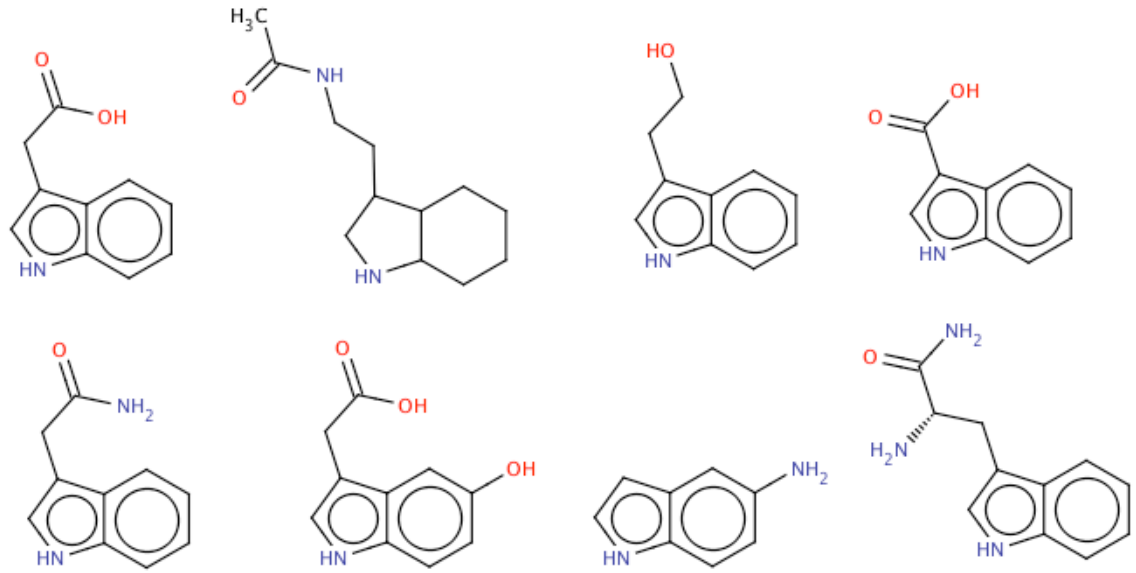


Figure E-7: *Cluster 7 from the DC100 dataset.*

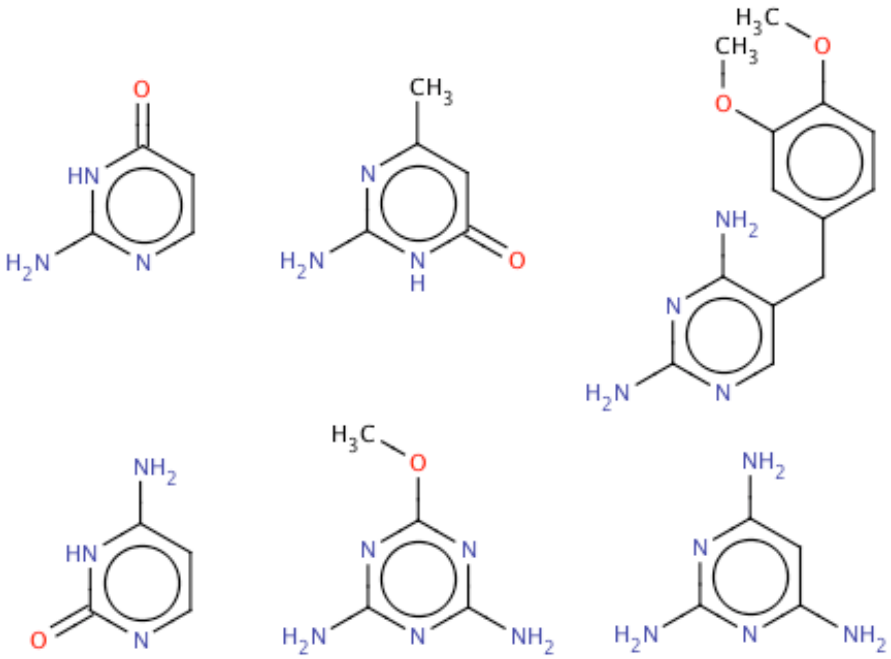


Figure E-8: *Cluster 8 from DC100 fragment class.*

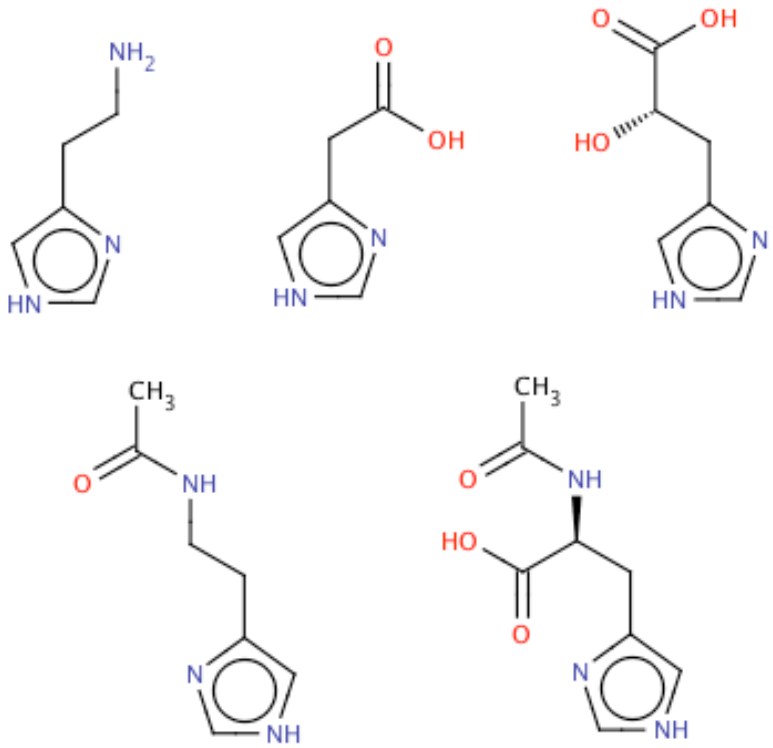


Figure E-9: *Cluster 9 from the DC100 fragment class.*

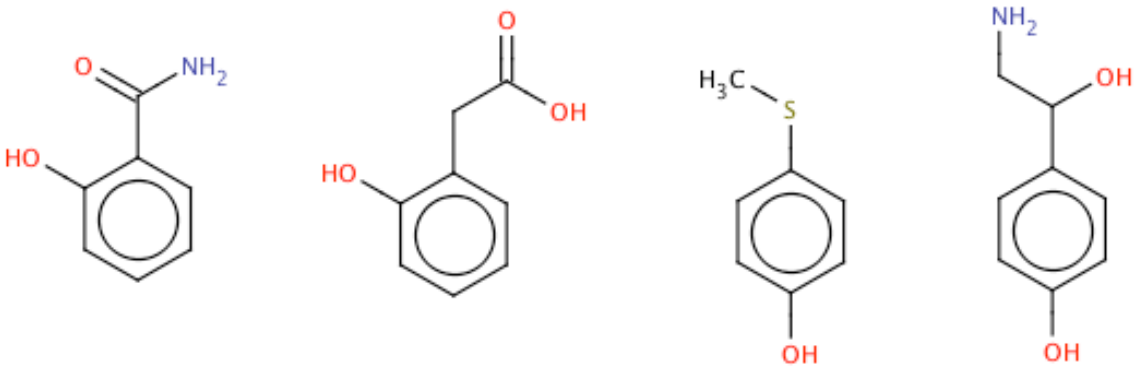


Figure E-10: *Cluster 10 from the DC100 dataset.*

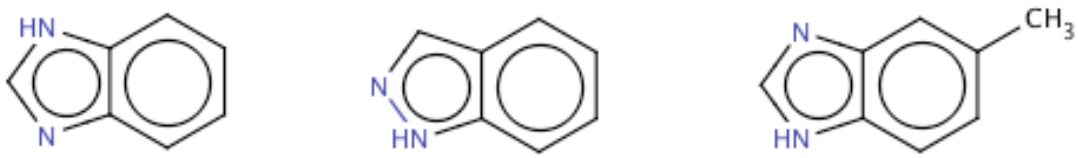


Figure E-11: *Cluster 11 from the DC100 fragment class.*

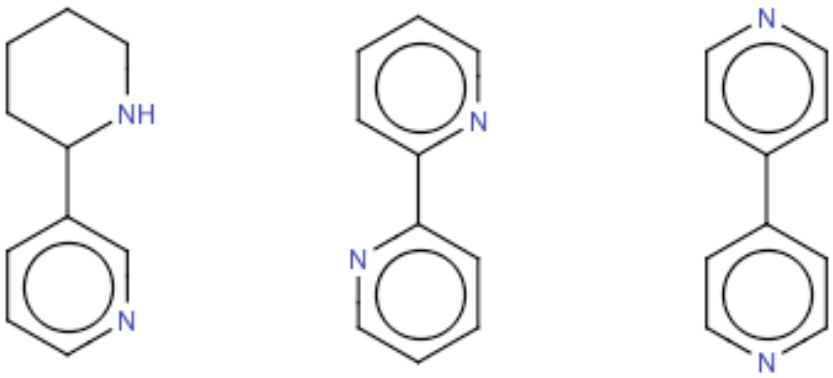


Figure E-12: *Cluster 12 from the DC100 fragment class.*

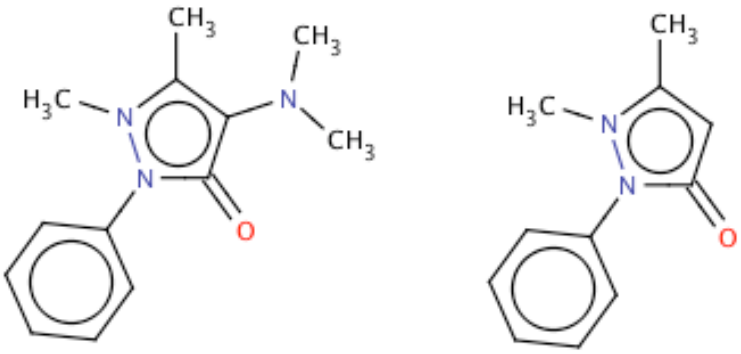


Figure E-13: *Cluster 13 from the DC100 dataset.*

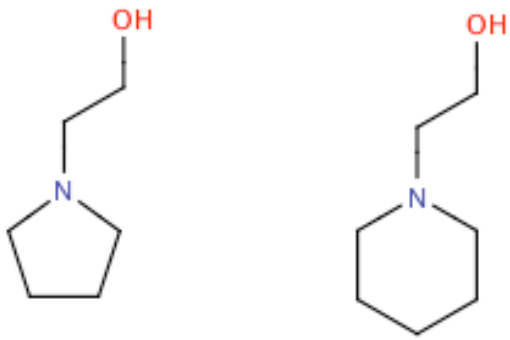


Figure E-14: *Cluster 14 from the DC100 dataset.*

Appendix F

Chapter 7 Extended Results

In this section the extended results for the clustering of the DC100 fragment class are shown. In ***Table F.1*** the comparison between each SVD-NOSC implementation and the Ideal set of clusters, when described by RDKit Circular fingerprints are shown. Similarly, ***Table F.2*** shows the Jaccard comparison between the SVD-NOSC and k-means methods when using RDKit Circular fingerprints. ***Tables F.3*** and ***F.4*** then show the results of an identical study using RDKit Linear fingerprints.

Clustering 1	Clustering 2				Jaccard		Clustering 1	Clustering 2				Jaccard	
	alpha	beta	ST	EVT	u	v		alpha	beta	ST	EVT	u	v
ideal	1	1	0.01	0.01	0.604	0.604	ideal	0.7	0.3	0.01	0.01	0.627	0.627
			0.01	0.001	0.604	0.604				0.01	0.001	0.627	0.627
			0.01	0.0001	0.604	0.604				0.01	0.0001	0.627	0.627
			0.001	0.01	0.604	0.604				0.001	0.01	0.627	0.627
			0.001	0.001	0.604	0.604				0.001	0.001	0.627	0.627
			0.001	0.0001	0.604	0.604				0.001	0.0001	0.627	0.627
			0.0001	0.01	0.604	0.604				0.0001	0.01	0.627	0.627
			0.0001	0.001	0.604	0.604				0.0001	0.001	0.627	0.627
			0.0001	0.0001	0.604	0.604				0.0001	0.0001	0.627	0.627
	0.9	0.1	0.01	0.01	0.627	0.627		0.6	0.4	0.01	0.01	0.627	0.612
			0.01	0.001	0.627	0.627				0.01	0.001	0.627	0.612
			0.01	0.0001	0.627	0.627				0.01	0.0001	0.627	0.612
			0.001	0.01	0.655	0.627				0.001	0.01	0.627	0.612
			0.001	0.001	0.655	0.627				0.001	0.001	0.627	0.612
			0.001	0.0001	0.655	0.627				0.001	0.0001	0.627	0.612
			0.0001	0.01	0.655	0.627				0.0001	0.01	0.627	0.612
			0.0001	0.001	0.655	0.627				0.0001	0.001	0.627	0.612
			0.0001	0.0001	0.655	0.627				0.0001	0.0001	0.627	0.612
	0.8	0.2	0.01	0.01	0.627	0.627		0.5	0.5	0.01	0.01	0.645	0.645
			0.01	0.001	0.627	0.627				0.01	0.001	0.645	0.645
			0.01	0.0001	0.627	0.627				0.01	0.0001	0.645	0.645
			0.001	0.01	0.627	0.627				0.001	0.01	0.645	0.645
			0.001	0.001	0.627	0.627				0.001	0.001	0.645	0.645
			0.001	0.0001	0.627	0.627				0.001	0.0001	0.645	0.645
			0.0001	0.01	0.627	0.627				0.0001	0.01	0.645	0.645
			0.0001	0.001	0.627	0.627				0.0001	0.001	0.645	0.645
			0.0001	0.0001	0.627	0.627				0.0001	0.0001	0.645	0.645

Table F-1: *Jaccard comparison between the Ideal clustering of the DC100 fragment class and the clusters produced when using the SVD-NOSC method, when using RDKit Circular fingerprints.*

Clustering 1	Clustering 2				Jaccard		Clustering 1	Clustering 2				Jaccard	
	alpha	beta	ST	EVT	u	v		alpha	beta	ST	EVT	u	v
k-means	1	1	0.01	0.01	0.630	0.630	k-means	0.7	0.3	0.01	0.01	0.601	0.601
			0.01	0.001	0.630	0.630				0.01	0.001	0.601	0.601
			0.01	0.0001	0.630	0.630				0.01	0.0001	0.601	0.601
			0.001	0.01	0.630	0.630				0.001	0.01	0.601	0.601
			0.001	0.001	0.630	0.630				0.001	0.001	0.601	0.601
			0.001	0.0001	0.630	0.630				0.001	0.0001	0.601	0.601
			0.0001	0.01	0.630	0.630				0.0001	0.01	0.601	0.601
			0.0001	0.001	0.630	0.630				0.0001	0.001	0.601	0.601
			0.0001	0.0001	0.630	0.630				0.0001	0.0001	0.601	0.601
	0.9	0.1	0.01	0.01	0.575	0.627		0.6	0.4	0.01	0.01	0.627	0.586
			0.01	0.001	0.575	0.627				0.01	0.001	0.627	0.586
			0.01	0.0001	0.575	0.627				0.01	0.0001	0.627	0.586
			0.001	0.01	0.575	0.627				0.001	0.01	0.627	0.586
			0.001	0.001	0.575	0.627				0.001	0.001	0.627	0.586
			0.001	0.0001	0.575	0.627				0.001	0.0001	0.627	0.586
			0.0001	0.01	0.575	0.627				0.0001	0.01	0.627	0.586
			0.0001	0.001	0.575	0.627				0.0001	0.001	0.627	0.586
			0.0001	0.0001	0.575	0.627				0.0001	0.0001	0.627	0.586
	0.8	0.2	0.01	0.01	0.601	0.627		0.5	0.5	0.01	0.01	0.645	0.645
			0.01	0.001	0.601	0.627				0.01	0.001	0.645	0.645
			0.01	0.0001	0.601	0.627				0.01	0.0001	0.645	0.645
			0.001	0.01	0.601	0.627				0.001	0.01	0.645	0.645
			0.001	0.001	0.601	0.627				0.001	0.001	0.645	0.645
			0.001	0.0001	0.601	0.627				0.001	0.0001	0.645	0.645
			0.0001	0.01	0.601	0.627				0.0001	0.01	0.645	0.645
			0.0001	0.001	0.601	0.627				0.0001	0.001	0.645	0.645
			0.0001	0.0001	0.601	0.627				0.0001	0.0001	0.645	0.645

Table F-2: *Jaccard comparison between the k-means clustering of the DC100 fragment class and the clusters produced when using the SVD-NOSC method, when using RDKit Circular fingerprints.*

Clustering 1	Clustering 2				Jaccard		Clustering 1	Clustering 2				Jaccard	
	alpha	beta	ST	EVT	u	v		alpha	beta	ST	EVT	u	v
ideal	1	1	0.01	0.01	0.655	0.655	ideal	0.7	0.3	0.01	0.01	0.627	0.627
			0.01	0.001	0.604	0.604				0.01	0.001	0.627	0.627
			0.01	0.0001	0.604	0.604				0.01	0.0001	0.627	0.627
			0.001	0.01	0.604	0.604				0.001	0.01	0.627	0.627
			0.001	0.001	0.604	0.604				0.001	0.001	0.627	0.627
			0.001	0.0001	0.604	0.604				0.001	0.0001	0.627	0.627
			0.0001	0.01	0.604	0.604				0.0001	0.01	0.627	0.627
			0.0001	0.001	0.604	0.604				0.0001	0.001	0.627	0.627
			0.0001	0.0001	0.604	0.604				0.0001	0.0001	0.627	0.627
	0.9	0.1	0.01	0.01	0.575	0.586		0.6	0.4	0.01	0.01	0.627	0.612
			0.01	0.001	0.575	0.586				0.01	0.001	0.627	0.612
			0.01	0.0001	0.575	0.586				0.01	0.0001	0.627	0.612
			0.001	0.01	0.655	0.627				0.001	0.01	0.627	0.612
			0.001	0.001	0.655	0.627				0.001	0.001	0.627	0.612
			0.001	0.0001	0.655	0.627				0.001	0.0001	0.627	0.612
			0.0001	0.01	0.655	0.627				0.0001	0.01	0.627	0.612
			0.0001	0.001	0.655	0.627				0.0001	0.001	0.627	0.612
			0.0001	0.0001	0.655	0.627				0.0001	0.0001	0.627	0.612
	0.8	0.2	0.01	0.01	0.627	0.627		0.5	0.5	0.01	0.01	0.645	0.645
			0.01	0.001	0.627	0.627				0.01	0.001	0.645	0.645
			0.01	0.0001	0.627	0.627				0.01	0.0001	0.645	0.645
			0.001	0.01	0.627	0.627				0.001	0.01	0.645	0.645
			0.001	0.001	0.627	0.627				0.001	0.001	0.645	0.645
			0.001	0.0001	0.627	0.627				0.001	0.0001	0.645	0.645
			0.0001	0.01	0.627	0.627				0.0001	0.01	0.645	0.645
			0.0001	0.001	0.627	0.627				0.0001	0.001	0.645	0.645
			0.0001	0.0001	0.627	0.627				0.0001	0.0001	0.645	0.645

Table F-3: *Jaccard comparison between the Ideal clustering of the DC100 fragment class and the clusters produced when using the SVD-NOSC method, when using RDKit Linear fingerprints.*

Clustering 1	Clustering 2				Jaccard		Clustering 1	Clustering 2				Jaccard	
	alpha	beta	ST	EVT	u	v		alpha	beta	ST	EVT	u	v
ideal	1	1	0.01	0.01	0.655	0.655	ideal	0.7	0.3	0.01	0.01	0.655	0.601
			0.01	0.001	0.655	0.655				0.01	0.001	0.655	0.601
			0.01	0.0001	0.655	0.655				0.01	0.0001	0.655	0.601
			0.001	0.01	0.639	0.639				0.001	0.01	0.655	0.601
			0.001	0.001	0.639	0.639				0.001	0.001	0.655	0.601
			0.001	0.0001	0.639	0.639				0.001	0.0001	0.655	0.601
			0.0001	0.01	0.639	0.639				0.0001	0.01	0.655	0.601
			0.0001	0.001	0.639	0.639				0.0001	0.001	0.655	0.601
			0.0001	0.0001	0.639	0.639				0.0001	0.0001	0.655	0.601
	0.9	0.1	0.01	0.01	0.575	0.612		0.6	0.4	0.01	0.01	0.627	0.586
			0.01	0.001	0.575	0.612				0.01	0.001	0.627	0.586
			0.01	0.0001	0.575	0.612				0.01	0.0001	0.627	0.586
			0.001	0.01	0.575	0.612				0.001	0.01	0.627	0.586
			0.001	0.001	0.575	0.612				0.001	0.001	0.627	0.586
			0.001	0.0001	0.575	0.612				0.001	0.0001	0.627	0.586
			0.0001	0.01	0.575	0.612				0.0001	0.01	0.627	0.586
			0.0001	0.001	0.575	0.612				0.0001	0.001	0.627	0.586
			0.0001	0.0001	0.575	0.612				0.0001	0.0001	0.627	0.586
	0.8	0.2	0.01	0.01	0.627	0.586		0.5	0.5	0.01	0.01	0.667	0.667
			0.01	0.001	0.627	0.586				0.01	0.001	0.667	0.667
			0.01	0.0001	0.627	0.586				0.01	0.0001	0.667	0.667
			0.001	0.01	0.627	0.586				0.001	0.01	0.667	0.667
			0.001	0.001	0.627	0.586				0.001	0.001	0.667	0.667
			0.001	0.0001	0.627	0.586				0.001	0.0001	0.667	0.667
			0.0001	0.01	0.627	0.586				0.0001	0.01	0.667	0.667
			0.0001	0.001	0.627	0.586				0.0001	0.001	0.667	0.667
			0.0001	0.0001	0.627	0.586				0.0001	0.0001	0.667	0.667

***Table F-4**: Jaccard comparison between the Ideal clustering of the DC100 fragment class and the clusters produced when using the SVD-NOSC method, when using RDKit Linear fingerprints.*

References

A-Razzak, M. & Glen, R.C. (1992). "Applications of rule-induction in the derivation of quantitative structure-activity relationships". *Journal of Computer-Aided Molecular Design*, **6** (4), 349-383.

Accelrys (2011). *Pipeline Pilot Version 8.5* [Online]. http://accelrys.com [Accessed 01/09/2013]

Atkins, P. & Paula, J.d. (2009). *Atkins' Physical Chemistry (9th edition).* Oxford: Oxford University Press.

Bajorath, J. (2001). "Selected concepts and investigations in compound classification, molecular descriptor analysis, and virtual screening". *Journal of Chemical Information and Computer Sciences*, **41** (2), 233-245.

Bajorath, J. (2002). "Integration of virtual and high-throughput screening". *Nature Reviews Drug Discovery*, **1** (11), 882-894.

Bajorath, J., Peltason, L., Wawer, M., Guha, R., Lajiness, M. & Van Drie, J. (2009). "Navigating structure–activity landscapes". *Drug Discovery Today*, **14** (13-14), 698-705.

Bender, A., Jenkins, J.L., Scheiber, J., Sukuru, S.C.K., Glick, M. & Davies, J.W. (2009). "How similar are similarity searching methods? A principal component analysis of molecular descriptor space". *Journal of Chemical Information and Modeling*, **49** (1), 108-119.

Bender, A., Mussa, H., Glen, R. & Reiling, S. (2004a). "Molecular similarity searching using atom environments, information-based feature selection, and a naive bayesian classifier". *Journal of Chemical Information and Computer Sciences*, **44** (1), 170-178.

Bender, A., Mussa, H., Glen, R. & Reiling, S. (2004b). "Similarity searching of chemical databases using atom environment descriptors (MOLPRINT 2D): evaluation of performance". *Journal of Chemical Information and Computer Sciences*, **44** (5), 1708-1718.

Berry, M., Do, T., O'Brien, G., Krishna, V. & Varadhan, S. (1993). *SVDPACKC user's guide*. CS-93-194, University of Tennessee. Report No.

Berry, M. & Sameh, A. (1989). "An overview of parallel algorithms for the singular value and symmetric eigenvalue problems". *Journal of Computational and Applied Mathematics*, **27** (1), 191-213.

Berry, M.J. & Linoff, G. (1997). *Data Mining Techniques: For Marketing, Sales and Customer Support.* John Wiley & Sons, Inc.

Berry, M.W., Mezher, D., Philippe, B. & Sameh, A. (2006). "Parallel algorithms for the singular value decomposition". *STATISTICS TEXTBOOKS AND MONOGRAPHS*, **184**, 117.

Biggs, N., Lloyd, E.K. & Wilson, R.J. (1986). *Graph theory, 1736-1936.* Oxford University Press, USA.

Bleicher, K.H., Böhm, H.-J., Müller, K. & Alanine, A.I. (2003). "Hit and lead generation: beyond high-throughput screening". *Nature Reviews Drug Discovery*, **2** (5), 369-378.

Blyth, T. & Robertson, E. (2002). *Basic linear algebra.* Springer Verlag.

Böhm, H., Flohr, A. & Stahl, M. (2004). "Scaffold hopping". *Drug Discovery Today: Technologies*, **1** (3), 217-224.

Brewer, M.L. (2007). "Development of a spectral clustering method for the analysis of molecular data sets". *Journal of Chemical Information and Modelling*, **47** (5), 1727-1733.

Brown, R.D. & Martin, Y.C. (1996). "Use of structure activity data to compare structure-based clustering methods and descriptors for use in compound selection". *Journal of Chemical Information and Computer Sciences*, **36** (3), 572-584.

Brown, R.D. & Martin, Y.C. (1997). "The information content of 2D and 3D structural descriptors relevant to ligand-receptor binding". *Journal of Chemical Information and Computer Sciences*, **37** (1), 1-9.

Butina, D. (1999). "Unsupervised data base clustering based on daylight's fingerprint and Tanimoto similarity: A fast and automated way to cluster small and large data sets". *Journal of Chemical Information and Computer Sciences*, **39** (4), 747-750.

Calinski, T. & Harabasz, J. (1974). "A dendrite method for cluster analysis". *Communications in Statistics-Simulation and Computation*, **3** (1), 1-27.

Campello, R.J.G.B. & Hruschka, E.R. (2006). "A fuzzy extension of the silhouette width criterion for cluster analysis". *Fuzzy Sets and Systems*, **157** (21), 2858-2875.

CERN (2011). *COLT Matrix Package* [Online]. http://acs.lbl.gov/software/colt/

ChemAxon (2010). *Marvin Sketch Software Version 5.4* [Online]. http://www.chemaxon.com/products/marvin/marvinsketch/ [Accessed 16/12/2010]

Cheminformatics.org (2010). *Cyclooxygenase-2 Inhibitor Dataset* [Online]. http://www.cheminformatics.org/ [Accessed 05/10/09]

Chi, Z., Yan, H. & Ph m, T. (1996). *Fuzzy algorithms: with applications to image processing and pattern recognition.* World Scientific Pub Co Inc.

Chung, F. (1997). "Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92)". *American Mathematical Society*, **3**, 8.

Congreve, M., Chessari, G., Tisi, D. & Woodhead, A.J. (2008). "Recent developments in fragment-based drug discovery". *Journal of Medicinal Chemistry*, **51** (13), 3661-3680.

Cramer, R., Redl, G. & Berkoff, C. (1974). "Substructural analysis. A novel approach to the problem of drug design". *Journal of Medicinal Chemistry*, **17** (5), 533.

Davenport, A.J., Moeller, C., Heifetz, A., Mazanetz, M.P., Law, R.J., Ebneth, A. & Gemkow, M.J. (2010). "Using electrophysiology and in-silico three-dimensional modeling to reduce human ether-a-go-go related gene K+ channel inhibition in a histamine H3 receptor antagonist program". *Assay and Drug Development Technologies*, **8** (6), 781-789.

Dawes, B. & Abrahams, D. (2002). *The Boost C++ metaprogramming library* [Online]. http://www.boost.org

Daylight (2002). *Daylight Software* [Online]. http://www.daylight.com/

De Lathauwer, L., De Moor, B. & Vandewalle, J. (2000). "A multilinear singular value decomposition". *SIAM Journal on Matrix Analysis and Applications*, **21** (4), 1253-1278.

Diestel, R. (2000) Springer-Verlag, New York.

DigitalChemistry (2005). *BCI Software* [Online]. http://www.digitalchemistry.co.uk/

Downs, G., Willett, P. & Fisanick, W. (1994). "Similarity searching and clustering of chemical-structure databases using molecular property data". *Journal of Chemical Information and Computer Sciences*, **34** (5), 1094-1102.

Downs, G.M. & Barnard, J.M. (2002). "Clustering methods and their uses in computational chemistry". *Reviews in Computational Chemistry, Vol 18*, 1-40.

Drews, J. (2000). "Drug discovery: a historical perspective". *Science*, **287** (5460), 1960-1964.

Duff, I.S., Grimes, R.G. & Lewis, J.G. (1992). "Users' guide for the Harwell-Boeing sparse matrix collection (Release I)".

Euler, L. (1736). "Solutio problematis ad geometriam situs pertinentis". *Commentarii Academiae Scientiarum Imperialis Petropolitanae*, **8**, 128-140.

Farmer, P. & Ariens, E. (1982). "Speculations on the design of nonpeptidic peptidomimetics". *Trends in Pharmacological Sciences*, **3**, 362-365.

FDA (2013). *NME approvals 2012* [Online]. http://www.fda.gov/downloads/Drugs/DevelopmentApprovalProcess/DrugInnovation/UCM337830.pdf

Flower, D.R. (1998). "On the properties of bit string-based measures of chemical similarity". *Journal of Chemical Information and Computer Sciences*, **38** (3), 379-386.

Gasteiger, J. & Zupan, J. (1993). "Neural networks in chemistry". *Angewandte Chemie International Edition in English*, **32** (4), 503-527.

Geus, R. (2002). *The Jacobi-Davidson algorithm for solving large sparse symmetric eigenvalue problems with application to the design of accelerator cavities*. Citeseer.

Ginsburg, G.S. & McCarthy, J.J. (2001). "Personalized medicine: revolutionizing drug discovery and patient care". *TRENDS in Biotechnology*, **19** (12), 491-496.

Givens, W. (1954). *Numerical computation of the characteristic values of a real symmetric matrix*. Oak Ridge National Lab. Report No.

Goldman, B. & Walters, W. (2006). "Machine learning in computational chemistry". *Annual Reports in Computational Chemistry*, **2**, 127-40.

Golub, G.H. & Van Loan, C.F. (1996). *Matrix computations.* Johns Hopkins Univ Pr.

Grimes, R.D., Lewis, J.G. & Simon, H.D. (1988). "The implementation of a block Lanczos algorithm with reorthogonalization methods".

Guénoche, A., Hansen, P. & Jaumard, B. (1991). "Efficient algorithms for divisive hierarchical clustering with the diameter criterion". *Journal of Classification*, **8** (1), 5-30.

Gund, P. (1977). "Three-dimensional pharmacophoric pattern searching". *Progress in Molecular and Subcellular Biology*, **5**, 117-143.

Hall, L. & Kier, L. (1991). "The molecular connectivity chi indexes and kappa shape indexes in structure-property modeling. Antiviral activity of benzimidazoles against flu virus". *VCH PUBLISHERS, NEW YORK, NY(USA).* **2**, 367-422.

Hall, L. & Kier, L. (2001). "Issues in representation of molecular structure:: The development of molecular connectivity". *Journal of Molecular Graphics and Modelling*, **20** (1), 4-18.

Hann, M.M., Leach, A.R. & Harper, G. (2001). "Molecular complexity and its impact on the probability of finding leads for drug discovery". *Journal of Chemical Information and Computer Sciences*, **41** (3), 856-864.

Haranczyk, M. & Holliday, J. (2008). "Comparison of similarity coefficients for clustering and compound selection". *Journal of Chemical Information and Modeling*, **48** (3), 498-508.

Harper, G. & Pickett, S.D. (2006). "Methods for mining HTS data". *Drug Discovery Today*, **11** (15-16), 694-699.

Hassan, M., Brown, R.D., Varma-O'Brien, S. & Rogers, D. (2006). "Cheminformatics analysis and learning in a data pipelining environment". *Molecular Diversity*, **10** (3), 283-299.

Heifetz, A., Barker, O., Verquin, G., Wimmer, N., Meutermans, W., Pal, S., Law, R.J. & Whittaker, M. (2013). "Fighting obesity with a sugar-based library: discovery of novel MCH-1R antagonists by a new computational–VAST approach for exploration of GPCR binding sites". *Journal of Chemical Information and Modeling*.

Hert, J., Willett, P., Wilton, D., Acklin, P., Azzaoui, K., Jacoby, E. & Schuffenhauer, A. (2004). "Comparison of topological descriptors for similarity-based virtual screening using multiple bioactive reference structures". *Organic & Biomolecular Chemistry*, **2** (22), 3256-3266.

Hert, J., Willett, P., Wilton, D., Acklin, P., Azzaoui, K., Jacoby, E. & Schuffenhauer, A. (2005). "Enhancing the effectiveness of similarity-based virtual screening using nearest-neighbor information". *Journal of Medicinal Chemistry*, **48** (22), 7049-7054.

Hert, J., Willett, P., Wilton, D., Acklin, P., Azzaoui, K., Jacoby, E. & Schuffenhauer, A. (2006). "New methods for ligand-based virtual screening: use of data fusion and machine learning to enhance the effectiveness of similarity searching". *Journal of Chemical Information and Modeling*, **46** (2), 462.

Hochstenbach, M. & Notay, Y. (2006). "The Jacobi-Davidson method". *GAMM Mitt*, **29**, 368-382.

Holliday, J., Rodgers, S., Willett, P., Chen, M., Mahfouf, M., Lawson, K. & Mullier, G. (2004). "Clustering files of chemical structures using the fuzzy k-means clustering method". *Journal of Chemical Information and Computer Sciences*, **44** (3), 894-902.

Householder, A.S. (1958). "A class of methods for inverting matrices". *Journal of the Society for Industrial and Applied Mathematics*, 189-195.

Hu, Y., Lounkine, E. & Bajorath, J. (2009). "Filtering and Counting of Extended Connectivity Fingerprint Features Maximizes Compound Recall and the Structural Diversity of Hits". *Chemical Biology & Drug Design*, **74** (1), 92-98.

Hubert, L. & Levin, J. (1976). "A general statistical framework for assessing categorical clustering in free recall". *Psychological Bulletin*, **83** (6), 1072-1080.

Ivanenkov, Y., Savchuk, N., Ekins, S. & Balakin, K. (2009). "Computational mapping tools for drug discovery". *Drug Discovery Today*, **14** (15-16), 767-775.

Jarvis, R. & Patrick, E. (1973). "Clustering using a similarity measure based on shared near neighbors". *IEEE Transactions on Computers*, **22** (11), 1025-1034.

Jencks, W.P. (1981). "On the attribution and additivity of binding energies". *Proceedings of the National Academy of Sciences*, **78** (7), 4046-4050.

Jia, Z. & Niu, D. (2003). "An implicitly restarted refined bidiagonalization Lanczos method for computing a partial singular value decomposition". *SIAM Journal on Matrix Analysis and Applications*, **25** (1), 246-265.

Johnson, M. & Maggiora, G. (1990). *Concepts and Applications of Molecular Similarity.* Wiley, New York.

Kaniel, S. (1966). "Estimates for some computational techniques in linear algebra". *Mathematics of Computation*, **20** (95), 369-378.

Kelley, L., Gardner, S. & Sutcliffe, M. (1996). "An automated approach for clustering an ensemble of NMR-derived protein structures into conformationally related subfamilies". *Protein Engineering Design and Selection*, **9** (11), 1063.

Khademhosseini, A. (2002). *Singular Value Decomposition(SVD) tutorial* [Online]. Cambridge, MA, USA: MIT. http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.html [Accessed 11/09/11]

Kola, I. & Landis, J. (2004). "Can the pharmaceutical industry reduce attrition rates?". *Nature Reviews Drug Discovery*, **3** (8), 711-716.

Kong, W.-Z., Sun, Z.-H., Yang, C., Dai, G.-J. & Sun, C. (2010). "Automatic spectral clustering based on eigengap and orthogonal eigenvector". *Dianzi Xuebao(Acta Electronica Sinica)*, **38** (8).

Krejsa, C., Horvath, D., Rogalski, S., Penzotti, J., Mao, B., Barbosa, F. & Migeon, J. (2003). "Predicting ADME properties and side effects: the BioPrint approach". *Current Opinion in Drug Discovery & Development*, **6** (4), 470.

Kubinyi, H. (1998). "Similarity and dissimilarity: a medicinal chemist's view". *Perspectives in Drug Discovery and Design*, **9**, 225-252.

Lance, G. & Williams, W. (1967). "A general theory of classificatory sorting strategies: 1. Hierarchical systems". *The Computer Journal*, **9** (4), 373.

Lanczos, C. (1950). "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators". *Journal of Research of the National Institute of Standards and Technology*, **45** (4), 255-282.

Landrum, G. (2006). *RDKit: Open-source cheminformatics* [Online]. http://www.rdkit.org [Accessed 03/04/2012]

Larsen, R.M. (1998). "Lanczos bidiagonalization with partial reorthogonalization". *DAIMI PB*, **27** (537).

Larsen, R.M. (2001). "Combining implicit restarts and partial reorthogonalization in Lanczos bidiagonalization". *SCCM, Stanford University*.

Laval, X., Delarge, J., Somers, F., Tullio, P., Henrotin, Y., Pirotte, B. & Dogne, J.M. (2000). "Recent advances in inducible cyclooxygenase (COX-2) inhibition". *Current Medicinal Chemistry*, **7** (10), 1041-1062.

Leach, A. & Gillet, V. (2007). *An introduction to chemoinformatics.* Springer Verlag.

Li, K. & Liu, Y.-S. (2007). "A spectral clustering algorithm based on self-adaption". *Machine Learning and Cybernetics, 2007 International Conference on*, Vol. 7, pp. 3965-3968. IEEE.

Lin, J.I. (1991). "Bounds on eigenvalues of finite element systems". *International Journal for Numerical Methods in Engineering*, **32** (5), 957-967.

Lipinski, C. & Hopkins, A. (2004). "Navigating chemical space for biology and medicine". *Nature*, **432** (7019), 855-861.

Macarron, R., Banks, M.N., Bojanic, D., Burns, D.J., Cirovic, D.A., Garyantes, T., Green, D.V., Hertzberg, R.P., Janzen, W.P. & Paslay, J.W. (2011). "Impact of high-throughput screening in biomedical research". *Nature Reviews Drug Discovery*, **10** (3), 188-195.

Martin, Y., Kofron, J. & Traphagen, L. (2002). "Do structurally similar molecules have similar biological activity?". *Journal of Medicinal Chemistry*, **45** (19), 4350-4358.

MatrixMarket (2011). *Harwell-Boeing matrix format* [Online]. http://math.nist.gov/MatrixMarket/formats.html - hb [Accessed 15/12/2012]

MDL (2006). *MDL Drug Data Report* [Online]. Website no longer active

Milligan, G. (1981). "A Monte Carlo study of thirty internal criterion measures for cluster analysis". *Psychometrika*, **46** (2), 187-199.

Milligan, G. & Cooper, M. (1985). "An examination of procedures for determining the number of clusters in a data set". *Psychometrika*, **50** (2), 159-179.

Morgan, H. (1965). "The generation of a unique machine description for chemical structures - A technique developed at chemical abstracts service". *Journal of Chemical Documentation*, **5** (2), 107-113.

Murtagh, F. (1983). "A survey of recent advances in hierarchical clustering algorithms". *The Computer Journal*, **26** (4), 354.

Nepusz, T., Sasidharan, R. & Paccanaro, A. (2010). "SCPS: a fast implementation of a spectral method for detecting protein families on a genome-wide scale". *BMC Bioinformatics*, **11**.

Neres, J., Brewer, M.L., Ratier, L., Botti, H., Buschiazzo, A., Edwards, P.N., Mortenson, P.N., Charlton, M.H., Alzari, P.M., Frasch, A.C., Bryce, R.A. & Douglas, K.T. (2009). "Discovery of novel inhibitors of Trypanosoma cruzi trans-sialidase from in silico screening". *Bioorganic & Medicinal Chemistry Letters*, **19** (3), 589-596.

Ng, A., Jordan, M. & Weiss, Y. (2002). "On spectral clustering: Analysis and an algorithm". *Advances in neural information processing systems*, **2**, 849-856.

Ng, A.Y., Zheng, A.X. & Jordan, M.I. (2001). "Link analysis, eigenvectors and stability". *International Joint Conference on Artificial Intelligence*, Vol. 17, pp. 903-910. Citeseer.

Nguyen, N., Milanfar, P. & Golub, G. (2001). "Efficient generalized cross-validation with applications to parametric image restoration and resolution enhancement". *Image Processing, IEEE Transactions on*, **10** (9), 1299-1308.

Oregon, U.o. (2012). "Mapping Life Expectancy".

Paccanaro, A., Casbon, J.A. & Saqi, M.A.S. (2006). "Spectral clustering of protein sequences". *Nucleic Acids Research*, **34** (5), 1571-1580.

Page, L., Brin, S., Motwani, R. & Winograd, T. (1999). "The PageRank citation ranking: bringing order to the web".

Paige, C.C. (1971). *The computation of eigenvalues and eigenvectors of very large sparse matrices.* University of London.

Parlett, B.N. (1998). *The symmetric eigenvalue problem.* Society for Industrial Mathematics.

Parsons, L., Haque, E. & Liu, H. (2004). "Subspace clustering for high dimensional data: a review". *ACM SIGKDD Explorations Newsletter*, **6** (1), 90-105.

Patterson, D., Cramer, R., Ferguson, A., Clark, R. & Weinberger, L. (1996). "Neighborhood behavior: a useful concept for validation of “molecular diversity” descriptors". *Journal of Medicinal Chemistry*, **39** (16), 3049-3059.

Paul, S.M., Mytelka, D.S., Dunwiddie, C.T., Persinger, C.C., Munos, B.H., Lindborg, S.R. & Schacht, A.L. (2010). "How to improve R&D productivity: the pharmaceutical industry's grand challenge". *Nature Reviews Drug Discovery*, **9** (3), 203-214.

Pedoe, D. (1988). *Geometry: A comprehensive course.* Dover Pubns.

Pena, J., Lozano, J. & Larranaga, P. (1999). "An empirical comparison of four initialization methods for the k-means algorithm". *Pattern recognition letters*, **20** (10), 1027-1040.

Perona, P. & Freeman, W. (1998). "A factorization approach to grouping". In: Burkhardt, H. & Neumann, B. (eds.), *Computer Vision — ECCV'98*, Vol. 1406, pp. 655-670. Springer Berlin / Heidelberg.

pharmainfo.net (2012). *FDA New Molecular Entities approved from 1999 - 2010* [Online]. http://www.pharmainfo.net/files/groupsimages/fda_approved_drugs_list_from_year_1999-2010.pdf

Pickett, S., Mason, J. & McLay, I. (1996). "Diversity profiling and design using 3D pharmacophores: pharmacophore-derived queries (PDQ)". *Journal of Chemical Information and Computer Sciences*, **36** (6), 1214-1223.

Press, W., Teukolsky, S., Vetterling, W. & Flannery, B. (1992). *Numerical recipes in FORTRAN: the art of scientific computing.* Cambridge Univ Pr.

Press, W.H., Flannery, B.P., Teukolsky, S.A. & Vetterling, W.T. (2007). *Numerical recipes.* Cambridge Univ Press.

Randic, M. (1975). "Characterization of molecular branching". *Journal of the American Chemical Society*, **97** (23), 6609-6615.

Read, R.C. & Corneil, D.G. (1977). "The graph isomorphism disease". *Journal of Graph Theory*, **1** (4), 339-363.

Rees, D.C., Congreve, M., Murray, C.W. & Carr, R. (2004). "Fragment-based lead discovery". *Nature Reviews Drug Discovery*, **3** (8), 660-672.

Rogers, D., Brown, R. & Hahn, M. (2005). "Using extended-connectivity fingerprints with Laplacian-modified Bayesian analysis in high-throughput screening follow-up". *Journal of Biomolecular Screening*, **10** (7), 682.

Rohde, D. (2009). *SVDLIBC-SVD C Library* [Online]. http://tedlab.mit.edu/~dr/SVDLIBC/ [Accessed 06/02/2012]

Rousseeuw, P.J. (1987). "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis". *Journal of Computational and Applied Mathematics*, **20** (0), 53-65.

Rumelhart, D., Hinton, G. & Williams, R. (2002). "Learning representations by back-propagating errors". *Cognitive modeling*, 213.

Salim, N., Holliday, J. & Willett, P. (2002). "Combination of Fingerprint-Based Similarity Coefficients Using Data Fusion". *Journal of Chemical Information and Computer Sciences*, **43** (2), 435-442.

Sams-Dodd, F. (2005). "Target-based drug discovery: is something wrong?". *Drug discovery today*, **10** (2), 139-147.

Sarkar, S. & Boyer, K. (1998). "Quantitative measures of change based on feature organization: Eigenvalues and eigenvectors". *Computer Vision and Image Understanding*, **71** (1), 110-136.

Schneider, G. & Wrede, P. (1998). "Artificial neural networks for computer-based molecular design". *Progress in biophysics and molecular biology*, **70** (3), 175-222.

Schuffenhauer, A., Brown, N., Ertl, P., Jenkins, J., Selzer, P. & Hamon, J. (2007). "Clustering and rule-based classifications of chemical structures evaluated in

the biological activity space". *Journal of Chemical Information and Modeling*, **47** (2), 325-336.

Scott, G. & Longuet-Higgins, H. (1990). "Feature grouping by relocalisation of eigenvectors of the proximity matrix". pp. 103–108.

Senger, S. (2009). "Using Tversky similarity searches for core hopping: finding the needles in the haystack". *Journal of Chemical Information and Modeling*, **49** (6), 1514-1524.

Shi, J. & Malik, J. (2000). "Normalized cuts and image segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22** (8), 888-905.

Simon, H.D. (1984). "The Lanczos algorithm with partial reorthogonalization". *Mathematics of Computation*, **42** (165), 115-142.

Sleijpen, G.L., Booten, A.G., Fokkema, D.R. & Van der Vorst, H.A. (1996). "Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems". *BIT Numerical Mathematics*, **36** (3), 595-633.

Sleijpen, G.L. & Van der Vorst, H.A. (2000). "A Jacobi--Davidson iteration method for linear eigenvalue problems". *SIAM Review*, **42** (2), 267-293.

Sleijpen, G.L., van der Vorst, H.A. & Meijerink, E. (1998). "Efficient expansion of subspaces in the Jacobi-Davidson method for standard and generalized eigenproblems". *Electronic Transactions on Numerical Analysis*, **7**, 75-89.

Stahl, M. & Rarey, M. (2001). "Detailed analysis of scoring functions for virtual screening". *Journal of Medicinal Chemistry*, **44** (7), 1035-1042.

Strang, G. (2003). *Introduction to linear algebra.* Wellesley Cambridge Pr.

Symyx (2007). *MDL software* [Online]. http://www.symyx.com/ [Accessed 10/10/10]

Taylor, R. (1995). "Simulation analysis of experimental design strategies for screening random compounds as potential new drugs and agrochemicals". *Journal of Chemical Information and Computer Sciences*, **35** (1), 59-67.

Trinajstic, N. (1992). *Chemical graph theory.* CRC press Boca Raton, FL.

Tripos (2007). *SYBYL Software* [Online]. http://www.optive.com/ [Accessed 10/10/10]

Trolltech, A. (2008). *Qt cross platform application framework* [Online]. http://qt-project.org

van den Eshof, J. (2002). "The convergence of Jacobi,ÄìDavidson iterations for Hermitian eigenproblems". *Numerical linear algebra with applications*, **9** (2), 163-179.

Varin, T., Bureau, R., Mueller, C. & Willett, P. (2009). "Clustering files of chemical structures using the Szekely-Rizzo generalization of Ward's method". *Journal of Molecular Graphics and Modelling*, **28** (2), 187-195.

Varin, T., Saettel, N., Villain, J., Lesnard, A., Dauphin, F., Bureau, R. & Rault, S. (2008). "3D Pharmacophore, hierarchical methods, and 5-HT4 receptor binding data". *Journal of Enzyme Inhibition and Medicinal Chemistry*, **23** (5), 593-603.

von Luxburg, U. (2007). "A tutorial on spectral clustering". *Statistics and Computing*, **17** (4), 395-416.

Walters, W.P., Stahl, M.T. & Murcko, M.A. (1998). "Virtual screening - an overview". *Drug Discovery Today*, **3** (4), 160-178.

Weininger, D. (1988). "SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules". *Journal of Chemical Information and Computer Sciences*, **28** (1), 31-36.

Weininger, D., Weininger, A. & Weininger, J. (1989). "SMILES. 2. Algorithm for generation of unique SMILES notation". *Journal of Chemical Information and Computer Sciences*, **29** (2), 97-101.

Weiss, Y. (1999) In *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*IEEE Computer Society, pp. 975.

Wermuth, C.G. (2004). "Selective optimization of side activities: another way for drug discovery". *Journal of Medicinal Chemistry*, **47** (6), 1303-1314.

Wild, D.J. & Blankley, C.J. (2000). "Comparison of 2D fingerprint types and hierarchy level selection methods for structural grouping using Ward's clustering". *Journal of Chemical Information and Computer Sciences*, **40** (1), 155-162.

Willett, P. (1987). *Similarity and clustering in chemical information systems.* Research Studies Press Letchworth.

Willett, P. (2009). "Similarity Methods in Chemoinformatics". *Annual Review of Information Science and Technology*, **43**, 3-71.

Willett, P., Barnard, J. & Downs, G. (1998). "Chemical similarity searching". *Journal of Chemical Information and Computer Sciences*, **38** (6), 983-996.

Willett, P., Winterman, V. & Bawden, D. (1986). "Implementation of nearest-neighbour searching in an online chemical structure search system". *Journal of Chemical Information and Computer Sciences*, **26** (1), 36-41.

Xia, X., Maliski, E., Gallant, P. & Rogerss, D. (2004). "Classification of kinase inhibitors using a Bayesian model". *Journal of Medicinal Chemistry*, **47** (18), 4463-4470.

Xiong, H., Tan, P. & Kumar, V. (2006). "Hyperclique pattern discovery". *Data Mining and Knowledge Discovery*, **13** (2), 219-242.

Zadeh, L. (1965). "Fuzzy Sets". *Information and Control*, **8** (3), 338-353.