Development of a comprehensive 30 structural database of human pan-proteomic interaction with the most valuable sold drugs

Edwards, William H.

Swansea University
Prifysgol Abertawe

Development of a comprehensive 3D structural database of human pan-proteomic interaction with the most valuable sold drugs

William Harry Edwards
Swansea University

PhD

Summary

Drug discovery is undertaken to discover new candidate medicines. Identifying new therapeutics is of critical medical, social and economic importance. In recent years the rising cost associated with drug discovery and development has necessitated the more frequent use of *in silico* approaches. The prediction of protein-ligand interactions using *in silico* approaches has become widely used to study biomolecular interactions and mechanisms. These approaches allow for high throughput virtual screening of thousands of potential drug candidates at specific protein drug targets. While the data generated provides ample opportunity for scientific and clinical exploitation, challenges are presented concerning the vast quantity of information generated and the ability to utilise this information fruitfully.

In this study *in silico* approaches are used to “virtually screen” the entire human proteome against the 20 most valuable sold drugs in the UK. Homology and protein threading approaches were used for protein structure modelling; AutoDock Vina and DOCK 6.0 were used for protein-ligand docking with the use of High-Performance Computing (HPC).

The large-scale application of these approaches was evaluated, and methods iteratively refined to improve predictive accuracy. A novel combinational forecasting method was developed to increase the accuracy of the predictions of the docking programs. The method produced a docking an overall accuracy of 77.05% for identifying known protein interactions and known protein misses correctly. A platform system was developed to allow the vast amount of data to be efficiently reported and visualised within a Graphical User Interface (GUI).

The developed database and system prototype have the potential to change the way drugs are developed in the drug discovery sector. This system is a powerful tool which can be used for the advancement of personalised medicine with incorporation of further knowledge of protein interactions as well as the effects of protein variation.

The work carried out in this project has contributed towards the development of a comprehensive in silico platform, Human3DProteome (human3dproteome.com), that utilises the system architecture, methodologies, data and methods of data analysis advanced in this project. Human3DProteome is the first public platform which aims to catalogue structural models for every protein in the human body alongside a comprehensive database of predicted small molecule interactions of interest.

Declaration

This work has not previously been accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed………………………………………………………………………….

Date…………………………………………………………………………….

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Where correction services have been used, the extent and nature of the correction is clearly marked in a footnote(s).

Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed………………………………………………………………………….

Date…………………………………………………………………………….

Statement 2

I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed………………………………………………………………………….

Date…………………………………………………………………………….

Contents

Abbreviations

2SD - mean and standard deviation outlier method

ABPI – The Association of the British Pharmaceutical industry

ADRs – Adverse Drug Reactions

AJAX - Asynchronous JavaScript and XML

API – Application Programming Interface

AUC – Area-under-the-receiver-operating-characteristic-curves

CASP – Critical Assessment of Techniques for Protein Structure Prediction

CPU – Central Processing Unit

Cryo-EM – Cryo-electron Microscopy

CSS – Cascading Style Sheets

DMS – Distributed molecular surface

DSX – Drug Score X

ERD – Entity Relationship Diagram

EVD – Extreme value distribution

GBSA – Generalised Born with solvent accessible Surface Area

GUI – Graphical User Interface

HPC – High-Performance Computing

HSP – High-scoring segment pairs

HTML – HyperText Markup Language

HTS – High-throughput screening

HUGO – Human Genome Organization

I-Tasser – Iterative Threading ASSEmbly Refinement

KEGG - Kyoto Encyclopaedia of Genes and Genomes

MSA – Multiple sequence alignment

MySQL – Structured Query Language

NCBI – National Centre for Biotechnology Information

NGS – Next Generation Sequencing

NMR – Nuclear Magnetic Resonance

NR – Non-Redundant

OS – Operating System

PDB – Protein Data Bank	
PHP – Personal Home Page	
RMSD – Root-mean-square distance	
ROC – Receiver Operating Characteristics	
SDF – Structure-data File	
SVM – Support vector machine	
TM-align – Template Model align	
VM – Virtual Machine	

List of Figures

List of Tables

Acknowledgements

Firstly, I would like to thank my supervisor Dr Jonathan Mullins for giving me this opportunity. His continuous support and patience throughout my PhD were invaluable. This constant backing has been crucial to all my achievements thus far.

I thank my fellow lab mates Karl Austin-Muttitt, Sam West, James Witts, and John Walshe for the support and for making the past four years enjoyable. I would also like to thank my parents for supporting me all the way through my PhD and being there when I've most needed it.

1 Introduction

1.1 Overview

The aim of this project is to develop a comprehensive 3D structural database of human proteomic variation and pan-proteomic interaction with the most widely sold drugs. The work is part of a more comprehensive project focused on the implementation and scaling of ligand (drug) docking algorithms on High-Performance Computing (HPC) Wales and Supercomputing Wales. These algorithms will be used in conjunction with pre-computed 3D-structures of the whole human proteome, with potential future application to the investigation of the impacts of genetic mutations which have been computed as part of the wider project but are not featured in this thesis, as the structural database also contains hundreds of thousands of variants as detailed in the UniProt protein database (UniProt Consortium, 2018). The results will be stored in a publicly accessible database for future work.

The interaction between proteins and chemical compounds is an inherently complex issue, bearing vital importance due to its relevance to the pharmaceutical, healthcare and medical research sectors. Recent advances in bioinformatics and molecular modelling have enabled *in silico* structural prediction approaches which have the potential for scaling to the whole- or multi-genome analysis. In particular, new technologies such as High-Performance Computing support rapid development in the bioinformatics field, enabling the generation of vast amounts of data relating to medically critical molecular interactions. While the data generated provides ample opportunity for scientific and clinical exploitation; challenges are presented concerning the vast quantity of information generated and the ability to access this information fruitfully.

Screening approaches for candidate drugs or chemical agents will continue to provide an incomplete picture unless tested against all possible variants in each organism. Furthermore, the scope for identifying potential adverse drug reactions by *in silico* approaches will not be realised unless a whole proteome approach is adopted.

Additionally, the knowledge and understanding of the molecular interactions of drug targets are of a significant commercial interest with direct relevance to the following processes;

a). Assist the lead discovery process

b). Re-purpose existing drugs to maximise revenue and

c). Reduce the risk (both financial and medical) of drug development by improving efficacy and reducing adverse reactions.

The prevailing approach to drug discovery in recent decades is that of rational drug design, which involves the development of new drugs with a specific target in mind (usually a specific site in a single or small number of protein targets which bind with the ligand). Adverse drug events due to off-target interactions are not currently assessed by *in silico* approaches. There are *in vitro* approaches that are increasingly employed, typically involving pharmacological profiling or toxicology panels comprising binding assays, such as for around 44 protein receptors (Bowes et al., 2012). Such *in vitro* toxicology panel data provide valuable validation information for our *in silico* approaches. There are, however, no panel data for most drugs in use today. Data for known toxins are given in the literature reviewed in Bowes et al., (2012), and prolifically in the NIH TOXNET database (Fowler and Schnall, 2014) and this data will be used to test and validate the proposed database platform.

1.2 The Human Genome Project

The Human Genome Project was an international collaboration, the objective of which was to fully map and sequence every gene within the human body and to make it freely available for research. Twenty different research groups from the United Kingdom, France, Germany, the United States, China, and Japan have helped to produce a draft of the genome sequences that cover around 94% of the human genome (Lander et al., 2001). The idea of sequencing the entire human genome began in the early 1980s and was discussed in a scientific meeting which was organised by the US Department of Energy. From this meeting, it was agreed that there was a need for a broader research objective. This included:

- The creation of genetic, physical and sequence maps of the human genome.
- The parallel efforts in key model organisms such as bacteria, yeast, worms, flies and mice.
- The development of technologies that would aid the process of reaching the objectives.
- Research into the ethical, legal and social issues raised by human genome research.

The project was initially launched by in the United States by the National Institute of Health and the Department of Energy. This began the international collaboration, which led to the Human Genome Organization (HUGO) being founded.

The Human Genome Project reached it's objective and was completed in 2003, which was two years ahead of their original deadline and successfully sequenced 20,000-25,000 genes. The work that has been carried out by the Human Genome Project has allowed researchers to delve deeper into the working of the human body and learn how it functions.

An essential component of this project is the structural modelling of a large variety of human proteins, and this work would not have been possible without the gene sequences provided by the Human Genome Project.

1.3 Next-Generation Sequencing (NGS)

Next-Generation Sequencing, which is also known as high-throughput sequencing is a collection of new modern sequencing technologies used for DNA sequencing. NGS has made it possible to sequence thousands to millions of DNA molecules simultaneously which increases the speed of research. These new methods of sequencing have and continue to be very beneficial to fields such as personalised medicine, genetic diseases and clinical diagnostics.

NGS platforms were built upon Sanger sequencing (Heather and Chain, 2016), which is still classed today as the leading method for sequencing DNA molecules and is used to validate the data generated by the NGS. Four different NGS platforms are currently in use, these being Illumina (Solexa) sequencing, Roche 454 sequencing, Ion torrent: Proton / PGM sequencing, and SOLiD sequencing.

The four different systems have common features to each other, these being the sample preparation, the sequencing machines, and the data outputted by them. Each of the NGS platforms requires a library of samples that have been obtained by either amplification or elongation with custom adaptor sequences (Head et al., 2014). The libraries used for the sequencing process. The common features of the sequencing machines used are that they all amplify library fragments onto solid surfaces with covalently attached DNA linkers which hybridise the library adaptors. This method is called bridge PCR which is mainly used by Illumina sequencing (Heather and Chain, 2016). The other method used is called emulsion PCR which involves amplification which creates clusters on DNA which each are generated by a single library fragment. With the emulsion PCR method, each of the clusters is classified as a single sequence reaction. All four of the NGS platform uses the emulsion PCR method (Kanagal-Shamanna, 2016). The final common feature is that all the NGS platforms provide the same raw data output at the end of the sequencing run. The raw data is a collection of DNA sequences that had been generated by each cluster in the sequencing process.

The differences between the four NGS platforms are based on the method used to carry out and record the sequencing reactions. There are four different methods which are pyrosequencing, sequencing by synthesis, sequencing by ligation, and ION semiconductor sequencing. Pyrosequencing is the basis of the Roche 454 NGS platform and uses pyrophosphate during the nucleotide incorporation process (Harrington et al., 2013). The phosphate is used in a series of chemical reactions which causes the generation of light which is captured by a camera that then records the appropriate sequence for that cluster. A single base is added at a time to the solution, and the light is measured which if detected degrades the other unincorporated bases. The method is then repeated with the next base and continues until the sequencing process is complete.

Sequencing by synthesis is the second method that is used by Illumina and is the most popular of the four methods available. The following method utilises the step by step incorporation of reversible florescent and terminated nucleotides for DNA sequencing. Instead of one base added at a time like in the pyrosequencing method, sequencing by synthesis adds all four nucleotides to the same sequencing chip. When one of the bases have been incorporated, the other three are washed away. The fluorescent signal from the incorporated base is then read and recorded for each cluster. The fluorescent and terminator group of the base is then cleaved and washed away; then the process is repeated until the sequencing reactions are complete. (Guo et al., 2010)

The SOLiD NGS platform uses sequencing by Ligation. The following method, instead of using DNA polymerase like Pyrosequencing and Sequencing by synthesis, which were discussed previously, sequencing by ligation, relies on 16 octamer oligonucleotide probes. Each of the oligonucleotide probes has one of the four fluorescent dyes attached there five prime ends which have been ligated. Each of the octamers has two probe specific bases with six degenerated bases. This means that each fluorescent light represents two bindings of two nucleotides. For the sequence reaction, this first step is to bind the primer to the adaptor, which is then hybridised with the appropriate probe. When one of the probes have been annealed, it is then ligated to the primer sequence through a DNA ligase. The probes which have not annealed are then washed away. The signal from the fluorescent dye is then detected and recorded. The last three bases of the octamer are cleaved, and the process is repeated for 7-cycles of the ligation. After this step, the primer is then removed, and a new primer is added but offset by one base compared to the previous primer. The process is then repeated with the new primer. This step is only repeated five times. (Garrido-Cardenas et al., 2017)

The final sequencing method, which is used is ION semiconductor sequencing that is used by the Ion torrent: Proton / PGM sequencing NGS platform. The method uses a similar step to pyrosequencing, but instead of using pyrophosphate, ION semiconductor sequencing measures the release of hydrogen ions during the sequence reaction to detect the sequence of a cluster. The clusters are all located above a semiconductor transistor that can detect pH changes within the solution. During the nucleotide incorporation, a single hydrogen ion is released into the solution which is then detected by the semiconductor. The Ion semiconductor method is a faster method of sequencing compared to pyrosequencing. (Alekseyev et al., 2018)

With the development of these NGS technologies, it has made it possible to look at genome-wide interactions and brought the possibility of personalised medicine even closer with personal whole genome sequencing.

1.4 An Introduction to Protein Structure

The human body has up to 20,000 different types of proteins within it, each with its own job to do (Ponomarenko et al., 2016). Proteins consist of a long chain of amino acids, the details of which provide their structure and function. It is possible to extract the amino acid sequences of these proteins from protein sequence databases such as UniProt (UniProt Consortium, 2018) and Swiss-Prot (Bairoch, 2000). But with only these amino acid sequences, there is limited information that can be derived about the precise function and mode of action of the protein. This is where the importance of knowing the protein structure comes to the fore. The three-dimensional structure is determined by the way the amino acids in the polypeptide chains interact with each other which cause folds to be generated from linear chains. The folded domains can be part of large assemblies like virus particles or muscle fibres. The three-dimensional structure of the protein ultimately defines its function, such as which molecules an enzyme will bind and process, or which atoms a transporter will carry, or how a protein might regulate the expression of DNA (Brändén and Tooze, 2009)

1.4.1 Amino Acids

Shown in table 1.1 are all 20 amino acids, their 3-letter and 1-letter symbols. All amino acids have a central carbon atom. A Hydrogen atom, amino group, and a carboxyl group are attached to this central carbon atom. An example is shown in figure 1.1. What distinguishes one amino acid from the other are that they have different side chains which are attached to the central carbon atom.

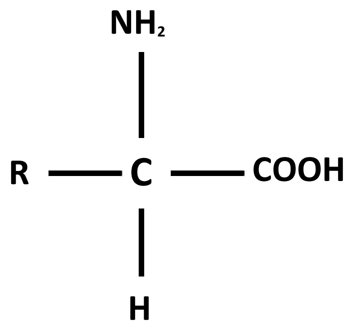


Figure 1.1: Example of the formula for a general amino acid – shown is the general formula for an amino acid. Each amino acid has a hydrogen atom (H), amino group (NH_2), a carboxyl group (COOH), and R-group (R) that is attached to a central carbon atom (C).

Shown in figure 1.2 is a Venn diagram showing the different types of amino acid property groups and which amino acid is a part of each group. There are amino acids that are classified in more than one group. Shown in table 1.2 is a list of each group and an explanation about the type of amino acid that is in that category.

Table 1.1:Amino acid names and abbreviations – shown are the 20 amino acids, and there 3 lettered and single lettered abbreviations.

Amino Acid Name	**Amino Acid 3-letter symbol**	**Amino Acid 1-letter symbol**
arginine	arg	R
Asparagine	asn	N
aspartic acid	asp	D
cysteine	cys	C
glutamine	gln	Q
glutamic acid	glu	E
glycine	gly	G
histidine	his	H
isoleucine	ile	I
leucine	leu	L
lysine	lys	K
methionine	met	M
phenylalanine	phe	F
proline	pro	P
serine	ser	S
threonine	thr	T
tryptophan	trp	W
tyrosine	tyr	Y
valine	val	V

Table 1.2: Different type of amino groups – shown are 6 different amino acid groups. Amino acids can cross into more than one of these groups, which are shown in figure 1.3.

Amino Acid Group	**Definition**
Aliphatic	the R-groups are nonpolar and hydrophobic
Aromatic	mostly nonpolar but all amino acids in this group can absorb ultraviolet light
Hydrophobic	Side-chains are mostly composed of carbon and hydrogen and tend to repel water.
Polar	side chains that can either be charged or can participate in hydrogen bonding.
Positively Charged	can be charged at pH=7 (basic side chain).
Negatively charged	can be negatively charged at pH=7 (acidic side chain).

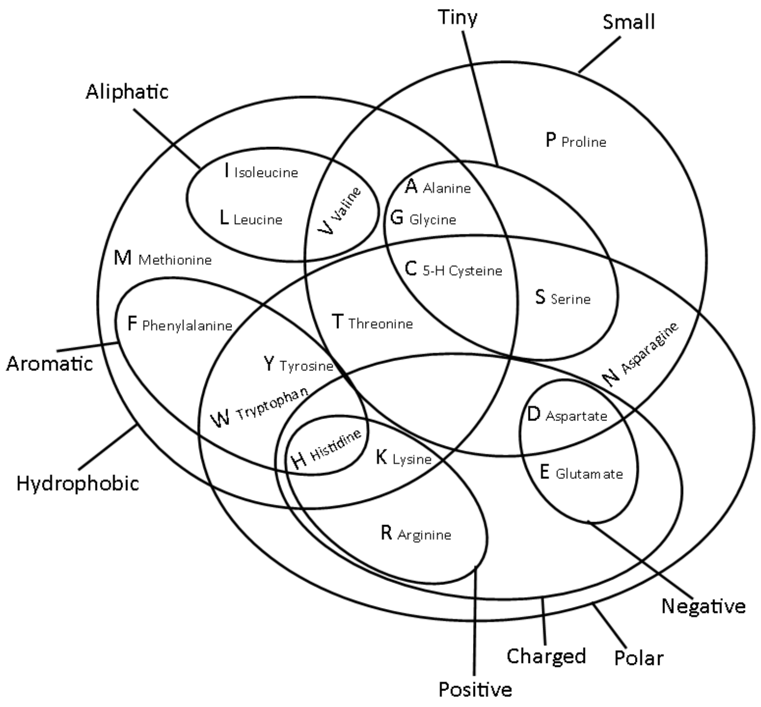


Figure 1.2: Venn diagram showing the different types of amino acid property groups – shown are the different types of amino acid groups that exist and displays the type of each amino acid.

For these amino acids to join to create a protein, the central carbon atom of one amino acid connects to the nitrogen atom of another amino acid via a polypeptide bond. Proteins can consist of very long polypeptide chains ranging from 100s of amino acids residues to thousands of them. Four levels of protein structure exist which are shown in figure 1.3. These are the Primary Structure, Secondary Structure, Tertiary Structure, and Quaternary Structure.

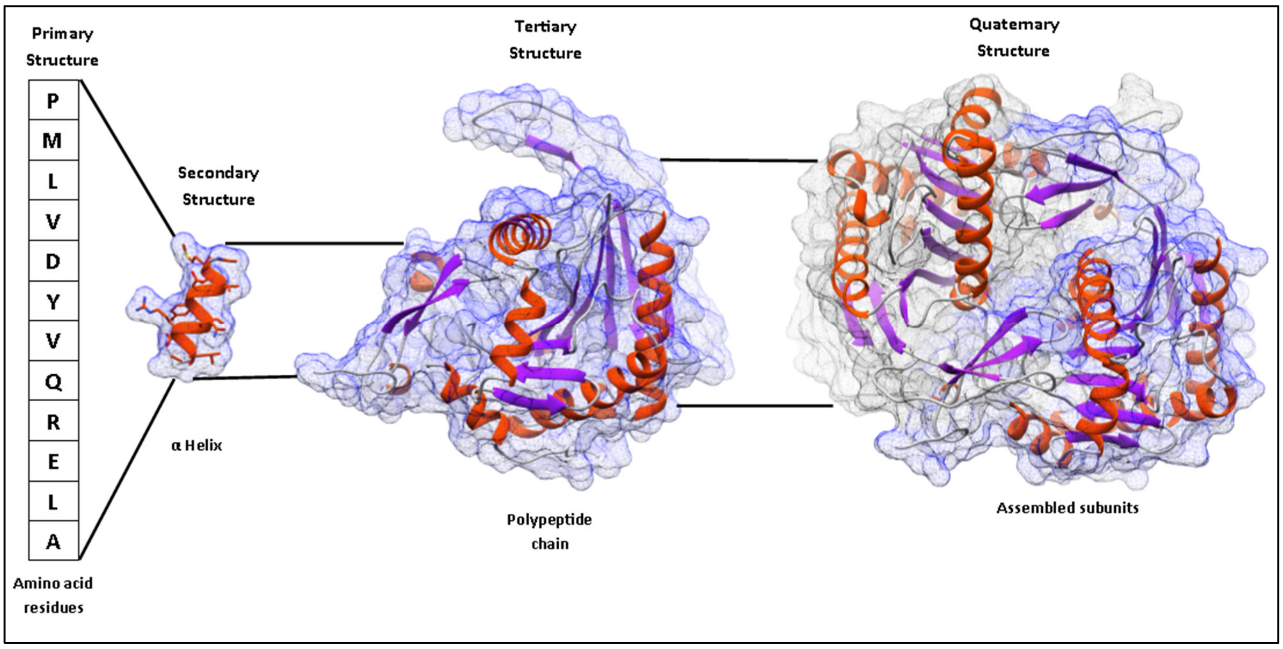


Figure 1.3:Four Levels of protein structure – shown is an image of the 4 levels of a protein's structure. The four levels consist of the primary structure, Secondary structure, Tertiary structure, and Quaternary Structure.

1.4.2 Primary structure

The Primary Structure of a protein is defined as a linear sequence of amino acids. The linear sequence is referred to as a polypeptide chain. These amino acids within the chain are held together with covalent bonds which are created during the process of protein synthesis/translation. These are the building blocks of a protein structure (Nelson and Cox, 2017). The primary structure of a protein is determined by the gene, which corresponds to that particular protein. A specific sequence of nucleotides is extracted from DNA and is transcribed into mRNA. The mRNA sequence is then read by ribosomes which are called translation. It was discovered that proteins have defining amino acid sequences by Frederick Sanger who sequenced the amino acids in insulin (Sanger, 1959). This discovery showed that an amino acid sequence determines the structure and function of a protein.

1.4.3 Secondary Structure

The Secondary Structure is the stable arrangements of amino acid residues which result in a structural pattern (Nelson and Cox, 2017). These relate to parts which are seen a lot in proteins being the α-helices and β turns. Coils in a protein structure are known as an α-helix (shown in figure 1.4), and folds in a protein structure are known as β turns or β-pleated sheets (shown in figure 1.5). An α-helix is formed by hydrogen bonding between every fourth amino acid. For a β turns to be formed two different regions of a polypeptide chains lay next to each other and are bound by a hydrogen bond. There are two different types of β turns that can occur these being a parallel β turn and an antiparallel β turn. The stability of the sheets is mainly determined by how the hydrogen bonds form. These bonds can form between a partially negative oxygen atom and a partially positive nitrogen atom (Rehman and Botelho, 2019). The secondary structure can be defined as consecutive fragments of protein sequences that corresponds to a region of protein structure that shows a distinctive geometrical feature (Pirovano and Heringa, 2009).

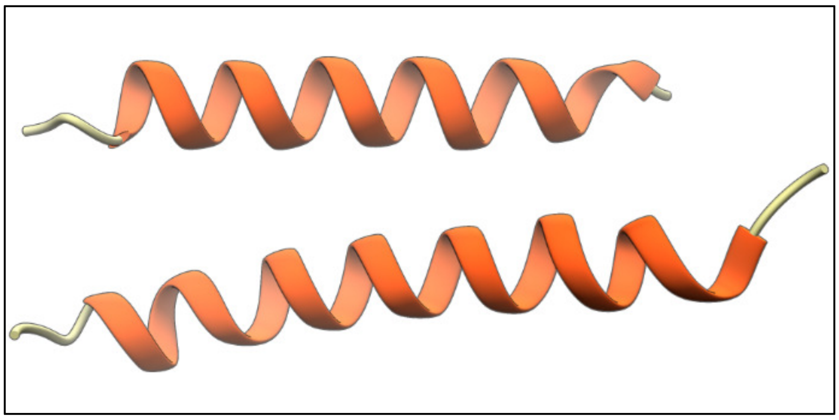


Figure 1.4:Example of an α-helix – shown is an example of a modelled α-helix.

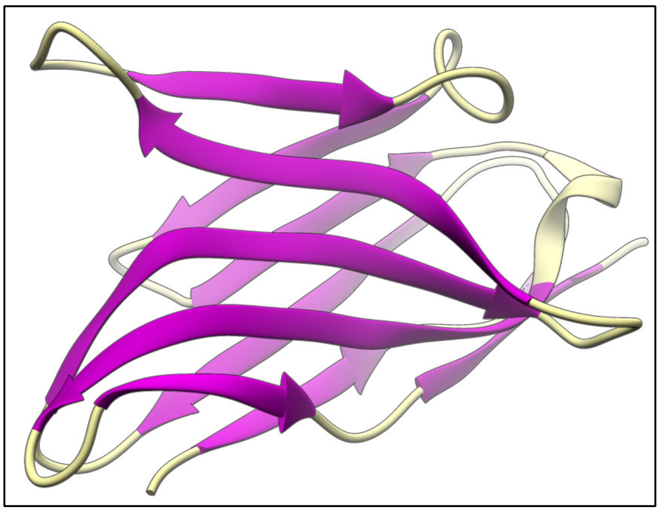


Figure 1.5:Example of a β turn/sheet – shown is an example of a modelled β turn/sheet.

1.4.4 Tertiary Structure

The Tertiary structure is all the aspects of the three-dimensional folding of a polypeptide (Nelson and Cox, 2017). This is the final arrangement of all the residues which are formed by the interactions of the side chains and create a 3D structure of the protein. The properties of the amino acid sequence determine the overall shape of the tertiary structure. (Rehman and Botelho, 2018) The tertiary structure is held together by the interactions between the R-group and the side-chain, which is created by hydrogen bonding, hydrophobic interaction, ionic bonding and disulphide bonding (Medarametla, 2014).

There are two different types of tertiary structure which are fibrous and globular. The fibrous tertiary structure is usually just having structural roles in the body whereas, for globular tertiary structures, these have a more complicated structure which has multiple types of secondary structure within one polypeptide chain (Nelson and Cox, 2017).

1.4.5 Quaternary Structure

Sometimes proteins contain different polypeptide chains (or subunits) that join to form a complex, and this is referred to as a Quaternary Structure (Nelson and Cox, 2017). These subunits can operate as a single functional multimer. There are different terms used depending on the number of subunits within the multimer. If there are 2 subunits within the multimer, these are called dimers, if there are 3 subunits these would be called a trimer, if there are 4 subunits these would be called tetramers, and finally, if there are 5 subunits within a multimer these would be called pentamers. Usually the subunits within a multimer are related to each other through symmetry operations. If a multimer contains identical subunits within the word “homo” is added to the start of the word for example if there was a tetramer within identical subunits this would be called a homotetramer. Multimers with different subunits would have the word “hetero” added to the front for example heterotetramer.

1.5 Genetic Variants/Mutations

Human DNA is not identical between individuals, so there may be 20,000 proteins in the human body, but no two people have the same sequenced proteins. Four different types of mutations occur, these being a missense mutation, Nonsense mutation, "silent" mutation, and a frameshift mutation. A missense mutation is when a single amino acid within a sequence changes to a different amino acid. These mutations can be classified as a polymorphism if the mutation has been observed in a fraction of the population or it could be a rare mutation which has been found in an individual or a small group of people (Zhang et al., 2012). A nonsense mutation is when a stop codon replaces an amino acid which causes premature termination of the translation (Lodish et al., 2000). This mutation leads to major changes to the sequence of a protein which in turn changes that proteins assembly and function (Ruwald et al., 2016). A "silent" mutation does not affect the amino acid sequence but can still cause phenotypic effects such as speeding up or slowing down protein synthesis. These mutations are classified as passenger events (Zheng, Kim and Verhaak, 2014). A frameshift mutation is when several nucleotides get deleted or inserted into the DNA of a gene which leads to a sequence being transitioned in an unnatural frame from the position of that mutation until the end of the gene (Old, 2013).

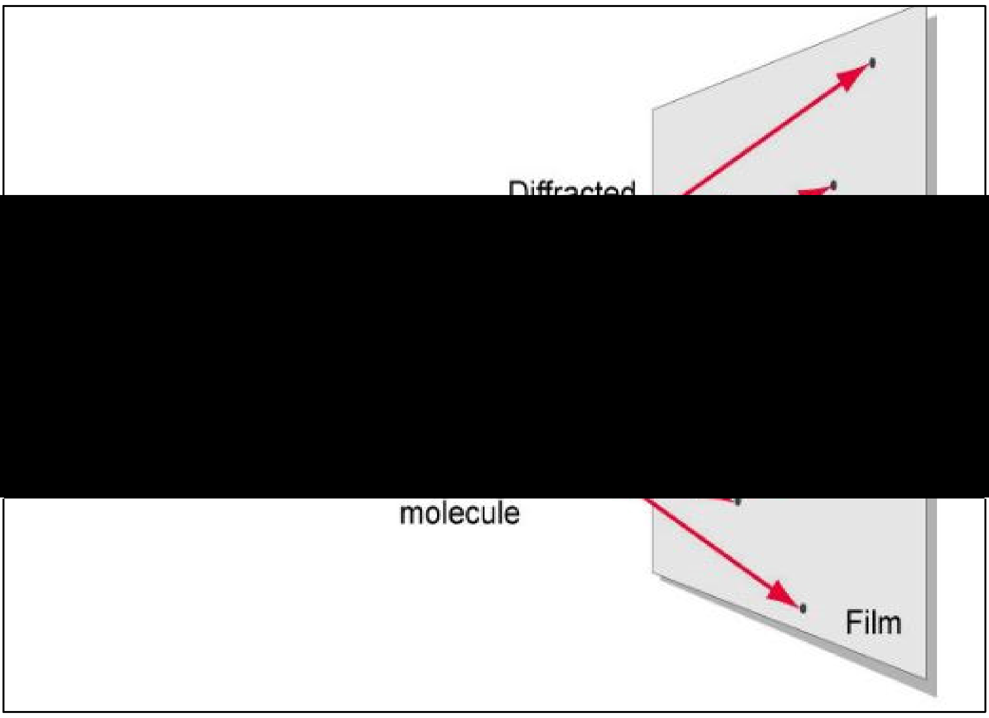
1.6 Structural modelling

Protein modelling is an approach used to convert a 1-Dimensional string amino acid sequence into a 3-Dimensional structural model that can be computationally displayed and processed further. There are many different methods used to determine protein structure, including experimental and computational methods. These methods are X-Ray Crystallography (Experimental), NMR Spectroscopy (Experimental), Cryo-EM (Experimental), Homology Modelling (Computational), Fold Recognition (Computational), and Ab Initio Modelling (Computational). The results of the experimental methods inform the computational methods in further prediction of the 3-D structure.

1.6.1 X-Ray Crystallography

X-ray crystallography is the most commonly used method for the determination of the 3D structure of a protein (Zheng et al., 2015). With the availability of a protein structure, it is possible to obtain a more in-depth understanding of its function.

The purpose of this method is to try and gather knowledge of a 3D structure from a crystal. A high concentration sample is taken then crystallized. This crystal is then exposed to an X-ray beam which generates a diffraction pattern. A diffraction pattern is an array of reflections of the diffracted x-ray beam that are collected by x-ray crystallography. These patterns are processed to gather yield information about the packing symmetry of the crystal and then the size of the repeating unit which forms that crystal. This information is gathered from the diffraction spot patterns that are created by the X-rays. Using the intensity of the spots, the structural characteristics can be determined and help generate an electron density map. If the electron density map is of sufficient quality, it is then possible to allow a molecular structure to be built. There are many different methods that can be used to enhance the quality of the electron density map at this stage. The structure that is then created is then refined so that it fits the electron density map to form a more thermodynamically favoured conformation (Smyth and Martin, 2000). Figure 1.6 summarises the process of X-ray crystallography and shows the protein sample is subjected to an X-ray beam. The X-ray beam is then diffracted into a pattern that is determined by the regular crystal lattice structure of the protein. The diffraction data that is collected is then inputted into a computer which can determine the 3D coordinates of all the residues present based on the electron densities that diffract the X-Ray beam. All this data is used to assemble the full 3D structure of the protein. Figure 1.7 shows a graph of how X-ray crystallography has increased its influence in the bioinformatics field.



[redacted]

Figure 1.6: X-Ray Crystallography process – shown is an Image of the process undertaken by X-Ray Crystallography (Available at: http://www.chem.ucla.edu/~harding/ec_tutorials/tutorial73.pdf [Accessed 4 May. 2018])

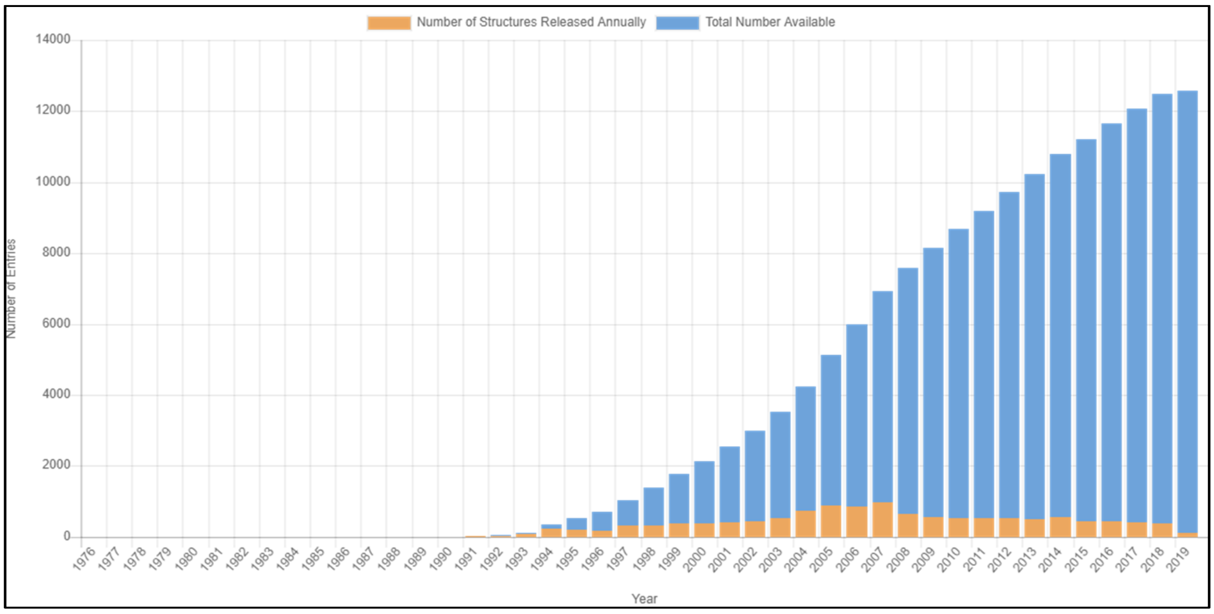


Figure 1.7: Graph shown the increasing number of structures being uploaded to the Protein Data Bank through X-Ray Crystallography. (Available at: https://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=explMethod-xray&seqid=100 [Accessed 15 Jun. 2019])

1.6.2 Nuclear Magnetic Resonance (NMR) Spectroscopy

Nuclear magnetic resonance (NMR) was first developed by physicists Felix Bloch and Edward Purcell in 1946. Nuclear magnetic resonance is another experimental approach to determine a protein structure from a solution of a protein. NMR spectroscopy has become an essential technique in helping with structure-based drug design (Pellecchia, Sem and Wüthrich, 2002). Unlike X-ray crystallography, this process does not use crystals. Instead, NMR is applied to a solution of the molecule. NMR uses the magnetic properties of specific atomic nuclei to gather information such as structure, dynamics, reaction state, and chemical environment of the molecule. The intramolecular magnetic field of an atom changes its resonance frequency. Due to this, it is possible to determine the electronic structure of a molecule and its functional groups.

NMR is also known for how it can be used as a valuable screening tool for ligand-protein interactions (Maity, Gundampati and Suresh Kumar, 2019). This process can detect and quantify interactions with high accuracy without the need for protein function information. It can also be used to collect information about the structure of a protein or ligand, which helps determine whether the binding is weak or strong. Within a magnetic field, this can cause some magnetic nuclei to accept a state of nuclear spin of different energies and radio-frequency radiation can cause the nuclei to go between these states. The chemical environment of specific magnetic nuclei provides its NMR properties (Fan and Lane, 2016). These include properties such as its resonance frequency. Due to this, NMR is now an essential tool for chemistry, the life sciences and medical diagnostics. This has also become a fundamental method in determining the 3-Dimensional structures of macromolecules. (Pellecchia, Sem and Wüthrich, 2002). NMR has contributed to the increasing accuracy of homology protein modelling and fold recognition modelling. There are currently 12675 NMR structures that have been deposited in the Protein Data Bank (PDB) (rcsb.org, 2019). Figure 1.8 shows a graph of how NMR has increased its influence in the bioinformatics field by the number of structures deposited since its invention. But NMR is now on a relative decline due to the higher costs of the approach.

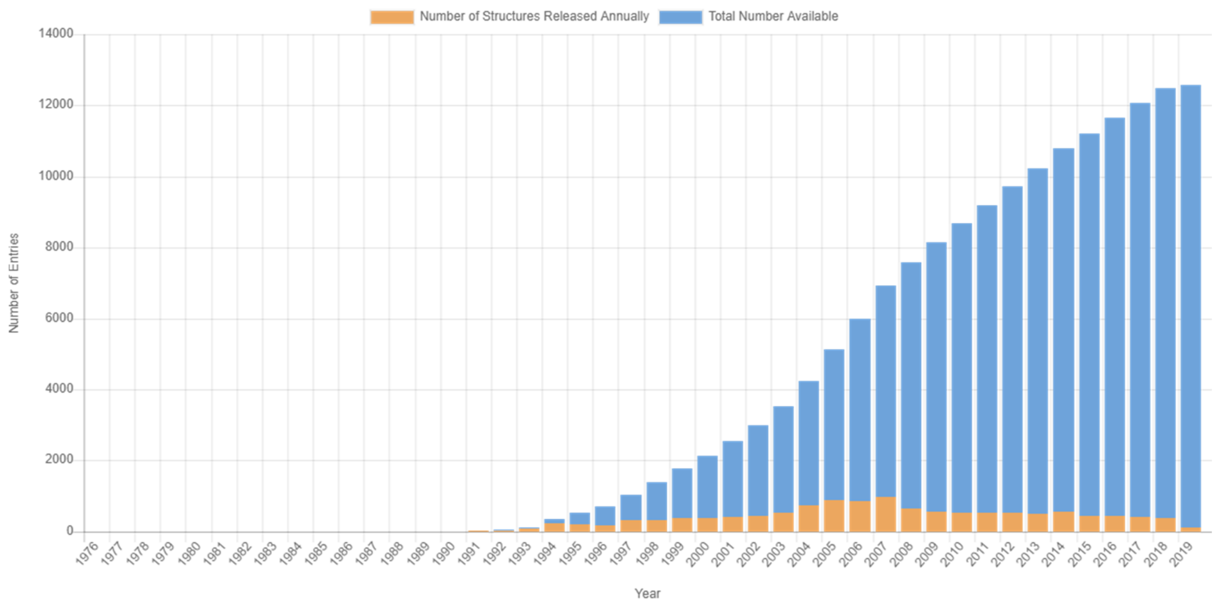


Figure 1.8: Graph shown the increasing number of structures being uploaded to the Protein Data Bank through NMR (Available at: https://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=explMethod-nmr&seqid=100 [Accessed 15 Jun. 2019])

1.6.3 Cryo-electron Microscopy (Cryo-EM)

Cryo-electron microscopy (Cryo-EM) is now one of the most widely used tools in structural biology (Merk et al., 2016). The basis of the Cryo-EM methodology is to create images of radiation-sensitive specimens in an electron microscope under cryogenic conditions. Cryo-EM is now used to image a wide range of biological structures, such as intact tissue sections, all the way down to individual bacteria and protein molecules. The methodologies all use parts of Cryo-EM to help analyse biological structures in different ways such as Cryo-electron tomography, Single-particle cryo-electron microscopy, and Electron crystallography. Cryo-EM's method involves taking multiple 2D projection images and combining them to determine the 3D structure. Shown in figure 1.9 is the process undertaken for Cryo-electron Microscopy. A purified sample of a macromolecule complex is frozen to the same temperature as liquid nitrogen. This helps protect and preserve the sample from the electrons. Cryo-EM uses a low number of electrons so that it does not cause any damage to the

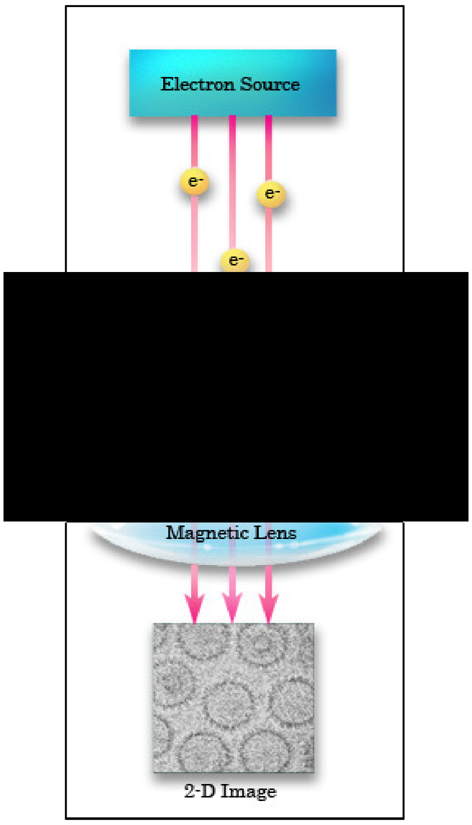


Figure 1.9: Cryo-electron Microscopy process - Shown is the process undertaken for Cryo-electron Microscopy. (Available at: http://www.eicn.ucla.edu/cryoem [Accessed 10 Jun 2018])

sample. The electrons are then directed at the sample and will either pass through small gaps in the sample or be refracted from denser areas. Magnetic coils are used to magnify and focus the electrons. A 2D image is then produced. These 2D images are then used to create a 3D model of the macromolecule. Computers are used to data merge the 2D images to create this. (Carroni and Saibil, 2016) These images, as with structures from NMR and X-Ray crystallography, are uploaded to the Protein Data Bank (PDB) (Berman, 2000). Cryo-EM is increasing its influence in the bioinformatics field by leading to the deposition of more significant numbers of structures to the PDB. Shown in figure 1.10 is the amount of structures Cryo-EM has added to the Protein Data Bank.

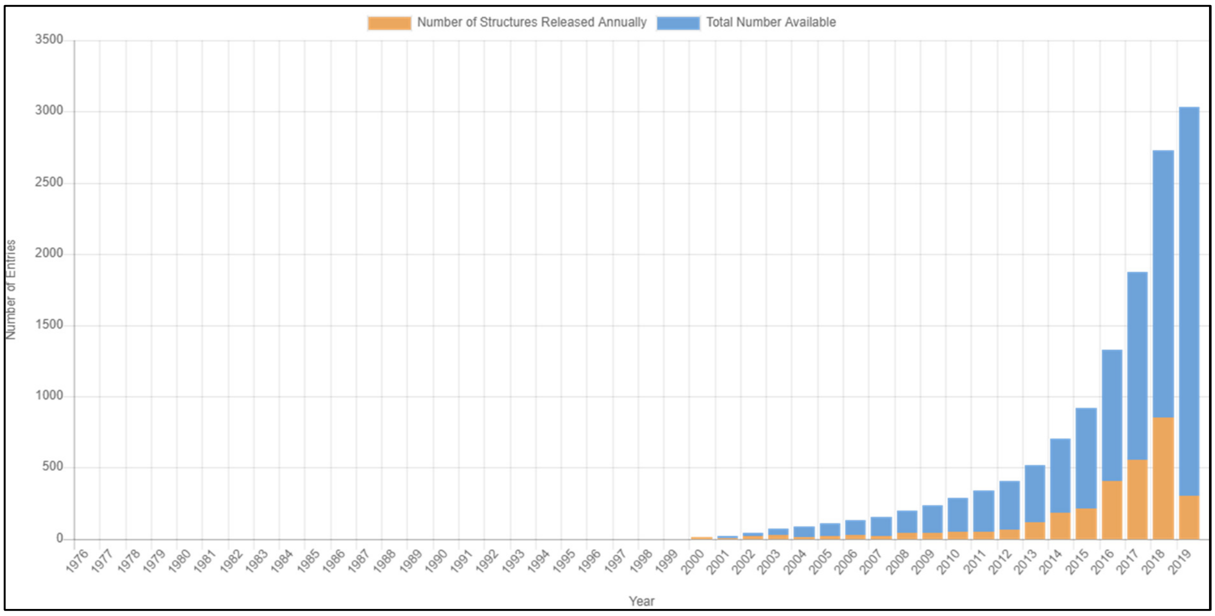
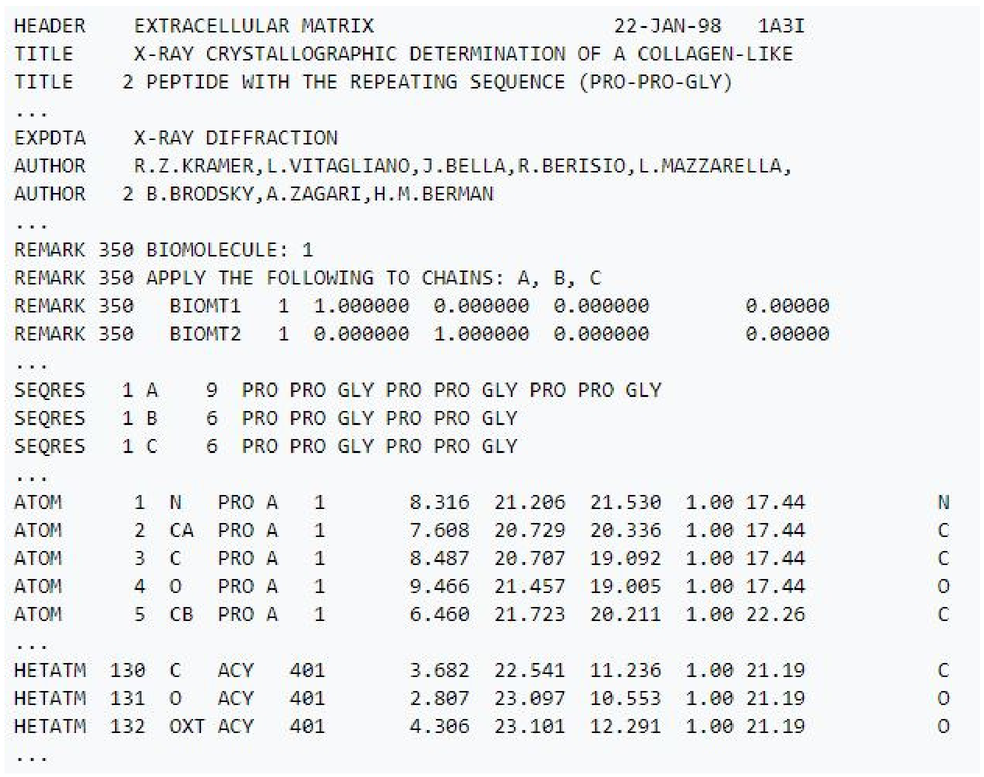


Figure 1.10: Graph shown the increasing number of structures being uploaded to the Protein Data Bank through Cryo-electron Microscopy (Available at: https://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=explMethod-em&seqid=100 [Accessed 15 Jun. 2019])

1.6.4 Homology Modelling

Homology modelling, which is also known as comparative modelling, is a computational method for modelling proteins from their amino acid sequence. Of all the computational methods used to generate 3D models of proteins from sequences, homology modelling is currently the most accurate (Muhammed and Aki-Yalcin, 2018) but protein threading is catching up quickly.

Homology modelling can predict the 3D structure of a protein from its amino acid sequence. This predicted structure is based on known structures of homologous proteins (templates). These determined structures have been produced by using X-Ray crystallography and NMR spectroscopy as discussed previously in the chapter, following their conversion to Protein Data Bank (PDB) format. The PDB file format is a computer-based archival file for macromolecular structures (Bernstein et al., 1978). This file contains atomic coordinates and partial bonds between atoms that have been gathered from the output of X-Ray crystallography/NMR. Figure 1.11 shows an example of the types of data contained and how it is formatted within the PDB file. Shown in table 1.3 is the stored data



```
HEADER    EXTRACELLULAR MATRIX                    22-JAN-98   1A3I
TITLE     X-RAY CRYSTALLOGRAPHIC DETERMINATION OF A COLLAGEN-LIKE
TITLE    2 PEPTIDE WITH THE REPEATING SEQUENCE (PRO-PRO-GLY)
...
EXPDTA    X-RAY DIFFRACTION
AUTHOR    R.Z.KRAMER,L.VITAGLIANO,J.BELLA,R.BERISIO,L.MAZZARELLA,
AUTHOR   2 B.BRODSKY,A.ZAGARI,H.M.BERMAN
...
REMARK 350 BIOMOLECULE: 1
REMARK 350 APPLY THE FOLLOWING TO CHAINS: A, B, C
REMARK 350   BIOMT1   1  1.000000  0.000000  0.000000        0.00000
REMARK 350   BIOMT2   1  0.000000  1.000000  0.000000        0.00000
...
SEQRES   1 A    9  PRO PRO GLY PRO PRO GLY PRO PRO GLY
SEQRES   1 B    6  PRO PRO GLY PRO PRO GLY
SEQRES   1 C    6  PRO PRO GLY PRO PRO GLY
...
ATOM      1  N   PRO A   1       8.316  21.206  21.530  1.00 17.44           N
ATOM      2  CA  PRO A   1       7.608  20.729  20.336  1.00 17.44           C
ATOM      3  C   PRO A   1       8.487  20.707  19.092  1.00 17.44           C
ATOM      4  O   PRO A   1       9.466  21.457  19.005  1.00 17.44           O
ATOM      5  CB  PRO A   1       6.460  21.723  20.211  1.00 22.26           C
...
HETATM  130  C   ACY   401       3.682  22.541  11.236  1.00 21.19           C
HETATM  131  O   ACY   401       2.807  23.097  10.553  1.00 21.19           O
HETATM  132  OXT ACY   401       4.306  23.101  12.291  1.00 21.19           O
...
```

Figure 1.11: Example of a Protein Data Bank File (PDB file) – shown is an example of how a protein PDB file looks and how the data is represented within it.

Table 1.3: Data which is stored in a PDB file – shown are all the data tags which are used within a PDB file and the definition of those tags

Tag	Data
(HEADER, TITLE, AUTHOR)	holds data such as the name of the protein, the file creator's name.
REMARK	this tag is for information about the experiment that was conducted to create the data, for example, the resolution of the camera used for the X-ray crystallography experiments. Also, if the file has been run through any programs, for example, DSSP, OBABEL.
SEQRES	This tag is used to input the amino acid sequence corresponding to the protein structure.
ATOM	Used to store the atom coordinates of the structure.
HETATM	This is used to store coordinates of solvents, ligands, and cofactors.

within the PDB file. These files can be visualized by molecular visualization software such as UCSF Chimera (Pettersen et al., 2004). All these files that have been generated from X-Ray Crystallography/NMR are all stored in the RCSB Protein Data Bank (Burley et al., 2018), which is accessible online. Currently (September 2018) there are approximately 153601 biological macromolecular structures stored on the database. This has increased over the last decade, and when this project first started (2014), there were 104866 biological macromolecular structures in the database. With homology modelling, a protein sequence with 30% identity or more is now able to be reliably modelled to a known structure and can often be predicted with an accuracy equivalent to a low-resolution X-ray structure (Lam et al., 2017). The pipeline that is used consists of four stages these being Identify the homologue of known structure, Align the query sequence to template structure (BLAST) (Altschul et al., 1990), Build the model based on the alignment (T-COFFEE) (Notredame, Higgins and Heringa, 2000), and Assess and refine the model.

For a protein that has 40% or more sequence identity, the alignment process is straightforward. In the model, there will be hardly any gaps, and 90% of main-chain atoms can be modelled with an RMSD (root-mean-square distance) error of about 1 Å (Mohammadi et al., 2016) . About 57% of all known sequences have at least one domain that is related to at least one PDB protein template, and this is continuously improving year by year. Figure 1.12 shows a flowchart for the Homology Modelling pipeline. It all begins with an amino acid sequence. The first stage of the process is to use the sequence to detect and select homologues that have known 3D structures (templates) that are similar to it. Developments of the alignment program BLAST (Basic Local Alignment Search Tool) (Altschul et al., 1990) are used to search for these templates.

1.6.4.1 Basic Local Alignment Search Tool (BLAST)

BLAST first takes the sequence that has been inputted and starts comparing it to all the sequences that are stored within the non-redundant database, which is provided by NCBI (National Centre for Biotechnology Information). This database contains non-redundant sequences from GenBank (Benson et al., 2012) translations and other databases such as RefSeq (Pruitt, Tatusova and Maglott, 2007), PDB (Burley et al., 2018), SwissProt (Bairoch, 2000), PIR (Protein Information Resource) (Barker, 2000).

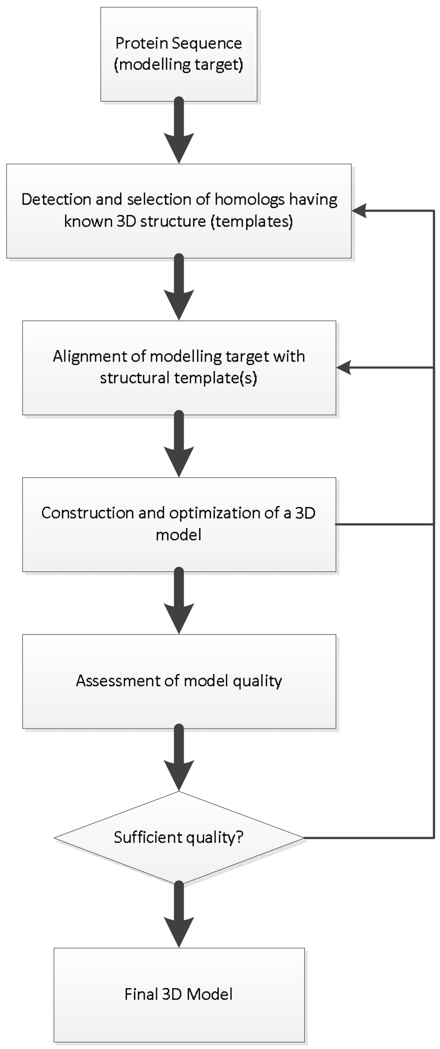

Protein Sequence (modelling target)
Detection and selection of homologs having known 3D structure (templates)
Alignment of modelling target with structural template(s)
Construction and optimization of a 3D model
Assessment of model quality
Sufficient quality?
Final 3D Model

Figure 1.12: Homology modelling flowchart – shown is a flowchart of the process used for homology modelling. Homology modelling consists of 7 steps: entering a query sequence, Detection and selection of homologs, Alignment of the query sequence with templates, construction of the 3D model, and quality assessment.

The BLAST process is shown in a flowchart in figure 1.13. BLAST considers the input sequence and searches the Non-Redundant (NR) database for matching sequences. It does this by listing “words” which are short sections of the amino acid sequence. This is called the k-letter. For example, if the k-letter = 4 BLAST will list every 4 amino acid sequence from the beginning of the input sequence until it reaches the end of the sequence. For each “word” BLAST creates every possible 3 letter “words” combination and compares them. BLAST then gives a score for each “word”. It does this by using a scoring matrix to score each residue pairing. The scoring matrix used is the BLOSUM62 weighting scheme (Styczynski et al., 2008) which is shown in figure 1-14. For example, if GEF is compared to GEQ and GAF, the respective scores would be 8 and 11. BLAST is then only concerned with the high scoring comparisons. This is all decided by the threshold *T*. For example if threshold *T* = 10 then the “word” GAF would be added to the list but GEQ would be discarded. The remaining words are then organised into an efficient search tree. This helps the program to be able to search the database

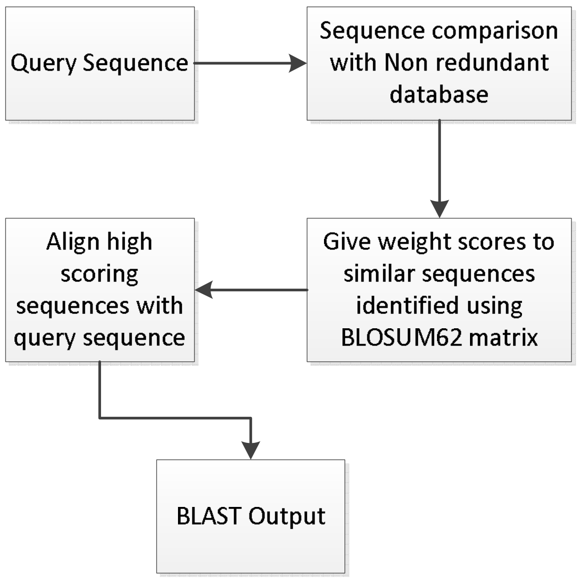

Query Sequence
Sequence comparison with Non redundant database
Give weight scores to similar sequences identified using BLOSUM62 matrix
Align high scoring sequences with query sequence
BLAST Output

Figure 1.13: BLAST process flowchart – shown is a flowchart describing the BLAST process. BLAST consists of 5 steps: enter a query sequence, compare the query sequence with the Non-redundant database, calculate a weight score for all identified sequences, align top-scoring sequences with the query sequence, and output a result.

quickly with the high scoring “words”. These two stages are repeated for all the *k-letters* in the input sequence. When the list has been compiled of all the high scoring “words”, BLAST then searches the NR database for exact matches. When a match is found, the two matching “words” are then aligned and seeded for a potential un-gapped alignment between the input sequence and the database sequence. This means that the two sequences are aligned at the point of the matching “word” and one amino at a time BLAST looks at each amino every side to see if there is a match for more than the 3-amino acid “word”. This is shown in figure 1.15. To determine when the seeding should stop on the sequence, high-scoring segment pairs (HSP) are used. By using the BLOSUM62 weighting scheme matrix, scores are used to compare each amino either side of the exact match. The seeding completes when the score turns negative. The new HSP’s with a higher score than the empirically determined cut off score are then listed. With the new list of HSP’s, BLAST then analyses the

ala	4																			
arg	-1	5																		
asn	-2	0	6																	
asp	-2	-2	1	6																
cys	0	-3	-3	-3	9															
gln	-1	1	0	0	-3	5														
glu	-1	0	0	2	-4	2	5													
gly	0	-2	0	-1	-3	-2	-2	6												
his	-2	0	1	-1	-3	0	0	-2	8											
ile	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
leu	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
lys	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
met	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
phe	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
pro	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
ser	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
thr	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
trp	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
tyr	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
val	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4
	ala	**arg**	**asn**	**asp**	**cys**	**gln**	**glu**	**gly**	**his**	**ile**	**leu**	**lys**	**met**	**phe**	**pro**	**ser**	**thr**	**trp**	**tyr**	**val**

Figure 1.14: BLOSUM62 Matrix – shown is the BLOSUM matrix used to calculate a weight score for every similar sequence identified by the database search.

statistical significance of each by using the Gumbel extreme value distribution (EVD). Multiple functions are used to determine this. This is to assess and produce a score that represents the un-gapped local alignment of the HSP. This is usually called the E score, E-value, or e-value, which is reported in the BLAST output.

Sometimes there is a need to join two or more HSPs together to create a more extended alignment. This can help provide more information on the relationship between the query sequence and the sequences within the database and increase the accuracy of the alignment. There are two scoring methods that are used, the Poisson method and the sum-of-scores method. The Poisson score is mostly used due to it giving a more significant result compared to the sum-of-score method. the sum-of-scores method is when the scores of the two or more HSPs that have been joined are added together. The sum-of-scores method prefers choosing the HSPs that have the highest score, whereas

the Poisson method uses equations to work out, which is the best HSP. BLAST creates a file that contains all the highest scoring HSP alignments with the respective scores. The results of BLAST are then forwarded to an alignment program called T-Coffee.

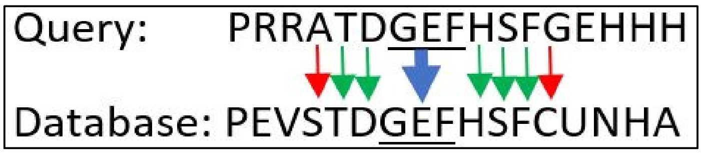


Figure 1.15 : Example of aligned seeding – shown is an example of how aligned seeding works.

Tree-based Consistency Objective Function for Alignment Evaluation (T-Coffee)

T-Coffee is a multiple sequence alignment (MSA) program that is used to generate multiple alignments of sequences using heterogeneous data sources (Notredame, Higgins and Heringa, 2000). MSA is a method used in protein structural analysis. MSA is a tool that is used to compare three or more sequences together to see if there is any relationship between them, such as a shared function, structural or evolutionary. For the homology modelling process, this is a pivotal step to pick the best-aligned templates of the BLAST results to be used in the modelling stage.

T-COFFEE's process is shown in figure 1.16. The input query of multiple sequences into T-COFFEE is run through two different pairwise alignment tools called ClustalW (Larkin et al., 2007) and Lalign (ebi.ac.uk, 2017). Lalign is used to do a local pairwise alignment, whereas ClustalW is responsible for the global pairwise alignment. Local and global pairwise alignments are two different techniques of aligning sequences. Local pairwise alignments compare a small portion of the query sequence against the other sequences within the file provided by BLAST, whereas for the global pairwise alignment, the entire query sequence is compared to the sequences in the BLAST file. This means that global pairwise alignments will produce more gaps between the query and the subject sequences than the local pairwise alignment. The results of ClustalW and Lalign are then put through a weighting process which will assign a weighted score to each pair of aligned sequences. The best weighted pair-wise alignments of each method are added to a primary library for each method.

The primary library of the best ClustalW weight scores and the primary library for the Lalign weighted scores are then combined. To do this efficiently, if the same pair of aligned sequences appear in both the ClustalW and Lalign primary libraries, these are then combined into one single record. The weighted scores for each are then also combined using addition. If there is no duplicate, a new record is created for the single pair-wise alignment which is being considered, using the weighted score that was calculated. Pairs that do not occur at all are not used. By doing this, the value of the information is increased due to examining the consistency of each of the pair of residues against every pair of residues alignment in the library (Notredame, Higgins and Heringa, 2000). This process is called library extension.

The next stage is progressive alignment where firstly the pair-wise alignments are used to create a distance matrix which is then used to create a phylogenetic guide tree. The phylogenetic guide tree is used to help group the sequences for the multiple sequence alignment. Dynamic programming is applied to the two closest sequences, which are in the tree matrix to align them. This uses the extended library weight scores to do the alignment. These sequences are then fixed, then the next two sequences are aligned, or the next sequence is aligned with the first two sequences depending

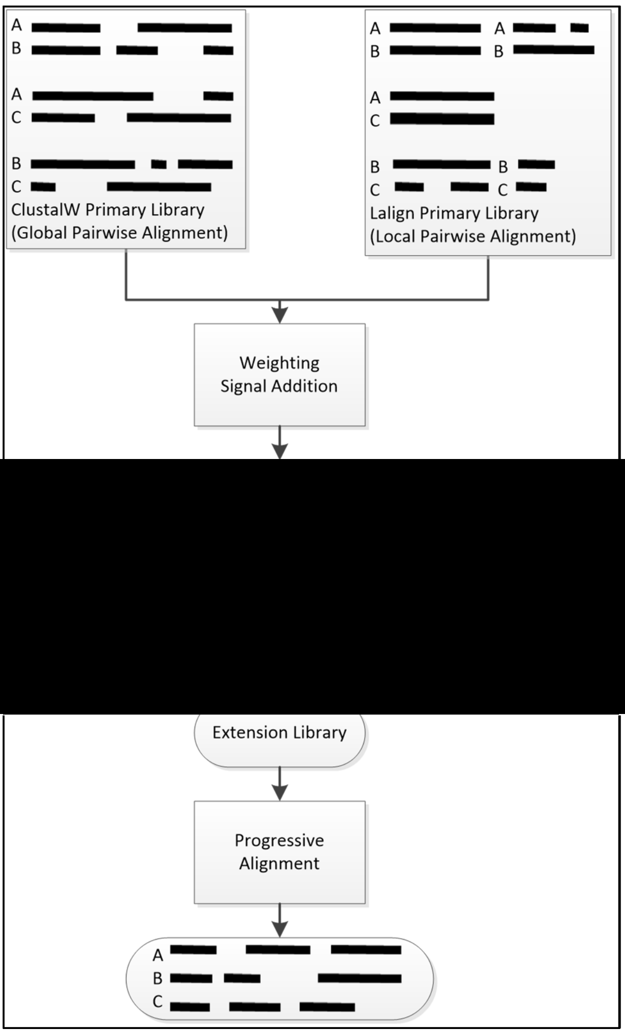


Figure 1.16: T-Coffee flowchart (adapted from Notredame, Higgins and Heringa, 2000)

on what the phylogenetic guide tree suggests. This continues until all sequences have been aligned. These groups of aligned sequences are then aligned against each other where the average of the extensive library weighted scores are used.

For the homology modelling pipeline, there is a mode within T-COFFEE called Expresso that is used. With the program Expresso (Armougom et al., 2006), it takes the sequences that BLAST has collected of the templates that are like the query sequence and reruns BLAST to download the 3D structures of the PDB templates like those in the BLAST results. These are then assigned to their corresponding sequence. 3DCOFFEE (Poirot et al., 2004) within Expresso uses computational methods like standard T-COFFEE it either a sequence-based alignment using Lalign, or a structural alignment using SAP. Structural based alignment is used for the Swansea Bioinformatics groups homology modelling pipeline. Firstly, a structure-based alignment is done using the templates; then the sequences are aligned with their corresponding structure. These are then all added to the library. Figure 1.17 shows a flowchart of this process. When the library has been completed, the standard T-COFFEE multiple sequence alignment algorithms are run.

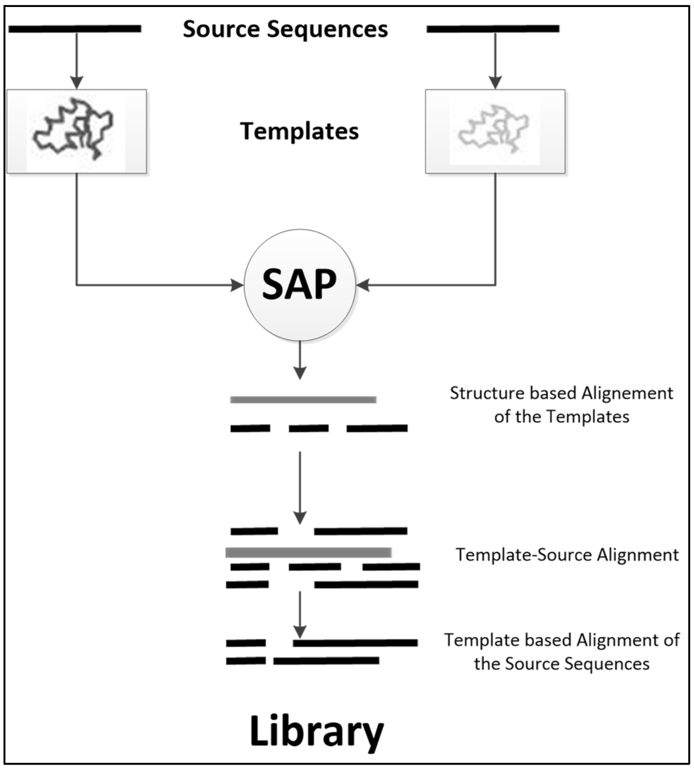

Source Sequences
Templates
SAP
Structure based Alignement of the Templates
Template-Source Alignment
Template based Alignment of the Source Sequences
Library

Figure 1.17: Flowchart of 3D COFFEE process. (Adapted from Armougom et al., 2006)

Modeller

Modeller (Webb and Sali, 2016) is the final stage of the homology modelling process. This stage is where the 3D models for the query sequences are generated. Modeller takes the T-COFFEE results file with all the aligned sequences and begins to go through the file in “chunks”. Modeller works through the file in vertical slices and considers only a few amino acids at a time. For each “chunk” of the T-COFFEE results that Modeller looks at, it assigns a slice of each template that is associated with the aligned sequences. It then looks at these segments of structure and searches these for consistency. Modeller then “wraps” the segments around the 3D structure and compares them to each other. Where MODELLER ((Mohammadi et al., 2016; Webb and Sali, 2016) finds similarities between the structures, it takes the similar segment 3D coordinates and stores them for the modelling stage along with the associated part of the sequence. It does this until the whole T-COFFEE file has been read.

For the modelling stage, when this file is complete MODELLER creates a 3D models automatically using a class called *automodel*. Modeller will produce 5 models that are based on the file that was produced with the 3D coordinates. Due to multiple models being created, MODELLER produces scores to provide information for which model is the best. There are many different scores that MODELLER can use so that the user can pick the best model according to the particular scoring that they want. MODELLER produces its own assessment score called the modeller objective function.

This also includes the DOPE (Shen and Sali, 2006) and SOAP assessment scores. These scores are ranked based on the alignments provided and is only an approximate measure.

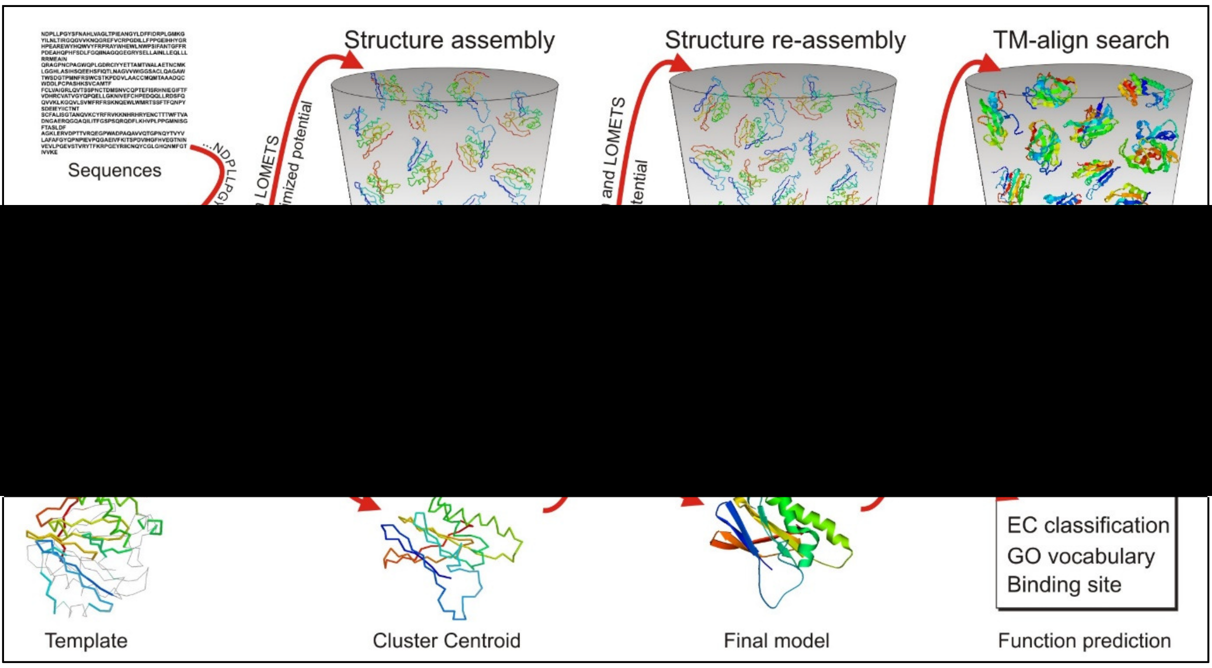
1.6.5 Protein Threading

Protein threading is another method of modelling a 3D protein structure. Also known as fold recognition, protein threading models amino acid sequences by identifying proteins that have same/similar folds to that of proteins with known structure. This method is used for proteins that cannot be modelled by homology modelling due to having low or no homologous coverage within the Protein Data Bank (PDB) (Burley et al., 2018) . Homology modelling has always been classed as the best modelling method between it and protein threading, but threading has continually improved over the last decade or so. With its ability to be able to model proteins with low homology and the steadily increasing quality of the structures it produces, threading is helping to narrow the gap between proteins that have known structure to those that have experimentally characterized structures. There are many protein threading suites available to model protein sequences in this way. A few examples are I-TASSER (Iterative Threading ASSEmbly Refinement) (Roy, Kucukural and Zhang, 2010), HHpred (Soding, Biegert and Lupas, 2005), and RAPTOR (XU et al., 2003)

For this project, I-Tasser was used. The reason why this program has been chosen is explained in the methods chapter. The following description of the threading method is based on I-TASSER. There are four components to protein threading approaches being Threading, Structural Assembly, Model selection and refinement, and Structure-based functional annotation. The flowchart shown in figure 1.18 summarises the method that I-TASSER uses.

Threading

"Threading refers to a bioinformatics procedure for identifying template proteins from solved structure databases that have a similar structure or similar structural motif as the query protein sequence." (Roy, Kucukural and Zhang, 2010) I-TASSER uses BLAST to identify evolutionary relatives. This is used to create a sequence profile that is then used to predict a secondary structure (using PSIPRED (Buchan et al., 2013)). When the profile of the secondary structure has been predicted, the secondary structure is threaded using LOMETS (Wu and Zhang, 2007) through a PDB structure library. LOMETS is a meta-threading server which uses multiple state-of-the-art threading programs which then find suitable templates. The threading programs used in LOMETS are FFAS-3D (Xu et al., 2013), HHsearch (Fidler et al., 2016), MUSTER (Wu and Zhang, 2008), pGenTHREADER (Lobley, Sadowski and Jones, 2009), PROSPECT2 (Kim et al., 2003), SP3 (Zhou and Skolnick, 2009), and SPARKS-X (Yang et al., 2011).



[redacted]

Figure 1.18: the I-Tasser process flowchart. (Roy, Kucukural, and Zhang 2010)

LOMETS then takes the templates and alignment generated by the 9 threading programs and assigns a Z-score. The Z-score is used to evaluate the significance of the alignments for a given template compared to the other templates (Yang et al., 2015). To determine if the templates are good or bad, each program has its own "cut-off point" Z-score. LOMETS stores the top 20 templates of each threading program, listed in order of how good the alignments are. I-TASSER uses this list to create spatial restraints and to help construct initial conformations that the modelling simulations will begin with.

Structural assembly

In the structural assembly stage, fragments within the threading alignments are used to build a structural conformation of all the sections that aligned well, combined with the unaligned regions that were generated by ab initio modelling. Ab initio modelling is a method which uses designed energy functions to conduct a conformational search and leads to a number of possible conformations being generated. I-TASSER uses a reduced model for the protein's chain. These models have residues which are represented by their Cα and side-chain centre of mass. The regions that are not aligned during the threading process are due to having a low modelling accuracy, in these regions the structure modelling is confined to a lattice system of grid 0.87 Å, which means that the entropy of conformation search can be reduced (Roy, Kucukural and Zhang, 2010).

Monte Carlo Simulation is used to assemble the fragments. The Monte Carlo simulation reassembles the PDB fragments that have been collected for the aligned regions. For proteins that do not have any template associated with it, the models are entirely created by ab initio modelling. The simulation is guided by “composite knowledge-based force field” (Roy, Kucukural and Zhang, 2010). This contains statistical terms that are stored within the PDB, spatial restraints from threading templates, and sequence-based contact predictions from SVMSEQ (Wu and Zhang, 2008). The conformations that have been generated in the refinement simulation are then clustered by a program called SPICKER. This is done so that the low-energy states can be identified. These are then put into a cluster centroid which stores the average 3D coordinates of all the structural decoys.

Model selection and refinement

The first part of this stage is the fragment assembly simulation stage and is performed by again using the selected cluster centroids. The second run of this stage is that steric clashes are removed and refines the global topology. Template Model align (TM-align) is used to pick out threading alignments and structures that are the closest to the cluster centroids. These results are then clustered. The structures that have produced the lowest energy scores are inputted into a program called REMO. REMO creates the final structures of the proteins which are built by using all atoms. These all-atom models are built using Cα traces through the optimization of hydrogen bonding networks.

Structure-based functional annotation

This last stage is where the final models are generated. The 3D models are structurally compared to protein templates from information within their PDB files. These protein templates that they are being compared are annotated by 3 structure/function databases and are represented as:

- 5,798 non-redundant entries with known EC numbers
- 26,045 non-redundant entries with known GO terms
- 19,658 non-redundant entries with known ligand binding sites

The global structural search is used to see if there are similar folds, while the local similarity search allows it to compare functions and active/binding sites. The functional analogues are ranked by using the Template Model Score (TM-score), RMSD, and sequence identity, coverage of structural alignment. The results are then ranked on how accurate each model is with a C-score. I-TASSER produces 5 PDB models for the user to choose from.

1.7 Virtual Drug Screening

In the pharmaceutical industry, the method that has widely been adopted to help the early stages of drug discovery is *in vitro* high-throughput screening (HTS) (Cronk, 2013). This method has been used to discover new targets and to test newly designed drugs on specific targets. Computational or *in silico* methods are being developed to reduce the vast chemical search space to increase the probability of identifying active ligands without prior knowledge of receptor structure (Cheng et al., 2012) One of these methods that have been developed is virtual drug screening. Due to the increase of knowledge of protein structure and being able to model/predict the structure of a protein target due to the advances of X-ray crystallography and NMR, it is possible to compute how a drug might interact with a specific target protein. This method is a cheaper and faster method than HTS.

All docking-based virtual screening approaches employ the same broad workflow, this being shown in figure 1.19. The first step is acquiring the protein or target of interest, and then to compile a library of the compounds/ligands that are relevant to the research being conducted. The structure files for these are then prepared and then docked. The docking algorithms used then produce a score that will represent the interaction between the protein of interest and the ligand that has

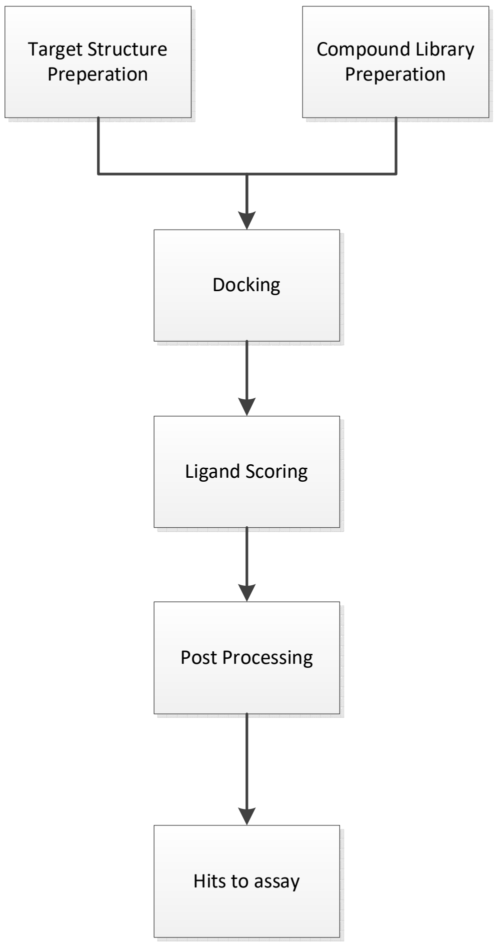


Figure 1.19: Docking-based virtual screening workflow – shown is the basic workflow to perform a virtual screening docking.

been docked. It also produces a conformation file so that it is possible to view these predicted dockings in 3D. Then these conformations and scores are analysed to see if the predicted dockings will be classified as predicted interactions or Hits. Many different docking algorithms are being used for virtual screening. Shown in figure 1.20 are all known docking programs available and their percentages of publications they have been involved in. If all the docking programs, the most popular are AutoDock, Gold, and Glide. These programs make it possible to predict whether there is a possible interaction between the protein of interest and any of the compounds/ligands in the compiled list. This is also more cost-effective than X-Ray Crystallography and NMR. These are both expensive and lengthy processes which can only be done one at a time whereas, for virtual docking, hundreds of protein-ligand dockings can take place within the same timeframe and for a fraction of the cost. However, docking predictions are only just that – predictive simulations. X-ray crystallography and NMR provide details of known actual structures. This means there is a pay-off between cost and convenience on the one hand and the introduction of error on the other. In this project, the chosen docking programs to use for the docking process are AutoDock Vina and DOCK 6.0 due to them being freely available and are deemed to be accurate.

1.7.1 AutoDock Vina

AutoDock Vina (Trott and Olson, 2009) is an open-source ligand-receptor docking program. It is possible for the program to predict how a ligand would interact/bind to a receptor/protein. This program is a docking algorithm that has been improved on its predecessor, AutoDock 4 (Morris et al., 2009). AutoDock Vina has increased the speed of docking, and the accuracy of binding predictions compared with AutoDock 4 (Trott and Olson, 2009). With the ability to now parallelise docking jobs it has been made possible to dock multiple protein-ligand pairs at the same time. AutoDock Vina then produces an affinity score which is associated with the calculated conformation and represents the global strength of an interaction.

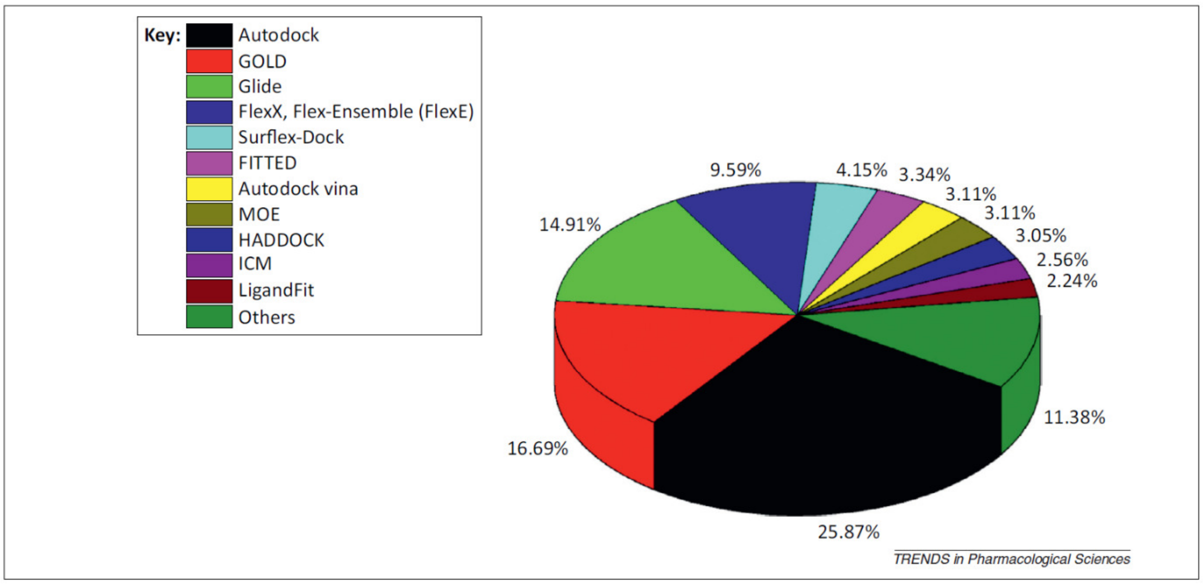
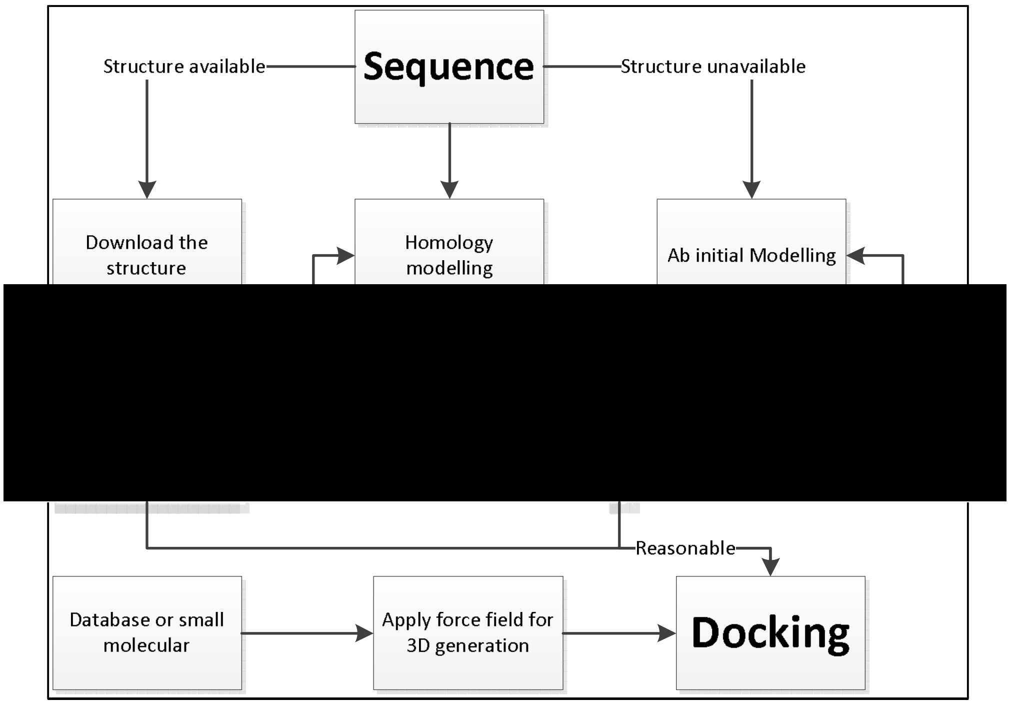


Figure 1.20 : All docking publications from 1990 to 2013 (Chen, 2015) – shown is a pie chart of all known virtual docking programs and the percentage of papers which have referenced these programs between 1990 to 2013.

Figure 1.21 shows a flowchart of the process needed to perform an AutoDock Vina's docking. The file format for structural models used in AutoDock Vina is PDBQT. These can be based on structures that are stored in the Protein Data Bank database or models that have been created using homology modelling or protein threading methods. AutoDock Vina uses a box search space for the receptor. The search space box can be set as the entire surface of the inputted receptor, or the box size and location can be determined by the user. The ligands that are required for docking have their PDB files downloaded and converted to PDBQT file format. These can be ligand files that have been downloaded or ligand PDB files that have been generated. Once the files for both the receptor and the ligands have been acquired, they are then docked using the AutoDock Vina's algorithm. AutoDock then outputs a top conformation accompanied by an affinity score which can give an idea of whether interaction has occurred. These conformation files can then be visualised using programs such as Chimera which can be used to analyse the docking and examine it in more depth, investigating such as the closest atoms between the ligand and the receptor.



[redacted]

Figure 1.21: Protein-Ligand docking process for AutoDock Vina (Adapted from Chen, 2015)

1.7.2 UCSF DOCK 6.0

UCSF DOCK 6.0 (Allen et al., 2015) is another open-source ligand-receptor docking program which is the latest development from the original DOCK program. The original DOCK program was created in the 1980s by Irwin Kunt's Group from the University of California, San Francisco. DOCK 6.0 was built as an extension to DOCK 5, which has improved sampling and scoring capabilities and has also improved the optimisation and testing for compatibility with RNA (Allen et al., 2015). DOCK 6.0 has several more scoring functions compared to AutoDock Vina such as Electrostatics Scoring, Hawkins-Cramer-Truhlar GB/SA solvation scoring with optional salt screening, B/SA solvation scoring, AMBER Binding energy scoring, AMBER Receptor Flexibility Scoring, AMBER Scoring, Contact Scoring, and Grid-Based Scoring. There are many more different parameters generated in the predicted docking that can be investigated by using DOCK 6.0 compared with other docking algorithms such as AutoDock Vina. The DOCK 6.0 process is shown in figure 1.22 where firstly the protein file is acquired. DOCK 6.0 uses MOL2 files of the protein and ligand to compute a docking, this being different to AutoDock Vina which mainly uses the PDBQT format. DOCK has two different methods of specifying a search space, these being Sphgen and Grid. For docking, Sphgen is used as the preferred method. This is an inbuilt program in the DOCK 6.0 suite. To determine the search space of the protein, spheres are used which are then used to attempt to dock a ligand within. Sphgen scans the entire 3D structure of the protein and uses multiple spheres to mark possible cavity locations where a ligand can potentially bind. There are three different options in Sphgen that can be used to determine the search space. The first option is to select the largest cluster of spheres that Sphgen has generated for the search space. The second option is to specify a radius from the protein and only use the spheres within it that Sphgen had generated. The final option is for the user to specify the spheres manually in the desired locations. An example of the output produced by Sphgen is shown in figure 1.23. DOCK will then use this as the search area for the docking algorithms, which is different from the box search area AutoDock Vina uses. After the search space has been specified the docking can then take place. The list of ligands that are under investigation is acquired with their

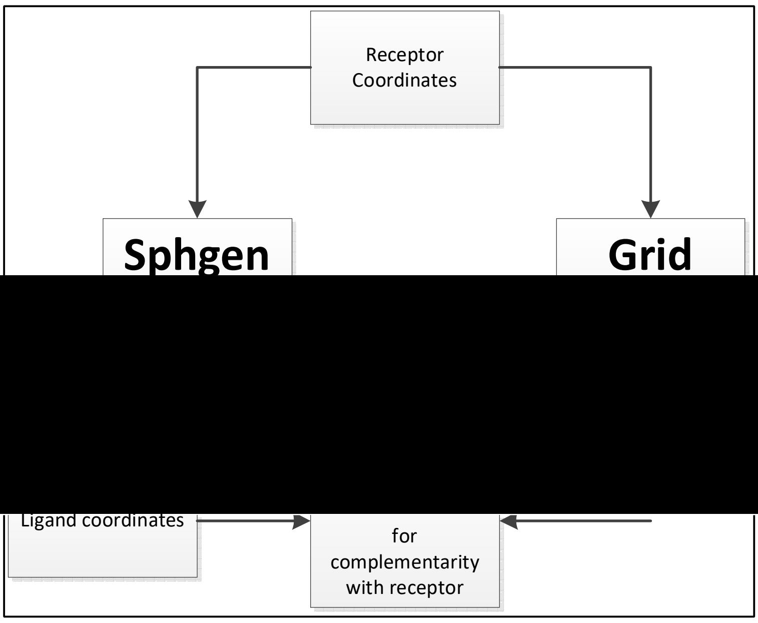


Figure 1.22: DOCK 6.0 process (Adapted from Dock.compbio.ucsf.edu, 2018)

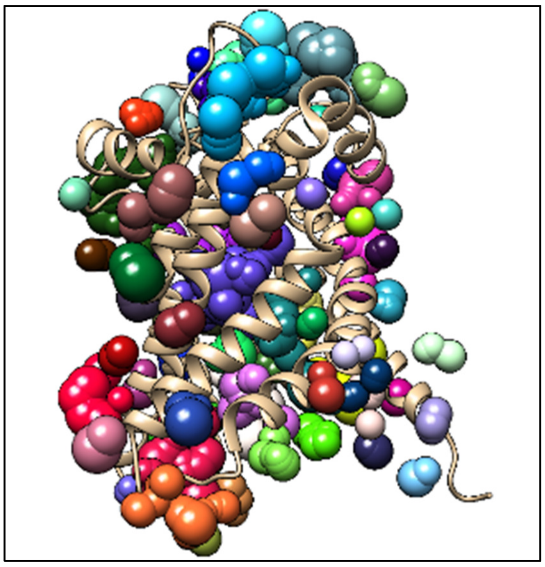


Figure 1.23: Example of Sphgen output which indicates possible locations for drugs to be docked.

corresponding MOL2 files which are then used for the docking process. DOCK 6.0 will then attempt to dock this list of ligands into the spheres that have been selected for the receptor using its docking algorithm. DOCK 6.0 outputs the top 10 conformations calculated for a docking pair accompanied by scores such as affinity and electrostatics. Like AutoDock Vina, these conformations can be visualised in programs such as Chimera for further analysis.

1.8 Protein Structure in Drug Discovery

Predominantly proteins are the machines that drive the biochemical life processes in the human body. If one of these processes is going wrong, that can be the cause of an illness or a disease. Protein structures have helped the pharmaceutical industry massively as it makes it possible to create new drugs that can target specific proteins associated with particular diseases or symptoms. The ability to be able to visualise a protein of interest and its predicted structure of that helped make it possible to be able to design drugs that can either stop a protein from working or modulate its function within the body. Methods such as Cryo-EM, X-ray crystallography, and NMR has made it possible to isolate a single protein and define its structure. When these methods are performed, a drug is often co-purified to see whether and how it binds or has an effect of the protein at all. This gives drug designers a clearer understanding of how proteins of interest work. Applying these methods is very expensive, which is the reason that only a small fraction of proteins have been structurally determined which means there is a need for alternative solutions to bridge the "structure gap" to be able to study the le the entire human proteome. This is where computational power is used. As it is not currently possible to crystallise all the proteins in the body, computational modelling is used to predict structures based on templates of homologous proteins that are within a database. This makes it possible for drug designers to identify drug targets with strongly predicted interactions with specific ligands on a large scale.

1.9 Adverse drug events

Adverse drug events are classed as an injury which occurs from a medical intervention-related drug. (Health.gov, 2018) These can be caused by Medication errors, Adverse drug reactions, Allergic reactions, and Overdoses. Adverse drug reactions can cause many problems for patients and put a lot of pressure on health services due to additional appointments and extended hospital stays. The following figures show how much of an issue adverse drug event is (Health.gov, 2018). Data for inpatient Adverse drug events account for an estimated 1 in 3 of all hospital adverse events, Affect about 2 million hospital stays a year, and Prolong hospital stays by 1.7 to 4.6 days. Outpatient adverse drug events account for each year over 3.5 million physician office visits, An estimated 1 million emergency department visits, and approximately 125,000 hospital admissions. Due to adverse drug events, more patients are staying longer in the hospital than they need to be. Physician time is being wasted due to them needing to deal with adverse drug events.

An example of a drug that is known for its off-target interaction is Tamoxifen. Tamoxifen is an estrogen modulator that acts to inhibit the binding of endogenous estrogens competitively. This is a drug that is widely used to treat breast cancer but can cause a number of adverse drug reactions. There have been 22 off-target interactions identified for Tamoxifen which include receptors such as histamine H1 and H3, and muscarinic M1, M4, and M5 subtypes, and dopamine D2 receptor (Flynn, Heale and Alisaraie, 2017).

1.10 High-Performance Computing (HPC) Wales / Supercomputing Wales

High-Performance Computing Wales (https://www.hpcwales.co.uk) was a £44 million project which began in 2009. The project was funded by a number of partners such as the Welsh European Funding Office, Department for Business, Innovation and Skills, and the Welsh Assembly Government. HPC Wales was the UK's largest distributed supercomputing network with an available 17,000 computing cores with a capacity of 320 teraflops. The purpose of the project was to provide high-performance computing easily accessible for businesses, researchers. The HPC Wales project ended in 2015 but since a new £15 million project has been launched called Supercomputing Wales (https://www.supercomputing.wales). This project was funded by the European Regional Development Fund through the Welsh Government. The aim of this project was slightly different to HPC Wales as it is aiming to give access to high-performance computing to university research teams within a consortium of universities, Cardiff University, Swansea University, Bangor University, and Aberystwyth University. The project is not able to be accessible for commercial use. The number of cores available are 13,080 cores which are less than the HPC project, but they use a more up to date, and faster type of CPU as the capacity of the system is 1 petaflop. With the amount of computing power available, HPC Wales and Supercomputing Wales have made the project possible which involves a large amount of modelling and computer calculations to generate the vast amount of data required.

1.11 Drugs

The Association of the British Pharmaceutical Industry (ABPI) produced a list of the top 30 most valuable sold drugs in the United Kingdom for the year of 2013. Within the top 30 list, there are 9 drugs that are proteins. As the docking algorithms that are being used do not support protein-protein interactions, these will be excluded from the study. Shown in table 1.4 is a list of the drugs that will not be used. The Drug Clexane (Enoxaparin) will also be excluded due to there not being an available structure. The remaining non-peptide drugs that are used in this study to test against the entire human proteome. Shown in table 1.5 are the drugs that will be included in the study. The table contains information such as drug name, Drugbank ID, PubChem ID, a brief description of the drug, it's known targets, and finally, the amount spent on the drugs in the United Kingdom.

Table 1.4:Drugs excluded from the thesis – shown are the drugs that have been excluded from the thesis due to them needing protein-protein interaction software which will not be used.

Drug Name	**Drugbank ID**
Enbrel (Etanercept)	DB00005
Herceptin (Trastuzumab)	DB00072
Humia (Adalimumab)	DB00051
Lantus (Insulin Glargine)	DB00047
Lucentis (Ranibizumab)	DB01270
Mabthera (Rituximab)	DB00073
Novarapid (Aspart)	DB01306
Privigen	DB00028
Remicade (Infliximab)	DB00065

Table 1.5 (A): Table of the top 19 most valuable sold drugs – shown are the 19 drugs that will be included in the study. The table contains the name of the drug, Drugbank ID, PubChem ID, a brief description of the drug, known targets of that drug, and the cost of the amount sold in the UK in 2013. Some of the drugs within this study have more than one ligand associated with them.

Drug Name	**Drugbank ID**	**Pubchem ID**	**Description**	**Known Targets (Uniprot ID)**	**Sales (2013) (000s)**
Seretide	DB00588	444036	Seretide also was known as Fluticasone Propionate is a drug which is used to relieve inflammatory and pruritic symptoms of dermatoses and psoriasis. It is also used for respiratory problems such as asthma (Andersson et al., 1999).	P04150, P06401, P47712, P08235	£472,444
Abilify	DB01238	60795	Abilify, also known as Aripiprazole, is an FDA-approved drug, which is an antipsychotic medication for the treatment of schizophrenia (Stip and Tourjman, 2010). It is also a treatment for acute manic and mixed episodes associated with bipolar disorder.	P28223, P14416, P08908, P28222, P28221, P28566, P28335, P46098, P50406, P34969, P21728, P21918, P35462, P21917, P35367, P35348, P35368, P08913, P18089, P18825, P11229, P08172, P20309, P08173, P08912	£86,455

Table 1.5 (B): continuation of table 1.4

Drug Name	**Drugbank ID**	**Pubchem ID**	**Description**	**Known Targets (Uniprot ID)**	**Sales (2013) (000s)**
Atripla	DB00300 DB00879	464205 60877	Atripla, a combination of two drugs being Tenofovir and Emtricitabine, is a member of the group of antiretroviral drugs which are known as analogue reverse transcriptase inhibitors (nRTIs). This drug has been designed to block reverse transcriptase, which is an essential enzyme which helps viral production in HIV infected patients (Fung, Stone and Piacenti, 2002). Both Tenofovir's and Emtricitabine's known target is Reverse transcriptase/RNaseH	Q72547	£137,663
Ezetrol	DB00973	150311	Ezetrol, also known as Ezetimibe, is a drug that is an anti-hyperlipidemic medication which is used to treat low cholesterol levels (Nutescu and Shapiro, 2003).	P35610, Q9UHC9, P15144	£70,194
Glivec	DB00619	5291	Glivec, also known as Imatinib, is a drug that is a kinase inhibitor which is used to treat certain types of cancer such as chronic myelogenous leukaemia (Sacha, 2013) and gastrointestinal stromal tumours (Lopes and Bacchi, 2009).	A9UF02, P10721, O43519, P04629, P07333, P16234, Q08345, P00519, P09619,	£102,058,000
Januvia	DB01261	4369359	Januvia also was known as Sitagliptin, is an anti-diabetic drug which can be used with or without metformin to help control type 2 diabetes mellitus (Plosker, 2014). It has been developed specifically to target the dipeptidyl peptidase 4 (DPP-4) protein.	P27487	£84,011,000

Table 1.5 (C): continuation of table 1.4

Drug Name	Drugbank ID	Pubchem ID	Description	Known Targets (Uniprot ID)	Sales (2013) (000s)
Keppra	DB01202	441341	Keppra also was known as Levetiracetam, is a drug that has been developed as an anticonvulsant medication which is used to treat epilepsy (Abou-Khalil, 2008).	Q7L0J3, Q00975	£67,716
Levothyroxine	DB00451	5819	Levothyroxine is a primary hormone derived from the thyroid gland, which is used to treat hypothyroidism (Javed and Sathyapalan, 2015).	P10827, P10828	£97,688
Lipitor	DB01076	60823	Lipitor also was known as Atorvastatin, is classed as a statin drug which is used to lower cholesterol.	P04035, P27487, P35869	£124,830
Lyrica	DB00230	5486971	Lyrica also was known as Pregabalin, is an anticonvulsant drug which is used to treat neuropathic pain (Verma, Singh and Jaggi, 2014), epilepsy (Ryvlin, Perucca and Rheims, 2008), and generalised anxiety disorder (Baldwin and Ajel, 2007).	O00555	£282,497
Neurontin	DB00996	3446	Neurontin also was known as Gabapentin, is a drug that has been created to treat epilepsy (McLean and Gidal, 2003). It is also used to treat pain, such as neuropathic pain (Backonja and Glanzman, 2003).	P54289, Q9NY47, Q00975, P30542, Q9UBS5, O75899	£76,448

Table 1.5 (D): continuation of table 1.4

Drug Name	**Drugbank ID**	**Pubchem ID**	**Description**	**Known Targets (Uniprot ID)**	**Sales (2013) (000s)**
Revlimid	DB00480	216326	Revlimid also was known as Lenalidomide is a drug that has been created to treat multiple myeloma (Chen et al., 2013) and has also been used for myelodysplastic syndromes (Giagounidis et al., 2007).	Q96SW2, O14788, P33151, P35354	£115,703
Spiriva	DB01409	3086655	Spiriva also was known as Tiotropium has been designed as a long-active anticholinergic bronchodilator drug which is used to manage chronic obstructive pulmonary disease (COPD) (McIvor, 2010).	P20309, P11229, P08172, P08173, P08912,	£246,624
Symbicort	DB01222, DB00983	5281004, 3410	Symbicort also was known as a combination of two drugs, Budesonide and Formoterol. Budesonide is a glucocorticoid which is used to manage asthma (Hodgson, Mortimer and Harrison, 2010). This is also used to treat different types of skin conditions and allergic rhinitis.	P04150, P07550, P08588, P13945	£224,851
Truvada	DB00300, DB00879	464205, 60877	Truvada is also known as a combination of two drugs, Tenofovir and Emtricitabine. Tenofovir and Emtricitabine are both a class of HIV drugs that are called nucleoside reverse transcriptase inhibitors (NRTIs). This drug helps block the critical enzyme that increases viral production (Masho, Wang and Nixon, 2007).	Q72547,	£108,071
Vesicare	DB01591	154059	Vesicare is also known as Solifenacin is a drug created to treat overactive bladder with urge incontinence (Basra and Kelleher, 2008).	P20309, P11229, P08172, P08173, P08912	£90,203

Table 1.5 (E): continuation of table 1.4

Drug Name	Drugbank ID	Pubchem ID	Description	Known Targets (Uniprot ID)	Sales (2013) (000s)
Zytiga	DB05812	132971	Zytiga also was known as Abiraterone is hormonal therapy drug which is used to treat advanced prostate cancer (de Bono et al., 2011).	P05093	£96,684

1.12 Whole proteome-scale screening

An extensive literature search for publications or documentation relating to whole proteome-scale screening revealed no record of it ever having been achieved before. There have been large docking programmes performed using supercomputing but not to this scale. Large docking projects that have taken place are mostly based on one target receptor (Protein) being docked against hundreds to thousands of drugs. An example of this is the work of Capuccini et al., 2017 where they have docked single receptors against around 2.2 million compounds using publicly available cloud supercomputing supplied by Amazon and the use of Googles MapReduce. Capuccini's project was more focused on the parallelization of the docking method and its performance rather than the biological data generated by the experiment.

A similar project has been carried out using high-performance computing for drug-protein docking. An experiment conducted by LaBute et al., 2014 looked at a computational method of predicting Adverse Drug Reactions (ADRs) combined with known information about ADRs collected from Drugbank and SIDER databases. The AutoDock Vina program was used to perform the dockings. These dockings were run for a virtual panel of 409 protein targets that were created from the data stored on the Drugbank database against a set of 906 small molecule drugs. The protein target 3D structures were all crystal structures extracted from the Protein Data Bank database. In total, this resulted in 370,554 calculated docking scores. This combined with the annotated targets of the compounds stored on the Drugbank database were used to train different groups of logistic regression models which in turn produced promising results. When comparing the AutoDock Vina off-target binding models with the Drugbank on-target models, they produced comparable AUCs (area-under-the-receiver-operating-characteristic-curves) during 10-fold cross-validation. There were PubMed entries that have supported several putative ADR proteins that the analysis had produced.

These docking projects though impressive, do not extend to the proteome-wide screening undertaken in this project. This project results in a system that allows the prediction of interactions with 20,386 wildtype proteins against the top 20 most valuable sold drugs in the UK, with the possibility of extending to dockings of known variants and Isoforms. The work of LaBute et al., 2014 looked explicitly at adverse drug reactions. The developed system has the potential to identify off-target interactions which may lead to adverse drug reactions and also repurpose drugs.

1.13 Aims

The aims of this project are as follows:

1. Develop a system which will help assist the lead discovery process, Re-purpose existing drugs, and Reduce the risk (both financial and medical) of drug development by improving efficacy and reducing adverse reactions.

1.14 Objectives

The objectives of this project are as follows:

1. Generate and evaluate 3D structural models the 20,386 wild-type amino acid sequences of the Human Proteome.
2. Undertake the molecular docking of the proteins of the human proteome against the 20 most valuable sold drugs.
3. Implement a versatile system architecture that will facilitate the display and analysis of molecular interactions.
4. Test and validate the accuracy of the docking process in the prediction of known receptor-drug interactions with target proteins and known non-interactions and identify the potential for further improvements if applicable.
5. Evaluate the performance of the system in terms of the scope for the use of the system in reliably predicting new drug leads, repurposing opportunities and adverse drug reactions.

2 Methodology

The core approach of the project that was undertaken is shown in figure 2.1. The study consists of 6

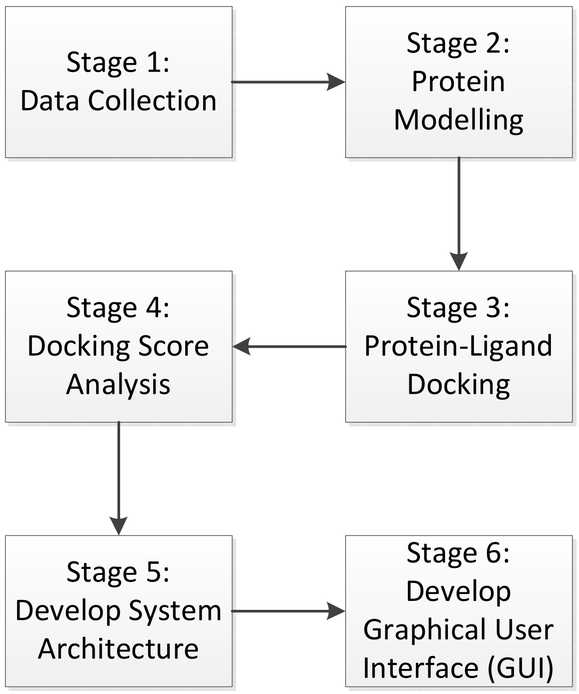


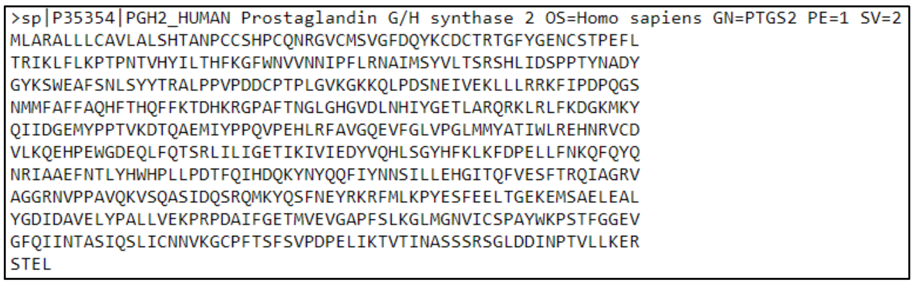
Figure 2.1: Methodology Flowchart – shown are the different stages that had to be undertaken to complete this study. There are 6 stages, which consist of Data Collection, Protein Modelling, Protein-Ligand Docking, Docking Score Analysis, Development of the system architecture, and the development of the graphical user interface

stages that had to be undertaken to complete it. Stage 1 was the collection of data for the proteins and ligands and the extraction of structures. The methods used are discussed in section 2.1. Stage 2 was the modelling of the protein sequences of the entire human proteome. The methods used for stage 2 are discussed in section 2.2. Stage 3 is the docking of the top 20 drugs against the entire human proteome. Methods used for this stage are discussed in section 2.3. The analysis of the docking scores produced by the docking algorithms is stage 4, for which the methods used in the development of an accurate prediction of dockings are discussed in section 2.4. Stage 5 is the development of the system architecture, for which the methods and technologies used are discussed in section 2.5. Finally, stage 6 is the development of the Graphical User Interface which will display all of the docking data generated. The technologies used in the development of the graphical user interface are discussed in section 2.6.

2.1 Data Collection

2.1.1 Human Proteome Amino Acid Sequences

the collection of all the amino acid sequence of the human proteome so that they could be modelled. The sequences are stored in a file format called FASTA. This file format is used in the bioinformatics field for storing nucleotide and peptide sequences. An example of a FASTA file is shown in figure 2.2. The FASTA file format contains the name or names of the protein to which the sequence relates which is placed above the amino acid sequence with the line starting with a “>” symbol. The source from which all of the Human proteome sequences were taken was UniProt (UniProt Consortium, 2018). UniProt is an extensive database of proteins which stores data such as sequence, protein function, known variants, known active sites etc. UniProt facilitates the download of the entire human proteome amino acid sequence in one large FASTA file. Only the Swiss-Prot (Bairoch, 2000) reviewed human proteome sequences were used for this project. This is because these proteins have been manually annotated and reviewed. These were downloaded so that the protein structures corresponding to the amino acid sequences could be modelled. The amount of amino acid sequences obtained from UniProt was 20,160.



```
>sp|P35354|PGH2_HUMAN Prostaglandin G/H synthase 2 OS=Homo sapiens GN=PTGS2 PE=1 SV=2
MLARALLLCAVLALSHTANPCCSHPCQNRGVCMSVGFDQYKCDCTRTGFYGENCSTPEFL
TRIKLFLKPTPNTVHYILTHFKGFWNVVNNIPFLRNAIMSYVLTSRSHLIDSPPTYNADY
GYKSWEAFSNLSYYTRALPPVPDDCPTPLGVKGKKQLPDSNEIVEKLLLRRKFIPDPQGS
NMMFAFFAQHFTHQFFKTDHKRGPAFTNGLGHGVDLNHIYGETLARQRKLRLFKDGKMKY
QIIDGEMYPPTVKDTQAEMIYPPQVPEHLRFAVGQEVFGLVPGLMMYATIWLREHNRVCD
VLKQEHPEWGDEQLFQTSRLILIGETIKIVIEDYVQHLSGYHFKLKFDPELLFNKQFQYQ
NRIAAEFNTLYHWHPLLPDTFQIHDQKYNYQQFIYNNSILLEHGITQFVESFTRQIAGRV
AGGRNVPPAVQKVSQASIDQSRQMKYQSFNEYRKRFMLKPYESFEELTGEKEMSAELEAL
YGDIDAVELYPALLVEKPRPDAIFGETMVEVGAPFSLKGLMGNVICSPAYWKPSTFGGEV
GFQIINTASIQSLICNNVKGCPFTSFSVPDPELIKTVTINASSSRSGLDDINPTVLLKER
STEL
```

Figure 2.2: Example of an amino acid sequence in FASTA format – shown is an example of a protein amino acid sequence in a FASTA file format. The first line starting with “>” contain the name of the protein and relevant information. The amino acid sequence then follows.

2.1.2 Top 20 Most Valuable Sold Drugs

The 3D structural models of the top 20 most valuable sold drug ligands that are listed in section 1.11 were extracted from the PubChem database (https://pubchem.ncbi.nlm.nih.gov) (Kim et al., 2018). PubChem is a database which stores chemical molecules and details of their activities in biological assays. This database stores 3D coordinate SDF files of the ligands. An SDF (Structure-data File) is a file format used in drug discovery which stores structural information and associated data items for a single or multiple ligands (Dalby et al., 1992). Not all the names used within the list provided by the ABPI were the same as the drug chemical names which were used in the PubChem database. To get the appropriate names of the top 20 drugs, a search was carried out on each drug to find their true chemical name. Once the names were gathered the next stage of the process was to download all the SDF file for those top 20 drugs from the PubChem database.

2.2 Protein Modelling

2.2.1 Homology Modelling

To model the amino acid sequences gathered for the proteins of the human proteome, two different methods were used, homology modelling and protein threading. Homology modelling is used as it is widely accepted as generally the most efficient way to compute protein structures. Many papers have been published that prove homology modelling can be very accurate, with models closely resembling how proteins look and function in real life. An example of this is the work of Abdelmonsef and colleagues whereby using a model of Rab38 generated by homology modelling researchers were able to identify an active site then using virtual screening were able to identify novel antagonists (Abdelmonsef et al., 2016). In this project, an in-house homology modelling pipeline was used for the initial modelling stage. The homology modelling pipeline was installed on a High-Performance Computer (HPC), increasing the throughput of the number of sequences that can be modelled in a day. By using HPC, it is possible to compute the structures of multiple protein sequences simultaneously, in parallel. All the sequences were stored in one FASTA file and inputted into the pipeline by uploading to HPC as one. Multiple scripts are subsequently run to prepare the sequences for modelling. When these scripts have completed, a submission script is sent which runs through each sequence and models them.

2.2.2 Protein Threading

Protein threading is used in this project to model those proteins that fail to generate a model using the homology modelling pipeline. Like homology modelling, threading aligns sequences to structure templates, but considers and uses many more templates and iteratively evaluates the resulting models by a score or quasi-energy function, and so is much more computationally intensive, which is why it is used as the second option after attempted homology modelling.

It is not possible to attain structural models for all proteins using homology modelling, due to the limits of homologue template coverage encountered by the approach, which samples relatively long sections of query sequence – template identity in selection of structural templates for homology modelling pipeline to produce a model of any meaningful quality for a sequence, that sequence has to have at least 20% global identity. Therefore, where this is not the case, protein threading is used. Sequences that do not produce a model using homology modelling were modelled by protein threading because protein threading can model sequences that only have about 10% of homologue identity. This is due to the method of how protein threading predicts a structure. In contrast to homology modelling, protein threading models by fold recognition. By modelling the sequences that could not be modelled by homology modelling, it was possible to have greater coverage of the human proteome for docking against the top 20 drugs.

The threading program that was used in this project is I-TASSER (Yang et al., 2015). I-Tasser is a suite of multiple programs that work together to predict the structure and function of a protein. I-TASSER was selected for this project due to how it compares to other similar programs. I-TASSER was ranked number one by the Critical Assessment of Techniques for Protein Structure Prediction (CASP) (http://www.predictioncenter.org). I-TASSER competed in CASP7 (2006), CASP8 (2008), CASP9 (2010), and CASP10 (2012), CASP11 (2014), CASP12 (2016) and came first in all six competitions, making it widely accepted as the best protein threading program available.

In this study, all the sequences that do not produce a model with homology modelling were run using I-TASSER to get more viable and higher quality structures to dock against the top 20 drugs. I-TASSER was also installed on HPC, which increased the output of sequences that can be modelled per day.

Compared to homology modelling, I-TASSER takes a lot more CPU time to model a given amino acid sequence for a protein. There is a stage in I-TASSER called the Monte Carlo simulation stage which runs up to 15 separate modelling simulations for each sequence. These can take up to 50 hours for each of the 15 or so simulations. By using HPC and multiple cores, it only takes up to a one 50-hour stint to complete all the simulations in parallel, much faster than having to carry out one simulation at a time. The completion time also depends on sequence length, the longer the sequence, the closer it gets to hitting the 50-hour limit for a simulation. Due to this time run issue, the sequences that did not model using homology modelling were split into different FASTA files depending on sequence length. The shortest sequences were modelled first, and when completed, the routine moved on to the next sequence length until the modelling of all sequences had been attempted. When these models were completed, they were prepared for docking with the two docking algorithms being used.

2.3 Protein-Ligand Docking

Protein-ligand docking algorithms are applied to try to predict the interactions between a predicted 3D protein structure and the 3D representation of a ligand. Protein-ligand docking incorporates molecular docking, which tries to predict the structure of the intermolecular complex formed between two or more constituent molecules. This has become a tool used widely for drug discovery. This field has been growing because of its application to medicine (Sousa et al., 2013).There are many challenges for drug discovery, for example a lack of characterised targets, so *in silico* predictions of absorption, distribution, metabolism, elimination and toxicity (ADMET) are being increasingly used to help focus medicinal chemistry into more ideal areas of biological and chemical property space, minimizing the number of ligands needed to be synthesized (White and Modi 2018).

The basis of ADMET predictions lies in the understanding of molecular interactions between given receptors and ligands, predicted by molecular docking. There were several docking programs available to perform this task, but only two were chosen for this project. These are AutoDock Vina (Trott and Olson, 2009) and UCSF DOCK6 (Allen et al., 2015). AutoDock Vina and USCF DOCK6 were the chosen as they are freely available and that they could easily be implemented on HPC. AutoDock Vina and DOCK6.0 were the chosen algorithms to be used to predict the dockings against the top 20 most prescribed drugs with the proteins of the human proteome.

2.3.1 AutoDock Vina

In the case of AutoDock Vina, the PDB files need to convert into a PDBQT format. A PDBQT file is very similar to a PDB file format, but it can store data such as atomic coordinates, partial charges and AutoDock atom types for both the receptor and the ligand used for the docking. Section 1.7.1 discusses in more detail the process that AutoDock Vina uses. In the development of more accurate protein-ligand docking predictions, Shipyard 1.5 incorporated AutoDock Vina into it is docking pipeline which section 2.3.4 discusses.

2.3.2 DOCK6

In the case of DOCK6.0, the PDB files must be converted into a MOL2 file format. A MOL2 file format is the file format for DOCK input and output. Compared to the PDB file format, MOL2 can store extra information which is essential for DOCK6.0 to compute a docking such as atom features, position, and connectivity. Discussed in section 1.7.2 is the DOCK6 process in more detail. In the development of a more accurate protein-ligand docking prediction, DOCK6 was also incorporated into a docking pipeline called Shipyard 1.5 which is discussed in section 2.3.4.

2.3.3 DoGSiteScorer

DoGSiteScorer is a program that can detect potential binding pockets and subpockets of a protein by analysing a PDB file (Volkamer et al., 2012). The program analyses geometric and physiochemical properties of the protein pockets and gives a prediction of the druggability of those pockets using a support vector machine (SVM). An SVM is a supervised machine learning model which is used for analysing data for classification and regression. Shown in figure 2.3 is a flowchart for the DoGSiteScorer process. Firstly, by analysing the surface of the protein to identify potential pockets by only using the

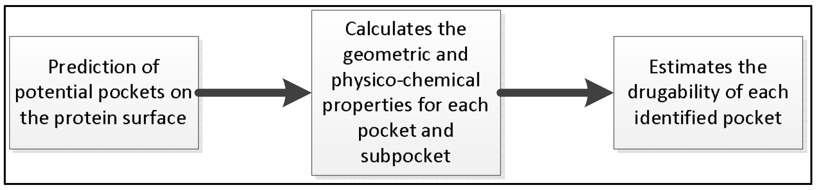

Prediction of potential pockets on the protein surface
Calculates the geometric and physico-chemical properties for each pocket and subpocket
Estimates the drugability of each identified pocket

Figure 2.3: DoGSiteScorer flowchart – shown is the flowchart of the DoGSiteScorer process.

coordinates of the heavy atoms. A grid is placed around the protein and points are labelled, which is dependent on whether there is any spatial overlap with any of the protein's atoms. A Gaussian filter is then applied to the grid which helps identify locations on the protein surface where a sphere-like object is favourable. The program then clusters all the identified positions into sub-pockets using a density threshold. Many geometric and physicochemical properties are calculated for all the identified pockets and sub-pockets automatically. Then the volume of each pocket is calculated by using the volume of the grid boxes which had been applied previously. The depth of the pockets is then calculated, and atoms that are within 4 angstroms of any of the pocket points is classified as a pocket atom. This leads to the composition of the pockets being determined, which describes the physicochemical features of those pockets.

The final stage of the DoGSiteScorer process is to calculate the drugability of each of the potential pockets. This stage is when an SVM is used to analyse and estimate the attractiveness of a pocket for a drug. The SVM model used has been trained and tested against a non-redundant version of a drugable dataset (Schmidtke and Barril, 2010). The SVM was trained by half of the drugable set, and the other half was used as test data with an outcome of a mean accuracy of 90% (Volkamer et al., 2012). The program then gives a score to each of the identified pockets between 0 and 1 with the higher score meaning, the more druggable an identified pocket is.

In this study, the DoGSiteScorer program is used to identify potential pockets that a drug could bind to for every 3D modelled protein either via homology modelling or protein threading. The generated information is then used personalised search areas for each unique protein for AutoDock Vina and DOCK6. Using this method gives the advantage of the processing speed for each docking increasing significantly. The processing speed increases due to AutoDock Vina and DOCK6 not needing to search the entire protein surface area but instead only to use a search area which contains the potential docking pockets.

2.3.4 Shipyard 1.5

Shipyard is a docking pipeline that was developed by Karl Austin-Muttitt, a colleague of the author at Swansea University Medical School as part of a more extensive Research Group project which became applicable to this study. The pipeline executes the process of docking the same group of proteins and ligands together using the two different docking algorithms, AutoDock Vina and DOCK6.0. A diagram of the Shipyard 1.5 workflow is shown in figure 2.4. Shipyard takes a folder containing all the proteins that want to be used and a folder containing all the ligands that are going to docked against them. The protein models would initially be in PDB format, and the Ligands would be in SDF format. The first stage carried out by the Shipyard pipeline is to prepare the Protein and Ligand files. On each of the protein PDB files, a program called "reduce" is run. This program adds

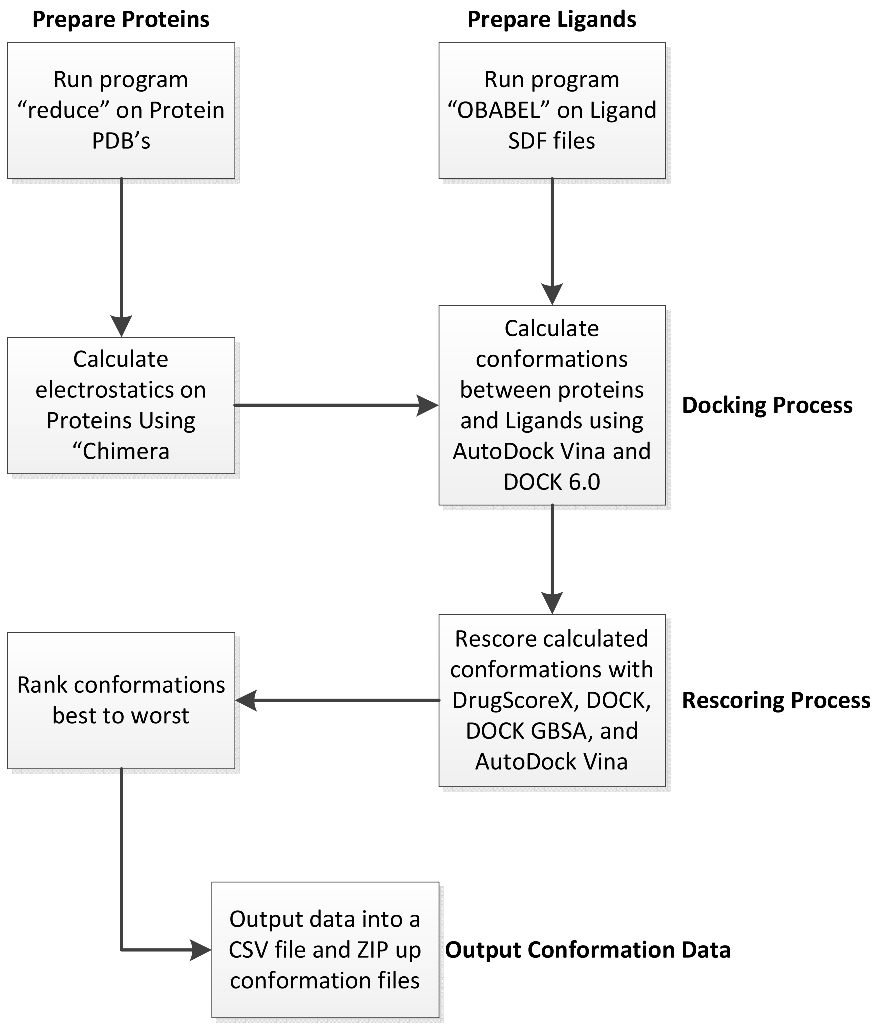

Prepare Proteins
Prepare Ligands
Run program "reduce" on Protein PDB's
Run program "OBABEL" on Ligand SDF files
Calculate electrostatics on Proteins Using "Chimera
Calculate conformations between proteins and Ligands using AutoDock Vina and DOCK 6.0
Docking Process
Rescore calculated conformations with DrugScoreX, DOCK, DOCK GBSA, and AutoDock Vina
Rescoring Process
Rank conformations best to worst
Output data into a CSV file and ZIP up conformation files
Output Conformation Data

Figure 2.4: Workflow of the docking process for Shipyard 1.5 – shown is the process shipyard 1.5 undertakes when performing a docking. Firstly, the protein PDB's and ligand SDF goes through a preparation stage which validates that the files are correct. An extra step is carried out for the protein PDB to calculate the electrostatics of the protein. The second stage is to dock the ligand and protein together using AutoDock Vina and DOCK6. The third stage is that the conformations generated by both docking programs are rescored using different scoring functions with DrugScoreX, DOCK, DOCK GBSA, and AutoDock Vina. The fourth stage is that all of the rescored conformations are ranked from best to worse, and the top 10 conformations are selected. The final stage is that all scoring results and conformations are inputted into a zip file.

hydrogen atoms to the PDB file which are not computed during the modelling process in either homology modelling or threading pipelines. These hydrogens need to be added before the docking process as they affect the ‘partial charge’ of atoms that they bond to. Hydrogens can affect and determine if an atom is negatively or positively charged. It is essential that these are added as this property determines the general of each protein. After all the hydrogen atoms have been added, a program called UCSF Chimera (Pettersen et al., 2004) is used to calculate the electrostatics of each protein PDB. UCSF Chimera is a program that can visualise and analyse molecular structures and related data such as protein structure and protein-ligand dockings. After adding the hydrogens, it is known which parts of a protein are positively or negatively charged, and by also knowing which ligands are positively or negatively charged, it is possible to know which regions of the protein are likely to possess a strong electrostatic attraction for a particular ligand.

The next stage involves the use of a program called DMS (Distributed molecular surface) to be used to define the surface area of the protein. It is essential for the docking process that the protein surface is defined as it will help determine where there are potential pockets for ligands can fit into. Finally, a program called Sphgen, which is an accessory program within DOCK6, is run on each protein PDB file. Sphgen processes a protein structure and calculates every possible space a ligand could be docked in. It does this by generating sets of overlapping spheres around the surfaces of the protein. These are then considered by AutoDock Vina and DOCK6.0 as areas where the algorithms should attempt to dock a ligand.

For the preparation of the ligands, a program called OBABEL (O'Boyle et al., 2011) is used to add hydrogen atoms onto them like the “reduce” program for the protein PDB files. Chimera is then used to calculate the charges of each atom in the ligand. When all the protein and ligand coordinate files have been prepared, they are docked together using the AutoDock Vina and DOCK6.0 algorithms which were discussed in section 1.7.1 and 1.7.2. Both docking algorithms will use the Sphgen spheres calculated for the proteins and attempt to dock them in each determined pocket. Both AutoDock Vina and DOCK6 algorithms produce up to 10 conformations for each docking pair and score them using their in-built scoring functions. When all docking has completed, all the calculated conformations are then run through a variety of different scoring functions. Shown in table 2.1 are the different scoring functions that Shipyard 1.5 can apply to every docking conformation generated.

Table 2.1 (A): Scoring functions calculated in Shipyard 1.5 – shown are all of the scoring functions that the Shipyard 1.5 pipeline calculates on every protein-ligand docking pair and the definition of what each scoring function calculates.

Scoring Function	**Definition**
DOCK Affinity	These are the components of DOCK's physics-based scoring function, which are linked by the relationship: DOCK_Affinity = DOCK_van_der_Waals + DOCK_Electrostatics.
DOCK van der Waals	A scoring function within DOCK which calculates the force between atom caused by induced polarity; necessary for neutral atoms
DOCK Electrostatics	Calculates the force between electrical charges
DOCK Internal energy	Calculates the energy needed to rotate the bonds of the ligand to a final conformation
GBSA (Generalised Born with solvent-accessible Surface Area) affinity	These are the components of the GBSA (generalised Born with solvent-accessible surface area) scoring function. This is a more in-depth physics-based function that attempts to account for solvation effects. The scores are related by: GBSA_Affinity = GBSA_van_der_Waals + GBSA_Screened_electrostatics + GBSA_Born_solvation + GBSA_Solvent-accessible_surface_area.
GBSA (Generalised Born with solvent-accessible Surface Area) van der Waals	A GBSA scoring function which calculates the force between atom caused by induced polarity; important for neutral atoms
GBSA (Generalised Born with solvent-accessible Surface Area) screened electrostatics	A GBSA calculation the force between atom caused by induced polarity but modified to take into account that the solvent has its own electrical properties
GBSA (Generalised Born with solvent-accessible Surface Area) Born solvation	Calculates the energy needed to displace water
GBSA (Generalised Born with solvent-accessible Surface Area) Solvent-accessible surface area	Calculates the energy of the interaction of the ligand and the water that surrounds it
Vina Affinity	These are the heuristic parameters used by AutoDock Vina's empirical scoring function, linked by: Vina_Affinity = Vina_Gauss_1 + Vina_Gauss_2 + Vina_Repulsion + Vina_Hydrophobic + Vina_H-bond + Vina_Rotatable_bond_entropy.
Vina Gauss 1	Gauss scoring functions model attractive atom-atom forces (and, correspondingly, the repulsion function models repulsive atom-atom forces).

Table 2.1 (B): Continuation of table 2.1.

Scoring Function	Definition
Vina Gauss 2	Gauss scoring functions model attractive atom-atom forces (and, correspondingly, the repulsion function models repulsive atom-atom forces).
Vina Repulsion	Vinas equivalent to calculating the van der Waals repulsion
Vina Hydrophobic	Calculates the effect that hydrophobic groups would prefer to be in contact
Vina H-bond	calculates the hydrogen bonds between the ligand and the protein
Vina Rotatable bond entropy	Energy term that calculates the effect of the flexible bonds within the ligand on the overall binding energy
DSX (Drug Score X) atom distances	These are the machine learning outputs used by DSX (Drug Score X) in its knowledge-based scoring system. The columns are related: DSX_Atom_distances = DSX_Contact_count × DSX_Per_contact
DSX (Drug Score X) Contact count	DSX_Per_contact models the strength of the interactions individual ligand atoms make with atoms in binding residues they are in contact with (in a ligand size-independent way).
DSX (Drug Score X) per contact	DSX_Atom_distances attempt to model the overall binding energy (which can depend on ligand size)

Each conformation is uploaded and scored using the scoring functions above. This is so that conformations from AutoDock Vina and DOCK6.0 can be compared with each other using a standardised scoring function. Shipyard then ranks each of the docking pair conformations from each docking algorithm and selects the top 10 scoring conformations. The scores of the top 10 conformations for each docking pair are then inputted into a large CSV file, in a format that it can be inputted into the database scores relate to, DOCK score, AutoDock Vina score, DOCK GBSA score, and DrugScoreX score.

2.4 Docking Score Analysis

2.4.1 Mean and Standard Deviation Outlier Method

The mean and standard deviation outlier method is a known mathematical calculation which is used to identify outlier within a mass amount of data. In this study, the mean and standard deviation outlier method is used on the docking data generated on each individual drug column. Firstly, the mean and standard deviation is calculated for each drug column within a docking data file. Using each individual docking pair score, the following formula is applied to calculate the number of standard deviations away from the drug column mean:

$$Standard\ deviations\ away\ from\ the\ mean = \frac{Docking\ Pair\ Score - drug\ column\ mean}{Drug\ column\ Standard\ Deviation}$$

With the results of this equation, the rules shown in table 2.2 are applied to identify outliers within each drug column.

Table 2.2: Rules applied to Standard Deviation Outlier Method results – shown are the scores and rules that are applied to determine if a docking is an interaction or a possible interaction.

Standard Deviations away from the Mean	**Rule**
Between -1 and -2	Docking classified as a potential interaction
-2 or less	Docking classified as interaction

The Mean and Standard Deviation Outlier Method is used throughout the docking analysis, which is discussed in section 4.

2.4.2 Combinational Forecasting

Combinational Forecasting is a method where many different parameters can be considered and combined into a single result which can increase the accuracy of predictions (Jiang, Zhang and Song, 2014). Within this study, combinational forecasting is used to increase the accuracy of identifying the known interaction of the top 20 drugs. As there were many docking scoring functions generated, combinational forecasting was used to help improve baseline results as individually, they were not very accurate, which is discussed in section 4.

2.4.3 Area Under the Curve (AUC) and Receiver Operating Characteristics (ROC) Curve

AUC – ROC curve is a measurement that can determine the performance or the quality of a diagnostic test. Within this study, this method is used to determine the accuracy of each docking scoring function used within shipyard 1.5 for correctly predicting known targets of drugs and known misses of drugs correctly. This is calculated by plotting the true positive rate against the True Negative rate. The true positive rate is the proportion of actual positives that have been correctly identified, and the True Negative rate is the proportion of actual negatives that have been correctly identified. When these have been determined, they are then plotted on a graph like the one shown in figure 2.5. When the graph has been plotted, the ROC curve can be drawn. The max number on either axis is 1. With the ROC curve determined, the AUC can be calculated. To calculate the AUC, the area underneath the curve must be split up into sections which are shown in figure 2.6. Area A, which is shown in figure 2.6 will have its area calculated using the following formula due to it being a rectangle:

$$Area\ in\ section\ A = Height\ of\ section\ A\ X\ Width\ of\ section\ A$$

For example, if the height of section B was 0.7, and the width of section B was 0.8, the area of section A would be 0.56. To calculate section B and C, the following equation is used due to both shapes being a triangle:

$$Area\ in\ section\ (B\ or\ C) = \frac{Height\ of\ the\ section\ X\ Width\ of\ the\ section}{2}$$

For example, section B has a height of 0.7 and a width of 0.2 which gives the area 0.07 by using the equation, and for section C which has a height of 0.3 and a width of 0.8 would give an area of 0.12. Now with all of the areas calculated, the final stage is to add all of them together for which in the example would be an area under the curve of 0.75. For the scores given for the AUC, the closer the score is to 1, the higher the accuracy.

It was possible to determine the most accurate scoring functions that shipyard 1.5 produced an is explained further in section 4.7.

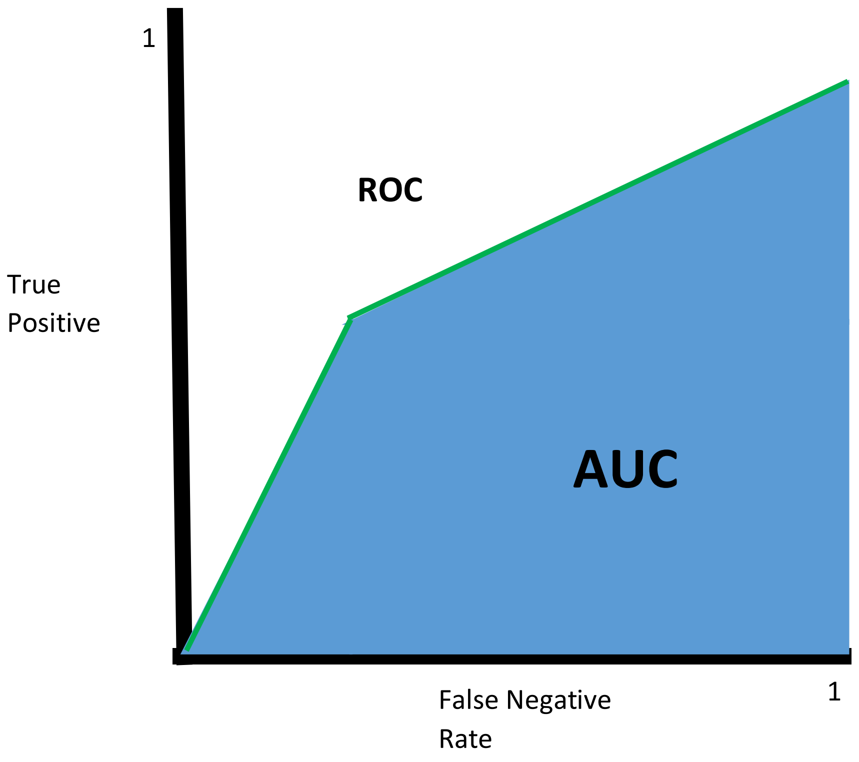


Figure 2.5:Example of an Area Under the Curve (AUC) and Receiver Operating Characteristics (ROC) Curve – shown is an example of how a ROC curve is plotted and the area that is used to calculate the AUC

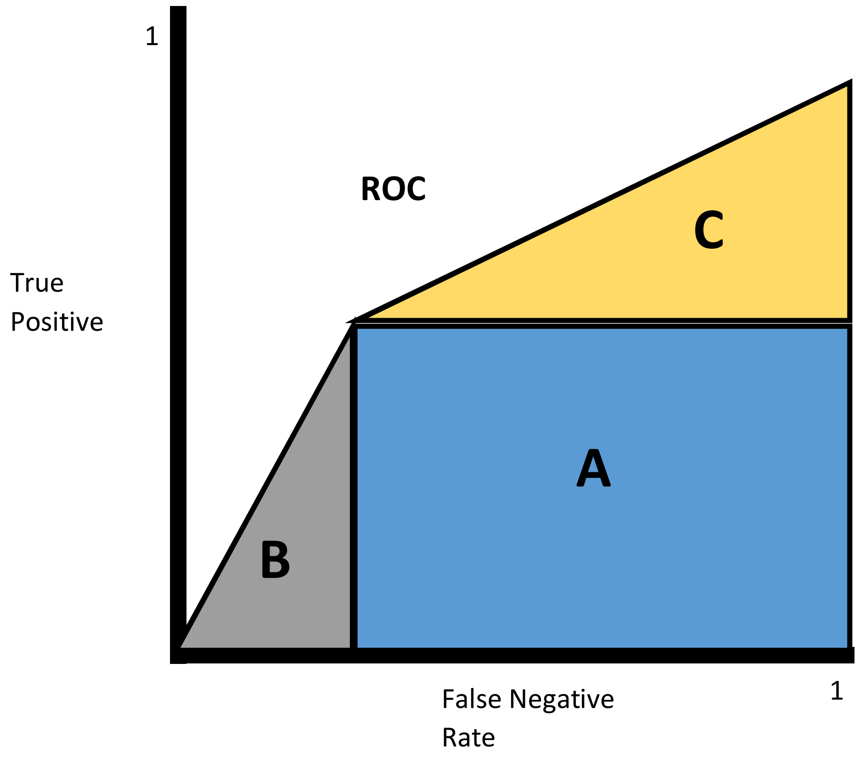


Figure 2.6:Example graph of how to calculate the area under the curve (AUC) – shown is an example of how the AUC is calculated by splitting the area into sections and calculating the area by using calculations explained in section 2.4.3.

2.5 System Architecture

2.5.1 Structured Query Language (MySQL)

MySQL is an open-source relational database management system which was created and maintained by the Oracle Corporation. MySQL is a server-side system which allows the ability to structure data into tables and create relationships between those different tables. With it being possible to create relationships with different tables, it avoids having to put all data within one table which would cause an array of problems such as database querying speed and the possibility of the data becoming too complicated to read. MySQL gives the ability to add, access, and process the data stored within that database and can work with web languages such as PHP which allows a web-based system the ability to query a database.

MySQL is used to create the core database of the study which will store all data such as protein and ligand information, docking scoring function results, and file locations for the protein, ligands, and generated conformations.

2.5.2 Ubuntu Server

Ubuntu server is an open-source Linux operating system (OS). The operating system was developed by Canonical Ltd, which maintains and releases security updates the Operating system. The version of the Ubuntu operating system used in this study is Ubuntu 16.04 LTS. The system is being used as it is a free server OS which is also being used on the server which will be storing and running the final database and graphical user interface for the created protein-ligand docking database system.

2.6 Graphical User Interface (GUI)

2.6.1 HyperText Markup Language (HTML)

Hypertext Markup Language is an open-source language, which is used for the structuring and inputting the content of a website page. HTML is the basis of a website, but it is not a programming language. The language uses “tags” to structure a website. These “tags” tell the web browser how a website should be displayed. For example, if there is a heading for the page, this would be put in-between a “<h1>” tag. This tells the browser that this is a heading so that the text will be more prominent. These “tags” are used mainly for the basic structure. For more advanced visual layouts, a language called Cascading Style Sheets is used.

2.6.2 Personal Home Page (PHP)

Personal Home Page or PHP: Hypertext Pre-processor, as it is known, was initially used to create dynamic web pages and is a server-side scripting language. PHP’s primary use is to interact with MySQL databases which are stored on a server. Example code of how PHP connects to a MySQL database is shown in figure 2.7.

```
<?php
$con=mysqli_connect("localhost","username","password","database_name");
// Check connection
if (mysqli_connect_errno())
        {
        echo "Failed to connect to MySQL: " . mysqli_connect_error();
        }
?>
```

Figure 2.7: PHP code which allows a connection to a MySQL database – shown is an example of code that can be used to connect the client to a MySQL database stored on a server. The information needed within the code are the IP address of the server (localhost), the username and password for the database (username, password), and the name of the database (database_name).

It has many other uses; for example, it can be used to run programs on a server’s command line and generate XML Documents. PHP can be embedded within an HTML website, but different from HTML, PHP is a programming language. An HTML page which has PHP embedded within it has a file name ending in “.php” instead of “.html”.

For this interface, PHP will be used to interact with the MySQL database stored on the local server. It will also be used to run some scripts on the Linux command line in some stages of the interface/server interactions.

2.6.3 Cascading Style Sheets (CSS)

Cascading Style Sheets is used to design the style and layout for displaying the content of the website. This links with the HTML “tags” within the website file and tells the web browser how the website should look and where components of the website should be placed. A CSS is usually a separate file in which only styling code is placed. This is linked to the HTML files using the code, as shown in the example below:

<link rel="stylesheet" href="(style sheet file name and location)">

When the website is loaded within a web browser, the browser loads the CSS file followed by the HTML code within the “<body>” tag as directed by the CSS file. For parts of the CSS design, a CSS framework called Bootstrap (http://getbootstrap.com/) is used.

2.6.4 JavaScript

JavaScript is used to code how a website behaves and is presented in the web browser. JavaScript is a client-side language that can work with the Document Object Model (DOM) of a web browser. It is mostly used to create interactive websites. The most common use for JavaScript is the creation of pop-up windows within a browser. This is used in this project to create an interactive table with data from the proteins, ligand and dockings tables.

2.6.5 AJAX

Asynchronous JavaScript and XML (AJAX) is a language that is used to communicate with a database or server to update parts of a webpage, without the need for refreshing the entire page. This is used to make the webpage more interactive and speed it up. Without using AJAX, there would be a need for the user to refresh a page every time the content needed to be changed. Shown in figure 2.8 is a diagram showing how AJAX works. AJAX language is based on internet standards and uses HTTP requests to exchange data with the database/server in the background. AJAX can then link with JavaScript/DOM to display the data and enhance user interaction with that data. This will be used to work with the database of this system, in the background, so that the interface table will be

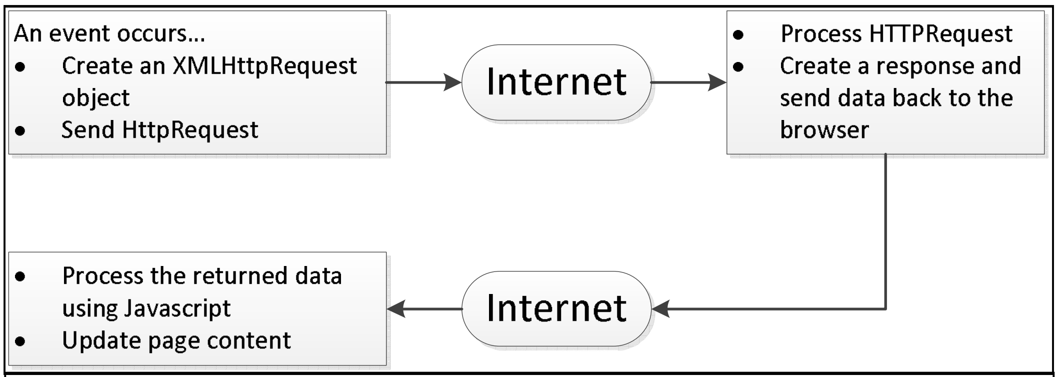

An event occurs...
Create an XMLHttpRequest object
Send HttpRequest
Internet
Process HTTPRequest
Create a response and send data back to the browser
Internet
Process the returned data using Javascript
Update page content

Figure 2.8: Diagram showing how AJAX works within a web page

continuously updated depending on the user’s choices. This language also works well with JavaScript and PHP, which are also being used in the interface.

2.6.6 PV – Javascript Protein Viewer

PV is a protein visualiser which has been developed in Javascript which can represent PDB files in 3D. The Viewer is used as the default protein viewer for SWISS-MODEL (Waterhouse et al., 2018). The PV protein viewer has featured such as support for different styles of rendering a protein, for example, Spheres (to chow protein surface) and ribbon (to show alpha-helix’s and beta sheets). Allows residues to be selected and distances to be measured between them and also the ability to label within the viewer. This viewer will be used to display proteins and ligands on their selected individual pages but also to display conformations for selected protein-ligand docking pairs on the graphical user interface.

2.6.7 DataTables

For designing the tables, a JQuery plug-in is used called DataTables (http://datatables.net). DataTables is a library of CSS and JavaScript files that have been created explicitly for displaying and giving enhanced functions to any HTML table. This plug-in is used to give search functions to the table window, and due to there being a considerable amount of data to display within the table it creates pages so that the user is able for example, views ten results at a time extending up to 25 results at a time. The DataTables plug-in is used on all data tables that are generated on the graphical user interface to give the user more flexibility for how they would like to search and interact with the data.

3 System Architecture

This chapter will describe the architecture of the system that has been developed to handle all of the data from protein and ligand information and generated conformations produced by the docking methodologies.

3.1 System Overview

The objective for this system is to provide an easy way for biomedical researchers and pharmaceutical companies to be able to view specific receptor-ligand interactions for re-purposing drugs and to investigate the potential molecular causes of adverse reactions for some users with particular medicines. The system needs to be able to store a large amount of data, including 3D Protein/Ligand models, Protein and ligand data, Docking results, and Structural conformation files. This large store of data allows the user to view all relevant data about the receptor and ligand interactions that have been computed using the docking algorithms.

The process of creating the system design is completed by using a Virtual machine (VM) with Ubuntu Server Operating system (OS). The next step is installing the Apache HTTP server software is installed with MySQL and PHP with a PhpMyAdmin control panel interface on this system, allowing ease of data manipulation between the MySQL database and the web interface. Storing the system on a local server which has internet access allows access by the end-user from any location.

Figure 3.1 shows the system flow chart. Each box represents a particular stage in the information workflow, moving from first obtaining the protein sequences and ligand structures through to the user viewing the vast amounts of data produced.

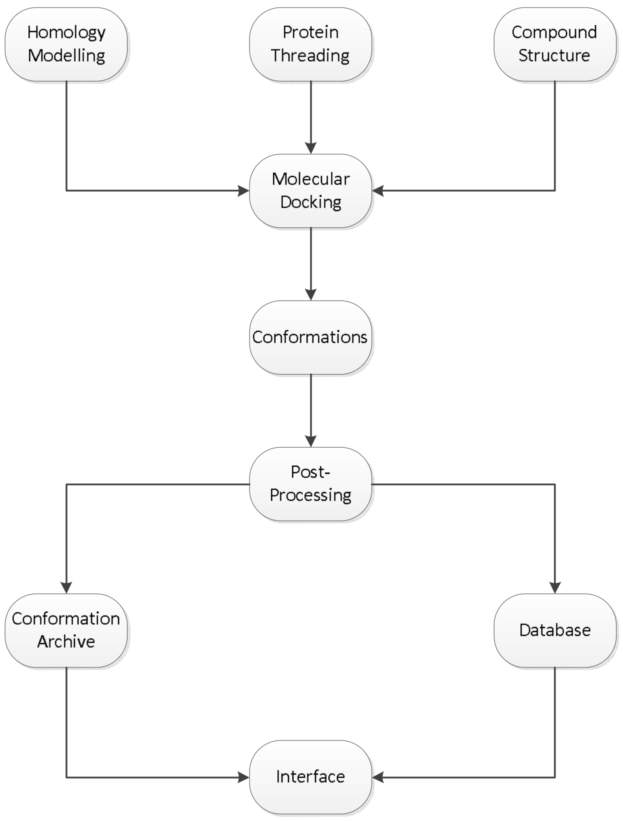


Figure 3.1: System Flowchart – this workflow shows the created system architecture.

3.1.1 Homology modelling

The processing of data begins with structural modelling. The sequences are collected and submitted to the homology modelling pipeline for modelling. An explanation of the homology modelling pipeline in more depth in section 1.6.4. All sequences that fail to produce a model are modelled using protein threading.

3.1.2 Protein Threading

Protein sequences that cannot be modelled using the homology modelling pipeline are sent to be modelled by protein threading. This process uses I-TASSER, refer to section 1.6.5 in the introduction. I-TASSER is a suite that incorporates eight different threading algorithms to determine the best templates that can be used to model a sequence. A more in-depth description of the process is in Chapter 4.

3.1.3 Ligand Structures

Collecting the Ligand structures from PubChem by using PubChem ids of gives the top 20 drug ligands used in the UK. The list is available from ABPI for the year 2013. The list of ligands is within the introduction.

3.1.4 Molecular Docking

The ligand models and protein models are gathered or generated before sending for docking against each other. The docking of each ligand with each protein is investigated to find possible interactions. The docking algorithms used are AutoDock Vina and DOCK6.0. The docking process is described further in chapter 4.

3.2 Database Prototype 1

All data for the proteins, ligands, and conformations are stored within a database. The database will be developed using MySQL language. Figure 3.2 shows the broad structure of the first database designed for the system.

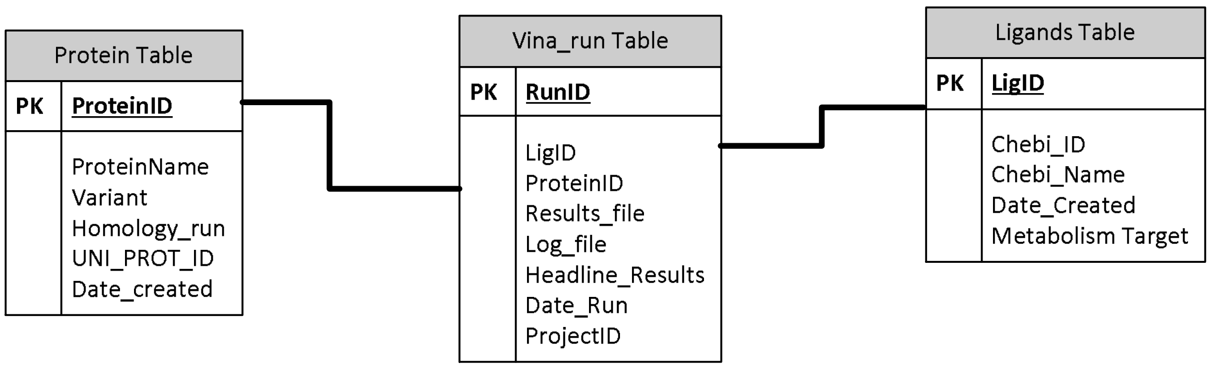


Figure 3.2: First Database Design showing 3 tables (Protein Table, Vina_run Table, Ligands Table). The Protein Table is linked to the Vina_Run Table with the common field "ProteinID". The Ligand Table is linked to the Vina_run Table with the common field "LigID".

The first database designed was very simple with only three tables, one for proteins, one for ligands and an (AutoDock) Vina runs the table. These tables contained unprocessed information about the proteins and ligands, and the basic Vina runs output. This first design of the database created before the completion of the docking results.

3.2.1 Protein Table

The Protein table holds relevant information about the proteins stored within the system. Within the Protein table, there are six columns. The primary key column "ProteinID" is for the unique ID of each protein. The system autoincrements the ID. The primary key is identified by queries searching to gather information about each protein. The column "ProteinName" is for the full name of the protein. The column "Variant" is so that if this protein is not the wild-type and has a mutation within it, then the logging of the mutation is done within this column. For example, "S235T" which stands for Serine has mutated at position 235 into Threonine. For the user, this is essential information for when investigating differences between wild-type and variant conformations. The column "Homology_run" is for storing the 3D protein models if modelled via homology modelling. This is so when selecting a protein; the user will know whether the modelling of the protein is via homology modelling or not. The “UNI_PROT_ID” column stores the UniProt code associated with the protein. The user can access more information about the protein if needed from the UniProt website and be able to search the database directly by UniProt accession code. The” Date created” column stores the date of the 3D model's generation. The PDB database is updated continuously, which means it would allow the sequences to be routinely re-modelled with an automated system to see if there are new templates that could improve the quality of the model structure. The system can, therefore, contain multiple models of the same protein, date referenced.

3.2.2 Ligands Table

The ligands table consists of 5 columns which store data about the ligands used in the dockings. The primary key is the “LigID”, which as the primary key in the proteins table, is the unique ID for the ligand. The “Chebi_ID” column is where the CHEBI ID number of the ligand is stored. On the interface, the user can link to the CHEBI database to find more information about the ligand. The “Chebi_Name” column is the name used in the CHEBI database for the given ligand. Therefore enabling the user to search the database for a specific ligand by name. Like the “Date_Created” column within the proteins database, storing the ligand downloaded date is similarly stored. Finally, the “Metabolism Target” stores known targets of the ligand, enabling the user to examine these directly through the interface.

3.2.3 Vina_run Table

This table holds all the data of the dockings and their results produced by AutoDock Vina and DOCK6.0. This table also links with the Protein and ligand tables. The table consists of 8 columns. The primary key for this table is the “RunID”, The unique ID number given to each separate docking.

The “LigID” contains the ID of the ligand used in the docking. This ID links with the “LigID” column within the ligand table. The “ProteinID” contains the ID of the protein used in the docking. This links to the “ProteinID” column in the protein table. With these tables linked together by these ID’s, the database is queried to obtain specific information about the protein and ligand used in each record of this table.

The “Results_file” column stores the location of the results files produced by AutoDock Vina and DOCK6.0. The “Log_file” column is for the location of the log file of the docking produced. The “Headline_Results” column is the best conformation score produced by DOCK6.0 and AutoDock Vina. The process of extracting from the conformation file produces the score. Like the “Date_Created” columns within the protein and ligand tables, the “Date_Run” column stores the date the docking had taken place. It is necessary due to the generation of multiple dockings of different models of the same protein and ligand. Finally, the “ProjectID” gives the name of the particular research project that those dockings are related to, for example, who or which research group or collaborator wants to have these proteins and ligands docked. This broad structure is adapted to future proof the system - currently, the system only holds data for the top 20 drugs against the entire human proteome, but this can be expanded to include many further ligands in the future.

3.3 Database Prototype 2

The preliminary version 1 database design development assumed that all that was needed were the docking scores and conformations and information about the proteins and ligands involved. After discussing with potential users, it was clear that this design needed to be expanded to include more information. It was clear that there was a need to connect the proteins and ligands with a broader range of databases used in the field, includes databases such as KEGG (Kyoto Encyclopaedia of Genes and Genomes) (Kanehisa et al., 2018)(https:// https://www.genome.jp/kegg/pathway.html) and the Reactome Pathway Database (Fabregat et al., 2017) (https:// https://reactome.org) which is an open-source pathway database. New tables and links had to be added to accommodate the new data. Figure 3.3 shows the updated database structure.

Adding two new data tables providing links to the protein table. Compared to the previous design, this database structure is more comprehensive and can store a more considerable amount of data. One table called KEGG has been added to store KEGG related data about the proteins within the system, linking to the existing protein information by using the “ProteinID” column in both. The purpose is to create a query for a particular protein that can gather all the relevant information about the protein with the single selected ID. The other new data table is “Reactome” as shown in figure 3.3. This table stores all the Reactome ID’s related to a selected protein using the “ProteinID” column. It is a many-to-many relationship as many

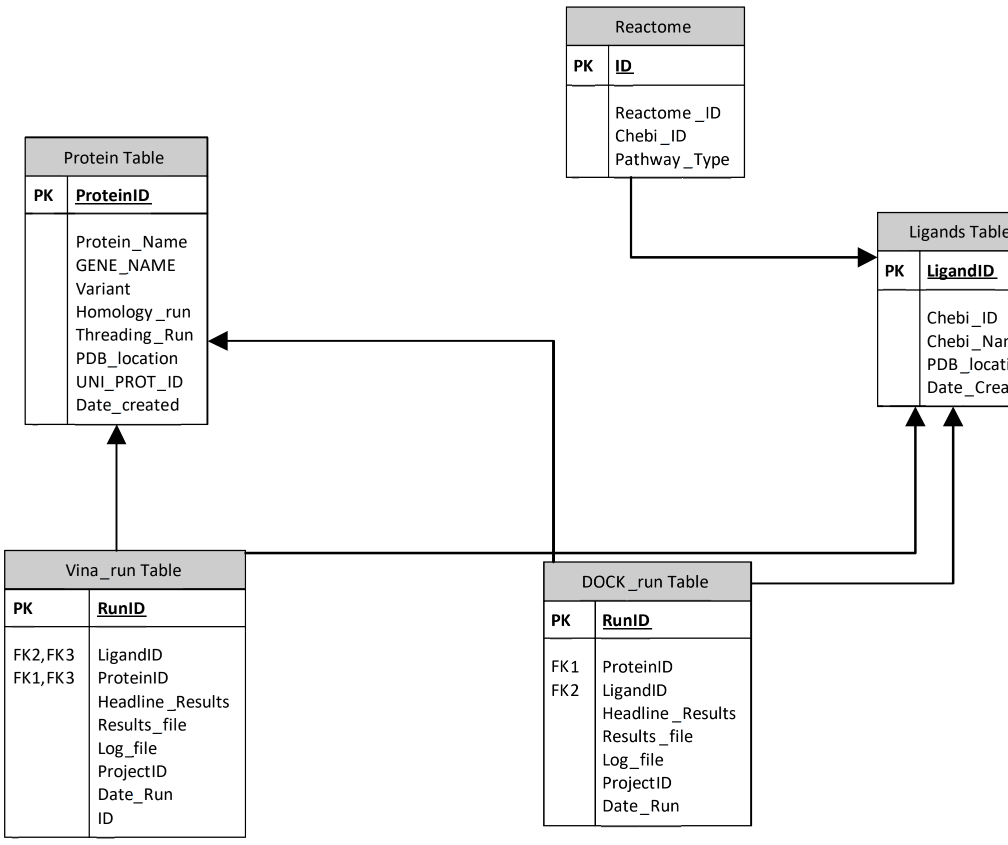


Figure 3.3: Entity Relationship Diagram (ERD) of second database design – Shown is the design for the prototype 2 database structure the Reactome and KEGG data.

Reactome and KEGG numbers with links to multiple proteins within the “Proteins Table”. The “Proteins Table” has also been modified to include three new columns compared to the original design. The new columns are “GENE_NAME”, “Threading_run”, and “PDB location”. The “GENE_NAME” column has been added to store the gene name associated with the protein. The information is helpful for the user to have, and this is another way for the user to search for a protein of interest instead of searching the UniProt accession code or the name of the protein. The inclusion of the “Threading_run” column is necessary as not all the structures would be modelled using homology modelling, meaning that on the interface, it will show the method used to model the protein. If there is a yes in the “Homology_run” column, it shows that the protein modelling is by homology modelling. If there is a yes in the “Threading_run” column, it shows that the protein modelling is via protein threading. Finally, the” PDB_location” is to store the location of the PDB file of the modelled protein, enabling the interface architecture to locate the 3D model so that it can be accessed and displayed.

The way of storing the docking scores are stored on the system involved modification in this version. Because of the use of two different docking algorithms, there are two different scores for every docking produced. To store all this data in one table is somewhat confusing, so in this version, two different tables are being used to store the scores, a “Vina_run Table” and a “DOCK_run Table”. Now the respective score is stored in the corresponding table, and the interface can display both algorithm scores to the user when a specific selection docking is selected.

3.4 Database Prototype 3

Following modifications to the molecular docking approaches, there were more scores that the docking algorithms produced which could be potentially beneficial to the users for their research. Previously the database contained only affinity scores produced by both algorithms. Following the modifications, other parameters generated by the docking algorithms have been included, which allows the calculation of multiple different scores and measurements about each molecular docking, the explanation in more detail is in section 4.5. Also, the algorithms produce up to 10 conformations for each docking calculated, meaning that there was a need to update the way scores are stored on the system so that it can hold multiple scores for multiple conformations for a single docking pair.

Version 2 of the database also needed to be made more efficient in how it stored its data and in having the database set up for the inclusion of further sets of proteins, ligands, and dockings that may require adding to it. The database was substantially re-constructed compared to version 2 and that the necessary information and structure would be present within the database to prepare it for future protein, ligand, and docking data.

3.4.1 Prototype 3 Database re-structure

The first thing studied was the core information which concerns the method of storing the protein and ligand information on the database.

Protein and Ligand Tables re-structure

In the previous version, all the information about proteins is stored in the Proteins table and similarly the ligand information in the Ligands table. The database design is adapted to cope with the problem of duplication within the same table, resulting in unduly increasing the amount of storage used and affecting database querying time. The solution is displayed in figure 3.8 with all key information about proteins in a single table and similarly, all key information about ligands in a single table. The two tables are linked via one-to-many relationships using the "UniProt" column, used in both tables. Many records within the "Proteins" table linked to one record within the "Protein_Info" table. Ensuring that there is no duplication of protein or ligand information within the proteins table, which will save storage space and speed up queries. Shown in table 3.1 are the columns and data that are within the "Protein_Info" table.

By developing the database in this way, it was also possible to add more information about the protein that was previously stored. The previous architecture will have led to substantial duplication of information if all the information above contained in a single proteins table which would have seen the required storage for the database increase massively. Data stored within the "Proteins" table relates to the UniProt code of the protein and information about the 3D model stored on the system. Shown in table 3.2 is the information stored within the "Proteins" table.

The "Ligands" table had the same structure as in prototype 2, as shown in figure 3.8. There had been some modification to the data stored and how it is stored. The "Ligands" table has a table called "Ligand_Info" associated with it. This table holds the PubChem ID which is how the tables link in a query, and the name linked with that ID. The format of the "KEGG" and "Reactome" tables did not change from the previous implementation. The data within the "Ligands" table has changed. Shown in table 3.3 are the columns and information now stored within the "Ligands" table.

Table 3.1: Columns set within the Protein_Info table – shown are the columns that have been set within the Protein_Info table and what type of data that is stored within them.

Column Name	**Data Stored**
UniProt	The column stores the UniProt code of the protein which information is stored in each record and is how the table links to the “Proteins” table.
Gene_Name	The column stores the gene name of the protein within the record.
Protein_Name	The column contains the long, human-readable name of the protein, for example, the name for a protein with UniProt code P36575 is Arrestin-C.
Other_gene_Names	Stores other known gene names associated with a single protein.
Organism	The protein’s organism type is stored in this column. This prototype will only contain Homo sapiens (Human) proteins, but in the future, the evolving system could contain a wider variety of organisms such as mouse and rat.
seq_length	the column stores the length of the sequence of the protein.
Proteomes	this column stores the species name (for example homo sapien)
Sequence	The column contains the full sequence of the protein in FASTA form.

Table 3.2: Data columns for the Proteins table – shown are all the data columns that will be stored within the Proteins table with the type of data that will be stored within them.

Column Name	**Data Stored**
Protein_Index	The column stores the unique ID of each record within the table.
genename	The column contains the gene name of the protein; this is used for linking to other tables.
pdb_file	The column stores the location of the PDB file of each protein stored.
uniprot	The column stores the uniprot code of the protein within the record and is used to link to the "Protein_Info" table.
Variant	The column stores details of variants of the wild-type protein which have been modelled. For the wild type, this contains the word "Wildtype". If an isoform it contains the word "Isoform, or if it is a mutation it stores for example "S235T" which stands for a Serine residue that has undergone a mutation at position 235 into Threonine.
md5	The column stores the md5 checksum is stored for the protein and its model; this will be explained in more detail in the data import system in section 3.5. A checksum is a value used to verify the integrity of a file or a data transfer. It is a sum that's used to check the validity of data.
model_type	The column stores details of whether the model has been created using the homology modelling method or by protein threading.
Homologue_coverage	The column stores the percentage of the homologue coverage of the model produced.
Procheck_res	The column stores the Procheck percentage quality score given to the 3D model.
Date_created	The column stores the date that the 3D model was uploaded to the system.

Table 3.3: Data columns for the Ligands table – shown are all the data columns that will be stored within the Ligands table with the type of data that will be stored within them.

Column Name	**Data Stored**
Lig_Index	this is the unique identifier for each record within the table
pdb_file	this is the location of the PDB model of the ligand
pubchem	this is for the PubChem ID for the ligand, and this column is linked to the attached information tables “KEGG”, “Reactome” and “Ligand_Info”.
drugbank_id	this is the drugbank ID for the ligand as a second identifier as not all ligands will have a PubChem ID.
md5	this is where the md5 code is stored for the protein and its model.
Date_created	This is the date that the 3D model was uploaded to the system.

Docking table re-structure

The next stage was to re-design the storage of docking results in the database. Due to the multiple different scores being produced by the two docking algorithms and more than one conformation produced for each docking pair, there was too much information for it to store in one dockings table. The amount of data could confuse and make it difficult to query. The change made was to create two tables for docking results. One to store the information about the docking pairs and the location of the conformation files; the other to store all scores that are associated with those docking pairs and link the two tables together using a one-to-many relationship. The “Docking_Table” stores all the data about the protein and ligand pairs used for each docking. Shown in table 3.4 are the columns and data stored within the Docking table.

Table 3.4: Data columns for the Docking table – shown are all the data columns that will be stored within the Docking table with the type of data that will be stored within them.

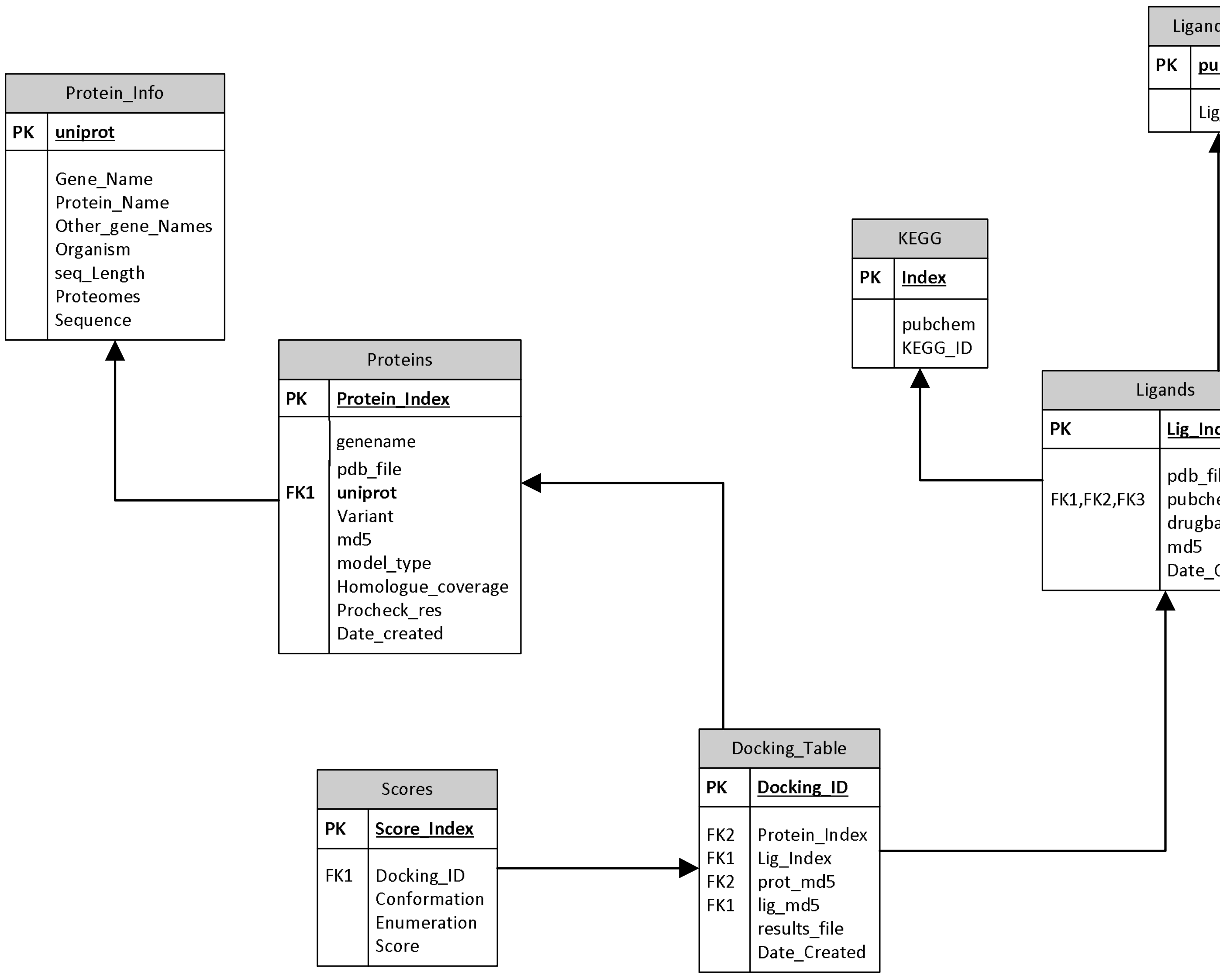
Column Name	Data Stored
Docking_ID	The column stores the unique identifier for each docking pair.
Protein_Index	The column stores the unique identifier for the protein used in the docking. This links with the protein table's Protein_Index column, which when queried gives the details of the protein used.
Lig_Index	The column stores the unique identifier for the ligand used in the docking. This links with the ligands table's Lig_Index column which when queried gives the details of the ligand used.
prot_md5	The column stores the md5 checksum for the protein files, which is used to link the docking table records with the correct protein record within the Proteins table. A checksum is a value used to verify the integrity of a file or a data transfer. It is a sum that's used to check the validity of data
lig_md5	The column stores md5 checksum for the ligand files which is used to link the docking table records with the correct ligand record within the Ligands table. A checksum is a value used to verify the integrity of a file or a data transfer. It is a sum that's used to check the validity of data
results_file	The column stores the location of the conformation file produced by the algorithms.
Date_Created	The column stores the date the conformation was added to the database.

This table is the main link to the interface, and all information is attained through it using a query. The “Score” table stores all the known scores for each of the pairs within the “Docking_Table”. Shown in table 3.5 are the columns and data stored within the “Scores” table. With this structure, it is possible to store a large amount of data without over-complicating or duplicating the data. The structure makes it very efficient to query and get all the appropriate data that is needed. Figure 3.4 shows the full database scheme in an entity-relationship diagram (ERD). This database structure is designed to hold a large amount of data, which is easy to query within the interface to access the correct data.

Table 3.5: Data columns for the Scores table – shown are all the data columns that will be stored within the Scores table with the type of data that will be stored within them.

Column Name	**Data Stored**
Score_Index	The column stores the unique identifier for each score record stored within the table.
Docking_ID	the column links to the Docking_ID within the Dockings table. This is the identifier for all the scores that are related to a docking pair.
Conformation	the docking algorithms produce multiple conformations of the same docking within the docking process. This column stores the conformation associated with that score for a specific docking pair.
Enumeration	the column stores the name of the attribute or parameter that the score is associated with. The docking algorithms that are used produce multiple types of scoring functions relating to a single docking pair. These include docking affinity, electrostatics, internal energy etc. These are discussed in more detail in section 2.3.4. So, for example, if the score was the docking affinity of the conformation, the column would have “Docking Affinity” within its field.
Score	The column scores contain the score associated with the particular conformation and enumeration within the record.

Figure 3.4: Entity-relationship diagram (ERD) of the 3rd and final system database design – shown is the final scheme for the protein-
database contains 8 tables which are the Protein_Info, Proteins, Ligand_Info, Ligands, Reactome, KEGG, Docking_Table, and the Sco



3.5 Data import system

With a large amount of data stored in the database, there needs to be a method of importing large amounts of docking data automatically. The method used will split the data and import the correct data into the correct tables and link the correct data together with the appropriate ID numbers. To make this automated process and to correctly upload the data to the correct tables requires the creation of multiple shell scripts and SQL files. Automating the uploading process lowers the risk of human error and corrupting data. With the developed scripts it has made it possible to upload a large amount of generated protein-ligand docking data and conformations by only launching a single upload script. Shown in figure 3.5 is a flowchart of the process for launching using a single parent script called Import_archive_dockings.sh, which is discussed further in section 3.5.2.5. On launching the parent script, the first script runs the Import_archive_protein.sh script which uploads all protein data and files. This import script is discussed further in section 3.5.2.1. Within the Import_archive_protein.sh script an SQL script called Create_proteins.sql is launched which uploads the formatted data to the database. Section 5.5.2.2 describes the script in more detail. After uploading the proteins, the parent script then launches the Import_archive_ligands.sh which prepares the ligand files and data to be uploaded. The script is discussed further in section 3.5.2.3.

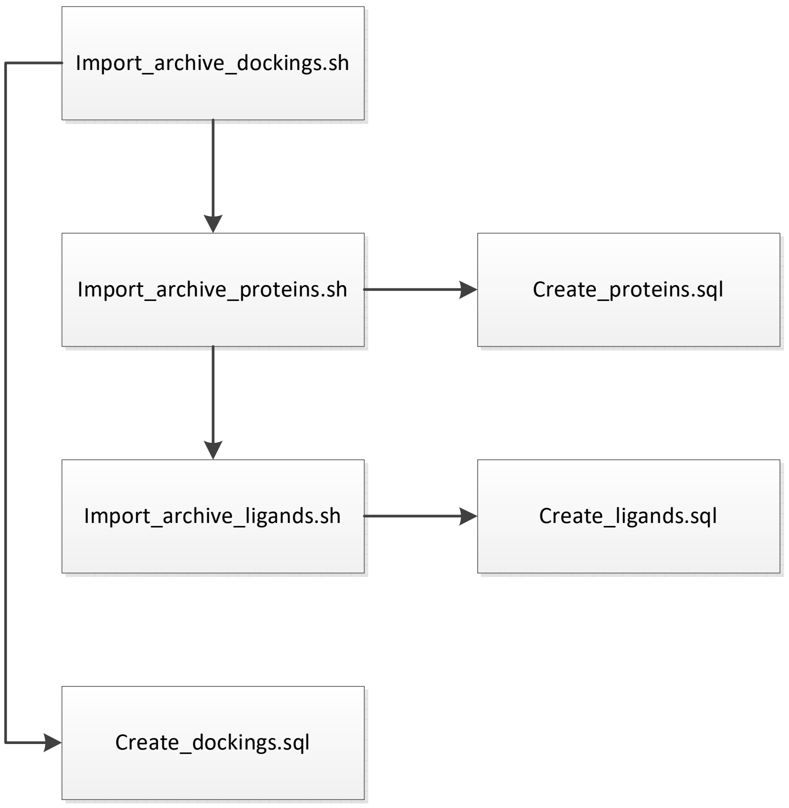

Import_archive_dockings.sh
Import_archive_proteins.sh
Create_proteins.sql
Import_archive_ligands.sh
Create_ligands.sql
Create_dockings.sql

Figure 3.5: Flowchart of the scripts lunched by the parent docking upload script – shown is the flowchart of scripts which are launched by a parent script to upload protein-ligand docking data and conformations.

Following the preparation of the ligand data, the Create_ligands.sql launches, which will upload the data to the database. Section 3.5.2.4 discusses the Create_ligand.sql script further. Finally, after all, proteins and ligands have been uploaded, the Create_dockings.sql script is launched to upload the docking data and to connect the relevant protein and ligand pairs. Section 3.5.2.6 describes the Create_docking.sql script further.

3.5.1 Data produced by docking algorithms

After post-processing the docking data conducted by Shipyard 1.5, a zip file is created which contains all of the protein and ligand files used in the protein-ligand docking, conformation that had been generated by Shipyard 1.5. Each protein-ligand docked pair, and a CSV file which contains all of the scores for every docking score function for each of the ten conformations generated for each protein-ligand docked pair.

3.5.1.1 CSV data file

After post-processing, the next step is creating a CSV file with all data about every molecular docking calculated in that batch. This file contains the protein name, ligand name, what conformation the scores are related to and 23 scoring function results, shown in table 2.1.

3.5.1.2 Conformation files

The post-processed zip file contains three main folders. Shown in figure 3.6 is the folder structure used. One is named _proteins_.zip, which contains the original protein MOL2 files used in the dockings. Another is named _ligands_.zip which contains the original MOL2 files of the ligands used in the dockings. Finally, in the remaining zip files contained within the main zip file are all the conformations calculated by the docking algorithms. Each protein has its zip file named after itself, for example, “sp_uniprot_Gene_HUMAN.zip”. These conformation zip files are named using the ligands PubChem ID number, for example, “pubchemID_3d.mol2”, and contain the ten best conformations produced by the algorithms.

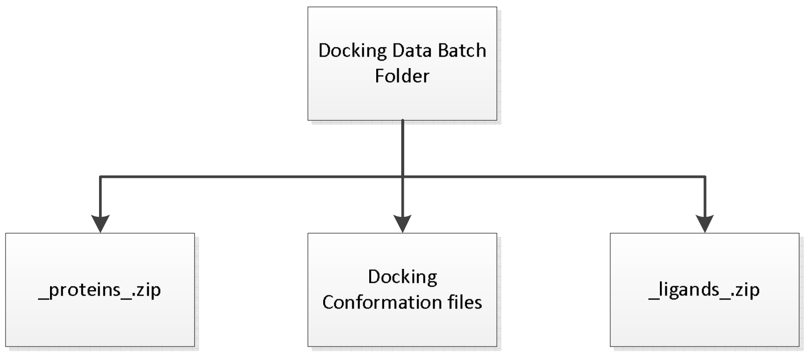


Figure 3.6: Folder structure for Docking Data produced by Shipyard 1.5 – shown is the file structure of the docking data generated by Shipyard 1.5 post-processing. The batch file contains a ligands folder (containing ligand files), Proteins folder (protein files), and a folder containing all conformations generated with a CSV containing protein-ligand conformation scores.

3.5.2 Import scripts

A single command on the command line runs the import scripts. This single script is a shell script which runs other shell scripts within in it and runs MySQL commands. Before explaining the main shell script, there is a need to explain the shell scripts that are launched within it first.

3.5.2.1 Import_archive_proteins.sh

This script collects and prepares all the data for the proteins within the _proteins_.zip file. The folder name that's given with the parent script passes to this script (import_archive_proteins.sh). The script then locates the directory used for the import of data.

Within the import data system, there are two directories to which information passes. The directories are named "Proteins" and "Ligands". For this script, the first step is to copy the _proteins_.zip file to Proteins file in the system. Shown in figure 3.7 is the bash command used to copy the folder.

```
cp (directory name)/_proteins_.zip Proteins/(directory name).zip
```

Figure 3.7:Bash command used to copy _proteins_.zip to the Proteins folder – shown is the bash command used to copy the _proteins_.zip file from the folder containing all docking data to the Proteins file within the system.

When these have been copied over to the Proteins folder, the next stage is to unzip the (directory name).zip file. The unzip process uses the bash command "unzip". The unzipped directory's data transfers to a directory with the name of the original pre-processed folder.

This folder now contains all the protein structure used in the molecular docking in MOL2 format. The viewer used in the user interface is only able to read PDB files, so there is a need to convert the MOL2 files to PDB files. A program called OBABEL is used for the conversion. There is an option in the program to convert MOL2 files to PDB files by using the command shown in figure 3.8.

```
Obabel (filename).MOL2 – opdb -m
```

Figure 3.8:Command used to launch OBABEL – shown is the command used to launch OBABEL which performs a file conversion from MOL2 to PDB for the file inputted into the "file name" tag.

Because all of the MOL2 files in the directory need converting instead of a specific (filename), MOL2 in the command is replaced with *.MOL2 which means that it will go through and convert every MOL2 file within that directory. The 3D viewer that is being used in the interface also needs the structure of the PDB file to be defined. The 3D viewer needs this coordinates file in order to display the protein structure. The assessing of the structure of the protein's PDB file utilises a program called DSSP (Touw et al., 2014). DSSP is a program that has been designed to calculate the most likely secondary structure and then to assign that to regions of a 3D model structure of a protein. Shown in figure 3.9 is the command used to assign the secondary structure to the PDB file.

```
Dssp -i (file name).pdb -o (file name).struct
```

Figure 3.9: Command used to launch DSSP – shown is the command used to launch DSSP which will calculate the secondary structure for the file inputted into the "file name" tag.

The calculation of multiple PDB files in the directory requires a "for loop" to be implemented to go through each PDB file. The next stage of the process is to calculate the checksums of each PDB file. A checksum is a value used to verify the integrity of a file or a data transfer. It is a sum that's used to check the validity of data. Using checksum allows the 3D models within the data to link to the correct data that should be associated. The checksum used is md5. When the checksums have been calculated for each PDB file, a tab-delaminated text file is created with two columns, one for the checksum called md5, and the other is for the filename associated with it. This text file will be used further on in the import process when the dockings are imported. The md5 checksum is calculated using the command shown in figure 3.10.

```
md5sum *.pdb > pdb.md5
```

Figure 3.10: Command used to calculate md5 checksums – Shown is the command used to calculate the md5 checksums for the protein PDB files.

The calculation provides the md5 checksum for every PDB file in the directory and stores it in a file called "pdb.md5". Column names are then added to each of the columns, and the file name is saved as "pdb_checksums.txt". A summary file is also created by storing data about each protein that has a PDB file. The format is that of a tab-delaminated text file with the Filename, Uniprot, and genename. Saving the data under the name "pdb_filenames.txt". The final stage of this script is to copy the "pdb_checksums.txt" and "pdb_filenames.txt" to the appropriate place for adding them to the database. This process uses a MySQL command with a .sql file used to import the data. The command used is shown in figure 3.11.

```
mysql -u (username) -p (password) <create_proteins.sql
```

Figure 3.11:Command to run the create_proteins.sql file – shown is the command used to run the create_proteins.sql within mysql which uploads all generated data and text files about the proteins to the MySQL database.

3.5.2.2 Create_proteins.sql

The script used to import the protein data extracted by the import_archive_proteins.sh script into the database. In order to assist with the import process, the creation of two tables within the database is necessary to store this information. A table called "import_temp_prot_checksums" is created to store the data contained in the "pdb_checksums.txt" file and another table called "import_temp_prot_filenames" which stores the data contained in the "pdb_filenames.txt" file. Shown in table 3.6 are the columns within the Import_temp_lig_checksums database table. Shown in table 3.7 are the columns within the Import_temp_prot_checksums table. Because this import system will be used multiple times, this script first truncates these two tables so that there is no data left from previous imports. The data from the two files are loaded into their respective tables. Following this, a new table is created called "import_temp_prot_both" which is used to link the checksums with the data within the "import_temp_prot_filenames" table. Shown in table 3.8 are the columns that are within the "import_temp_prot_both" table.

Table 3.6:import_temp_prot_checksums tables – shown are the columns within the Import_temp_prot_filename table

Import_temp_prot_filename
filename
uniprot
genename

Table 3.7: import_temp_prot_filename – shown are the columns within the Import_temp_prot_checksums table

Import_temp_prot_checksums
md5
filename

The two tables are joined together by joining them using the filename column of both the "import_temp_prot_filenames" and "import_temp_prot_checksums" table. The SQL command used is shown in figure 3.13. After the import_temp_lig_filenames and import_temp_lig_checksums tables have been joined, a new table is created which will use a "LEFT OUTER JOIN" SQL query to collect all data from the "import_temp_prot_both" table and join it to the Proteins table via the

Table 3.8: Columns within the import_temp_prot_both table – shown are the columns within the import_temp_prot_both table and what data from the "import_temp_prot_filenames" and "import_temp_prot_checksums" tables are stored.

Column Name	**Data**
genename	The column stores the protein name from the "import_temp_prot_filenames" table.
pdb_file	The column stores the filename from the "import_temp_prot_filenames" table
uniprot	The column stores the uniprot ID from the "import_temp_prot_filenames" table.
md5	The column stores the md5 checksum from the "import_temp_prot_checksums" table.

```
JOIN import_temp_prot_filenames on import_temp_prot_checksums.filename =
import_temp_prot_filenames.filename
```

Figure 3.12: Command to join the Import_temp_prot_filenames and import_temp_prot_checksums tables – shown is the command used to join the Import_temp_prot_filenames and import_temp_prot_checksums tables by linking them using the filename column as the link for the records.

"md5" column. During the process of inputting the data into the Proteins table, all data within that table is imports even if it has nothing within the "md5" column resulting in a classification of NULL. Finally, the data within the import_temp_prot_both_new table is inserted into the Proteins table using the Insert command in SQL, shown in figure 3.14.

```
INSERT INTO Proteins

(genename, pdb_file, uniprot, Variant, md5, model_type)

SELECT genename, pdb_file, uniprot, "WildType", md5, "Homology"

FROM import_temp_prot_both_new

WHERE 1;
```

Figure 3.13: Command used to import data from the import_temp_prot_both_new to the Proteins table – shown is the SQL command used to import data stored within the import_temp_prot_both_new table into the Proteins table.

3.5.2.3 Import_archive_ligands.sh

This script is essentially similar to import_archive_proteins.sh shell script, just with some modifications. To prepare the ligand files within the _ligands_.zip file firstly it is copied to the Ligands directory and under the name of the pre-processed file like the Proteins. Then from this stage, the process is the same as the import_archive_proteins.sh shell script. After unzipping the file, the next step is moving the _ligands_directory into the unzipped folder.

Same as for the proteins, the ligand files are in the MOL2 format and need to be converted to PDB format so that the 3D viewer will be able to display them. The command used is shown in figure 3.18. With the ligand PDB files, there is no need for the secondary structure to be assigned so that it can be viewed in the 3D viewer, so this is not included within this script. The next step was to calculate the md5 checksums as for the proteins. The next step is outputting these into a tab-delaminated pdb_checksums.txt with the md5 checksums and the filename associated with it. An import summary file is also generated but with the filename and PubChem columns.

The next step is outputting the columns are into a tab- delaminated pdb_filenames.txt file followed by copying to the relevant area allowing uploading to the database. MySQL is used to upload the data using a SQL file called "create_ligands.sql".

3.5.2.4 Create_ligands.sql

This SQL file is identical to the "Create_proteins.sql" file. Creating the same tables, but instead of having the word prot in them, replacing it with the word lig. The columns appearing in the tables vary according to the requirements.

For the ligands, within the database are two tables that have been created to store the data produced by the "import_archive_ligand.sh" file. These are called "import_temp_lig_filename" and "import_temp_lig_checksums". As with the proteins, truncating these tables is necessary because of the multiple usages of the script. Failing to do this will cause duplication of data within the database. Table 3.9 and 3.10 illustrate the tables. The "import_temp_lig_filename" will have the data within the "pdb_filenames.txt" loaded into it, and the "import_temp_lig_checksums" table will have the data contained in the "pdb_checksums.txt" file loaded into it. A new table is then

Table 3.9:import_temp_prot_checksums tables – shown are the columns within the Import_temp_prot_filename table

Import_temp_lig_filename
filename
pubchem

Table 3.10: import_temp_prot_filename – shown are the columns within the Import_temp_prot_checksums table

Import_temp_lig_checksums
md5
filename

created to join the "import_temp_lig_filename" and the "import_temp_lig_checksums" tables together. This table is called "import_temp_lig_both". The resulting table contains the columns and data shown in table 3.11.

Table 3.11: Columns within the import_temp_lig_both table – shown are the columns within the import_temp_lig_both table and what data from the "import_temp_lig_filenames" and "import_temp_lig_checksums" tables are stored.

Column Name	Data Stored
pdb_file	this is the filename from the "import_temp_lig_filenames" table
pubchem	This is the PubChem ID from the "import_temp_lig_filenames" table.
md5	This is the md5 checksum from the "import_temp_lig_checksums" table.

As for "create_proteins.sql" file, the two tables are joined together by using the filename column of both the "import_temp_lig_filenames" and "import_temp_lig_checksums" table. Shown in figure 3.14 is the join SQL command used.

```
JOIN import_temp_lig_filenames on import_temp_lig_checksums.filename =
import_temp_lig_filenames.filename
```

Figure 3.14: Command to join the Import_temp_lig_filenames and import_temp_lig_checksums tables – shown is the command used to join the Import_temp_lig_filenames and import_temp_lig_checksums tables by linking them using the filename column as the link for the records.

After the import_temp_lig_filenames and import_temp_lig_checksums tables have been joined, creating a new table which will use a "LEFT OUTER JOIN" SQL query to collect all data from the "import_temp_lig_both" table and join it to the Ligands table via the "md5" column. So that when inputting the data into the Ligands table, it will import all data within that table even if it has nothing within the "md5" column which instead classifies it as a NULL.

Finally, like the "create_ligands.sql" file the data within the import_temp_lig_both_new table is inserted into the Ligands table using the Insert command in SQL. Figure 3.15 illustrates this.

```
INSERT INTO Ligands(pdb_file, pubchem, md5)
SELECT pdb_file, pubchem, md5
FROM import_temp_lig_both_new
WHERE 1;
```

Figure 3.15: Command used to import data from the import_temp_lig_both_new to the Ligands table – shown is the SQL command used to import data stored within the import_temp_lig_both_new table into the Ligands table.

3.5.2.5 Import_archive_docking.sh

Import_archive_docking.sh is the parent shell script file which will run the entire import sequence. When an import is needed, the script shown in figure 3.16 is run.

```
./import_archive_dockings.sh (data directory)
```

Figure 3.16: command used to launch the protein-ligand docking import – shown id the command used to launch the data import for the data directory inserted into the "data directory" tag.

The data directory is the directory which contains all the docking algorithm data. The shell script takes this argument and uses it as the location of the directory. The first stage is to run the "import_archive_proteins.sh" and "import_archive_ligands.sh" shell script as spoken about previously with the directory selected. Figure 3.17 shows the code.

```
v1=$1
archivename=${v1%/*}
./import_archive_proteins.sh ${v1}
./import_archive_ligands.sh ${v1}
```

Figure 3.17: commands launched within the Import_archive_docking.sh master script – shown is the code used within the Import_archive_docking.sh file which launches the import scripts for the proteins and the ligands in a sequence.

When these have been run and completed the master script then handles the data directory so that it can access the CSV file called "summary.csv" that the docking algorithm post-processing had generated for the data imported. Not all the data within the "summary.csv" file is required so before the upload there is a need to slim down the CSV file and only have the data that the database requires. Using the awk command, the required data is extracted from the "summary.csv" file and outputted to a new text file called "newsummary.txt". Shown in table 3.12 are the only results needed from the "summary.csv" file. The format of the file is also changed. The names of what the scores represent, for example, "Vina_affinity," classed as a column name in the previous CSV file and placed within their column called Enumeration, for the scores to be able to be inputted into the scores table correctly. An example of the converted CSV file to be uploaded to the database shown in figure 3.19. This conversion is saved as a new text file called "summaryscore.txt". When the conversion process has completed, the "newsummary.txt" file and "newscores.txt" file are copied to the appropriate area for uploading to the database. Then these are uploaded to the MySQL database using the command and SQL file shown in figure 3.18.

3.5.2.6 create_dockings.sql

This SQL script is critical and creates all the correct links necessary between all of the main tables within the database, which are the Proteins, Ligands, Docking, and Scores tables. There are multiple temporary tables used within this script to prepare all the data. The first stage is to load the two text files created by the "Import_archive_dockings.sh" script into the database. There are two temporary tables created to store this data, one is called "Import_temp_dockings", and the other is "Import_temp_dockscores". The "import_temp_dockings" table will store the data within the newsummary.txt file, and the "import_temp_dockscores" will store the data within the newscores.txt file. Both tables are truncated before every import so that there is no duplication or confusion of data due to this script being used multiple times when uploading new data. Loading the data begins the linking process one to another. Creating a table called "import_temp_dockings_md5" is the first step.

Table 3.12: Columns within the "summary.csv" file – shown are the columns that are stored within the "summary.csv" file which will be extracted and converted into a new file called "newsummary.txt"

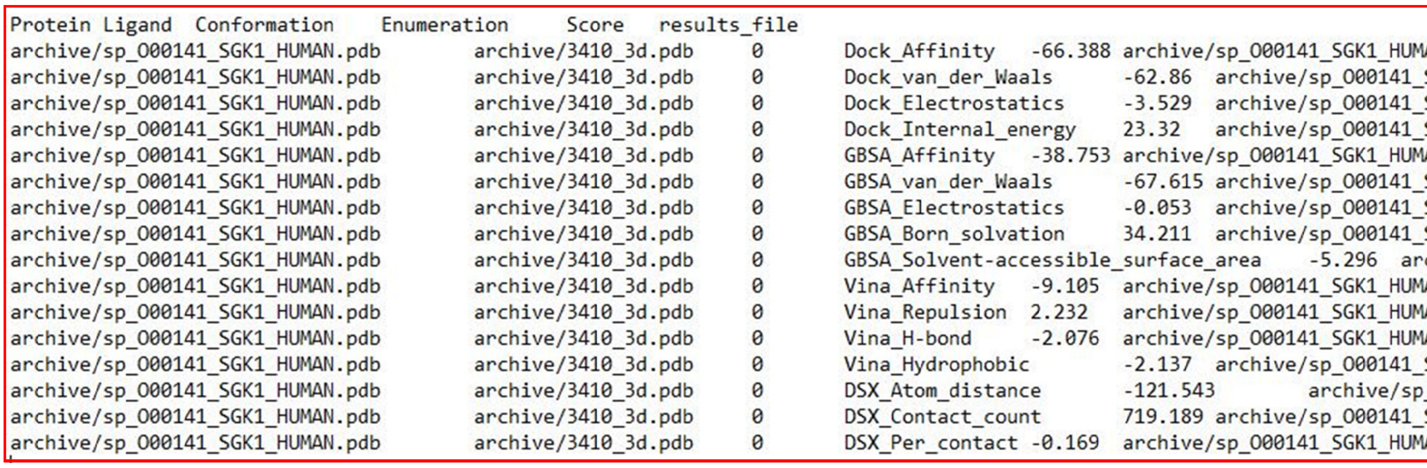
Column Names
Protein
Ligand
Conformation
DOCK_Affinity
DOCK_van_der_Waals
DOCK_Electrostatics
DOCK_Internal_energy
GBSA_Affinity
GBSA_van_der_Waals
GBSA_Screened_electrostatics
GBSA_Born_solvation
GBSA_Solvent-accessible_surface_area
Vina_Affinity
Vina_Repulsion
Vina_Hydrophobic
Vina_H-bond
Vina_rotatable_bond_entropy
Drug Score X(DSX)_Atom_distance
DSX_Contact_count
DSX_Per_contact
Results_file

```
mysql -u (username) -p (password) < create_dockings.sql
```

Figure 3.18: SQL code to upload docking data – this is the code used to upload the docking data to the MySQL database.

This table will link the "import_temp_dockscores" table results with their related Proteins and Ligands within the "import_temp_prot_checksums" and "import_temp_lig_checksums" tables created in the previous step when loading the proteins and ligand data to the database. The next step is combining these using the "filename" column as the link. The SQL query shown in figure 3.20 is used to create the "import_temp_dockings_md5" table.

Figure 3.19:Example of a converted CSV file – Shown is an example of how the converted CSV file which will be used to upload the protein-ligand



```
Protein Ligand  Conformation    Enumeration     Score   results_file
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       Dock_Affinity   -66.388 archive/sp_O00141_SGK1_HUM
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       Dock_van_der_Waals      -62.86  archive/sp_O00141_
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       Dock_Electrostatics     -3.529  archive/sp_O00141_
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       Dock_Internal_energy    23.32   archive/sp_O00141_
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       GBSA_Affinity   -38.753 archive/sp_O00141_SGK1_HUM
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       GBSA_van_der_Waals      -67.615 archive/sp_O00141_
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       GBSA_Electrostatics     -0.053  archive/sp_O00141_
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       GBSA_Born_solvation     34.211  archive/sp_O00141_
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       GBSA_Solvent-accessible_surface_area    -5.296  ar
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       Vina_Affinity   -9.105  archive/sp_O00141_SGK1_HUM
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       Vina_Repulsion  2.232   archive/sp_O00141_SGK1_HUM
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       Vina_H-bond     -2.076  archive/sp_O00141_SGK1_HUM
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       Vina_Hydrophobic        -2.137  archive/sp_O00141_
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       DSX_Atom_distance       -121.543        archive/sp
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       DSX_Contact_count       719.189 archive/sp_O00141_
archive/sp_O00141_SGK1_HUMAN.pdb        archive/3410_3d.pdb     0       DSX_Per_contact -0.169  archive/sp_O00141_SGK1_HUM
```

CREATE TABLE import_temp_docking_md5 AS (

SELECT Conformation, Enumeration, Score, results_file, import_temp_lig_checksums.md5 AS lig_md5, import_temp_p

FROM import_temp_dockscores

JOIN import_temp_prot_checksums

ON import_temp_prot_checksums.filename = import_temp_dockscores.Protein

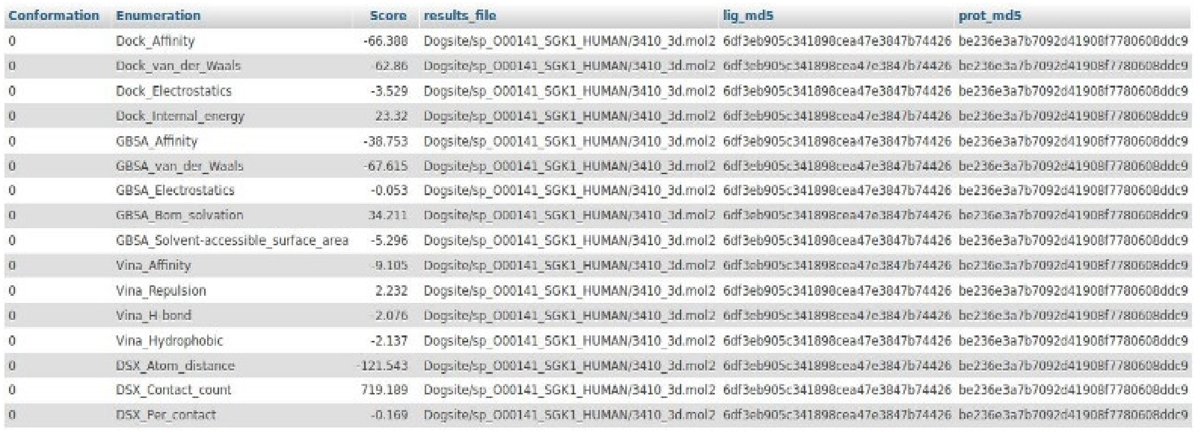
JOIN import_temp_lig_checksums

ON import_temp_lig_checksums.filename = import_temp_dockscores.Ligand

WHERE 1);

Figure 3.20: Command used to create the import_temp_docking_md5 table – shown is the SQL command used to create the import_temp_docki
the import_temp_dockscores table and joining data from the import_temp_prot_checksums and import_temp_lig_checksums via the filename c

The SQL command creates the table, as the example in figure 3.21 demonstrates. The "dockscores" table is linked with the proteins and ligands used with each docking but using the md5 checksums as the protein and ligand identifiers. Creating the table leads on to the next operation of the script to insert the data from the "import_temp_dockings" data into the Dockings table. Through the INSERT SQL command, the data from this table is also joined to the "import_temp_prot_checksums" and "import_temp_lig_checksums". So, this means that the md5 checksums of the protein and ligands are used to identify with the correct scores within the Docking table instead of the file locations. The code used is shown in figure 3.22.



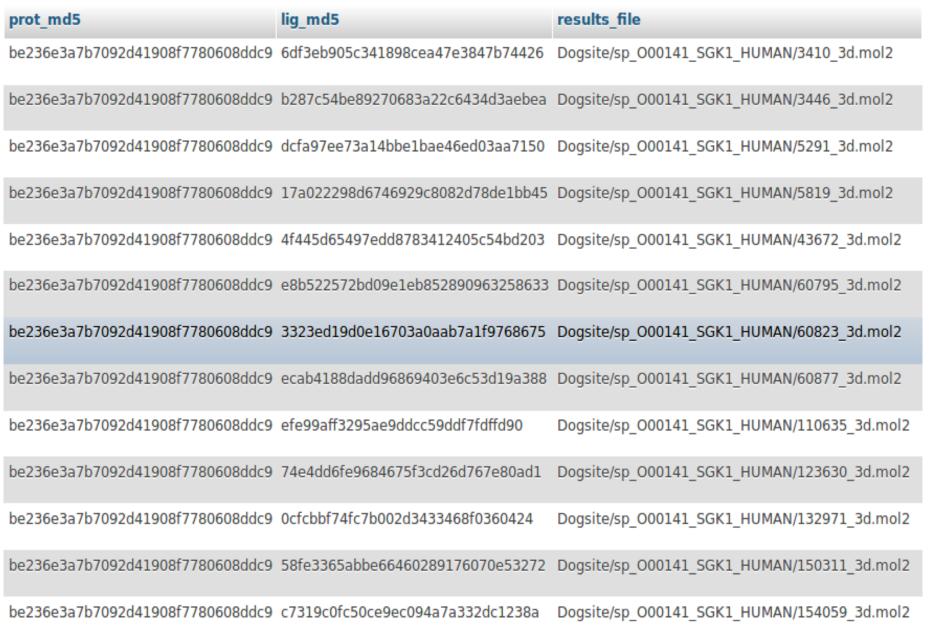
Conformation	Enumeration	Score	results_file	lig_md5	prot_md5
0	Dock_Affinity	-66.388	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	Dock_van_der_Waals	-62.86	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	Dock_Electrostatics	-3.529	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	Dock_Internal_energy	23.32	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	GBSA_Affinity	-38.753	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	GBSA_van_der_Waals	-67.615	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	GBSA_Electrostatics	-0.053	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	GBSA_Born_solvation	34.211	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	GBSA_Solvent-accessible_surface_area	-5.296	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	Vina_Affinity	-9.105	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	Vina_Repulsion	2.232	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	Vina_H_bond	-2.076	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	Vina_Hydrophobic	-2.137	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	DSX_Atom_distance	-121.543	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	DSX_Contact_count	719.189	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9
0	DSX_Per_contact	-0.169	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2	6df3eb905c341898cea47e3847b74426	be236e3a7b7092d41908f7780608ddc9

Figure 3.21: Example of the import_temp_docking_md5 table created – shown is an example of the created import_temp_docking_md5 after running the SQL command shown in figure 3.24

```
INSERT INTO Docking_Table
(prot_md5, lig_md5, results_file)
SELECT import_temp_prot_checksums.md5, import_temp_lig_checksums.md5, results_file
FROM import_temp_dockings
JOIN import_temp_prot_checksums
ON import_temp_prot_checksums.filename = import_temp_dockings.Protein
JOIN import_temp_lig_checksums
ON import_temp_lig_checksums.filename = import_temp_dockings.Ligand
WHERE 1;
```

Figure 3.22: Command used to import data into the Docking_Table – shown is the SQL code used to link the correct protein and ligand to the correct protein- ligand docking pairs within the Docking_Table by using the md5 checksums.

This script uploads the data into the related columns as shown in figure 3.23, the md5 codes in the lig_md5 and prot_md5 are used to link with the Protein and Ligand tables at the end of the script. When the following data insert has completed, the next stage of the script is to insert the docking scores data into the "Scores" table. With the insert of this data, there is a need to link the scores to the docking table using the Docking ID of the records with this table. To do this the md5 checksums



prot_md5	lig_md5	results_file
be236e3a7b7092d41908f7780608ddc9	6df3eb905c341898cea47e3847b74426	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2
be236e3a7b7092d41908f7780608ddc9	b287c54be89270683a22c6434d3aebea	Dogsite/sp_O00141_SGK1_HUMAN/3446_3d.mol2
be236e3a7b7092d41908f7780608ddc9	dcfa97ee73a14bbe1bae46ed03aa7150	Dogsite/sp_O00141_SGK1_HUMAN/5291_3d.mol2
be236e3a7b7092d41908f7780608ddc9	17a022298d6746929c8082d78de1bb45	Dogsite/sp_O00141_SGK1_HUMAN/5819_3d.mol2
be236e3a7b7092d41908f7780608ddc9	4f445d65497edd8783412405c54bd203	Dogsite/sp_O00141_SGK1_HUMAN/43672_3d.mol2
be236e3a7b7092d41908f7780608ddc9	e8b522572bd09e1eb852890963258633	Dogsite/sp_O00141_SGK1_HUMAN/60795_3d.mol2
be236e3a7b7092d41908f7780608ddc9	3323ed19d0e16703a0aab7a1f9768675	Dogsite/sp_O00141_SGK1_HUMAN/60823_3d.mol2
be236e3a7b7092d41908f7780608ddc9	ecab4188dadd96869403e6c53d19a388	Dogsite/sp_O00141_SGK1_HUMAN/60877_3d.mol2
be236e3a7b7092d41908f7780608ddc9	efe99aff3295ae9ddcc59ddf7fdffd90	Dogsite/sp_O00141_SGK1_HUMAN/110635_3d.mol2
be236e3a7b7092d41908f7780608ddc9	74e4dd6fe9684675f3cd26d767e80ad1	Dogsite/sp_O00141_SGK1_HUMAN/123630_3d.mol2
be236e3a7b7092d41908f7780608ddc9	0cfcbbf74fc7b002d3433468f0360424	Dogsite/sp_O00141_SGK1_HUMAN/132971_3d.mol2
be236e3a7b7092d41908f7780608ddc9	58fe3365abbe66460289176070e53272	Dogsite/sp_O00141_SGK1_HUMAN/150311_3d.mol2
be236e3a7b7092d41908f7780608ddc9	c7319c0fc50ce9ec094a7a332dc1238a	Dogsite/sp_O00141_SGK1_HUMAN/154059_3d.mol2

Figure 3.23: Example of the Docking table with updated md5 checksums – shown is the updated Dockings_Table with the md5 checksums for the proteins and ligands which links to the Proteins and Ligands tables.

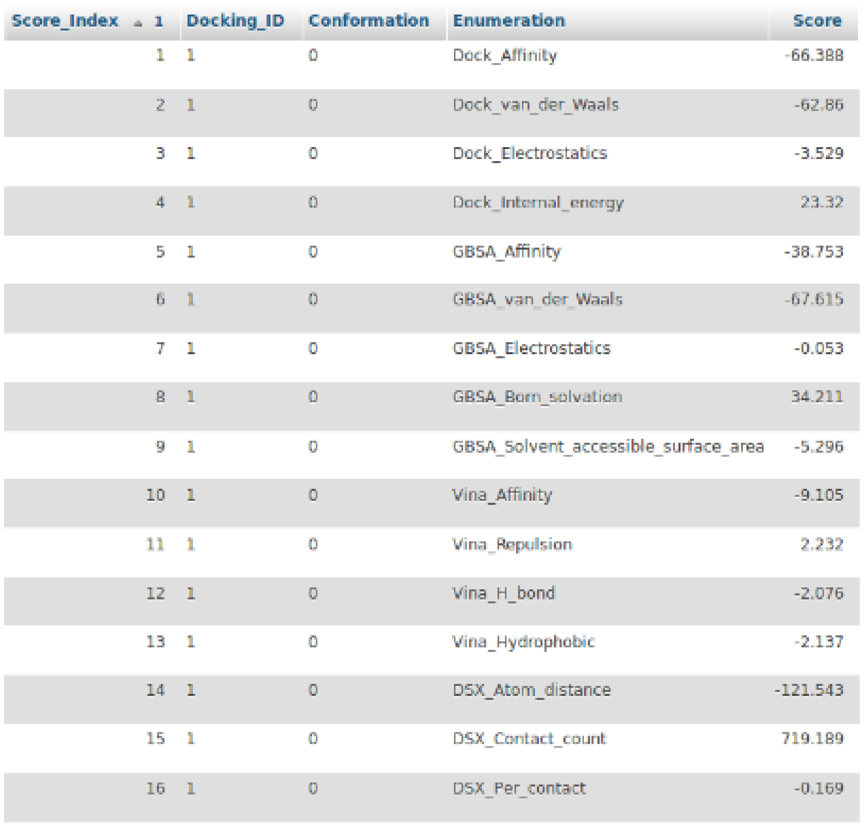
of both the Docking table and the temporary table "import_temp_dockings_md5" are used to link the docking and Scores tables together. Figure 3.24 shows the SQL insert command.

Now within the Scores table, each record has an associated Docking_ID so that when docking is requested the Docking_ID of the requested docking can be used to collect all associated scores from the Scores table for that ID. Figure 3.25 shows an example of the table data created in the Scores table.

```
INSERT INTO Scores
(Docking_ID, Conformation, Enumeration, Score)
SELECT Docking_Table.Docking_ID, Conformation, Enumeration, Score
FROM import_temp_dockings_md5
JOIN Docking_Table
ON Docking_Table.prot_md5 = import_temp_dockings_md5.prot_md5
AND Docking_Table.lig_md5 = import_temp_dockings_md5.lig_md5
WHERE 1;
```

Figure 3.24:Command used to insert data into the Scores table – shown is the SQL command used to upload docking score data into the Scores table and linking the data to the correct protein-ligand record within the Docking_Table.

This stage involves uploading all the essential data to the database. The final stage is now to link the Dockings table with the Ligands and Proteins tables. The md5 checksums within the Dockings table are used to verify that the correct protein and ligand index ID's are being inputted into the correct records. An UPDATE SQL command and a JOIN command is used so that the correct Index ID's are updated into the table. Shown in figure 3.26 is the code used to perform the update.



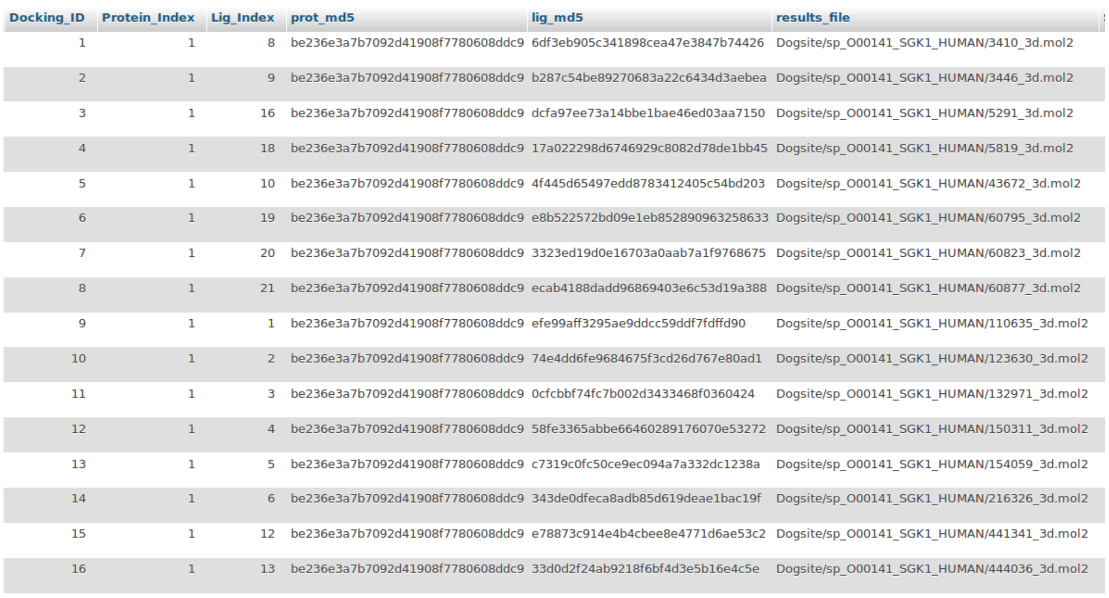
Score_Index ▲ 1	Docking_ID	Conformation	Enumeration	Score
1	1	0	Dock_Affinity	-66.388
2	1	0	Dock_van_der_Waals	-62.86
3	1	0	Dock_Electrostatics	-3.529
4	1	0	Dock_Internal_energy	23.32
5	1	0	GBSA_Affinity	-38.753
6	1	0	GBSA_van_der_Waals	-67.615
7	1	0	GBSA_Electrostatics	-0.053
8	1	0	GBSA_Born_solvation	34.211
9	1	0	GBSA_Solvent_accessible_surface_area	-5.296
10	1	0	Vina_Affinity	-9.105
11	1	0	Vina_Repulsion	2.232
12	1	0	Vina_H_bond	-2.076
13	1	0	Vina_Hydrophobic	-2.137
14	1	0	DSX_Atom_distance	-121.543
15	1	0	DSX_Contact_count	719.189
16	1	0	DSX_Per_contact	-0.169

Figure 3.25: Example of data within Scores Table – shown is the Scores table after inserting docking scores using the SQL command shown in figure 3.24.

```
UPDATE `Docking_Table`
JOIN Proteins on Docking_Table.prot_md5 = Proteins.md5
SET Docking_Table.Protein_Index = Proteins.Protein_Index
WHERE 1;
UPDATE `Docking_Table`
JOIN Ligands on Docking_Table.lig_md5 = Ligands.md5
SET
Docking_Table.Lig_Index = Ligands.Lig_Index
WHERE 1;
```

Figure 3.26: Code used to update Dockings_Table – shown above is the code used to update the Dockings_Table with the md5 codes for the proteins and ligand pairs and their Unique ID which links the table to the Proteins and Ligands Tables.

An example of the final data filled table is shown in figure 3.27 which shows the Protein_Index and Lig_Index columns with the correct Index ID is that link to the Protein_Index and Lig_Index ID's within the Proteins and Ligand Tables. The data upload process is now completed.



Docking_ID	Protein_Index	Lig_Index	prot_md5	lig_md5	results_file
1	1	8	be236e3a7b7092d41908f7780608ddc9	6df3eb905c341898cea47e3847b74426	Dogsite/sp_O00141_SGK1_HUMAN/3410_3d.mol2
2	1	9	be236e3a7b7092d41908f7780608ddc9	b287c54be89270683a22c6434d3aebea	Dogsite/sp_O00141_SGK1_HUMAN/3446_3d.mol2
3	1	16	be236e3a7b7092d41908f7780608ddc9	dcfa97ee73a14bbe1bae46ed03aa7150	Dogsite/sp_O00141_SGK1_HUMAN/5291_3d.mol2
4	1	18	be236e3a7b7092d41908f7780608ddc9	17a022298d6746929c8082d78de1bb45	Dogsite/sp_O00141_SGK1_HUMAN/5819_3d.mol2
5	1	10	be236e3a7b7092d41908f7780608ddc9	4f445d65497edd8783412405c54bd203	Dogsite/sp_O00141_SGK1_HUMAN/43672_3d.mol2
6	1	19	be236e3a7b7092d41908f7780608ddc9	e8b522572bd09e1eb852890963258633	Dogsite/sp_O00141_SGK1_HUMAN/60795_3d.mol2
7	1	20	be236e3a7b7092d41908f7780608ddc9	3323ed19d0e16703a0aab7a1f9768675	Dogsite/sp_O00141_SGK1_HUMAN/60823_3d.mol2
8	1	21	be236e3a7b7092d41908f7780608ddc9	ecab4188dadd96869403e6c53d19a388	Dogsite/sp_O00141_SGK1_HUMAN/60877_3d.mol2
9	1	1	be236e3a7b7092d41908f7780608ddc9	efe99aff3295ae9ddcc59ddf7fdffd90	Dogsite/sp_O00141_SGK1_HUMAN/110635_3d.mol2
10	1	2	be236e3a7b7092d41908f7780608ddc9	74e4dd6fe9684675f3cd26d767e80ad1	Dogsite/sp_O00141_SGK1_HUMAN/123630_3d.mol2
11	1	3	be236e3a7b7092d41908f7780608ddc9	0cfcbbf74fc7b002d3433468f0360424	Dogsite/sp_O00141_SGK1_HUMAN/132971_3d.mol2
12	1	4	be236e3a7b7092d41908f7780608ddc9	58fe3365abbe66460289176070e53272	Dogsite/sp_O00141_SGK1_HUMAN/150311_3d.mol2
13	1	5	be236e3a7b7092d41908f7780608ddc9	c7319c0fc50ce9ec094a7a332dc1238a	Dogsite/sp_O00141_SGK1_HUMAN/154059_3d.mol2
14	1	6	be236e3a7b7092d41908f7780608ddc9	343de0dfeca8adb85d619deae1bac19f	Dogsite/sp_O00141_SGK1_HUMAN/216326_3d.mol2
15	1	12	be236e3a7b7092d41908f7780608ddc9	e78873c914e4b4cbee8e4771d6ae53c2	Dogsite/sp_O00141_SGK1_HUMAN/441341_3d.mol2
16	1	13	be236e3a7b7092d41908f7780608ddc9	33d0d2f24ab9218f6bf4d3e5b16e4c5e	Dogsite/sp_O00141_SGK1_HUMAN/444036_3d.mol2

Figure 3.27: Image of the Docking table after the data import had completed – shown is an example of the complete Docking_Table after all data imports have completed. The Protein_Index and Lig_Index columns are now populate with the relevant ID's for each docking pair record.

3.6 Application Programming Interface (API)

The Application Programming Interface is web-based with communication over HTTP. The API will allow users to be able to access the database without a need to download or install any specific software. The end-user only needs a web browser and an internet connection, making the database widely available and accessible to a wide variety of different devices.

The Graphical User Interface (GUI) is coded using HTML and JavaScript. The background code will link the GUI to the SQL Database by using AJAX function calls and PHP coded files. Section 5.3 has the implementation and a more detailed explanation of the use of the API.

3.7 Conclusion

The system architecture developed can store a large amount of data produced by the docking predictions. The task was to create a database that could store the data relating to proteins and drugs and their respective 3D models, each docking pair and their respective conformation scores, and each predicted conformation file. There were many versions of the database structure design due to the relentless increase in the amount of information to be stored. As discussed in section 3.3, the original architecture for the database shown in figure 3.2 was overly simple and did not incorporate the extra information that would be required to satisfy users of the system. Basing the original design on only using AutoDock Vina and DOCK 6.0 results. After revision of the original design, the second prototype shown in figure 3.3 had an additional table for the DOCK 6.0 results and new tables for data relating to the drugs within the database from the KEGG and Reactome databases. New columns were added to tables that already existed, for example, in the protein table, a column noting whether a model was created via homology modelling or protein threading.

A third prototype of the database was developed after evaluating the second prototype database, which is shown in figure 3.4. As discussed in section 3.4.1, the evaluation revealed that duplicate information was being stored for the proteins as there could often be different models generated and considered for the same protein query. A decision was made to introduce a separate table that would only store data for one version of every individual protein and that this would be linked to the protein tables using the UniProt code — doing this saved storage space due to the elimination of duplicate data, therefore releasing more storage for future addition of richer data. Due to different models representing the same protein, introducing an md5 checksum column ensures that each protein links to the correct 3D model uploaded with it to the system. The same process applies to the ligand data and 3D models.

The final change from the second prototype database was the method of storing the docking scores. The results produced by Shipyard 1.5's pipeline calculates around 19 different scoring functions. Instead of having each row of data in the docking table having 20 columns of data each and placing all the scores stored in a separate scores table which is linked to the docking table using the docking ID index. The new table helps to increase the querying speed and leads to less scope for confusion in the viewing of the dockings table.

This database is a robust architecture that was able to handle a large amount of docking data produced by the docking programs without any trouble and made it possible for the web-based interface to be used to query the broader data to obtain the specific information that the user requires efficiently.

4 Protein Modelling and Molecular docking results

4.1 Introduction

This chapter will discuss the process and analysis undertaken for protein modelling and molecular docking. The molecular docking is analysed by using the known targets and misses for the 20 most valuable drugs and to determine if they are identified correctly. Information about the known protein targets for the 20 most valuable sold drugs is stored on the DrugBank (Wishart et al., 2017) and TOXCAST (Sipes et al., 2013). The scores for these docking pairs were extracted from the very substantial amount of data created by AutoDock Vina and DOCK 6.0 and then analysed to see how the predicted docking performed. A baseline will be set by modelling the proteins and molecular docking them to the 20 most valuable sold drugs using the original pipeline with no modifications.

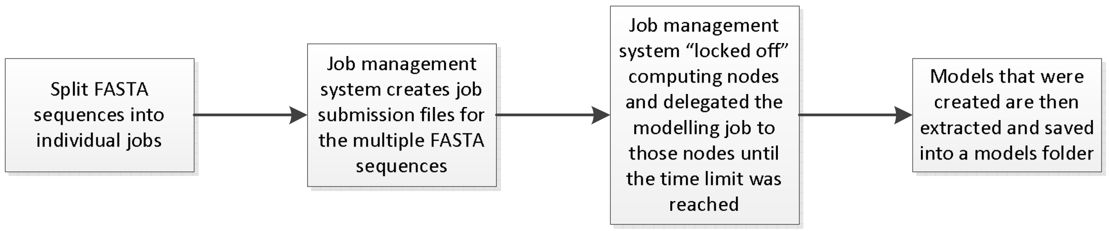
4.2 Protein Modelling

The first stage of the protein modelling process is to collect and collate the FASTA sequences for all proteins in the human proteome. These FASTA sequences were downloaded from the UniProt database (UniProt) which has 20,160 human proteome sequences. The FASTA sequences download consisted of one large FASTA file which contained all 20,160 FASTA sequences. As there were so many sequences to model, it was not possible to model 20,160 FASTA sequences simultaneously. A python script to split the FASTA file into seven different files which would contain sequences based on their number of amino acids. Shown in Table 4.1 are the 7-different sequence length categories applied.

Table 4.1: Sequence length categories – the table shows all of the different categories the 20,160 FASTA sequences were split into for the modelling process and the amount of sequences that were in each category

Category	**Number Of sequences**
500 or less	12,152
500 – 800	4,318
800 – 1000	1,304
1000 – 1200	803
1200 – 1500	652
1500 – 2000	483
2000 +	448

Shorter sequences were uploaded first to enhance progress with the HPC Wales computing resources as they take less time to run than the more extended sequences. As there were a large number of sequences with 500 or less amino acid sequences, these were further split into batches of 2000 and uploaded onto HPC Wales to be modelled using the homology modelling pipeline. The multiple sequences in these groups could be modelled in parallel, which increased the number of proteins modelled at once. Shown in figure 4.1 is the process that was carried out and described in more detail below. Each sequence usually takes around 20 minutes each for the 3D modelling to complete.

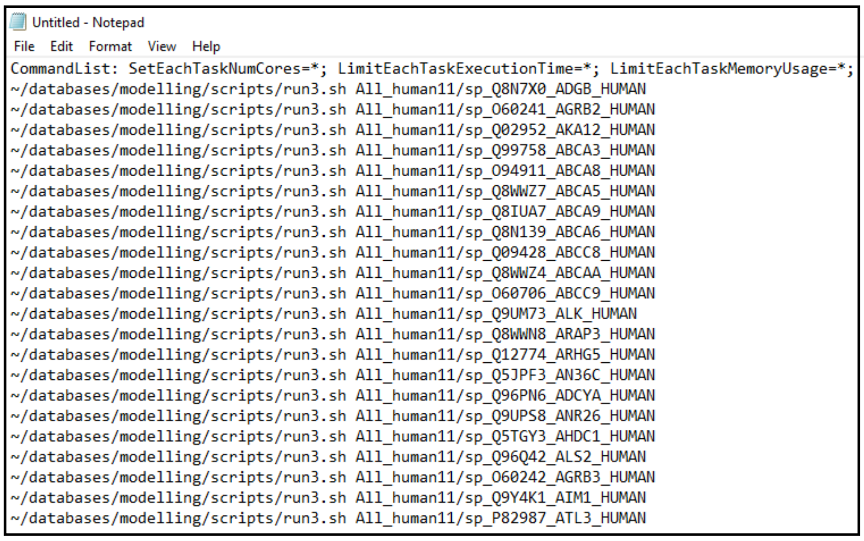


Split FASTA sequences into individual jobs
Job management system creates job submission files for the multiple FASTA sequences
Job management system "locked off" computing nodes and delegated the modelling job to those nodes until the time limit was reached
Models that were created are then extracted and saved into a models folder

Figure 4.1: Modelling set up and process flowchart – the flowchart shows the four stages that were needed for the homology modelling pipeline to be run on HPC Wales. First the FASTA sequences are split into individual jobs, then the job management system creates job submission scripts that will be used to execute the homology modelling pipeline on HPC Wales. When the job management system has access to a node it the "locks off" that node and delegates modelling jobs to it until the time limit on that node runs out. Finally, all of the 3D models created are then moved to a model's folder.

A Perl script was created to split the sequences into specific jobs and copy and define the relevant files to run the homology modelling pipeline. Within this, a CSV file was created with all the modelling jobs that need to be undertaken. Shown in figure 4.2 is an example of one of these CSV files.

This approach is used for the job submission to the HPC node clusters. Because of the number of sequences, to run the jobs in parallel, a job management system was used to help distribute jobs to



```
CommandList: SetEachTaskNumCores=*; LimitEachTaskExecutionTime=*; LimitEachTaskMemoryUsage=*;
~/databases/modelling/scripts/run3.sh All_human11/sp_Q8N7X0_ADGB_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_O60241_AGRB2_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q02952_AKA12_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q99758_ABCA3_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_O94911_ABCA8_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q8WWZ7_ABCA5_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q8IUA7_ABCA9_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q8N139_ABCA6_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q09428_ABCC8_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q8WWZ4_ABCAA_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_O60706_ABCC9_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q9UM73_ALK_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q8WWN8_ARAP3_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q12774_ARHG5_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q5JPF3_AN36C_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q96PN6_ADCYA_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q9UPS8_ANR26_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q5TGY3_AHDC1_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q96Q42_ALS2_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_O60242_AGRB3_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_Q9Y4K1_AIM1_HUMAN
~/databases/modelling/scripts/run3.sh All_human11/sp_P82987_ATL3_HUMAN
```

Figure 4.2: Example of CSV job submission file – at the top of the file are the options that need to be determined for the job management system. These options are "SetEachTaskNumCores" which tell the management system how many tasks it should launch on each single core of the node. The "LimitEachTaskExecutionTime" tells the management system the time each job should not exceed and if so, will cancel that job. Finally, the "LimitEachTaskMemoryUsage" tells the management system how much memory each core should have access to. Below these options are all of the job command lines that will be delegated and launched to each available core.

each node that was being used. Without this, due to how busy HPC Wales can be in having over 500 users, if each job was to be submitted separately, as soon as the single sequence was complete that node would then be given to the next user in the queue for their job. This method would mean that modelling the 20,160 thousand sequences would take a considerable length of time. With the assumption that 100 sequences are completed a day without any issues or queues with other users, modelling the 20,160 sequences would take ten months to complete. With the job management system developed by Karl Austin-Muttitt, each node that was allocated was “locked off”, and the management system would delegate the jobs within the CSV file to them. Using this management system meant that the node could be used the entire time until it reached its time limit and multiple sequences could be modelled one after each other on each node. When applied across several nodes, this sped up the modelling process dramatically. Instead of taking ten months to complete the modelling, this took less than half of that time. When the modelling had completed, a script named and copied all the 3D model files into a folder.

Not all the sequences produced a 3D model; this is due to low homology and no template coverage. Of the 20,160 sequences, 9,321 produced plausible models. The 11,000 sequences that did not produce a model underwent modelling using the threading method by using I-TASSER. The 9,321 sequences that did produce a model were run through PROCHECK (Laskowski et al., 1993), which checks that the models are of good quality and analyses the residue-by-residue geometry. Only models that have a score of 90%+ were used for docking against the 20 most widely used drugs. The 9,321 homology models set contained only 49 of the 67 known targets for the top 20 drugs which are listed in DrugBank (Wishart et al., 2017). Therefore the remaining 18 would have to be modelled via protein threading. For the 49 known protein targets, there are 54 known interactions with the top 20 most widely prescribed drugs as there are specific proteins that interact with more than one drug. The files for the top 20 most widely prescribed drugs which were used for molecular docking were downloaded in SDF files format from the DrugBank database.

4.3 Molecular Docking Baseline Results

As stated previously, these results were raw data produced by AutoDock Vina and DOCK 6.0 which were used to assess how well the pipeline works and how accurate the algorithms were by seeing if they could accurately identify the known targets of each ligand being docked. The 9,321 3D models that predicted plausible models through the homology modelling pipeline were used for the molecular docking process. The homology modelling pipeline outputs the protein 3D models in a PDB format which neither AutoDock Vina nor DOCK 6.0 use to predict dockings. The PDB files were converted to the respective formats for AutoDock Vina (PDBQT) and DOCK 6.0 (MOL2). The program OBABEL (O'Boyle et al., 2011) was used to convert the PDB files. With the 3D protein models converted to the correct file formats, they are then docked against the top 20 most widely prescribed drugs.

4.3.1 Docking results using the original docking pipeline

There are two sets of results that are investigated, those of AutoDock Vina and those of DOCK 6.0.

4.3.1.1 AutoDock Vina Results

AutoDock Vina only outputs a single scoring function called the binding affinity. The method that will be used to determine if AutoDock Vina can identify the known targets of the top 20 most widely prescribed drugs, a mean and standard deviation outlier method (2SD) will be applied to the binding affinity scores. The mean and standard deviation outlier method is applied to each ligand column and consists of the following calculations. For each ligand column, the total mean is calculated. The variance is then calculated for each ligand's column. Finally, the standard deviation is calculated for the entire scores associated with each ligand. With the mean, variance and standard deviation calculated for each ligand, the standard deviations were then calculated for all binding affinity scores in the docking set. Shown in table 4.2 is an example of an excel spreadsheet that shows the results of the calculations. The standard deviations are calculated for each column. For example, to calculate the standard deviation for all the predicted scores in the ligand column "Tadalafil", each docking pairs binding affinity score will firstly have the total mean of the column subtracted.

Table 4.2: example of calculations made to the docking scores – shown is the output generated after the 2 standard deviation rule (2SD) was applied to the Autodock Vina affinity results of each docking pair. The example shows a sample of results for the Ligands Tadalafil and Tazobactam and 2SD results against 8 proteins. The 2SD rule is applied column-based, and if an affinity is either -2 standard deviations away or more, it is classified as an interaction (green highlight). If an affinity score is between -2 and -1 standard deviation away, it is then classified as a potential interaction (yellow highlight).

	Tadalafil	**Tazobactam**
Mean	**-48.787**	**-44.939**
Variance	**53.621**	**52.045**
Standard Deviation	**7.323**	**7.214**
Dehydrogenase/reductase SDR family member 12	0.094	-0.997
Sodium/glucose cotransporter 5	0.079	0.066
Calcium and integrin-binding family member 4	-0.048	0.167
Anoctamin-9	-0.095	-0.347
Acetolactate synthase-like protein	-0.414	-0.647
Synaptonemal complex central element protein 3	1.468	1.682
Peptidyl-prolyl cis-trans isomerase A-like 4G	0.08	-0.322
Threonine--tRNA ligase 2, cytoplasmic	-2.857	-1.399

When the mean has been subtracted from the binding affinity, the resultant number is then divided d by the standard deviation of the entire ligand column. This calculation will give each docking pair binding affinity a result of how many standard deviations the binding affinity is from the standard deviation of the ligand column. The following rules were applied to the calculated standard deviation scores; If the score is -2 or less away from the standard deviation, the docking pair is classed as an interaction. The second rule being if it is between -1 and -2, it is classed as a potential interaction. As shown in Table 4.2, the dockings that are possible interactions are highlighted in yellow and the dockings classed as interactions are highlighted in green.

After the mean and standard deviation outlier method had been applied to the entire docking homology modelling set, the known protein targets were then extracted. Table 4.3 shows the predicted binding affinities standard deviation results for AutoDock Vina. A script was developed to analyse the entire file of docking results that would calculate the mean, variance, and standard deviation for each ligand automatically and then generates two extra columns called “Affinity is a hit” and “Affinity is a potential interaction” which will either have the words “TRUE” or “FALSE”. The docking binding affinity would be classified as a hit if there was a “TRUE” within the “Affinity is a hit” column. The docking binding affinity would be classified as a potential interaction if a “TRUE” was within the “Affinity is a potential interaction” column. If both columns have a “FALSE” within them, the docking pair would be classified as a miss. The baseline results for AutoDock Vina shown in table 4.3 were not very promising. Of the 54 known protein target interactions within the homology modelling set, only two were classified as being a hit. 12 of the 54 known protein target interactions within the homology modelling set were classified as a potential interaction. This result is of very low accuracy as only 3.7% of the known protein target interactions were being classified correctly as a hit.

Table 4.3: AutoDock Vina Results from preliminary docking –the table shows the results of the known targets for the top 20 drugs and indicates if the docking pair is classified as and interaction or potential interaction. Of the 54 known interactions, two were classified as interactions and 18 were classified as potential interactions.

Ligand	Protein	Affinity SD	Affinity is a hit	Affinity is a Potential Interaction
LENALIDOMIDE	Q96SW2_HUMAN_CRBN	0.0338076	FALSE	FALSE
LENALIDOMIDE	P33151_HUMAN_CADH5	1.1242816	FALSE	FALSE
LENALIDOMIDE	P35354_HUMAN_PGH2	-0.560996	FALSE	FALSE
LENALIDOMIDE	O14788_HUMAN_TNF11	-0.16446	FALSE	FALSE
FLUTICASONE PROPIONATE	P04150_HUMAN_GCR	-1.021302	FALSE	TRUE
FLUTICASONE PROPIONATE	P06401_HUMAN_PRGR	-0.067525	FALSE	FALSE
FLUTICASONE PROPIONATE	P47712_HUMAN_PA24A	-1.108009	FALSE	TRUE
IMATINIB	P00519_HUMAN_ABL1	-0.859543	FALSE	FALSE
IMATINIB	P09619_HUMAN_PGFRB	-0.94922	FALSE	FALSE
IMATINIB	P16234_HUMAN_PGFRA	-0.94922	FALSE	FALSE
TADALAFIL	O76074_HUMAN_PDE5A	-1.228432	FALSE	TRUE
EZETIMIBE	Q9UHC9_HUMAN_NPCL1	-0.813683	FALSE	FALSE
EZETIMIBE	P15144_HUMAN_AMPN	-0.900076	FALSE	FALSE
FORMOTEROL	P07550_HUMAN_ADRB2	-1.441168	FALSE	TRUE
GABAPENTIN	Q05586_HUMAN_NMDZ1	-1.72506	FALSE	TRUE
GABAPENTIN	Q12879_HUMAN_NMDE1	0.6783725	FALSE	FALSE
GABAPENTIN	Q13224_HUMAN_NMDE2	-0.713088	FALSE	FALSE
GABAPENTIN	Q14957_HUMAN_NMDE3	-1.472067	FALSE	TRUE
GABAPENTIN	O60391_HUMAN_NMD3B	-0.586592	FALSE	FALSE
GABAPENTIN	P30542_HUMAN_AA1R	-0.333599	FALSE	FALSE
GABAPENTIN	O15399_HUMAN_NMDE4	-1.345571	FALSE	TRUE
LEVETIRACETAM	Q7L0J3_HUMAN_SV2A	-2.608714	TRUE	FALSE
BUDESONIDE	P04150_HUMAN_GCR	-0.831187	FALSE	FALSE
ARIPIPRAZOLE	P08172_HUMAN_ACM2	-0.903412	FALSE	FALSE
ARIPIPRAZOLE	P08173_HUMAN_ACM4	-0.223935	FALSE	FALSE
ARIPIPRAZOLE	P08908_HUMAN_5HT1A	-0.733543	FALSE	FALSE
ARIPIPRAZOLE	P08912_HUMAN_ACM5	0.6254114	FALSE	FALSE
ARIPIPRAZOLE	P11229_HUMAN_ACM1	-1.24315	FALSE	TRUE
ARIPIPRAZOLE	P14416_HUMAN_DRD2	-1.837693	FALSE	TRUE
ARIPIPRAZOLE	P18089_HUMAN_ADA2B	-1.41302	FALSE	TRUE
ARIPIPRAZOLE	P18825_HUMAN_ADA2C	-0.393804	FALSE	FALSE
ARIPIPRAZOLE	P20309_HUMAN_ACM3	-0.903412	FALSE	FALSE
ARIPIPRAZOLE	P21728_HUMAN_DRD1	-1.158216	FALSE	TRUE
ARIPIPRAZOLE	P21917_HUMAN_DRD4	-1.41302	FALSE	TRUE
ARIPIPRAZOLE	P21918_HUMAN_DRD5	-0.054066	FALSE	FALSE
ARIPIPRAZOLE	P28221_HUMAN_5HT1D	-1.41302	FALSE	TRUE
ARIPIPRAZOLE	P28222_HUMAN_5HT1B	-0.988347	FALSE	FALSE
ARIPIPRAZOLE	P28223_HUMAN_5HT2A	-0.733543	FALSE	FALSE
ARIPIPRAZOLE	P28335_HUMAN_5HT2C	-0.903412	FALSE	FALSE
ARIPIPRAZOLE	P28566_HUMAN_5HT1E	-0.478739	FALSE	FALSE
ARIPIPRAZOLE	P34969_HUMAN_5HT7R	0.2856728	FALSE	FALSE
ARIPIPRAZOLE	P35348_HUMAN_ADA1A	-1.073281	FALSE	TRUE
ARIPIPRAZOLE	P35367_HUMAN_HRH1	0.2007382	FALSE	FALSE
ARIPIPRAZOLE	P35368_HUMAN_ADA1B	-0.563673	FALSE	FALSE
ARIPIPRAZOLE	P50406_HUMAN_5HT6R	-1.497954	FALSE	TRUE
SITAGLIPTIN	P27487_HUMAN_DPP4	-1.079737	FALSE	TRUE
TIOTROPIUM	P08172_HUMAN_ACM2	-0.520533	FALSE	FALSE
TIOTROPIUM	P11229_HUMAN_ACM1	-1.57036	FALSE	TRUE
TIOTROPIUM	P20309_HUMAN_ACM3	-0.940464	FALSE	FALSE
SOLIFENACIN	P08172_HUMAN_ACM2	-1.228106	FALSE	TRUE
SOLIFENACIN	P08173_HUMAN_ACM4	-0.851307	FALSE	FALSE
SOLIFENACIN	P08912_HUMAN_ACM5	-0.323788	FALSE	FALSE
SOLIFENACIN	P11229_HUMAN_ACM1	-1.755625	FALSE	TRUE
SOLIFENACIN	P20309_HUMAN_ACM3	-2.057064	TRUE	FALSE

4.3.1.2 DOCK 6.0 results

Table 4.4: Figure 15: DOCK 6.0 Results from preliminary docking - the table shows the results of the known targets for the top 20 drugs and indicates if the docking pair is classified as and interaction or potential interaction. Of the 54 known interactions, two were classified as interactions and 18 were classified as potential interactions.

Ligand	Protein	Affinity SD	Affinity is a hit	Affinity is a potential Interaction
LENALIDOMIDE	Q96SW2_HUMAN_CRBN	-0.381899	FALSE	FALSE
LENALIDOMIDE	P33151_HUMAN_CADH5	1.058491	FALSE	FALSE
LENALIDOMIDE	P35354_HUMAN_PGH2	-0.805543	FALSE	FALSE
LENALIDOMIDE	O14788_HUMAN_TNF11	0.182153	FALSE	FALSE
FLUTICASONE PROPIONATE	P04150_HUMAN_GCR	-1.221498	FALSE	TRUE
FLUTICASONE PROPIONATE	P06401_HUMAN_PRGR	0.6006648	FALSE	FALSE
FLUTICASONE PROPIONATE	P47712_HUMAN_PA24A	-0.358854	FALSE	FALSE
IMATINIB	P00519_HUMAN_ABL1	-0.990565	FALSE	FALSE
IMATINIB	P09619_HUMAN_PGFRB	-0.75223	FALSE	FALSE
IMATINIB	P16234_HUMAN_PGFRA	-0.016082	FALSE	FALSE
TADALAFIL	Q9HCR9_HUMAN_PDE11	-0.548684	FALSE	FALSE
TADALAFIL	O76074_HUMAN_PDE5A	-1.823438	FALSE	TRUE
EZETIMIBE	Q9UHC9_HUMAN_NPCL1	-0.762048	FALSE	FALSE
EZETIMIBE	P15144_HUMAN_AMPN	-1.187156	FALSE	TRUE
FORMOTEROL	P07550_HUMAN_ADRB2	-0.686749	FALSE	FALSE
GABAPENTIN	Q05586_HUMAN_NMDZ1	-0.862706	FALSE	FALSE
GABAPENTIN	Q12879_HUMAN_NMDE1	-1.228385	FALSE	TRUE
GABAPENTIN	Q13224_HUMAN_NMDE2	0.243769	FALSE	FALSE
GABAPENTIN	Q14957_HUMAN_NMDE3	0.0833419	FALSE	FALSE
GABAPENTIN	O60391_HUMAN_NMD3B	-0.905172	FALSE	FALSE
GABAPENTIN	P30542_HUMAN_AA1R	1.3148556	FALSE	FALSE
GABAPENTIN	O15399_HUMAN_NMDE4	-1.398249	FALSE	TRUE
DB01076	P04035_HUMAN_HMDH	0.1516774	FALSE	FALSE
LEVETIRACETAM	Q7L0J3_HUMAN_SV2A	-1.70653	FALSE	TRUE
BUDESONIDE	P04150_HUMAN_GCR	-1.065434	FALSE	TRUE
ARIPIPRAZOLE	P08172_HUMAN_ACM2	-0.581759	FALSE	FALSE
ARIPIPRAZOLE	P08173_HUMAN_ACM4	0.0747974	FALSE	FALSE
ARIPIPRAZOLE	P08908_HUMAN_5HT1A	-2.080225	TRUE	FALSE
ARIPIPRAZOLE	P08912_HUMAN_ACM5	0.4566719	FALSE	FALSE
ARIPIPRAZOLE	P11229_HUMAN_ACM1	-0.684485	FALSE	FALSE
ARIPIPRAZOLE	P14416_HUMAN_DRD2	-1.789911	FALSE	TRUE
ARIPIPRAZOLE	P18089_HUMAN_ADA2B	0.0323669	FALSE	FALSE
ARIPIPRAZOLE	P18825_HUMAN_ADA2C	-0.680019	FALSE	FALSE
ARIPIPRAZOLE	P20309_HUMAN_ACM3	-2.292377	TRUE	FALSE
ARIPIPRAZOLE	P21728_HUMAN_DRD1	-1.591158	FALSE	TRUE
ARIPIPRAZOLE	P21917_HUMAN_DRD4	-1.611256	FALSE	TRUE
ARIPIPRAZOLE	P21918_HUMAN_DRD5	0.0524656	FALSE	FALSE
ARIPIPRAZOLE	P28221_HUMAN_5HT1D	-1.856907	FALSE	TRUE
ARIPIPRAZOLE	P28222_HUMAN_5HT1B	-0.782745	FALSE	FALSE
ARIPIPRAZOLE	P28223_HUMAN_5HT2A	-0.776046	FALSE	FALSE
ARIPIPRAZOLE	P28335_HUMAN_5HT2C	0.2556853	FALSE	FALSE
ARIPIPRAZOLE	P28566_HUMAN_5HT1E	0.4790037	FALSE	FALSE
ARIPIPRAZOLE	P34969_HUMAN_5HT7R	-0.088225	FALSE	FALSE
ARIPIPRAZOLE	P35348_HUMAN_ADA1A	-0.293678	FALSE	FALSE
ARIPIPRAZOLE	P35367_HUMAN_HRH1	-0.432135	FALSE	FALSE
ARIPIPRAZOLE	P35368_HUMAN_ADA1B	0.409775	FALSE	FALSE
ARIPIPRAZOLE	P50406_HUMAN_5HT6R	-0.930135	FALSE	FALSE
SITAGLIPTIN	P27487_HUMAN_DPP4	-0.968113	FALSE	FALSE
TIOTROPIUM	P08172_HUMAN_ACM2	0.5587708	FALSE	FALSE
TIOTROPIUM	P11229_HUMAN_ACM1	-1.047007	FALSE	TRUE
TIOTROPIUM	P20309_HUMAN_ACM3	0.1487851	FALSE	FALSE

The same mean and standard deviation outlier method was applied to the predicted DOCK 6.0 results. The analysing script used on the AutoDock Vina results was also used for the DOCK 6.0 results to apply the mean and standard deviation outlier method. Shown in table 4.4 are the predicted binding affinities standard deviation results for DOCK 6.0. Like the AutoDock Vina results, DOCK 6.0 has not predicted promising results as it as well has only predicted 2 of the 54 known protein interactions in the homology modelling set as hits. It has also categorised 11 of the 54 known protein target interactions as potential interactions. Questions were subsequently raised regarding the poor predictive performance, which led to a review of the modelling and molecular docking approaches.

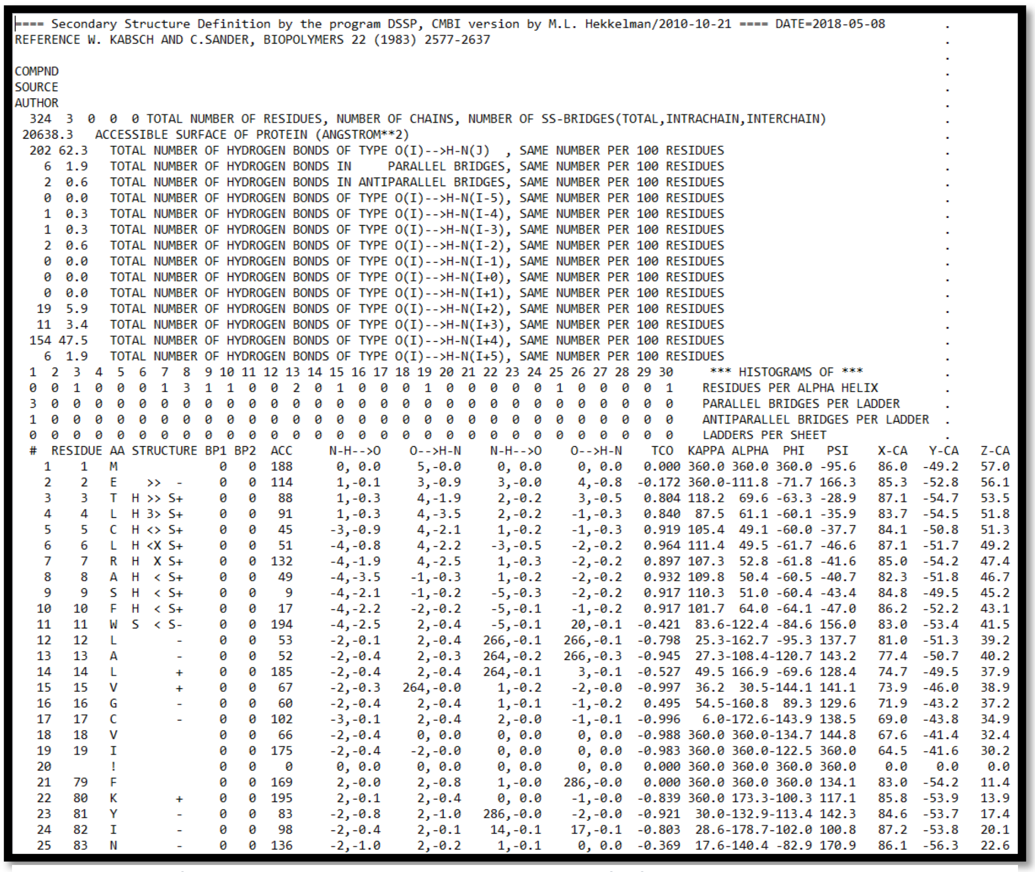
4.4 An investigation into the poor predictive performance of Molecular docking

The review began by investigating the quality of the generated 3D models by the homology modelling. To analyse the created homology modelled 3D protein, the models created for the 49 known protein targets within the homology set were analysed.

4.4.1 Analysis of the 49 Known protein targets homology models

When investigating the homology 3D models for the 49 known protein targets, it was clear to see that the models were not of the highest quality. The quality of modelling was due to inadequate homologue coverage to model the protein structures. Of the 49 known protein targets, ten had poor quality models predicted. The inadequate homologue coverage of the 3D protein models could be the cause for such a poor docking performance by AutoDock Vina and DOCK 6.0.

This analysis then raised another question, “could more models within the homology modelling set be of poor quality, causing the poor docking performance?”. It would be an unrealistic task to analyse each protein model’s quality by hand, which led to the development of a python script that automated the process. The python script is dependent on a program called DSSP (Touw et al., 2014) to determine the homologue coverage of a 3D protein model. DSSP is used to assign secondary structure to 3D protein models by using a vast database of structure assignment for all proteins stored in the Protein Data Bank (Burley et al., 2018). Within the development model analysis script, DSSP was run against each 3D protein model file of the homology set which created a dssp file for each. Shown in figure 4.3 is an example of a dssp file that was created for each 3D protein model file. With the created dssp file, it is then possible to calculate the percentage of coverage of each homology model. The developed model analysis script uses a column within the dssp file called “structure” to calculate the percentage. This column contains information detailing which amino acids have an associated secondary structure. If the column contains an “S+”, it signifies that the corresponding amino acid has a known structure. By using the length of the FASTA sequence and the



```
==== Secondary Structure Definition by the program DSSP, CMBI version by M.L. Hekkelman/2010-10-21 ==== DATE=2018-05-08     .
REFERENCE W. KABSCH AND C.SANDER, BIOPOLYMERS 22 (1983) 2577-2637                                                           .
                                                                                                                            .
COMPND                                                                                                                      .
SOURCE                                                                                                                      .
AUTHOR                                                                                                                      .
  324  3  0  0  0 TOTAL NUMBER OF RESIDUES, NUMBER OF CHAINS, NUMBER OF SS-BRIDGES(TOTAL,INTRACHAIN,INTERCHAIN)             .
 20638.3   ACCESSIBLE SURFACE OF PROTEIN (ANGSTROM**2)                                                                      .
  202 62.3   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(J)  , SAME NUMBER PER 100 RESIDUES                           .
    6  1.9   TOTAL NUMBER OF HYDROGEN BONDS IN     PARALLEL BRIDGES, SAME NUMBER PER 100 RESIDUES                           .
    2  0.6   TOTAL NUMBER OF HYDROGEN BONDS IN ANTIPARALLEL BRIDGES, SAME NUMBER PER 100 RESIDUES                           .
    0  0.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I-5), SAME NUMBER PER 100 RESIDUES                           .
    1  0.3   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I-4), SAME NUMBER PER 100 RESIDUES                           .
    1  0.3   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I-3), SAME NUMBER PER 100 RESIDUES                           .
    2  0.6   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I-2), SAME NUMBER PER 100 RESIDUES                           .
    0  0.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I-1), SAME NUMBER PER 100 RESIDUES                           .
    0  0.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+0), SAME NUMBER PER 100 RESIDUES                           .
    0  0.0   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+1), SAME NUMBER PER 100 RESIDUES                           .
   19  5.9   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+2), SAME NUMBER PER 100 RESIDUES                           .
   11  3.4   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+3), SAME NUMBER PER 100 RESIDUES                           .
  154 47.5   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+4), SAME NUMBER PER 100 RESIDUES                           .
    6  1.9   TOTAL NUMBER OF HYDROGEN BONDS OF TYPE O(I)-->H-N(I+5), SAME NUMBER PER 100 RESIDUES                           .
  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30     *** HISTOGRAMS OF ***          .
  0  0  1  0  0  0  1  3  1  1  0  0  2  0  1  0  0  0  1  0  0  0  0  0  1  0  0  0  0  1    RESIDUES PER ALPHA HELIX         .
  3  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0    PARALLEL BRIDGES PER LADDER      .
  1  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0    ANTIPARALLEL BRIDGES PER LADDER  .
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0    LADDERS PER SHEET                .
  #  RESIDUE AA STRUCTURE BP1 BP2  ACC     N-H-->O    O-->H-N    N-H-->O    O-->H-N    TCO  KAPPA ALPHA  PHI   PSI    X-CA   Y-CA   Z-CA
    1    1   M              0   0  188      0, 0.0     5,-0.0     0, 0.0     0, 0.0   0.000 360.0 360.0 360.0 -95.6   86.0  -49.2   57.0
    2    2   E    >>  -     0   0  114      1,-0.1     3,-0.9     3,-0.0     4,-0.8  -0.172 360.0-111.8 -71.7 166.3   85.3  -52.8   56.1
    3    3   T  H >> S+     0   0   88      1,-0.3     4,-1.9     2,-0.2     3,-0.5   0.804 118.2  69.6 -63.3 -28.9   87.1  -54.7   53.5
    4    4   L  H 3> S+     0   0   91      1,-0.3     4,-3.5     2,-0.2    -1,-0.3   0.840  87.5  61.1 -60.1 -35.9   83.7  -54.5   51.8
    5    5   C  H <> S+     0   0   45     -3,-0.9     4,-2.1     1,-0.2    -1,-0.3   0.919 105.4  49.1 -60.0 -37.7   84.1  -50.8   51.3
    6    6   L  H <X S+     0   0   51     -4,-0.8     4,-2.2    -3,-0.5    -2,-0.2   0.964 111.4  49.5 -61.7 -46.6   87.1  -51.7   49.2
    7    7   R  H  X S+     0   0  132     -4,-1.9     4,-2.5     1,-0.3    -2,-0.2   0.897 107.3  52.8 -61.8 -41.6   85.0  -54.2   47.4
    8    8   A  H  < S+     0   0   49     -4,-3.5    -1,-0.3     1,-0.2    -2,-0.2   0.932 109.8  50.4 -60.5 -40.7   82.3  -51.8   46.7
    9    9   S  H  < S+     0   0    9     -4,-2.1    -1,-0.2    -5,-0.3    -2,-0.2   0.917 110.3  51.0 -60.4 -43.4   84.8  -49.5   45.2
   10   10   F  H  < S+     0   0   17     -4,-2.2    -2,-0.2    -5,-0.1    -1,-0.2   0.917 101.7  64.0 -64.1 -47.0   86.2  -52.2   43.1
   11   11   W  S  < S-     0   0  194     -4,-2.5     2,-0.4    -5,-0.1    20,-0.1  -0.421  83.6-122.4 -84.6 156.0   83.0  -53.4   41.5
   12   12   L        -     0   0   53     -2,-0.1     2,-0.4   266,-0.1   266,-0.1  -0.798  25.3-162.7 -95.3 137.7   81.0  -51.3   39.2
   13   13   A        -     0   0   52     -2,-0.4     2,-0.3   264,-0.2   266,-0.3  -0.945  27.3-108.4-120.7 143.2   77.4  -50.7   40.2
   14   14   L        +     0   0  185     -2,-0.4     2,-0.4   264,-0.1     3,-0.1  -0.527  49.5 166.9 -69.6 128.4   74.7  -49.5   37.9
   15   15   V        +     0   0   67     -2,-0.3   264,-0.0     1,-0.2    -2,-0.0  -0.997  36.2  30.5-144.1 141.1   73.9  -46.0   38.9
   16   16   G        -     0   0   60     -2,-0.4     2,-0.4     1,-0.1    -1,-0.2   0.495  54.5-160.8  89.3 129.6   71.9  -43.2   37.2
   17   17   C        -     0   0  102     -3,-0.1     2,-0.4     2,-0.0    -1,-0.1  -0.996   6.0-172.6-143.9 138.5   69.0  -43.8   34.9
   18   18   V              0   0   66     -2,-0.4     0, 0.0     0, 0.0     0, 0.0  -0.988 360.0 360.0-134.7 144.8   67.6  -41.4   32.4
   19   19   I              0   0  175     -2,-0.4    -2,-0.0     0, 0.0     0, 0.0  -0.983 360.0 360.0-122.5 360.0   64.5  -41.6   30.2
   20        !              0   0    0      0, 0.0     0, 0.0     0, 0.0     0, 0.0   0.000 360.0 360.0 360.0 360.0    0.0    0.0    0.0
   21   79   F              0   0  169      2,-0.0     2,-0.8     1,-0.0   286,-0.0   0.000 360.0 360.0 360.0 134.1   83.0  -54.2   11.4
   22   80   K        +     0   0  195      2,-0.1     2,-0.4     0, 0.0    -1,-0.0  -0.839 360.0 173.3-100.3 117.1   85.8  -53.9   13.9
   23   81   Y        -     0   0   83     -2,-0.8     2,-1.0   286,-0.0    -2,-0.0  -0.921  30.0-132.9-113.4 142.3   84.6  -53.7   17.4
   24   82   I        -     0   0   98     -2,-0.4     2,-0.1    14,-0.1    17,-0.1  -0.803  28.6-178.7-102.0 100.8   87.2  -53.8   20.1
   25   83   N        -     0   0  136     -2,-1.0     2,-0.2     1,-0.1     0, 0.0  -0.369  17.6-140.4 -82.9 170.9   86.1  -56.3   22.6
```

Figure 4.3: DSSP file example – the example shows the type of information that the DSSP program adds to a protein PDB file. DSSP adds additional information such as hydrogen bonds.

amount of "S+" symbols contained in the "structure" column of the dssp file, it is possible to calculate the percentage of structure coverage.

4.4.2 Model analysis for the homology modelling set

Shown in figure 4.4 is the result of the structure quality of the 9,321 homology models after applying the developed model analysis script. Figure 4.4 shows that only 3,420 of the 9,321 homology models had a structured coverage of 80% and above. Out of the 49 known protein targets within the set, only 19 were in this bracket. The results in figure 4.4 have determined that 5,901 3D protein models have less than 80% structure coverage.

This outcome could be the cause of why the molecular docking programs have poorly performed as there is potential for a drug to be docked in an incorrect location, which can cause false-positive and false-negative results. To discover if the 3D protein models caused the poor docking performance only the scores for the 3,420 models with 80% and above structure coverage were further analysed.

4.4.2.1 Analysis of the 80%+ structures coverage homology models

The predicted binding scores for the 3,420 high-quality homology models were extracted from the excel file created initially. The same mean and standard deviation outlier method was applied to the results of AutoDock Vina and DOCK 6.0, as discussed in section 4.3.1.1.

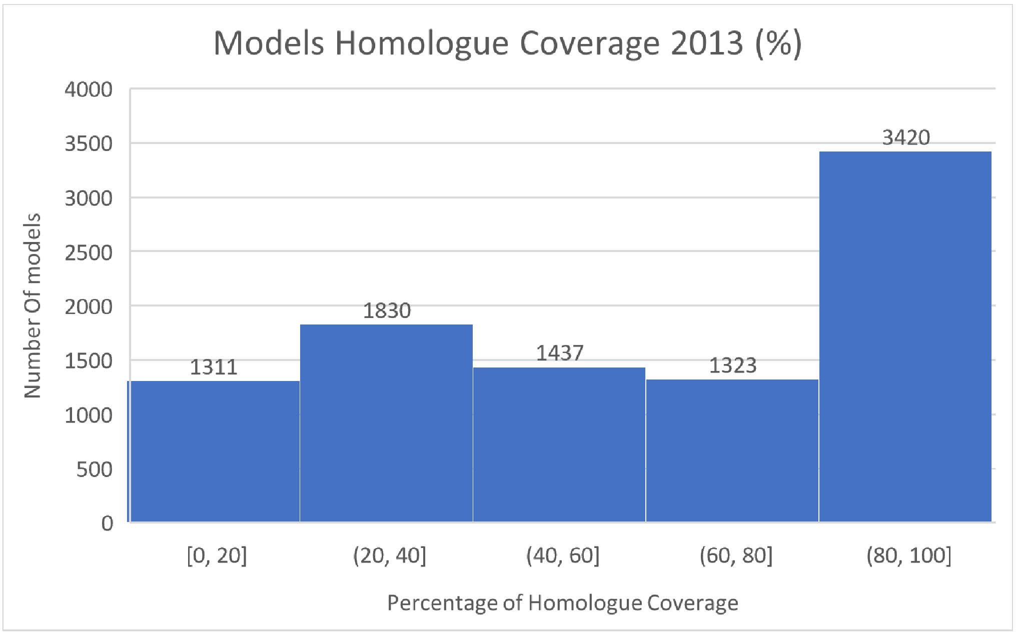


Figure 4.4:Structual coverage of the 9,321 homology models – the graph shows the different percentage groups of structural coverage and how many of the 9,321 homology models were classified in each group.

AutoDock Vina results

Shown in table 4.5 are the extracted AutoDock Vina results for the high-quality homology models. Only 17 of the 49 known targets were included in the high-quality homology model set. The results surprisingly deteriorated further, not predicting any of the targets in the set as a hit and identifying only 6 of them were classified as potential interaction. Neither of the two known interactions that were previously predicted correctly was within the extracted set.

DOCK 6.0 results

Shown in table 4.6 are the extracted DOCK 6.0 results for the high-quality homology models. The shown results in table 5 are not qualitatively any different for predicting the known interactions. Like the AutoDock Vina results, none of the known interactions was strongly predicted as a hit, though

Table 4.5: AutoDock Vina results for the 3,420 high-quality models – the table shows the results for the known target interactions that were within the high structural coverage. the results show that none of the known targets were classified as interaction and 6 of them were classified as potential interactions.

Ligand	Protein	Affinity SD	Affinity is a hit	Affinity is a potential interaction
LENALIDOMIDE	Q96SW2_HUMAN_CRBN	0.299	FALSE	FALSE
LENALIDOMIDE	P35354_HUMAN_PGH2	-0.375	FALSE	FALSE
FLUTICASONE PROPIONATE	P47712_HUMAN_PA24A	-0.937	FALSE	FALSE
EZETIMIBE	P15144_HUMAN_AMPN	-0.694	FALSE	FALSE
FORMOTEROL	P07550_HUMAN_ADRB2	-1.327	FALSE	TRUE
ARIPIPRAZOLE	P14416_HUMAN_DRD2	-1.711	FALSE	TRUE
ARIPIPRAZOLE	P28221_HUMAN_5HT1D	-1.253	FALSE	TRUE
ARIPIPRAZOLE	P18825_HUMAN_ADA2C	-0.156	FALSE	FALSE
ARIPIPRAZOLE	P35367_HUMAN_HRH1	0.483	FALSE	FALSE
ARIPIPRAZOLE	P18089_HUMAN_ADA2B	-1.253	FALSE	TRUE
ARIPIPRAZOLE	P28222_HUMAN_5HT1B	-0.796	FALSE	FALSE
ARIPIPRAZOLE	P08908_HUMAN_5HT1A	-0.522	FALSE	FALSE
ARIPIPRAZOLE	P28335_HUMAN_5HT2C	-0.705	FALSE	FALSE
ARIPIPRAZOLE	P21917_HUMAN_DRD4	-1.253	FALSE	TRUE
ARIPIPRAZOLE	P28566_HUMAN_5HT1E	-0.248	FALSE	FALSE
ARIPIPRAZOLE	P08172_HUMAN_ACM2	-0.705	FALSE	FALSE
SITAGLIPTIN	P27487_HUMAN_DPP4	-0.892	FALSE	FALSE
TIOTROPIUM	P08172_HUMAN_ACM2	-0.306	FALSE	FALSE
SOLIFENACIN	P08172_HUMAN_ACM2	-1.012	FALSE	TRUE

Table 4.6: DOCK 6.0 results for the 3,420 high-quality models - the table shows the results for the known target interactions that were within the high structural coverage. the results show that none of the known targets were classified as interaction and 7 of them were classified as potential interactions.

Ligand	Protein	Affinity SD	Affinity is a hit	Affinity is a potential interaction
LENALIDOMIDE	Q96SW2_HUMAN_CRBN	-1.000	FALSE	TRUE
LENALIDOMIDE	P35354_HUMAN_PGH2	1.035	FALSE	FALSE
FLUTICASONE PROPIONATE	P47712_HUMAN_PA24A	1.01	FALSE	FALSE
EZETIMIBE	P15144_HUMAN_AMPN	1.035	FALSE	FALSE
FORMOTEROL	P07550_HUMAN_ADRB2	1.036	FALSE	FALSE
ARIPIPRAZOLE	P14416_HUMAN_DRD2	-1.329	FALSE	TRUE
ARIPIPRAZOLE	P28221_HUMAN_5HT1D	-1.345	FALSE	TRUE
ARIPIPRAZOLE	P18825_HUMAN_ADA2C	-1.057	FALSE	TRUE
ARIPIPRAZOLE	P35367_HUMAN_HRH1	-0.996	FALSE	FALSE
ARIPIPRAZOLE	P18089_HUMAN_ADA2B	-0.883	FALSE	FALSE
ARIPIPRAZOLE	P28222_HUMAN_5HT1B	-1.082	FALSE	TRUE
ARIPIPRAZOLE	P08908_HUMAN_5HT1A	-1.4	FALSE	TRUE
ARIPIPRAZOLE	P28335_HUMAN_5HT2C	-0.828	FALSE	FALSE
ARIPIPRAZOLE	P21917_HUMAN_DRD4	1.035	FALSE	FALSE
ARIPIPRAZOLE	P28566_HUMAN_5HT1E	1.035	FALSE	FALSE
ARIPIPRAZOLE	P08172_HUMAN_ACM2	1.035	FALSE	FALSE
SITAGLIPTIN	P27487_HUMAN_DPP4	-1.128	FALSE	TRUE
TIOTROPIUM	P08172_HUMAN_ACM2	1.027	FALSE	FALSE

seven were identified as potential interactions, still a very disappointing performance regarding predicting known interactions.

Conclusion of the High-quality homology model set results

The results that have been analysed for the high-quality homology model set have determined that the binding affinity scoring function together with the mean and standard deviation outlier method could not identify the known target interactions of the vast amount of data. After firstly investigating the quality of the protein models, it was discovered that only 36.7% had structure coverage of 80% or above. This finding indicated that the models could be the cause for the poor docking performance. This presumption was then disproved after analysing the results data for the 3,420 high-quality proteins as it failed to rank any of the known interactions as hits. The result leads to the conclusion that attempting to predict a known interaction using only the binding affinity scoring function does not work. After this conclusion, to try and develop a method to determine known interaction with a higher level of accuracy, new scoring functions will be obtained for all docking results.

4.5 Development of higher-accuracy molecular docking

The baseline results were gained early in the project (2013), so it was necessary to re-model the sequences in 2017 following updating the pipeline databases such as the UniProt and PDB database (September 2017), which is continually updated with new template structures.

4.5.1 New molecular docking pipeline

Following examination of the baseline results, it was decided to revise aspects of the molecular docking approaches. A new molecular docking pipeline will be used to dock the newly homology modelled proteins against the top 20 most valuable sold drugs. The pipeline called Shipyard 1.5, developed by Karl Austin-Muttitt was used for the molecular docking. Shipyard 1.5 is discussed further in section 2.3.4. A new addition to the pipeline is the ability to identify possible protein docking sites. In the baseline homology docking work; there was no defined search space applied in use of either AutoDock Vina or DOCK 6.0 algorithms. The default search space given is effectively a 3D grid that covers the entire protein model surface, but this could lead to ligands being docked in incorrect positions, which would not be biologically plausible in real life. For many functionally essential proteins, the active site has been characterised, and this information is available on the UniProt database (UniProt Consortium, 2018). Not all active sites are fully determined, though, so for the next round of dockings, a program called DogSite (Volkamer et al., 2012) is employed to determine possible active sites. DogSite is a program that can calculate/predict the location of where a protein models' active site is located. Using this program, it is possible to reduce in a fully automated way the search space considered by the docking algorithms, which should improve accuracy and also speed up each docking process. Shown in figure 4.5 is an example of the search space of a protein without applying DogSite and shown in figure 4.6 is an example of the search space adopted after applying DogSite. As shown, it dramatically lowers the search space used to attempt to calculate a docking. This method can potentially increase the accuracy of the dockings as only the large active sites are being considered for the positioning of the ligand and better-quality dockings will result from the focusing of search space.

Another addition to the molecular docking pipeline is new scoring functions. These added scores that were produced by the Shipyard docking pipeline are GBSA (Generalised Born with solvent-accessible Surface Area) (Genheden and Ryde, 2015) and Drug Score X (DSX). The GBSA scoring function is a more in-depth physics-based function that attempts to account for solvation effects. Drug Score X(DSX) is a machine learning knowledge-based scoring function which attempts to model

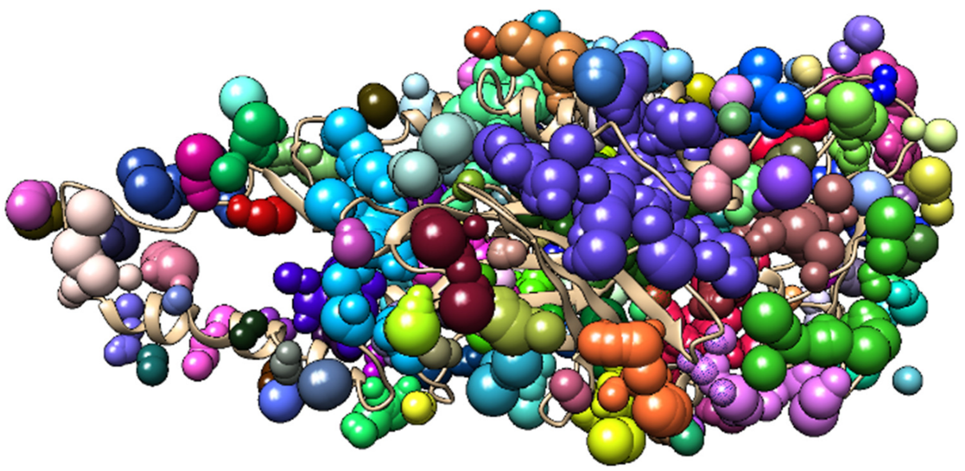


Figure 4.5: Image of docking search space without DogSite applied - shown is an example of an all the potential sites that Autodock Vina and DOCK 6.0 would attempt to dock a ligand too. Each colour of bundle spheres represents a different potential active site.

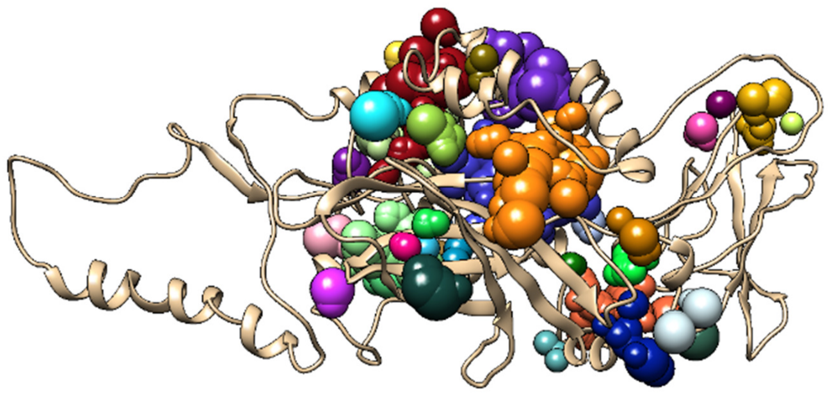


Figure 4.6:Image of docking search space after DogSite being applied - shown is an example of an output of DogSite's potential active site search which would then be inputted as a search areas for AutoDock Vina and DOCK 6.0 would attempt to dock a ligand. Each colour of bundle spheres represents a different potential active site.

the overall binding energy. Both of these new scoring methodologies produce multiple different scoring functions. The scoring functions shown in Table 2.1 were applied to the newly homology modelled protein. These scoring functions will also further help to determine if there is an interaction between the ligand and the protein.

4.5.2 Updated Homology Protein models

All sequences that were previously modelled were re-modelled using the same homology modelling pipeline but with updated BLAST databases (September 2017) and updated Protein Data Bank database (September 2017). As these databases are updated regularly, it could make a difference in the quality of each model and could increase the number of models that have 80% or above structural content (2013). Shown in figure 4.7 is a graph that shows the new homology modelling coverage of the whole proteome set (2017). Overall, 16,806 of the 20,386 sequences modelled have generated a model. The number of created models is an increase of 7,000 models compared to the original run in 2013. The result provides more evidence that homology modelling is improving all the time as more and more protein structures are being added to the PDB database from structural determination studies, leading to greater availability of more suitable templates for homology modelling. In the baseline homology model set results, only 17 of the 67 known protein targets were within the top bracket of structural coverage of 80% or more. In the new homology modelling set, 20 of the 67 known protein targets are within the top bracket of structure coverage. Figure 4.7 shows that the number of good models generated has increased from the previous 3,420 in the original set (2013) to 6,653 good models in the new set (2017).

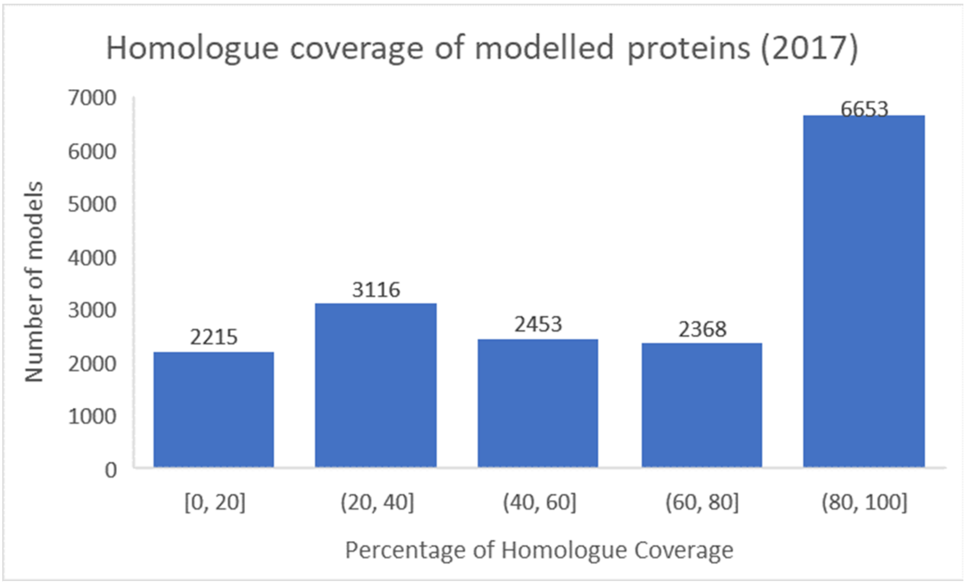


Figure 4.5: Homologue coverage of 2017 re-modelled proteins – the graph shows the different percentage groups of structural coverage and how many of the 16,806 homology models were classified in each group.

4.5.3 New homology modelling set molecular docking results

The new set of 6,653 high-quality homology models, from the 16,806 new homology models were then docked against the top 20 ligands using the same method and scoring functions. For the following results, only the affinity scoring functions for each docking scoring method (Vina, DOCK, GBSA, DrugScore X) will be analysed. The reason being that these are the primary scoring function for determining if a protein-ligand docking has had an interaction. For the new results, the overall score rule applied to the docking data is If any of the affinity "hit" columns have a TRUE within it, then the overall meta-score will equal TRUE and If any of the affinity "med" columns have a TRUE within it, then the overall meta-score will equal TRUE. Figure 4.9 shows the predicted TRUE's and FALSE's for the "hit" scoring function columns for Shipyard's predicted scores for the new homology model set (2017). The rules applied to the docking scoring function for the "hit" column were if a docking pairs score is -2 or more standard deviations away from the mean, then the scoring function classifies the docking pair as a TRUE. Figure 4.10 shows the predicted TRUE's and FALSE's for the "med" scoring function columns for Shipyard's predicted scores for the new homology model set (2017). The rules applied to the docking scoring function for the "med" column were if a docking pairs score is between -1 and -2 standard deviations away from the mean, then the scoring function classifies the docking pair as a TRUE. The table used to generate figure 4.9 and 4.10 is shown in the Appendix Table 8.2 and 8.3.

The DOCK 6.0 Affinity hit scoring function predicted 2 of the 23 known protein interactions as a hit and these being the same ones that previously were predicted correctly in the original 3,420 high-quality homology models set. The DOCK 6.0 med predicted Six of the 23 known targets as a potential interaction. These DOCK affinity hit results were an improvement from the previous 3,420 high-quality homology model set as DOCK did not predict any of the known protein interactions as a hit. The new AutoDock Vina hit scoring function predicted 3 of the 23 known protein interactions as hits The AutoDock Vina med scoring function predicted 6 of the 20 known targets as a potential interaction. The AutoDock Vina affinity hit results were also an improvement on the previous 3,420 high-quality homology model set, where the AutoDock Vina hit scoring function did not predict any of the known targets as a hit. The GBSA Affinity hit scoring function predicted 1 of the 23 known targets as a hit. The GBSA Affinity med scoring function predicted six of the 23 known targets as a potential interaction. As the GBSA affinity hit scoring function is a new scoring function added, there was no previous data to compare it. Finally, the Drug Score X hit scoring function predicted none of the known targets as a hit. The Drug Score X med scoring function predicted 6 of 23 known targets as potential interactions. As the DSX affinity score is a new scoring function added, there is no previous data to compare it to, but it has performed poorly compared to the other three scoring functions being analysed.

The next columns of the docking scores shown in figure 4.9 are the Meta-score-hit. Using the rule applied, 4 of the 23 known targets were predicted as a hit. The last column shown in figure 4.10 is the meta-score-med. Using the same rule as applied previously, 13 of the 23 known targets were predicted as potential interactions. These results are an improvement compared to the baseline high-quality homology model set results as neither AutoDock Vina or DOCK 6.0 binding affinity 4.5.3.1 scores classified any of the known protein interactions as hits. Overall there had been a slight improvement to the results with the new homology models (2017) that were created using the homology modelling pipeline but not nearly as much as had been hoped. The next step was to investigate if the docking algorithms were correctly predicting known misses of each of the ligands, that is to investigate whether the algorithms are producing false positives for proteins that are known to not interact with ligands.

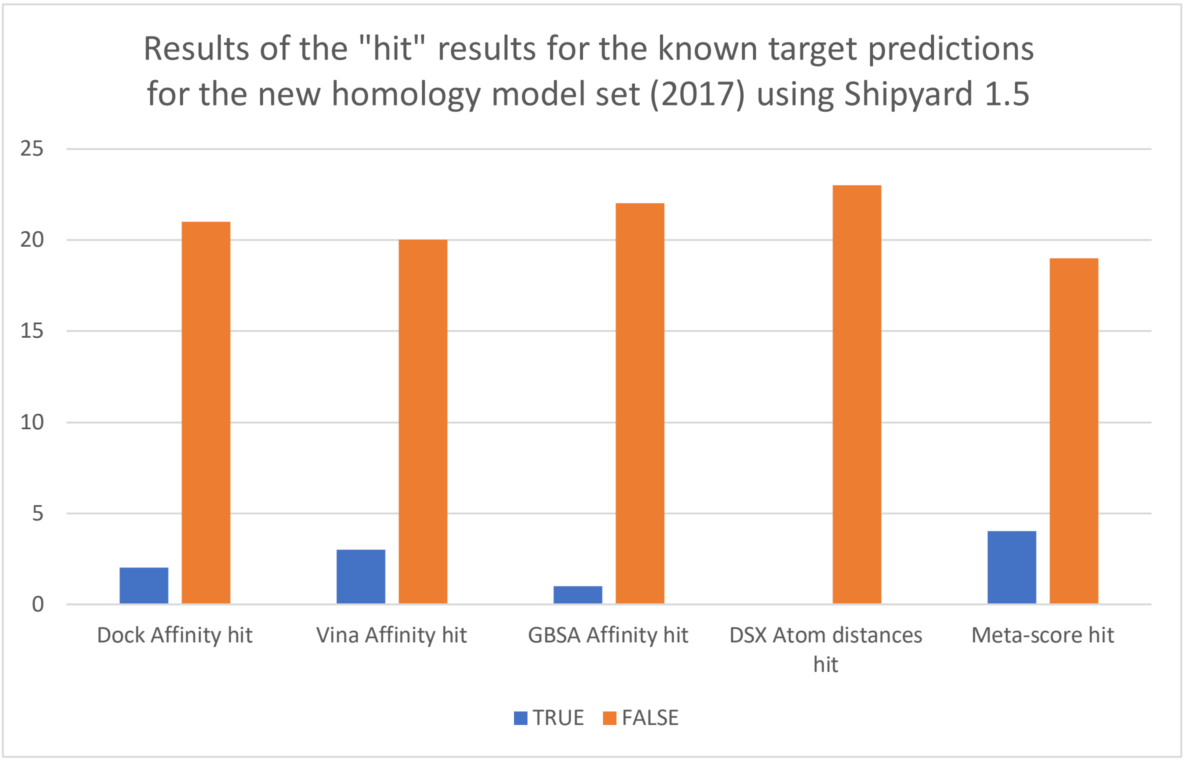


Figure 4.6: the "hit" result for the Scores produced by Shipyard for the known protein interactions using the new homology modelling set (2017) – shown in the graph are hit results for the analysed scoring functions produced by Shipyard 1.5. the graph displays the number of TRUE's and FALSE's every scoring function predicted by following the rule of the docking pair score being between -1 and -2 standard deviations from the mean. The meta-score med scoring function follows a different rule of If any of the affinity "hit" rule results have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in Appendix Table 8.1.

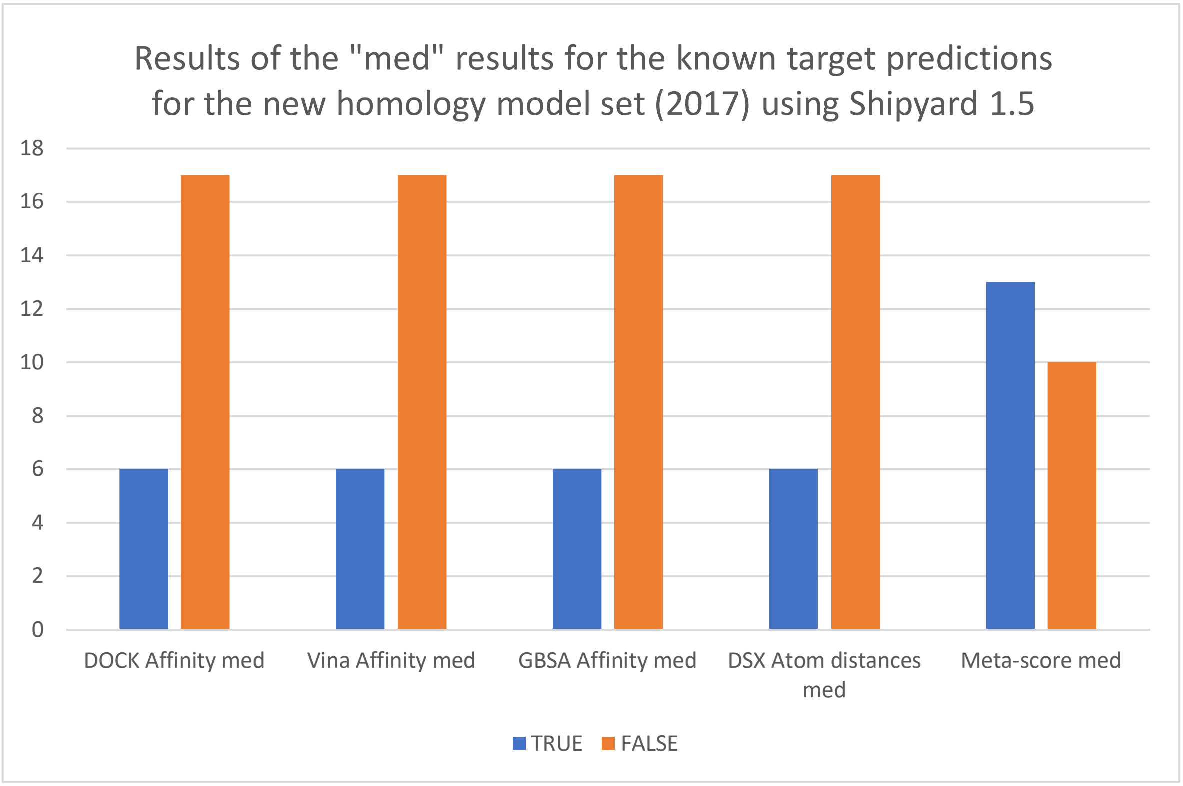


Figure 4.7: the med result for the scores produced by Shipyard for the known protein interactions using the new homology modelling set (2017) – shown in the graph are med results for the analysed scoring functions produced by Shipyard 1.5. the graph displays the number of TRUE's and FALSE's every scoring function predicted by following the rule of the docking pair score being between -1and -2 standard deviations from the mean. The meta-score med scoring function follows a different rule of If any of the affinity "med" rule results have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in Appendix Table 8.2.

4.5.3.1 Known protein misses for the top 20 most valuable sold drugs

As well as known interactions confirmed by *in vitro* work, each of the top 20 most valuable sold drugs also has known misses which have been stored in databases such as DrugBank (Wishart et al., 2017) and ToxCast (Sipes et al., 2013). Known misses are ligands that have been tested against these proteins *in* vitro and have provided a non-interaction. With this information, it is possible to extract the relevant protein-ligand data from the docking data for comparison. The same rule that has been applied to the known protein interaction results previously in section 4.3.1.1 will also be applied to the known protein misses' results. Shown in figure 4.11 are hit results and shown in figure 4.12 are the med results for the known misses that have had dockings performed for their respective ligands. Only 9 of the known misses were included in the 6,653 set, but 7 of these were for the same protein, namely Nuclear factor NF-kappa-B p105 subunit. With the Dock Affinity hit score scoring function, 8 of the known misses were correctly classified as FALSE. The docking pair that gave a false positive result was the ligand Imatinib (PubChem ID: 5291) against the protein Nuclear factor NF-kappa-B p105 subunit (UniProt code: P19838). For the Dock Affinity med score, It classified 3 of the 9 known non-interactions as a TRUE. The Dock affinity result only predicting one as a hit overall. AutoDock Vina Affinity hit score classified 2 of the 9 known non-interactions as TRUE. The AutoDock Affinity med score classified 1 of the 9 non-interactions as a potential interaction. The predicted hits were Imatinib/ Nuclear factor NF-kappa-B p105 subunit docking (same as the Dock algorithm) but with the addition of the Lenalidomide (PubChem ID: 216326) and Nuclear factor NF-kappa-B p105 subunit (UniProt code: P19838) docking. The GBSA Affinity hit score predicted 1 of the 9 known non-interactions as TRUE. The GBSA Affinity med score predicted two of the 9 as a potential interaction. The docking that had predicted as a hit is the same conformation as AutoDock Vina results predicted (Lenalidomide/ Nuclear factor NF-kappa-B p105 subunit). Finally, Drug Score X hit scores predicted 2 of the 9 known non-interactions as TRUE, and the Drug Score X med score predicted three of the 9 as potential interactions. One of these dockings predicted as a hit is the Lenalidomide/ Nuclear factor NF-kappa-B p105 subunit docking which has consistently been predicted as a hit with 3 of the 4 scoring functions, the other being the Levothyroxine (PubChem ID: 5819) against the protein Transforming growth factor beta-1 (UniProt: P01137) which was the first time this pair had been flagged as a hit. Overall the meta-score hit column shown in figure 4.11 predicted 3 of the nine known misses as a hit. To possibly discover the reason for the three false positives, the predicted dockings were examined to check for any anomalies more closely. The Meta-score med column shown in figure 4.12 predicted 5 of the 9 known non-interactions as a potential interaction.

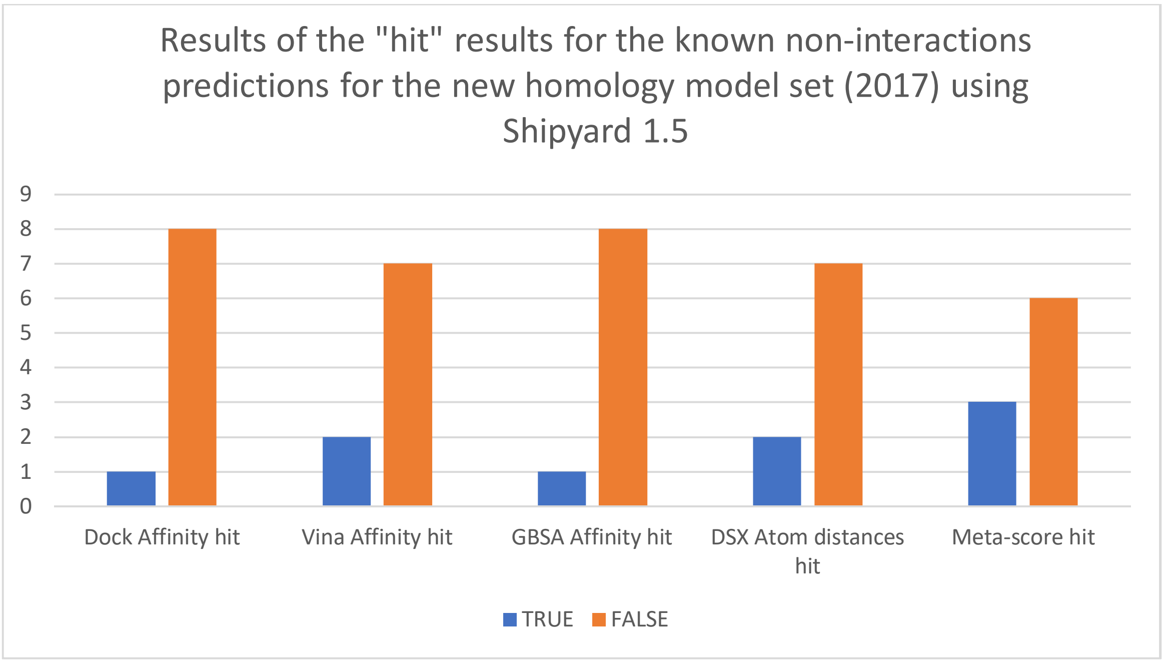


Figure 4.8: the hit result for the Scores produced by Shipyard for the known protein non-interactions using the new homology modelling set (2017) – shown in the graph are hit results for the analysed scoring functions produced by Shipyard 1.5. the graph displays the number of TRUE's and FALSE's every scoring function predicted by following the rule of the docking pair score being between -1and -2 standard deviations from the mean. The meta-score hit scoring function follows a different rule of If any of the affinity "hit" rule results have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in Appendix Table 8.3.

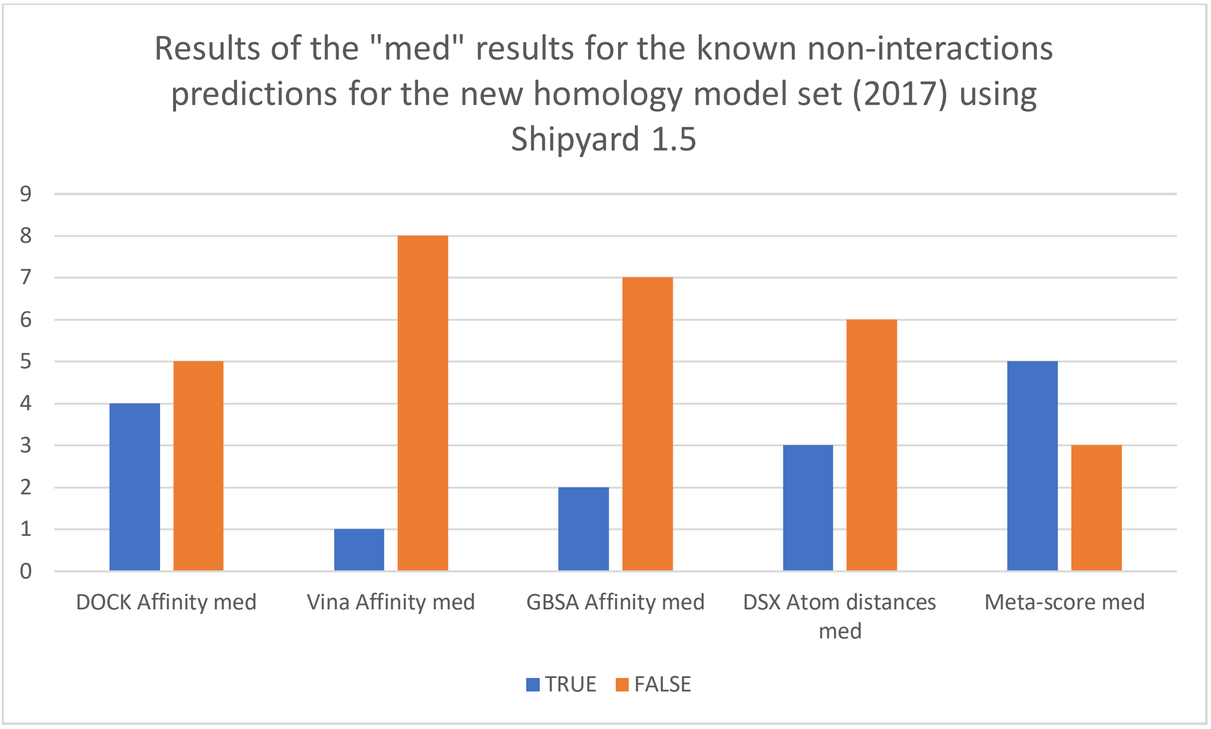


Figure 4.9: the med result for the Scores produced by Shipyard for the known protein non-interactions using the new homology modelling set (2017) – shown in the graph are med results for the analysed scoring functions produced by Shipyard 1.5. the graph displays the number of TRUE's and FALSE's every scoring function predicted by following the rule of the docking pair score being between -1 and -2 standard deviations from the mean. The meta-score hit scoring function follows a different rule of If any of the affinity "med" rule results have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in Appendix Table 8.4.

One of the dockings that were investigated was the Lenalidomide/ Nuclear factor NF-kappa-B p105 subunit docking as this docking was predicted to be a hit with 3 of the four scoring functions used. Images of the docking are shown in figures 4.13 and 4.14 where the ligand is being docked in a region where there is low homologue coverage which can affect the overall docking result. Low homologue coverage areas can affect docking as the algorithm might incorrectly regard a particular

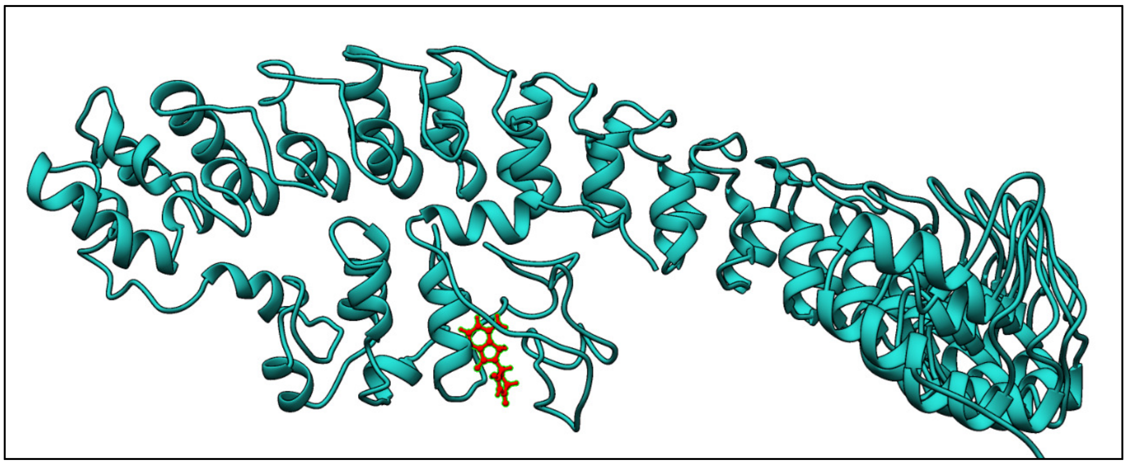


Figure 4.10: Docking conformation for the Lenalidomide/ Nuclear factor NF-kappa-B p105 subunit – shown is the generated conformation for Lenalidomide/ Nuclear factor NF-kappa-B p105 subunit where the ligand Lenalidomide has docked near a low homologue coverage area of the protein.

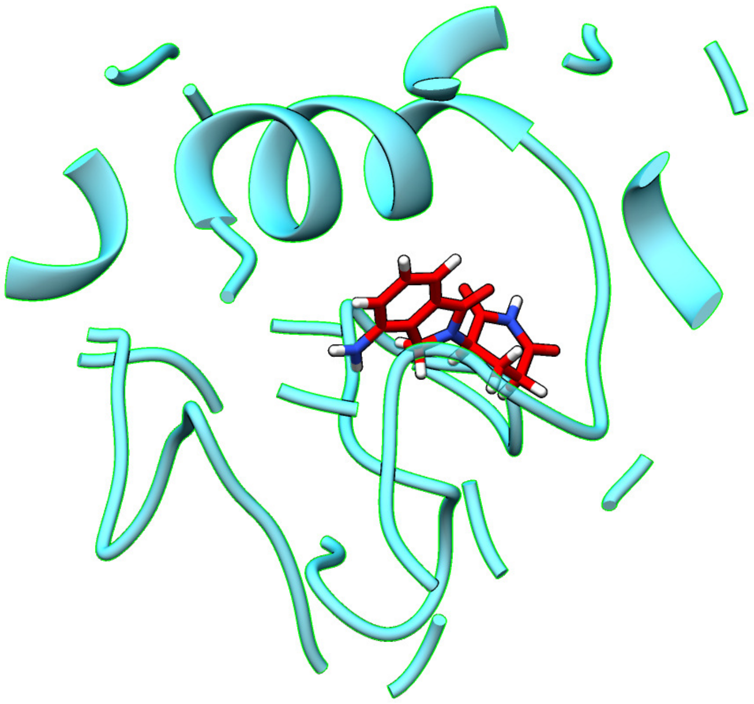


Figure 4.11: Docking conformation for the Lenalidomide/ Nuclear factor NF-kappa-B p105 subunit close up – shown is a close up of the generated conformation for Lenalidomide/ Nuclear factor NF-kappa-B p105 subunit where the ligand Lenalidomide has docked near a low homologue coverage area of the protein.

region area as a potential docking site. The low homologue coverage area might be the reason why this docking is being predicted as a false positive. The analysis outcome led to the development of a docking analysis script.

4.5.3.2 Development of a docking analysis script

To investigate if drugs were docking in low homologue areas of the protein might be the cause for the poor docking performance, a python script was developed that could analyse each predicted docking conformation. This script would be able to parse through every docking conformation predicted through Shipyard 1.5 and remove docking results that could be classed as incorrect due to the ligand being docked in a location with low homologue coverage. The script first considers each protein-ligand pair and calculates the distances of each residue of the protein from the ligand docked with it. The residues that are within 12 angstroms of the ligand are then put into a list. The 12 angstrom threshold is based on the upper end of the range of amino acid side chain length (to include regions of models where the position and conformation of the backbone is computed accurately, but where the exact positions of the side chains may not have been computed accurately), added to the typical distance range of interaction applied in analysis of docking results, which is around 4 angstroms. This list of residues is cross-referenced with data produced by the DSSP program (Touw et al., 2014) to see if there is a known structure related to those residues. When this phase is over, a rule is then applied to the effect that if at least 75% or more of the known binding location has a structure, then the docking result is kept, but if not, the entire docking pair conformation result is discarded.

4.5.3.3 New Homology model set (2017) after docking location analysis

The newly formed CSV file for the new homology model set (2017) only contains the dockings that had good structure around the docking site. The same rules and methods that were applied to the new homology set (2017) before the docking location analysis was also applied to the new extracted homology model set.

After the docking location analysis script had been applied to the new homology model set (2017), a new data file was generated. The new data consisted of 79,000 results compared to the original 128,400 results. The outcome showed that 38.5% of all the new homology model set results had drugs that were docked in low homologue coverage areas of the protein. For the analysis, the known protein interaction data were extracted from the newly created CSV file. Shown in table 8 are the results for the known interactions. Shown in figure 4.15 are the hit results for the newly revised homology docking set, and the results turned out the same with 4 of the known protein

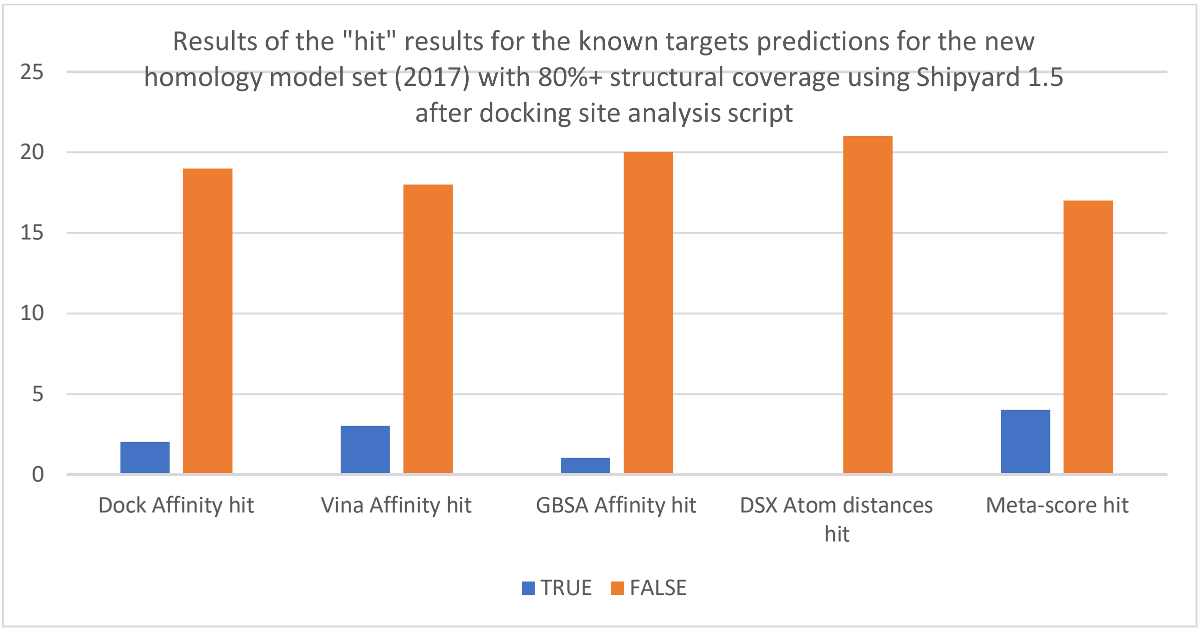


Figure 4.12: the hit result for the Scores produced by Shipyard for the known protein targets using the new homology modelling set (2017) with 80%+ structural coverage after docking site analysis script – shown in the graph are hit results for the analysed scoring functions produced by Shipyard 1.5 after docking site analysis script. the graph displays the number of TRUE's and FALSE's every scoring function predicted by following the rule of the docking pair score being between -1 and -2 standard deviations from the mean. The meta-score hit scoring function follows a different rule of If any of the affinity "hit" rule results have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in Appendix Table 8.5.

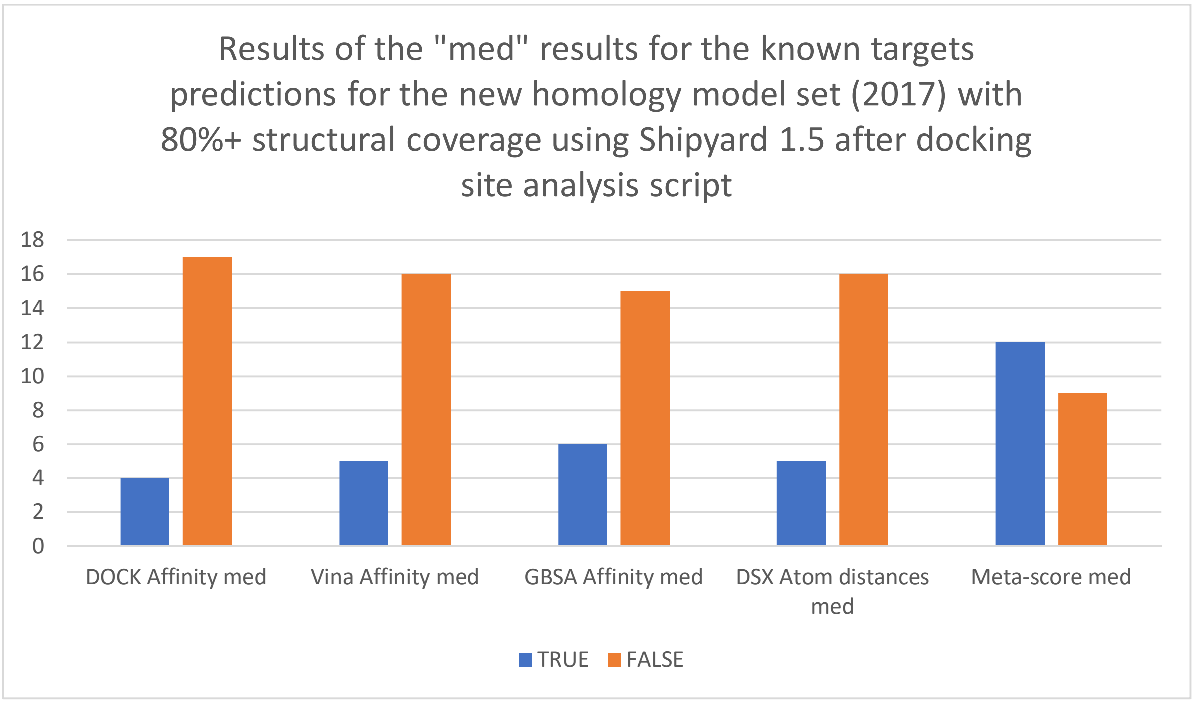


Figure 4.13: the med result for the Scores produced by Shipyard for the known protein targets using the new homology modelling set (2017) with 80%+ structural coverage after docking site analysis script – shown in the graph are med results for the analysed scoring functions produced by Shipyard 1.5 after docking site analysis script. the graph displays the number of TRUE's and FALSE's every scoring function predicted by following the rule of the docking pair score being between -1 and -2 standard deviations from the mean. The meta-score med scoring function follows a different rule of If any of the affinity "med" rule results have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in Appendix Table 8.6.

targets being classified as a hit. shown in figure 4.16 are the med results for the same revised set which 12 of the 21 known targets remaining as a TRUE. Only 2 known protein interaction dockings were lost after running the docking location analysis script. As it seemed that homology models could be causing an issue, analysis began on the protein models that had been generated using protein threading.

4.6 Proteins Modelled by Protein Threading

To see if there is a difference in accuracy, the models generated by protein threading were put through the same steps as the homology modelling set. This set contained the protein models that could not be modelled by the homology modelling pipeline from the baseline results in section 4.4.2 which consisted of 10,839 sequences, 5,901 protein sequences that produced homology models with less than 80% structural coverage, and also included the 67 known targets for the top 20 most valuable sold drugs. This set was used in docking against the 20 most valuable sold drugs using the same method as the homology modelling set.

4.6.1 Protein threading model quality

The first step was to run the model structure coverage script that was run on the homology modelling set on the new set that had undergone protein threading. A total of 7,821 models were generated using protein threading. Figure 4.16 shows the results of the structure coverage check. Compared to the homology modelling set, nearly all models processed by the script had 80% or

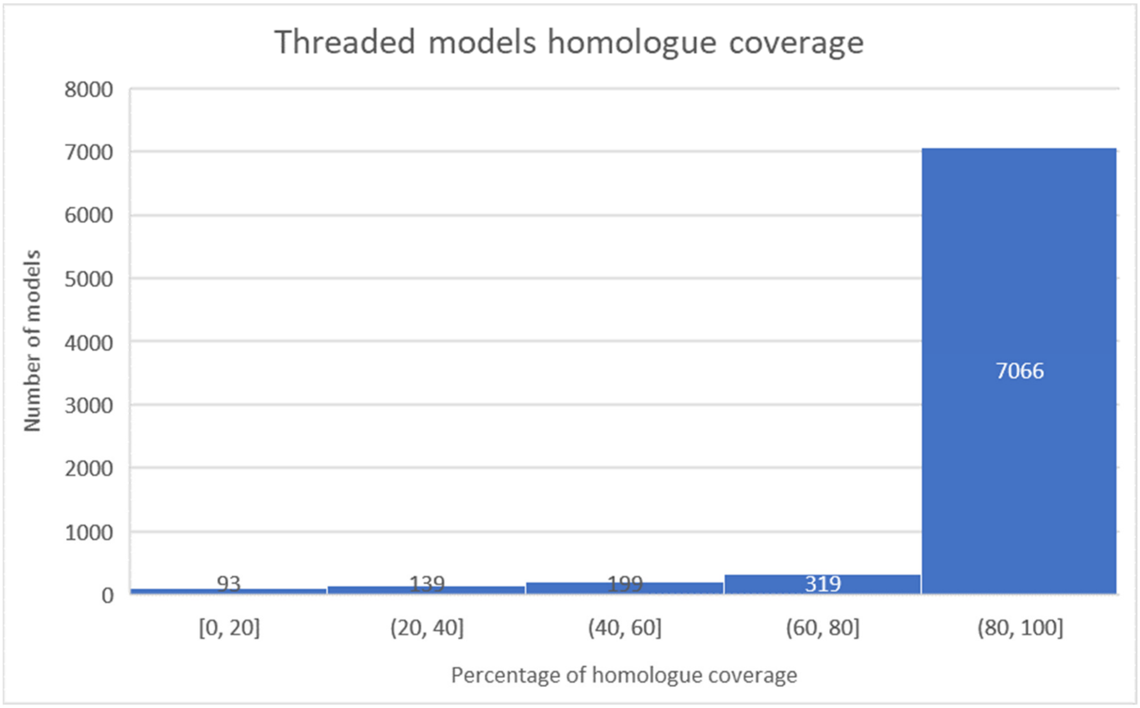


Figure 4.14: Percentages of homologue coverage for the 7,821 threaded model set – shown are the numbers of protein models within the protein threaded sequences which were within each structural coverage percentage group. The majority of the models possessed 80% or more structural coverage.

more homologue coverage with only 750 models (less than 10%) not being given a high-quality model status. Compared to the homology modelling set this is a much more significant percentage of models being given a high-quality model status, only 36.69% of the homology models were given good models status compared to the threaded models of which 90.35% were given good models status. These results are a massive improvement on the homology modelling pipeline results. Like the homology model's set, only the high-quality models will be used and docked against the top 20 most valuable sold drugs using the Shipyard 1.5 pipeline. Firstly, the known protein interactions were extracted from the overall set that had undergone protein threading.

4.6.2 Known protein interaction results

Shown in figure 4.17 are the hit column results for the extracted results for the known protein interactions within the set that had undergone protein threading. Like the homology modelling set discussed in section 4.5.3, the same methods and rules were applied to the results. The two standard deviation rules were only looking at the affinity scores of the four types of scoring functions (Vina, DOCK, GBSA, DSX). There was an increase in known target proteins within the high-quality model's bracket compared to the homology modelling set. The high-quality set that had undergone protein threading contained 33 of the 67 known protein interactions. The results have shown an increase of 19.4% of the known protein targets generating models. Looking firstly at the DOCK Affinity hit results, only two of the 33 known targets in the set were predicted correctly as TRUE these being Histamine H1 receptor (UniProt: P35367) with the ligand Aripiprazole (PubChem: 60795) and Dipeptidyl peptidase 4 (UniProt: P27487) and Atorvastatin (PubChem: 60823). Shown in figure 4.18 were the med column results for the protein threaded set. The DOCK Affinity med results predicted 10 of the 33 known targets as a TRUE which is a potential interaction. AutoDock Vina hit results predicted 2 of the 33 known targets as a TRUE, these being the Tumour necrosis factor ligand superfamily member 11 (Uniprot: O14788) and Lenalidomide (PubChem: 216326) pairing and the Synaptic vesicle glycoprotein 2A (Uniprot: Q7L0J3) and Levetiracetam (PubChem: 441341) pairing. The AutoDock Vina med results predicted 15 of the 33 known targets a TRUE and potential interaction. The GBSA affinity hit scoring function predicted 3 of the 33 known targets as TRUE's these being Steroid 17-alpha-hydroxylase/17,20 lyase (UniProt: P05093) and Abiraterone (PubChem: 132971), Cytosolic phospholipase A2 (UniProt: P47712) and Fluticasone Propionate (PubChem: 444036), and Histamine H1 receptor (UniProt: P35367) and Aripiprazole (PubChem: 60795) which are shown in table "" of the Appendix. The GBSA affinity med results predicted 5 of the 33 as a potential interaction.

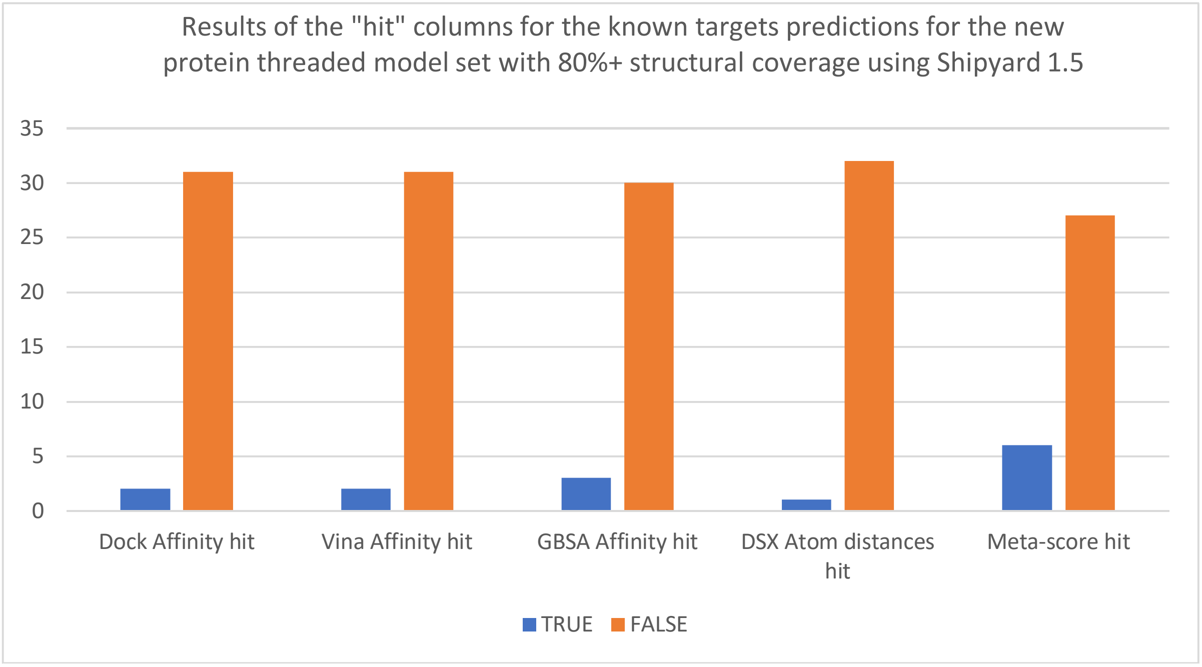


Figure 4.15: the hit result for the Scores produced by Shipyard for the known protein targets using the new protein threaded model set with 80%+ structural coverage using Shipyard 1.5 – shown in the graph are hit results for the analysed scoring functions produced by Shipyard 1.5 after docking site analysis script. the graph displays the number of TRUE's and FALSE's each scoring function predicted by following the rule of the docking pair score being between -1 and -2 standard deviations from the mean. The meta-score hit scoring function follows a different rule of If any of the affinity "hit" column s have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in the Appendix Table 8.7, 8.8, and 8.9.

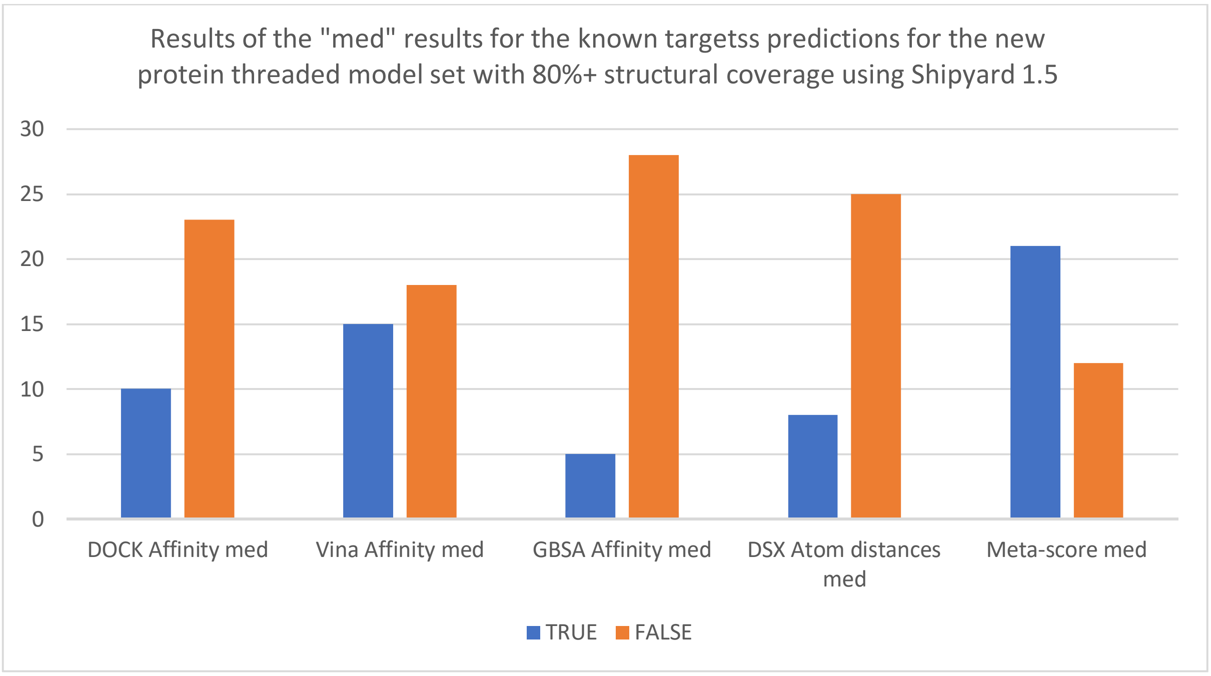


Figure 4.16: the med result for the Scores produced by Shipyard for the known protein targets using the new protein threaded model set with 80%+ structural coverage using Shipyard 1.5 – shown in the graph are med results for the analysed scoring functions produced by Shipyard 1.5 after docking site analysis script. the graph displays the number of TRUE's and FALSE's every scoring function predicted by following the rule of the docking pair score being between -1 and -2 standard deviations from the mean. The meta-score med scoring function follows a different rule of If any of the affinity "med" rule results have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in Appendix Table 8.10, 8.11, and 8.12.

The DSX Atom distances hit scoring function predicted 1 of the 33 known targets as a hit, this being the pairing of Cytosolic phospholipase A2 (UniProt: P47712) and Fluticasone Propionate (PubChem: 444036). The DSX Atom distances med results predicted 8 of the 33 known targets as a potential interaction.

These final two columns are the "Meta-score-hit" and "Meta-score-med" (potential interaction). The columns followed the same rules as previously for the homology modelling protein sets this being if at least one of the scoring functions have a TRUE in it, it will be classed as a TRUE overall. The columns with "hit" in the name will be linked to the Meta-score column and the columns with "med" will be linked to the Meta-score-med column. The Meta-score column predicted 6 of the known targets as a hit. The result is an increase of two compared with the 2017 homology modelling set. The Meta-score-med column predicted 21 of the 33 known targets as dockings to investigate. Comparing this to the 2017 homology modelling, this is an increase of 8 targets being identified as a potential interaction.

4.6.3 Known Protein non-interaction results

The known protein misses were also extracted from the set that had undergone protein threading so that they could also be analysed. Shown in figure 4.19 are the results for hit columns for the known non-protein interactions for the set that had undergone protein threading. There were 71 known protein-ligand misses within the high-quality set that had undergone protein threading set, compared to only 9 in the 2017 homology modelling set. The number of models generated for the known protein misses is a substantial increase of protein-ligand interaction models for testing. A miss is denoted by a "FALSE" statement in the columns. The DOCK affinity hit results correctly predicted 67 of the 71 known non-interactions as a FALSE. The results by DOCK gave an overall accuracy of 94.3% for correctly classifying a known interaction correctly. Within the DOCK affinity "med" column (potential interaction) shown in figure 4.20, 13 of the 67 known non-interactions were classified as potential interactions. The Vina Affinity "hit" column, 69 of the 71 were predicted correctly as a miss. The Vina binding affinity has given an accuracy of 97.2% for predicting non-interactions correctly, which is very good. For the Vina Affinity med column (potential interaction), 18 of the 67 known non-interactions were classified as potential interactions.

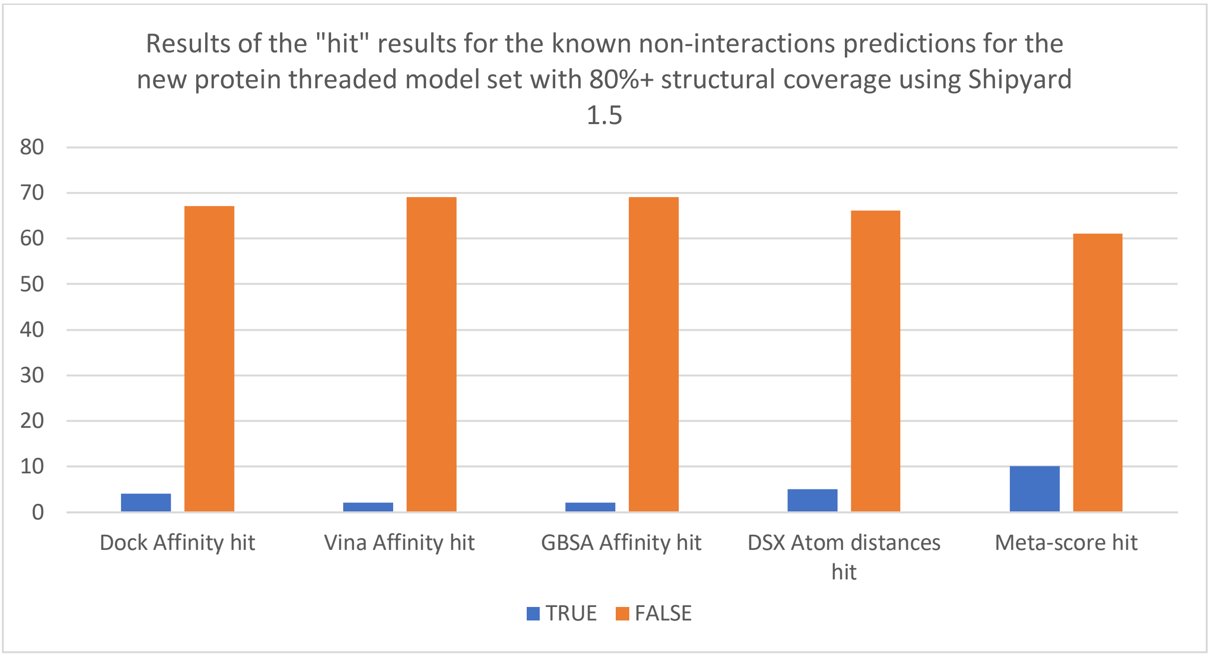


Figure 4.17: the hit result for the Scores produced by Shipyard for the known non-interactions predictions for the new protein threaded model set with 80%+ structural coverage – shown in the graph are hit results for the analysed scoring functions produced by Shipyard 1.5 after docking site analysis script. the graph displays the number of TRUE's and FALSE's every scoring function predicted by following the rule of the docking pair score being between -1 and -2 standard deviations from the mean. The meta-score hit scoring function follows a different rule of If any of the affinity "hit" rule results have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in Appendix Table 8.13, 8.14, and 8.15.

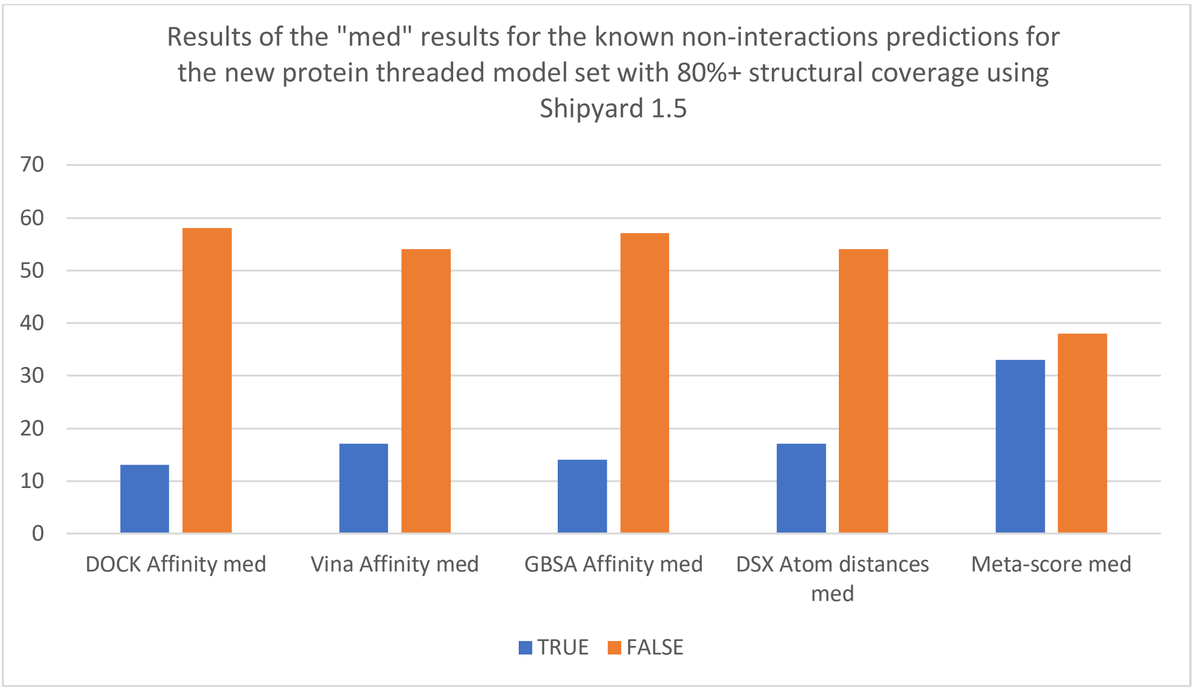


Figure 4.18: the med result for the Scores produced by Shipyard for the known non-interactions predictions for the new protein threaded model set with 80%+ structural coverage – shown in the graph are med results for the analysed scoring functions produced by Shipyard 1.5 after docking site analysis script. the graph displays the number of TRUE's and FALSE's every scoring function predicted by following the rule of the docking pair score being between -1 and -2 standard deviations from the mean. The meta-score med scoring function follows a different rule of If any of the affinity "med" rule results have a TRUE within it, then the overall meta-score will equal TRUE. The data used to create the graph is shown in Appendix Table 8.16, 8.17, and 8.18.

The GBSA affinity hit column, 69 of the 71 were predicted correctly as a non-interaction. The GBSA affinity has predicted the same number of known non-interactions correctly as the Vina binding affinity. The GBSA affinity med results predicted 57 of the 71 known non-interactions correctly as a FALSE. For the DSX atom distance column, the hit column shown in figure 4.19, predicted 66 of 71 known misses correctly, which is a 92.5% accuracy. The DSX atom distance med column predicted 16 of the 71 known non-interactions as potential interactions. Finally, the meta-score hit (figure 4.19) and meta-score-med (potential interaction) (figure 4.20) columns. The same rule as was applied to the homology models set (2017) was applied to the high-quality set that had undergone protein threading. Firstly, looking at the Meta-Score hit column, 61 of the 71 known non-interactions were correctly classified which is an accuracy of 86%. For the meta-score-med column 31 of the 71 were classified as potential interactions.

These results appear to show that overall, the set that had undergone protein threading did not perform better than the homology models for both the 2013 and 2017 sets. Comparing the known targets which overlap the 2017 homology modelling and the protein threaded sets, the accuracy had marginally increased with 18.2% of the known protein interactions being correctly classified compared to the homology modelling models set that only predicted 17.4% correctly. This is not a significant improvement which would indicate that one of the modelling methodologies are better than the other for helping to generate higher accuracy docking conformations. From the following results it has determined that the method used for the analysis of both the homology model (2017) and the set that had undergone protein threading is not suitable to determine interactions. With this conclusion, another analysis method was investigated to attempt to increase docking accuracy.

4.7 Development of a second docking results analysis method

A new method known as Sensitivity (true positive rate) and specificity (true negative rate) was applied to the docking conformations generated for the 2017 homology modelling and protein threaded sets with the aim to improve the accuracy of the docking algorithms. These are used to measure the actual positives that are being correctly predicted and true misses that are correctly being predicted. With the collected data of both the 2017 homology modelled and protein threaded sets, firstly there is a need to pick the best scoring functions within the sets that are the most accurate in predicting the known targets and known misses correctly. To do this a Receiver operating characteristic curve (ROC Curve) needs to be created for each of the scoring functions separately. A ROC curve is a way of plotting the sensitivity tests (true positive rate) which are plotted on the y-axis and the specificity (false positive rate) plotted on the x-axis. The ROC method can evaluate the quality or performance of a diagnostic test. (Park, Goo and Jo, 2004)

The data used to create the ROC curve were the known targets of the top 20 drugs and the known non-interactions for the top 20 most widely prescribed drugs. Within the protein threaded set there were 33 known targets and 70 known non-interaction, and within the 2017 homology modelling set, 21 known targets and 33 known non-interaction conformations have been generated. A python script was created to calculate out the ROC curve and the area under the curve. Calculating the area under the curve helps identify the best scoring functions that predict a known target correctly as well as predicting known protein-ligand non-interaction correctly. Shown in table 4.5 are the Area Under the Curve scores given for the high-quality homology model set (2017) and in figure 4.6 shows the Area Under the Curve scores for the set of high-quality models which had undergone protein threading.

For this analysis, the scoring functions with a result of 0.7 or higher will be used. The most accurate scoring functions are used for the next part of the analysis. The most accurate scoring functions for the high-quality homology model set are the GBSA Affinity, GBSA Solvent-accessible surface area, Vina Gauss 1, Vina Gauss 2, Vina Rotatable Bond entropy, and the DSX Contact count.

Table 4.5: Area Under the Curve Scores for high-quality homology modelling set (2017) – shown are all of the results after the ROC and AUC calculations. The best scoring functions were the GBSA Affinity, GBSA Solvent-accessible surface area, Vina Gauss 1, Vina Gauss 2, Vina Rotatable Bond entropy, and the DSX Contact count.

Scoring Function	**Area Under the curve**
DOCK Affinity	0.65
Vina Affinity	0.62
GBSA Affinity	0.76
DSX Atom distances	0.86
DOCK van der Waals	0.66
DOCK Electrostatics	0.51
DOCK internal Energy	0.60
GBSA van der Waals	0.65
GBSA Screened electrostatics	0.53
GBSA Born Solvation area	0.57
GBSA Solvent-accessible Surface Area	0.73
Vina Gauss 1	0.74
Vina Gauss 2	0.78
Vina Repulsion	0.66
Vina Hydrophobic	0.60
Vina H-Bond	0.54
Vina Rotatable Bond entropy	0.74
DSX Contact count	0.83
DSX per contact	0.70

The most accurate scoring functions for the set of high-quality models which had undergone protein threading are the Vina Affinity, GBSA Affinity, DSX Atom distances, Vina Gauss 1, Vina Gauss 2, and the DSX Contact count. Comparing the two lists of scoring functions, it seems that the scoring functions GBSA Affinity, Vina Gauss 1, Vina Gauss 2, and DSX Contact count are the most accurate in both instances. These scoring functions will now be used to do some combinational forecasting to see if it is possible to make a model that is more accurate to pick out known protein interactions and non-interactions correctly.

Table 4.6: Area Under the Curve Scores for a set of models which have undergone protein threading - shown are all of the results after the ROC, and AUC calculations for the protein threaded set. The best scoring functions were the Vina Affinity, GBSA Affinity, DSX Atom distances, Vina Gauss 1, Vina Gauss 2, and the DSX Contact count.

Scoring Function	Area Under the curve
DOCK Affinity	0.68
Vina Affinity	0.71
GBSA Affinity	0.75
DSX Atom distances	0.72
DOCK van der Waals	0.68
DOCK Electrostatics	0.56
DOCK internal Energy	0.52
GBSA van der Waals	0.69
GBSA Screened electrostatics	0.59
GBSA Born Solvation area	0.62
GBSA Solvent-accessible Surface Area	0.65
Vina Gauss 1	0.75
Vina Gauss 2	0.71
Vina Repulsion	0.54
Vina Hydrophobic	0.68
Vina H-Bond	0.67
Vina Rotatable Bond entropy	0.65
DSX Contact count	0.73
DSX per contact	0.56

4.7.1 Analysis of combinational forecasting for the high-quality homology model set

For the high-quality homology model set selected scoring functions that have been determined as the most accurate, thresholds were then calculated for each. The threshold was determined by which score produced the most accurate predictions of targets/misses. A python script was created to work out the Area Under the Curve of the ROC curve for each of the scoring function. The python script would run through a variety of thresholds between the lowest and the highest scores within that scoring function to determine the best threshold. Shown in table 4.7 is the determined threshold that was used for each of the scoring functions.

These scores will now be used to help predict the known targets and misses to the highest accuracy. For the scores in the selected threshold, if a docking pair score is less than the calculated threshold

Table 4.7: Threshold scores selected for the combinational forecasting for the high-quality homology model dataset.

Scoring Function	Threshold
GBSA Affinity	-23.9
GBSA Solvent-accessible surface area	-4.6
Vina Gauss 1	-3
Vina Gauss 2	-6.3
Vina Rotatable Bond entropy	3.5
DSX Contact count	634.4

for that particular scoring function, the score will be classed as true. However, if the score is higher than the calculated threshold for that scoring function, that score will be classed as false. The thresholds for each scoring function determined are used to attempt to increase the accuracy of the selected scores by using combinational forecasting. Combinational forecasting is a method of making predictions. It refers to the idea that predictions can be made from a consensus of models rather than just one model. In this case, the models are the scoring functions. These rules help the combinational forecasting determine if a docking pair is a potential interaction or a non-interaction. The method that combinational forecasting uses is to set criteria of how many of the scoring functions have to be classified as a true for the docking pair to be determined as a potential interaction. As the best criteria had not been identified for the combinational forecasting predictions, a test was carried out on every possible criterion. Shown in table 4.13 are all the possible criteria for the combinational forecasting and the percentages of accuracy.

The criteria that were tested were if 1/6 of the scoring function are classified as true; the overall prediction is true. If 2/6 of the scoring function are classified as true, the overall prediction is true. If 3/6 of the scoring function are classified as true, the overall prediction is true. If 4/6 of the scoring function are classified as true, the overall prediction is true. If 5/6 of the scoring function are classified as true, the overall prediction is true. If 6/6 of the scoring function are classified as true, the overall prediction is true. From the results shown in table 4.8, it can be determined that the most accurate criteria to be used for the combinational forecasting is 4/6 of the scoring results have to be classified as a true for the overall prediction to be classified as a potential interaction. This result was auspicious as it has increased the accuracy of predicting known protein interactions by 73.1%. The prediction of known non-interactions is still low at 42.4% which means there are still some docking pairs that are being mispredicted.

The result of the combinational forecasting is a very positive step to the aim of increasing the prediction accuracy of docking results. With an overall prediction accuracy of 66.45% with results that have not had docking sites analysed is a promising outcome. As it was previously proven that

Table 4.8: the table shows all the possible criteria options and their resulting prediction accuracies for the high-quality homology model dataset.

Criteria	Known protein interactions predicted correctly (%)	Known non-interactions correctly predicted (%)	Overall Accuracy (%)
1/6	100%	0%	50%
2/6	100%	0%	50%
3/6	100%	0%	50%
4/6	90.5%	42.4%	66.45%
5/6	76.2%	51.5%	63.85%
6/6	19%	100%	59.5%

protein threading had generated more high-quality protein models than homology modelling and there is a bigger test set of known protein interactions and known protein non-interactions, the same analysis will be carried out to determine the overall accuracy of the combinational forecasting method.

4.7.2 Analysis of combinational forecasting for the high-quality protein threading set

The analysis carried out on the high-quality homology model set will now be applied to the high-quality protein model set. The thresholds were calculated using the same method by using a ROC curve for each scoring function and calculate the area under the curve. Shown in Table 4.9 are the calculated thresholds for each selected scoring function. With the best combinational forecasting criteria not yet determined, the same test was carried out on all possible criteria options to determine the one with the highest accuracy. Shown in table 4.10 are the results for the combinational forecasting criteria tests. From the results, it was determined that the most accurate criteria to use are if 4/6 scoring functions have to be classified as true for the overall prediction to be classified as a potential interaction. Comparing the overall accuracy with that of the homology model set (2017), the set of models which had undergone protein threading combinational forecast was 5.6% more accurate. The result that gives the highest overall accuracy is that the 4/6 criteria

Table 4.9: Threshold scores selected for the combinational forecasting for the high-quality set of models which had undergone protein threading.

Scoring Function	Threshold
DSX Atom distances	-98
GBSA Affinity	34.5
Vina Affinity	-6.6
Vina Gauss 1	-2.6
Vina Gauss 2	732.2
DSX Contact count	-6.2

successfully predicted 90.9% of the known protein targets correctly, which is similar to the homology modelling set's 90.5%.

Table 4.10: the table shows all the possible criteria options and their resulting prediction accuracies for the high-quality set that had undergone protein threading dataset.

Criteria	Known protein interactions predicted correctly (%)	Known protein non-interactions predicted correctly (%)	Overall Accuracy (%)
1/6	100%	0%	50%
2/6	100%	0%	50%
3/6	90.9%	44.6%	67.75%
4/6	90.9%	53.2%	72.05%
5/6	81.8%	57.6%	69.7%
6/6	51.5%	82.7%	67.1%

4.7.3 Overall conclusion of the developed combinational forecasting method

Overall this method has shown that it can be accurate for predicting known targets and predicting known misses correctly. These results have also shown that it seems that a set of models which had undergone protein threading are more likely to predict correctly than the homology modelling models. With an overall accuracy of 72.05%, the set of models which had undergone protein threading seemed to give better docking predictions than homology.

4.7.4 Investigation of Conformations

After examining the entire set of docking results and using the combinational forecasting to get a high correct predicting percentage, the next stage is to investigate the data greater in-depth to see if the calculated docking data produces useful data for drug discovery or to repurpose drugs. For the analysis, known targets were used, including panel 44 proteins. These are looked at more closely to see how they have been docked and classified within the 7,066 high-quality threaded sets. Panel 44 is a group of 44 target proteins which are used by four large pharmaceutical companies (AstraZeneca, GlaxoSmithKline, Novartis and Pfizer) to evaluate a drug's potential for causing adverse drug events (ADR's) (Bowes et al., 2012). Within the high-quality protein threaded set are 23 of the panel 44 set in which 9 of these are known targets of the top 20 most valuable sold drugs which are listed in table 4.11. The analysis of the biological accuracy of the conformations was a good test to see how the combinational forecasting performs to detect known targets.

Results were split into four sections, those being true positive predictions, True negative predictions, false-positive predictions, and false-negative predictions. Due to there being no score as such to determine how "strong" an interaction is a heuristic has been developed to solve this problem.

Table 4.11 : table shown drug from the top 20 most widely prescribed drug and their known targets within the panel 44 set.

Protein Target	**Drug**
Prostaglandin G/H synthase 2	Lenalidomide
D(1A) dopamine receptor	Aripiprazole
D(2) dopamine receptor	Aripiprazole
Muscarinic acetylcholine receptor M3	Solifenacin, Tiotropium
5-hydroxytryptamine receptor 2A	Aripiprazole
5-hydroxytryptamine receptor 1A	Aripiprazole
Muscarinic acetylcholine receptor M1	Solifenacin, Tiotropium
Histamine H1 receptor	Aripiprazole
Muscarinic acetylcholine receptor M2	Aripiprazole, Solifenacin, Tiotropium

4.7.4.1 Development of a heuristic for determining a docking predictions strength

A heuristic indicator for predicting the strength of binding of a docked protein-ligand complex was developed. Each calculated docking score is represented in terms of standard deviations away from the mean for that scoring function. For example, the AutoDock Vina score for a protein-ligand complex would be -2.5 if it is 2.5 standard deviations lower than the mean for the distribution of all AutoDock Vina scores measured in this study. Scoring functions have natural differences in scale. For example, DOCK scores are typically around -60, whereas AutoDock Vina scores are typically around -8. By converting all the scores to the same scale of "standard deviations away from the mean", such that they are all in the approximate range -3 to +3, it is possible to meaningfully take an average of them. This step enables us to create a consensus "meta-score" (a.k.a. "z-score") later. The creation of this meta-score is now discussed.

Firstly, the "two standard deviation" (2SD) rule was applied to the scores as previously defined in section 4.3.1.1. This heuristic is worked so that if a score is better-than two standard deviations

Ligand	Protein	Score 1	Score 1 Combinational forecasting	Score 2	Score 2 combinational forecasting	Score 3	Score 3 combinational forecasting	Meta score	Overall Score
Ligand-1	Protein-1	-2.01	TRUE	-0.05	FALSE	-3.02	TRUE	TRUE	-5.03
Ligand-1	Protein-2	-0.02	FALSE	0.1	FALSE	-2.01	TRUE	FALSE	0.08
Ligand-1	Protein-3	-2.00	TRUE	-3.43	TRUE	-5.3	TRUE	TRUE	-10.73
Ligand-1	Protein-4	-0.56	FALSE	-0.15	FALSE	-0.78	FALSE	FALSE	-1.49

Table 4.12: Example of the results table that is created by the developed heuristic

below the mean, it is counted as a strong prediction of binding activity. A consensus ensemble methodology is used to create a system of voting between the three different scoring functions. If a simple majority of the scoring functions meet the 2SD rule, then the meta-score prediction is that a strong binding took place (TRUE in Table 4.12); otherwise, if there is no majority vote amongst the different score methods, the meta-score prediction will be that the docked conformation does not illustrate a binding of interest (FALSE in Table 4.12). To calculate the numerical component of the meta-score, the winning voters in the ensemble - e.g. if the meta-score prediction is TRUE, then the scores that met the 2SD rule - are summed. For example, shown in table 4.12 is an example table of the predicted results after the mean and standard deviation outlier method and the combinational forecasting has been applied. There are four rows of data, each with three scoring functions these being scored 1, score 2, and score 3. Each of these scores has a combinational forecasting result, these being Score 1 Combinational forecasting, score 2 Combinational forecasting, and Score 3 Combinational forecasting. Finally, there are two columns of meta-score and an Overall score. Meta-score is the overall combinational forecasting result for each protein pairing (If it is classified as a potential interaction or a non-interaction). The Overall score is the final interaction score that will determine the interactions “strength”. The Meta-score is a crucial element of deciding how the standard deviations are combined. The developed heuristic determines which standard deviation scores are combined depending on the result of the Meta-score. If the Meta-Score is TRUE (classified as a potential interaction), all standard deviation that has the combinational forecast result equalling TRUE will be combined. If the Meta-Score is FALSE (classified as a miss), all standard deviation that has the combinational forecast result equalling FALSE will be combined. For example, as shown in table 4.12, in row one is the docking pairing of Ligand-1 and Protein-1. This pairing shows that score 1 and score three have been classified as true and score two has been classified as false. The meta-score for this pairing has been classified as a potential interaction (true). As the meta-score is true, to work out the overall score for that pairing only the score that had a combinational result of true will be combined. In this case score 1 (-2.01) and score 3 (-3.02) will be combined which equals an overall score of -5.03. For the second row of data in table 4.12, the docking pair of Ligand-1 and Protein-2, the meta-score has been classified as a non-interaction (false). The difference the data in row 1 and the data in row 2 are that the meta-score is classified as false. This classification determines that only scores that have a combinational forecasting result of false will be combined. In this case score, 1 and two will be combined which gives an overall score of 0.08. With row 3 all of the scoring functions have been predicted as true which produces the overall meta score as true. In turn this means that all of the scoring function 2SD score are added together to generate the strength score. This also goes for row 4 as the overall meta score is equal to false which then means that all of the scoring functions are added together to generate the overall strength score.

With this heuristic, the lower the overall score is, the "stronger" the result. The heuristic will help distinguish the non-interactions from the possible interactions where there should be a clear score difference and provide non-interactions with a lower score than those classified as a possible interaction. In the case of the example, it can be seen that the docking pair of ligand-1 and protein-1 is a much "stronger" interaction than ligand-1 and protein-2.

4.7.4.2 Conformation analysis

Shown in Table 4.13 are known targets of the top 20 most valuable sold drugs that are within the panel 44 set. These protein-drug interactions were analysed to determine how well the combinational forecasting is performing and if so, are the predicted conformations matching the result given by the combinational forecasting. The panel 44 set had been selected for this analysis as these are classified as the industry standard in the pharmaceutical industry. The purpose of this conformation analysis is to determine if the scores predicted correlate with biologically viable conformations.

True Positive Predictions

Of the 11 predicted dockings for the panel 44 set with known interactions, all interactions were classified as potential interactions. Shown in table 4.18 are all of the strength scores calculated for all of the panel 44 known targets within the protein threaded set. All of the conformations for the known interactions were analysed to determine if the dockings were biologically viable. Discussed below are 3 examples of the analysis that had taken place.

Table 4.13: Panel 44 and known targets within the protein threaded set – shown in the table are all of the known targets for the panel 44 that are within the protein threaded set with the strength score calculated using the developed heuristic.

Protein	Uniprot	Ligand	PubChem	Strength Score
Prostaglandin G/H synthase 2	P35354	Lenalidomide	216326	0.09
5-hydroxytryptamine receptor 2A	P28223	Aripiprazole	60795	-6.65
5-hydroxytryptamine receptor 1A	P08908	Aripiprazole	60795	-5.07
D(1A) dopamine receptor	P21728	Aripiprazole	60795	-3.08
D(2) dopamine receptor	P14416	Aripiprazole	60795	-2.74
Muscarinic acetylcholine receptor M3	P20309	Solifenacin	154059	-2.31
Muscarinic acetylcholine receptor M1	P11229	Solifenacin	154059	-2.58
Histamine H1 receptor	P35367	Aripiprazole	60795	-6.54
Muscarinic acetylcholine receptor M2	P08172	Aripiprazole	60795	-3.02
Muscarinic acetylcholine receptor M2	P08172	Solifenacin	154059	-2.27
Muscarinic acetylcholine receptor M2	P08172	Tiotropium	5487427	4.88
Muscarinic acetylcholine receptor M3	P20309	Tiotropium	5487427	5.57
Muscarinic acetylcholine receptor M1	P11229	Tiotropium	5487427	3.34

Prostaglandin G/H synthase 2 and Lenalidomide

The combinational forecasting classified this docking pair as a potential interaction. Prostaglandin G/H synthase 2 is a known target of Lenalidomide which has been predicted correctly. Using the created heuristic to determine the strength of the interaction, the calculated result of 0.09, which was the lowest strength score generated of the whole panel 44 threaded set. The strength score does seem to indicate that there is no interaction, though it must be stated that the strength score is used to help rank the conformations from strongest to weakest and so does not necessarily mean that no prediction is predicted. Information that was stored on the UniProt database which gave known active site residues for Prostaglandin G/H synthase 2, these being atom 106 (ARG), 193 (HIS), 341 (TYR), 371 (TYR), and 516 (SER). Shown in figure 4.21 is the top calculated conformation for this docking pair. Lenalidomide (magenta) has docked near to the known active residue 93 (HIS). Figure 4.22 shows a closeup of the active atom 193 where Lenalidomide has docked 4.919 angstroms away. This result proves that the combinational forecasting has predicted this conformation correctly. This result also provides some explanation as to why the “strength” score was weak as Lenalidomide is not close enough to have the potential for interaction.

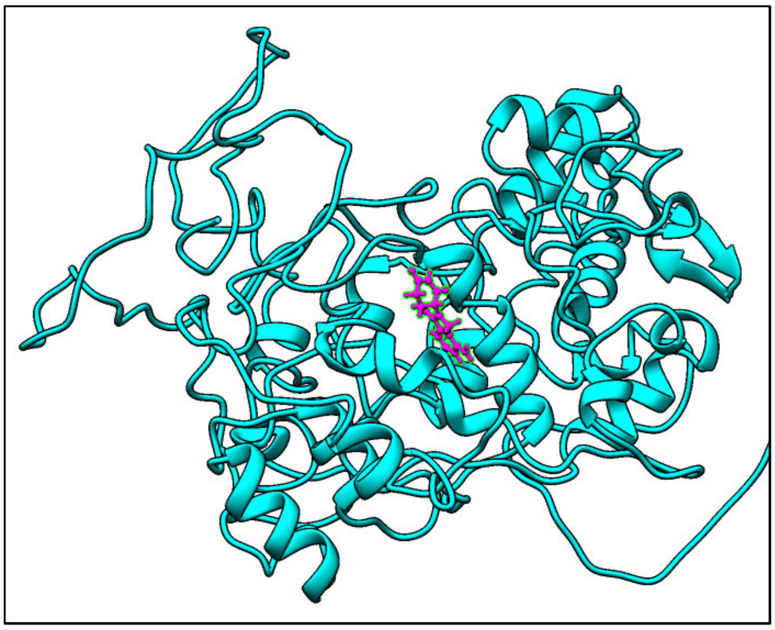


Figure 4.19 : Prostaglandin G/H synthase 2 and Lenalidomide docking - Image of the top conformation for the Prostaglandin G/H synthase 2 and Lenalidomide docking.

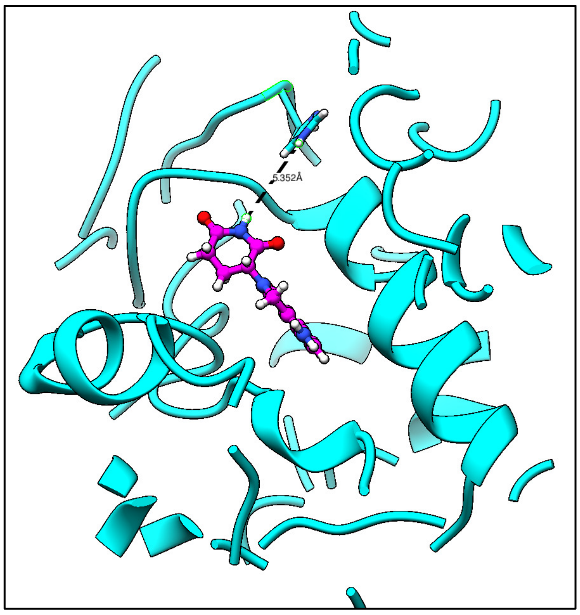


Figure 4.20:close up image of the Prostaglandin G/H synthase 2 and Lenalidomide docking - Image Lenalidomide docking near the known active atom 193 (HIS) of Prostaglandin G/H synthase 2

5-hydroxytryptamine receptor 2A and Aripiprazole

The combinational forecasting classified this docking pair as a potential interaction. 5-hydroxytryptamine receptor 2A is a known target of Aripiprazole which has been predicted correctly. Using the created heuristic, the “strength” score for the predicted interaction was -6.65, which indicates that this is a strong interaction. This strength score was the highest score of all the panel 44-drug pairings shown in table 4.18. Shown in figure 4.23 is the top conformation predicted for 5-

hydroxytryptamine receptor 2A and Aripiprazole (magenta). The UniProt database had stored information about known active residues for 5-hydroxytryptamine receptor 2A this atom being 229 (LEU). Shown in figure 4.24 is a close up of the known active atom 229 and it shows that Aripiprazole has docked near. When the distance between the two was measured, it showed that Aripiprazole had docked 5.352 angstroms away from the leucine residue. The distance is close enough to bring

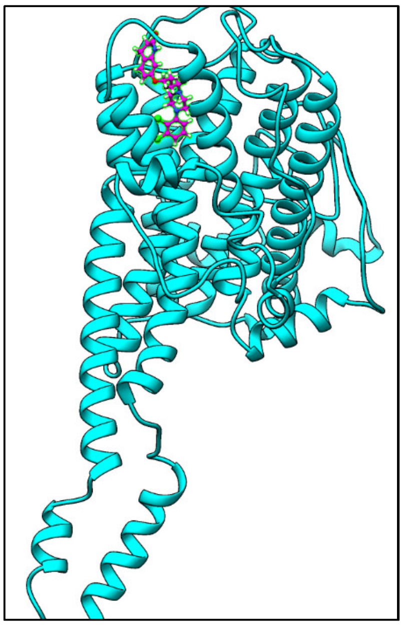


Figure 4.23: 5-hydroxytryptamine receptor 2A/ Aripiprazole docking – shown is the top conformation generated for the 5-hydroxytryptamine receptor 2A/ Aripiprazole docking

about an interaction and justifies the strong "strength" score produced by the created heuristic.

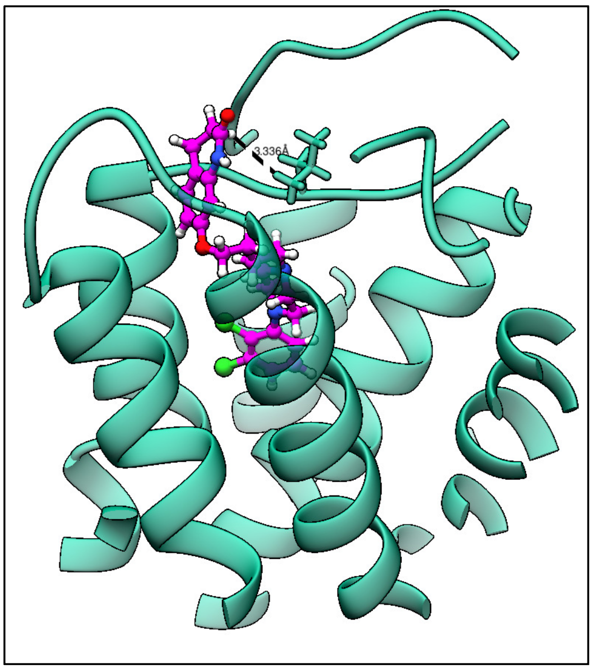


Figure 4.24: close up image of 5-hydroxytryptamine receptor 2A/ Aripiprazole docking – shown is a close up image of the 5-hydroxytryptamine receptor 2A/ Aripiprazole docking which is 3.336 angstroms away from a known active residue 229 (LEU).

5-hydroxytryptamine receptor 1A and Aripiprazole

The combinational forecasting classified the docking pair as a potential interaction. 5-hydroxytryptamine receptor 1A is also a known target for Aripiprazole which had been predicted correctly. The created heuristic gave a “strength” score of -5.07, which indicates a strong interaction. Shown in figure 4.23 is the top conformation predicted for the 5-hydroxytryptamine receptor 1A and Aripiprazole (magenta) docking pair. There was no known data about known active residues for 5-hydroxytryptamine receptor 1A on the UniProt database to help determine if Aripiprazole had docked in the correct location. However, there was a known crystal structure for a close relative 5-

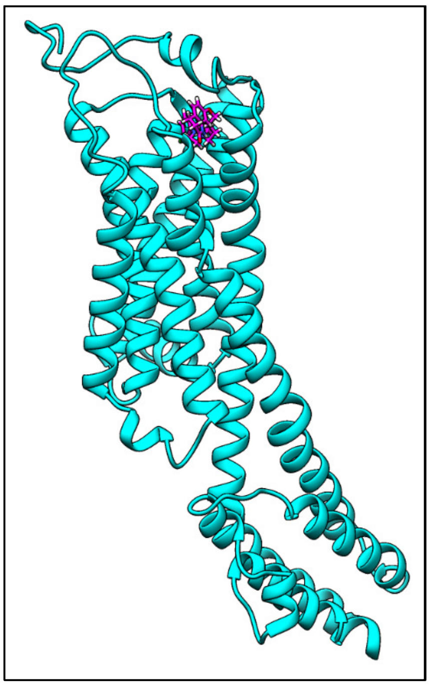


Figure 4.21: 5-hydroxytryptamine receptor 1A and Aripiprazole docking - Image of the top generated conformation for the 5-hydroxytryptamine receptor 1A and Aripiprazole docking pair.

hydroxytryptamine receptor 2B (gene HTR2B). The crystal structure 5TUD (Ishchenko et al., 2017) has the drug ERM bound. The RMSD score between 5-hydroxytryptamine receptor 1A threaded model and the 5TUD crystal structure was 1.947. Shown in figure 4.24 is an image of the crystal structure 5TUD docked ligand ERM (red) superimposed onto the generated model for 5-hydroxytryptamine receptor 1A with both Aripiprazole (magenta) bound. Both drugs have docked in the same location, which indicates that the predicted docking is biologically plausible. This outcome has proven that both the combinational forecasting and the “Strength” score had predicted correctly.

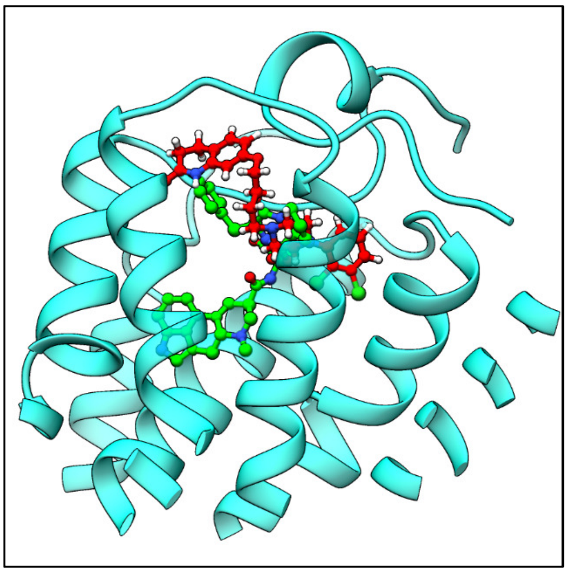


Figure 4.22: Close up image of top conformation 5-hydroxytryptamine receptor 2A/ Aripiprazole docking superimposed with 5TUD crystal structure docking.

True Negative Predictions

Due to none of the panel 44 protein having *in vitro* data that indicates that they are a known non-interaction with any of the top 20 most widely prescribed drugs. Randomly selected docking pairs involving panel 44 proteins are used to determine if the combinational forecasting and "strength" score have predicted correctly. Shown in Table 4.14 are the 3 predicted interactions that are analysed.

Table 4.14:Examples of True Negative within the protein threaded set – shown are 3 known non-interactions for the top 20 ligands docked against the protein threaded set from In vitro data which have been predicted correctly using the developed combinational forecasting.

Protein	Uniprot	Ligand	PubChem	Strength Score
D(2) dopamine receptor	P14416	Pregabalin	5486971	4.08
D(1A) dopamine receptor	P21728	Gabapentin	3446	3.172
Adenosine receptor A2a	P29274	Levothyroxine	5819	8.43

D(2) dopamine receptor and Pregabalin

is no *in vitro* data to suggest that the docking pair of D(2) dopamine receptor and Pregabalin should have an interaction. The combinational forecasting predicted the docking pair as a non-interaction and an interaction "strength" score of 4.08, which indicates a very poor interaction. UniProt contains information about known active residues for D(2) dopamine receptor these being residues 114 (ASP), 194 (SER), and 197 (SER). Shown in figure 4.25 is the top predicted conformation for the docking pairing of D(2) dopamine receptor and Pregabalin (magenta) and the known active residues highlighted (red). From this single image, it shows that this is a non-interaction. Initially, the first indication that this conformation is a non-interact is the fact that Pregabalin has not docked near any of the known active residues of the D(2) dopamine receptor. The second indication is that pregabalin has docked in a location that biologically would not be accessible due to it being blocked off by a span domain of lipids. The area of the protein that pregabalin has docked is known as the protein-binding domain. The outcome indicates that the combinational forecasting has correctly predicted that this docking pair is a non-interaction and the “strength” score was also correct.

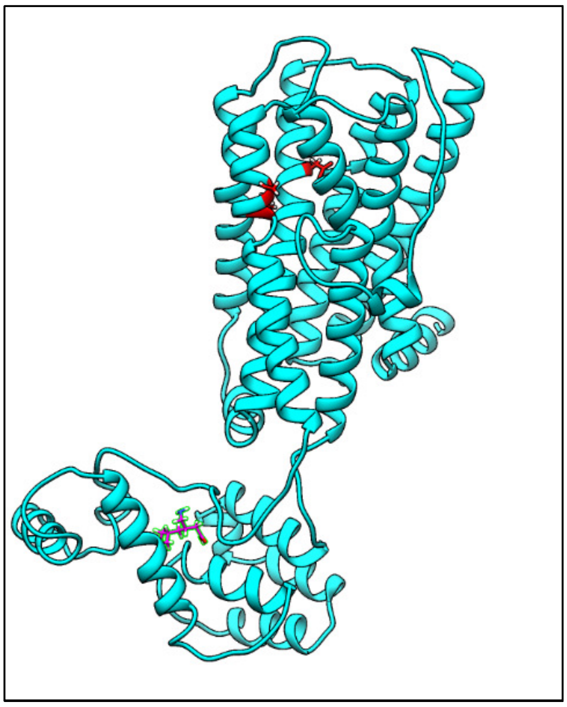


Figure 4.23 : D(2) dopamine receptor and Pregabalin docking - Image of the top predicted conformation for the D(2) dopamine receptor and Pregabalin docking pair

D(1A) dopamine receptor and Gabapentin

There is no in vitro data to suggest that the docking pair of D(1A) dopamine receptor and Gabapentin should have an interaction. The combinational forecasting predicted the docking pair as a non-interaction and the “strength” score given was 3.172, suggesting a very weak interaction. There is no known active residues information stored on UniProt about D(1A) dopamine receptor, but there is a known crystal structure for a close relative D(4) dopamine receptor. The crystal structure 5WIV (Wang et al., 2017) has an RMSD of 1.957 when matched with the D(1A) dopamine receptor generated model. Shown in figure 4.26 is an image of the top-ranked conformation for the D(1A) dopamine receptor and Gabapentin (magenta) docking pair. It seems that Gabapentin has been docked near an area of low homologue coverage which is caused by low template coverage. Shown in figure 4.27 is an image of the crystal structure 5WIV (dark blue) docked ligand Nemonapride (AQD)(green) superimposed onto the generated model for D(1A) dopamine receptor (Gold) with both Gabapentin (magenta) bound. From the image, it displays that Gabapentin has docked in a different location to the crystallised docking of Nemonapride. With both findings, it can be determined that this is a non-interaction and that the combinational forecasting had predicted the conformation correctly. The “strength” score also gave a valid score of 3.172.

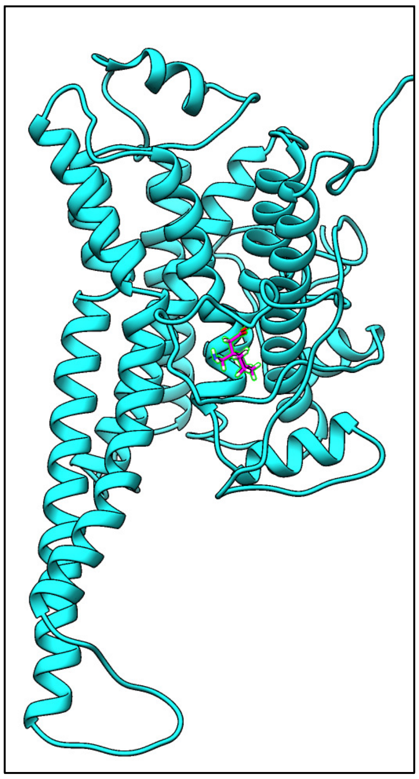


Figure 4.24 : D(1A) dopamine receptor and Gabapentin docking - Image of the top-ranked conformation for the D(1A) dopamine receptor and Gabapentin docking pair.

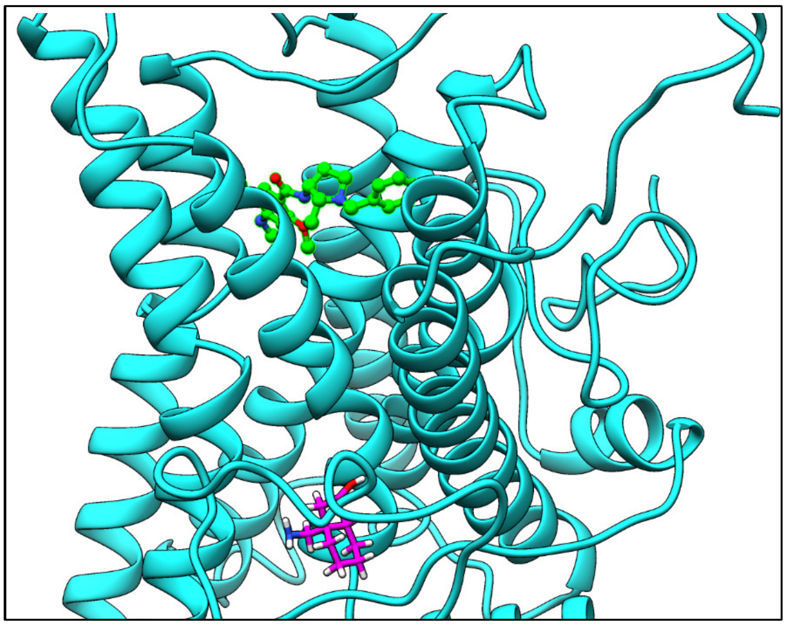


Figure 4.25 :close up image of the crystal structure 5WIV bound ligand Nemonapride (AQD)(green) – shown is a superposition of the crystal structure 5WIV’s bound ligand AQD onto the generated model for D(1A) dopamine receptor with Gabapentin (magenta) bound locations.

Adenosine receptor A2a and Levothyroxine

No in vitro data could be obtained to suggest that the docking pair of Adenosine receptor A2a and Levothyroxine should have an interaction. The combinational forecasting predicted the docking pair as a non-interaction and gave a “Strength” score of 8.43 which also ranks the interaction as weak. There is no known active atom information stored on the UniProt database to analyse if Levothyroxine has docked near them. However, there is a known crystal structure for Adenosine receptor A2a, which has the drug Adenosine bound. The crystal structure 2YDO (Lebon et al., 2011) has an RMSD of 1.998 when compared to the generated model for Adenosine receptor A2a. Shown in figure 4.28 is an image of the top-ranked conformation predicted for the Adenosine receptor A2a and Levothyroxine (magenta) docking pair. It's clearly shown that Levothyroxine has docked on the outside of Adenosine receptor A2a. From this image, it is already clear that the prediction suggests that this is a non-interaction. Shown in figure 4.29 is an image of the crystal structure 2YDO’s docked ligand Adenosine (green) superimposed with the generated model of Adenosine receptor A2a with Levothyroxine (magenta) bound. From this image, it can be seen that Levothyroxine has not bound in the correct place which justifies the combinational forecasting and “strength” score predicted results for this docking pair.

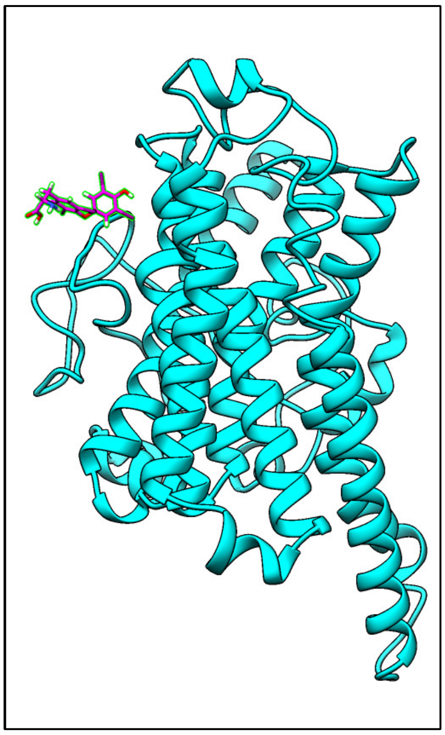


Figure 4.26: Adenosine receptor A2a and Levothyroxine docking - shown is an image of the top-ranked conformation predicted for the Adenosine receptor A2a and Levothyroxine docking pair.

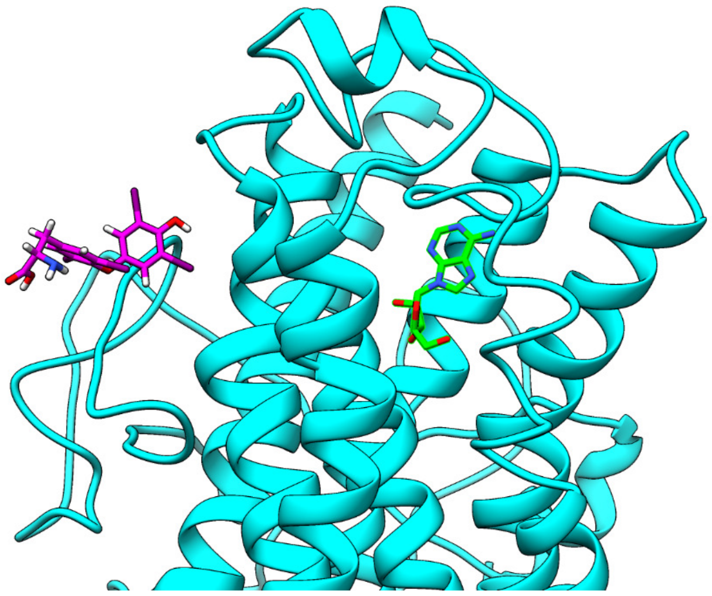


Figure 4.27: Image of the crystal structure 2YDO docked ligand Adenosine (green) superimposed onto the generated model of Adenosine receptor A2a with Levothyroxine (magenta) bound.

4.7.4.2.1 False Positive predictions

There were false positives that had been predicted by the combinational forecasting method. Some conformations that the combinational forecasting had predicted as a possible interaction even though the ligand had bound in locations that for some were not biologically plausible. Here is the analysis of 3 of the false positive that had been predicted. The analysed predicted dockings are shown in table 4.15.

Table 4.15: Examples of False Positive within the protein threaded set – shown are 3 known non-interactions for the top 20 ligands docked against the protein threaded set from In vitro data which have been predicted correctly using the developed combinational forecasting.

Protein	Uniprot	Ligand	PubChem	Strength score
D(2) dopamine receptor	P14416	Tadalafil	110635	-5.365
Adenosine receptor A2a	P29274	Tazobactam	123630	-0.642
D(2) dopamine receptor	P14416	Budesonide	5281004	-1.457

D(2) dopamine receptor and Tadalafil

There are no *in vitro* data that indicates that Tadalafil should have an interaction or should not have an interaction with D(2) dopamine receptor. However, the combinational forecasting had predicted the docking pair as a potential interaction and a “strength” score of -5.365. UniProt’s database had stored data about known active residues for D(2) dopamine receptor, these being residues 114 (ASP), 194 (SER), and 197 (SER). Shown in figure 4.30 is an image of the top-ranked conformations generated for the D(2) dopamine receptor and Tadalafil (magenta) docking pair. The known active residues are highlighted in orange and shown that Tadalafil has not bound near any of them, which indicates that this is a non-interaction. The drug has found a flexible protein-protein interaction domain. The domain is largely hydrophobic, and the molecule is very nonpolar overall which could be the reason that the drug has been attached to that area. In real life, that region would not be exposed to ligands as it is in the intracellular domain. The part of the protein that is inside of the cell, the ligand interactions would happen at the other end. AutoDock Vina & DOCK 6.0 does not know this fact, and it could be that Tadalafil would bind to the G-protein domain area if the protein binding domain was not accessible. After this analysis, it concluded that this was a false positive.

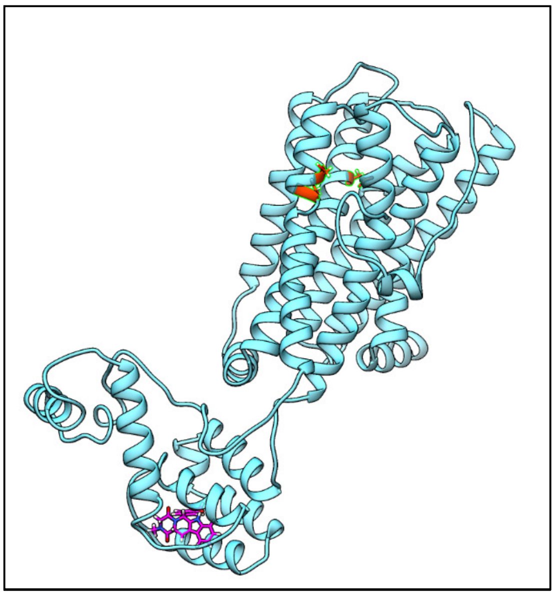


Figure 4.28: D(2) dopamine receptor and Tadalafil docking - Image of the top-ranked conformations generated for the D(2) dopamine receptor and Tadalafil docking pair.

Adenosine receptor A2a and Tazobactam

There are no *in vitro* data that indicates that Tazobactam has a known interaction with Adenosine receptor A2a. However, the combinational forecasting has classified the docking pair as a potential interaction. The result is then causing the “strength” score to be -0.642, which is a weak interaction. The crystal structure 2YDO was used to analyse the conformation and the location of Tazobactam docking. Shown in figure 4.31 is an image of the top-ranked conformation for the Adenosine receptor A2a and Tazobactam docking pair. Tazobactam appears to have bound to the outside of the protein and not within. This is due to a modelling error. The G-protein domain has modelled with low structural content to the side of the protein, where the cell wall would be in vivo. This has made a "fake" pocket which would not exist in life, and it's that pocket which has captured the drug in the simulation. This discovery has indicated that this would be a non-interaction. Figure 4.32 shows an image of the crystal structure 2YDO bound ligand Adenosine (green) bound superimposed onto the generated model for Adenosine receptor A2a with Tazobactam (magenta) and. Figure 4.32 clearly shows that Adenosine (green) has bound within the cavity of Adenosine receptor A2a which is the active site. This outcome gives another indication that the docking pair of Adenosine receptor A2a and Tazobactam should have been classified as a non-interaction.

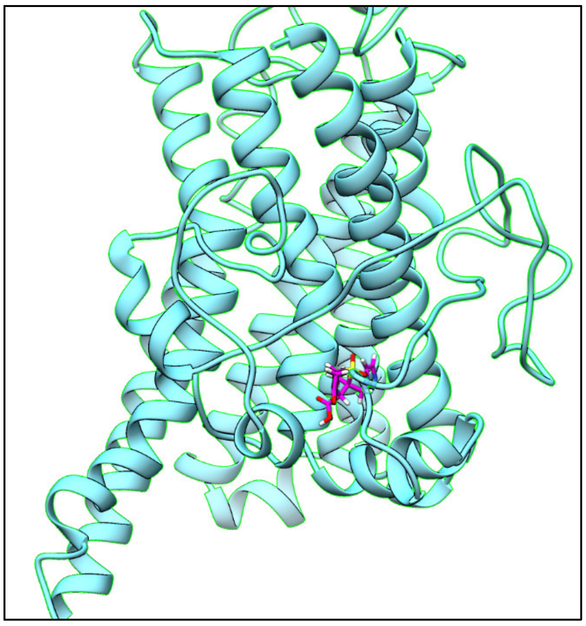


Figure 4.29 : Adenosine receptor A2a and Tazobactam docking – Shown is an image of the top-ranked conformation for the Adenosine receptor A2a and Tazobactam docking pair.

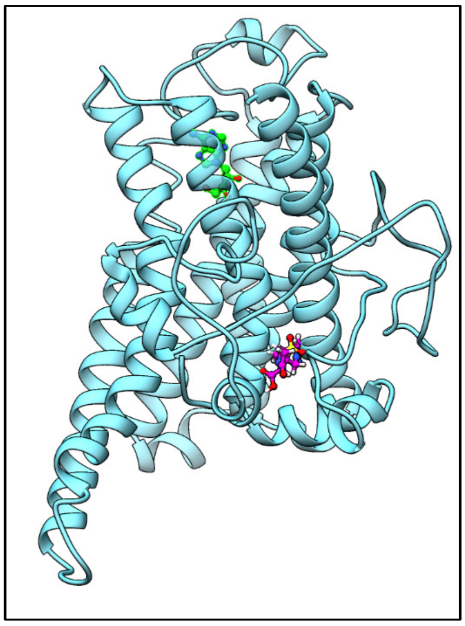


Figure 4.30: Image of the crystal structure 2YDO (purple) overlapping the generated model for Adenosine receptor A2a (gold) with both drugs Tazobactam (light blue) and Adenosine (green) bound.

D(2) dopamine receptor and Budesonide

There are no *in vitro* data that indicates that Budesonide should have an interaction with the D(2) dopamine receptor. The combinational forecasting predicted the docking pair as a potential interaction which also led to the “strength” score of -1.457, which indicates a good interaction. D(2) dopamine receptor has got known active residues these being atom 114 (ASP), 195 (SER), and 197 (SER). Shown in figure 4.33 is an image of the top-ranked conformation for the D(2) dopamine receptor and Budesonide docking pair. Budesonide has docked far away from the known active residues (orange) which indicates that this is a non-interaction. It seems like the docking programs had found a good shape-matching ("lock and key") between an area of the protein that is hydrophilic and the similarly hydrophilic hydroxyl groups on those phenol rings. This region would be blocked off by lipids in the cell wall in life, so the drugs would never be bound within that area *in vivo*.

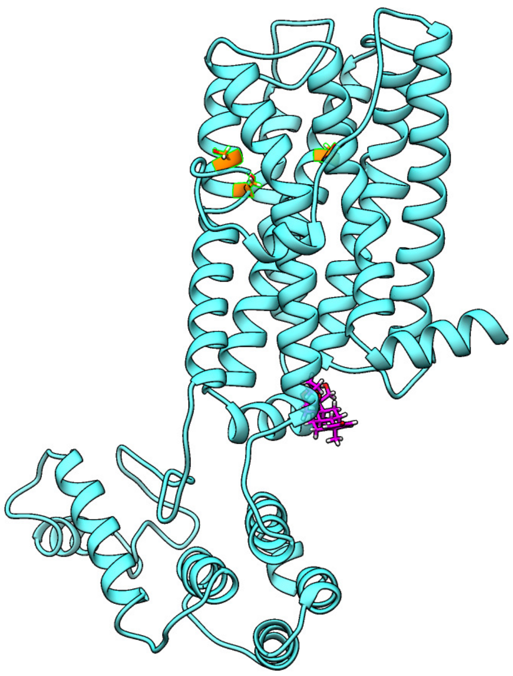


Figure 4.31: D(2) dopamine receptor and Budesonide docking – Shown is an image of the top-ranked conformation for the D(2) dopamine receptor and Budesonide docking pair.

4.7.4.2.2 False Negative predictions

These predictions are when a drug that is known to have an interaction with a protein, yet the combinational forecasting does not classify as a potential interaction. Of the 33 known protein interactions within the set of high-quality models which had undergone protein threading, only three were classified as a non-interaction. Shown in table 4.16 are three protein-ligand pairings that were classified as non-interactions by the combinational forecasting with their calculated strength score. These three dockings were investigated to determine the cause of why they were classified as non-interactions by the combinational forecasting.

Table 4.16:False Negative results – the following table shows the three protein-ligand pairings that the combinational forecasting classified as a non-interactions instead of being classified as a potential interaction. These have been shown with their calculated strength scores.

Protein	Uniprot	Ligand	PubChem	Strength Score
Synaptic vesicle glycoprotein 2A	Q7L0J3	Levetiracetam	441341	3.15
Adenosine receptor A1	P30542	Gabapentin	3446	3.39
Thyroid hormone receptor beta	P10828	Levothyroxine	5819	5.1

Synaptic vesicle glycoprotein 2A and Levetiracetam

There is *in vitro* data that has confirmed that Synaptic vesicle glycoprotein 2A is a known target of Levetiracetam (Lynch et al., 2004). However, the combinational forecasting has classified the docking pair as a non-interaction. When investigating the conformation, it was discovered that the generated model for Synaptic vesicle glycoprotein 2A had many areas of low homologue coverage which was surprising as it was part of the set of high-quality models which had undergone protein modelling. Figure 4.34 shows an image of the top-ranked conformation for the Synaptic vesicle glycoprotein 2A and Levetiracetam (magenta) docking pairing. Levetiracetam has bound to the outside of the protein, which does indicate that this is a non-interaction. This outcome could be as the areas of low-homologue coverage could be blocking entrances into the protein which otherwise would be accessible. This is a false negative result, but from the predicted conformation, the combinational forecasting has predicted correctly that this is a non-interaction. This result was not caused due to

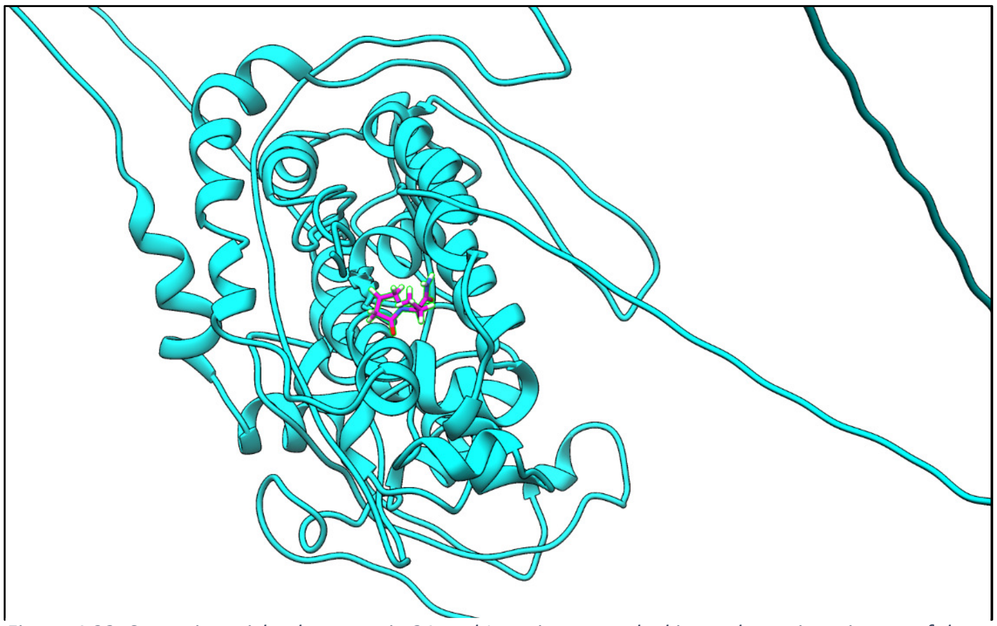


Figure 4.32: Synaptic vesicle glycoprotein 2A and Levetiracetam docking – shown is an image of the top ranked conformation for the Synaptic vesicle glycoprotein 2A and Levetiracetam docking pairing.

the combinational forecasting but by the generated threaded model of Synaptic vesicle glycoprotein 2A.

Adenosine receptor A1 and Gabapentin

There is *in vitro* data that has proven that Adenosine receptor A1 is known to interact with Gabapentin (Zuchora, Wielosz and Urbańska, 2005). However, the combinational forecasting has classified the docking pair as a non-interaction. Shown in figure 4.35 is an image of the top-ranked conformation for the Adenosine receptor A1 and Gabapentin (magenta) docking pair. Gabapentin is shown to have docked on the top of Adenosine receptor A1 on the outside. As this seemed like an excellent generated protein model, it was not clear why Gabapentin did not dock within the protein. After investigating the model further, it was clear to see the reason why the docking programs had bound Gabapentin to that location. Shown in figure 4.36 is an image of the conformation from above which shows that there is an area of low-homologue coverage which is blocking the entrance into the protein. The area of low homologue coverage is highlighted in orange. Based on the location that Gabapentin has bound, the combinational forecasting correctly classified the docking. The cause of the erroneous result was a weakness in the generated protein model.

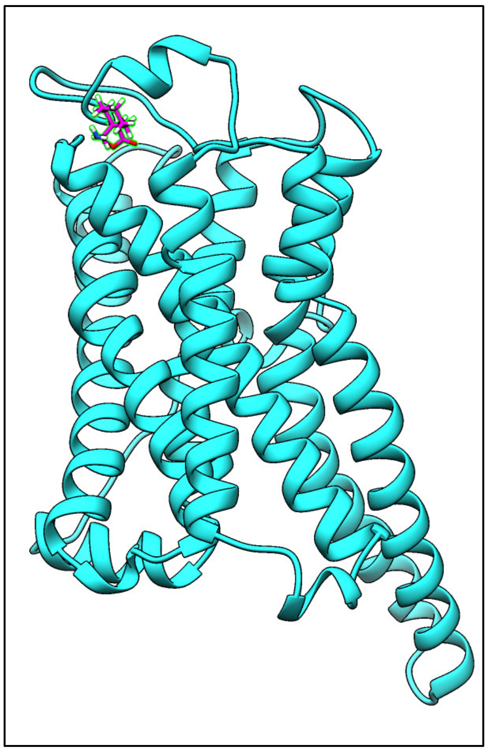


Figure 4.33: Adenosine receptor A1 and Gabapentin docking -shown is an image of the top-ranked conformation for the Adenosine receptor A1 and Gabapentin (magenta) docking pair

.

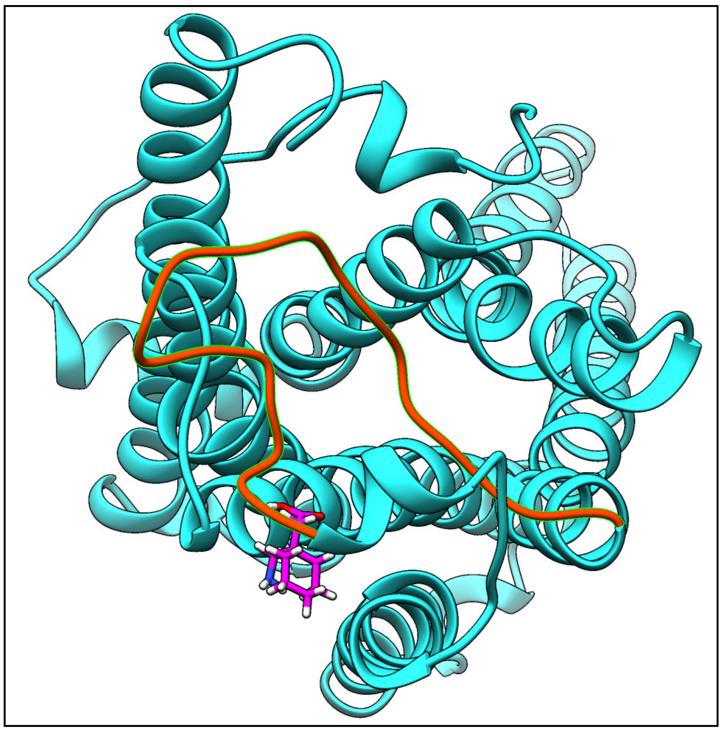


Figure 4.34:Low-homologue modelled area of Adenosine receptor A1 – shown is an image of the conformation from above which shows that there is an area of low-homologue coverage which is blocking the entrance into the protein. The area of low homologue coverage is highlighted in orange.

Thyroid hormone receptor beta and Levothyroxine

There is much *in vitro* data that indicates that the Thyroid hormone receptor has an interaction with the synthetic thyroid hormone drug Levothyroxine (DrugBank). However, the combinational forecasting has classified the docking pair as a non-interaction. This outcome led to investigating the predicted conformation. Shown in figure 4.37 is an image of the top-ranked conformation for the Thyroid hormone receptor beta and Levothyroxine docking pair. The Thyroid hormone receptor has had a high-quality model generated as this is a very well characterised protein with many crystal structures like 1XZX (Sandler et al., 2004) and 1Y0X (Sandler et al., 2004). Levothyroxine (magenta) has ended up being docked on the outside of the protein. the only conclusion is that this was caused by a docking failure due to possible errors caused within the process.

4.7.4.2.3 Conclusion of the conformation analysis

After analysing the conformations to evaluate the performance of the combinational forecasting method that was developed, the outcome was very promising. The combinational forecasting had clearly performed well with the true positive dockings that had been classified correctly as potential interactions. All 11-known interaction with proteins within the panel 44 set were classified correctly. Three of the 11-known interaction from proteins within the panel 44 set were investigated to determine if the conformations were biologically plausible to be a potential interaction. Each of the

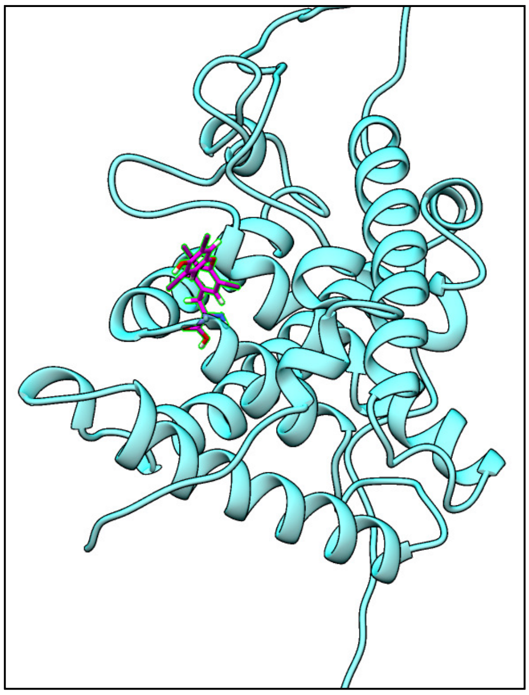


Figure 4.35: Thyroid hormone receptor beta and Levothyroxine docking – shown is an image of the top ranked conformation for the Thyroid hormone receptor beta and Levothyroxine docking pair.

investigated docking pairs concluded that the drugs were bound correctly near either know active residues or in a similar location to known crystal structures. The true negatives conformation results had shown good performance for the combinational forecasting developed method. A sample of 3 proteins from panel 44 was selected that were classified as a non-interaction with one of the top 20 most widely prescribed drugs. All 3 dockings that were investigated provided conformations that would be determined as a non-interaction due to the locations that the drugs had docked. Two of the drugs had docked into a protein-binding domain which is not biologically plausible as this would be blocked by lipids. Another had bound outside of the protein. The false positives conformations results showed that there is a combination of protein modelling and protein-ligand docking errors that have caused the non-interactions to be classified incorrectly by the combinational forecasting. For two of the three false-positive examples that were investigated, the docking programs had docked the proteins in areas that would not be accessible. The docking programs do not have the knowledge of protein domain areas of proteins or areas that would not be accessible, which is the reason they have been docked in those positions. For the other docking pair, the issue was that the model had an area of the protein with low-homologue coverage which causes a "fake" pocket to be created. The false-negative conformation results that were examined were 3 known targets of the top 20 most widely prescribed drugs that were classified as non-interactions by the combinational forecasting. When the conformations were examined, it was clear to see that the generated threading models for 2 proteins had areas of low-homologue coverage that either blocked entrances that generally drugs would enter which caused for all three cases for the drug to bind to the outside of the protein. The third docking pairing of Thyroid hormone receptor beta and Levothyroxine was due to a docking error which had caused Levothyroxine to dock on the outside of the protein. The combinational forecasting had predicted correctly from the scoring function results given as these were non-interactions based on the conformation predicted.

Overall the combinational forecasting method has performed exceptionally well. Considering that no *a priori* analysis of binding regions of proteins had been undertaken , to get this level of accuracy is promising. The combinational forecasting had predicted all the analysed examples correctly from only the scores that were produced by the docking programs with no interaction with the conformation themselves. There is potential for this method to be improved with the addition of conformation analysis prior to loading data through the combinational forecasting method.

4.8 Conclusion

The final accuracy produced by the combinational forecasting method developed has been very promising for predicting known targets from a vast field of docking data. The study provided many valuable results in the process of developing more accurate predictive methods. Firstly, from the raw homology modelled baseline structural data without any proccessing it was shown that the identification of known targets of the ligands from a large set of virtual dockings was less than 50% accurate. This would not give sufficiently useful results to be beneficial for the users of the system for lead drug discovery or repurposing. Reasons for this were discovered such as low homologue coverage of some protein models and incorrect ligand binding sites predicted by AutoDock Vina and DOCK6. This led to the adoption of the Shipyard 1.5 pipeline which calculated a total of 19 different scoring functions for each protein-ligand conformation also with the added stage of DoGSite which calculates possible pockets within a protein that could be favourable for docking a ligand. With the use of this docking pipeline, the development of the database was improved so that it was able to store the wide array of data being generated. The protein sequences were also re-modelled using an updated PDB template database (2017) which generated 3,233 more high homology covered protein models. With the extra scoring function and improved model generation, the possibility of improved accuracy of dockings being generated would be expected to improve compared to unprocessed data. This was the case but not an improvement that would substantially increase the value of the data. With the outcome of the analysis, the data for the known misses for the top 20 drugs were analysed to evaluate that the analysis method was predicting them correctly as non-interactions. Of the known misses within the new homology modelled set, 66.6% were predicted correctly, but the overall accuracy of correct prediction of interactions and non-interactions was still low, leading to many possible targets/off-target interactions being missed.

A protein threading modelling approach that had an overlap of FDA known target proteins with the homology modelling set was used to determine if the model quality was affecting docking results. The protein threading provided 413 more high-quality proteins compared to the re-modelled homology modelling set (2017). The protein threaded models were run using the same Shipyard 1.5 pipeline as for the homology modelling set. The results outcome was not significantly different to the new homology modelling set which led to the conclusion that a more thorough analysis was needed of the scoring functions used in order to improve the accuracy of the predicted outcomes with the aim of reaching as near to 100% as possible. A new approach using combinational forecasting and analysis of the scoring function using ROC curves was introduced. The ROC curve identified the most accurate scoring functions for the homology modelling set and the protein threading set. Each of the set showed only 6 of the 19 scoring functions were producing 70% or more accuracy for predicting known targets and known non-interactions correctly. The results showed that there were 4 scoring functions that were consistently accurate in both models sets, these being the GBSA Affinity, Vina Gauss1, Vina Gauss 2, and DSX Contact count scoring functions. Threshold scores were calculated for each of the scoring function which would produce the highest accuracy possible for the predictions. A threshold then had to be calculated for the combinational forecasting which would produce the most accurate results possible which led to the outcome of 4 of the 6 identified scoring functions needing to pass a threshold for a protein-ligand pairing to be classified as a potential interaction. The accuracy produced for correctly predicting known targets and known non-interactions increased dramatically compared to the raw baseline results at the beginning. The combinational forecasting used for the homology modelling set produced an overall accuracy of 66.45%. The combinational forecasting used for the protein threaded set produced a better result of 72.05%. Both sets produced a 90%+ accuracy for predicting known targets correctly, the only difference being that the combinational forecasting set had a higher accuracy of predicting known non-interactions correctly with 53.2% compared with homology sets 42.4%. To verify that conformations generated were not producing false scores, a conformation analysis was undertaken on the panel 44 proteins and on known non-interactions. Shown in table 4.17 is a summary of the findings of the 12-conformation example shown in section 4.7.4.2.

This level of accuracy that has been produced from a large amount of docking data generated is a promising outcome and indicates that the approach can yield useful data that can lead to off-target interactions being identified or potential new targets due to strongly indicated interactions being identified. The combinational forecasting can be improved by adding other factors to the modelling and docking process, but this has produced a promising outcome which can lead to further development and the use of large-scale virtual screening in the future.

Table 4.17: the results of the investigation of the conformations – shown are the summary of results discovered in the conformation analysis discussed in section 4.7.4.2

Protein	Ligand	Combinational Forecasting Result	Combinational forecasting Correct/incorrect	Reason?
Prostaglandin G/H synthase 2	Lenalidomide	TRUE	Correct	Bound near active residue
5-hydroxytryptamine receptor 2A	Aripiprazole	TRUE	Correct	Bound near active residue
5-hydroxytryptamine receptor 1A	Aripiprazole	TRUE	Correct	Bound in the same location as crystal structure drug
D(2) dopamine receptor	Pregabalin	FALSE	Correct	Bound in protein binding domain
D(1A) dopamine receptor	Gabapentin	FALSE	Correct	Bound in incorrect area near an area of low homologue coverage
Adenosine receptor A2a	Levothyroxine	FALSE	Correct	Bound on outside of protein adjacent to area of low homologue coverage
D(2) dopamine receptor	Tadalafil	TRUE	Incorrect	Bound in the protein binding domain
Adenosine receptor A2a	Tazobactam	TRUE	Incorrect	Bound in an area of low homologue coverage which created a "fake" pocket
D(2) dopamine receptor	Budesonide	TRUE	Incorrect	Binding region inaccessible due to membrane lipid bilayer
Synaptic vesicle glycoprotein 2A	Levetiracetam	FALSE	Incorrect	Modelling error
Adenosine receptor A1	Gabapentin	FALSE	Incorrect	Weakness in the generated model
Thyroid hormone receptor beta	Levothyroxine	FALSE	Incorrect	Docking error

5 Graphical User Interface

This chapter describes the architecture, assembly and functionality of the Graphical User Interface (GUI).

5.1 Graphical User Interface Overview

The Objective for this Graphical User Interface (GUI) is to provide a simple and easy way for the users of the system to access and visualise critical data from the MySQL database. This GUI should be easy to read, intuitive for the user to use and not be complicated. The Graphical User Interface will be a web-based interface and will need to be able to be used to undertake the following tasks:

- Link to the MySQL database
- Visualise data within the Proteins, Ligand, and Docking tables
- Search within the Proteins, Ligand, and Dockings tables
- Visualise a 3D docking
- Display all result scores for each docking selected

Shown in figure 5.1 is a flowchart of the process for design and the coding needed to produce the final graphical user interface for the protein-ligand docking database. The first stage would be to create a

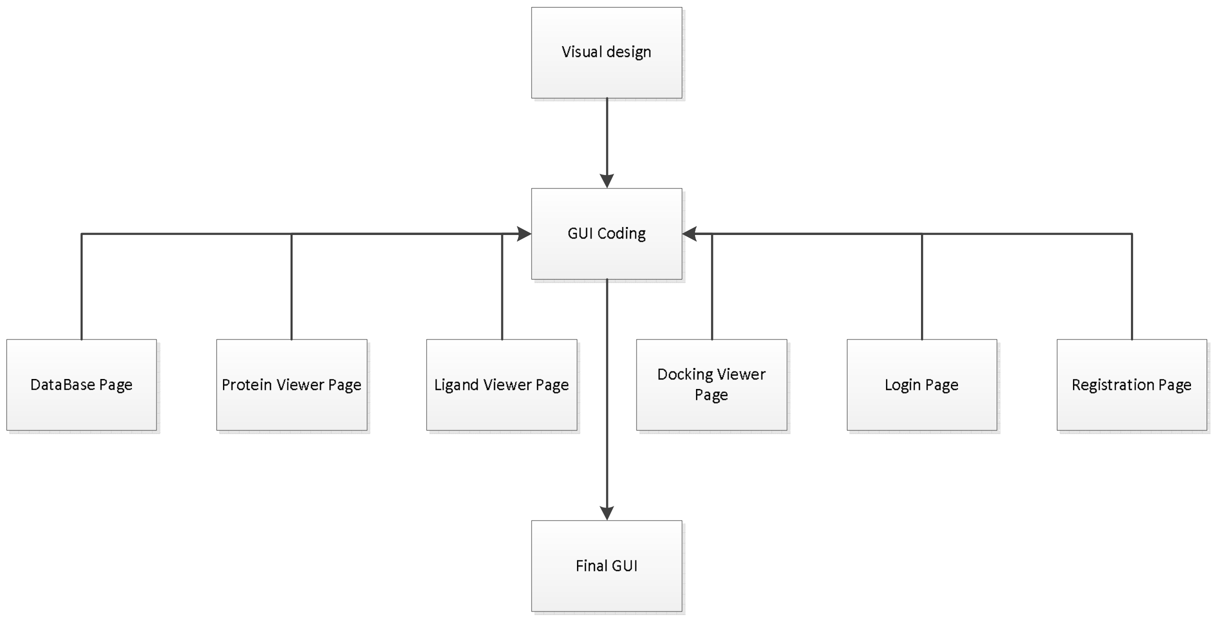


Figure 5.1:Flowchart for process of creating the Graphical User Interface (GUI) – shown is a flowchart of the process undertaken to produce a working graphical user interface for a user to view protein-ligand docking data within the database

basic design of the screen layout for each page to determine where certain items should be located. The basic design process is explained in section 5.2. and this acts as a guide to develop code that will produce the visual design and also create the backend code that will make it possible for the frontend to be used to be able to communicate and extract data from the MySQL database. This process is discussed further in section 5.3. After the coding has been completed, the final graphical user interface is then implemented which users will then be able to use and view protein-ligand docking data which is discussed in section 5.4 with details of what the user is able to do on the system.

5.2 GUI Design

The first stage of creating the Graphical user interface is the design - how the interface should look and what data needs to be available for the user to view and the method of accessing the vital data they need.

5.2.1 Visual Design

The first page the user will view is the login page. A login page is required to enable users to save lists within the system under a personal account name. The login page is a simple login panel where users can either register or login. The login panel as shown in figure 5.2 was placed in the middle of the page and have the standard username and password inputs with a login button and also a register button for users who do not have an account.

The data that the user would need to view is within the Proteins and Ligands tables. This is the first important stage for the user to select which proteins and ligand dockings they would like to view. This requires two tables to be displayed listing all the available Proteins and Ligands on the system. The user would also need to view the results that are available on the system in order to select the desired docking to view. This necessitates another table, to list the available dockings

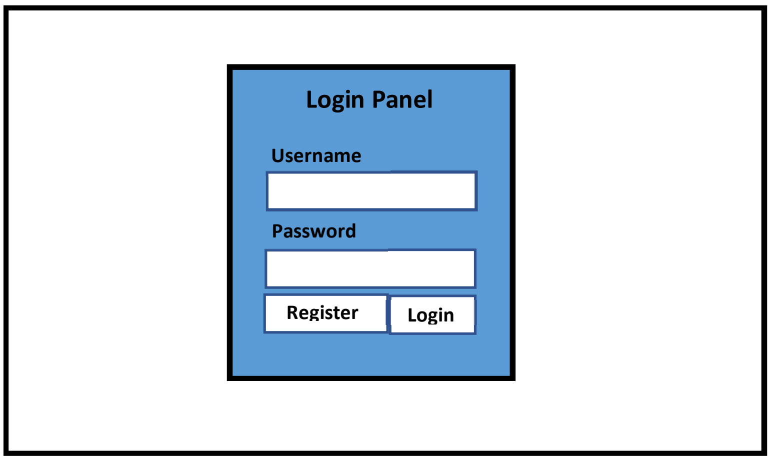


Figure 5.2: Basic design of Login Page – shown is a basic design of how the login page would look to the user

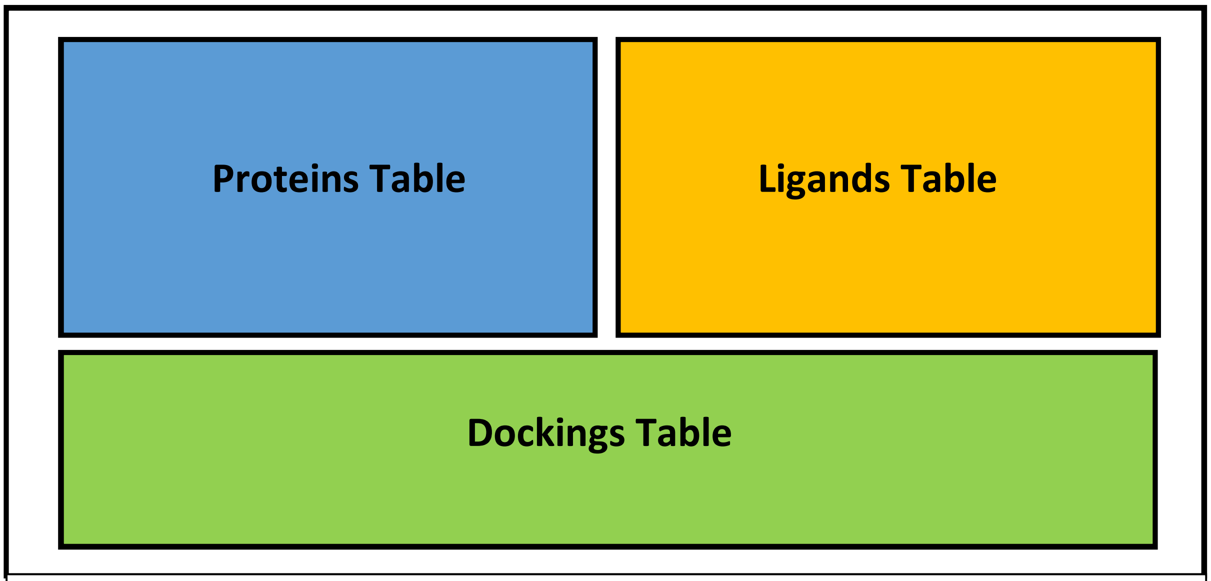


Figure 5.3: Basic design of the database page - shown is a basic design of how the database page would look to the user after login in to the system.

on the system. The basic layout for the design of the database web page is shown in figure 5.3. It allocates sections of the screen to each table. The design has been created to encourage the user to use the system in an intuitive manner when selecting the items required. The tables are placed in the order shown in figure 5.3, which gives the user the option to either select a protein or a ligand first as users may have a different preference for what they are investigating. This would make the results concise and easy to read for the user.

The method of the best way of displaying 3D results was the next important decision. The results required three separate pages to display detailed information about the protein, ligand, and docking selected by the user. The first page displayed specific protein data, the second for ligand-specific data and the third for displaying specific docking pair data. The design had to be user-friendly, displaying the data in a clear and simple format in each case. The design adopted for the protein data page is shown in figure 5.4. There are three panels within the page design. The Proteins Data table contains all the relevant data for the protein selected, such as Protein name, Gene name, sequence, modelling type etc. These were displayed in a vertical table with column name along the y-axis. To the right side of the protein, the data table is the 3D viewer. The 3D viewer shows the structure of the selected proteins obtained by either homology modelling or protein threading. This enables the user to visualise the protein they are researching. The third table, below this 3D viewer provides a list of all the drugs that have been docked against the selected protein stored on the system. The user has the option of looking at the selected protein-specific dockings making it easier for the user to find the results they require. Figure 5.5 shows the design for the ligand data page.

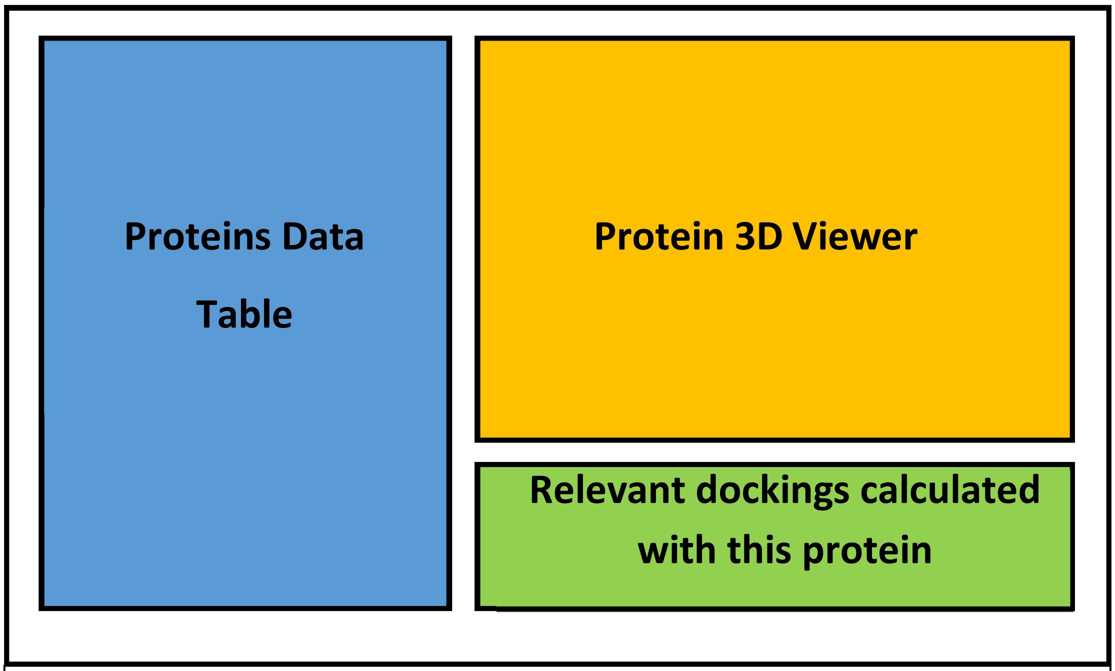


Figure 5.4: Design of the Proteins results page - shown is a basic design of how the protein viewer page would look to the user after selecting a protein within the protein table on the database page. A table contain all about the selected protein will be place on the left of the page. On the right of the page the 3D viewer and relevant docking s table will be placed.

This is a similar broad layout to the protein data page but displays data about the ligand selected instead of the protein. The ligands data table displays information such as ligand name, PubChem ID, Reactome ID's, KEGG ID's etc. The Ligand 3D Viewer displays the 3D PDB file of the Ligand selected. Finally, below the 3D viewer is a table with all dockings that have been performed using the ligand selected.

The docking's results page has details about both the protein and ligand involved in the docking that has been selected. This required separate tables for the protein data and ligand data. The following design shown in figure 5.6 was adopted for the docking results page. As shown, there is a Proteins data table on the left of the screen which displays the data of the protein involved in the 3D docking selected. On the right side of the screen, there is a ligands data table which displays information about the ligand involved in the selected docking. In-between these two tables is a 3D viewer which displays the 3D conformation of the calculated docking that has been selected. Finally, below these is a docking results table. This displays the results for the selected docking and all the scores that have been calculated for the conformation, which are stored on the system database.

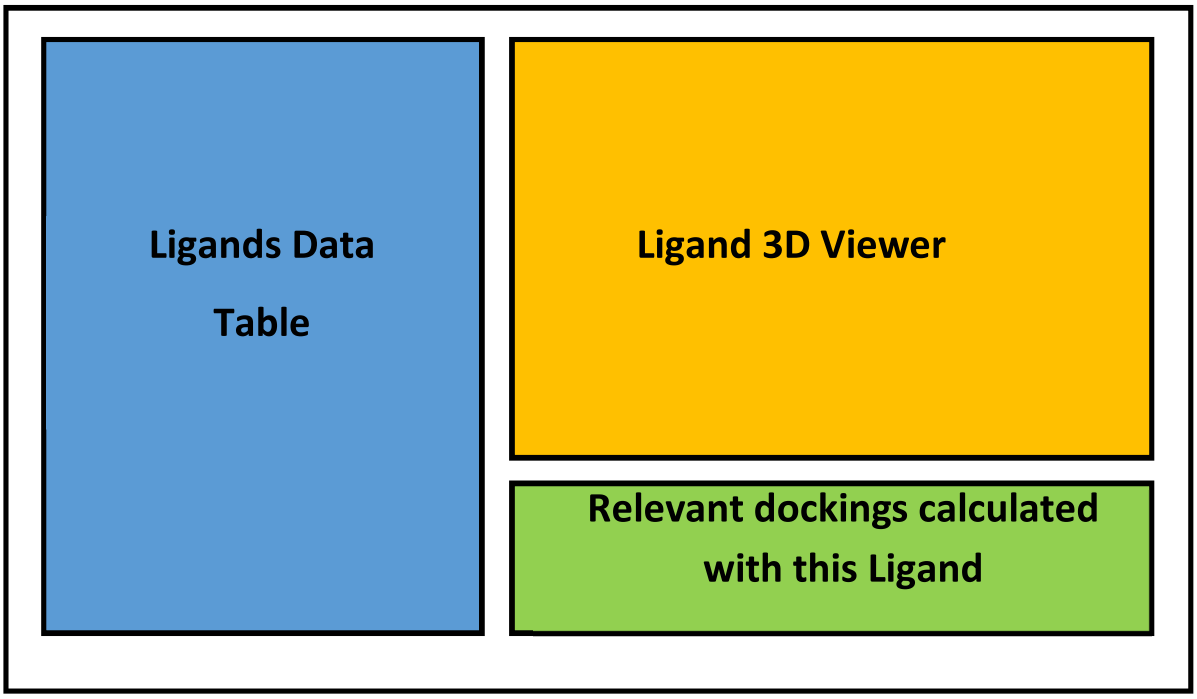


Figure 5.5: Design of the Ligand results page - shown is a basic design of how the ligand viewer page would look to the user after selecting a ligand within the ligand table on the database page. A table contain all about the selected ligand will be place on the left of the page. On the right of the page the 3D viewer and relevant dockings table will be placed.

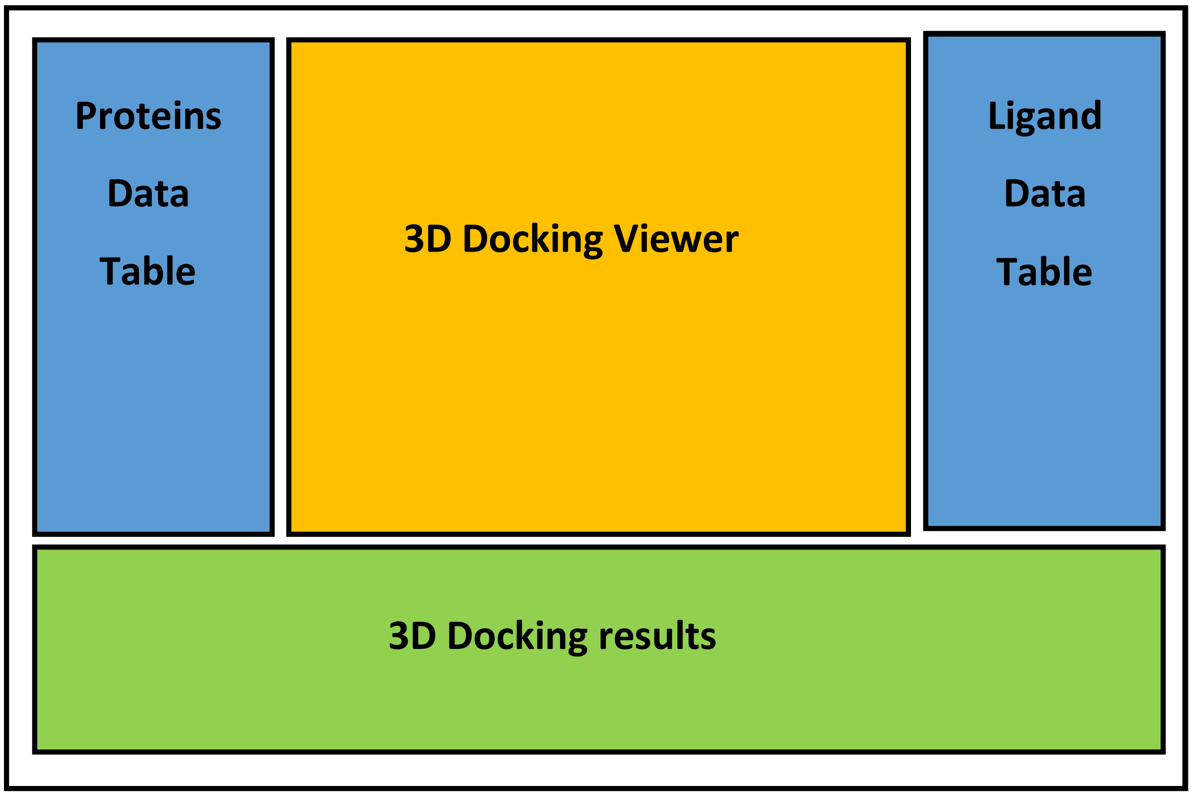


Figure 5.6: Basic design for the Docking results page - shown is a basic design of how the protein-ligand docking viewer page would look to the user after selecting a specific docking pair within the docking table on the database page. A table contain all about the selected protein will be place on the left of the page. On the right of the page a table containing the data about the selected ligand will be placed. the 3D viewer will be place in between the two tables with a table with the docking scoring function results at the bottom of the page.

5.3 GUI coding

To Create the GUI required the key components required are PHP pages which are used for database querying and to create the visual pages for the user. Also JQuery/JavaScript/AJAX pages which are used for seamless interaction between the visual pages and the MySQL database.

5.3.1 Database Page

How Protein data is generated and displayed on the GUI

The process of bringing protein data from the database to the GUI requires the use of PHP, AJAX, JavaScript, and HTML. The first part of the process was to create a method of querying the database to extract the correct information.

Protein Table Query

A PHP file was created to call ajax_proteins_from_temp-2.php to store the query to extract the protein data needed. This same file is also used to query the Ligand table; this will be explained further on. This file is used to query different tables when required; the queries are placed within an IF statement. Information is passed to the PHP file with a request to extract data from a particular table and which "temp" table to use to gather the protein ID's required. These are stored as variables; the table is stored as "$opt_table" and the temp table under "$opt_level". For the protein data, the query extracts data from the Proteins table of the database. This is then linked with the Protein_Info table which has more information about the proteins such as name and other associated ID's. The query checks the "temp" table created for the user which stores all of the ids of the proteins that need to be extracted. For this query, the "$opt_table" equals "Proteins" and the "$opt_level" equals "0". This is then inputted into another variable which equals the name of the temp table within the database. In this case the variable is called "$table_name" and is equal to "'temp_'.$opt_table.'_'.$opt_level". For the IF statement as the table is equal to "Proteins" all the set variables for this query are shown in table 5.1. The variables set by the IF statement are then inputted the MySQL query shown in row 1 of Table 5.2.

Table 5.1: Variables set within the ajax_proteins_from_temp-2.php file – the table shows the four variables that are set within the ajax_proteins_from_temp-2.php to extract protein data from the MySQL database.

Variable Name	Variable
$idcolname	'Protein_Index'
$info	'Protein_info.uniprot'
$info2	'Proteins.uniprot'
$table_info	'Protein_Info'

Table 5.2: Coding Table – this table shows all of the key code of MySQL, PHP, Javascript, and AJAX that were used in the creation of the system.

Row Number	Code
1	SELECT * FROM $opt_table INNER JOIN $table_info ON $info2 = $info WHERE $idcolname IN (SELECT * FROM $table_name);
2	$connection = mysqli_connect(“Server”, “username”, “password”, “database”) ;
3	$result = mysqli_query($connection, (SQL Query));
4	$data = array();
5	$(‘(TABLE TAG)’).html(rowHtml);
6	$(‘(TABLE TAG)’).DataTable();
7	SELECT * FROM Docking_Table JOIN Proteins ON Docking_Table.Protein_Index = Proteins.Protein_Index JOIN Ligands ON Docking_Table.Lig_Index = Ligands.Lig_Index JOIN Protein_Info ON Proteins.uniprot = Protein_Info.uniprot JOIN Ligand_Info ON Ligands.pubchem = Ligand_Info.pubchem WHERE Docking_ID IN (SELECT * FROM $table_name);
8	SELECT $table_name.*, Docking_Table.Protein_Index, Docking_Table.Lig_Index, Proteins.uniprot, Proteins.Variant, Proteins.model_type, Ligands.pubchem, Protein_Info.*, Ligand_Info.* FROM $table_name JOIN Docking_Table ON $table_name.id = Docking_TableDocking_ID JOIN Proteins ON Docking_Table.Protein_Index = Proteins.Protein_Index JOIN Ligands ON Docking_Table.Lig_Index = Ligands.Lig_Index JOIN Protein_Info ON Proteins.uniprot = Protein_Info.uniprot JOIN Ligand_Info ON Ligands.pubchem = Ligand_Info.pubchem;
9	$(‘(TABLE TAG) tbody’).on(‘click’,’tr’,function(){
10	Window.open((PHP file name).php?ID=+selectedValue);
11	var selectedIDs = document.getElementById(“runid”).innerHTML;
12	SELECT * FROM $opt_table WHERE $idcolname = ‘$selectvalue’;
13	$exe_str2= “cd (location of where all protein PDB’s are stored) ; dssp2pdb -35 (name of pdb structure file to be created) (protein pdb file location) > (new Protein pdb file name) 2>&1”;
14	$execute=shell_exec($exe_str2);
15	If (file_exists($pdb) && is_readable($pdb))
16	mysqli_select_db($connection, '3DProteomeDatabase'); $result=mysqli_query($connection, " SELECT * FROM $opt_table WHERE Docking_ID = '$RunID' ");
17	$result = mysqli_query($connection, "SELECT FROM $opt_table WHERE $idcolname = '$selectvalue' ");

Before this query can be executed, a connection between the PHP script and the database has to be made. The “$connection” variable is created so that a command can be inputted to establish a connection. The “$connection” variable was defined at the start of each PHP file that requires a database connection by the code shown in table 5.2 in row 2. The “Server” is the machine on which the database is stored. The “username” is the username of the MySQL server, and “password” is the password for that server. Finally, the name of the database is placed where “database” is located in the command. Now that a connection is established; the query can be run and be able to output a result. The query is executed by using the PHP code shown in table 5.2 in row 3. The output of this query is put into an array. This is so that the data can be easily accessed and sorted through using JavaScript. For this a “$data” variable by using the code shown in table 5.2 in row 4. This created a blank array. The array is used in a while loop which inputs each row of data one by one as an array record. This is subsequently encoded in JSON and is echoed so that the JavaScript file will be able to decrypt and read the results.

JavaScript protein query process

The JavaScript file decodes the JSON encoded data by using the “JSON.parse” function. The data is sorted through to obtain the relevant information needed for the table. A FOR loop is used to go through each record in the array and gather the relevant information. Within the FOR loop, the following variables are created, and a record column is assigned to it where the information can be extracted. The variables used are shown in table 5.3.

Table 5.3: Protein Query variables – shown in the table are what variables have been set for each column for the data extracted via JSON.

Variable Name	Variable
ProteinID	output[‘Protein_Index’]
Gene	output[‘proteinname’]
UniProt	output[‘UniProt’]
Variant	output[‘Variant’]
Modeltype	output[‘model_type’]
Date	output[‘Date_Created’]
proteinName	output[‘Protein_Name’]

The data extracted from each record is added to a variable called “rowHtml” which contains HTML code for creating a table row. The extracted data is placed in the order of the columns, which are a part of the graphical user interface table. Once the FOR loop has completed, and each record’s data has been extracted the table is loaded onto the interface. The HTML code stored in the variable "rowHtml" is outputted using the JavaScript code shown in row 5 of table 5.2 with the "TABLE TAG" being set as "#protein_table". This code outputs the HTML within the 'protein_table' DIV that has been created for the table in the HTML of the main page. This is initialized in DataTables, using the JavaScript code shown in row 6 of table 5.2 with the "TABLE TAG" variable being set as "#protein_table". This outputs the table with added functionality such as a search bar and also options of how many records the user would like to view on each page. It will also add the selected CSS styling chosen for the table.

Ligand Table Query

The query used to extract data from the Ligand table is the same as the query used for the Protein Table. The only differences are the variables used within the query. The same PHP file is used as the protein table query. As an IF statement was created, the options that are passed to the PHP file will be different from those for the protein query. The "$opt_table" variable is set as 'Ligands', and for this, the "$opt_level" variable is set as '0'. The IF statement follows the "$opt_table" variable in order to set the other variables. For this query the variables set to query the Ligands table are shown in table 5.4.

Table 5.4: Set variables for extracting Ligand table data – shown are the set varables used to extract data from the Ligands Table. These will be used in the same query used for the protein table query.

Variable Name	Variable
$idcolname	'Lig_Index'
$info	'Ligand_info.uniprot'
$info2	'Ligands.uniprot'
$table_info	'Ligand_Info'

This is inputted into the same query as the proteins query, which is shown in row 1 of Table 5.2. Like the proteins query, a connection is established with the database, and the query is executed with the same commands. The output is inputted into an array and encoded using JSON.

JavaScript Ligand query process

The JavaScript process to extract the data from the Ligand data array provided by the PHP file is the same as the proteins query, the only difference being the columns of data which are collected. The encoded data is decoded with the "JSON.parse" function and put into a FOR loop to go through each record one at a time. The columns that are extracted are shown in table 5.5.

Table 5.5: Ligand Query variables – shown in the table are what variables have been set for each column for the data extracted via JSON.

Variable Name	Variable
ligID	output['Lig_Index']
pubchem	output['pubchem']
Ligname	output['Lig_Name']
Drugbank	output['drugbank_id']
Date	output['Date_Created']

The data that is extracted from each record is added to a "rowHtml" variable the same as the protein query within the loop. Once the FOR loop has completed the "rowHtml" variable is outputted to the graphical user interface using the JavaScript code shown in row 5 of Table 5.2. The "ligand_table" is the DIV that the HTML coded on the ligand viewer page. The DataTables is initialized on this HTML code by the JavaScript code shown in row 6 of table 5.2 with the "TABLE_TAG" set to "ligand_table". Like the protein query, this gives the table added functionality.

Docking Table Query

For querying the docking table, a new separate PHP file is created. This newly created PHP file is required so that a large query can be passed to the database as there is a large amount of data involving multiple tables that have to be extracted. To provide the correct data for the user in the Docking table of the interface, 6 tables have to be queried within the database. These tables are the Docking_table, Protein_table, Protein_Info, Ligands_table, Ligand_info, and finally the temp table level, which is in use. The dockings that are extracted from the database all depend on the ids within the temp level table. Like the previous queries for the temp table level and for tables that are requested, the query criteria is as follows: the "$opt_table" equals "Docking_Table" and the "$opt_level" equals "0". This then creates the "table_name" variable as "temp_Docking_Table_(user)_0". The original query created for this is shown in table 5.2 in row 7. This query firstly requests all data within the Dockings table. The next step is to join this table with the Proteins table. This is so that details about the proteins involved in the docking can be matched. The tables are joined by using the "Protein_Index" columns of each table. Next, the Ligands table is joined to the Docking Table. This is done by using the link of the "Lig_Index" column of each table. Once these tables have been linked the next stage is to link the Proteins and Ligands table to each of their information tables. For the proteins table the "uniprot" column is used to link to the "Protein_Info" table, and for the Ligands table, it is linked with the "Ligand_Info" table using the "pubchem" column. The dockings requested to be displayed depending on the ID's stored within the “temp_Docking_Table_(user)_0” table. The temp docking table number can vary according to the number of filters the user applied, e.g. “temp_Docking_Table_(user)_1”, “temp_Docking_Table_(user)_2”, etc. To do this a WHERE and IN clause is added to the end of the query. The WHERE clause is used so that only the records that follow this rule are extracted. The “Docking_ID” column is used within an IN clause which means if the “Docking_Id” is within for example a list, those results would be extracted. After the IN clause the “(SELECT * FROM $table_name)” is placed. This MySQL query extracts every ID within the temp table used as a list for the IN clause.

When running this query on PHPMyAdmin, the query was extremely slow due to the number of records stored within the “Docking_Table”. It was taking up to 10 minutes to load and sometimes it crashed it. There was a need to optimize the query as 10 minutes is too long for a user to wait for results to load. After looking at the original MySQL query created and looking at different ways to optimize a query, the best method was only to extract specific columns from each table. For the Docking table, the only columns that are required are the Docking ID, Protein_Index, Lig_Index, and Date Created. Using an IN clause in a query slowed down the query considerably. The temp table only stores one column which is the id, and this id has numbers that relate to the Docking_ID in the docking table. So instead of using the IN clause, the temp docking table is included at the start within the SELECT clause. The query shown in row 8 of Table 5.2 is used to query the docking table of the MySQL database. The first line with the SELECT clause asks for specific columns from the Docking_Table, Proteins table, and Ligands table. Then for the Protein_Info, Ligand_Info, and temp table, the query asks for all the data within them. This is telling MySQL specifically what data is needed for the table so instead of getting every record and every piece of data associated with it the query now only gets the minimum amount of data required to satisfy the table outputted to the user. The query is run in the same manner as the Protein and Ligand queries using the “mysqli_query” command. The results are put into an associated array and encoded using JSON.

JavaScript Docking query process

As for the Protein and Ligand user interface tables, the JSON encoded data outputted by the PHP file is decoded and parsed using the “JSON.parse(output)” command. This is put into a FOR loop so that for each record the required data is extracted and inputted into a “rowHtml” variable with HTML code so that it can be uploaded to the User interface. The data required from each record within the array were the Docking ID, Protein Name, Gene Name, UniProt Code, Variant, Model type, Ligand Name, and PubChem ID. Once the loop has completed, and all the data has been extracted it is outputted to the user interface using the same code as the Protein and Ligand tables which are shown in row 5 of table 5.2 where the “TABLE TAG” variable is set to “#docking_table”. DataTables is initialised using following code shown in row 6 of table 5.2 where the "TABLE TAG" is also set to "#docking_table". Like the Protein and Ligand tables, this will give the Docking table extra functionality.

Protein Selection

The way the system has been designed enables the user to click on one of the proteins within the table outputted to the interface which creates another tab which displays all data about the selected protein, a 3D viewer showing the structure predicted and also a table showing a list of all dockings that have been calculated using that specific protein. Firstly, a is code created that makes it possible for a user to click a protein record in the table. To make it possible, the javascript code shown in row 9 of Table 5.2 is used with the "TABLE TAG" set as "#protein_table". This launches when a row is clicked within the "tbody" where the table items are stored. The function determines where the new window is created, and it contains all the information related to the selected protein.

For the protein-specific page to be created the protein ID need to be pushed to a PHP file which extracts all the relevant data for it. The code shown in row 10 of Table 5.2 is used to launch a new tab within the web browser with the "PHP file name" set as Proteins_view.php. The "window.open" function creates a new tab in the web browser instead of loading the page in the current tab. This enables the user to look at multiple proteins at a time instead of the option of only looking at one protein at a time in a single tab. This opens the "Proteins_view.php" page and pushes the Protein "ID" of the row selected to it.

5.3.2 Protein Viewer Page

The "Proteins_view.php" page displays all information about the selected protein and ligands that this protein has been docked with, in two different tables. This page has a mixture of HTML, PHP and JavaScript within it. Between PHP tags the pushed "ID" is added to a variable in the page called "$RunID". This is echoed in the page and is used to extract the relevant data from the database. A JavaScript file called "pViewer.js" is used to load up all the tables and the 3D Viewer with the data.

Protein Viewer Page Protein data table

The first data to be displayed is the Protein information table which is on the left side of the screen. A JavaScript function called "tableData" launches when the "Proteins_view.php" is opened. Within the function, a variable is created called "selectedIDs" which will store the ID of the Protein that has been selected. This is extracted from the PHP file "Proteins_view.php" by using the code shown in row 11 of Table 5.2. By using AJAX, to get the protein data, the protein ID is pushed to another PHP file called "get_database_data2.php". The name of the table that needs to be queried is also pushed to this file.

The “get_database_data2.php” file is the primary way for data to be extracted for a single protein, Ligand or docking for the system. Due to this file being used for multiple query purposes, an IF statement is within it so that depending on what table data is requested it will set the correct variables for the query.

To extract data from the proteins table, the table name pushed to this file is “Proteins” and is added to the variable “$opt_table”. This sets a variable called “$idcolname” to “Protein_Index” which is the column of the table that the variable “$selectedvalue” will be searched against. The “selectedvalue” variable will be the Protein ID that is pushed from the “pViewer.js” JavaScript file. Like the other queries for the main page, a connection is established to the MySQL database. This connection is used to submit the query shown in row 12 of table 5.2 using the variables. This gets all data from the Proteins table that has a “Protein_Index” value equal to the “$selectedvalue” value. The result of this query is put into an array and encoded using JSON to push back to the “pViewer.js” JavaScript file.

The “pViewer.js” JavaScript file decodes the JSON encoded array and inputs it into a FOR loop so that each column of data needed can be extracted and inputted into a variable. The data extracted from the following array were the Protein ID, Gene name, UniProt, Model type, and Variant. After this, a second query is required to extract the relevant data from the “Protein_Info” table. This uses the “get_database_data2.php” file again to query the table, the difference being the “$selectedvalue” variable pushed is the uniprot code of the protein and the “$opt_table” variable is set as “Protein_Info”. As there is an IF statement in the PHP file this sets the “$idcolname” variable as “uniprot”. The same query is used as for querying the Proteins table earlier but using the variable set here. This will output all data within that table, which has an equal uniprot column value as the “$selectedvalue” value. These are also put into an array and encoded in JSON to be pushed to the JavaScript file.

Using the same method as before the output is extracted to get all data from each column in the array and each column gives a separate variable. The data extracted from the query were the Protein Name, Organism, Other Known Gene names, Sequence length, and Sequence. All of this data is going to be stored into a “rowHtml” variable with HTML code which is uploaded to the user interface as a table. This table is outputted to a DIV called “prottable” which puts it into the correct location on the page which the CSS file has set it.

Protein Viewer Page Docking results table

On the protein viewer page, there is also a table which will include a list of drugs that have had a docking calculated with the selected protein. The user is also able to click on one of the drugs which will then open another window and display that selected docking. To display this, a function called "dockingdata" is run on page load which will extract the docking data for the selected protein. Within this JavaScript function, it uses the Protein ID that is echoed in the page from earlier to search the docking table for all relevant results for that protein. The protein ID is set as the variable "selectedIDs" like previously discussed for the protein data table, and this is then pushed to a PHP file. Using AJAX, the ID and table used are pushed to the PHP file called "get_database_data.php". For this query, as there are if statements which set a variable for the query, the table variable is set as "dockdata". This is because this file is used for other queries and there is already an IF statement option with the table "Docking_Table". By pushing these options, the variables set to query the database are shown in table 5.6.

Table 5.6: Variables set within the get_database_data.php file for extracting docking data – the table shows the three variables that are set within the get_database_data.php to extract docking data from the MySQL database.

Variable name	Variable
$selectedvalue	Protein ID pushed via AJAX/JSON
$idcolname	'Protein_Index'
$opt_table	'Docking_Table'

These variables are applied to the query in row 12 in table 5.2. The next step is to set up a connection to the database and run the query. The query results are inputted into an array and encoded using JSON. This is pushed back to the JavaScript file. The "pViewer.js" file decodes the JSON output and is put into a loop to go through each record and extract the relevant column needed for the table. The columns extracted are the Docking ID, Protein ID, and the Ligand ID. The next stage is to get the information about the ligand so that it can be displayed in the table. The ligand ID that is extracted and pushed to the same PHP file but instead the "selectedvalue" is now equal to the ligand ID, and the table is now equal to the "Ligands" table. As the table is now the "Ligands", the IF statement sets the variables shown in table 5.7 for the query.

Table 5.7: Variables set within the get_database_data.php file for extracting ligand data for protein docking table – the table shows the three variables that are set within the get_database_data.php to extract ligand data from the MySQL database.

Variable name	Variable
$ selectedvalue	Ligand ID pushed from the pViewer.js file via AJAX/JSON
$idcolname	'Lig_Index'
$opt_table	'Ligands'

These are put into the same query discussed above. These results are added into an array and encoded using JSON. As with the other results, these are decoded in the JavaScript file and run through a loop to extract data from each record. The data extracted from the ligand table is the PubChem ID. This is used to get the ligand name from the “Ligand_Info” table by using the same method but the variables set are shown in table 5.8.

Table 5.8: Variables set within the get_database_data.php file for extracting ligand information data for the protein docking table – the table shows the three variables that are set within the get_database_data.php to extract ligand information data from the MySQL database.

Variable name	Variable
$ selectedvalue	PubChem Id pushed from “pViewer.js” via AJAX/JSON
$idcolname	‘pubchem’
$opt_table	‘Ligand_Info’

The Ligand name is extracted from the array produced by the query. When all of the data has been gathered, all data is inputted into a “rowHtml” variable with HTML code which will create a table. This is outputted to the “docking_table” HTML tag in the protein viewer page, and DataTables is initialised to add extra functionality to it.

Protein Viewer page 3D Viewer

On the protein viewer page will be a 3D viewer which enables the user to view the predicted structure from either the homology or threading pipelines. The 3D viewer is created by using an open-source JavaScript library called PV - JavaScript Protein Viewer (https://biasmv.github.io/pv/). The viewer can show PDB structures and has a built-in secondary structure assignment to give the best visuals of the protein. Within the “pViewer.js” file the location of the PDB file of the protein is needed so that it can be loaded into the 3D viewer. A function called “load3Dprotein” was created for this process. Firstly, the protein ID is put into a variable called “thisPID”. This is pushed (svalue) with the table “Proteins” to the “get_database_data.php” page using AJAX to retrieve the location. The variables set within the PHP file are shown in table 5.9.

Table 5.9: Variables set to extract protein PDB file location – shown are the variable and their set data required to extract the file location of the PDB file for a selected protein from the MySQL database.

Variable Name	Variable
$selectvalue	svalue pushed from Javascript file
$opt_table	table pushed from Javascript file

This variable, after going through the IF statement, sets the “$idcolname” variable as Protein_index which are used in the query for the database. The results of the query are put into an array and encoded using JSON. The “load3Dprotein” function decodes the output, and a loop is run to access the correct column of the record. The data extracted from the record is the Protein Name and the location of the PDB. To access the PDB file, a PHP script called “pdb_proxy2.php” is used to open it and extract all of the text within it and also process that data. Using AJAX, the protein ID as the variable “pdb” and the table the data is stored in is pushed from the “pviewer.js” JavaScript file to this PHP file. Within the PHP file, the protein ID is set as the variable “filename” and the table as “opt_table”.

Firstly, in the PHP file, a query is used on the Proteins table to get the location of the PDB file for the protein with the selected ID using the “Protein_Index” column. The PDB file location is given as the variable “filename2”. For the protein to be able to show its secondary structure in the 3D viewer, this needs to be calculated before being uploaded to it. To do this a program called DSSP is used. DSSP is a program that has been developed by Wolfgang Kabsch and Chris Sander which has a database of secondary structure assignments for all proteins that have been inputted into the Protein Data Bank. To run this program in PHP, the code shown in row 13 of Table 5.2 is used. to execute that command within the PHP file the function shown in row 14 of Table 5.2 is used which allows the command to be executed in a shell which will, in turn, run the dssp program on the desired PDB file. This creates a new file with data about the secondary structure of the protein. The next stage is for the newly created file to be read and pushed to the JavaScript file. Firstly, an IF statement is created which checks that a location of the original PDB is set by using the “isset” function. If this condition is true, within the statement, a variable is created called “pdb” which will set the location of the newly created PDB file. When pdb variable has been set, another IF statement is created within the same IF statement that first checks if the file exists and if the file is readable. The IF statement used is shown in row 15 of table 5.2 where the “pdb” variable is inserted. If this condition is true, the content with the PDB file is extracted by using the “file_get_contents” function and is echoed. The file contents are passed back and put into a variable called “thisProteinText”. Some other variables are also created which are shown in table 5.10 which store

Table 5.10: Set variables for pdb information – shown are variable that are set in the javascript file that store additional information about the extracted protein PDB file.

Variable Name	Variable
thisIndex	thisPID
thisProtName	ProteinName

other information about the extracted PDB file. The final variable that is created is “thisProtein”. This is where the molecules are constructed for the model using an I/O function built in the PV viewer. The final variable set is the “proteinStackData” which is set as a dictionary to store the data shown in table 5.11.

Table 5.11: Variables set to for the data stored within the protein PDB – shown are what parts of the extracted protein PDB are set to within the “pdb_proxy2.php” file.

Variable Name	Variable
struct: thisProtein	the calculated molecules for the viewer
text: thisProteinText	the contents of the protein PDB file
name: thisProtName	Name of the protein

Finally, a function called “drawSingleProt” is launched to initialise the 3D viewer in the Protein viewer page. The “drawSingleProt” contains the code that sets the settings for the 3D viewer. Firstly, two variables are created, one is called “protein” which stores the data stored within the “proteinStackData” variable, which is used in the “load3Dprotein” function. The other being “drawType” which sets how the 3D protein model looks within the 3D viewer. The viewer can display the protein in different ways that are built into the viewer, such as balls and sticks, lines, spheres, and cartoon. A Switch statement is used so that when a particular condition is set, this will determine how the protein will be viewed. The Switch function is shown in figure 5.5. The variable “drawType” is set to “drawTypep”; this sets options that are for the protein in the viewer and nothing else. The “drawTypep” is automatically set to “cartoonStr” when the protein viewer page is first loaded, but this can be changed by the user depending on which button the user clicks for the viewer which will be explained further on. This means that the function will set the “thisProtStruct” as “viewer.cartoon('protein', protein['struct']);” which will initialise the viewer with that setting. The user will have multiple buttons next to the 3D viewer on the Graphical user interface for the protein viewer. Each of these buttons is linked to a function within the “pViewer.js” file which sets the value of the “drawTypeP” variable which in turn changes what “thisProtStruct” variable is equal to and alters the way the Protein is displayed on the screen. The protein viewer is outputted to the “protimg” div tag which places it in the centre of the screen by the linked CSS. The User is able to zoom in and of the structure and also alter the viewing angle by turning it using the mouse.

```
switch(drawType){
                              case ("sticks"):
                                        var thisProtStruct=viewer.ballsAndSticks('protein', protein['struct']);
                                        break;
                              case ("lines"):
                                        var thisProtStruct=viewer.lines('protein', protein['struct']);
                                        break;
                              case ("spheres"):
                                        var thisProtStruct=viewer.spheres('protein', protein['struct']);
                                        break;
                              case ("cartoonPos"):
                                        var thisProtStruct=viewer.cartoon('protein', protein['struct'], { color:
color.rainbow() });
                                        break;
                              case ("cartoonStr"):
                                        var thisProtStruct=viewer.cartoon('protein', protein['struct']);
                                        break;
                              case ("tube"):
                                        var thisProtStruct=viewer.tube('protein', protein['struct']);
                                        break;
                              case ("points"):
                                        var thisProtStruct=viewer.points('protein', protein['struct']);
                                        break;
                    }
```

Figure 5.5: Switch function which will change view type of the 3D protein within the 3D viewer

Ligand Selection

The user is also able to click a ligand within the table outputted to the interface which creates another tab which displays all data about the selected Ligand, a 3D viewer showing the structure, and also a table showing a list of all dockings that have been calculated using that specific Ligand. Like the proteins table, the code created makes it possible for a user to click a Ligand record in the table. To make it possible, a JavaScript function is created. The code that is used is shown in row 9 of table 5.2 which is applied to each table row. This code launches when a row is clicked within the "tbody" where the table items are stored. Within the function, the new window is created and has all the information related to the selected Ligand. Firstly, for the ligand-specific page to be created the ligand ID is pushed to a PHP file which extracts all the relevant data for it. The code shown in row 10 of Table 5.2 is used to launch a new web browser window. The "PHP file name" in the code is set to "Ligand_view.php" with the selected value being the ligand ID of the clicked table row. The "window.open" function creates a new tab in the web browser instead of loading the page in the current tab. This is so that the user is able to look at multiple proteins at a time instead of the option of only looking at one protein at a time in a single tab.

5.3.3 Ligand Viewer Page

The “Ligand_view.php” page displays a 3D image of the Ligand and two tables, one with information about the Ligand and one with all proteins that have stored dockings which have been calculated using the selected ligand. This page has the same structure as the Protein viewer page. The only difference being the tables that are queried to extract the information from the database. Like the protein viewer page, a variable “$RunID” is pushed from this page to a JavaScript page called “lViewer.js” which sets up the page with the correct information and structure.

Ligand Viewer Page Ligand data table

Like the protein viewer page, the first item to be displayed is the Ligand data table which stores all data about the selected ligand. The only difference between the “pViewer.js” file and the “lViewer.js” file is that it uses Ligand data to extract the database instead of protein data. When the page is first loaded the function “tableData” is launched straight away which creates and fills the Ligand data table. The “$RunID” that was extracted from the “Ligand_Viewer.php” file is put as the value of the variable “selectedIDs” in the JavaScript file. By using AJAX, this variable is pushed to the “get_database_data.php” file the table value as “Ligands”. The data is extracted from the Ligand table for the selected ID and added to an array which is encoded using JSON. Shown in table 5.12 are the variables that were set in the “get_database_data.php” file.

Table 5.12: Variables set within the get_database_data.php file – the table shows the three variables that are set within the get_database_data.php to extract ligand data from the MySQL database.

Variable Name	Variable
$selectedvalue	Ligand ID pushed via AJAX/JSON
$idcolname	‘Lig_Index’
$opt_table	‘Ligands’

These variables are inputted into the query shown in row 12 of table 5.2 and executed. The JSON array is decoded by the “lViewer.js” JavaScript file, and the data extracted were the Ligand ID, PubChem ID, and Drugbank ID. After these have been extracted the PubChem ID is used to extract the correct ligand data by being pushed to the “get_database_data.php” file as the svalue and the table being “Ligand_Info”. By using the table name, shown in table 5.13 are the variables that were set. These variables are used in the query shown in row 12 of table 5.2 to extract the Ligand info for the selected ligand. The result is added to an array and encoded. The encoded array is decoded by the “lViewer.php” file and by using a loop extracts the Ligand name. When all the appropriate data has been extracted, these are added to a “rowHtml” variable which will set up the HTML code for the table to be created. Like the Protein viewer page, this is uploaded to the “ligtable” DIV tag which is placed in the location on the screen determined by the CSS file.

Table 5.13: Variable set within the get_database_data.php file to extract ligand data from the Ligand_Info table - the table shows the three variables that are set within the get_database_data.php to extract ligand information data from the Ligand_Info table within the MySQL database

Variable Name	Variable
$selectvalue	PubChem ID pushed via AJAX/JSON
$idcolname	‘pubchem’
$opt_table	‘Ligand_Info’

Ligand Viewer Page Docking results table

Like the protein viewer page, there is a table on the right side of the screen which will have a list of all the proteins that have had dockings calculated using the selected Ligand. The user is also able to click on them, which will open another tab with the selected docking. To create this table like the protein viewer page, a function called “dockingdata” is run on the page load. This function will be used to extract the docking data and data of the proteins from the database. The core of this function is the same as the one used in the protein viewer page, the only difference being the variables pushed to the PHP files, and that ligand data is extracted not protein data. This function firstly gets the ID of the Ligand selected and push this to the “get_database_data.php” file as the “svalue” variable and the “table” variable will be set as “dockdata2”. Within the “get_database_data.php” file the dockdata2 the variables set are the same as shown in table 5.12 the only difference being that the variable “$opt_table” is set to “Docking_Table” instead of “Ligands”. These variables are entered into the query shown in row 12 in table 5.2 like for the Ligand Viewer Page Ligand data table. The results are added to an array and encoded in JSON. The encoded results are pushed back to the JavaScript file where they are decoded. The decoded array is put into a loop so that each row is read individually to the Docking ID, Protein ID, and Ligand ID. For each record, the Protein ID is pushed to the “get_database_data.php” file so that data about the proteins can be extracted. The “table” variable is set to “Proteins” which sets the variables shown in table 5.14.

Table 5.14: Variable set within the get_database_data.php file to extract protein data from the Proteins table - the table shows the three variables that are set within the get_database_data.php to extract protein data from the Proteins table within the MySQL database.

Variable Name	Variable
$opt_table	‘Proteins’
$svalue	Protein ID pushed via AJAX/JSON
$idcolname	‘Protein_Index’

These variables are inputted into the query shown in row 12 of table 5.2 like previously for Ligand viewer page ligand data table. Like previously, these results are added to an array and encoded. The JSON array is decoded, and the UniProt, Model type, and Date created columns are extracted. The UniProt value is used to extract the Protein data from the “Protein_Info” table for each protein. So again the “get_database_data.php” file is used to extract this data with the UniProt value and the “table” variable set to “Protein_Info”. This sets the variables shown in table 5.15.

Table 5.15: Variable set within the get_database_data.php file to extract protein data from the Protein_Info table - the table shows the three variables that are set within the get_database_data.php to extract protein information data from the Protein_Info table within the MySQL database.

Variable Name	Variables
$opt_table	‘Protein_Info’
$selectvalue	uniprot value pushed via AJAX/JSON
$dcolname	‘uniprot’

These are also used in the query shown in row 12 of Table 5.2. The query is executed and added to an array. The array is encoded and pushed back to the “lViewer.js” JavaScript file. The JSON array is decoded and inserted into a loop where the Protein Name and Gene column data are extracted. The Docking ID, Protein Name, Gene, Model type, and Date created for each docking are all added to a “rowHtml” variable which adds HTML tags to them so that a table can be created. The code that is used is shown in row 5 and 6 of table 5.2, which uploads the table to the “Ligand_view.php” interface and initialise DataTables. The “TABLE TAG” for both lines of code were set to “#dock_table”. Like the protein viewer page, the user is able to select individual dockings which open a new tab with that docking.

Ligand Viewer page 3D Viewer

Identical to the 3D viewer in the Protein_viewer.php web page, the user is able to view every single ligand on their page which is selected from the ligand table of the database.php page. The “Ligand_viewer.php” page has its JavaScript file called “lViewer.js” which has all the code that is within the “pViewer.js” file but modified to satisfy the need to display the ligand instead of a protein. To get the location of the ligand PDB file, a function called “load3DLigand” is used. This is the same as the “load3Dprotein” function used in the “pViewer.js” file. The “get_database_data.php” is passed the variables shown in table 5.16.

Table 5.16: Variables set within the “get_database_data.php” to extract Ligand PDB data – shown are the variables set to extract Ligand PDB data for use within the 3D viewer.

Variable Name	Variable
Table	Ligands
Pdb	The ID of the Ligand

Within the PHP file, the information about that ligand is extracted from the Ligand table and outputs it as a JSON array back to the JavaScript file which the ligand file location is extracted with the PubChem ID. The next stage is to access the ligand PDB and extract its contents. A PHP file is used called “pdb_proxy3.php” which uses the drug ID used at the beginning and the table “Ligands” as its variables which are pushed to it using AJAX. The PHP file is based on the “pdb_proxy2.php” file used for the Protein viewer but with some minor changes. There is no need for DSSP to be used on the Ligand as it is not a protein with secondary structure.

Within the PHP file, the PDB file is opened if it exists and the contents within are extracted which are then echoed back to the JavaScript file so that it can be read. All the contents, as for the protein viewer page are added to a dictionary called “ligandStackData” with the variables shown in table 5.17.

Table 5.17: Variables set to for the data stored within the Ligand PDB – shown are the part of the Ligand PDB that is set to within the “pdb_proxy2.php” file.

Variable Name	Variable
struct: thisLigand	the calculated molecules for the viewer
text: thisLigandText	the contents of the Ligand PDB file
name: thisLigName	Name of the Ligand

By using the same method as that used in the protein viewer section, the contents are pushed to the “drawSingleLigand” which is the same as “drawSingleProt” function used in “pViewer.js” file, which displays the Ligand in the viewer on the page. The slight differences compared to the protein viewer page is that the drawType sets as “drawTypel”. This is so that only display options with drawType = drawTypel can be used for changing how the ligand is displayed in the 3D viewer, for example, to use the view options Tube, sticks, spheres and dots. The Ligand viewer is outputted to the “ligimg” div tag which places it in the centre of the screen by the linked CSS. The user is able to zoom in and of the structure and alter the view by turning it using the mouse.

Docking Selection

Finally, the user will be able to click a single docking within the docking table on the main page, which will open a single tab for that selected docking. The docking page will contain a table with information about the protein used in the selected docking and information about the ligand used in the selected docking. Also, a 3D Viewer which shows the 3D docking which the user is able to view and analyse. The user is also able to view any of the ten conformations predicted by the docking algorithms, separately or all together, in the single viewer. Finally, there is a table displaying all the associated scores with the selected conformation of the selected docking. The entire page is loaded by using a single JavaScript file called "viewer.js".

Protein information and Ligand Information Tables

These are created by using the same method used in the Protein Viewer and Ligand Viewer pages to display all the data associated with the protein used within the selected docking. The function called "dockingdata" is used to gather the information needed for the tables. Firstly, there is a need to extract what protein and ligand are used in the selected docking. To do this, AJAX is used to extract this information from the database. The ID of the docking selected is pushed to the JavaScript file as the variable "thisDID" which is used as the docking ID pushed to the "get_database_data.php" with the Table variable as "Docking_Table". Within the PHP file, the data of the selected docking is extracted from the "Docking_Table" which is encoded using JSON and pushed back to the JavaScript file. The function decodes the data and extracts the Protein ID and Ligand ID of the protein and ligand used in the selected docking and adds them to the variables shown in table 5.18.

Table 5.18: Variables set for the get_database_data.php file to extract docking data for the protein and ligand information data tables – these are the set variables to extract protein and ligand information form the protein_info and lig_info tables in the MySQL database.

Variable Name	Variable
LigaID	Ligand ID
ProtaID	Protein ID

These ID's is used to extract the associated data for the protein and ligand. The data for the protein is first extracted using the identical method used for the Protein Viewer page. Using the same AJAX call to the "get_database_data.php" the protein ID extracted previously (ProtaID) is used as the svalue with the table variable being set as "Proteins". This will extract all the information about the protein with the same ID in the Proteins table and pushes it back to the "viewer.js" JavaScript file.

With the results pushed, the function extracts the Protein Name, Uniprot ID, model type, and Variant. Next, the uniprot id's that were extracted are used to get the information associated with the selected protein from the "Protein_Info" table. The PHP file returns the extracted data to the JavaScript file, which extracts the protein data Gene, alternative names, sequence length, protein sequence. These are all put into a variable called "rowHtml" with HTML code which creates a table on the selected docking viewer page. This is outputted to the docking viewer page using the code shown in row 5 of table 5.2 with the "TABLE TAG" being set as "#protein_table".

The same process is used but, in this case, to extract the Ligand data from the database. The "LigaID" is pushed to the "get_database_data.php" with the table variable "Ligands" which will extract the data associated with the ligand ID. This is encoded and pushed back to the "viewer.js" JavaScript file which decodes and extracts the PubChem ID, drugbank ID, and Date loaded to the database. The PubChem ID is used to extract the data from the "Ligand_Info" table for the ligand used in the docking. The data extracted from this is the Ligand name. The data collected is added to a variable "rowHtml2" with HTML code which creates a table with the ligand data within it. The code shown in row 5 of Table 5.2 is used to add it to the ligand_table div on the page with the "TABLE TAG" being set to "#ligand_table".

5.3.4 Docking 3D Viewer

The 3D viewer on the docking viewer page is the same as the 3D viewers on the protein and ligand viewer pages. The only difference being how the protein and conformations are loaded into the viewer within the "viewer.js" JavaScript file. The basic structure is the same compared to the "pViewer.js" and "lViewer.js" JavaScript files, but as there are multiple conformations that can be loaded into the 3D viewer, the new code has been developed to accommodate this.

Protein PDB upload

Like the protein viewer page, a function called "load3Dprotein" is used to load the protein into the viewer. The ID of the protein involved in the selected docking was extracted previously within the "tableData" function, described earlier, which is used in this function to get the location of the PDB file. This is pushed to the "get_database_data.php" file as the svalue variable with the variable table equalling "Proteins" by using AJAX. This extracts the record associated with the ID pushed within the Proteins table and encode it in JSON and return it to the JavaScript function. This is decoded and inputted into a loop which then only extracts the data from the column's protein name and the location of the PDB. After this data has been extracted the protein ID is used again and pushed to the "pdb_proxy2.php" file which runs DSSP on the PDB to add secondary structure data using the code shown in row 13 in table 5.2. The PDB file is read, and the contents are pushed back to the function within the JavaScript file. The contents of that PDB file are added to the variable “thisProteinText”, and the protein name and ID are set as the variables “thisProtName” and “thisIndex”. These are added to a dictionary called “proteinStackData” which stores all the data as shown in table 5.19. Finally, the function “drawSingleProt” is started which initialises and loads the protein PDB to the 3D viewer on screen.

Table 5.19: Variables set to for the data stored within the protein PDB – shown are what part of the protein PDB are set to within the “pdb_proxy2.php” file.

Variable Name	**Variable**
struct: thisProtein	the calculated molecules for the viewer
text: thisProteinText	the contents of the protein PDB file
name: thisProtName	Name of the protein

Ligand PDB upload

This is where the code differs compared to the ligand viewer page as there are up to 10 conformations that will be able to be displayed on the 3D viewer with the protein involved. This all starts a JavaScript function called “load3Ddocking”. Within this function, the conformation file is opened, and each conformation is split into its entity which makes it possible for the user to select what conformation to view on the screen.

The Docking ID that was pushed to the docking viewer page is used to gather the information about the location of the conformation file for the selected docking. AJAX is used to extract this information from the database. A PHP file called “get_ligand_models2.php” is used to extract this data by using the variables shown in table 5.20, which are pushed to it from the JavaScript function.

Table 5.20: Variables set to extract ligand conformation file location – shown are the variable set within the get_ligand_models2.php file which would allow the location of the specific conformation file for the selected docking pair to be extracted from the MySQL database

Variable Name	**Variable**
pdb	Docking ID
Table	Docking_Table
Numofres	1

The first thing is for these to be set as different variables within the PHP file, so the PDB variable is set as pdb2, and a filename, the table as $opt_table, and the Numofres is set as Numoflig. By using these variables, the first thing the PHP file does is to extract the selected docking record from the Docking_Table by using the following code shown in figure 5.6. When the record has been extracted

```
mysqli_select_db($connection, '3DProteomeDatabase');
$result=mysqli_query($connection, "
            SELECT *
            FROM $opt_table
            WHERE Docking_ID = '$RunID'
            ");
```

Figure 5.6: code that establishes a connection with the MySQL - the shown code is used to extract the specific docking record from the Docking_Table within the MySQL database by setting the $opt_table and $RunID variables.

from the table, within a while loop the "results_file" column is extracted and set as the variable "filename2". Once the conformation file location has been extracted, there is a need to unzip the file. The way the file system is set up each protein has its zip file which will contain all the ligand conformation file that had been calculated for that protein. The file is zipped to save hard drive space on the system. Shellcode is used to extract the files stored in the zip file, which allows access to the conformation data required.

An IF statement is used to check firstly that the file that is required exists and that it is readable. If both statements are true, Using OBABEL the conformation file is converted to the correct PDB file format . The conformation files are in a MOL2 format which is the output given by the docking algorithms. OBABEL also splits all the ligand conformations stored within the file into their own entity. Once the MOL2 file has been converted and split is to access each of those separate conformations and extract the contents and add them to an array which is encoded using JSON and pushed back to the JavaScript function "load3Ddocking". These are decoded, and a for loop is used to extract each conformation data separately. For each of the conformations within the array, they are added to a dictionary called "ligandStackData". Shown in table 5.21 are the variables which will store the data for each of the conformations.

Table 5.21: Variables set for the data stored within the ligand conformation PDB – shown are what part of the ligand conformation PDB are set to within the file.

Variable Name	Variable
Struct: thisStruct	the calculated molecules for the viewer
text: thisLig	the contents of the ligand PDB file
i: LigandID	Ligand ID number
j:j	Conformation number within the file

Once all the conformations have been added to the dictionary, the function "drawLigands" is launched which displays the conformations within the 3D viewer. Within this function, it is determined what conformation is displayed within the 3D viewer. The user can choose which conformation 1 to 10 or all at the same time by selecting one of the radio buttons on the screen. On page load, the top conformation is displayed automatically. The “drawLigands” function first sets the “drawType” to “drawTypeL” which is for functions that control how the ligand conformation will look in the 3D viewer. An IF statement is opened which first checks that the “ligandStackData” dictionary is not empty because if it was that means there are no conformations to display. If the dictionary does contain conformation information, the function enters another for loop which extracts the conformations required. The code shown in figure 5.7 is used to upload the required conformation(s) to the 3D viewer. The variable “thisLig” is where the number of the conformation which is required to be shown is set. The variable “conf” is set by the radio button which is selected by the user on the screen. In this case, only the top conformation is shown in the viewer, so the “conf” variable is set to 0. This will extract the top conformation from the “ligandStackData” dictionary. There are different options of how the conformation looks on screen, the default being sticks. This displays the top conformation with the selected protein within the 3D viewer in its calculated docking coordinates.

Docking conformation results table

Finally, on the screen is the table which displays all the calculated scores for the conformations that have been selected to be displayed within the 3D viewer. The user is able to view the scores of individual conformations 1 to 10 or view them all at once. To display the requested conformation data, a JavaScript function called “dockingResults” is used. First thing this function does is to collect all the Scores associated with the selected docking. Within this function, the docking ID of the page is used and set as the variable “thisRID”. The function is used to extract all the scores that are stored

```
for (j in usedLigandPoses){
		var thisLig=usedLigandPoses[conf];
		var ligName='ligand';
		switch (drawType){
			case ("sticks"):
				var thisLigStruct=viewer.ballsAndSticks(ligName, thisLig['struct']);
				break;
			case ("lines"):
				var thisLigStruct=viewer.lines(ligName, thisLig['struct']);
				break;
			case ("spheres"):
				var thisLigStuct=viewer.spheres(ligName, thisLig['struct']);
				break;
			case ("dots"):
				var thisLigStruct=viewer.customMesh(ligName);
				ligCenters.addSphere(thisLig['struct'].center(),0.5,{color:'#0000FF'});
				viewer.requestRedraw();
				break;
			}
```

Figure 5.7: Switch function which will change view type of the 3D ligand within the 3D viewer – shown is the code used that will allow the user to change the view type of the ligand within the 3D viewer.

in the database table “Scores” which are linked to this ID. To extract this data an AJAX call is used to send and receive data to the PHP file “get_dockdata.php” which extracts the data from the database. The variables set within the PHP file are shown in table 5.22.

Table 5.22: Variable set within the get_database_data.php file to extract docking scoring data from the Scores table - the table shows the three variables that are set within the get_dockdata.php to extract docking score information data from the Scores table within the MySQL database.

Variable Name	Variable
$selectvalue	selected docking ID
$opt_table	Scores (table within the database)
$idcolname	Docking_ID (column within the Scores table)

The query shown in row 17 of Table 5.2 is used to extract the data from the Scores table, which links to the Docking ID of the selected docking. For all the scores extracted an array is created to store all the data. A WHILE loop is used to split up the data so that all scores associated with one conformation and is added to a dictionary within the array so that all scores linked to one conformation are put together under one entry. When the creation of the array has been completed, it is encoded using JSON and pushed back to the JavaScript function.

The JSON array is decoded and read by the function. For extracting the data from the array, there are two options which are decided by the user. One being if the user would like to view all the conformations data and scores at once or to view each conformation individually. This requires an IF statement which determines what data is uploaded to the page. The variable “confdata” will determine what the IF statement will do. When the page is loaded “confdata” automatically set to 0 which is the top result of all the conformations. The two options, one being if “confdata” is set as “allposes”, and an ELSE statement if it is not set to it. For this case, “confdata” is not set to “allposes” so the ELSE statement would be used. Within this statement, many variable values are set, which are

```
var Dock_affinity= obj[confdata].Dock_Affinity;
var Dock_van_der_Waals = obj[confdata].Dock_van_der_Waals;
var Dock_Electrostatics = obj[confdata].Dock_Electrostatics;
var Dock_Internal_energy = obj[confdata].Dock_Internal_energy;
var GBSA_Affinity = obj[confdata].GBSA_Affinity;
var GBSA_van_der_Waals = obj[confdata].GBSA_van_der_Waals;
var GBSA_Electrostatics = obj[confdata].GBSA_Electrostatics;
var GBSA_Born_solvation = obj[confdata].GBSA_Born_solvation;
var GBSA_Solvent_accessible_surface_area =
obj[confdata].GBSA_Solvent_accessible_surface_area;
var Vina_Affinity = obj[confdata].Vina_Affinity;
var Vina_Repulsion = obj[confdata].Vina_Repulsion;
var Vina_H_bond = obj[confdata].Vina_H_bond;
var Vina_Hydrophobic = obj[confdata].Vina_Hydrophobic;
var DSX_Atom_distance = obj[confdata].DSX_Atom_distance;
var DSX_Contact_count = obj[confdata].DSX_Contact_count;
var DSX_Per_contact = obj[confdata].DSX_Per_contact;
```

Figure 5.8: Scoring function variables for the docking data table – shown are all the variables that are set for all the scoring functions extracted from “confdata”.

shown in figure 5.8 so that a table can be created on the webpage. The important variable is the "confdata" variable which is used to extract the correct scores associated with that conformation ID. The code shown above are the variables that are created for all the scores associated with the conformation number. The "obj[confdata]" code is telling which conformation dictionary to extract the data from, for example, the dictionary with ID 0. The name of the column is used to extract the score associated with it. The results are added to a variable called "rowHtml3" with some HTML code which will create a table when added to the Docking viewer page. This is uploaded to the docking viewer page by using the code shown in row 5 of table 5.2 with the "TABLE TAG" set to "Dockres" to load the table to the DIV tag "Dockres" which is hardcoded into the main docking viewer page.

After the docking viewer page has loaded, the user is able to change which conformation data is shown on screen by using the radio buttons provided. These radio buttons are linked to a click function within the "viewer.js" file that when selected will change the data within the docking results table automatically. This is done by using If and ELSE IF statements. Within each option, two variables are given values depending on what conformation is selected. Shown in table 5.23 are the variables used within the ELSE IF statement.

Table 5.23: Variables set within the IF ELSE statements within the viewr.js file – shown are the variables set within the IF ELSE statement which will allow the user to select which conformation they would like to view within the 3D viewer.

Variable Name	Variable
conf	the number of the conformation selected for what conformation to display in the 3D viewer
confdata	the number of the conformation selected for what data to display in the docking results table

Within each of the IF and ELSE IF statements the functions "drawLigands" (to draw the conformation within the 3D viewer) and "dockingResults" (to display the associated results in the docking results table) are launched. This makes viewing the results for the user interactive.

5.3.5 Database Page HTML

The visual page is a PHP file but consists of mainly HTML code. This is where the content is inputted. This page consists of a "head" tag and a "body" tag. The "head" tag has within its things such as links to external libraries and JavaScript files and is linked to CSS files to the page. Within the "body" tag is where all the content is inputted into.

Firstly, within the “body” tags, DIV tags are added which have ids and classes associated with them. The created JavaScript file can send content to these tags which position the content needed and links with CSS which tells where and how the content should look on the screen. There are DIV’s used to identify different areas of the page so that the CSS can position the tables to the desired area. The database page has three different DIV’s for the locations of the tables. The protein table DIV is the top left of the page; the ligand table is the top right of the page in line with the proteins table, which is shown in figure 5.3. These will take up 47% of the screen width each. The DIV for the docking table is below both other tables and be set for 95.6% of the screen length. CSS is used to tell the web browsers where these DIV’s should be placed on the screen. The following CSS is used for these 3 DIV’s shown in figure 5.9.

Protein and Ligand DIV’s

```
.col-md-6 {
Width: 47%;
}
```

Docking DIV

```
.col-md-11 {
Width: 95.6%
}
```

Figure 5.9: CSS used for page containers – the code shown is used to set columns within the webpage which will then be used to help set locations for tables and the 3D viewers

For designing the tables, a JQuery plug-in is used called DataTables (http://datatables.net). DataTables is a library of CSS and JavaScript files that have been created explicitly for displaying and giving enhanced functions to any HTML table. To design how the table looks, Bootstrap (http://getbootstrap.com/) is used. The DataTables code is within the main JavaScript file for the main page, which initializes the DataTable functions once the data has been generated to display.

To prepare the database page for the data that is generated by the JavaScript file, a component needs to be added to the HTML code so that the JavaScript functions can be coded to interact with them. The Protein and Ligand table sections have two radio buttons and three submission buttons respectively. The two radio buttons are for two options for the user either to view all the proteins/ligands in the system or to look at a saved list that the user may have created. There is an Update button which when clicked displays the number of proteins available from the list that has been selected. The next button within the Protein and ligand table panels is for adding a database filter. This is how the user will be able to filter down the results they are looking at to their specific search. JavaScript is used to interact with this button which adds the option, which will be explained further on in the chapter. The final button within these panels is an update query button. When clicked this will start a function that extracts the data that matches the criteria of the search and output it into the panel using DataTables. To create the radio buttons for the panels, the “input” tag is used within the HTML with the option “type=’radio’” set. Both radio buttons are given the same name so that the JavaScript file can identify the two different options which can be linked with a single function. The only difference is the “value” variable given to each radio button. To create the submission buttons on the page, the HTML tag “button” is used with the option “type=’button’” set. Each of these buttons is given a different ID which associates with a function within the JavaScript file when clicked. The only difference between the Protein and ligand panels and the docking panel are that there’s one extra radio button that can be selected within the docking panel. The user can select multiple proteins and ligands from the protein and ligand panel. The extra radio button option is so when selected; the docking table only displays dockings involving those selected on the page. Finally, on the database page is the three main DIV with the ID’s of “protein_table”, “ligand_table”, and “docking_table”. These are used to output the results generated for the protein, ligand, and docking tables.

Search filtering

As mentioned before having temp tables that store all the ID’s of the proteins/ligands/dockings to extract and display in the tables will enable them to be filtered down according to the decisions made by the user. The user is able to add filters that they can search so that it narrows down the number of proteins/ligands/dockings that are displayed. This helps the user to find the desired results from a large amount of data that is stored in the system. The user is able to add these filters by using a button on the table panel. The user is able to add as many filters as they want, but for every filter added a new temp table is created as will now be explained. By using JavaScript, every time a filter is added and executed a new temp table will be created in the MySQL database which stores all the ID’s that are matched with the search criteria inputted by the user. Figure 5.10 shows the design of the proteins table of the main page, which includes the database filter button. When all the filters have been added the user then clicks the button “Update Table” which then updates and loads all the data that equals all the database filters set. This set up is a similar design to the Ligand and Docking table on the database page but set up for Ligand table data and docking table data respectively.

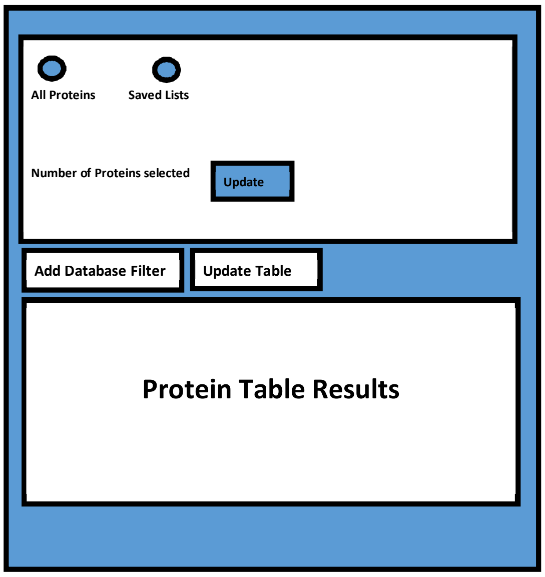


Figure 5.10: Protein table filter design – shown is the design of the protein table panel of the database page.

5.4 GUI Implementation

This section is how the graphical user interface has been implemented and how a user can use it for protein, ligand, and protein interaction investigations.

5.4.1 Login page

The login page is the first page that the user will view after typing in the URL for the website. Shown in figure 5.11 shows the created login page. The created logo for the site is displayed in the middle of the page. When the user clicks the logo, a form will open where they can log in to the database or sign-up. If the user has not got an account, there is a hyperlink on the form which will allow them to sign-up to use the database. By clicking the hyperlink, it takes the user to a form shown in figure

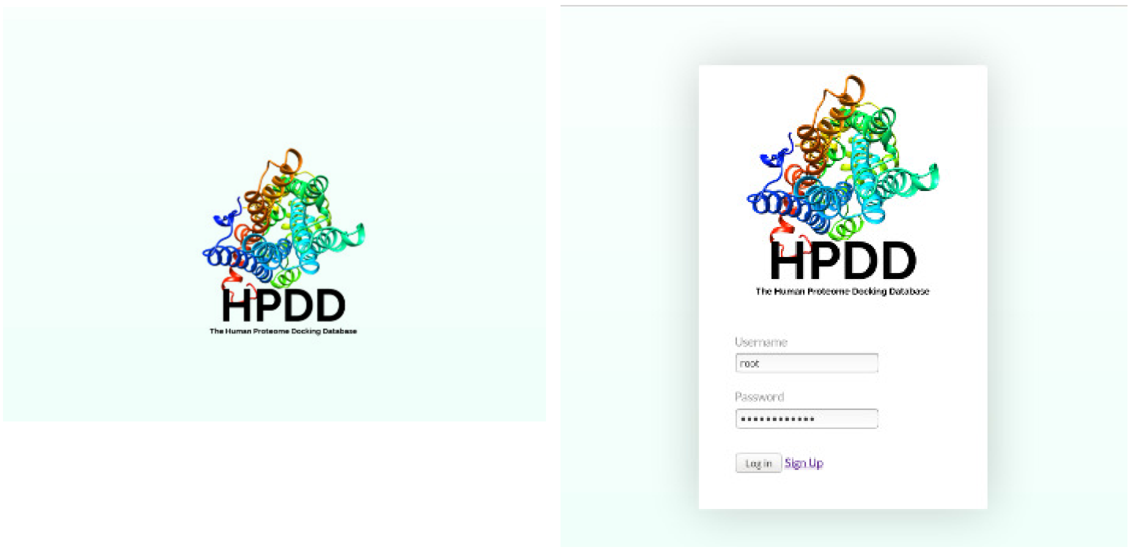
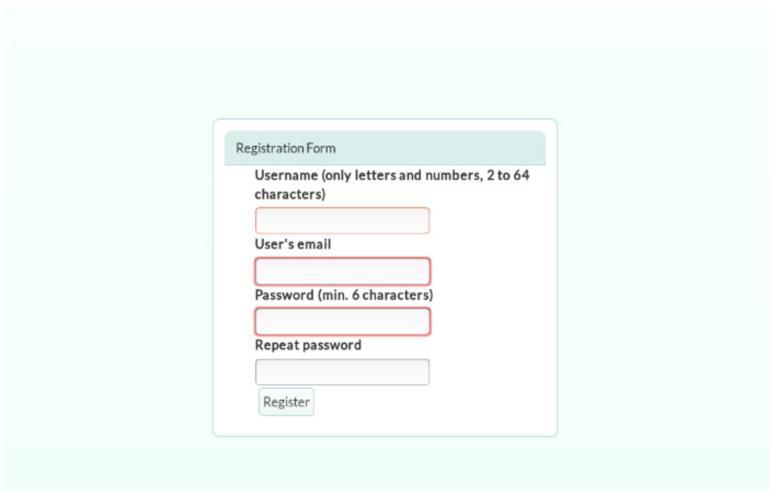

HPDD
The Human Proteome Docking Database
HPDD
The Human Proteome Docking Database
Username
root
Password
•••••••••••••
Log in
Sign Up

Figure 5.11: login page – shown is the login page which the user views. The user clicks the logo shown in the left photo which will open the login form. The login form has a username and password inputs with a login button and sign up (registration) button.

5.12. There are four input boxes on the form. The top one is for the user to create a username to use to log in. The second input box is for the user to give their email address.



Registration Form
Username (only letters and numbers, 2 to 64 characters)
User's email
Password (min. 6 characters)
Repeat password
Register

Figure 5.12: registration form – shown is the registration form which allows the user to create an account for the system which will grant them access to the docking database.

Finally, the last two input boxes are for the user to set a password. There are two boxes which are used to verify that the password meets the credentials needed for it to be accepted by the system and to verify that the users have typed the correct password they would like to use. When the user has completed the form, they then proceed by clicking the register button. If all the information is correct, then the user will then be taken back to the login page so that they can proceed to log in. In figure 5.11, when the user has login credentials, they input the username into the first input box followed by the password in the second input box. When completed the final step is to click the login button to gain entry to the database.

5.4.2 Database page

If all login credentials were correct, this then allows the user to access the central database page. The database page is where the user will be able to select proteins, ligands, and docking pairs to view in the system. The database page viewed is shown in figure 5.13. Three forms are visible, one for protein selection, ligand selection, and docking selection.

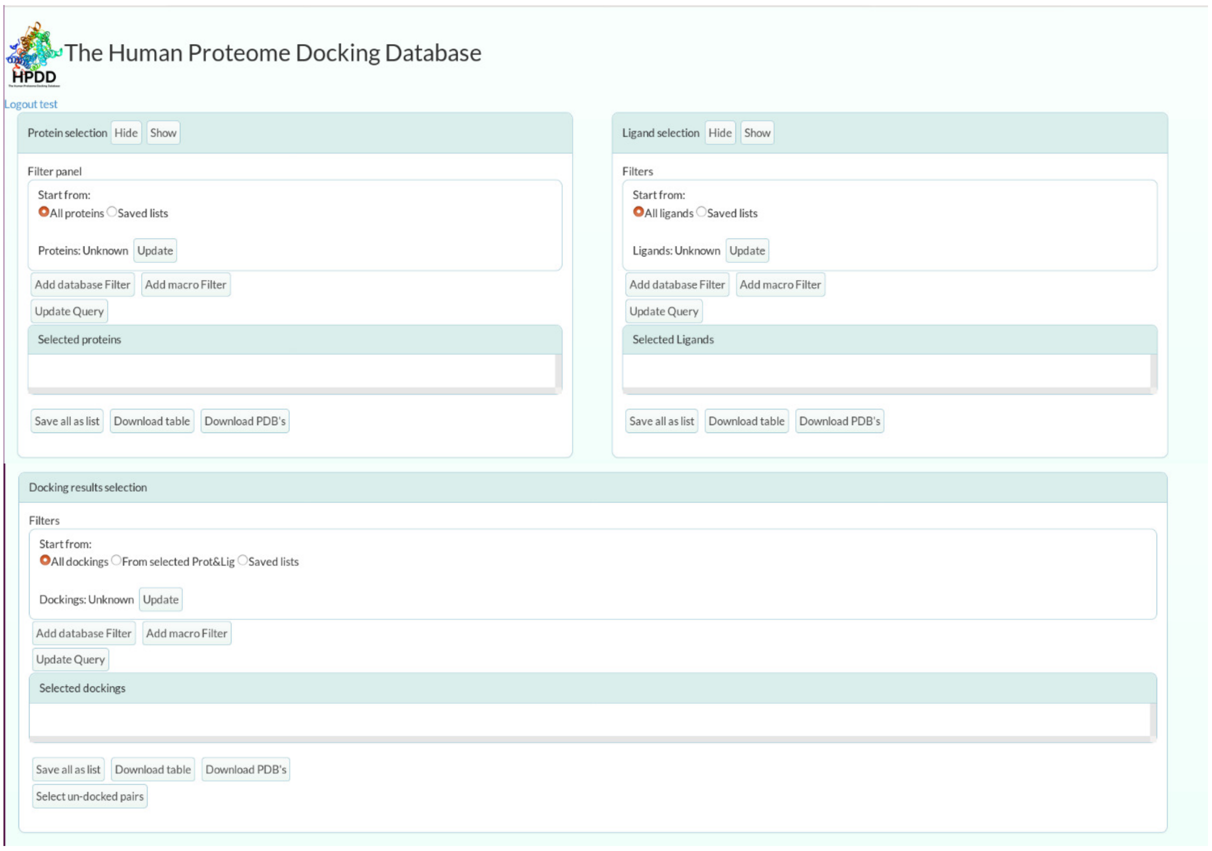


Figure 5.13: database page - shown is the database page which the user views after login. The page displays a proteins selection panel (top left), Ligand selection panel (top right), and a docking selection panel (bottom).

Protein selection table

The protein selection table displays the proteins that are stored in the database which is shown in figure 5.14. The first options for the user involve deciding whether they would like to view all the stored proteins in the database or look at a saved list of specific proteins. The saved list selection will display all saved lists for that specific user if they have previously saved a group of proteins. After a list or the "All Protein" option has been selected the user clicks the "Update" button to load in those proteins. Shown in figure 5.15 shows how many proteins are within the user's selected set. When the loaded proteins have been selected, the user will have options to use other filters to narrow down the loaded protein set to find the specific protein of interest.

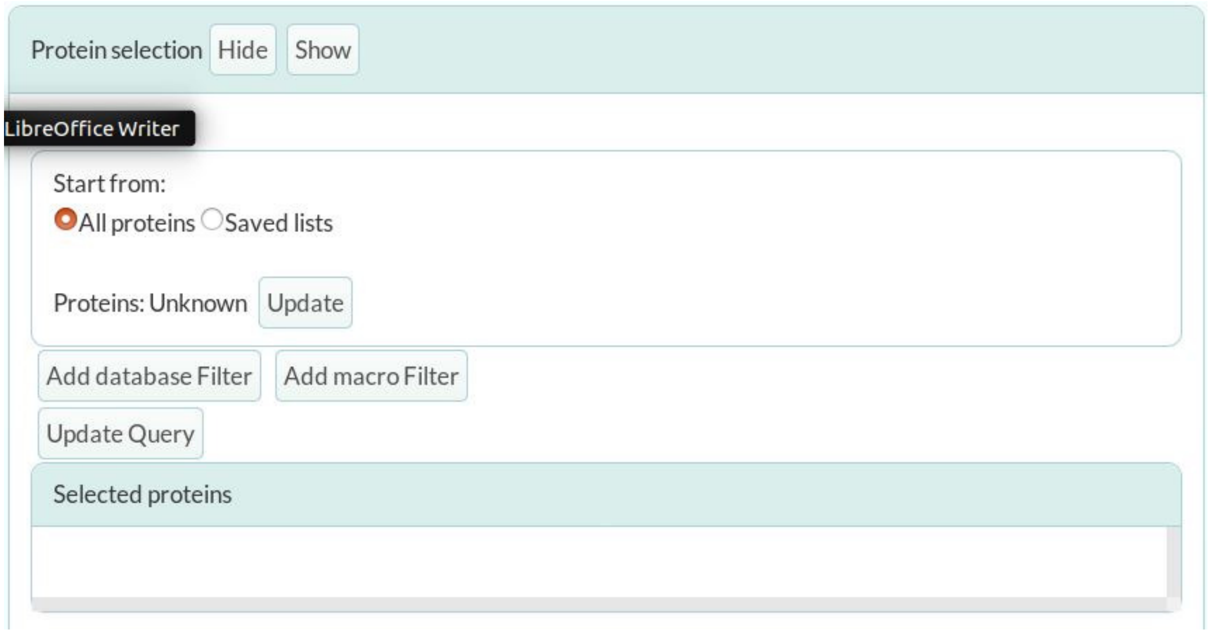


Figure 5.14: Protein selection panel – shown is the Protein selection panel were the user can select the protein of interest which are stored in the database. The user can add filters and view a list of proteins in table form.

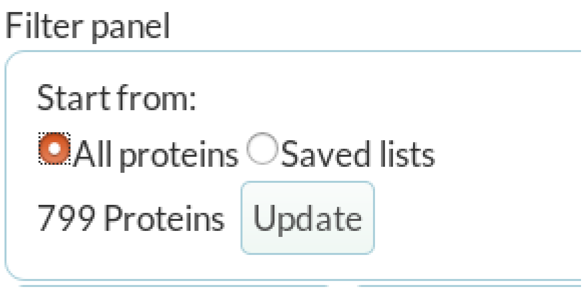


Figure 5.15: filter option panel within the proteins selection panel – shown is an option panel where the user can select either to search all proteins which are stored in the database or a saved list they may have created in a previous session.

Protein table filter options

Shown in figure 5.16 is an example of a filter that can be selected by the user to narrow down the results of the loaded proteins. The user can add a filter by clicking the “Add database Filter” button shown in the red box in figure 5.16. Shown in figure 5.17 are the different columns that the user can apply a filter too. The user has the option to either select columns of stored data within the database or can search using a string. Once the user has selected the options or search term desired, the “Update” button shown in figure 5.16 within the green box is the clicked. The system will then gather all the proteins that meet the selected criteria. Shown in the blue square in figure 5.16 is the number of proteins that are within the filtered set. The user can remove any the filter by clicking the remove button shown in the purple box in figure 5.16.

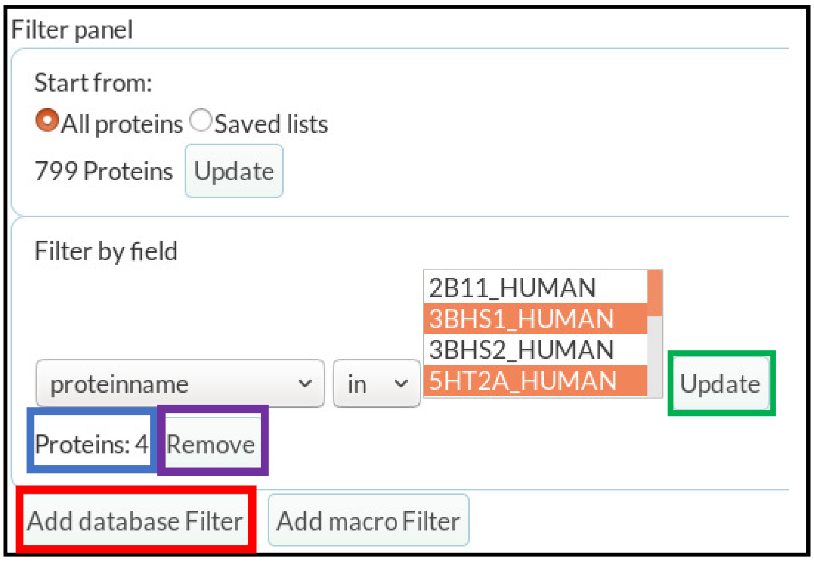


Figure 5.16: shows the view of the protein filtering options available to the user: Shown in the green box is the “Update” button to set the filter. Shown in the blue box is an indicator of how many proteins meet that criteria. Shown in the purple box is a “Remove” button which deletes the filter. Shown in the red box is how a user can add a filter another filter to the query.

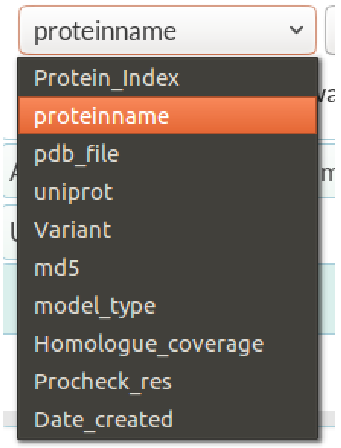
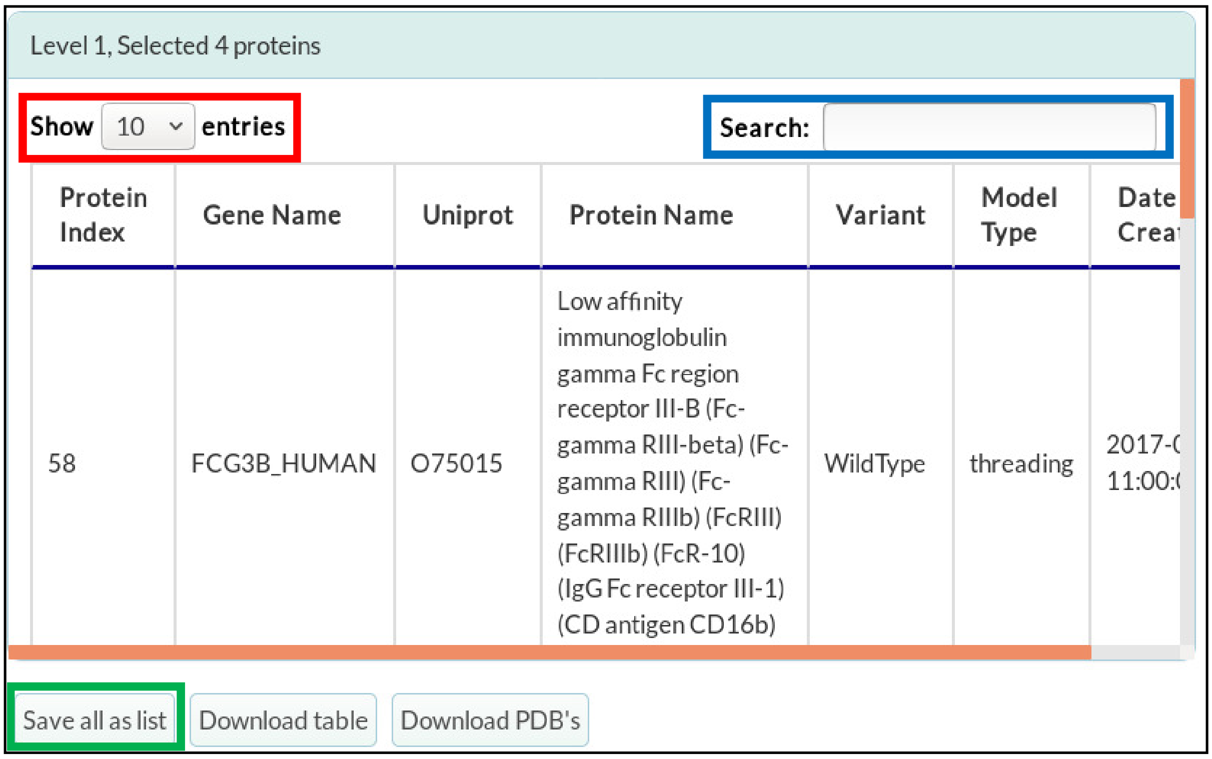


Figure 5.17: filter column options for the protein table – shown are the different columns whichare able to be searched using the search filter in the protein panel.

Protein results table

With the selected protein chosen, the user would then click the “Update Query” button which will load the data for that selected protein into a table. Figure 5.18 shows the full table after the “Update Query” button had been clicked. The full table shows the Protein Index, Gene Name, UniProt ID, Protein Name, Variant, Model Type, and the Date Created.Within the red box shown in figure, 5.18, the user can select the number of data rows are shown on each page of the data table. The options that are selected for the number of results displayed on each page are 10, 25, 50 and 100. The blue box shown in figure 5.18 is where the user can search the full table for specific strings such as a protein name or date created. The user has the option to save the loaded protein results as a personal list. Instead of searching for those selected proteins every time the user logs in, they will be able to pre-load the saved list with those proteins of interest. The green box shown in figure 5.18 shows the “Save all as list” button which will save the loaded proteins as a list. To view further data and a 3D model of any of the loaded proteins in the protein table, the results row is clicked opening a dedicated page for that protein.


Level 1, Selected 4 proteins

Show 10 entries

Search:

Protein Index	Gene Name	Uniprot	Protein Name	Variant	Model Type	Date Crea
58	FCG3B_HUMAN	O75015	Low affinity immunoglobulin gamma Fc region receptor III-B (Fc-gamma RIII-beta) (Fc-gamma RIII) (Fc-gamma RIIIb) (FcRIII) (FcRIIIb) (FcR-10) (IgG Fc receptor III-1) (CD antigen CD16b)	WildType	threading	2017-(11:00:(

Figure 5.18: Protein results table view - Shown is the table that is loaded with protein data for the proteins for the selected protein. shown in the blue box is a search bar that can be used to search the proteins within the table. Shown in the red box are option of how many results to show on one page. Shown in the green box is the "Save all as list" which saves a private list of all the proteins within the table.

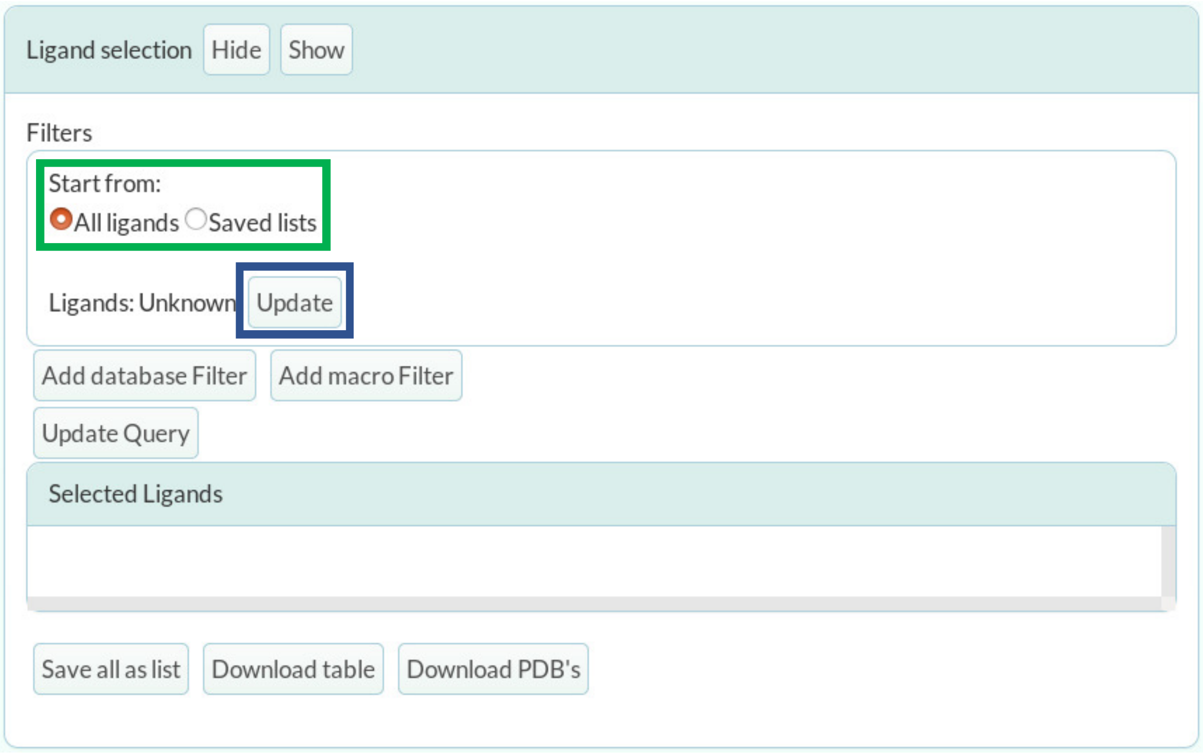
Ligand selection table

The ligand selection table is the same as the protein selection table discussed previously. The screen view of the ligand selection table is shown in figure 5.19. Like the protein selection table, the first option to choose is to either view all of the ligands that are stored within the database or load up a saved list. This selection is shown in the green box in figure 5.19. Once either option has been selected, the "Update" button should be clicked which will load up the selected proteins from the database.

Ligand table filter options

Like the filter options discussed for the protein selection table, by clicking the "Add database filter" button, the user has the option to add a filter to the currently loaded ligands. Figure 5.20 shows how this function will look on the screen. The columns can have filters to the Ligand Name, PubChem ID, DrugBank ID, and Date created columns.

Either a string can search within the selected column or options can be selected from a list of stored data in the selected column. After a search string or data filters have been selected, the "Update"


Ligand selection Hide Show

Filters

Start from:
[✓ All ligands] Saved lists

Ligands: Unknown Update

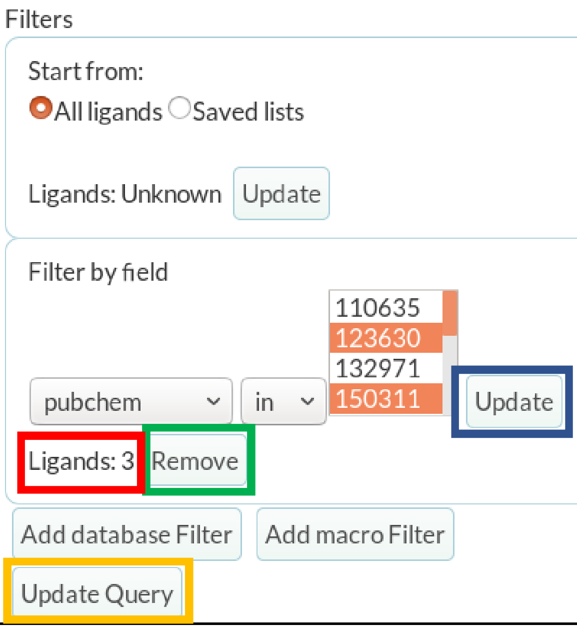
Add database Filter Add macro Filter

Update Query

Selected Ligands

Save all as list Download table Download PDB's

Figure 5.19: Database page for Ligand selection panel - shown in the green box are options to view all ligands in the database or a saved list. Shown in the blue box is an "Update" button which will load the selected protein data.


Filters

Start from:
[✓ All ligands] Saved lists

Ligands: Unknown Update

Filter by field

pubchem in 110635 123630 132971 150311 Update

Ligands: 3 Remove

Add database Filter Add macro Filter

Update Query

Figure 5.20: The database page's protein selection panel filters - shown is the view of the ligand filtering: shown in the blue box is an "Update" button that will apply the filter. Shown in the red box are the number of ligands that meet the criteria of the filter. Shown in the yellow box is the "Update Query" button which will update the main ligand data table. Shown in the green box is the "Remove" button which the user clicks to remove the added filter.

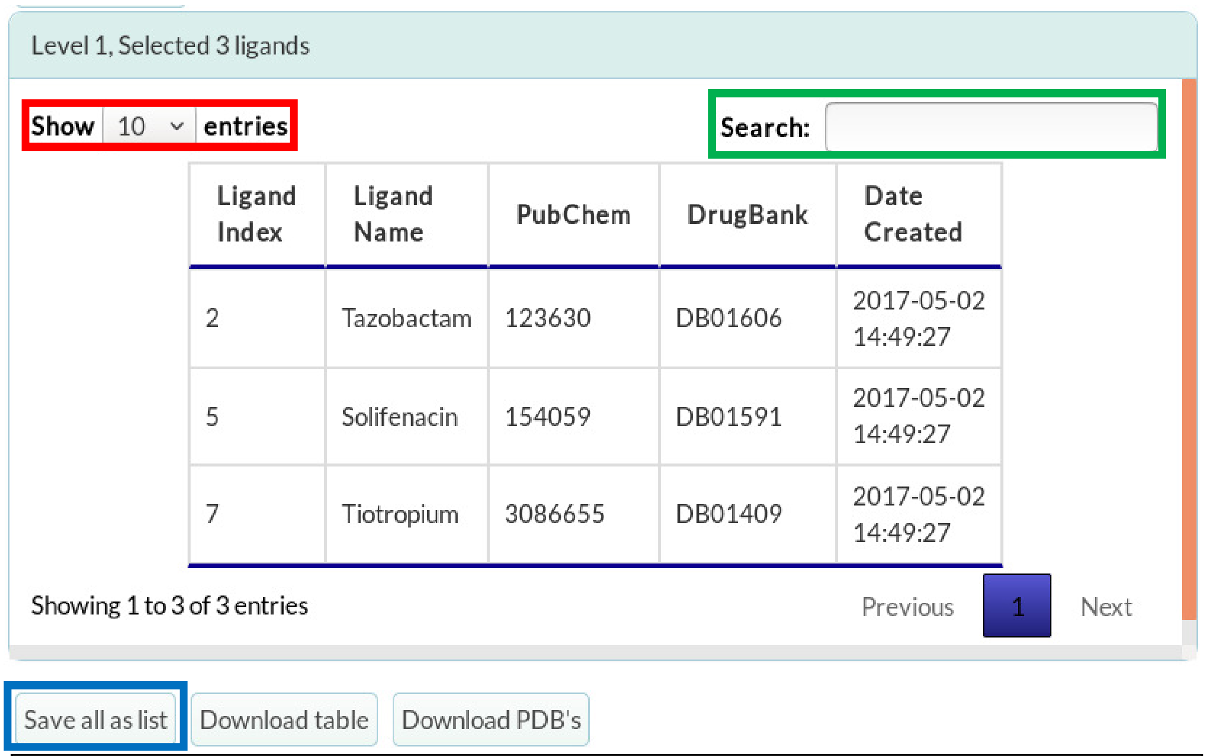
button is then clicked which will apply the filter. The number of ligands that meet the selected criteria of the filter is shown in the red box in figure 5.20. To remove any filter, click the Remove button shown in the green box in figure 5.20. After all filters have been applied, clicking the "Update Query" button loads all the ligands that meet the selected criteria.

Ligand results table

Like the proteins results table, the ligand results will be loaded into a table, as shown in figure 5.21. The data shown for the ligands within the ligand results table are the Ligand Index, Ligand Name, PubChem ID, DrugBank ID, and the Date Created. The number of results shown on each page can be selected by a drop-down list shown by the red box in figure 5.21. The options being to show 10, 25, 50, and 100 on a single page. The data in the table can also be searched using the search bar shown in the green box in figure 5.21. The blue box shown in figure 5.21 shows the "Save all as list" button which will save the loaded ligands as a list. To view further data and a 3D model of any of the loaded ligands in the ligand table, The results row is clicked opening a dedicated page for that ligand.

Docking selection table

The docking selection table will display the docking data that is stored in the database. Figure 5.22 shows the docking selection table as it is viewed on the webpage. Like both the proteins selection table and the ligand selection table, there are two options to either load all the stored docking data



Level 1, Selected 3 ligands

Show 10 entries

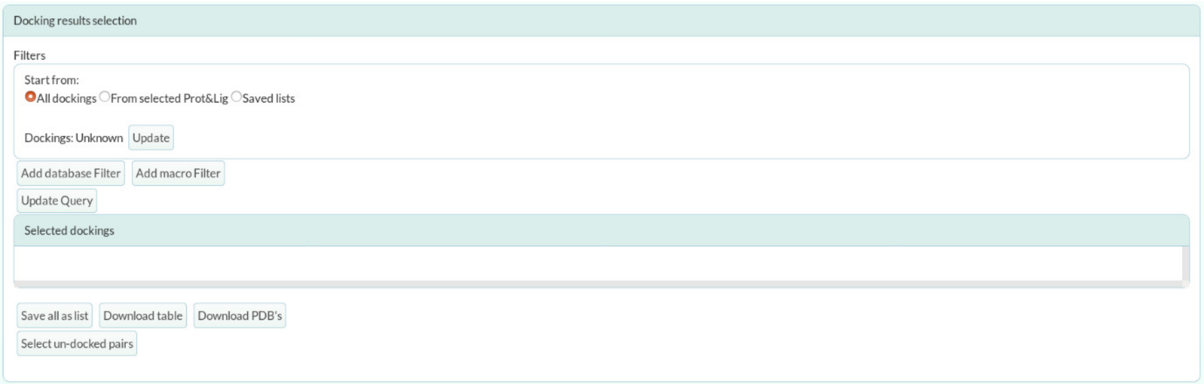
Search:

Ligand Index	Ligand Name	PubChem	DrugBank	Date Created
2	Tazobactam	123630	DB01606	2017-05-02 14:49:27
5	Solifenacin	154059	DB01591	2017-05-02 14:49:27
7	Tiotropium	3086655	DB01409	2017-05-02 14:49:27

Showing 1 to 3 of 3 entries

Figure 5.21: Example table of ligand panel results on the database page - Shows the table that is loaded with ligand data for the selected ligand: shown in the green box is a search bar that can be used to search the ligands within the table. Shown in the red box are option of how many results to show on one page. Shown in the blue box is the "Save all as list" which saves a private list of all the ligands within the table

within the database or load up a previously saved list. When one of these options is selected, the "Update" button is clicked which loads the docking results into the form.


Docking results selection

Filters

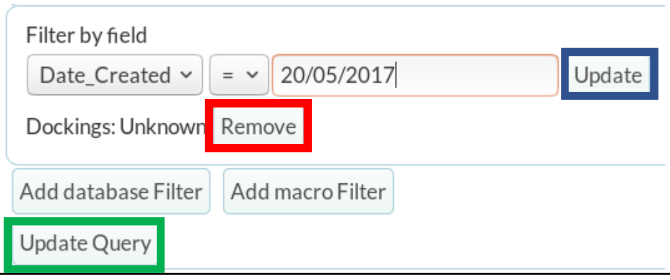
Start from:
[✓ All dockings] From selected Prot&Lig Saved lists

Dockings: Unknown Update

Figure 5.22: Database page docking selection panel – Shows the docking selection panel where can search for specific docking pairs.

Docking table filter options

Like previous tables, filters can be added to the loaded docking data, as shown in figure 5.23. When a search string or data criteria has been selected, the "Update" button shown in the blue box in figure 5.23 is then clicked which will then load only the data that meets that criteria. Any of the filters can be removed by the "Remove" button shown in the red box in figure 5.23.


Filter by field

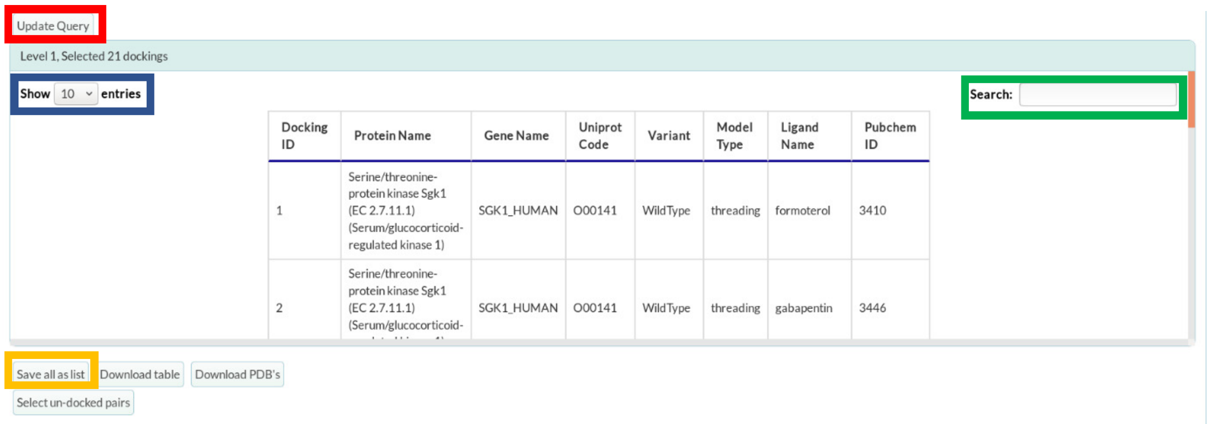
Date_Created = 20/05/2017

Dockings: Unknown

Figure 5.23: Database page's dockings selection panel filters - shown is the view of the ligand filtering: shown in the blue box is an "Update" button that will apply the filter. Shown in the green box is the "Update Query" button which will update the main ligand data table. Shown in the red box is the "Remove" button which the user clicks to remove the added filter.

Docking results table

Like both the protein and ligands selection tables, the selected docking results are loaded into a table, as shown in figure 5.24 when the "Update Query" button is clicked. The "Update Query" button is shown in the red box in figure 5.24. The docking data loaded into the table are the Docking ID, Protein Name, Gene Name, UniProt Code, Variant, Model Type, Ligand Name, and the PubChem ID. The number of results shown on each page can be selected in the blue box shown in figure 5.24. The options being to show 10, 25, 50, and 100. The docking data loaded into the table can be searched by using the search bar shown in the green box in figure 5.24. The filtered docking results table can be saved as a personal list. This can be done by clicking the "Save all as list" button shown in the yellow box in figure 5.24.


Update Query

Level 1, Selected 21 dockings

Show 10 entries

Search:

Docking ID	Protein Name	Gene Name	Uniprot Code	Variant	Model Type	Ligand Name	Pubchem ID
1	Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1)	SGK1_HUMAN	O00141	WildType	threading	formoterol	3410
2	Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-	SGK1_HUMAN	O00141	WildType	threading	gabapentin	3446

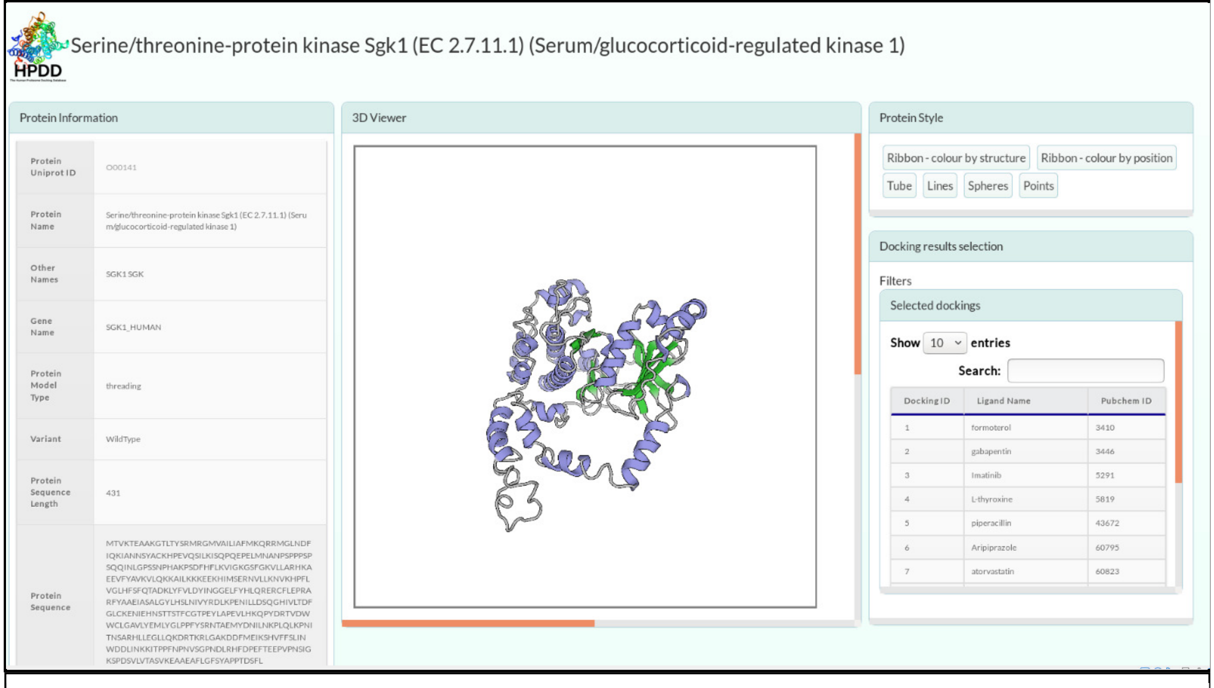
Figure 5.24: Database page's docking selection results table – Shown is the table that is loaded with docking data for the selected proteins and ligands. shown in the green box is a search bar that can be used to search the dockings within the table. Shown in the blue box are option of how many results to show on one page. Shown in the yellow box is the "Save all as list" which saves a private list of all the dockings within the table

5.4.3 Protein viewer page

When a user clicks one of the proteins from the protein selection table, a dedicated window opens for the selected protein. Shown in figure 5.25 is an example of a dedicated protein viewer page for Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1).

Protein information panel

Shown on the left side of the screen in figure 5.25 is the protein information table. The table will display information about the selected protein. With the information for UniProt, the user is able to interact by clicking on the ID numbers, which will then open a new window with the UniProt database webpage linked to the selected protein.


Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1)

Protein Information

Protein Uniprot ID	O00141
Protein Name	Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1)
Other Names	SGK1 SGK
Gene Name	SGK1_HUMAN
Protein Model Type	threading
Variant	WildType
Protein Sequence Length	431
Protein Sequence	MTVKTEAAKGTLTYSRMRGMVAILIAFMKQRRMGLNDFIQKIANNSYACKHPEVQSILKISQPQEPELMNANPSPPPSPSQQINLGPSSNPHAKPSDFHFLKVIGKGSFGKVLLARHKAEEVFYAVKVLQKKAILKKKEEKHIMSERNVLLKNVKHPFLVGLHFSFQTADKLYFVLDYINGGELFYHLQRERCFLEPRARFYAAEIASALGYLHSLNIVYRDLKPENILLDSQGHIVLTDFGLCKENIEHNSTTSTFCGTPEYLAPEVLHKQPYDRTVDWWCLGAVLYEMLYGLPPFYSRNTAEMYDNILNKPLQLKPNITNSARHLLEGLLQKDRTKRLGAKDDFMEIKSHVFFSLINWDDLINKKITPPFNPNVSGPNDLRHFDPEFTEEPVPNSIGKSPDSVLVTASVKEAAEAFLGFSYAPPTDSFL

HPDD
3D Viewer
Protein Style
Ribbon - colour by structure
Ribbon - colour by position
Tube
Lines
Spheres
Points
Docking results selection
Filters
Selected dockings

Show 10 entries

Search:

Docking ID	Ligand Name	Pubchem ID
1	formoterol	3410
2	gabapentin	3446
3	Imatinib	5291
4	L-thyroxine	5819
5	piperacillin	43672
6	Aripiprazole	60795
7	atorvastatin	60823

Figure 5.25: Protein view page – shown is an image of a view of the protein viewer page for the protein Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1).

3D viewer panel

In the centre of the screen shown in figure 5.25 is the 3D viewer. This viewer is used to show the predicted 3D model of the selected docking. Shown in figure 5.26 is a closeup of the 3D viewer for the protein Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated

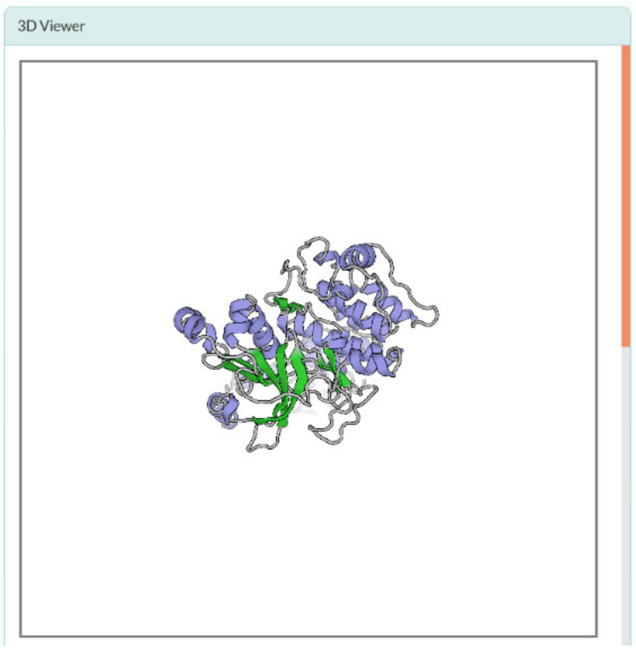


Figure 5.26: Close view of the 3D viewer in the protein viewer page for the protein Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) – shown is an image of the 3D viewer which will be displayed within the protein view page

kinase 1). The user can change the angle at which they view the protein. Shown in table 5.24 are the controls for using the 3D Viewer. The user will also be able to change how the protein looks in the 3D viewer by using option buttons which are on the top right on the page.

Table 5.24: Controls for the 3D viewer – shown are the controls which allow the user to change the view of the object within the 3D viewer.

Action	Control
Rotate	Click and drag with the left mouse button.
Shift sideways	Click and drag with the middle button (often found by pressing down the scroll wheel).
Zoom	Use scroll wheel, up for zoom, down for zoom in.
Re-centre	double click with the left button on a residue or atom. The view will centre on and rotate around the selected location.

Shown in figure 5.27 are the view option buttons for the 3D viewer. Each of these buttons is linked to a type of visual style that can be applied to the protein. Shown in figure 5.28 are the different protein views that can be chosen. Both “Ribbon – colour by structure” and “Ribbon – colour by position” would be selected if the user is interested in viewing the alpha-helix and beta sheets of the structure of a protein. The “Tube” option would be selected if the user would like to view the backbone structure of the protein. The “Lines” option will be selected if the user is interested in viewing the proteins atoms and their bonds. The “Spheres” option would be selected if the user was interested in viewing the protein surface.

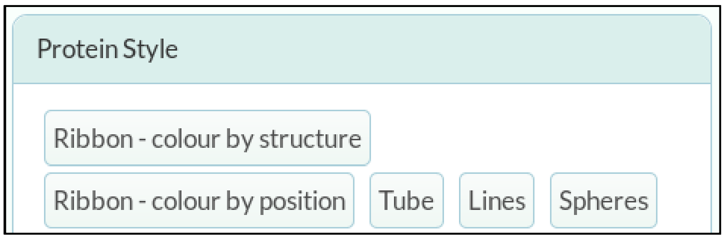


Figure 5.27: option buttons to change the view style of the 3D viewer – shown are the different options available to the user to change the display type of the protein.

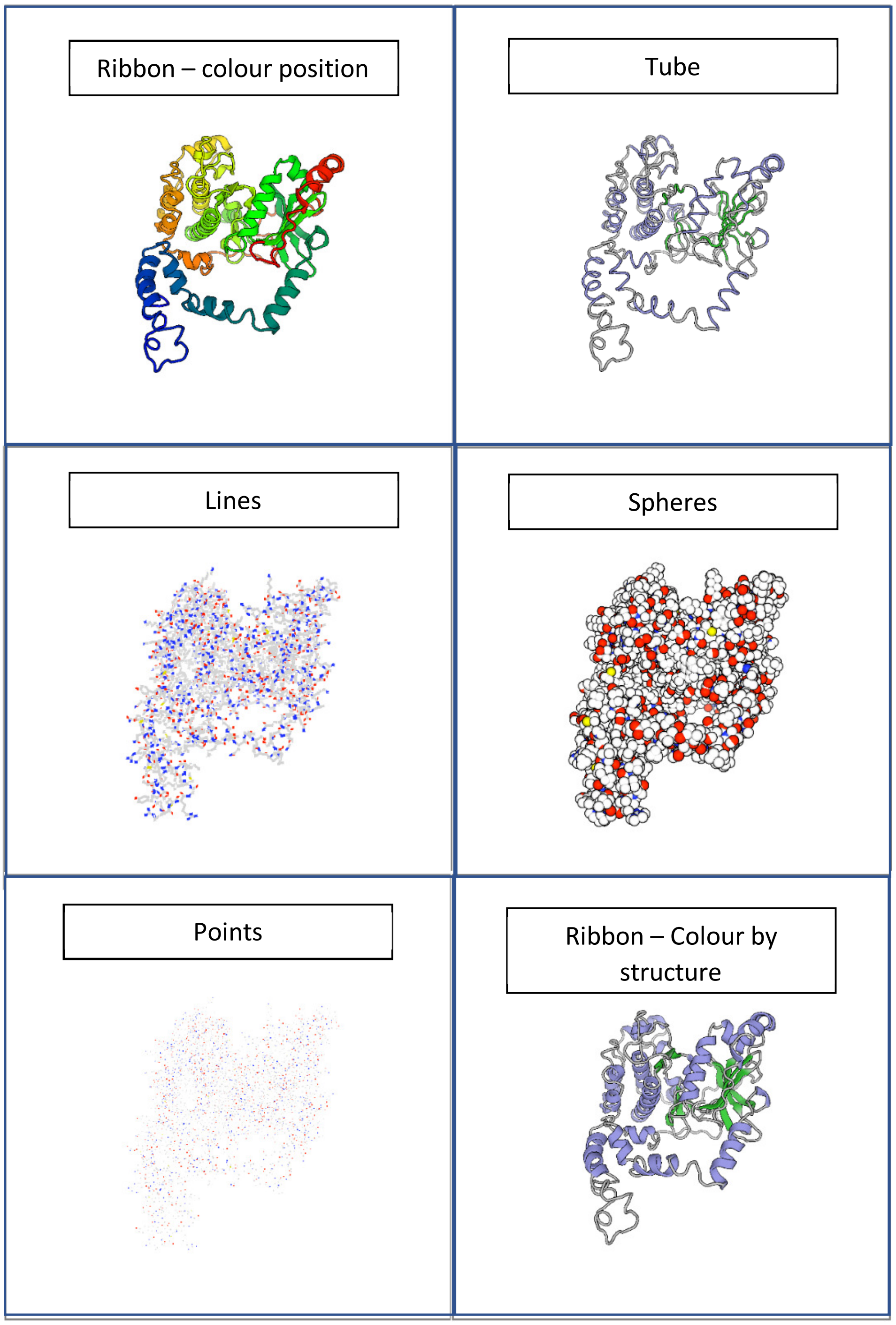
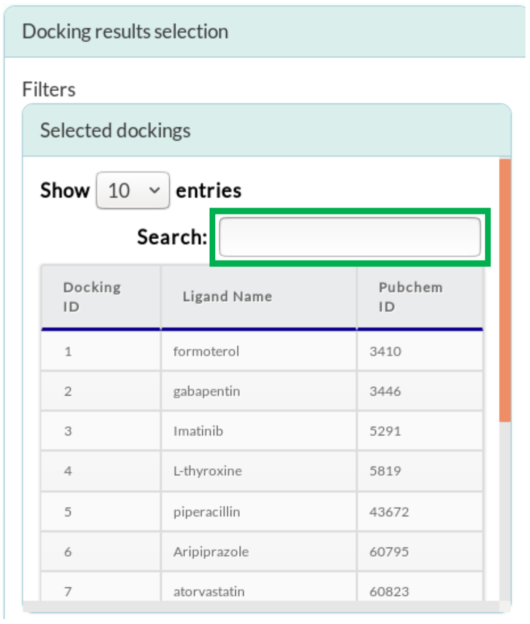


Figure 5.28: 3D Viewer protein display types – shown are Images of the different types of protein views that are available in the Graphical User Interface. the protein views that are shown are Ribbon – colour position, Tube, Lines, Spheres, Points, and Ribbon – Colour by structure.

Docking results selection panel

The last part of the protein viewer page is the docking results selection form. Shown in figure 5.29 is an image of the docking results selection that will be viewable on the protein view webpage. The table will show a list of all drug that has had a docking predicted with the selected protein. The user is


Docking results selection

Filters

Selected dockings

Show 10 entries

Search:

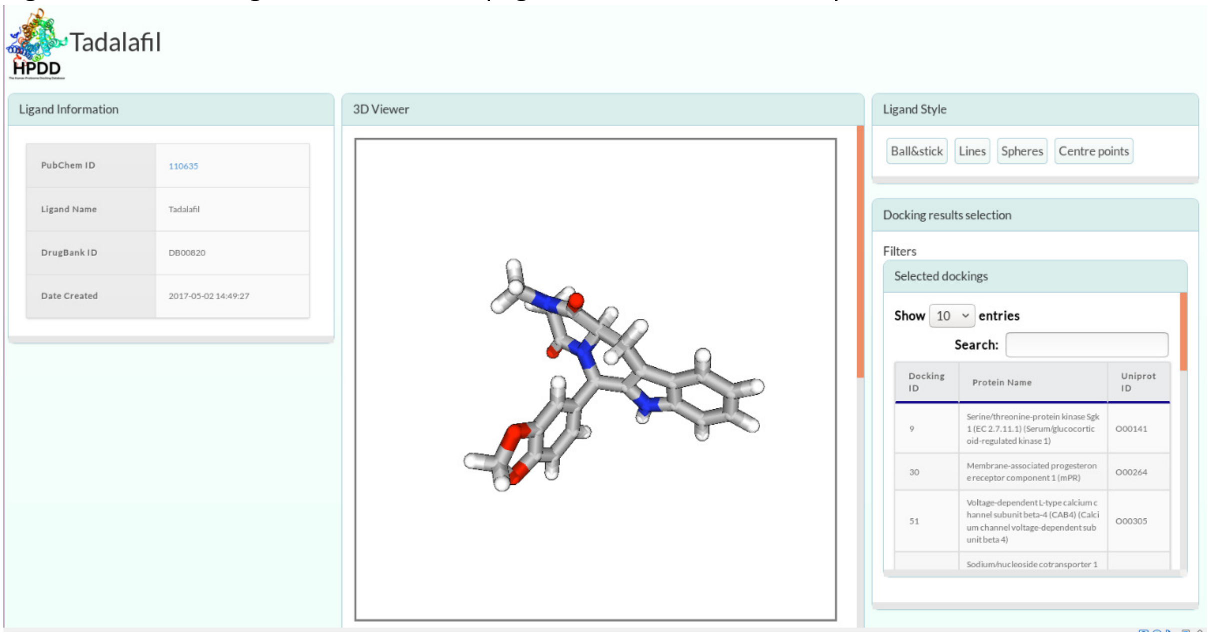
Docking ID	Ligand Name	Pubchem ID
1	formoterol	3410
2	gabapentin	3446
3	Imatinib	5291
4	L-thyroxine	5819
5	piperacillin	43672
6	Aripiprazole	60795
7	atorvastatin	60823

Figure 5.29: Protein view docking selection panel – shown is an image of the docking results selection displayed on the protein view webpage.

also, able to search this list using the search bar shown in the green box of figure 5.29. The user can select the drug of interest from the list, which will then open another window which will display the selected drugs conformations with the protein of interest.

5.4.4 Ligand viewer page

When a user has clicked on a ligand of interest from the loaded list within the ligand selection form on the database page, this action would then open a new window with a dedicated page. Shown in figure 5.30 is an image of the dedicated page that would be viewed by the user.



Ligand Information

PubChem ID	110635
Ligand Name	Tadalafil
DrugBank ID	DB00820
Date Created	2017-05-02 14:49:27

Docking results selection

Filters

Selected dockings

Show 10 entries

Search:

Docking ID	Protein Name	Uniprot ID
9	Serine/threonine-protein kinase Sgk 1 (EC 2.7.11.1) (Serum/glucocortic oid-regulated kinase 1)	O00141
30	Membrane-associated progesteron e receptor component 1 (mPR)	O00264
51	Voltage-dependent L-type calcium c hannel subunit beta-4 (CAB4) (Calci um channel voltage-dependent sub unit beta 4)	O00305
	Sodium/nucleoside cotransporter 1	

Figure 5.30: Ligand view page – shown is an image of the dedicated page for the ligand Tadalafil when selected from the ligand selection panel on the database page.

The ligand shown in figure 5.30 is Tadalafil. The Ligand viewer page has an identical setup to the protein viewer page. There are four panels that are a part of the dedicated ligand viewer page, these being the Ligand Information panel, the 3D viewer panel, Ligand style panel, and Docking results selection panel.

Ligand information panel

The ligand information panel will display all known information about the ligand that is stored in the database. The ligand information that's shown will be the PubChem ID, Ligand Name, DrugBank ID, Reactome ID's, and KEGG ID's. With the information for PubChem and DrugBank, the user is able to interact by clicking on the ID numbers which will then open a new window with the PubChem or DrugBank database webpage linked to the selected ligand.

Ligand 3D Viewer panel

The 3D viewer is the same as the one shown for the protein viewer page. Shown in figure 5.31 is an image of the ligand Tadalafil being shown in the 3D viewer. The only difference is that there are fewer options for the way the ligand is displayed. The options available are Ball & Sticks, Lines, and Sphere views. Shown in figure 5.32 are images of how the ligand will display in the 3D viewer for each view option. Like the 3D viewer on the protein viewer page, the 3D image of the ligand can be turned, zoomed, and the centre point of view can be changed. These functions can be done by using the same control instruction discussed for the protein viewer page's 3D Viewer.

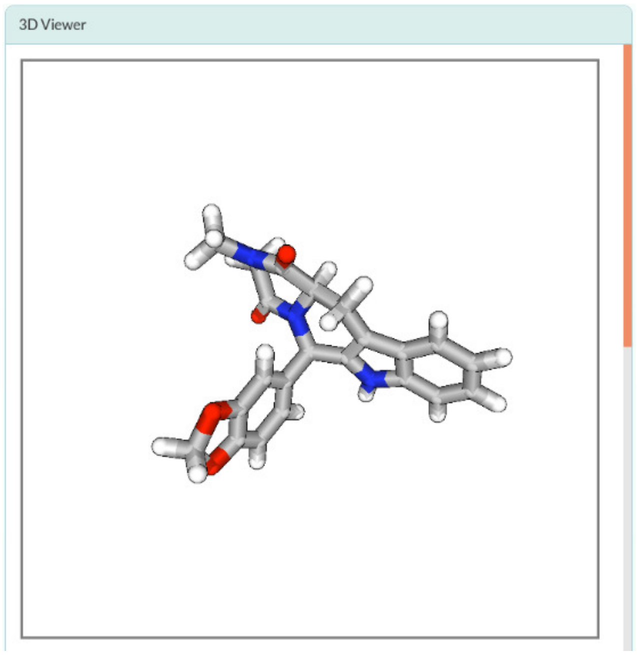


Figure 5.31: Ligand view page 3D viewer – shown is an image of the ligand Tadalafil being shown in the 3D viewer.

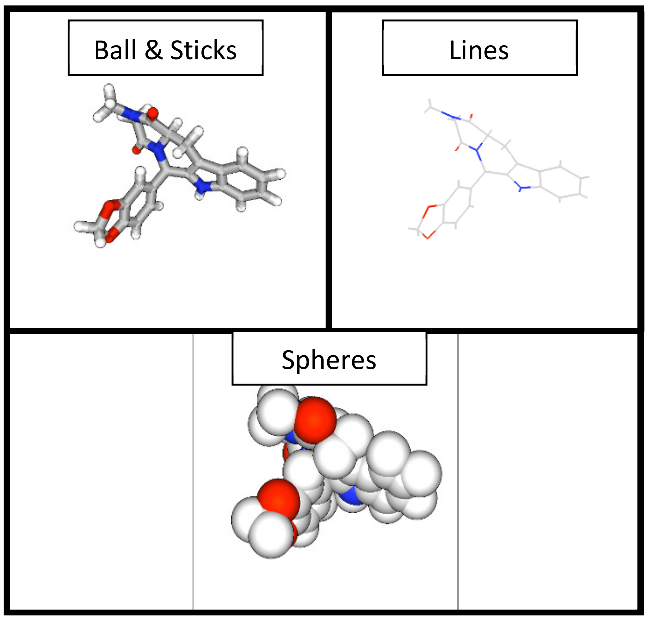
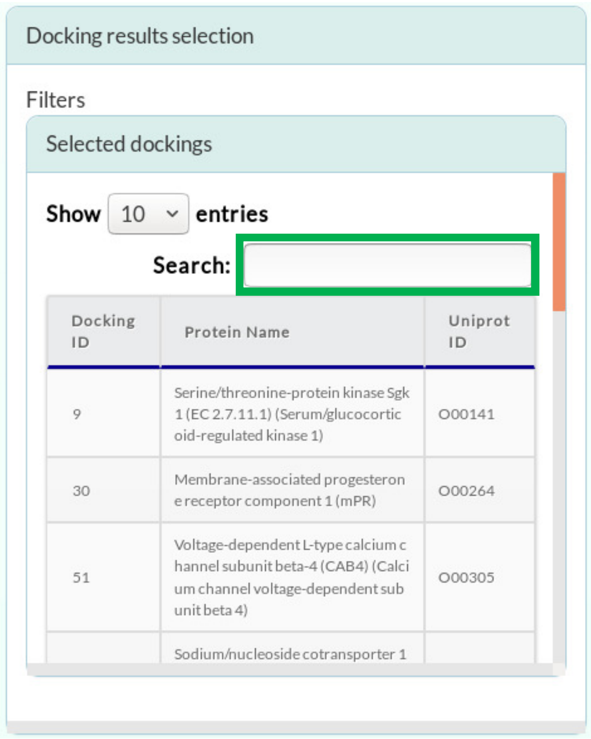


Figure 5.32: 3D Viewer ligand display types – Shown are images of how the ligand can be displayed in the 3D viewer for each display option available. The options are Ball & Sticks, Lines, and Spheres.

Ligand Docking results selection panel

Like the protein viewer page, but instead of ligands, the docking results selection panel will display a list of every protein within the database that has had a conformation predicted with the ligand of interest. Shown in figure 5.33 is an image of the docking results selection panel with a list of proteins



Docking ID	Protein Name	Uniprot ID
9	Serine/threonine-protein kinase Sgk 1 (EC 2.7.11.1) (Serum/glucocortic oid-regulated kinase 1)	O00141
30	Membrane-associated progesteron e receptor component 1 (mPR)	O00264
51	Voltage-dependent L-type calcium c hannel subunit beta-4 (CAB4) (Calci um channel voltage-dependent sub unit beta 4)	O00305
	Sodium/nucleoside cotransporter 1	

Figure 5.33: Ligand view docking selection panel – shown is an image of the docking results selection displayed on the Ligand view webpage for Tadalafil.

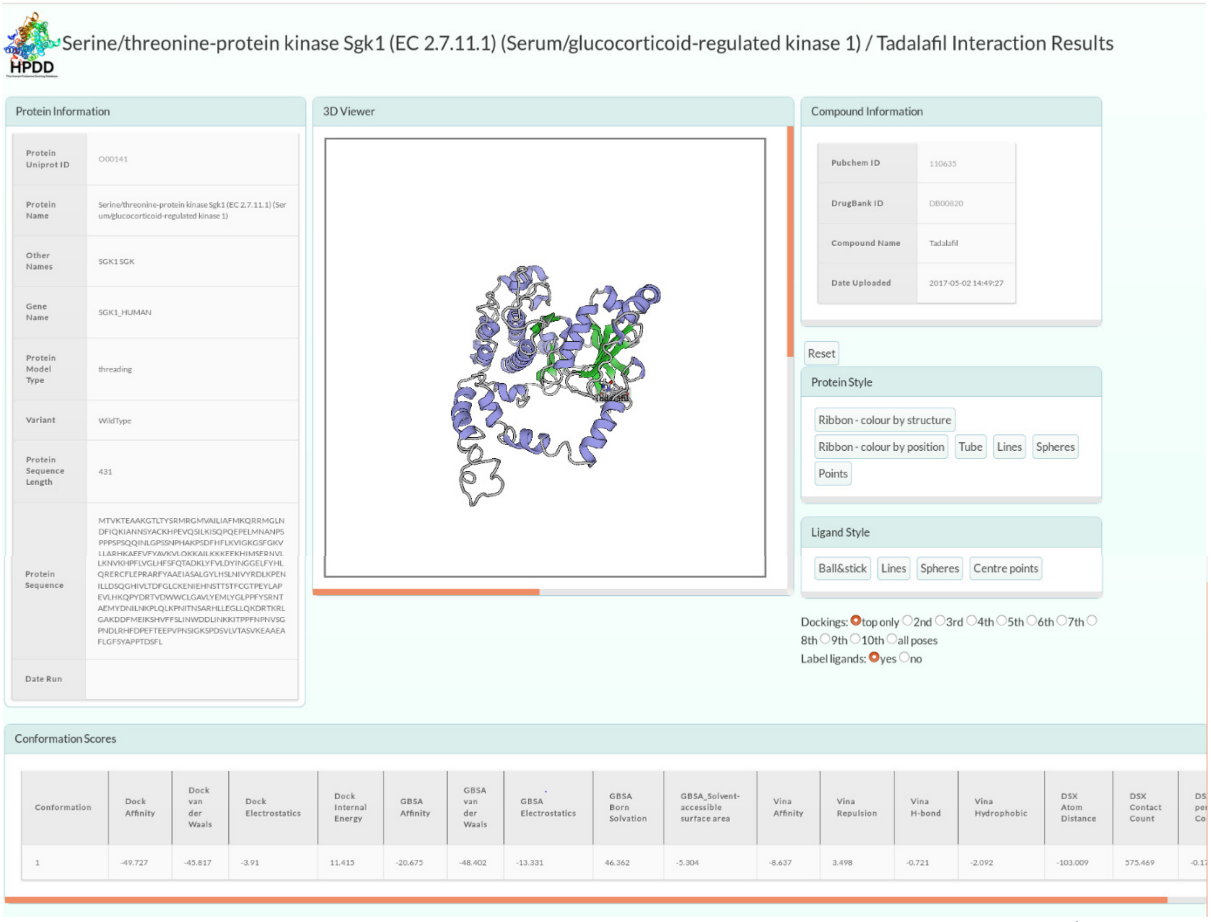
that have had conformations predicted for Tadalafil. The user can search for specific proteins within the list using the search bar shown in the green box in figure 5.33. The user can click one of the proteins in the list, which will then open a new window with the selected protein with the conformation of the ligand of interest.

5.4.5 Docking Viewer Page

The docking viewer page is where the user will be able to view 3D docking conformations between a protein and a ligand. Shown in figure 5.34 is an image of the docking viewer page for the docking pair of Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) and Tadalafil. There are 5 panels that are on the docking viewer page, these being the Protein Information panel, 3D Viewer panel, Ligand Information, Protein style panel, Ligand style panel, and the Conformation scores panel.

Protein Information Panel

Like in the protein viewer page, the protein information panel will display all stored information about the protein within a table. The data shown in the panel are the Protein UniProt ID, Protein Name, Other Names, Gene Names, Protein Model type, Variant, Protein sequence length, Protein sequence, and the Date run.



Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) / Tadalafil Interaction Results

Protein Information

Protein Uniprot ID	O00141
Protein Name	Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1)
Other Names	SGK1 SGK
Gene Name	SGK1_HUMAN
Protein Model Type	threading
Variant	WildType
Protein Sequence Length	431
Protein Sequence	MTVKTEAAKGTLTYSRMRGMVAILIAFMKQRRMGLN DFIQKIANNSYACKHPEVQSILKISQPQEPELMNANPS PPPSPSQQINLGPSSNPHAKPSDFHFLKVIGKGSFGKV LLARHKAEEVFYAVKVLQKKAILKKKEEKHIMSERNVL LKNVKHPFLVGLHFSFQTADKLYFVLDYINGGELFYHL QRERCFLEPRARFYAAEIASALGYLHSLNIVYRDLKPEN ILLDSQGHIVLTDFGLCKENIEHNSTTSTFCGTPEYLAP EVLHKQPYDRTVDWWCLGAVLYEMLYGLPPFYSRNT AEMYDNILNKPLQLKPNITNSARHLLEGLLQKDRTKRL GAKDDFMEIKSHVFFSLINWDDLINKKITPPFNPNVSG PNDLRHFDPEFTEEPVPNSIGKSPDSVLVTASVKEAAEA FLGFSYAPPTDSFL
Date Run	

3D Viewer

Compound Information

Pubchem ID	110635
DrugBank ID	DB00820
Compound Name	Tadalafil
Date Uploaded	2017-05-02 14:49:27

Dockings: [✓ top only] 2nd 3rd 4th 5th 6th 7th 8th 9th 10th all poses
Label ligands: [✓ yes] no

Conformation Scores

Conformation	Dock Affinity	Dock van der Waals	Dock Electrostatics	Dock Internal Energy	GBSA Affinity	GBSA van der Waals	GBSA Electrostatics	GBSA Born Solvation	GBSA_Solvent-accessible surface area	Vina Affinity	Vina Repulsion	Vina H-bond	Vina Hydrophobic	DSX Atom Distance	DSX Contact Count	DS pe Co
1	-49.727	-45.817	-3.91	11.415	-20.675	-48.402	-13.331	46.362	-5.304	-8.637	3.498	-0.721	-2.092	-103.009	575.469	-0.1

Figure 5.34: Docking view page – shown is an image of the docking viewer page for the docking pair of Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) and Tadalafil.

3D Viewer panel

The 3D viewer panel will display the 3D conformations of the selected protein and ligand docking pair. Shown in figure 5.35 is an image of the top-ranked conformation between Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) and Tadalafil in the 3D viewer. Like the 3D viewer used in the protein viewer page, which is discussed in section 5.4.3, it will show the generated 3D model for the protein which is a part of the selected docking pair. The difference being that also loaded in the 3D viewer will be the top-ranked predicted conformation for the ligand which is a part of the selected docking pair. The same controls are used as discussed in section 5.4.3 as to how to manipulate the view of the conformation shown in the 3D viewer. However, the 3D viewer on the docking viewer page has two extra features. The first extra feature is if the user hovers the mouse over any of the protein or the drug atoms within the 3D viewer, the text will appear below the viewer identifying residue. Shown in figure 5.36 is an image of the 3D viewer with an atom of the protein highlighted and it is identified by text shown in the red box.

The second extra feature is the ability for the user to measure distances between two atoms. To measure a distance between two atoms, firstly by using the mouse, click the first atom of interest within the 3D viewer. Secondly, click on the second atom of interest within the 3D viewer. When both atoms have been selected, the distance is calculated and displayed within the viewer. Angstroms is the unit of measurement for all distances. This feature can be handy for the user as they are able to find the closest interacting atoms with the bound ligand. With both additional features, it allows the user to do an essential analysis of the docking pair conformations.

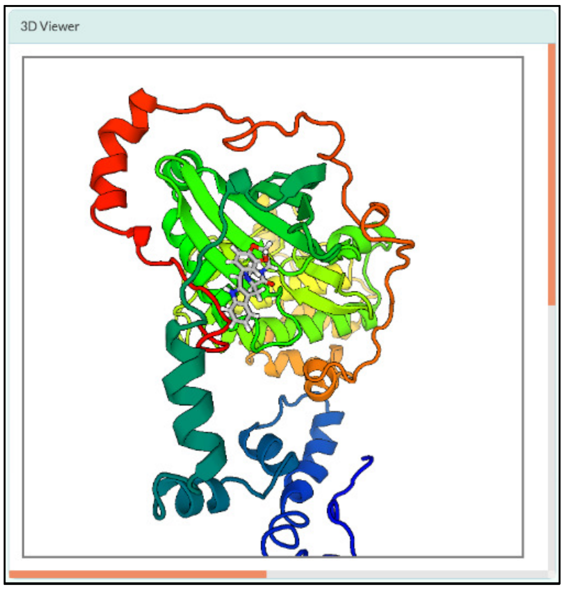


Figure 5.35:Docking page 3D Viewer – shown is an image of the top-ranked conformation between Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) (Ribbon – colour by position protein view) and Tadalafil (Ball & stick ligand view) in the 3D viewer.

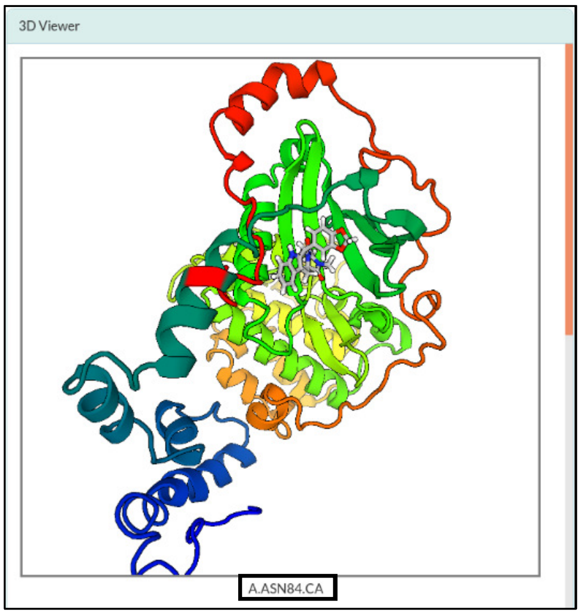


Figure 5.36: Docking page 3D viewer amino acid Identifier – shown is an image of the 3D viewer with a protein atom highlighted which is being identified by an atom identifier by text shown in the red box.

Protein Style Panel

Like the protein style panel on the protein viewer page, the protein style panel is used to change the view type of the protein that is visualised in the 3D viewer. Shown in figure 5.27 are the options available and shown in figure 5.28 are how the different view options are displayed.

Ligand Style Panel

Like the ligand style panel on the ligand viewer page, this panel is used to change the view type of the ligand that is visualised in the 3D viewer. Shown in figure 5.32 are how the different types of view that are available are displayed.

Ligand Information Panel

Like the ligand viewer page, the ligand information panel displays all stored information about the drug within the selected docking pair. The information that is displayed is the PubChem ID, DrugBank ID, Ligand Name, KEGG ID's, Reactome ID's, and the Date Uploaded.

Conformation Options panel

Below the Ligand style panel are options for displaying different conformations. The docking program predicts the 10 best conformations and ranks them from 1 to 10. These options give the user the ability to examine all top 10 conformations that were predicted by the docking. programs from 1 to 10.

Shown in figure 5.37 is an image of the different conformation options that can be selected. When a radio button is selected, the similar conformation will then be displayed on the 3D viewer at the same time. Shown in figure 5.38 is an image of all 10 predicted conformations for the docking pair of Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) and Tadalafil displayed in the 3D viewer. This view can show the user all the areas of the protein that the docking programs found possible docking "pockets" to dock a ligand. The other with the selected protein. One of the options is to display every generated conformation for the docking pair within the 3D options shown in figure 5.37 and to also label the ligands displayed in the 3D viewer. Shown in figure 5.38 is an image of the Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) and Tadalafil docking with the label "Tadalafil" attached to the ligand. These labels will help the user to identify which conformations are being displayed, and where, on the 3D viewer.

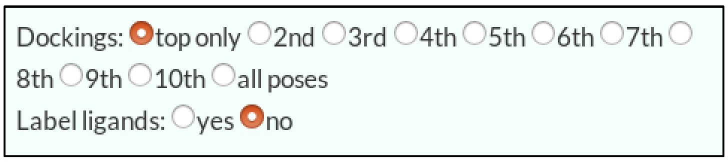

Dockings: [✓ top only] 2nd 3rd 4th 5th 6th 7th 8th 9th 10th all poses
Label ligands: yes [✓ no]

Figure 5.37:Docking view page conformation selector – shown is an image of the conformation options that are available to display on the 3D viewer within the docking viewer page.

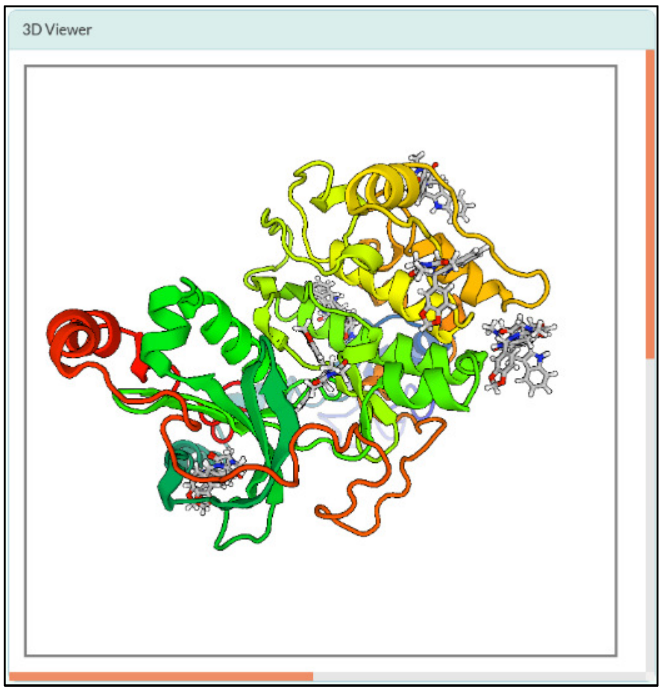
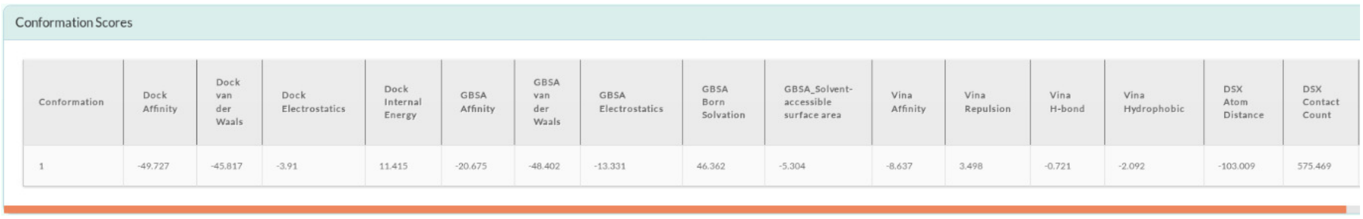


Figure 5.38: docking view page 3D viewer with all 10 conformations – shown is an image of all 10 predicted conformations for the docking pair of Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) and Tadalafil displayed in the 3D viewer.

Conformation scores panel

The conformation scores panel is used to display all stored scores related to the selected docking pair. Shown in figure 5.39 is an image of conformation scores for the top-ranked conformation for the docking pair of Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) and Tadalafil. The conformation scores panel is also linked to the conformation options panel. Whichever conformation that has been selected to be viewed on in the 3D viewer, only those scores related to that conformation will be displayed. The user can see all conformations scores at once by selecting to view all poses in the conformation options panel shown in figure 5.37, all 10 scores will be displayed within the conformation scores panel.


Conformation Scores

Conformation	Dock Affinity	Dock van der Waals	Dock Electrostatics	Dock Internal Energy	GBSA Affinity	GBSA van der Waals	GBSA Electrostatics	GBSA Born Solvation	GBSA_Solvent-accessible surface area	Vina Affinity	Vina Repulsion	Vina H-bond	Vina Hydrophobic	DSX Atom Distance	DSX Contact Count
1	-49.727	-45.817	-3.91	11.415	-20.675	-48.402	-13.331	46.362	-5.304	-8.637	3.498	-0.721	-2.092	-103.009	575.469

Figure 5.39: Docking view page conformation scores table – shown is an image of conformation scores for the top-ranked conformation for the docking pair of Serine/threonine-protein kinase Sgk1 (EC 2.7.11.1) (Serum/glucocorticoid-regulated kinase 1) and Tadalafil.

5.5 Conclusion

The graphical user interface implemented has made it possible to visualise any of the stored structures and dockings in the extensive database of protein-ligands dockings. It would be an onerous task for an individual to go through and search for particular dockings within these huge set, but this intuitive graphical user interface has made it possible for any protein-ligand docking pair to be searched for and have the results shown in a user-friendly graphical format.

The big advantage of the developed system is the great flexibility of options provided to the user. If a user requires information about a protein of interest, the system provides detailed information about the protein in isolation (one protein to many ligands). This is illustrated in figure 5.25. Another user might require information about a specific drug of interest, and in this case the system would provide detailed information about the drug (one ligand to many proteins). This is illustrated in figure 5.30. In a more complex application, the user might be interested in an interaction between any protein in a group of proteins, considering all the potential drugs of interest. The system provides detailed information about the predicted dockings with options to analyse the interactions further to effectively “mine” for new leads.

This compares very favourably with the situation beforehand, where if a user wanted to investigate a protein-ligand docking pair of interest manually, they would first have to find the docking pair conformation in the vast array of folders that store the results and then to display them by export to a program like UCSF Chimera (Pettersen et al., 2004). If they wished to examine the predicted scores and outcome for that docking pair, they would have to explore a vast data file which in turn contains different ranked conformation for each docking pair, which can be impeding. This kind of search would take a long time to do, particularly for most programmes of work, which need to consider multiple protein-ligand pairs. The current Graphical User Interface linked with the developed system architecture has created a seamless, easy to use system that can enable users to investigate a vast array of valuable data in a fraction of the time spent with manual analysis.

6 Discussion

6.1 Overview

This study has incorporated the use of supercomputing, database and web development technologies, and *in silico* docking approaches to develop a vast database of 3D models of the human proteome with predicted binding conformation data against the 20 top-selling drugs in the UK, which are all able to be visualised on a graphical user interface for the purpose of aiding research in drug discovery. The intention of this research was to address the need for a comprehensive interactive resource which would aid the development and repurposing of drugs. There was no such resource in existence that addressed this need. The system gives access to a vast source of docking data that can be interrogated by users in order to answer specific questions about drug interactions. The aim was to develop a system which is readily adaptable to various areas of research within the drug development field. This involved developing a system architecture which answers these specific needs. The system had to efficiently handle a vast amount of data generated from protein modelling and molecular docking. The system has proven to be a useful and broadly accurate tool with the development of a new docking analysis methodology which gives an overall accuracy of 72.05%, which is encouraging for further development.

With the development of NGS and the cost of screening individual genome steadily decreasing, the advancement of personalised medicine is at the forefront of research. Personalised medicine has the potential to transform healthcare including reducing the number of patients being prescribed less suitable drugs for them that could result in non-response or cause side effects (Mathur and Sutton, 2017). The system that has been developed here can help with first steps of identifying possible protein target variants which modify how drugs are metabolised and identify potential off-target interactions which could cause adverse drug events and patient illness. With the accuracy that was achieved by the combinational forecasting approach developed, the system can reduce the time and costs involved in identifying these instances. The developed database and system prototype have the potential to change the way drugs are developed in the drug discovery sector by facilitating rapid investigation, classification and comparison of predicted protein-ligand interactions across the whole human proteome.The system is a powerful tool which can be used to determine if a ligand has scope for specific interaction with a protein of interest before deciding to further investigate in the laboratory. With all of the available docking data and 3D viewer to display and enable the user to analyse the protein, ligand and conformations, the system has the potential to identify new leads or establish opportunities for repurposing existing drugs. Drug development is a risky and costly affair (Mohs and Greig, 2017) which this system has the ability to reduce by the vast amount of data and analysis available, to systematically narrow down potential targets to investigate. The system can contribute towards the advancement of personalised medicine with incorporation of further knowledge of protein interactions and the impacts of protein variation.

The cost of developing a new drug is very expensive, which has led to an increase in drug repurposing (repositioning). Pharmaceutical companies turn to this as it is a means of appropriating drugs faster for clinical use. Drug developers are increasingly looking into the use of computational techniques which can help aid them in the process (Xue et al., 2018). The developed system reported in this thesis is a tool which can be applied to both the repurposing case and discovery of new targets for existing drugs currently in the healthcare system. It has the ability to help identify ligands that can be linked with the treatment of another illness or disease by virtue of a predicted interaction with a protein of interest. The system can help pharmaceutical researchers expedite the repurposing process even further by the vast scale visualisation and analysis of stored conformations.

6.2 Applications for the system

The system can be used to help answer questions such as Investigate protein structures, Investigation of drugs for repurposing, Investigating drug “off-target” hits, The potential for investigating the interactions of “new” ligands.

6.3 Limitations of this study

The main practical limitation of this study was the sporadic availability of HPC computational resources. Due to HPC Wales having more than 500 users, at peak usage times there would be long queues which hampered the progression rate for homology modelling, protein threading and docking. There were also issues with HPC Wales occasionally going offline altogether which affected progress. This mainly affected the protein threading as this is the longest process to complete. Depending on the length of the sequence that has been inputted to be modelled, it can take up to 60 hours to complete a single 3D model.

The docking algorithms used do not allow for proteins movement, which in turn can affect docking results. This is due to that protein movement can change the shape of the proteins pockets which can alter how a ligand interacts. Updates in the future might include the incorporation of docking algorithms which allow for side-chain flexibility. As computer hardware gets faster, eventually it could be possible to apply molecular dynamics simulations in the same high-throughput way as we currently apply docking.

6.4 Future Work

6.4.1 System architecture

The future work that will be carried out on the system architecture is to incorporate the meta-score from the developed combinational forecasting model within the database so that it can give users an initial idea as to whether the docking has been classed as a possible interaction or a non-interaction. This will be undertaken by introducing another table called “Data_analysis” which would store data on each docking pair as either a possible interaction or a non-interaction. This would be linked to the docking table via the docking ID. An example of how the new database structure would look is shown in figure 6.1. There would be another column which would have either “TRUE” or “FALSE” within it, if it is true it means that the docking is a potential interaction and if false it is a non-interaction. By having this added it would make it possible to display the results on the screen to the user at the point of when they have first selected a protein-ligand docking pair to view. This database structure could help users to determine earlier if it might be worth investigating a specific interaction further.

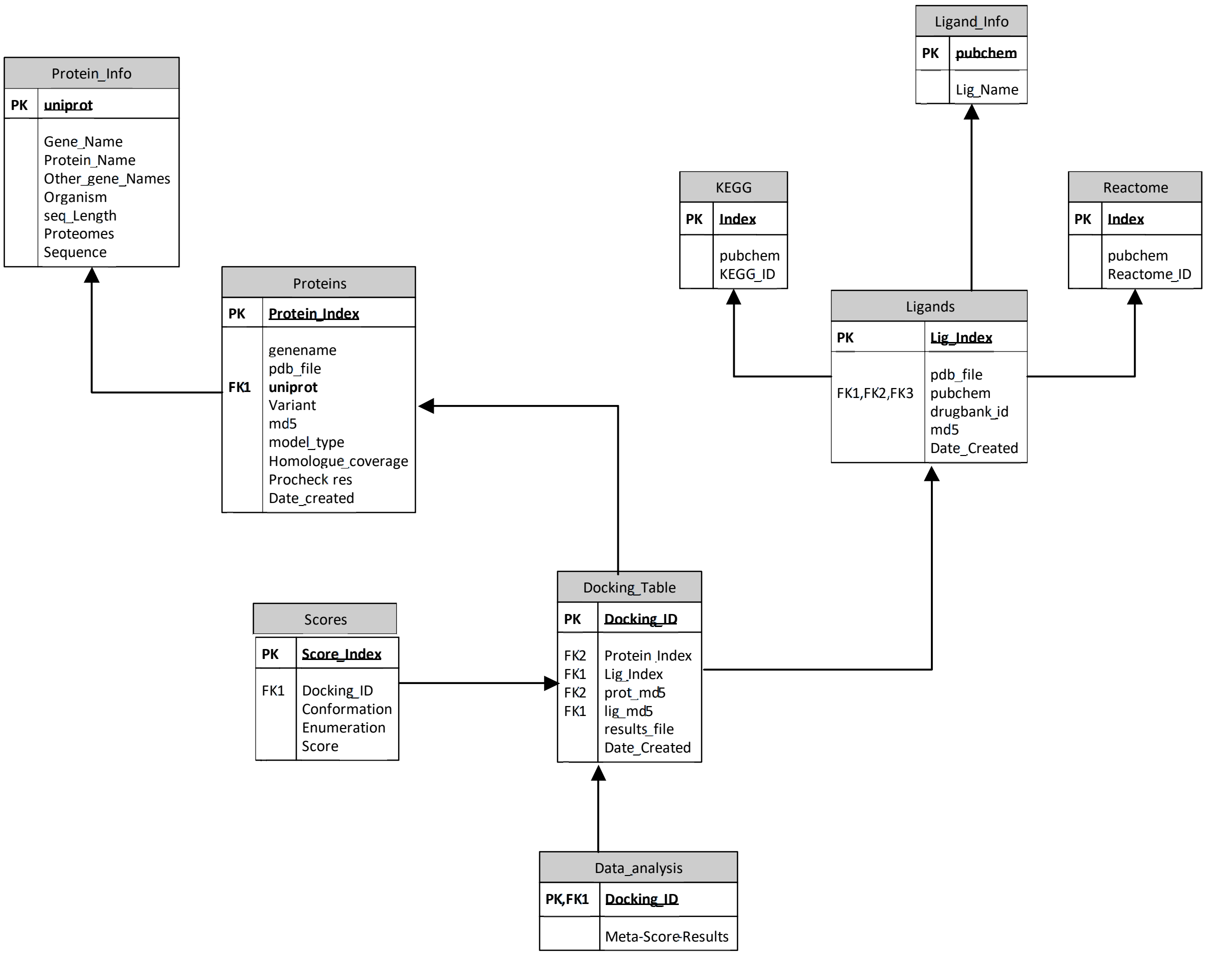


Figure 6.1: Entity relationship diagram for future improvements of the database structure – shown is an ERD diagram which displays the future improvement of the database by the addition of the Data_analysis table to store combinational forecasting results.

Another addition to the database structure will be to add another table, linked to the proteins table, that ranks the quality/structural coverage of the 3D protein model. The quality of a model structure would be reported by the structure coverage script described in the protein modelling and molecular docking chapter. The model structure quality will be ranked as 80%+ - GOLD, between 50% and 80% - Silver, and between 1% and 50% - Bronze. As there will be a mixture of homology models and threaded protein models, this would let the user know beforehand whether the resulting docking is more likely to be of good accuracy rather than following up on a low-quality model.

6.4.2 Protein Modelling

The protein modelling reported in the thesis was to some extent complicated by having to apply both homology modelling and protein threading. A potential way of moving the research forward is to attempt to combine homology modelling and protein threading in one modelling methodology. Homology modelling can generate a model if the sequence has 30% or more homologue coverage. Protein threading can generate a model with a sequence with as low as 15%. Combining the two methods holds some potential for “filling in the gaps” of homology modelling in terms of cross-proteome and intra-protein coverage. This may not be easily done due to the two modelling pipelines using different methods but is an idea to investigate.

6.4.3 Protein-Ligand Molecular Docking

The molecular docking analysis gave a good accuracy of 72.05%, but this can be further improved. A method can be applied to the protein models before the docking process which could utilise biological data to better define the search space explored by the docking algorithms. The method could help reduce the number of false-positive docking results where a ligand is predicted to interact in a location which is not biologically feasible in life and increase the number of correctly predicted dockings in known active sites in proteins for which some functional data is known.

6.4.4 Graphical User Interface

There is also a need to improve the way that users are able to save search results and lists for use in future sessions. A modification can be added to the database; an indicator applied to the docking view page which can clearly display to the user if the system deems the selected conformation a potential interaction. The indicator would provide the scope to save time in the search for new leads from the vast amount of data stored.

6.4.5 Database Expansion

The next stage of development will be the addition of the protein-ligand dockings of variant forms of proteins against the 20 top-selling drugs in the UK. This has the potential of the addition of

16,000,000 new data points and could be of use in assessing the potential for adverse drug events across populations due to polymorphic variation.

Another likely addition will be to incorporate the full FDA approved drug set with additional dockings with the proteins stored within the database, which will provide great potential for pursuing investigations of repurposing opportunities.

6.5 Concluding remarks

No other system has been developed that can be used to visualise and analyse such an extensive database of hundreds of thousands of protein-ligand interactions in such a comprehensive, user-friendly way. The developed system has the potential to easily increase the data stored to include more drugs such as FDA approved to also have the potential to analyse newly developed drugs. This will be an important part of its future development.

The work carried out in this project has contributed towards the development of a comprehensive *in silico* platform, Human3DProteome (human.3dproteome.swan.ac.uk), that utilises the system architecture, methodologies, data and methods of data analysis advanced in this project.

The system is freely available at: http://proteins.swan.ac.uk/hpdd

7 References

Abou-Khalil, B. (2008). Levetiracetam in the treatment of epilepsy. *Neuropsychiatric Disease and Treatment*, 4(3), pp.507-523.

Allen, W., Balius, T., Mukherjee, S., Brozell, S., Moustakas, D., Lang, P., Case, D., Kuntz, I. and Rizzo, R. (2015). DOCK 6: Impact of new features and current docking performance. *Journal of Computational Chemistry*, 36(15), pp.1132-1156.

Altschul, S., Gish, W., Miller, W., Myers, E. and Lipman, D. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, 215(3), pp.403-410.

Andersson, O., Cassel, T., Grönneberg, R., Brönnegård, M., Stierna, P. and Nord, M. (1999). In vivo modulation of glucocorticoid receptor mRNA by inhaled fluticasone propionate in bronchial mucosa and blood lymphocytes in subjects with mild asthma. *Journal of Allergy and Clinical Immunology*, 103(4), pp.595-600.

Armougom, F., Moretti, S., Poirot, O., Audic, S., Dumas, P., Schaeli, B., Keduas, V. and Notredame, C. (2006). Expresso: automatic incorporation of structural information in multiple sequence alignments using 3D-Coffee. *Nucleic Acids Research*, 34(Web Server), pp.W604-W608.

Backonja, M. and Glanzman, R. (2003). Gabapentin dosing for neuropathic pain: Evidence from randomized, placebo-controlled clinical trials. *Clinical Therapeutics*, 25(1), pp.81-104.

Bairoch, A. (2000). The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Research*, 28(1), pp.45-48.

Baldwin, D. and Ajel, K. (2007). Role of pregabalin in the treatment of generalized anxiety disorder. *Neuropsychiatric Disease and Treatment*, 3(2), pp.185-191.

Barker, W. (2000). The Protein Information Resource (PIR). *Nucleic Acids Research*, 28(1), pp.41-44.

Basra, R. and Kelleher, C. (2008). A review of solifenacin in the treatment of urinary incontinence. *Therapeutics and Clinical Risk Management*, 4(1), pp.117-128.

Benson, D., Cavanaugh, M., Clark, K., Karsch-Mizrachi, I., Lipman, D., Ostell, J. and Sayers, E. (2012). GenBank. *Nucleic Acids Research*, 41(D1), pp.D36-D42.

Bernstein, F., Koetzle, T., Williams, G., Meyer, E., Brice, M., Rodgers, J., Kennard, O., Shimanouchi, T. and Tasumi, M. (1978). The protein data bank: A computer-based archival file for macromolecular structures. *Archives of Biochemistry and Biophysics*, 185(2), pp.584-591.

Bowes, J., Brown, A., Hamon, J., Jarolimek, W., Sridhar, A., Waldron, G. and Whitebread, S. (2012). Reducing safety-related drug attrition: the use of in vitro pharmacological profiling. *Nature Reviews Drug Discovery*, 11(12), pp.909-922.

Bowes, J., Brown, A., Hamon, J., Jarolimek, W., Sridhar, A., Waldron, G. and Whitebread, S. (2012). Reducing safety-related drug attrition: the use of in vitro pharmacological profiling. *Nature Reviews Drug Discovery*, 11(12), pp.909-922.

Brändén, C. and Tooze, J. (2009). *Introduction to protein structure*. New York, NY: Garland Pub.

Buchan, D., Minneci, F., Nugent, T., Bryson, K. and Jones, D. (2013). Scalable web services for the PSIPRED Protein Analysis Workbench. *Nucleic Acids Research*, 41(W1), pp.W349-W357.

Capuccini, M., Ahmed, L., Schaal, W., Laure, E. and Spjuth, O. (2017). Large-scale virtual screening on public cloud resources with Apache Spark. *Journal of Cheminformatics*, 9(1).

Chem.ucla.edu. (2018). [online] Available at: http://www.chem.ucla.edu/~harding/ec_tutorials/tutorial73.pdf [Accessed 6 Mar. 2018].

Chen, C., Baldassarre, F., Kanjeekal, S., Herst, J., Hicks, L. and Cheung, M. (2013). Lenalidomide in multiple myeloma – a practice guideline. *Current Oncology*, 20(2).

Chen, Y. (2015). Erratum: Beware of Docking!. *Trends in Pharmacological Sciences*, 36(9), p.617.

Cheng, T., Li, Q., Zhou, Z., Wang, Y. and Bryant, S. (2012). Structure-Based Virtual Screening for Drug Discovery: a Problem-Centric Review. *The AAPS Journal*, 14(1), pp.133-141.

Coward, R. and Carson, C. (2018). Tadalafil in the treatment of erectile dysfunction. *Therapeutics and Clinical Risk Management*, 4(6), pp.1315-1330.

Dalby, A., Nourse, J., Hounshell, W., Gushurst, A., Grier, D., Leland, B. and Laufer, J. (1992). Description of several chemical structure file formats used by computer programs developed at Molecular Design Limited. *Journal of Chemical Information and Modeling*, 32(3), pp.244-255.

Database Resources of the National Center for Biotechnology Information. (2016). *Nucleic Acids Research*, 45(D1), pp.D12-D17.

de Bono, J., Logothetis, C., Molina, A., Fizazi, K., North, S., Chu, L., Chi, K., Jones, R., Goodman, O., Saad, F., Staffurth, J., Mainwaring, P., Harland, S., Flaig, T., Hutson, T., Cheng, T., Patterson, H., Hainsworth, J., Ryan, C., Sternberg, C., Ellard, S., Fléchon, A., Saleh, M., Scholz, M., Efstathiou, E., Zivi, A., Bianchini, D., Loriot, Y., Chieffo, N., Kheoh, T., Haqq, C. and Scher, H. (2011). Abiraterone and Increased Survival in Metastatic Prostate Cancer. *New England Journal of Medicine*, 364(21), pp.1995-2005.

Dock.compbio.ucsf.edu. (2018). *DOCK 6.8 User Manual*. [online] Available at: http://dock.compbio.ucsf.edu/DOCK_6/dock6_manual.htm [Accessed 7 Apr. 2017].

Dock.compbio.ucsf.edu. (2018). *Tutorial: Generating Spheres*. [online] Available at: http://dock.compbio.ucsf.edu/DOCK_6/tutorials/sphere_generation/generating_spheres.htm [Accessed 10 Jan. 2016].

Don, C. and Smieško, M. (2018). Microsecond MD simulations of human CYP2D6 wild-type and five allelic variants reveal mechanistic insights on the function. *PLOS ONE*, 13(8), p.e0202534.

White, A., Modi, S. (2018). *In silico ADMET models: is the future really bright? - Drug Discovery Today*. [online] Drugdiscoverytoday.com. Available at: http://www.drugdiscoverytoday.com/view/24705/in-silico-admet-models-is-the-future-really-bright/ [Accessed 15 Mar. 2016].

Fidler, D., Murphy, S., Courtis, K., Antonoudiou, P., El-Tohamy, R., Ient, J. and Levine, T. (2016). Using HHsearch to tackle proteins of unknown function: A pilot study with PH domains. *Traffic*, 17(11), pp.1214-1226.

Flynn, M., Heale, K. and Alisaraie, L. (2017). Mechanism of Off-Target Interactions and Toxicity of Tamoxifen and Its Metabolites. *Chemical Research in Toxicology*, 30(7), pp.1492-1507.

Fung, H., Stone, E. and Piacenti, F. (2002). Tenofovir disoproxil fumarate: A nucleotide reverse transcriptase inhibitor for the treatment of HIV infection. *Clinical Therapeutics*, 24(10), pp.1515-1548.

Genheden, S. and Ryde, U. (2015). The MM/PBSA and MM/GBSA methods to estimate ligand-binding affinities. *Expert Opinion on Drug Discovery*, 10(5), pp.449-461.

Giagounidis, A., Germing, U., Haase, S. and Aul, C. (2007). Lenalidomide: a brief review of its therapeutic potential in myelodysplastic syndromes. *Therapeutics and Clinical Risk Management*, 3(4), pp.553-562.

Hastings, J., Owen, G., Dekker, A., Ennis, M., Kale, N., Muthukrishnan, V., Turner, S., Swainston, N., Mendes, P. and Steinbeck, C. (2015). ChEBI in 2016: Improved services and an expanding collection of metabolites. *Nucleic Acids Research*, 44(D1), pp.D1214-D1219.

Hatherley, R., Brown, D., Glenister, M. and Tastan Bishop, Ö. (2016). PRIMO: An Interactive Homology Modeling Pipeline. *PLOS ONE*, 11(11), p.e0166698.

Health.gov. (2018). *Overview - Adverse Drug Events - health.gov*. [online] Available at: https://health.gov/hcq/ade.asp [Accessed 20 May 2018].

Hodgson, D., Mortimer, K. and Harrison, T. (2010). Budesonide/formoterol in the treatment of asthma. *Expert Review of Respiratory Medicine*, 4(5), pp.557-566.

Ishchenko, A., Wacker, D., Kapoor, M., Zhang, A., Han, G., Basu, S., Patel, N., Messerschmidt, M., Weierstall, U., Liu, W., Katritch, V., Roth, B., Stevens, R. and Cherezov, V. (2017). Structural insights into the extracellular recognition of the human serotonin 2B receptor by an antibody. *Proceedings of the National Academy of Sciences*, 114(31), pp.8223-8228.

Javed, Z. and Sathyapalan, T. (2015). Levothyroxine treatment of mild subclinical hypothyroidism: a review of potential risks and benefits. *Therapeutic Advances in Endocrinology and Metabolism*, 7(1), pp.12-23.

Jones, G., Willett, P., Glen, R., Leach, A. and Taylor, R. (1997). Development and validation of a genetic algorithm for flexible docking 1 1Edited by F. E. Cohen. *Journal of Molecular Biology*, 267(3), pp.727-748.

Källberg, M., Wang, H., Wang, S., Peng, J., Wang, Z., Lu, H. and Xu, J. (2012). Template-based protein structure modeling using the RaptorX web server. *Nature Protocols*, 7(8), pp.1511-1522.

Katsila, T., Spyroulias, G., Patrinos, G. and Matsoukas, M. (2016). Computational approaches in target identification and drug discovery. *Computational and Structural Biotechnology Journal*, 14, pp.177-184.

Kim, D., Xu, D., Guo, J., Ellrott, K. and Xu, Y. (2003). PROSPECT II: protein structure prediction program for genome-scale applications. *Protein Engineering Design and Selection*, 16(9), pp.641-650.

Kim, S., Thiessen, P., Bolton, E., Chen, J., Fu, G., Gindulyte, A., Han, L., He, J., He, S., Shoemaker, B., Wang, J., Yu, B., Zhang, J. and Bryant, S. (2015). PubChem Substance and Compound databases. *Nucleic Acids Research*, 44(D1), pp.D1202-D1213.

LaBute, M., Zhang, X., Lenderman, J., Bennion, B., Wong, S. and Lightstone, F. (2014). Adverse Drug Reaction Prediction Using Scores Produced by Large-Scale Drug-Protein Target Docking on High-Performance Computing Machines. *PLoS ONE*, 9(9), p.e106298.

Lambert, C., Leonard, N., De Bolle, X. and Depiereux, E. (2002). ESyPred3D: Prediction of proteins 3D structures. *Bioinformatics*, 18(9), pp.1250-1256.

Lander, E., Linton, L., Birren, B., Nusbaum, C., Zody, M., Baldwin, J., Devon, K., Dewar, K., Doyle, M., FitzHugh, W., Funke, R., Gage, D., Harris, K. and Heaford, A. (2001). Initial sequencing and analysis of the human genome. *Nature*, 409(6822), pp.860-921.

Larkin, M., Blackshields, G., Brown, N., Chenna, R., McGettigan, P., McWilliam, H., Valentin, F., Wallace, I., Wilm, A., Lopez, R., Thompson, J., Gibson, T. and Higgins, D. (2007). Clustal W and Clustal X version 2.0. *Bioinformatics*, 23(21), pp.2947-2948.

Laskowski, R., MacArthur, M., Moss, D. and Thornton, J. (1993). PROCHECK: a program to check the stereochemical quality of protein structures. *Journal of Applied Crystallography*, 26(2), pp.283-291.

Lebon, G., Warne, T., Edwards, P., Bennett, K., Langmead, C., Leslie, A. and Tate, C. (2011). Agonist-bound adenosine A2A receptor structures reveal common features of GPCR activation. *Nature*, 474(7352), pp.521-525.

Livingstone, C. and Barton, G. (1993). Protein sequence alignments: a strategy for the hierarchical analysis of residue conservation. *Bioinformatics*, 9(6), pp.745-756.

Lobley, A., Sadowski, M. and Jones, D. (2009). pGenTHREADER and pDomTHREADER: new methods for improved protein fold recognition and superfamily discrimination. *Bioinformatics*, 25(14), pp.1761-1767.

Lopes, L. and Bacchi, C. (2009). Imatinib treatment for gastrointestinal stromal tumour (GIST). *Journal of Cellular and Molecular Medicine*, 14(1-2), pp.42-50.

Lynch, B., Lambeng, N., Nocka, K., Kensel-Hammes, P., Bajjalieh, S., Matagne, A. and Fuks, B. (2004). The synaptic vesicle protein SV2A is the binding site for the antiepileptic drug levetiracetam. *Proceedings of the National Academy of Sciences*, 101(26), pp.9861-9866.

Masho, S., Wang, C. and Nixon, D. (2007). Review of tenofovir-emtricitabine. *Therapeutics and Clinical Risk Management*, 3(6), pp.1097-1104.

McIvor, R. (2010). Tiotropium and chronic obstructive pulmonary disease. *BMJ*, 340(feb19 1), pp.c833-c833.

McLean, M. and Gidal, B. (2003). Gabapentin dosing in the treatment of epilepsy. *Clinical Therapeutics*, 25(5), pp.1382-1406.

Merk, A., Bartesaghi, A., Banerjee, S., Falconieri, V., Rao, P., Davis, M., Pragani, R., Boxer, M., Earl, L., Milne, J. and Subramaniam, S. (2016). Breaking Cryo-EM Resolution Barriers to Facilitate Drug Discovery. *Cell*, 165(7), pp.1698-1707.

Morris, G., Huey, R., Lindstrom, W., Sanner, M., Belew, R., Goodsell, D. and Olson, A. (2009). AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility. *Journal of Computational Chemistry*, 30(16), pp.2785-2791.

Mullins, J. (2012). Structural Modelling Pipelines in Next Generation Sequencing Projects. *Challenges and Opportunities of Next-Generation Sequencing for Biomedical Research*, pp.117-167.

Nelson, D. and Cox, M. (2017). *Lehninger principles of biochemistry*. New York: W. H. Freeman.

Notredame, C., Higgins, D. and Heringa, J. (2000). T-coffee: a novel method for fast and accurate multiple sequence alignment 1 1Edited by J. Thornton. *Journal of Molecular Biology*, 302(1), pp.205-217.

Nutescu, E. and Shapiro, N. (2003). Ezetimibe: A Selective Cholesterol Absorption Inhibitor. *Pharmacotherapy*, 23(11), pp.1463-1474.

O'Boyle, N., Banck, M., James, C., Morley, C., Vandermeersch, T. and Hutchison, G. (2011). Open Babel: An open chemical toolbox. *Journal of Cheminformatics*, 3(1), p.33.

Orry, A. and Abagyan, R. (2012). *Homology modeling*. New York: Humana Press.

Park, S., Goo, J. and Jo, C. (2004). Receiver Operating Characteristic (ROC) Curve: Practical Review for Radiologists. *Korean Journal of Radiology*, 5(1), p.11.

Pellecchia, M., Sem, D. and Wüthrich, K. (2002). Nmr in drug discovery. *Nature Reviews Drug Discovery*, 1(3), pp.211-219.

Peter J., W. (1994). Protein Structure-Based Drug Design. *Annual Review of Biophysics and Biomolecular Structure*, 23(1), pp.349-375.

Pettersen, E., Goddard, T., Huang, C., Couch, G., Greenblatt, D., Meng, E. and Ferrin, T. (2004). UCSF Chimera?A visualization system for exploratory research and analysis. *Journal of Computational Chemistry*, 25(13), pp.1605-1612.

Pettersen, E., Goddard, T., Huang, C., Couch, G., Greenblatt, D., Meng, E. and Ferrin, T. (2004). UCSF Chimera?A visualization system for exploratory research and analysis. *Journal of Computational Chemistry*, 25(13), pp.1605-1612.

Pirovano, W. and Heringa, J. (2009). Protein Secondary Structure Prediction. *Methods in Molecular Biology*, pp.327-348.

Plosker, G. (2014). Sitagliptin: A Review of Its Use in Patients with Type 2 Diabetes Mellitus. *Drugs*, 74(2), pp.223-242.

Poirot, O., Suhre, K., Abergel, C., O'Toole, E. and Notredame, C. (2004). 3DCoffee@igs: a web server for combining sequences and structures into a multiple sequence alignment. *Nucleic Acids Research*, 32(Web Server), pp.W37-W40.

Pruitt, K., Tatusova, T. and Maglott, D. (2007). NCBI reference sequences (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. *Nucleic Acids Research*, 35(Database), pp.D61-D65.

Rehman, I. and Botelho, S. (2018). *Biochemistry, Tertiary Structure, Protein*. [online] Ncbi.nlm.nih.gov. Available at: https://www.ncbi.nlm.nih.gov/books/NBK470269/#!po=16.6667 [Accessed 28 Sep. 2018].

Roy, A., Kucukural, A. and Zhang, Y. (2010). I-TASSER: a unified platform for automated protein structure and function prediction. *Nature Protocols*, 5(4), pp.725-738.

Rozov, A., Demeshkina, N., Westhof, E., Yusupov, M. and Yusupova, G. (2015). Structural insights into the translational infidelity mechanism. *Nature Communications*, 6(1).

Ryvlin, P., Perucca, E. and Rheims, S. (2008). Pregabalin for the management of partial epilepsy. *Neuropsychiatric Disease and Treatment*, 4(6), pp.1211-1224.

Sacha, T. (2013). IMATINIB IN CHRONIC MYELOID LEUKEMIA: AN OVERVIEW. *Mediterranean Journal of Hematology and Infectious Diseases*, 6(1), p.2014007.

Sánchez, R. and Šali, A. (1997). Evaluation of comparative protein structure modeling by MODELLER-3. *Proteins: Structure, Function, and Genetics*, 29(S1), pp.50-58.

Sandler, B., Webb, P., Apriletti, J., Huber, B., Togashi, M., Lima, S., Juric, S., Nilsson, S., Wagner, R., Fletterick, R. and Baxter, J. (2004). Thyroxine-Thyroid Hormone Receptor Interactions. *Journal of Biological Chemistry*, 279(53), pp.55801-55808.

Sandler, B., Webb, P., Apriletti, J., Huber, B., Togashi, M., Lima, S., Juric, S., Nilsson, S., Wagner, R., Fletterick, R. and Baxter, J. (2004). Thyroxine-Thyroid Hormone Receptor Interactions. *Journal of Biological Chemistry*, 279(53), pp.55801-55808.

Shen, M. and Sali, A. (2006). Statistical potential for assessment and prediction of protein structures. *Protein Science*, 15(11), pp.2507-2524.

Sipes, N., Martin, M., Kothiya, P., Reif, D., Judson, R., Richard, A., Houck, K., Dix, D., Kavlock, R. and Knudsen, T. (2013). Profiling 976 ToxCast Chemicals across 331 Enzymatic and Receptor Signaling Assays. *Chemical Research in Toxicology*, 26(6), pp.878-895.

Smyth, M. and Martin, J. (2000). X Ray crystallography. *Molecular Pathology*, 53(1), pp.8-14.

Soding, J., Biegert, A. and Lupas, A. (2005). The HHpred interactive server for protein homology detection and structure prediction. *Nucleic Acids Research*, 33(Web Server), pp.W244-W248.

Sousa, S., Fernandes, P. and Ramos, M. (2006). Protein-ligand docking: Current status and future challenges. *Proteins: Structure, Function, and Bioinformatics*, 65(1), pp.15-26.

Sousa, S., Fernandes, P. and Ramos, M. (2006). Protein-ligand docking: Current status and future challenges. *Proteins: Structure, Function, and Bioinformatics*, 65(1), pp.15-26.

Stevens, R. (2003). The cost and value of three-dimensional protein structure. *Drug Discovery World Summer 2003*, [online] pp.35 - 48. Available at: https://www.ddw-online.com/enabling-technologies/p148412-the-cost-and-value-of-three-dimensional-protein-structure-summer-03.html [Accessed 8 Apr. 2018].

Stip, E. and Tourjman, V. (2010). Aripiprazole in schizophrenia and schizoaffective disorder: A review. *Clinical Therapeutics*, 32, pp.S3-S20.

Touw, W., Baakman, C., Black, J., te Beek, T., Krieger, E., Joosten, R. and Vriend, G. (2014). A series of PDB-related databanks for everyday needs. *Nucleic Acids Research*, 43(D1), pp.D364-D368.

Touw, W., Baakman, C., Black, J., te Beek, T., Krieger, E., Joosten, R. and Vriend, G. (2014). A series of PDB-related databanks for everyday needs. *Nucleic Acids Research*, 43(D1), pp.D364-D368.

Trott, O. and Olson, A. (2009). AutoDock Vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of Computational Chemistry*, p.NA-NA.

UniProt Consortium, T. (2018). UniProt: the universal protein knowledgebase. *Nucleic Acids Research*, 46(5), pp.2699-2699.

Verma, V., Singh, N. and Jaggi, A. (2014). Pregabalin in Neuropathic Pain: Evidences and Possible Mechanisms. *Current Neuropharmacology*, 12(1), pp.44-56.

Volkamer, A., Kuhn, D., Rippmann, F. and Rarey, M. (2012). DoGSiteScorer: a web server for automatic binding site prediction, analysis and druggability assessment. *Bioinformatics*, 28(15), pp.2074-2075.

Wang, C., Zhang, H., Zheng, W., Xu, D., Zhu, J., Wang, B., Ning, K., Sun, S., Li, S. and Bu, D. (2015). FALCON@home: a high-throughput protein structure prediction server based on remote homologue recognition. *Bioinformatics*, 32(3), pp.462-464.

Wang, S., Wacker, D., Levit, A., Che, T., Betz, R., McCorvy, J., Venkatakrishnan, A., Huang, X., Dror, R., Shoichet, B. and Roth, B. (2017). D4dopamine receptor high-resolution structures enable the discovery of selective agonists. *Science*, 358(6361), pp.381-386.

Waterhouse, A., Bertoni, M., Bienert, S., Studer, G., Tauriello, G., Gumienny, R., Heer, F., de Beer, T., Rempfer, C., Bordoli, L., Lepore, R. and Schwede, T. (2018). SWISS-MODEL: homology modelling of protein structures and complexes. *Nucleic Acids Research*, 46(W1), pp.W296-W303.

Webb, B. and Sali, A. (2016). Comparative Protein Structure Modeling Using MODELLER. *Current Protocols in Bioinformatics*, pp.5.6.1-5.6.37.

Wexler, P. (2001). TOXNET: An evolving web resource for toxicology and environmental health information. *Toxicology*, 157(1-2), pp.3-10.

Wishart, D., Feunang, Y., Guo, A., Lo, E., Marcu, A., Grant, J., Sajed, T., Johnson, D., Li, C., Sayeeda, Z., Assempour, N., Iynkkaran, I., Liu, Y., Maciejewski, A., Gale, N., Wilson, A., Chin, L., Cummings, R., Le, D., Pon, A., Knox, C. and Wilson, M. (2017). DrugBank 5.0: a major update to the DrugBank database for 2018. *Nucleic Acids Research*, 46(D1), pp.D1074-D1082.

Wu, S. and Zhang, Y. (2007). LOMETS: A local meta-threading-server for protein structure prediction. *Nucleic Acids Research*, 35(10), pp.3375-3382.

Wu, S. and Zhang, Y. (2008). MUSTER: Improving protein sequence profile-profile alignments by using multiple sources of structure information. *Proteins: Structure, Function, and Bioinformatics*, 72(2), pp.547-556.

Xiang, Z. (2006). Advances in Homology Protein Structure Modeling. *Current Protein & Peptide Science*, 7(3), pp.217-227.

Xu, D., Jaroszewski, L., Li, Z. and Godzik, A. (2013). FFAS-3D: improving fold recognition by including optimized structural features and template re-ranking. *Bioinformatics*, 30(5), pp.660-667.

XU, J., LI, M., KIM, D. and XU, Y. (2003). RAPTOR: OPTIMAL PROTEIN THREADING BY LINEAR PROGRAMMING. *Journal of Bioinformatics and Computational Biology*, 01(01), pp.95-117.

Yang, J. and Zhang, Y. (2015). Protein Structure and Function Prediction Using I-TASSER. *Current Protocols in Bioinformatics*, pp.5.8.1-5.8.15.

Yang, J., Yan, R., Roy, A., Xu, D., Poisson, J. and Zhang, Y. (2015). The I-TASSER Suite: protein structure and function prediction. *Nature Methods*, 12(1), pp.7-8.

Yang, Y., Faraggi, E., Zhao, H. and Zhou, Y. (2011). Improving protein fold recognition and template-based modeling by employing probabilistic-based matching between predicted one-dimensional structural properties of query and corresponding native properties of templates. *Bioinformatics*, 27(15), pp.2076-2082.

Zhou, H. and Skolnick, J. (2009). Protein Structure Prediction by Pro-Sp3-TASSER. *Biophysical Journal*, 96(6), pp.2119-2127.

Zuchora, B., Wielosz, M. and Urbańska, E. (2005). Adenosine A1 receptors and the anticonvulsant potential of drugs effective in the model of 3-nitropropionic acid-induced seizures in mice. *European Neuropsychopharmacology*, 15(1), pp.85-93.

Fowler, S. and Schnall, J. (2014). TOXNET. AJN, American Journal of Nursing, 114(2), pp.61-63.

Heather, J. and Chain, B. (2016). The sequence of sequencers: The history of sequencing DNA. Genomics, 107(1), pp.1-8.

Head, S., Komori, H., LaMere, S., Whisenant, T., Van Nieuwerburgh, F., Salomon, D. and Ordoukhanian, P. (2014). Library construction for next-generation sequencing: Overviews and challenges. BioTechniques, 56(2).

Kanagal-Shamanna, R. (2016). Emulsion PCR: Techniques and Applications. Clinical Applications of PCR, pp.33-42.

Harrington, C., Lin, E., Olson, M. and Eshleman, J. (2013). Fundamentals of Pyrosequencing. Archives of Pathology & Laboratory Medicine, 137(9), pp.1296-1303.

Guo, J., Yu, L., Turro, N. and Ju, J. (2010). An Integrated System for DNA Sequencing by Synthesis Using Novel Nucleotide Analogues. Accounts of Chemical Research, 43(4), pp.551-563.

Garrido-Cardenas, J., Garcia-Maroto, F., Alvarez-Bermejo, J. and Manzano-Agugliaro, F. (2017). DNA Sequencing Sensors: An Overview. Sensors, 17(3), p.588.

Alekseyev, Y., Fazeli, R., Yang, S., Basran, R., Maher, T., Miller, N. and Remick, D. (2018). A Next-Generation Sequencing Primer—How Does It Work and What Can It Do?. Academic Pathology, 5, p.237428951876652.

Ponomarenko, E., Poverennaya, E., Ilgisonis, E., Pyatnitskiy, M., Kopylov, A., Zgoda, V., Lisitsa, A. and Archakov, A. (2016). The Size of the Human Proteome: The Width and Depth. International Journal of Analytical Chemistry, 2016, pp.1-6.

Zhang, Z., Miteva, M., Wang, L. and Alexov, E. (2012). Analyzing Effects of Naturally Occurring Missense Mutations. Computational and Mathematical Methods in Medicine, 2012, pp.1-15.

Lodish, H., Berk, A., S Zipursky, L., Matsudaira, P., Baltimore, D. and Darnell, J. (2000). Molecular Cell Biology. 4th ed. New York: W. H. Freeman.

Ruwald, M., Xu Parks, X., Moss, A., Zareba, W., Baman, J., McNitt, S., Kanters, J., Shimizu, W., Wilde, A., Jons, C. and Lopes, C. (2016). Stop-codon and C-terminal nonsense mutations are associated with a lower risk of cardiac events in patients with long QT syndrome type 1. Heart Rhythm, 13(1), pp.122-131.

Zheng, S., Kim, H. and Verhaak, R. (2014). Silent Mutations Make Some Noise. Cell, 156(6), pp.1129-1131.

Old, J. (2013). Hemoglobinopathies and Thalassemias. Emery and Rimoin's Principles and Practice of Medical Genetics, pp.1-44.

Fabregat, A., Jupe, S., Matthews, L., Sidiropoulos, K., Gillespie, M., Garapati, P., Haw, R., Jassal, B., Korninger, F., May, B., Milacic, M., Roca, C., Rothfels, K., Sevilla, C., Shamovsky, V., Shorser, S., Varusai, T., Viteri, G., Weiser, J., Wu, G., Stein, L., Hermjakob, H. and D'Eustachio, P. (2017). The Reactome Pathway Knowledgebase. Nucleic Acids Research, 46(D1), pp.D649-D655.

Kanehisa, M., Sato, Y., Furumichi, M., Morishima, K. and Tanabe, M. (2018). New approach for understanding genome variations in KEGG. Nucleic Acids Research, 47(D1), pp.D590-D595.

Sanger, F. (1959). Chemistry of Insulin: Determination of the structure of insulin opens the way to greater understanding of life processes. Science, 129(3359), pp.1340-1344.

Rehman, I. and Botelho, S. (2019). Biochemistry, Secondary Protein Structure. Treasure Island (FL): StatPearls.

Medarametla, R. (2014). Prediction of Genome and Protein Structures of Homo Sapiens Neanderthalensis Using NCBI Tools. Journal of Data Mining in Genomics & Proteomics, 05(02).

Kim, S., Chen, J., Cheng, T., Gindulyte, A., He, J., He, S., Li, Q., Shoemaker, B., Thiessen, P., Yu, B., Zaslavsky, L., Zhang, J. and Bolton, E. (2018). PubChem 2019 update: improved access to chemical data. Nucleic Acids Research, 47(D1), pp.D1102-D1109.

Abdelmonsef, A., Dulapalli, R., Dasari, T., Padmarao, L., Mukkera, T. and Vuruputuri, U. (2016). Identification of Novel Antagonists for Rab38 Protein by Homology Modeling and Virtual Screening. Combinatorial Chemistry & High Throughput Screening, 19(10).

Waterhouse, A., Bertoni, M., Bienert, S., Studer, G., Tauriello, G., Gumienny, R., Heer, F., de Beer, T., Rempfer, C., Bordoli, L., Lepore, R. and Schwede, T. (2018). SWISS-MODEL: homology modelling of protein structures and complexes. Nucleic Acids Research, 46(W1), pp.W296-W303.

Schmidtke,P. and Barril,X. (2010) Understanding and predicting druggability. a highthroughput method for detection of drug binding sites. J. Med. Chem., 53, 5858–5867

Jiang, C., Zhang, J. and Song, F. (2014). Selecting Single Model in Combination Forecasting Based on Cointegration Test and Encompassing Test. The Scientific World Journal, 2014, pp.1-8.

Pettersen, E., Goddard, T., Huang, C., Couch, G., Greenblatt, D., Meng, E. and Ferrin, T. (2004). UCSF Chimera?A visualization system for exploratory research and analysis. Journal of Computational Chemistry, 25(13), pp.1605-1612.

O'Boyle, N., Banck, M., James, C., Morley, C., Vandermeersch, T. and Hutchison, G. (2011). Open Babel: An open chemical toolbox. Journal of Cheminformatics, 3(1).

Zheng, H., Handing, K., Zimmerman, M., Shabalin, I., Almo, S. and Minor, W. (2015). X-ray crystallography over the past decade for novel drug discovery – where are we heading next?. Expert Opinion on Drug Discovery, 10(9), pp.975-989.

Maity, S., Gundampati, R. and Suresh Kumar, T. (2019). NMR Methods to Characterize Protein-Ligand Interactions. Natural Product Communications, 14(5), pp.1934578X1984929.

Rcsb.org. (2019). PDB Statistics: Growth of Structures from NMR Experiments Released per Year. [online] Available at: https://www.rcsb.org/stats/growth/nmr [Accessed 3 Mar. 2019].

Carroni, M. and Saibil, H. (2016). Cryo electron microscopy to determine the structure of macromolecular complexes. Methods, 95, pp.78-85.

Muhammed, M. and Aki-Yalcin, E. (2018). Homology modeling in drug discovery: Overview, current applications, and future perspectives. Chemical Biology & Drug Design, 93(1), pp.12-20.

Burley, S., Berman, H., Bhikadiya, C., Bi, C., Chen, L., Di Costanzo, L., Christie, C., Dalenberg, K., Duarte, J., Dutta, S., Feng, Z., Ghosh, S., Goodsell, D., Green, R., Guranović, V., Guzenko, D., Hudson, B., Kalro, T., Liang, Y., Lowe, R., Namkoong, H., Peisach, E., Periskova, I., Prlić, A., Randle, C., Rose, A., Rose, P., Sala, R., Sekharan, M., Shao, C., Tan, L., Tao, Y., Valasatava, Y., Voigt, M., Westbrook, J., Woo, J., Yang, H., Young, J., Zhuravleva, M. and Zardecki, C. (2018). RCSB Protein Data Bank: biological macromolecular structures enabling research and education in fundamental biology, biomedicine, biotechnology and energy. Nucleic Acids Research, 47(D1), pp.D464-D474.

Mohammadi, M., Nejatollahi, F., Sakhteman, A. and Zarei, N. (2016). Insilico analysis of three different tag polypeptides with dual roles in scFv antibodies. Journal of Theoretical Biology, 402, pp.100-106.

Lam, S., Das, S., Sillitoe, I. and Orengo, C. (2017). An overview of comparative modelling and resources dedicated to large-scale modelling of genome sequences. Acta Crystallographica Section D Structural Biology, 73(8), pp.628-640.

Ebi.ac.uk. (2017). LALIGN < Pairwise Sequence Alignment < EMBL-EBI. [online] Available at: https://www.ebi.ac.uk/Tools/psa/lalign/ [Accessed 3 Feb. 2017].

Wu, S. and Zhang, Y. (2008). A comprehensive assessment of sequence-based and template-based methods for protein contact prediction. Bioinformatics, 24(7), pp.924-931.

Cronk, D. (2013). High-throughput screening. Drug Discovery and Development, pp.95-117.

Sousa, S., Ribeiro, A., Coimbra, J., Neves, R., Martins, S., Moorthy, N., Fernandes, P. and Ramos, M. (2013). Protein-Ligand Docking in the New Millennium – A Retrospective of 10 Years in the Field. Current Medicinal Chemistry, 20(18), pp.2296-2314.

Mathur, S. and Sutton, J. (2017). Personalized medicine could transform healthcare. Biomedical Reports, 7(1), pp.3-5.

Mohs, R. and Greig, N. (2017). Drug discovery and development: Role of basic biological research. Alzheimer's & Dementia: Translational Research & Clinical Interventions, 3(4), pp.651-657.

Xue, H., Li, J., Xie, H. and Wang, Y. (2018). Review of Drug Repositioning Approaches and Resources. International Journal of Biological Sciences, 14(10), pp.1232-1244.

Styczynski, M., Jensen, K., Rigoutsos, I. and Stephanopoulos, G. (2008). BLOSUM62 miscalculations improve search performance. Nature Biotechnology, 26(3), pp.274-275.

8 Appendix

Table 8.1: Results of the "hit" results for the known target predictions for the new homology model set (2017) using Shipyard 1.5 – shown is the
shown in figure 4.8

Ligand	Protein	DOCK Affinity SD	DOCK Affinity hit	Vina Affinity SD	Vina Affinity hit	GBSA Affinity SD	GBSA Affinity hit	DSX Atom distances SD
Ezetimibe	P15144_AMPN_HUMAN	-1.071	FALSE	-1.509	FALSE	-1.835	FALSE	-0.604
Solifenacin	P08172_ACM2_HUMAN	-0.414	FALSE	-0.004	FALSE	0.682	FALSE	-1.744
Solifenacin	P08173_ACM4_HUMAN	-0.062	FALSE	-0.257	FALSE	-0.225	FALSE	0.069
Lenalidomide	P35354_PGH2_HUMAN	-1.500	FALSE	-2.085	TRUE	-1.328	FALSE	-0.767
Lenalidomide	Q96SW2_CRBN_HUMAN	0.200	FALSE	0.104	FALSE	-0.275	FALSE	0.174
Tiotropium	P08172_ACM2_HUMAN	-0.921	FALSE	-1.228	FALSE	0.052	FALSE	-0.780
Formoterol	P07550_ADRB2_HUMAN	-1.195	FALSE	-1.486	FALSE	0.363	FALSE	-1.407
Gabapentin	P30542_AA1R_HUMAN	0.610	FALSE	0.036	FALSE	-0.069	FALSE	0.015
Fluticasone Propionate	P47712_PA24A_HUMAN	-2.218	TRUE	-2.283	TRUE	-2.697	TRUE	-0.346
Aripiprazole	P08172_ACM2_HUMAN	0.008	FALSE	0.814	FALSE	0.286	FALSE	0.052
Aripiprazole	P08173_ACM4_HUMAN	-0.058	FALSE	-0.459	FALSE	0.385	FALSE	-1.731
Aripiprazole	P08908_5HT1A_HUMAN	-0.170	FALSE	-0.142	FALSE	-0.103	FALSE	-0.301
Aripiprazole	P08913_ADA2A_HUMAN	-0.069	FALSE	-1.449	FALSE	0.379	FALSE	-1.561
Aripiprazole	P14416_DRD2_HUMAN	-1.006	FALSE	0.026	FALSE	-1.188	FALSE	0.378
Aripiprazole	P18089_ADA2B_HUMAN	-0.567	FALSE	-1.843	FALSE	-0.533	FALSE	-1.217
Aripiprazole	P18825_ADA2C_HUMAN	-0.863	FALSE	-0.566	FALSE	-0.412	FALSE	-0.501
Aripiprazole	P21917_DRD4_HUMAN	-0.472	FALSE	-0.481	FALSE	-1.181	FALSE	-0.259
Aripiprazole	P28221_5HT1D_HUMAN	-1.454	FALSE	-2.217	TRUE	-1.500	FALSE	-0.499
Aripiprazole	P28222_5HT1B_HUMAN	0.151	FALSE	-0.753	FALSE	-0.316	FALSE	0.021
Aripiprazole	P28335_5HT2C_HUMAN	-1.096	FALSE	-0.072	FALSE	-0.908	FALSE	0.801
Aripiprazole	P28566_5HT1E_HUMAN	-0.774	FALSE	-0.462	FALSE	-0.975	FALSE	-0.224
Aripiprazole	P35367_HRH1_HUMAN	-2.086	TRUE	-1.599	FALSE	-1.790	FALSE	-1.058
Aripiprazole	P35462_DRD3_HUMAN	0.293	FALSE	-0.064	FALSE	-0.541	FALSE	0.277

Table 8.2: Results of the "med" results for the known target predictions for the new homology model set (2017) using Shipyard 1.5 - shown is the graph shown in figure 4.9

Ligand	Protein	DOCK Affinity SD	DOCK Affinity med	Vina Affinity SD	Vina Affinity med	GBSA Affinity SD	GBSA Affinity med	DSX Atom distances SD
Ezetimibe	P15144_AMPN_HUMAN	-1.071	TRUE	-1.509	TRUE	-1.835	TRUE	-0.604
Solifenacin	P08172_ACM2_HUMAN	-0.414	FALSE	-0.004	FALSE	0.682	FALSE	-1.744
Solifenacin	P08173_ACM4_HUMAN	-0.0622	FALSE	-0.257	FALSE	-0.225	FALSE	0.069
Lenalidomide	P35354_PGH2_HUMAN	-1.500	TRUE	-2.085	FALSE	-1.328	TRUE	-0.767
Lenalidomide	Q96SW2_CRBN_HUMAN	0.200	FALSE	0.104	FALSE	-0.275	FALSE	0.174
Tiotropium	P08172_ACM2_HUMAN	-0.921	FALSE	-1.228	TRUE	0.052	FALSE	-0.780
Formoterol	P07550_ADRB2_HUMAN	-1.195	TRUE	-1.486	TRUE	0.363	FALSE	-1.407
Gabapentin	P30542_AA1R_HUMAN	0.610	FALSE	0.036	FALSE	-0.069	FALSE	0.015
Fluticasone Propionate	P47712_PA24A_HUMAN	-2.218	FALSE	-2.283	FALSE	-2.697	FALSE	-0.346
Aripiprazole	P08172_ACM2_HUMAN	0.008	FALSE	0.814	FALSE	0.286	FALSE	0.052
Aripiprazole	P08173_ACM4_HUMAN	-0.058	FALSE	-0.459	FALSE	0.385	FALSE	-1.731
Aripiprazole	P08908_5HT1A_HUMAN	-0.170	FALSE	-0.142	FALSE	-0.103	FALSE	-0.301
Aripiprazole	P08913_ADA2A_HUMAN	-0.069	FALSE	-1.449	TRUE	0.379	FALSE	-1.561
Aripiprazole	P14416_DRD2_HUMAN	-1.006	TRUE	0.026	FALSE	-1.188	TRUE	0.378
Aripiprazole	P18089_ADA2B_HUMAN	-0.567	FALSE	-1.843	TRUE	-0.533	FALSE	-1.217
Aripiprazole	P18825_ADA2C_HUMAN	-0.863	FALSE	-0.566	FALSE	-0.412	FALSE	-0.501
Aripiprazole	P21917_DRD4_HUMAN	-0.472	FALSE	-0.481	FALSE	-1.181	TRUE	-0.259
Aripiprazole	P28221_5HT1D_HUMAN	-1.454	TRUE	-2.217	FALSE	-1.500	TRUE	-0.499
Aripiprazole	P28222_5HT1B_HUMAN	0.151	FALSE	-0.753	FALSE	-0.316	FALSE	0.021
Aripiprazole	P28335_5HT2C_HUMAN	-1.096	TRUE	-0.072	FALSE	-0.908	FALSE	0.801
Aripiprazole	P28566_5HT1E_HUMAN	-0.774	FALSE	-0.462	FALSE	-0.975	FALSE	-0.224
Aripiprazole	P35367_HRH1_HUMAN	-2.086	FALSE	-1.599	TRUE	-1.790	TRUE	-1.058
Aripiprazole	P35462_DRD3_HUMAN	0.293	FALSE	-0.064	FALSE	-0.541	FALSE	0.277

Table 8.3: Results of the "hit" results for the known non-interactions predictions for the new homology model set (2017) using Shipyard 1.5 - sho
the graph shown in figure 4.10

Ligand	**Protein**	**DOCK Affinity SD**	**DOCK Affinity hit**	**Vina Affinity SD**	**Vina Affinity hit**	**GBSA Affinity SD**	**GBSA Affinity hit**	**DSX Atom distances SD**
Tadalafil	P19838_NFKB1_HUMAN	-0.298	FALSE	0.688	FALSE	-0.201	FALSE	-0.329
Abiraterone	P19838_NFKB1_HUMAN	0.46	FALSE	-0.274	FALSE	1.235	FALSE	-0.240
Lenalidomide	P19838_NFKB1_HUMAN	-1.992	FALSE	-2.078	TRUE	-2.309	TRUE	-2.197
Formoterol	P19838_NFKB1_HUMAN	-1.742	FALSE	-1.229	FALSE	-0.829	FALSE	-1.378
Imatinib	P19838_NFKB1_HUMAN	-2.004	TRUE	-2.163	TRUE	-0.821	FALSE	-1.877
Levothyroxine	P01137_TGFB1_HUMAN	-0.413	FALSE	0.575	FALSE	0.757	FALSE	-2.172
Levothyroxine	P19838_NFKB1_HUMAN	-1.211	FALSE	-0.18	FALSE	-1.163	FALSE	0.103
Levothyroxine	P22415_USF1_HUMAN	-0.51	FALSE	-0.047	FALSE	0.533	FALSE	-0.383
Aripiprazole	P19838_NFKB1_HUMAN	-1.292	FALSE	-0.99	FALSE	-1.217	FALSE	-1.373

Table 8.4: Results of the "med" results for the known non-interactions predictions for the new homology model set (2017) using Shipyard 1.5 - sh
create the graph shown in figure 4.11

Ligand	Protein	DOCK Affinity SD	DOCK Affinity med	Vina Affinity SD	Vina Affinity med	GBSA Affinity SD	GBSA Affinity med
Tadalafil	P19838_NFKB1_HUMAN	-0.298	FALSE	0.688	FALSE	-0.201	FALSE
Abiraterone	P19838_NFKB1_HUMAN	0.46	FALSE	-0.274	FALSE	1.235	FALSE
Lenalidomide	P19838_NFKB1_HUMAN	-1.992	TRUE	-2.078	FALSE	-2.309	FALSE
Formoterol	P19838_NFKB1_HUMAN	-1.742	TRUE	-1.229	TRUE	-0.829	FALSE
Imatinib	P19838_NFKB1_HUMAN	-2.004	FALSE	-2.163	FALSE	-0.821	FALSE
Levothyroxine	P01137_TGFB1_HUMAN	-0.413	FALSE	0.575	FALSE	0.757	FALSE
Levothyroxine	P19838_NFKB1_HUMAN	-1.211	TRUE	-0.18	FALSE	-1.163	TRUE
Levothyroxine	P22415_USF1_HUMAN	-0.51	FALSE	-0.047	FALSE	0.533	FALSE
Aripiprazole	P19838_NFKB1_HUMAN	-1.292	TRUE	-0.99	FALSE	-1.217	TRUE

Table 8.5: Results of the "hit" results for the known targets predictions for the new homology model set with 80%+ structural coverage after doc
table of results used to create the graph shown in figure 4.14

Ligand	Protein	DOCK Affinity SD	DOCK Affinity hit	Vina Affinity SD	Vina Affinity hit	GBSA Affinity SD	GBSA Affinity hit	DSX Atom distances SD
Ezetimibe	sp_P15144_AMPN_HUMAN	-1.062	FALSE	-1.496	FALSE	-1.819	FALSE	-0.607
Solifenacin	sp_P08172_ACM2_ HUMAN	-0.410	FALSE	0.000	FALSE	0.677	FALSE	-1.774
Solifenacin	sp_P08173_ACM4_ HUMAN	-0.061	FALSE	-0.251	FALSE	-0.222	FALSE	0.061
Lenalidomide	sp_P35354_PGH2_ HUMAN	-1.489	FALSE	-2.067	TRUE	-1.320	FALSE	-0.769
Lenalidomide	sp_Q96W2_CRBN_ HUMAN	0.203	FALSE	0.110	FALSE	-0.270	FALSE	0.169
Tiotropium	sp_P08172_ACM2_ HUMAN	-0.918	FALSE	-1.216	FALSE	0.055	FALSE	-0.783
Formoterol	sp_P07550_ADRB2_ HUMAN	-1.184	FALSE	-1.471	FALSE	0.364	FALSE	-1.407
Gabapentin	sp_P30542_AA1R_ HUMAN	0.611	FALSE	0.040	FALSE	-0.065	FALSE	0.011
Fluticasone Propionate	sp_P47712_PA24A_ HUMAN	-2.197	TRUE	-2.265	TRUE	-2.677	TRUE	-0.352
Aripiprazole	sp_P08172_ACM2_ HUMAN	0.007	FALSE	0.809	FALSE	0.284	FALSE	0.038
Aripiprazole	sp_P08173_ACM4_ HUMAN	-0.058	FALSE	-0.453	FALSE	0.382	FALSE	-1.742
Aripiprazole	sp_P08908_5HT10A_ HUMAN	-0.170	FALSE	-0.139	FALSE	-0.102	FALSE	-0.314
Aripiprazole	sp_P08913_ADA2A_ HUMAN	-0.069	FALSE	-1.435	FALSE	0.376	FALSE	-1.572
Aripiprazole	sp_P14416_DRD2_ HUMAN	-0.999	FALSE	0.028	FALSE	-1.180	FALSE	0.363
Aripiprazole	sp_P18825_ADA2C_ HUMAN	-0.858	FALSE	-0.559	FALSE	-0.410	FALSE	-0.514
Aripiprazole	sp_P21917_DRD4_ HUMAN	-0.469	FALSE	-0.475	FALSE	-1.173	FALSE	-0.272
Aripiprazole	sp_P28221_5HT1D_ HUMAN	-1.443	FALSE	-2.197	TRUE	-1.490	FALSE	-0.513
Aripiprazole	sp_P28335_5HT2C_ HUMAN	-1.088	FALSE	-0.070	FALSE	-0.903	FALSE	0.785
Aripiprazole	sp_P28566_5HT1E_ HUMAN	-0.769	FALSE	-0.456	FALSE	-0.968	FALSE	-0.237
Aripiprazole	sp_P35367_HRH1_ HUMAN	-2.070	TRUE	-1.584	FALSE	-1.779	TRUE	-1.070
Aripiprazole	sp_P35462_DRD3_HUMAN	0.290	FALSE	-0.062	FALSE	-0.538	FALSE	0.262

Table 8.6: Results of the "med" results for the known targets predictions for the new homology model set with 80%+ structural coverage using S shown is the original table of results used to create the graph shown in figure 4.15

Ligand	Protein	DOCK Affinity SD	DOCK Affinity med	Vina Affinity SD	Vina Affinity med	GBSA Affinity SD	GBSA Affinity med
Ezetimibe	sp_P15144_AMPN_HUMAN	-1.062	TRUE	-1.496	TRUE	-1.819	TRUE
Solifenacin	sp_P08172_ACM2_ HUMAN	-0.410	FALSE	0.000	FALSE	0.677	FALSE
Solifenacin	sp_P08173_ACM4_ HUMAN	-0.061	FALSE	-0.251	FALSE	-0.222	FALSE
Lenalidomide	sp_P35354_PGH2_ HUMAN	-1.489	TRUE	-2.067	FALSE	-1.320	TRUE
Lenalidomide	sp_Q96W2_CRBN_ HUMAN	0.203	FALSE	0.110	FALSE	-0.270	FALSE
Tiotropium	sp_P08172_ACM2_ HUMAN	-0.918	FALSE	-1.216	TRUE	0.055	FALSE
Formoterol	sp_P07550_ADRB2_ HUMAN	-1.184	TRUE	-1.471	TRUE	0.364	FALSE
Gabapentin	sp_P30542_AA1R_ HUMAN	0.611	FALSE	0.040	FALSE	-0.065	FALSE
Fluticasone Propionate	sp_P47712_PA24A_ HUMAN	-2.197	FALSE	-2.265	FALSE	-2.677	FALSE
Aripiprazole	sp_P08172_ACM2_ HUMAN	0.007	FALSE	0.809	FALSE	0.284	FALSE
Aripiprazole	sp_P08173_ACM4_ HUMAN	-0.058	FALSE	-0.453	FALSE	0.382	FALSE
Aripiprazole	sp_P08908_5HT10A_ HUMAN	-0.170	FALSE	-0.139	FALSE	-0.102	FALSE
Aripiprazole	sp_P08913_ADA2A_ HUMAN	-0.069	FALSE	-1.435	TRUE	0.376	FALSE
Aripiprazole	sp_P14416_DRD2_ HUMAN	-0.999	FALSE	0.028	FALSE	-1.180	TRUE
Aripiprazole	sp_P18825_ADA2C_ HUMAN	-0.858	FALSE	-0.559	FALSE	-0.410	FALSE
Aripiprazole	sp_P21917_DRD4_ HUMAN	-0.469	FALSE	-0.475	FALSE	-1.173	TRUE
Aripiprazole	sp_P28221_5HT1D_ HUMAN	-1.443	TRUE	-2.197	FALSE	-1.490	TRUE
Aripiprazole	sp_P28335_5HT2C_ HUMAN	-1.088	TRUE	-0.070	FALSE	-0.903	FALSE
Aripiprazole	sp_P28566_5HT1E_ HUMAN	-0.769	FALSE	-0.456	FALSE	-0.968	FALSE
Aripiprazole	sp_P35367_HRH1_ HUMAN	-2.070	FALSE	-1.584	TRUE	-1.779	TRUE
Aripiprazole	sp_P35462_DRD3_HUMAN	0.290	FALSE	-0.062	FALSE	-0.538	FALSE

Table 8.7: Results of the "hit" columns for the known targets predictions for the new protein threaded model set with 80%+ structural coverage of results used to create the graph shown in figure 4.17

Ligand	**Protein**	**DOCK Affinity SD**	**DOCK Affinity hit**	**Vina Affinity SD**	**Vina Affinity hit**	**GBSA Affinity SD**	**GBSA Affinity hit**	**DS di**
Abiraterone	P05093 CP17A HUMAN	-1.272	FALSE	-1.605	FALSE	-2.401	TRUE	-
Ezetimibe	P15144 AMPN HUMAN	-0.921	FALSE	-1.736	FALSE	-0.786	FALSE	-
Solifenacin	P08172 ACM2 HUMAN	-0.178	FALSE	-0.928	FALSE	0.231	FALSE	-
Solifenacin	P08173 ACM4 HUMAN	-0.536	FALSE	-1.640	FALSE	-0.682	FALSE	(
Solifenacin	P08912 ACM5 HUMAN	-1.082	FALSE	-0.460	FALSE	-1.025	FALSE	-
Solifenacin	P11229 ACM1 HUMAN	-0.130	FALSE	-1.053	FALSE	0.817	FALSE	-
Solifenacin	P20309 ACM3 HUMAN	-0.665	FALSE	-0.678	FALSE	-0.152	FALSE	-
Lenalidomide	Q14788 TNF11 HUMAN	-1.377	FALSE	-2.276	TRUE	-1.661	FALSE	-
Lenalidomide	P35354 PGH2 HUMAN	-1.837	FALSE	-1.326	FALSE	0.042	FALSE	-
Lenalidomide	Q96SW2 CRBN HUMAN	-1.105	FALSE	0.626	FALSE	-1.208	FALSE	-

Table 8.8: Continuation of table 8.7

Ligand	**Protein**	**DOCK Affinity SD**	**DOCK Affinity hit**	**Vina Affinity SD**	**Vina Affinity hit**	**GBSA Affinity SD**	**GBSA Affinity hit**	**D d**
Tiotropium	P08172 ACM2 HUMAN	-0.821	FALSE	-1.093	FALSE	-0.018	FALSE	
Tiotropium	P11229 ACM1 HUMAN	-0.501	FALSE	-1.159	FALSE	-0.051	FALSE	
Tiotropium	P20309 ACM3 HUMA	-1.126	FALSE	0.004	FALSE	-0.116	FALSE	
Gabapentin	P30542 AA1R HUMAN	-0.213	FALSE	-0.374	FALSE	-0.585	FALSE	
Sitagliptin	P27487 DPP4 HUMAN	-1.554	FALSE	-0.915	FALSE	-0.093	FALSE	
Levitiracetam	Q7L013 SV2A HUMAN	-1.558	FALSE	-2.106	TRUE	-0.607	FALSE	
Fluticasone propionate	P47712 PA24A HUMAN	-1.773	FALSE	-1.774	FALSE	-2.119	TRUE	
Levothyroxine	P10828 THB HUMAN	-0.801	FALSE	-1.102	FALSE	-0.924	FALSE	
Aripiprazole	P08172 ACM2 HUMAN	-0.544	FALSE	-0.365	FALSE	-0.742	FALSE	
Aripiprazole	P08173 ACM4 HUMAN	0.599	FALSE	-0.270	FALSE	-0.778	FALSE	-0.
Aripiprazole	P08908 5HT1A HUMAN	-1.239	FALSE	-1.409	FALSE	-1.888	FALSE	-0.
Aripiprazole	P08912 ACM5 HUMAN	0.601	FALSE	0.194	FALSE	-0.667	FALSE	0.0

Table 8.9: Continuation of table 8.7

Ligand	Protein	DOCK Affinity SD	DOCK Affinity hit	Vina Affinity SD	Vina Affinity hit	GBSA Affinity SD	GBSA Affinity hit	DS dist
Aripiprazole	P11229 ACM1 HUMAN	0.396	FALSE	-0.476	FALSE	-0.577	FALSE	-
Aripiprazole	P14416 DRD2 HUMAN	-0.251	FALSE	-0.664	FALSE	-0.170	FALSE	-
Aripiprazole	P18825 ADA2C HUMAN	-0.710	FALSE	-1.997	FALSE	-0.502	FALSE	-
Aripiprazole	P20309 ACM3 HUMAN	-0.571	FALSE	-0.931	FALSE	-0.742	FALSE	
Aripiprazole	P21728 DRD1 HUMAN	-.0.966	FALSE	-0.633	FALSE	-0.621	FALSE	-
Aripiprazole	P21917 DRD4 HUMAN	0.323	FALSE	-0.813	FALSE	-0.382	FALSE	-
Aripiprazole	P28223 5HT2A HUMAN	-0.523	FALSE	-1.273	FALSE	-0.534	FALSE	-
Aripiprazole	P28335 5HT2C HUMAN	0.187	FALSE	-1.784	FALSE	-0.673	FALSE	-
Aripiprazole	P34969 5HT7R HUMAN	-0.178	FALSE	-0.187	FALSE	-0.621	FALSE	-
Aripiprazole	P35367 HRH1 HUMAN	-2.260	TRUE	-1.831	FALSE	-2.115	TRUE	-
Aripiprazole	P27487 DPP4 HUMAN	-2.304	TRUE	-1.548	FALSE	-1.428	FALSE	-

Table 8.10 Results of the "med" rule for the known targets predictions for the new protein threaded model set with 80%+ structural coverage us
results used to create the graph shown in figure 4.18

Ligand	Protein	DOCK Affinity SD	DOCK Affinity med	Vina Affinity SD	Vina Affinity med	GBSA Affinity SD	GBSA Affinity med	DS d
Abiraterone	P05093 CP17A HUMAN	-1.272	TRUE	-1.605	TRUE	-2.401	FALSE	-1.
Ezetimibe	P15144 AMPN HUMAN	-0.921	FALSE	-1.736	TRUE	-0.786	FALSE	-0.
Solifenacin	P08172 ACM2 HUMAN	-0.178	FALSE	-0.928	FALSE	0.231	FALSE	-0.
Solifenacin	P08173 ACM4 HUMAN	-0.536	FALSE	-1.640	TRUE	-0.682	FALSE	0.2
Solifenacin	P08912 ACM5 HUMAN	-1.082	TRUE	-0.460	FALSE	-1.025	TRUE	-1.
Solifenacin	P11229 ACM1 HUMAN	-0.130	FALSE	-1.053	TRUE	0.817	FALSE	-0.
Solifenacin	P20309 ACM3 HUMAN	-0.665	FALSE	-0.678	TRUE	-0.152	FALSE	-0.
Lenalidomide	Q14788 TNF11 HUMAN	-1.377	TRUE	-2.276	TRUE	-1.661	TRUE	-1.
Lenalidomide	P35354 PGH2 HUMAN	-1.837	TRUE	-1.326	TRUE	0.042	FALSE	-1.
Lenalidomide	Q96SW2 CRBN HUMAN	-1.105	TRUE	0.626	FALSE	-1.208	TRUE	-1.

Table 8.11: Continuation of table 8.10

Ligand	Protein	DOCK Affinity SD	DOCK Affinity med	Vina Affinity SD	Vina Affinity med	GBSA Affinity SD	GBSA Affinity med
Tiotropium	P08172 ACM2 HUMAN	-0.821	FALSE	-1.093	TRUE	-0.018	FALSE
Tiotropium	P11229 ACM1 HUMAN	-0.501	FALSE	-1.159	TRUE	-0.051	FALSE
Tiotropium	P20309 ACM3 HUMA	-1.126	TRUE	0.004	FALSE	-0.116	FALSE
Gabapentin	P30542 AA1R HUMAN	-0.213	FALSE	-0.374	FALSE	-0.585	FALSE
Sitagliptin	P27487 DPP4 HUMAN	-1.554	TRUE	-0.915	FALSE	-0.093	FALSE
Levitiracetam	Q7L013 SV2A HUMAN	-1.558	TRUE	-2.106	FALSE	-0.607	FALSE
Fluticasone propionate	P47712 PA24A HUMAN	-1.773	TRUE	-1.774	TRUE	-2.119	FALSE
Levothyroxine	P10828 THB HUMAN	-0.801	FALSE	-1.102	TRUE	-0.924	FALSE
Aripiprazole	P08172 ACM2 HUMAN	-0.544	FALSE	-0.365	FALSE	-0.742	FALSE
Aripiprazole	P08173 ACM4 HUMAN	0.599	FALSE	-0.270	FALSE	-0.778	FALSE
Aripiprazole	P08908 5HT1A HUMAN	-1.239	TRUE	-1.409	TRUE	-1.888	TRUE
Aripiprazole	P08912 ACM5 HUMAN	0.601	FALSE	0.194	FALSE	-0.667	FALSE

Table 8.12: continuation of table 8.10

Ligand	Protein	DOCK Affinity SD	DOCK Affinity med	Vina Affinity SD	Vina Affinity med	GBSA Affinity SD	GBSA Affinity med	D dis
Aripiprazole	P11229 ACM1 HUMAN	0.396	FALSE	-0.476	FALSE	-0.577	FALSE	
Aripiprazole	P14416 DRD2 HUMAN	-0.251	FALSE	-0.664	FALSE	-0.170	FALSE	
Aripiprazole	P18825 ADA2C HUMAN	-0.710	FALSE	-1.997	TRUE	-0.502	FALSE	
Aripiprazole	P20309 ACM3 HUMAN	-0.571	FALSE	-0.931	FALSE	-0.742	FALSE	
Aripiprazole	P21728 DRD1 HUMAN	-.0.966	FALSE	-0.633	FALSE	-0.621	FALSE	
Aripiprazole	P21917 DRD4 HUMAN	0.323	FALSE	-0.813	FALSE	-0.382	FALSE	
Aripiprazole	P28223 5HT2A HUMAN	-0.523	FALSE	-1.273	TRUE	-0.534	FALSE	
Aripiprazole	P28335 5HT2C HUMAN	0.187	FALSE	-1.784	TRUE	-0.673	FALSE	
Aripiprazole	P34969 5HT7R HUMAN	-0.178	FALSE	-0.187	FALSE	-0.621	FALSE	
Aripiprazole	P35367 HRH1 HUMAN	-2.260	FALSE	-1.831	TRUE	-2.115	FALSE	
Aripiprazole	P27487 DPP4 HUMAN	-2.304	FALSE	-1.548	TRUE	-1.428	TRUE	

Table 8.13: Results of the "hit" columns for the known non-interactions predictions for the new protein threaded model set with 80%+ structural results used to create the graph shown in figure 4.19

Ligand	Protein	DOCK Affinity SD	DOCK Affinity hit	Vina Affinity SD	Vina Affinity hit	GBSA Affinity SD	GBS Affin hi
Tadalafil	P04637_P53_HUMAN	-0.593	FALSE	-0.771	FALSE	1.097	FAL
Tadalafil	P11511 CP19A HUMAN	-1.479	FALSE	-1.103	FALSE	0.432	FAL
Tadalafil	P37231 PPARG HUMAN	-1.616	FALSE	-1.243	FALSE	-0.259	FAL
Tadalafil	Q16236 NF2L2 HUMAN	0.307	FALSE	-0.227	FALSE	1.315	FAL
Abiraterone	P04637 P53 HUMAN	-1.398	FALSE	-0.908	FALSE	-0.275	FAL
Abiraterone	P11511 CP19A HUMAN	-0.987	FALSE	-0.880	FALSE	-1.316	FAL
Lenalidomide	P04637 P53 HUMAN	-0.018	FALSE	0.350	FALSE	1.593	FAL
Lenalidomide	P11511 CP19A HUMAN	0.242	FALSE	-0.129	FALSE	-0.861	FAL
Lenalidomide	P37231 PPARG HUMAN	-0.899	FALSE	-0.716	FALSE	-0.462	FAL
Lenalidomide	Q16236 NF2L2 HUMAN	0.411	FALSE	0.237	FALSE	0.765	FAL
Formoterol	P04637 P53 HUMAN	-0.548	FALSE	-0.132	FALSE	-0.230	FAL
Formoterol	P11511 CP19A HUMAN	-1.102	FALSE	-0.748	FALSE	-1.139	FAL
Formoterol	P37231 PPARG HUMAN	-2.416	TRUE	-1.499	FALSE	-1.213	FAL
Formoterol	Q16236 NF2L2 HUMAN	0.696	FALSE	1.647	FALSE	1.145	FAL
Budesonide	O43524 FOXO3 HUMAN	0.020	FALSE	0.496	FALSE	2.421	FAL
Budesonide	P01106 MYC HUMAN	-0.278	FALSE	-1.002	FALSE	0.476	FAL
Budesonide	P10276 RARA HUMAN	-1.793	FALSE	-2.038	TRUE	-2.138	TRU
Budesonide	P10826 RARB HUMAN	0.032	FALSE	0.967	FALSE	-0.024	FAL
Budesonide	P11474 ERR1 HUMAN	-1.194	FALSE	-0.426	FALSE	0.773	FAL
Budesonide	P11511 CP19A HUMAN	-1.316	FALSE	-1.105	FALSE	-1.532	FAL
Budesonide	P13631 RARG HUMAN	0.449	FALSE	-0.422	FALSE	2.086	FAL
Budesonide	P15976 GATA1 HUMAN	0.355	FALSE	0.320	FALSE	0.281	FAL
Budesonide	P19793 RXRA HUMAN	-0.077	FALSE	-0.339	FALSE	-0.364	FAL
Budesonide	P26367 PAX6 HUMAN	-0.594	FALSE	-0.695	FALSE	-0.604	FAL
Budesonide	P28702 RXRB HUMAN	0.219	FALSE	-0.013	FALSE	-0.650	FAL
Budesonide	P35398 RORA HUMAN	-0.438	FALSE	-1.105	FALSE	-0.318	FAL

Table 8.14: Continuation of Table 8.13

Ligand	Protein	DOCK Affinity SD	DOCK Affinity hit	Vina Affinity SD	Vina Affinity hit	GBSA Affinity SD
Budesonide	P37231 PPARG HUMAN	0.305	FALSE	0.154	FALSE	0.775
Budesonide	P40763 STAT3 HUMAN	0.896	FALSE	0.059	FALSE	2.356
Budesonide	P41235 HNF4A HUMAN	0.837	FALSE	-0.494	FALSE	2.899
Budesonide	Q01094 E2F1 HUMAN	1.237	FALSE	1.510	FALSE	0.664
Budesonide	Q14994 NR1I3 HUMAN	-0.646	FALSE	-0.761	FALSE	-0.659
Budesonide	Q15797 SMAD1 HUMAN	-0.065	FALSE	-1.134	FALSE	1.405
Budesonide	Q7Z6R9 AP2D HUMAN	0.303	FALSE	0.083	FALSE	-0.180
Budesonide	Q92481 AP2B HUMAN	-0.336	FALSE	-1.083	FALSE	-1.145
Budesonide	Q92753 RORB HUMAN	-0.212	FALSE	-3.885	TRUE	-1.163
Budesonide	Q9UBC0 HNF6 HUMAN	-1.388	FALSE	-1.204	FALSE	-1.861
Budesonide	P04637 P53 HUMAN	-1.138	FALSE	-0.714	FALSE	-1.312
Imatinib	P37231 PPARG HUMAN	-0.340	FALSE	-1.108	FALSE	-0.399
Imatinib	Q16236 NF2L2 HUMAN	1.812	FALSE	0.975	FALSE	3.197
Imatinib	P04637 P53 HUMAN	-0.018	FALSE	-0.245	FALSE	-0.538
Pregabalin	P11511 CP19A HUMAN	-1.845	FALSE	-1.483	FALSE	-1.920
Pregabalin	P37231 PPARG HUMAN	0.148	FALSE	-0.315	FALSE	-0.437
Pregabalin	O43524 FOXO3 HUMAN	-0.105	FALSE	0.090	FALSE	-0.843
Levothyroxine	P01106 MYC HUMAN	-1.110	FALSE	-1.488	FALSE	-1.369
Levothyroxine	P01137 TGFB1 HUMAN	-0.698	FALSE	0.258	FALSE	0.475
Levothyroxine	P10276 RARA HUMAN	-0.841	FALSE	-0.233	FALSE	-0.947
Levothyroxine	P10826 RARB HUMAN	0.244	FALSE	0.090	FALSE	0.861
Levothyroxine	P10914 IRF1 HUMAN	-3.013	TRUE	-1.117	FALSE	-1.369
Levothyroxine	P11474 ERR1 HUMAN	-0.853	FALSE	0.036	FALSE	0.919
Levothyroxine	P11511 CP19A HUMAN	-2.137	TRUE	-0.831	FALSE	-2.115
Levothyroxine	P13631 RARG HUMAN	0.366	FALSE	0.375	FALSE	1.067
Levothyroxine	P14921 ETS1 HUMAN	-0.917	FALSE	0.203	FALSE	-0.064
Levothyroxine	P15976 GATA1 HUMAN	-0.154	FALSE	0.985	FALSE	-0.809

Table 8.15: continuation of Table 8.13.

Ligand	Protein	DOCK Affinity SD	DOCK Affinity hit	Vina Affinity SD	Vina Affinity hit	GBSA Affinity SD	A
Levothyroxine	P19793 RXRA HUMAN	-0.449	FALSE	-1.550	FALSE	-0.644	F
Levothyroxine	P22415 USF1 HUMAN	-1.012	FALSE	-1.015	FALSE	0.113	F
Levothyroxine	P26367 PAX6 HUMAN	-0.258	FALSE	-0.642	FALSE	0.723	F
Levothyroxine	P40763 STAT3 HUMAN	0.617	FALSE	0.935	FALSE	0.633	F
Levothyroxine	P41235 HNF4A HUMAN	-2.050	TRUE	-1.423	FALSE	-1.608	F
Levothyroxine	P55055 NR1H2 HUMAN	-0.750	FALSE	-0.749	FALSE	-1.223	F
Levothyroxine	Q01094 E2F1 HUMAN	0.557	FALSE	1.099	FALSE	0.588	F
Levothyroxine	Q13133 NR1H3 HUMAN	-0.883	FALSE	-0.120	FALSE	-0.730	F
Levothyroxine	Q15797 SMAD1 HUMAN	1.619	FALSE	1.721	FALSE	1.617	F
Levothyroxine	Q7Z6R9 AP2D AP2D HUMAN	-0.127	FALSE	-0.991	FALSE	0.377	F
Levothyroxine	Q92481 AP2B HUMAN	-0.513	FALSE	-0.559	FALSE	0.032	F
Levothyroxine	Q9UBC0 HNF6 HUMAN	-1.225	FALSE	-1.298	FALSE	-1.285	F
Aripiprazole	Q9UJU2 LEF1 HUMAN	0.230	FALSE	0.816	FALSE	0.170	F
Emtricitabine	Q9Y261 FOXA2 HUMAN	-0.059	FALSE	0.566	FALSE	0.457	F
Emtricitabine	P11511 CP19A HUMAN	-1.279	FALSE	-1.224	FALSE	-0.678	F
Emtricitabine	P04637 P53 HUMAN	-0.402	FALSE	-0.612	FALSE	-0.203	F
Emtricitabine	P11511 CP19A HUMAN	-0.321	FALSE	-0.528	FALSE	0.559	F
Emtricitabine	P37231 PPARG	-0.417	FALSE	-0.590	FALSE	-0.443	F

Table 8.16: Results of the "med" rule for the known non-interactions predictions for the new protein threaded model set with 80%+ structural co used to create the graph shown in figure 4.20

Ligand	Protein	DOCK Affinity SD	DOCK Affinity med	Vina Affinity SD	Vina Affinity med	GBSA Affinity SD	GBS Affin me
Tadalafil	P04637_P53_HUMAN	-0.593	FALSE	-0.771	TRUE	1.097	FAL
Tadalafil	P11511 CP19A HUMAN	-1.479	TRUE	-1.103	TRUE	0.432	FAL
Tadalafil	P37231 PPARG HUMAN	-1.616	TRUE	-1.243	FALSE	-0.259	FAL
Tadalafil	Q16236 NF2L2 HUMAN	0.307	FALSE	-0.227	FALSE	1.315	FAL
Abiraterone	P04637 P53 HUMAN	-1.398	TRUE	-0.908	FALSE	-0.275	FAL
Abiraterone	P11511 CP19A HUMAN	-0.987	FALSE	-0.880	FALSE	-1.316	TRU
Lenalidomide	P04637 P53 HUMAN	-0.018	FALSE	0.350	FALSE	1.593	FAL
Lenalidomide	P11511 CP19A HUMAN	0.242	FALSE	-0.129	FALSE	-0.861	FAL
Lenalidomide	P37231 PPARG HUMAN	-0.899	FALSE	-0.716	FALSE	-0.462	FAL
Lenalidomide	Q16236 NF2L2 HUMAN	0.411	FALSE	0.237	FALSE	0.765	FAL
Formoterol	P04637 P53 HUMAN	-0.548	FALSE	-0.132	FALSE	-0.230	FAL
Formoterol	P11511 CP19A HUMAN	-1.102	TRUE	-0.748	FALSE	-1.139	TRU
Formoterol	P37231 PPARG HUMAN	-2.416	FALSE	-1.499	TRUE	-1.213	TRU
Formoterol	Q16236 NF2L2 HUMAN	0.696	FALSE	1.647	FALSE	1.145	FAL
Budesonide	O43524 FOXO3 HUMAN	0.020	FALSE	0.496	FALSE	2.421	FAL
Budesonide	P01106 MYC HUMAN	-0.278	FALSE	-1.002	TRUE	0.476	FAL
Budesonide	P10276 RARA HUMAN	-1.793	TRUE	-2.038	FALSE	-2.138	FAL
Budesonide	P10826 RARB HUMAN	0.032	FALSE	0.967	FALSE	-0.024	FAL
Budesonide	P11474 ERR1 HUMAN	-1.194	TRUE	-0.426	FALSE	0.773	FAL
Budesonide	P11511 CP19A HUMAN	-1.316	TRUE	-1.105	TRUE	-1.532	TRU
Budesonide	P13631 RARG HUMAN	0.449	FALSE	-0.422	FALSE	2.086	FAL
Budesonide	P15976 GATA1 HUMAN	0.355	FALSE	0.320	FALSE	0.281	FAL
Budesonide	P19793 RXRA HUMAN	-0.077	FALSE	-0.339	FALSE	-0.364	FAL
Budesonide	P26367 PAX6 HUMAN	-0.594	FALSE	-0.695	FALSE	-0.604	FAL
Budesonide	P28702 RXRB HUMAN	0.219	FALSE	-0.013	FALSE	-0.650	FAL
Budesonide	P35398 RORA HUMAN	-0.438	FALSE	-1.105	TRUE	-0.318	FAL

Table 8.17: continuation of Table 8.16.

Ligand	Protein	DOCK Affinity SD	DOCK Affinity med	Vina Affinity SD	Vina Affinity med	GBSA Affinity SD
Budesonide	P37231 PPARG HUMAN	0.305	FALSE	0.154	FALSE	0.775
Budesonide	P40763 STAT3 HUMAN	0.896	FALSE	0.059	FALSE	2.356
Budesonide	P41235 HNF4A HUMAN	0.837	FALSE	-0.494	FALSE	2.899
Budesonide	Q01094 E2F1 HUMAN	1.237	FALSE	1.510	FALSE	0.664
Budesonide	Q14994 NR1I3 HUMAN	-0.646	FALSE	-0.761	FALSE	-0.659
Budesonide	Q15797 SMAD1 HUMAN	-0.065	FALSE	-1.134	TRUE	1.405
Budesonide	Q7Z6R9 AP2D HUMAN	0.303	FALSE	0.083	FALSE	-0.180
Budesonide	Q92481 AP2B HUMAN	-0.336	FALSE	-1.083	TRUE	-1.145
Budesonide	Q92753 RORB HUMAN	-0.212	FALSE	-3.885	FALSE	-1.163
Budesonide	Q9UBC0 HNF6 HUMAN	-1.388	FALSE	-1.204	TRUE	-1.861
Budesonide	P04637 P53 HUMAN	-1.138	TRUE	-0.714	FALSE	-1.312
Imatinib	P37231 PPARG HUMAN	-0.340	TRUE	-1.108	TRUE	-0.399
Imatinib	Q16236 NF2L2 HUMAN	1.812	FALSE	0.975	FALSE	3.197
Imatinib	P04637 P53 HUMAN	-0.018	FALSE	-0.245	FALSE	-0.538
Pregabalin	P11511 CP19A HUMAN	-1.845	FALSE	-1.483	TRUE	-1.920
Pregabalin	P37231 PPARG HUMAN	0.148	TRUE	-0.315	FALSE	-0.437
Pregabalin	O43524 FOXO3 HUMAN	-0.105	FALSE	0.090	FALSE	-0.843
Levothyroxine	P01106 MYC HUMAN	-1.110	FALSE	-1.488	TRUE	-1.369
Levothyroxine	P01137 TGFB1 HUMAN	-0.698	TRUE	0.258	FALSE	0.475
Levothyroxine	P10276 RARA HUMAN	-0.841	FALSE	-0.233	FALSE	-0.947
Levothyroxine	P10826 RARB HUMAN	0.244	FALSE	0.090	FALSE	0.861
Levothyroxine	P10914 IRF1 HUMAN	-3.013	FALSE	-1.117	TRUE	-1.369
Levothyroxine	P11474 ERR1 HUMAN	-0.853	FALSE	0.036	FALSE	0.919
Levothyroxine	P11511 CP19A HUMAN	-2.137	FALSE	-0.831	FALSE	-2.115
Levothyroxine	P13631 RARG HUMAN	0.366	FALSE	0.375	FALSE	1.067
Levothyroxine	P14921 ETS1 HUMAN	-0.917	FALSE	0.203	FALSE	-0.064
Levothyroxine	P15976 GATA1 HUMAN	-0.154	FALSE	0.985	FALSE	-0.809

Table 8.18: Continuation of Table 8.16.

Ligand	Protein	DOCK Affinity SD	DOCK Affinity med	Vina Affinity SD	Vina Affinity med	GBSA Affinity SD	A
Levothyroxine	P19793 RXRA HUMAN	-0.449	FALSE	-1.550	TRUE	-0.644	F
Levothyroxine	P22415 USF1 HUMAN	-1.012	TRUE	-1.015	TRUE	0.113	F
Levothyroxine	P26367 PAX6 HUMAN	-0.258	FALSE	-0.642	FALSE	0.723	F
Levothyroxine	P40763 STAT3 HUMAN	0.617	FALSE	0.935	FALSE	0.633	F
Levothyroxine	P41235 HNF4A HUMAN	-2.050	FALSE	-1.423	TRUE	-1.608	T
Levothyroxine	P55055 NR1H2 HUMAN	-0.750	FALSE	-0.749	FALSE	-1.223	T
Levothyroxine	Q01094 E2F1 HUMAN	0.557	FALSE	1.099	FALSE	0.588	F
Levothyroxine	Q13133 NR1H3 HUMAN	-0.883	FALSE	-0.120	FALSE	-0.730	F
Levothyroxine	Q15797 SMAD1 HUMAN	1.619	FALSE	1.721	FALSE	1.617	F
Levothyroxine	Q7Z6R9 AP2D AP2D HUMAN	-0.127	FALSE	-0.991	FALSE	0.377	F
Levothyroxine	Q92481 AP2B HUMAN	-0.513	FALSE	-0.559	FALSE	0.032	F
Levothyroxine	Q9UBC0 HNF6 HUMAN	-1.225	TRUE	-1.298	TRUE	-1.285	T
Aripiprazole	Q9UJU2 LEF1 HUMAN	0.230	FALSE	0.816	FALSE	0.170	F
Emtricitabine	Q9Y261 FOXA2 HUMAN	-0.059	FALSE	0.566	FALSE	0.457	F
Emtricitabine	P11511 CP19A HUMAN	-1.279	FALSE	-1.224	TRUE	-0.678	F
Emtricitabine	P04637 P53 HUMAN	-0.402	FALSE	-0.612	FALSE	-0.203	F
Emtricitabine	P11511 CP19A HUMAN	-0.321	FALSE	-0.528	FALSE	0.559	F
Emtricitabine	P37231 PPARG	-0.417	FALSE	-0.590	FALSE	-0.443	F