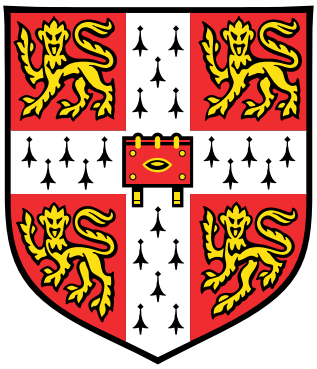
Protein Shape Description and its Application to Shape Comparison



Michal Tykač

MRC Laboratory of Molecular Biology

University of Cambridge

This dissertation is submitted for the degree of

Doctor of Philosophy

Wolfson College

October 2018

Protein Shape Description and its Application to Shape Comparison

Michal Tykač

There are currently over 138,000 known macromolecular structures deposited in the wwPDB (Worldwide Protein Data Bank) database. While all the macromolecular structure files contain information about a particular structure, the collection of these files also allows combining the macromolecular structures to obtain statistical information about macromolecules in general. This fact has been the basis for many structural biology methods including the molecular replacement method used in X-ray crystallography or homologous structure restraints in the refinement methods. With the success of methods based on prior information, it is feasible that novel methods could be developed and current methods improved using further prior information; more specifically, by using the structure density-map shape similarity instead of sequence or model similarity. Therefore, this project introduces a mathematical framework for computing three different measures of macromolecular three-dimensional shape similarity and demonstrates how these descriptors can be applied in symmetry detection and protein-domain clustering. The ability to detect cyclic (C), dihedral (D), tetrahedral (T), octahedral (O) and icosahedral (I) symmetry groups as well as computing all associated symmetry elements has direct applications in map averaging and reducing the storage requirements by storing only the asymmetric information. Moreover, by having the capacity to find structures with similar shape, it was possible to reduce the size of the BALBES protein domain database by more than 18.7% and thus achieve proportional speed-up in the searching parts of its applications. Finally, the development of the method described in this project has many possible applications throughout structural biology. The method could, for example, facilitate matching and fitting of protein domains into the density maps produced by the electron-microscopy techniques, or it could allow for molecular-replacement candidate search using shape instead of sequence similarity. To allow for the development of any further applications, software for applying the methods described here is also presented and released for the community.

Declaration

I hereby declare that this dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration except as declared in the Acknowledgements and specified in the text. It is not substantially the same as any that I have submitted, or, is being concurrently submitted for a degree or diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Acknowledgements and specified in the text. I further state that no substantial part of my dissertation has already been submitted, or, is being concurrently submitted for any such degree, diploma or other qualification at the University of Cambridge or any other University or similar institution except as declared in the Acknowledgements and specified in the text. This dissertation contains fewer than 60,000 words excluding appendices, bibliography, footnotes, tables and equations.

Michal Tykač

October 2018

Acknowledgements

I would like to sincerely thank my supervisor Dr. Garib N. Murshudov for his immense help with every step of this project, from its very beginnings to proof-reading the drafts of this thesis. Many of the ideas in this thesis were born from the multitude of discussions with him and the project would not have happened without him. Moreover, he allowed me to explore my own ideas as well as guiding and motivating me throughout the project and especially when ideas were not working out. I have really enjoyed my PhD as his student.

Furthermore, I would also like to thank Dr. Robert A. Nicholls for being always available for discussion, suggesting many interesting ideas and giving me access to parts of his code. His help was irreplaceable, especially for the computational part of the project.

Moreover, I would like to thank Dr. Fei Long for helping me with understanding the background and details of the BALBES protein domain database and making partial results of his work available to me. Without his help, the project would not have progressed as smoothly as it did.

I would like to further thank Dr. Oleg Kovalevskiy, who was extremely helpful in explaining details of the experimental methods and particularities of the computational data. He also happily tested many versions of the software and patiently reported bugs to me along with his data, thus allowing me to correct the issues.

Dr. Paul Emsley has contributed considerably to this project by asking many important questions and giving many suggestions, as well as being immensely help-

ful in solving many linking and library usage problems faced in the development of the software and for this I am very thankful to him.

Finally, I would like to thank my university supervisor Professor Ben Luisi and my secondary supervisor Dr. M. Madan Babu for the helpful discussions and ideas they provided me with, as well as the encouragement during our meetings.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Brief introduction to structural biology methods

Structural biology is the field of study of the structures of biological macromolecules, typically proteins, DNA and RNA. Structural biology comprises of methods for determining the atomic models of molecules, as well as methods for exploring the atomic model structure and function. The prominent methods for determining atomic models of biological macromolecules include X-ray crystallography, nuclear magnetic resonance (NMR) and electron microscopy. This introduction will, however, cover only basics of X-ray crystallography and electron microscopy, as the results of these two methods will be the basis of this work. For more detailed introduction to the field of molecular biology, see for example Banaszak (2000), Frank (2006), Rupp (2010) or Alberts (2014).

1.1.1 X-ray crystallography

X-ray crystallography is an experimental method using the packing of molecules in a crystal and the diffraction of X-rays by a crystal to determine the electron density of such molecules. When applied in structural biology, the standard procedure starts with obtaining a molecule of interest in high concentration and subsequently reducing the concentration of other solvents in order to cause the molecule of

interest to form ordered crystals - a process called crystallisation. The particular crystallisation conditions required to obtain crystals of a molecule of interest do vary considerably and many molecules were not successfully crystallised as of yet. For detailed review of protein crystallisation, see for example McPherson (2004).

Once a crystal of a molecule of interest is obtained, it is subjected to an incident beam of monochromatic (single wavelength) X-rays. This results in the incident X-rays being diffracted by the crystal in all directions. While in most directions the diffracted X-rays cancel out through destructive interference, in directions given by the Bragg's law (Bragg, 1913), the diffracted X-rays combine through constructive interference. Bragg's law can be stated as follows:

$$2d \sin(\theta) = n\lambda \tag{1.1}$$

where:

d is the distance between neighbouring crystal lattice planes.

θ is the scattering angle (i.e. the angle at which the beam hits the crystal lattice plane).

λ is the wavelength of the incident X-ray.

n is a positive integer.

The result of this physical phenomenon is a set of measurable diffraction spots called *reflections*, with each reflection having particular intensity given by the crystal contents and a position on the detector determined by the crystal lattice. The set of reflections produced by a single crystal in a particular orientation is called an image; all the reflections in an image can be recorded on a film, using image sensors, or using the modern detectors such as the Pilatus (Kraft et al., 2009) and Eiger (Johnson et al., 2012) detectors, which use the photoelectric effect in silicon to detect X-rays. However, one such image represents only a slice of the Fourier coefficients space (also called the reciprocal space) of the crystal. Therefore, the crystal needs to be rotated and diffraction image recorded along the full 180° rotation, unless the crystal symmetry allows for smaller rotation.

Once all data are recorded, the reflections need to be indexed; that is the reflections need to have their reciprocal space positions assigned and the cell parameters and space group need to be determined. This is typically done using the autoindexing algorithms described by, for example, Kabsch (1988), Kabsch (1993), Steller et al. (1997) or Powell (1999). These algorithms typically use 1D Fourier methods for this task, for example as employed by `LABELIT` (Sauter et al., 2004) and `MOSFLM` (Leslie, 2006), or a combination of 1D and 3D Fourier methods such as described by Gildea et al. (2014) and used in `DIALS` (Parkhurst et al., 2016) software. Subsequently, the individual diffraction intensities can be merged together by identifying which reflections appear in multiple diffraction images and scaling the intensities derived from them accordingly. Once the reflections are indexed, integrated, scaled and merged, it is possible to obtain the amplitudes of individual reflections.

With the intensities computed, an issue arises: The computed intensities are related to the squared modulus of the structure factors, that is, the complex numbers describing the reciprocal (Fourier) space of the X-ray diffraction. However, since it is not possible to reconstruct the structure factors from the measured intensities, the diffraction theory equations for computing density from structure factors cannot be directly applied. This issue is known as the *phase problem*.

There are several possible approaches to solving the phase problem. One possibility is using the computational approaches known as direct methods, which use statistical information to derive initial set of phases (Hauptman, 1986); however, these approaches are computationally expensive and better suited for small molecules rather than macromolecules. Alternative computational approach is based on a known homologous structure of similar shape - this approach is called molecular replacement and the initial phase information is derived from an already known structure. When neither computational approach can be used, it is possible to use experimental approaches based on using the knowledge about *heavy atoms* (atoms with high atomic number, *e.g.* metals) either already present in

the structure or introduced experimentally. Methods based on isomorphous replacement such as SIR (Blow and Rossmann, 1961) and MIR (Blow, 1958) are based on introducing the heavy metals into the crystal and using the resulting structure-factors differences to solve the sub-structure. Similarly, methods such as SAD (Yang et al., 2003) and MAD (Hendrickson and Ogata, 1997) are also based on using heavy atoms (either already present or introduced into the crystal) and their ability to break the Friedel's law (which states that pairs of Bragg reflections related by inversion through the origin have equal amplitude and opposite phase; *i.e.* $|F_{hkl}| = |F_{\overline{hkl}}|$ and $\varphi_{hkl} = -\varphi_{\overline{hkl}}$ as explained, for example, in Merritt (2011)) and become *anomalous scatterers*. The anomalous scatterers cause changes in the reflection intensities and this fact is used to determine the initial phases of the structure. The combination of anomalous scattering methods (SAD, MAD) and isomorphous replacement (MIR, SIR) are currently also used; for a review of the experimental methods, see for example Taylor (2010) or Hendrickson (2014).

With the initial phases present, atomic model can be build into the resulting density map either automatically or manually. Examples of automatic model building software include `ARP/wARP` (Perrakis et al., 1997), `TEXTAL` (Ioerger and Sacchettini, 2003), `RESOLVE` (Terwilliger, 2004), `Bucanneer` (Cowtan, 2006) or `AutoSol` (Terwilliger et al., 2009), while interactive manual model building software examples include `O` (Jones et al., 1991) and `Coot` (Emsley et al., 2010). Once atomic model has been built into the electron density map, its fit can be improved by the refinement process as done by, for example, `phenix.refine` (Afonine et al., 2012) or `REFMAC5` (Kovalevskiy et al., 2018). Typically, the procedures of model building and refinement are done iteratively.

1.1.2 Electron microscopy

Electron microscopy is a microscopy technique based on using electrons as the light source instead of the visible light. This approach has the potential advantage of higher resolution, as the wavelength of electrons is approximately 0.02 Å at 300

keV (Milne et al., 2013), while the visible light photon wavelength is approximately $4,000$ to $7,000$ Å at around 2 keV; however, the resolution is currently limited by radiation damage, which restrains the dose of electrons that can be shot at the sample. Therefore, electron microscopes use electron guns to produce a steady stream of electrons of the required voltage instead of a mirror reflecting photons from a bulb that the standard light microscopes employ.

However, with different light source, the electron microscope also requires a different focusing approach as optical lenses do not focus electrons. Instead, electrostatic lenses using several cylinders with differing voltage to create an electric field to focus the electrons proportionally to their speed can be used to act as a lens. Another possibility is using electromagnetic lenses, which are based on an isolated coil wrapped around the electron path; as current flows through the coil, it produces magnetic field perpendicular to the axis of the lens and this magnetic field then acts as an electron lens. For detailed information about electron lenses, see for example El-Kareh and El-Kareh (1970).

Moreover, as the high voltage required for the electron gun cathode could produce an arc with the ground, high vacuum is required in the electron microscope to avoid this; typically, a vacuum of 10^{-4} to 10^{-9} Pascals is required to avoid an arc, depending on the voltage of the electron gun cathode. The vacuum also serves another purpose - to reduce the interaction between the electrons and any gas present in the microscope as this interaction would affect the focusing and ultimately the resolution observed by the microscope. Therefore, the electron microscopes are usually operated in a vacuum.

While all the previous features are present in all electron microscopes, it is worth noting that there are several different types of electron microscopy. The transmission electron microscope (TEM) uses the electron beam to illuminate the sample and detects the electrons exiting the sample to produce an image; this however requires the sample to be partially transparent to electrons. In order for the sample to have this property, it typically needs to be very thin (approximately

1, 000 Å). Different images are produced by the scanning electron microscope (SEM), which focuses the electron beam to a section of the sample and records the particles emitted as a result of the electron interaction with the sample. These include the low energy secondary electrons, X-rays or even visible light. This information is then associated with the part of the sample which the beam was focused on and the process is repeated in a raster over the whole sample. The image is then reconstructed from the recorded particles resulting from the interaction instead of the primary electrons. Advantage of the SEM approach is that the sample does not need to be transparent to electrons, while the disadvantage rests in the resolution of SEM being much lower than TEM as the electron beam focusing cannot be done perfectly. For more information about SEM and TEM approaches, see for example the reviews of Vernon-Parry (2000), Muscariello et al. (2005), Kourkoutis et al. (2012), Winey et al. (2014) or Borrajo-Pelaez and Hedstrom (2017)

Generally, when biological samples are subjected to direct stream of electrons, the covalent bonds may be broken and free radicals can be introduced as described, for example, in Glaeser et al. (1971), Henderson (1990) and Egerton et al. (2004). One possible solution to reduce the radiation damage of the sample is to image it under the cryogenic temperatures (typically the liquid nitrogen temperatures around -180° C or 93.15 K (Kuhlbrandt, 2014)) as suggested by Glaeser et al. (1971). Furthermore, when the sample is rapidly frozen to cryogenic temperatures as described by Dubochet et al. (1988), the sample is effectively fixed in place, as molecular movement is proportional to the temperature and therefore the cryogenic temperatures remove the need for sample fixation (Adrian et al., 1984) as well as reducing radiation damage.

One possible approach using electron microscopy to determine macromolecular structures is the use of EM tomography. This method is based on the TEM approach with the sample being rapidly frozen and then tilted as it is being recorded. This method has typical resolution of between 50 and 100 Å (Milne et al., 2013); a

resolution generally not sufficient for determination of macromolecular structures. However, when multiple copies of the structure are imaged, the resolution can be improved by averaging the tomograms of individual particles as described by, for example, Robinson et al. (2007), Bartesaghi et al. (2008) and Briggs (2013). While averaging is not sufficient for structure determination of individual small proteins, it has been used to determine structures of viral envelope proteins, for example by White et al. (2010). For more detailed description of EM tomography, see for example, Hoenger and Bouchet-Marquis (2011).

Another fast evolving application of electron microscopy in the structural biology field is the cryo-electron microscopy (cryo-EM) method based on TEM. This method requires the sample to be placed on a (typically carbon) grid and rapidly frozen to cryogenic temperatures. The sample is then imaged in different locations of the grid, creating multitude of images each with hundreds of particles. These particles are then picked (located) either by hand or automatically using software such as `AutoPicker` (Langlois et al., 2014) or `FastParticlePicker` (Xiao and Yang, 2017), or similar functionality offered by the EM software suites such as `EMAN1` (Ludtke et al., 1999), `Cyclops` (Plaisier and Abrahams, 2007), `EMAN2` (Tang et al., 2007), `IMAGIC` (van Heel, 2012) or `RELION` (Scheres, 2012). Subsequently, the 2D particle images need to be aligned and classified into 2D classes; this can be done either by stand-alone software, such as `ISAC` (Yang et al., 2012) or `FREALIGN` (Lyumkis et al., 2013); or by one of the aforementioned software suites for EM map reconstruction.

Once classified, the 2D classes can be averaged to obtain 2D projections of the 3D shape of the structure with higher signal-to-noise ratio and resolution than available in the individual images. The 3D map of the structure can then be reconstructed from these 2D projections using the central projection theorem. This theorem states that Fourier transform of a 2D projection is a central slice of the Fourier transform of the 3D structure and thereby allows finding the relative orientations of the 2D projections required for the 3D reconstruction. Details

of this procedure are discussed, for example, by Crowther et al. (1970) and van Heel (1987) and the procedure is available in EM software suites such as EMAN1 (Ludtke et al., 1999), Cyclops (Plaisier and Abrahams, 2007), EMAN2 (Tang et al., 2007), IMAGIC (van Heel, 2012) or RELION (Scheres, 2012). Finally, if the map resolution is high enough, atomic model fitting into the EM map can be attempted manually using for example Chimera (Pettersen et al., 2004) or Coot (Emsley et al., 2010). This atomic model can also be refined to optimise its fit to the map by using, for example, phenix.real_space_refine (Afonine et al., 2012) or REFMAC5 (Kovalevskiy et al., 2018). The model building and refinement can be repeated iteratively to improve the model.

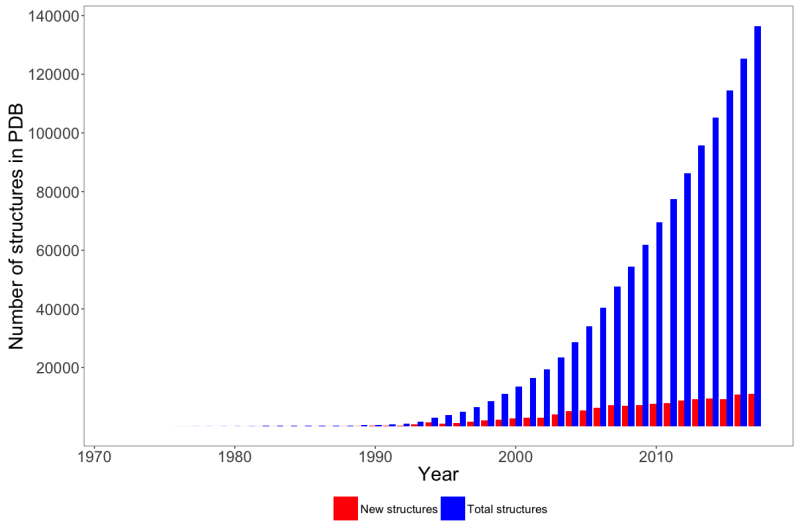
1.1.3 Available structures

The Worldwide Protein Data Bank (wwPDB; Berman et al. (2000)) is the main repository for submitting and retrieving known atomic models of proteins and nucleic acids. It currently contains over 138,000 structures (RCSB PDB, 2018a); approximately 89% of which were determined by X-ray crystallography, 10% by nuclear magnetic resonance (NMR) and over 1% were determined by electron microscopy. Figure 1.1 shows the total number of structures deposited to the PDB database in the last four decades.

However, not all the deposited structures are unique and therefore the total number of unique structures is lower than the 138,000 mentioned above. Specifically, if the structure sequences are used to remove sequence redundancy, the total number of unique sequence structures drops considerably as shown in table 1.1, which lists the total number of unique sequence structures as a function of the sequence identity threshold.

It is also worth noting that while the annual increase in new structures in the wwPDB is steadily increasing, the relative composition of experimental methods used to obtain the new structures is changing. This can be seen from figure 1.2, which shows the annual number of new structures submitted to the wwPDB

Figure 1.1: Growth of the wwPDB since 1972



This figure shows the growth of the PDB database between 1972 and 2017. The total number of structures deposited to the PDB database is shown in blue, while the annual change in the total number is shown in red. The data were obtained from RCSB PDB (2018b).

Table 1.1: Total number of unique sequences in the wwPDB

Sequence identity threshold	Number of unique sequences
100 %	75, 984
95 %	53, 121
90 %	50, 368
70 %	44, 097
50 %	37, 686
40 %	33, 289
30 %	28, 326

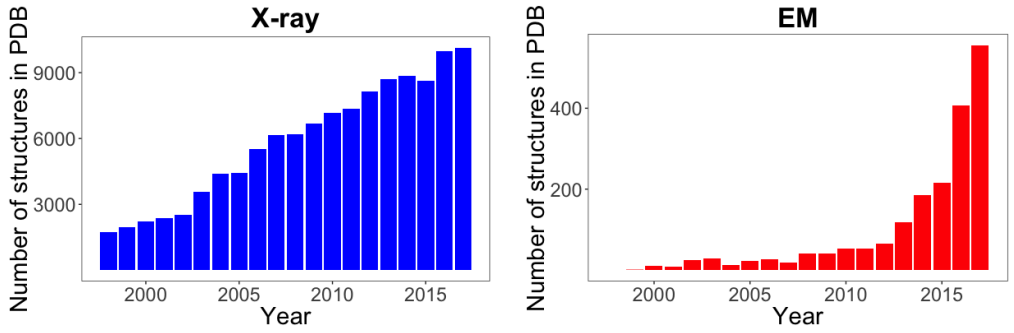
This table shows the total number of PDB database entries with unique sequence for different values of the sequence identity threshold. The sequence clustering was done using the `blast` software (Altschul et al., 1990). The data were obtained from RCSB PDB (2018c).

for the X-ray crystallography and the electron microscopy methods. From this figure, it is clear that the number of structures solved by X-ray crystallography increases linearly, while the number of structures solved by electron microscopy increases exponentially, at least in the past two decades. This fact, among other considerations, suggests a promising future for the cryo-EM method as discussed, for example, by Callaway (2015).

1.2 Project motivation

Given the amount of known protein structures available to the researchers, it is possible to use these data to derive empirical information about the protein

Figure 1.2: Growth of new structures by experimental method



This figure shows the annual number of structures solved by the X-ray crystallography (blue) and electron microscopy (red) methods in the past two decades. Note that the y-axes are different in the two graphs. The data were obtained from RCSB PDB (2018d) and RCSB PDB (2018e).

structures in general, beyond the information available in any structure on its own. Furthermore, multiple research projects have focused on this task, such as the PIDD database of protein inter-atomic distances (Wu et al., 2007), the CATH (Sillitoe et al., 2013) and the SCOP (Andreeva et al., 2008) databases of protein classification or the BALBES pipeline for molecular replacement (Long et al., 2008) to name just a few. Given the success of this approach so far, it is feasible that novel approaches and methods can be developed to further the determination of new structures using the already available prior information.

1.2.1 Project aims

The first step required to obtain the prior information from previously determined macromolecular structures is to be able to determine which structures are actually similar, so that the commonalities of these similar structures could be used as prior knowledge. To address this similarity detection problem, this project aims to develop a mathematical framework for comparing three-dimensional objects in a reliable and fully automated manner. The framework should be general enough to allow comparisons between any three-dimensional objects including co-ordinate files as well as density map files, while allowing for missing phase information (such as in the case of the Patterson maps), various resolutions of the structures and other typical macromolecular data features.

Moreover, the project aims to implement the developed mathematical framework into a stand-alone software tool, which will be made available to the community. This software tool needs to be sufficiently fast to allow finding similarities between macromolecular structures without the need for a large dedicated computational system. It also should be optimised for standard macromolecular data, both in terms of the algorithms as well as in terms of automatically making as many decisions as possible in order to minimise the amount of information the user needs to supply.

Once the mathematical framework and the software tool implementing it are developed, it will be possible to obtain distances between entries of any structural database. Furthermore, a clustering algorithm can be implemented to cluster the entries of any such database and therefore remove (or at least reduce) the redundancies in the database. As an example, the BALBES protein-domain database (Long et al., 2008), which is used in the BALBES molecular replacement pipeline, will be used (see section 1.3.2 for explanation why this database). If shape redundancies can be removed, then the molecular replacement pipeline would have decreased computational cost of searching routines in proportion to the number of removed redundancies, thus making the BALBES molecular replacement pipeline faster.

1.2.2 Possible applications

There are many other possible applications for a software tool capable of automatic shape-similarity detection in structural biology; however, they will not be developed as part of this project. One such possible application is that if a whole protein-domain database were to be searched against the whole wwPDB database of all known protein structures, then all parts of any of the wwPDB database entries not associated with a protein domain would be a novel protein domain from the point of view of the protein-domain database. Therefore, if an algorithm for deciding whether a fragment of a structure with no match in the protein-domain

database is a domain or just a small fragment (or domain linker) were developed, then automated domain detection could be achieved.

Another possible application for this project's outcome is finding possible co-ordinate-data matches in EM maps. This application stems from the project aims, as a software tool capable of comparing density maps and co-ordinate data could be used to search a density map, or its fragments, against a database of co-ordinate models and any matches could then be used as initial models for EM map model fitting. While it is true that there is already available software for fitting domains (or any other co-ordinate or density map data) into EM maps - such as `ESSENS` (Kleywegt and Read, 1997), `MOLREP` (Vagin and Teplyakov, 1997), `FFFEAR` (Cowtan, 1998), `FOLDHUNTER` (Jiang et al., 2001), `Situs` (Wriggers and Birmanns, 2001), `Modeller` (Eswar et al., 2006), `FOLD-EM` (Saha and Morais, 2012) - there is no software the author is aware of which would not require the user to supply the structural data that are to be fitted to the map.

Another possible application of the proposed software tool is searching for molecular replacement candidates using the shape information, either on its own, or in combination with the currently used sequence similarity approach. This application could also be developed almost immediately by using the intended software tool, assuming the tool could search for shape similarity in the Patterson maps space (*i.e.* Patterson maps (Patterson, 1934) are maps obtained from intensities instead of structure factors and their peaks correspond with interatomic vectors). This application could be especially useful in the case of protein domains with homologous structures but divergent sequence. While it is yet unclear how frequent such cases are, given that cases where similar sequences lead to different structures were reported by Kosloff and Kolodny (2008) as well as cases where similar structure is obtained using dissimilar sequences were reported by Lesk and Chothia (1980), it is reasonable to assume such cases do exist.

1.3 Protein domains

With the project defined, it is clear that both the development of the mathematical framework and its implementation as a software tool will require testing on biological macromolecular structure data. Given that some of the direct applications of this project can be accomplished by using a protein domain database, it seems appropriate to use a protein-domain database for the testing of the tool throughout its development. It is, therefore, worth looking into the definitions of protein domains and how they have been used in the field so far. In general, the term *protein domain* has been used in several different contexts; in the **structural context** for example by Richardson (1981), in the **functional context** for example by Rentzsch and Orengo (2013) and in the **folding context** for example by Wetlaufer (1973). In this thesis, the term protein domain will be used in the structural context, that is as an independent part of a protein with conserved tertiary structure.

1.3.1 Finding a definition of a protein domain

Regarding a more detailed definition, there has been a little consensus between the authors in the field. For example, the two well known protein domain databases, CATH (Orengo et al., 2002) and SCOP (Andreeva et al., 2008) were both created using a combination of manual approach with application of automated tools (although CATH has been created with more emphasis on the automated approach). Nonetheless, Csaba et al. (2009) reports that "out of the 27, 553 proteins which are classified in both hierarchies, for only 19, 266 (about 70%) the domain definitions are similar enough". This means that about 30% of the domain definitions are not comparable between the two databases.

The issue of different protein domain definitions becomes even clearer when the total number of domains is compared between different databases. Table 1.2 shows the total number of protein domains reported by different databases. From

this table it is clear that the numbers of reported protein domains vary wildly between different protein domain databases and by extension, so do the protein domain definitions.

Table 1.2: Total number of protein domains in different databases

Database name	Reference	Last update	Number of protein domains
ADDA	Heger and Holm (2003)	January 2006	1, 181, 071
BALBES	Long et al. (2008)	August 2017	13, 719
CATH	Orengo et al. (2002)	January 2018	461, 130
CDD	Marchler-Bauer et al. (2017)	March 2017	56, 066
Dali Domain Dictionary	Holm and Sander (1998)	January 2001	3, 724
ECOD	Cheng et al. (2014)	November 2017	578, 136
PDBeFOLD	Krissinel and Henrick (2014)	April 2014	131, 362
Pfam	Finn et al. (2014)	March 2017	16, 712
SCOP	Andreeva et al. (2008)	June 2009	110, 800
SCOPe	Fox et al. (2014)	December 2017	274, 253

This table shows the total number of protein domains as reported by the different protein domain databases using their definition of protein domains.

The conclusion to draw from this discussion is that there is no single clear definition of protein domains even in the structural context. In terms of this project and more specifically in terms of the intention of testing the software tool on protein domain data, it seems reasonable to use the same protein domain definition as the database which will be used for this purpose. Therefore, such starting point now needs to be decided.

1.3.2 Selecting the protein-domain database

In order to decide the starting point database for this project, the databases listed in table 1.2 were considered. Given that the project database should be based on purely structural similarity, it seems reasonable that the starting point database should give priority to the structural information over the sequence alignments and functional considerations. Therefore, the BALBES database of Long et al. (2008) was selected as the starting point of this project.

1.3.3 BALBES protein-domain database

The BALBES molecular replacement database was created as follows: All wwPDB entries available in 2008, which were determined using X-ray crystallography, refined against resolution better than 3.5 Å and had length of at least 15 amino acids were obtained and manually analysed for domains. Specifically, Long et al. (2008) describe the protein domain detection as follows: "All domains were analysed and checked manually. The main criteria for domain definition were three-dimensional compactness and separability from other parts of the subunit".

Consequently, the redundancy of protein domains obtained was reduced using both the sequence information and the structural information. This was achieved by using the Kabsch algorithm (Kabsch, 1976) to determine the minimal relative root-mean-square deviation (RMSD) and the Needleman & Wunsch algorithm (Needleman and Wunsch, 1970) to align the domain sequences. Two domains were then considered identical if their minimal relative RMSD (of C^{α} atoms) was ≤ 1.0 Å **and** the two domains had sequence identity of 80% or more. The resulting database of protein domains contains 13, 719 domains; however, there is still some structural redundancy in the database.

1.4 Shape-similarity detection methods

Now, the final part of this introductory chapter will be dedicated to a brief discussion of the available shape-similarity detection methods. Although not all of the discussed methods will be used in the final software, the decision should include understanding of the alternatives that could have been explored instead.

1.4.1 Root-mean-square deviation (RMSD)

One of the best known approaches to finding structural similarity, at least in the field of structural biology, is the root-mean-square deviation (RMSD) distance.

RMSD measure is based on one-to-one correspondence between points defining the two compared 3D objects and it is a prime example of a landmark-based shape-comparison method. Assuming that the one-to-one correspondence holds, the Kabsch algorithm (Kabsch, 1976) - also known as the *procrustes* superimposition - can be used to find the optimal superimposition in terms of the minimal squared distance between the corresponding points. In other words, the Kabsch algorithm finds the rotation and translation which places one structure so that the RMSD distance to the other structure is minimised; then, the RMSD distance is computed by the following equation:

$$RMSD(A,B) = RMSD(B,A) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} |\delta_{A_iB_i}|^2} \tag{1.2}$$

where:

N: is the total number of compared points.

$\delta_{A_iB_i}$: is the distance measure, typically the Euclidean distance (for Cartesian co-ordinates) from point A_i to point B_i.

The one-to-one correspondence assumption

One of the issues with this measure of similarity is the assumption of one-to-one correspondence, as this clearly does not hold for most protein domain pairs (or protein pairs in general). To extend the RMSD measure to pairs of structures which differ in the number of points and therefore do not comply with the one-to-one correspondence requirement, a typical approach is to obtain structural alignment of smaller fragments and calculate the RMSD for aligned fragments only. Different authors who implemented such measures use varied techniques to obtain structural alignment; however, all of these introduce new parameters to determine the mismatch and gap penalties.

For example, the CE structural aligner (Shindyalov and Bourne, 1998) requires determining the number of allowed gaps, two fragment similarity thresholds

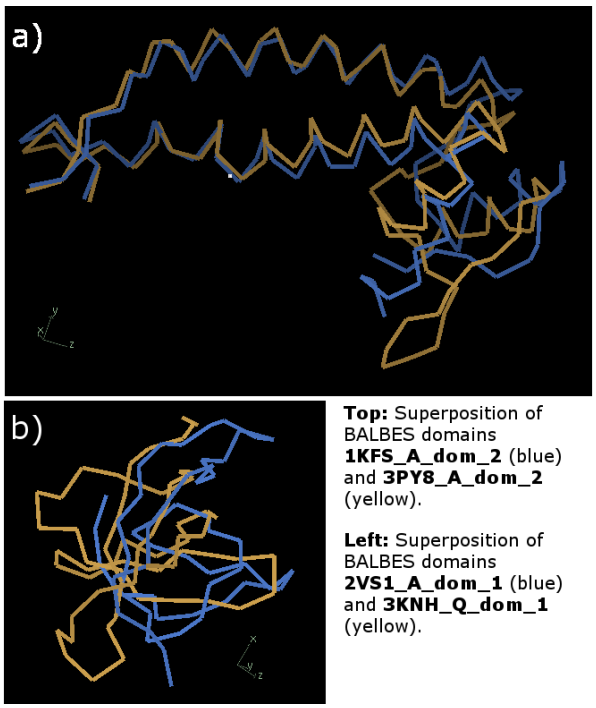
and the length of aligned fragments, while FATCAT (Ye and Godzik, 2003) required the length of aligned fragments, as well as three different thresholds (the twist penalty, mismatch penalty and gap penalty). Similarly, the Dali *Z-score* (Holm and Rosenstrom, 2010) requires a similarity threshold, envelope function (which down-weights long distance pairs) and "strong match" threshold, and ProSMART (Nicholls et al., 2014) uses fragment length and gap penalty. For further review of protein structural aligners, see for example Hasegawa and Holm (2009).

RMSD Size Dependence

Another issue with using the RMSD as general shape similarity measure is its dependence on the number of points that are used in the comparison (*i.e.* the size of the compared objects). Figure 1.3 demonstrates one such case using two medium sized BALBES database domains and two small sized domains. The RMSD distance between the pair of the medium sized domains is approximately 3.53 Å (Figure 1.3 part **a)**), while the RMSD distance between the small domains is approximately 3.48 Å (Figure 1.3 part **b)**). This example demonstrates that two small and rather different shapes can have a smaller RMSD value than two similar, but larger shapes; therefore, the RMSD value needs to be normalised using the size of the compared shapes if it is to be used as general shape descriptor.

Nonetheless, the issue of size-dependence of the RMSD measure has been known for some time now and as a result, many authors have attempted to modify the measure to account for the size difference. There are multiple reasons for the RMSD dependence on the size of the compared structures as discussed, for example, by Carugo (2007); perhaps the main cause of the size dependence is that due to RMSD using rigid structures, any bends in the protein backbone will cause large difference in the RMSD value and the larger the size of the compared structures, the more bends are likely to occur, as suggested by Carugo (2007).

Figure 1.3: Medium and small domain pairs with very similar RMSD value



Part **a)** shows the SSM (Krissinel and Henrick, 2004) superposition of the C^{α} traces of two medium size BALBES domains (1KFS_A_dom_2 with 867 atoms and 3PY8_A_dom_2 with 929 atoms) as visualised by Coot (Emsley et al., 2010). Part **b)** shows the same superposition and visualisation for two small domains (2VS1_A_dom_1 with 442 atoms and 3KNH_Q_dom_1 with 565 atoms). The RMSD value for both these pairs is rather similar, with the small domains distance being 3.48 Å, while the medium sized domain distance being 3.53 Å.

RMSD Normalisation

The RMSD normalisation approaches include, for example, the SARF2 normalised S score (Alexandrov and Fischer, 1996) calculated as per equation 1.3, the $RMSD\_100$ score (Carugo and Pongor, 2001) obtained by using equation 1.4 or the r_f score (Carugo, 2007) defined in equation 1.5.

$$S(A,B) \;=\; S(B,A) \;=\; \frac{3 \times N}{1 + RMSD(A,B)} \tag{1.3}$$

$$RMSD\_100(A,B) \;=\; RMSD\_100(B,A) \;=\; \frac{RMSD(A,B)}{1 + ln\,\left(\sqrt{\frac{N}{100}}\right)} \tag{1.4}$$

$$r_f(A,B) \;=\; r_f(B,A) \;=\; 0.148 \;+\; 4.233e^{(\;-0.179\;\times\;RMSD\_100(A,B)\;)} \tag{1.5}$$

where:

N: is the total number of compared points.

$RMSD(A,B)$ is defined as per equation 1.2.

In order to determine the effects of these RMSD normalisation approaches, 1, 441, 838 RMSD comparisons were computed between the BALBES database domains. Note that these are not all possible combinations, but rather a random selection of these. Consequently, the total length of the aligned fragments was plotted against the RMSD value, RMSD value normalised by the alignment length and the three aforementioned RMSD normalisation approaches. The resulting plots are shown in figure 1.4.

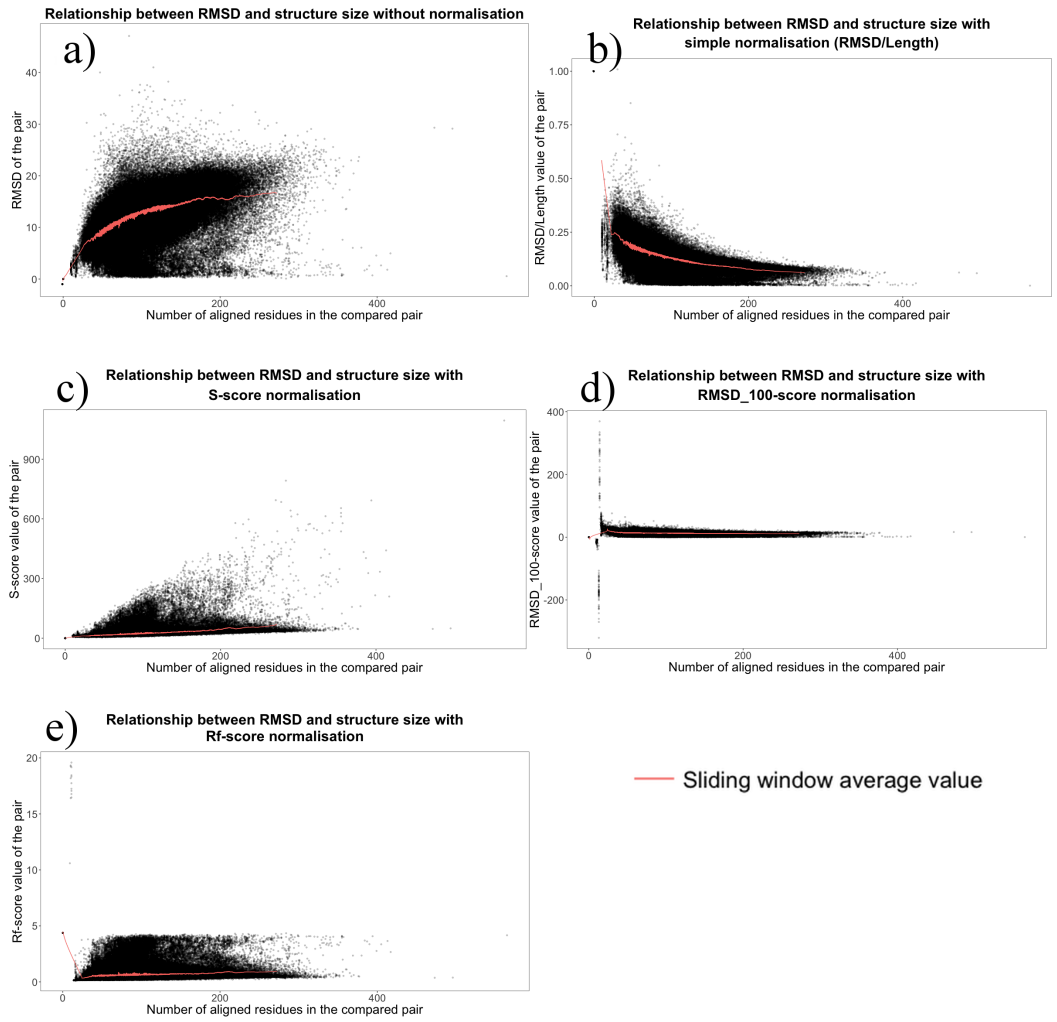
If there was no relationship between the total length of the aligned fragments and the distance measure, the sliding window average would be expected to be close to a flat line. Nonetheless, part a) of figure 1.4 shows that this is not the case for the RMSD measure on its own. Furthermore, part b) of the same figure demonstrates, that this relationship is not simply proportional to the total number of aligned residues, as there is still a relationship when the RMSD is normalised by the alignment length. On the other hand, parts c), d) and e) show that the RMSD normalisation approaches of Alexandrov and Fischer (1996), Carugo and Pongor (2001) and Carugo (2007) (respectively) do mostly remove this relationship, at least for the test dataset.

Further features of the RMSD measure

The previous discussion of the RMSD measure shows that with normalisation and proper structural alignment, the RMSD distances can be used to compute shape similarity. However, computing the RMSD distances for the test dataset of approximately 1.5 million pairs has shown that the computational cost of these calculations is rather high. This is caused by the impossibility of pre-computing the structural alignments; in other words, the structural alignments need to be computed for each pair of structures anew.

The result of these facts is that if all BALBES domains were to be compared against each other, this would require $^{13,719 \times 13,718}/_{2} = 94,098,621$ comparisons. Assuming one comparison (including the structural alignment) can be optimised

Figure 1.4: Effect of different RMSD normalisation techniques



All five plots show the effect of changing the number of aligned residues (representing the total structural pair size) on the magnitude of the similarity score. The plots were produced using the RMSD values obtained by random selection of $1,441,838$ unique pairs of BALBES domains. Plot **a)** shows the unaltered RMSD measure, plot **b)** shows the simple per residue normalisation, plot **c)** shows the `SARF2` S-score measure (Alexandrov and Fischer, 1996) as per equation 1.3, plot **d)** shows the $RMSD\_100$ measure (Carugo and Pongor, 2001) and equation 1.4 and plot **e)** shows the r_f-score (Carugo, 2007) defined in equation 1.5. Note that different similarity measures do **not** have the same scale and therefore the plots have different y-axis scales. The red lines in all plots show the sliding window average values.

to take 0.2 second, the total required time would be ≈ 217.8 days. While it is true that these calculations can be done in parallel, this is still a rather computationally costly computation. Therefore, it seems worth exploring alternative approaches to shape similarity detection.

1.4.2 Interatomic distances

One possible alternative approach to finding shape similarity is to compute all the interatomic distances in a structure. The resulting distribution of interatomic distances then can serve as a general descriptor of the shape. In order to compare two such distributions and obtain a structure similarity distance, it is possible to use, for example, the central-moments distribution descriptors; these are computed as shown:

$$n^{th}\ central\ moment\ =\ \frac{1}{N-n}\sum_{i=1}^{N}|x_i-\mu|^n \tag{1.6}$$

where:

N is the number of values.

μ is the mean value.

There are several advantages of using the interatomic distances distribution as shape similarity measure; one such advantage is that no one-to-one correspondence is required to produce comparison of a structure pair. Furthermore, the interatomic distances distribution can be pre-computed for each structure separately, as well as the distribution moments. This feature means that once the interatomic distance distribution are computed, comparison of any pair of structures should be possible in a very short time. Moreover, there even already exists a database of interatomic distances computed for protein structures - PIDD (Wu et al., 2007).

Another advantage of this shape descriptor is that it is intrinsically rotation invariant as well as translation invariant. These features stem from the fact that vector lengths are rotation and translation invariant and therefore so is their distribution and its descriptors. It is worth noting that the same advantage does apply to the RMSD measure as well. Finally, it is interesting to consider the possibility of weighting the interatomic distances according to their length; such approach could emphasise the overall shape similarity if the longer interatomic distances

were weighted more than the short distances, while the detailed shape similarity would be the focus of a descriptor up-weighting the short interatomic distances.

Regarding the disadvantages, the interatomic distances are, by definition, based on the positions of atoms. Therefore, it would be difficult to apply them to density maps; an issue that could be addressed by peak detection in the maps in order to find atoms or by using all map grid points and weighting the distances based on the density value. However, both these suggested solutions would greatly increase the computational cost of the comparison.

1.4.3 Voxelisation

Voxelisation method for shape description comes from the computer graphics field, where objects are rendered using lower number of voxels (three-dimensional pixels) when far from camera to save rendering time. From this idea, it can be seen that objects shape can be described by the change the object undergoes as its voxelisation (*i.e.* the number of voxels it is divided into) changes. In other words, distribution of the object co-ordinates (or density) within a given voxelisation grid can be computed, for example using the distribution central moments (equation 1.6). This can be repeated for different voxelisation grids to obtain matrix A.

$$\begin{array}{l} \\ Voxelisation\ 1 \\ Voxelisation\ 2 \\ \vdots \\ Voxelisation\ x \end{array} \begin{array}{c} \begin{array}{cccc} 1^{st}\ Moment & 2^{nd}\ Moment & \dots & n^{th}\ Moment \end{array} \\ \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{x1} & a_{x2} & \dots & a_{xn} \end{pmatrix} \end{array} = A \tag{1.7}$$

Once the matrix A is computed, it is possible to calculate the change in the distribution of central moments. This change is direct function of the shape of the described object; and therefore, by calculating descriptor matrix A_D, a numerical description of shape is obtained.

$$\begin{array}{l} \\ Vox.\ 1 - Vox.\ 2 \\ Vox.\ 2 - Vox.\ 3 \\ \vdots \\ Vox.\ (x-1) - Vox.\ x \end{array} \begin{array}{c} \begin{array}{cccc} 1^{st}\ Moment & 2^{nd}\ Moment & \dots & n^{th}\ Moment \end{array} \\ \begin{pmatrix} a_{11} - a_{21} & a_{12} - a_{22} & \dots & a_{1n} - a_{2n} \\ a_{21} - a_{31} & a_{22} - a_{32} & \dots & a_{2n} - a_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{(x-1)1} - a_{x1} & a_{(x-1)2} - a_{x2} & \dots & a_{(x-1)n} - a_{xn} \end{pmatrix} \end{array} = A_D \quad (1.8)$$

It is worth noting that while the method is typically used with cubical voxels, there is the possibility to use differently shaped voxels as well. One interesting possibility, therefore, is to use spherical voxels (that is defining a voxel as space between two concentric spheres with different radii); such definition leads to intrinsically rotation invariant description of objects. The voxelisation method seems not to be frequently used for macromolecular structures, although Tsukamoto et al. (2009) have used it for protein cavity detection.

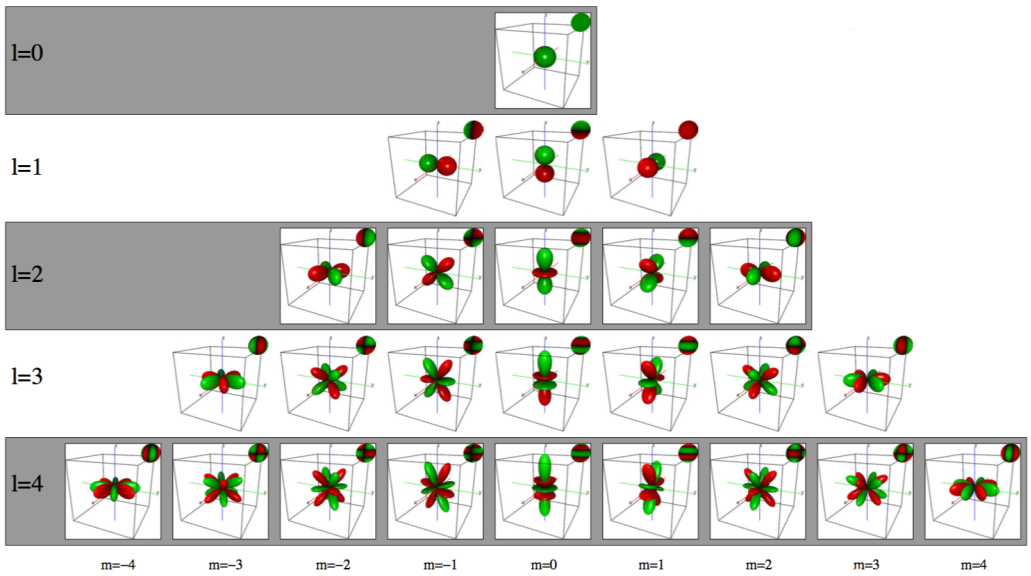
Regarding the advantages of this method, the A matrices can be pre-computed for any structure in advance and therefore the distance computation of a pair of structures becomes simply computation of the A_D matrix from two A matrices. Also, this method is directly amenable to be applied to map data as well as the co-ordinate data; and furthermore, as mentioned in the previous paragraph, by using the spherical voxels, the descriptor can be made rotation invariant - assuming that the centre of the structures is used to place the spherical voxels around.

Nonetheless, there are also some disadvantages to this method; firstly, it is not translation invariant and presumably can be sensitive to the grid placement (i.e. differently placed grids could lead to different numerical descriptors). Also, its rotation invariance is based on spherical voxels, usage of which may lead to loss of information about the shape.

1.4.4 Spherical-harmonics expansion

Spherical harmonics are an infinite set of orthonormal functions defined on the surface of a unit sphere. The infinite set is parametrised by two integer arguments, l (the band) and m (the order). The arguments are constrained by $l \geq 0$ and $m \in [-l, l]$. Generally, as the band of the spherical harmonic function increases, the more detailed the function becomes; this is demonstrated in figure 1.5.

Figure 1.5: First few real spherical harmonic functions



The first five real-valued spherical harmonic bands are shown with all their orders. Note that although spherical harmonics are defined on sphere surface, here they are depicted as the normals of the surface value and therefore as being defined in 3D. Red colour signifies negative values, while green mean positive. The little spheres in the right hand corners show the spherical harmonics values on the unit sphere surface. Figure adapted from Schonefeld (2005).

Formally, spherical harmonic functions are defined as:

$$Y_l^m(\theta, \phi) = N_l^m \times P_l^m(cos(\theta)) \times e^{im\phi} \tag{1.9}$$

where:

$Y_l^m(\theta, \phi)$ is the spherical harmonics function for band l, order m inclination angle θ and azimuthal angle ϕ.

N_l^m is the normalisation coefficient defined in equation 1.10.

P_l^m is the associated Legendre polynomial, defined in equation 1.11.

with the normalisation coefficient N and the associated Legendre polynomials P being defined by the following equations:

$$N_l^m = \sqrt{\frac{2l+1}{4\pi} \times \frac{(l-m)!}{(l+m)!}} \tag{1.10}$$

$$P_l^m\left(cos\left(\theta\right)\right) = \frac{(-1)^m}{2^l l!} \times \sqrt{\left(1-x^2\right)^m} \frac{d^{l+m}}{dx^{l+m}} \left(x^2-1\right)^l \tag{1.11}$$

Regarding the shape similarity detection, the spherical harmonics expansion can be thought of as a Fourier transformation done on the surface of a sphere. In other words, the spherical harmonics coefficients obtained as per equation 1.12 signify the weight with which the particular spherical harmonics function needs to be applied to sphere surface in order to reproduce the original (*i.e.* expanded) function.

$$c_{l,m} = \int_{\theta=0}^{2\pi} \int_{\phi=0}^{\pi} \left(Y_l^m\left(\theta,\phi\right) \times A(\phi,\theta)\right)\ d\phi\ d\theta \tag{1.12}$$

where:

$A(\phi,\theta)$ is the unit sphere mapping of the object to be expanded into spherical harmonics coefficients.

$c_{l,m}$ is the spherical harmonics coefficient for band l and order m.

Regarding the advantages of spherical harmonics expansion, since the method expands a function defined on a sphere surface, it should be directly applicable to density maps, as these are functions defined in three-dimensional space. Also, the spherical harmonics expansion can be pre-computed for each structure separately and the computation of distance between two shapes should then be limited to comparison of several matrices, a relatively inexpensive computational task.

In terms of the rotational invariance, the spherical harmonics coefficients are not intrinsically rotationally invariant. Nonetheless, Kazhdan et al. (2003) have shown that the so called *energy levels* (descriptors derived from the spherical har-

monics coefficients by summing over the order values for each band) computed as per equation 1.13 do become rotationally invariant; this follows from the fact that the sum of frequencies in any band l does not change under rotation. Therefore, the spherical harmonics expansion can be made rotationally-invariant at the cost of to the loss of information caused by summing over the orders.

$$\pi_l = \sqrt{\sum_{m=-l}^{l} |c_{l,m}|^2} \tag{1.13}$$

The main disadvantages of the spherical harmonics expansion include translation variance and the resulting need to centre the structures before comparing them. Also, since the spherical harmonics functions are defined on the surface of a sphere and not for the volume of a sphere, the three-dimensional data need to be mapped onto a sphere surface. This process is bound to lead to loss of information, although it can be minimised by using multitude of differently placed spheres.

Finally, it is worth noting that the spherical harmonic expansion is extensively used in the image analysis field (presumably due to its predisposition to deal with 2D data), for example by Yotter et al. (2011) for correcting topological artefacts on cortical surfaces of human brain images. It has, furthermore, been used for several different purposes in the bioinformatics field, for example, by Cai et al. (2001) to filter high throughput screening for protein-ligand interactions, by Morris et al. (2005) for binding pocket and ligand comparisons or by DiMaio et al. (2009) for molecular recognition in density maps.

1.4.5 3D Zernike Moments

The 3D Zernike moments expansion is similar to the spherical harmonic expansion described in the previous section (1.4.4), except that the basis functions (known as 3D Zernike polynomials) are now a combination of spherical harmonics functions

defined in equation 1.9 and radial function R defined in equation 1.15. The formal definition of 3D Zernike polynomials, as derived in Canterakis (1996), is as follows:

$$Z_{l,m,n}(r,\theta,\phi) = R_n^m(r) \times Y_m^l(\theta,\phi) \tag{1.14}$$

where:

l is a non-negative integer $l \geq 0$.

m is an integer $m \in [-l, l]$.

n is an integer $n \in [-m, m]$.

$Z_{l,m,n}$ is the 3D Zernike polynomial.

r is the radial distance from centre of sphere.

θ is the inclination angle.

ϕ is the azimuthal angle.

R_l^m is the radial polynomial defined in equation 1.15.

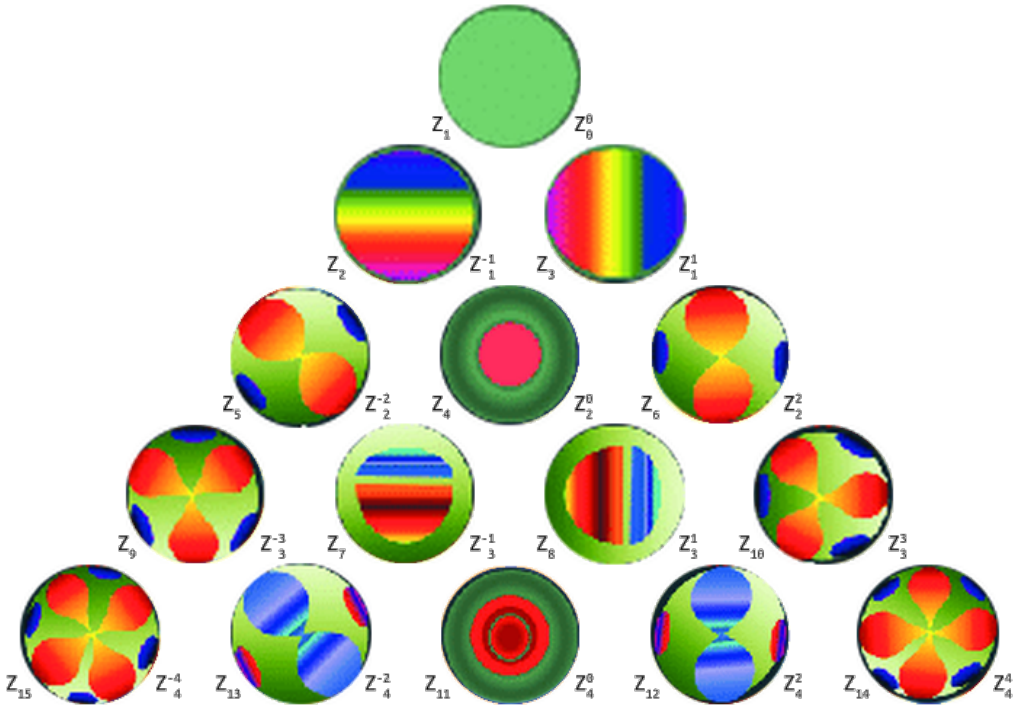
Y_m^n is the spherical harmonic function defined in equation 1.9.

$$R_n^m(r) = \sum_{k=0}^{\frac{n-m}{2}} \left(\frac{(-1)^k \times (n-k)!}{k! \times \left(\frac{n+m}{2} - k\right)! \times \left(\frac{n-m}{2} - k\right)!} \times r^{n-2k} \right) \tag{1.15}$$

The radial function comes from the 2D Zernike polynomials (Zernike, 1934), which are parametrised exactly as the spherical harmonics discussed above, that is $l \geq 0$ and $m \in [-l, l]$; although they are defined on a unit disk as shown in figure 1.6.

The difference between the spherical harmonics expansion and the 3D Zernike moments expansion is in the presence of the radial function, which allows 3D Zernike moments expansion to encompass the whole unit sphere and not only its surface (in other words, the 3D Zernike polynomials are defined in unit sphere as well as on its surface). Consequently, equation 1.16 can be used to expand any function defined in three-dimensional Cartesian space onto its 3D Zernike moments

Figure 1.6: First five 2D Zernike polynomial bands



All orders for first five bands of the 2D Zernike polynomials are shown. Red colour means negative value, while blue signifies positive values. Figure adapted from Prakash and Jhanji (2016).

coefficients ($c_{l,m,n}^{Zernike}$); which similarly to the spherical harmonics coefficients can be seen as the weights which need to be applied to the particular 3D Zernike polynomial functions in order to reproduce the expanded three-dimensional function from a unit sphere.

$$c_{l,m,n}^{Zernike} = \int_{r=0}^{1} \int_{\theta=0}^{2\pi} \int_{\phi=0}^{\pi} \left(Z_{l,m,n}\left(r,\theta,\phi\right) \times A(r,\phi,\theta) \right) \; dr \; d\theta \; d\phi \tag{1.16}$$

where:

$A(r,\phi,\theta)$ is the spherical co-ordinates mapping of the object to be expanded.

The shape similarity detection features of the 3D Zernike moments expansion are very similar to the spherical harmonics expansion; specifically, due to the inherent expansion of a function defined in three-dimensional space, the 3D Zernike moments expansion is well suited for expanding density maps. Furthermore, it is also well suited for pre-computing the $c_{l,m,n}^{Zernike}$ coefficients for each structure

separately and consequently computing distances between pairs of structures by comparing two three-dimensional arrays of coefficients.

Identically to the spherical harmonics coefficients, the 3D Zernike moments coefficients are not rotation invariant. Nonetheless, intrinsically rotation invariant representation can be obtained by using the following formula discussed, for example, by Grandison et al. (2009).

$$\pi_{l,n}^{Zernike} = \sqrt{\sum_{m=-l}^{l} \left| c_{l,m,n}^{Zernike} \right|^2} \tag{1.17}$$

where the sum is taken over all defined values of m for given combination of l and n.

The main disadvantage of the 3D Zernike moments expansion is its translation variance. Also, similarly to the spherical harmonics expansion, it needs to be seen how much information is lost when the equation 1.17 is applied to obtain rotation invariance, as information about one dimension is lost in the process.

Examples of structural biology application of 3D Zernike moments include protein tertiary structure description using molecular surface by Sael et al. (2008), ligand density reconstruction by Gunasekaran et al. (2009), description of functional movement in macromolecular structures by Grandison et al. (2009) or determination of fluctuation scattering profiles for free-electron laser crystallography by Liu et al. (2011). It is also worth noting that the 2D Zernike moment expansion was shown by Teh and Chin (1988) to outperform other moments based methods in signal to noise ratio recovered when applied to 2D images.

1.4.6 Further shape-similarity detection methods and software

It should be noted that while all the methods described in the previous sections are described in terms of their mathematical basis, this list is not complete and there are many other methods that could have been explored as well. Furthermore,

there are many applied computational methods described in the literature for similar tasks as described in the project aims section (1.2.1).

In terms of describing the 3D protein shapes, there are several already published methods; for example, Rogen and Bohr (2003) described a method based on integral formulas of the Vassiliev knot invariants or Daras et al. (2006) have presented a method based on spherical-trace transform of atomic co-ordinates. Although both these methods are not applicable to density maps and in this sense are different from the aims of this thesis, they demonstrate the fact that protein classification is an ongoing and important problem in the field.

Similarly, there are multiple methods and associated software for matching co-ordinate data and density maps to each other; examples of such methods include `ESSENS` (Kleywegt and Read, 1997), `MOLREP` (Vagin and Teplyakov, 1997), `FFFEAR` (Cowtan, 1998), `FOLDHUNTER` (Jiang et al., 2001), `Situs` (Wriggers and Birmanns, 2001), `Modeller` (Eswar et al., 2006), `FOLD-EM` (Saha and Morais, 2012). Moreover, most of the aforementioned software does have some measure of similarity (albeit maybe not defined as such); however, the structure matching software is typically optimised for direct matching of two structures and not for multiple comparisons. This means that the computational cost of using any of the aforementioned software is much higher than just using the RMSD and therefore none of this software is really suited for the purposes of this project.

1.4.7 Selecting a shape-similarity detection method

The previous sections (1.4.1 to 1.4.5) have discussed the backgrounds, advantages and disadvantages of different methods for similarity detection between pairs of shapes. This list is, however, not exhaustive and other methods do exist. Nonetheless, the decision as to which of these (if any) methods should be used to implement the shape similarity measure upon which this project is to be based, needs to be taken.

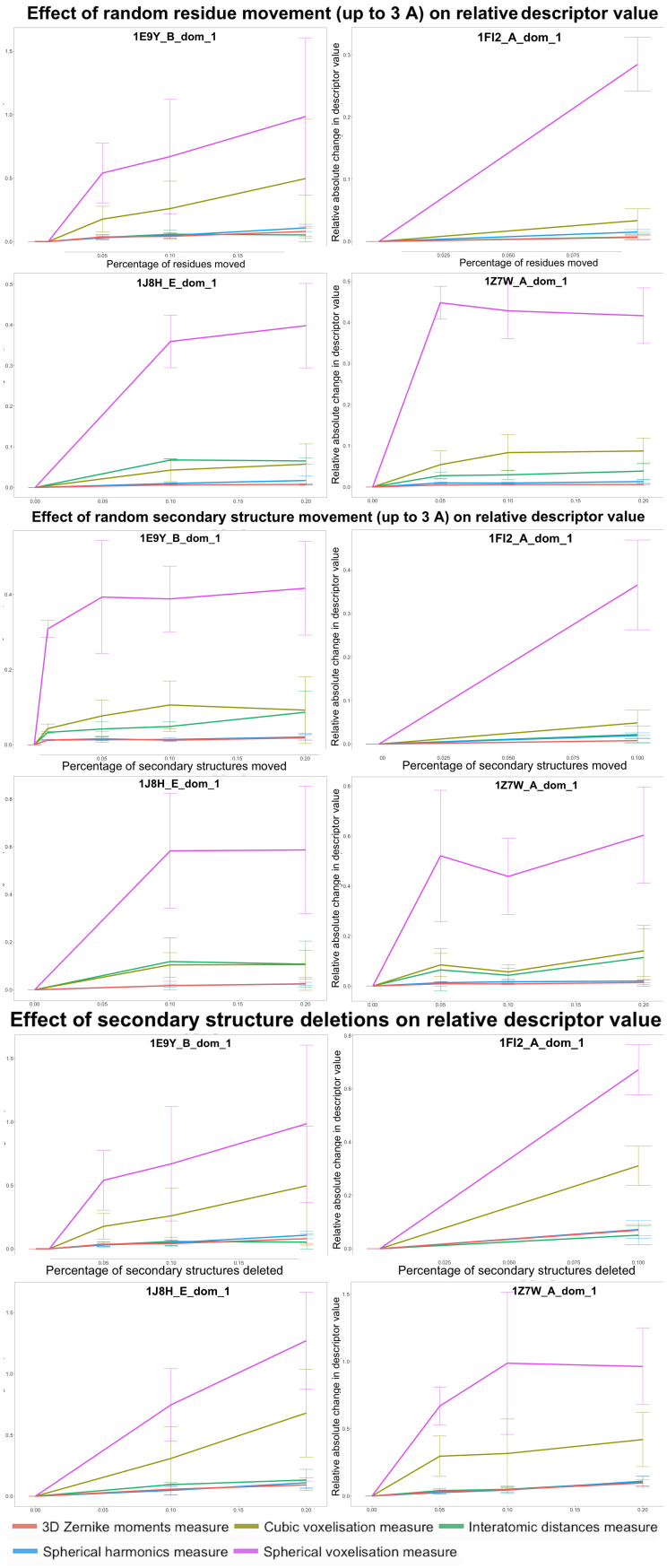
To help with the decision, a procedure was implemented to modify any PDB file containing a macromolecular structure in one of the following ways: a) random movement of a given percentage of the residues, b) random movement of a given percentage of the secondary structures as determined by the SSM software of Krissinel and Henrick (2004) and c) random deletion of a given percentage of secondary structures as determined by the SSM.

Next, four structures were randomly selected from the BALBES database (1E9Y_B_dom_1, 1FI2_A_dom_1, 1J8H_E_dom_1 and 1Z7W_A_dom_1) and the procedures a), b) and c) were repeatedly (as the residue and secondary structure selection is stochastic) applied on their own as well as in combination. The resulting modified structures were then idealised using the REFMAC5 (Kovalevskiy et al., 2018) idealisation procedure to resolve any modifications breaking the biochemical prior knowledge.

Finally, the following shape-similarity measures were computed for the four structures and all their modifications: interatomic-distances distribution differences, cubical-voxelisation distribution differences, spherical-voxelisation distribution differences, spherical-harmonics expansion differences and 3D-Zernike-moments expansion differences. The relative change in the descriptor value with respect to the percentage of the structure elements affected was then calculated. The RMSD measure was not used as its disadvantages (difficult extension to density maps and impossibility of pre-computation of results) were considered too constrictive for the purpose of this project. The resulting plots are shown in figure 1.7.

By examining the plots in figure 1.7, several interesting conclusions can be drawn about the robustness of the descriptors. Firstly, the spherical-voxelisation distribution-difference descriptor is very sensitive to all three different structure modifications for all four different structures. In the extreme case of deleting 20% of the secondary structures of the 1J8H_E_dom_1 domain, the descriptor value have on average more than doubled. It can therefore be concluded that the spherical-

Figure 1.7: The effect of various structure modification on the relative change in descriptor values



This figure is composed of 12 plots. The top 4 plots contain the effect of randomly moving a given percentage (x-axis) of the structure residues up to 3 Å and consequently comparing the descriptor values between the original and modified structures for the 4 different and randomly selected structures. The 5 different lines show the development of relative descriptor value change for the 5 different descriptors - the interatomic distances distribution central moments difference (in bright green), the cubic voxelisation distribution central moments difference (in mossy green), the spherical voxelisation distribution central moments difference (in pink), the spherical harmonics expansion coefficients difference (in blue) and the 3D Zernike moments expansion coefficients difference (in red); while the error bars represent a single standard deviation in both directions. Similarly, the middle 4 plots show the descriptor relative value change in response to random percentage of secondary structures (for secondary structures as detected by the SSM (Krissinel and Henrick, 2004)) being moved up to 3 Å. The error bars represent a single standard deviation in both directions. Finally, the last 4 plots show the effect of randomly deleting a given percentage of secondary structures as determined by the SSM. The error bars represent a single standard deviation in both directions.

voxelisation descriptor would not be appropriate method for this project, as it is too sensitive to small changes in the shape.

Furthermore, the cubic-voxelisation distribution-difference shape-descriptor also appears to be quite sensitive, especially to random residue movement and secondary structure deletions, where for structure 1E9Y_B_dom_1 the relative descriptor value changed more than 50% when 20% of the residues were moved up to 3 Å, or when 20% of secondary structures were removed. This leads to the conclusion that the cubic-voxelisation descriptor is too sensitive to small changes in the shape and therefore not optimal for the purposes of this project.

Regarding the interatomic-distances distribution-difference descriptor, it seems quite robust to the secondary structure deletions for all four structures. However, it also appears almost as sensitive to secondary structure movement as the cubic-voxelisation descriptor just discussed. This is demonstrated by the plot for structure 1Z7W_A_dom_1, where simple movement of only 5% of secondary structures caused change in the relative descriptor value by almost 10% on average. Therefore, it is concluded that this descriptor may not be optimal for the purposes of this project.

Finally, regarding the spherical-harmonics and the 3D-Zernike-moments expansion descriptors, they both seem quite robust to small structural changes and figure 1.7 does not offer much ground to distinguish between the two methods, at least with this limited sample. Therefore, in order to decide with which method the project should be continued, the availability of computational libraries for implementing and using the method was considered.

In terms of this criterion, the author could not find a freely available implementation of 3D Zernike moments expansion aside from the `SASTBX` (Liu et al., 2012) software which uses a combination of C++ and Python languages to implement the expansion. On the other hand, for spherical harmonics expansion calculations, there is the `SOFT2.0` library written by Kostelec and Rockmore (2007) in the C language as well as several other libraries. Since the implementa-

tion of a software tool is one of the outcomes of this project and it is intended to be done in the C/C++ programming language and given that it is more convenient for the author to link together C and C++ code that it is to link together C++ and Python code, the spherical harmonics expansion shape descriptor was selected as the basis of this project.

Chapter 2

Spherical harmonics expansion

This chapter discusses the spherical harmonics functions in greater details in order to provide the background required to explain details of the mathematical framework for computing distances between pairs of shapes. Given that the spherical harmonics functions are to be used as basis functions for expansion, it is worth exploring how are they derived.

2.1 Spherical harmonics derivation

The problem that is being addressed by the spherical harmonics functions can be defined as follows: given that the discrete Fourier transform method in Cartesian space uses the sin $(2\pi ik/N)$ and cos $(2\pi ik/N)$ basis functions, (where $2\pi i$ is a constant, k is the wavelength and N is the dimensionality of the discrete Fourier transform), what are the basis functions for the discrete Fourier transform in spherical co-ordinate space?

To answer this question, one of the tenets of the Sturm-Liouville theory offers a good starting point; specifically, that the normalised eigenfunctions form an orthonormal basis in a Hilbert space (*i.e.* any finite or infinite vector space where calculus and inner products of vectors can be computed) - for detailed discussion of the Sturm-Liouville theory see, for example, Zettl (2005). With

this information, it is interesting to also note that basis functions of the Fourier transform in Cartesian space (*i.e.* sin $(2\pi ik/N)$ and cos $(2\pi ik/N)$ or by Euler's identity also written as $e^{2\pi ik/N}$) are actually the solution to the Laplacian operator in Cartesian space.

$$\boldsymbol{\nabla^2} = \nabla_x^2 + \nabla_y^2 + \nabla_z^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2} \tag{2.1}$$

where:

$\boldsymbol{\nabla^2}$ is the Laplacian operator **vector**.

∇_x^2 is the Laplacian operator along the *x-axis* .

∂ is the partial derivative (and ∂^2 is the second partial derivative).

Therefore, in order to construct an orthonormal set of basis functions in the spherical co-ordinate system, the same approach could be used, that is, the eigenvectors of the Laplacian operator in the spherical co-ordinate space need to be computed and normalised. Now, using the mapping from Cartesian to spherical co-ordinates $f : (x, y, z) \mapsto (r, \theta, \phi)$ and its reverse, the co-ordinate conversion equations can be stated as follows:

$$\begin{aligned}
r &= \sqrt{x^2 + y^2 + z^2} \\
\theta &= \sin^{-1}\left(\frac{z}{r}\right) \\
\phi &= \tan^{-1}\left(\frac{y}{x}\right) \\
x &= r \times \cos(\phi) \times \cos(\theta) \\
y &= r \times \sin(\phi) \times \cos(\theta) \\
z &= r \times \sin(\theta)
\end{aligned} \tag{2.2}$$

where:

r is the radial distance from the centre of co-ordinates.

θ is the inclination angle.

ϕ is the azimuthal angle.

With the mapping specified, the Laplacian operator in spherical co-ordinates can now be written as:

$$\boldsymbol{\nabla^2_{spherical}} = \nabla^2_r + \frac{1}{r^2}\nabla^2_\Omega \tag{2.3}$$

where:

$\boldsymbol{\nabla^2_{spherical}}$ is the Laplacian operator **vector** in spherical co-ordinate space.

∇^2_r is the Laplacian operator along the radial axis.

∇^2_Ω is the Laplacian operator along the angular axis.

and the radial Laplacian operator and the angular Laplacian operator being:

$$\begin{aligned} \nabla^2_r &= \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right) \\ \nabla^2_\Omega &= \frac{1}{\sin\,\theta}\frac{\partial}{\partial\theta}\left(\sin\,\theta\frac{\partial}{\partial\theta}\right) + \frac{1}{\sin^2\,\theta}\frac{\partial^2}{\partial\phi^2} \end{aligned} \tag{2.4}$$

The eigenvectors can therefore be computed by solving the following equation:

$$\nabla^2_r\Psi\left(r,\theta,\phi\right) + \frac{1}{r^2}\nabla^2_\Omega\Psi\left(r,\theta,\phi\right) + k^2\Psi\left(r,\theta,\phi\right) = 0 \tag{2.5}$$

where:

$\Psi\left(r,\theta,\phi\right)$ is a function defined in spherical co-ordinates.

Here, it can be realised that the equation 2.5 is a Helmholtz equation in spherical co-ordinates. This realisation allows this equation to be approached by separation of variables, that is:

$$\Psi(r,\theta,\phi) = R(r)\,\Omega(\theta,\phi) = R(r)\,\Theta(\theta)\,\Phi(\phi) \tag{2.6}$$

where:

$R(r)$ is a function of the radial variables.

$\Omega(\theta,\phi)$ is a function of angular variables.

$\Theta(\theta)$ is a function of the inclination angle variables.

$\Phi(\phi)$ is a function of the azimuthal angle variables.

For more details about the derivation of spherical harmonics and the Laplacian operator in spherical co-ordinates, see for example Freeden and Gutting (2013).

2.1.1 Solving for the angular part

From here, it can be proven (for complete proof, see for example Young (2009)) that the solutions to the separation of variables (equation 2.6) are:

$$\begin{aligned}\Theta(\theta) &= P_l^m(\cos\theta)\\ \Phi(\phi) &= e^{im\phi}\end{aligned} \tag{2.7}$$

where:

P_l^m are the associated Legendre polynomials defined in equation 1.11.

l is an integer constrained to range $(0,\infty)$.

m is an integer constrained to range $(-l,l)$.

Now, by setting the radial value to 1 for all such solutions, a set of functions defined on the surface of a unit sphere (as the radial part is 1) orthogonal to each other are obtained for each value of l. However, these functions are not orthonormal, *i.e.* the functions are orthogonal, but not normalised so that their product contains only Dirac delta functions. It can, nonetheless, be proven (for

example in Schonefeld (2005)) that these functions can be normalised by adding the normalisation function (also known as weighting function) N_l^m:

$$N_l^m = \sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}} \tag{2.8}$$

With this, a set of orthonormal functions defined on a surface of a sphere can be defined as per equation 2.9, which is identical to the equation 1.9. These functions are known as spherical harmonics. It also follows that using the spherical harmonics as basis functions for Fourier transform is identical to performing the Fourier transform in spherical co-ordinate space with radial part equal to 1 - *i.e.* on the surface of a unit sphere.

$$Y_l^m(\theta,\phi) = N_l^m \times P_l^m(\cos(\theta)) \times e^{im\phi} \tag{2.9}$$

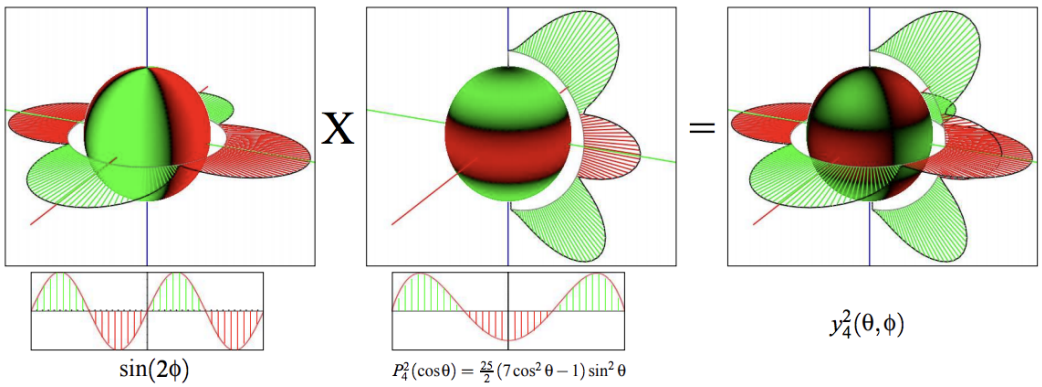
where:

N_l^m is the normalisation function defined in equation 2.8.

P_l^m is the associated Legendre polynomial defined in equation 1.11.

The figure 2.1 shows how the imaginary part of the spherical harmonics function with band four, order two (*i.e.* $Y_4^2(\theta,\phi)$) is constructed from the combination of the eigenvector solutions (equation 2.7) along the two angular dimensions. The left part of the figure shows the result of applying the imaginary part of the spherical harmonics function ($\sin(m\phi)$) along the azimuthal dimension of a unit sphere, while the centre part of the figure shows the result of applying the associated Legendre polynomial of appropriate band and order to the inclination dimension of a unit sphere. The result obtained by of combining the two dimensions is the imaginary part of the spherical harmonics function $Y_4^2(\theta,\phi)$ shown in the right part of the figure.

Figure 2.1: Construction of $\mathrm{Im}\left(Y_4^2\right)$ from the angular dimensions components



The figure shows the result of applying the angular solutions of the Helmholtz equation in spherical co-ordinates to an unit sphere. The left part shows the imaginary part of the solution for azimuthal angle being applied along the azimuthal dimension. The centre part shows the result of applying the inclination angle solution along the inclination dimension. The right part of the figure then shows the result of combining the two solutions into a single function. Figure adapted from Schonefeld (2005).

2.1.2 Solving for the radial part

Regarding the radial part of equation 2.6, it can be proven (for example in Young (2009)) that one solution to the radial part of the spherical-co-ordinate-space Laplacian-operator eigenvector-problem are the spherical Bessel functions $j_l(r)$. These functions are related to the Bessel functions as follows:

$$j_l(r) = \sqrt{\frac{\pi}{2r}} J_{l+1/2}(r) \tag{2.10}$$

where:

$j_l(r)$ is the spherical Bessel function of order l.

$J_l(r)$ is the Bessel function of order l defined as per equation 2.11.

Where the Bessel functions are defined as in equation 2.11 - for detailed discussion of the Bessel functions, see for example Korenev (2002).

$$J_l(r) = \sum_{m=0}^{\infty} \left(\frac{(-1)^m}{m!\,\Gamma(m+l+1)} \left(\frac{r}{2}\right)^{2m+l} \right) \tag{2.11}$$

where:

$J_l(r)$ is the Bessel function of order l.

$\Gamma(r)$ is the Gamma function defined as in equation 2.12.

$$\Gamma(z) = \int_{x=0}^{\infty} \left(x^{z-1}e^{-x}\right) dx \tag{2.12}$$

2.1.3 Spherical harmonics coefficients

Equipped with equation 2.9 to be used as a set of basis functions for spherical harmonics expansion and keeping in mind that the spherical harmonics expansion is a realisation of the Fourier transform on the surface of an unit sphere, it becomes clear that a function defined on the surface of an unit sphere can be expanded onto spherical harmonics coefficients by using the following equation:

$$\begin{aligned} f(\theta,\phi) &= \sum_{l=0}^{\infty} \left(\sum_{m=-l}^{l} \left(c_l^m \times Y_l^m(\theta,\phi)\right) \right) \\ \therefore \qquad c_l^m &= \int_{\theta=0}^{2\pi} \left(\int_{\phi=0}^{\pi} \left(f(\theta,\phi) \times Y_{l,m}^*(\theta,\phi)\right) \sin\phi \; d\phi \right) d\theta \end{aligned} \tag{2.13}$$

where:

$f(\theta,\phi)$ is a function defined on the surface of a unit sphere to be expanded.

c_l^m is the spherical harmonics expansion coefficient for band l and order m.

$Y_{l,m}^*(\theta,\phi)$ is the complex conjugate of the $Y_{l,m}(\theta,\phi)$ spherical harmonics function.

In terms of numerical computation, it is obvious that it is not possible to compute the infinitely many coefficients as the band goes to infinity. However, it can be noted that as the band and order increases, the frequency of associated Legendre polynomials increases proportionally. Similarly, the frequency of the azimuthal angle solution $e^{im\phi}$ is directly related to the order value m; and therefore, by limiting the number of bands for which the numerical computation is done, only high frequency signals are lost. It then follows that by applying a cap on the number of bands computed, the result of this limitation on the equation 2.13 is equivalent to applying a low-pass filter to the function before the spherical harmonics expansion. Therefore, the numerical computation is still useful, even when a cap on the number of bands used is applied.

2.2 Radial information in spherical harmonics expansion

With the spherical harmonics derived and their background shown, the question of how the information from the radial dimension should be included in the spherical harmonics transform needs to be addressed. This question arises from the three-dimensional property of the data; when the spherical harmonics are used on two-dimensional data, this question does not apply as the data can be mapped onto a sphere surface, a feature that cannot be done for the three-dimensional data without considerable information loss.

2.2.1 Spherical Bessel expansion

One possible solution is to use the spherical Bessel expansion briefly discussed in the section 2.1.2 above. To do this, the solution for the radial part of the Helmholtz equation in spherical co-ordinate space (*i.e.* spherical Bessel functions defined in equation 2.10) can be plugged into the separation of variables equation 2.6 along with the angular solutions (equation 2.7), leading to:

$$\Psi_{k,l,m}(r,\theta,\phi) = R_{k,l}(r)\,\Theta_{l,m}(\theta)\,\Phi_{l,m}(\phi) = \hat{N}_k^l\; j_l(kr)\;\; Y_l^m(\theta,\phi) \tag{2.14}$$

where:

k is the radial limit.

$\hat{N}_k^l$ is the normalisation function.

$j_l(kr)$ is the spherical Bessel function defined in equation 2.10.

Note that the normalisation function is different to the normalisation function used in spherical harmonics. Moreover, its particular form depends on the formulation of the boundary conditions; for more details about the function and its derivation see, for example, Wang et al. (2008a).

With the basis function defined by the equation 2.14, a Fourier transform can now be done not on the surface of an unit sphere, but rather in spherical co-ordinate space as a whole and so removing the radial information limitation of the spherical harmonics expansion. The resulting spherical Bessel expansion coefficients can then be computed as follows:

$$\begin{aligned} f(r,\theta,\phi) &= \sum_{k=0}^{\infty}\left(\sum_{l=0}^{\infty}\left(\sum_{m=-l}^{l}\left(s_{k,l,m}\times\Psi_{k,l,m}(r,\theta,\phi)\right)\right)\right) \\ \therefore \qquad s_{k,l,m} &= \int_{r=0}^{\infty}\left(\int_{\theta=0}^{2\pi}\left(\int_{\phi=0}^{\pi}\left(f(r,\theta,\phi)\times\Psi^*_{k,l,m}(r,\theta,\phi)\right)\right)\right)\sin\phi\; dr\; d\theta\; d\phi \end{aligned} \tag{2.15}$$

where:

$f(r,\theta,\phi)$ is a function defined in spherical co-ordinates.

$s_{k,l,m}$ is the spherical Bessel expansion coefficient.

This approach to inclusion of the radial information to the spherical harmonics expansion is used, for example, in the molecular replacement software such as `Phaser` (McCoy et al., 2007) or `MOLREP` (Vagin and Teplyakov, 2010).

2.2.2 Using multiple concentric spheres

While the spherical Bessel transform solves the issue of adding radial information to the spherical harmonics expansion, it could also considerably increase the computational cost of the computation. Therefore, an alternative solution was explored instead; this solution is based on using multiple concentric spheres centred at the origin of the spherical co-ordinate system and consequently mapping only a fraction (shell) of the three-dimensional object onto the closest sphere.

The advantage of this approach is that it does not require the computation of spherical Bessel functions and their normalisation, which are computationally costly. It is also worth noting that this approach does not require multiple computation of the spherical harmonics functions as they can be computed once and then stored and re-used for all spheres. Therefore, the increase in computational cost is only linear with the number of shells, as each shell needs to be mapped onto a surface of a sphere and expanded using the pre-computed spherical harmonics functions.

The disadvantage of this approach is that there still is some loss of information, albeit smaller than if no radial information was to be used. In order to minimise the information loss, the placement of the concentric spheres and their angular resolution needs to be done with regard to the resolution of the three-dimensional object.

Sphere placement and resolution

Now, assuming a three-dimensional object defined in a rectangular grid of dimensions $a \times b \times c$ with each grid point having a single value and defining the object through it, the centre is clearly at the co-ordinates $a/2$; $b/2$; $c/2$. In order to make sure each object grid point is used in the expansion (and therefore minimising the information loss), all the concentric spheres can have a maximum of 1 grid point along the radial axis on either side of the sphere. Given that the smallest

distance between grid points is 1, the concentric spheres should have their radii differ by 2 grid distances in between the grid points, starting at a distance 1.5 grid distances up to the maximum concentric sphere distance given by applying the Pythagoras theorem on the largest and second largest dimension in order to obtain the maximum grid point distance from the centre of the grid.

By applying this approach, any point on the surface of any of the concentric spheres can be approximated by using the trilinear interpolation, that is, by estimating a value at given non-grid co-ordinate by considering the eight surrounding grid co-ordinates at which values are known. Moreover, by considering that the values on the surfaces of the concentric spheres also need to be placed into a grid and further assuming this grid being a square grid, the maximum attainable resolution for these angular grids (angular resolution) can be obtained by considering the number of object grid points available to the largest concentric sphere. Specifically, the maximum angular resolution is $max(2a + 2b,\ 2a + 2c,\ 2b + 2c)$, as this is the maximum circumference of the box and therefore the maximum number of available points for interpolation on the surface of a sphere, thus giving a limit on the resolution the sphere grid.

At this point, it is also worth noting that Kostelec and Rockmore (2007) suggests using the angular resolution of $2B$, where B is the maximum band for which the spherical harmonics coefficients are computed (also known as *bandwidth*). Therefore, by determining the maximum angular resolution as per the previous paragraph, it is possible to obtain bandwidth for the computations as well by simply dividing the maximum angular resolution by two. Given all these features of the concentric spheres alternative to the spherical Bessel expansion, it was decided that priority should be given to the concentric spheres approach in terms of implementation of this project.

Spherical harmonics expansion with radial information

With the radial information added to the spherical harmonics expansion, it should be noted that the equation 2.13 showing the spherical harmonics coefficient calculation needs to be modified to reflect these changes. The modified version of the equation can be written as:

$$\begin{aligned} f(\theta,\phi) &= \sum_{l=0}^{\infty}\left(\sum_{m=-l}^{l}\left(\int_{r=0}^{r_{max}} c_l^m(r)\times Y_l^m(\theta,\phi)\,r^2 dr\right)\right) \\ \therefore \quad c_l^m(r) &= \int_{\theta=0}^{2\pi}\left(\int_{\phi=0}^{\pi}\left(f_r(\theta,\phi)\times Y_{l,m}^*(\theta,\phi)\right)\sin\phi\; d\phi\right)d\theta \end{aligned} \tag{2.16}$$

where:

$f(\theta,\phi)$ is a function defined in three-dimensional co-ordinates..

$f_r(\theta,\phi)$ is the mapping of function $f(\theta,\phi)$ to sphere with radius r.

$c_l^m(r)$ is the spherical harmonics coefficient for band l, order m and radius r.

$Y_{l,m}(\theta,\phi)$ is the spherical harmonics function of band l and order m.

2.3 Translation variance of spherical harmonics expansion

With the radial information inclusion approach decided, the translation variance property of the spherical harmonics expansion should be discussed. The translation variance means that translation of the expanded object in any direction causes change in the resulting expansion coefficients. This is not the preferred property for a shape descriptor, as it requires both shapes between which the distance is to be computed to be centred optimally in order to obtain the minimal distance between them.

2.3.1 Centre of density

One possible approach to remove or at least reduce the effect of this property of the spherical harmonics expansion is to centre all objects so that their centre of density (COD) is at the co-ordinate origin. The centre of density can be calculated using the equation 2.17, or in other words, it is the sum of all positions weighted by their map density.

$$\boldsymbol{r_{COD}} = \frac{1}{D}\sum_{i=1}^{n}(d_i \times \boldsymbol{r_i}) \tag{2.17}$$

where:

$\boldsymbol{r_{COD}}$ is a vector of co-ordinates of the centre of density.

$\boldsymbol{r_i}$ is a vector of co-ordinates for point i.

n is the total number of grid points.

d_i is the density at point i.

D is the total density in the map.

This approach has the advantage of solving the problem of translation variance in the optimal case of two identical, but differently positioned shapes. However, the robustness of this approach can be questioned, as any change in the number of points or small positional changes will lead to slightly different centre of density. This issue can easily be demonstrated by considering two shapes, for example globular protein domains. Assuming the two domains to be identical, except for having different length in a loop connecting them to the other domains in the protein, it is clear that the domain with longer linker region will have its COD shifted towards the linker region. The amount of this positional difference will be proportional to the size of the protein domains as all points in the equation 2.17 are weighted by the density, but otherwise equally. This theoretical example demonstrates that while the COD approach to translational variance can be used to reduce the translational variance, it cannot fully remove it.

2.3.2 Using Patterson maps

An alternative approach to removing the translation variance would be to remove the phase information from the density maps (or for co-ordinate files, to compute the theoretical density maps and subsequently remove the phase information from these) and use the resulting Patterson maps, which are intrinsically translation invariant. The Patterson maps are given by the Patterson function described in Patterson (1935) as follows:

$$P(u,v,w) = \mathrm{Re}\left(\sum_{h,k,l} |F_{h,k,l}|^2 \, e^{-2\pi i(hu+kv+lw)}\right) \tag{2.18}$$

where:

$P(u,v,w)$ is the Patterson function.

$F_{h,k,l}$ is the structure factor at Miller index position h, k and l.

h, k and l are the the Miller indices.

$\mathrm{Re}(...)$ means the real part of ... .

and the sum is taken over all space defined by the h, k and l indices.

The advantage of using this approach is that it is intrinsically translation invariant; this is because the Patterson map peaks are given by the interatomic vectors and the translation invariance then follows. This property of the Patterson maps has been used to solve small molecule crystals. However, by removing the phase information to obtain the property of translational invariance, a considerable amount of information about the shape is lost; this is basically re-introduction of the phase problem discussed in section 1.1.1.

Therefore, this method is not advisable for general usage, as the Patterson map does intrinsically centre the map, but also causes considerable loss of information. Nonetheless, the reason for discussing this approach to removing the translation variance of the spherical harmonics expansion is its possible application to searching for molecular replacement candidates. In this particular case, the query shape will not be in the state of solved structure (if this was the case, there

would be no need for molecular replacement), but instead will be in the Patterson map state. Therefore, it is useful for this project to implement both approaches to removing the translation variance of the spherical harmonics expansion method; one based on the Patterson function to be potentially used for molecular replacement candidate search and one to be based on the centre of density approach to be used when the phase information is available.

2.4 Rotation variance of spherical harmonics expansion

As well as being translation sensitive, the spherical harmonics expansion is also not rotation invariant, that is, two identical shapes with different rotation operators applied to them will have different spherical harmonics expansion coefficients. This is not the preferred property for this project, as using rotation invariant shape distance measure would be considerably easier and computationally cheaper than using rotation sensitive shape distance measure and having to try various rotations (*i.e.* three dimensional search) to find the rotation which minimises the shape similarity measure.

2.4.1 Eigenvectors-based approach

The rotation variance of the spherical harmonics expansion could be reduced by firstly making sure that all the object have optimal rotational orientation before starting the spherical harmonics expansion. This approach is similar to the COD approach for translation invariance in the sense that it does not remove the property of the spherical harmonics expansion as such, but rather changes the input objects so that the property does not have an effect on them.

A possible approach would be to obtain the eigenvectors for both the compared objects and consequently rotating the objects so that the largest eigenvector lies on the x axis, the second largest on the y axis and the smallest eigenvector on the z axis (or in any other, but constant, order of the axes). This approach should

not be too computationally expensive and would be a solution for the optimal case of two identical, but differentially rotated shapes. However, this approach could be very sensitive to small changes in the shapes.

To demonstrate the sensitivity of this approach to reducing rotational variance, it is worth noting that if the shape were a perfect sphere, then the three eigenvectors would be degenerate (*i.e.* not uniquely defined). Now, if a protein shape were to be mostly spherical with some minor differences (which may be the case for globular proteins, albeit not as much for protein domains), then it would have three rather similarly sized eigenvectors. It is easy to imagine that a copy of this protein structure with one of the surface loops being longer or having extra few residues in the linker region could have its eigenvector lengths order swapped and therefore would end up being rotated by 90° (or $\pi/2$ radians) differently than the other structure, even though the two structures could be very similar. Given that this approach has a sensitivity to the surface regions, it seems not to be the optimal approach to employ in this projects and that other solutions to rotational invariance need to be explored.

Chapter 3

Shape similarity descriptors

With the spherical harmonics derivation and properties discussed and explained, it is now possible to focus on how shape similarity could be recognised and enumerated in a fashion allowing comparisons between different shapes. Furthermore, the question of how the accuracy and efficiency of the proposed solutions should be measured needs to be addressed, as decisions regarding different approaches to shape similarity measures will have to be taken in a comprehensive and reliable manner.

3.1 Creating a test set

In order to facilitate the development of the shape similarity measures, a test dataset is required. As the intended purpose of the test set is to aid decisions about shape descriptors for detecting similarity in protein domain shapes, it became clear that none of the available *gold standard* object-similarity test-sets, such as those of Viksten et al. (2009) (industrial objects with different rotations) or Rodola et al. (2014) (human shapes undergoing transformation) are really applicable to this project. The reason is that while these datasets are useful for applications in their respective fields, testing different approaches for shape-similarity detection on them would not lead to optimisation for protein-domain shape-similarity detection.

Another possibility would be to use any of the protein domain databases discussed in section 1.3.1. All of these databases could be adapted to form a test set with examples of protein domain shapes being classified as similar or different. However, in order to avoid using any criteria other than the shape itself to determine similarity, only databases which cluster protein domains based purely on their shape should be considered. This realisation leads to a very similar question as briefly discussed in section 1.3.2 and it seems reasonable to answer similarly. Therefore, the BALBES database was selected to be used to create a test set for protein domain shape descriptors.

3.1.1 Generation of the test set from the BALBES database

To generate a test dataset, two different versions of the BALBES database were kindly supplied by Dr. Fei Long, the current version with $13,719$ entries and an earlier version with $43,468$ entries. Subsequently, the domain similarity tables used to generate the current version of the BALBES database, also kindly supplied by Dr. Fei Long were used and simple name matching algorithm was developed to extract domains which were clustered together based on similar crystallographic cell parameters, low RMSD value and sequence similarity during the creation of the BALBES database. The BALBES database creation is discussed in more details in section 1.3.3. This procedure resulted in a dataset containing 402 protein-domain co-ordinate files clustered in 104 groups.

3.1.2 Test-set properties

The test set comprises of 104 groups with mean group size 3.87 domains (median is 2 domains), minimal size of 2 domains and maximum of 19 domains. In terms of the number of atoms, the smallest domain contains 301 atoms, while the largest domain consists of $5,415$ atoms with the mean being $1,118$ and the median 843 atoms.

Regarding the secondary structure composition of the test set, the test set was analysed using the DSSP (Touw et al., 2015) software. Then, four different types of structures were defined by the secondary-structure content - if less than 40% of the structure consisted of α-helices and β-sheets as predicted by DSSP, then the structure was assigned as not having secondary structure; if more that 80% of the secondary structures detected were α-helices, the structure was assigned as α-helical structre; if more than 80% of the secondary structures detected were β-sheets, then the structure was assigned as being β-sheets and if none of the above conditions was met, the structure was assigned as being mixed secondary structure. Table 3.1 shows the results of the analysis of the dataset.

Table 3.1: Secondary structure content of the test dataset

Secondary structure content class	Number of dataset entries
α-helical domains	110
β-sheet domains	53
Mixed domains	233
No secondary structure	6

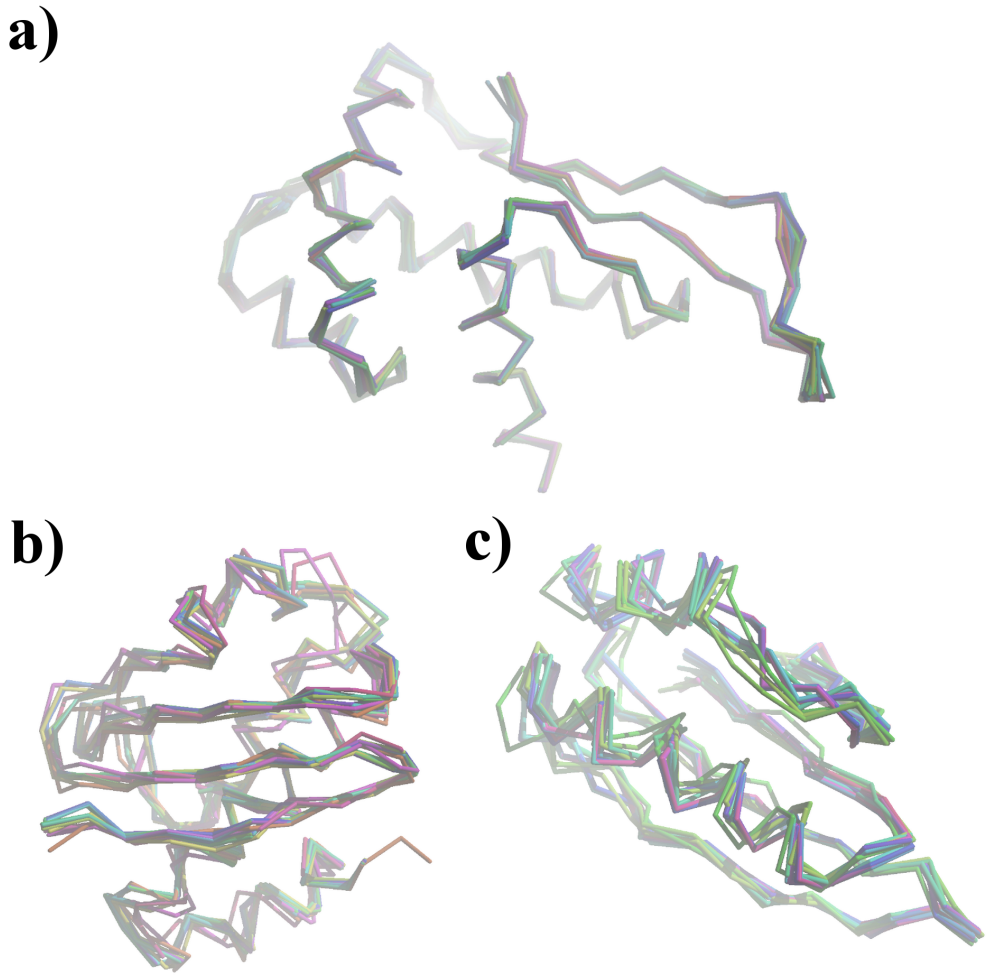
This table shows the total number of the test dataset domains with given secondary structure content. The secondary structure was predicted using DSSP (Touw et al., 2015). The secondary structure classes are defined as described above.

In order to visually confirm that the groups defined in the test dataset actually contain protein domains with very similar shape, the three largest groups (with group sizes 19, 17 and 17 domains) are visualised in figure 3.1.

3.2 Energy-levels-based shape-similarity descriptor

With the test dataset defined and available to facilitate decisions about the protein-domain shape-descriptor accuracy, the focus can now be aimed at developing suitable shape descriptors. Starting with the spherical-harmonics energy-level-descriptors discussed previously (section 1.4.4), it should be recalled that while they provide the preferred property of being rotation invariant, they do lead to a loss of information. This fact is clearly seen as the spherical harmonics expansion coefficients form an array organised by the sphere, band and order parameters,

Figure 3.1: Visualisation of the three largest groups in the test dataset



This figure shows the three largest groups in the test dataset in its three parts a), b) and c). The structures in each group were structurally aligned using the `SSM` (Krissinel and Henrick, 2004) software and visualised using the `Coot` (Emsley et al., 2010).

while the spherical-harmonics energy-level-descriptor forms an array organised by only two parameters, sphere and band. Therefore, in order to decrease the loss of information caused by using the energy levels instead of full spherical harmonics coefficients, the order parameter information needs to be, at least partially, retained.

3.2.1 Cross-correlation and spherical harmonics coefficients

In order to retain some of the information held by the individual frequencies of the spherical-harmonics band orders, while still introducing the rotation invariance of the energy level descriptors, the following idea was developed: Instead of using the spherical harmonics coefficients sum over the order parameter to obtain the energy levels descriptor, the sum could be made over the cross-correlation values (*i.e.* the correlation in the frequency domain of Fourier transform) of the spherical harmonics coefficients bands with no displacement; thus leading to the following definition:

$$\tau_l(r, r') = \sum_{m=-l}^{l} \left(c^*_{l,m}(r) \times c_{l,m}(r') \right) \tag{3.1}$$

where:

r and r' denote two spheres from the concentric spheres set. Note: $r = r'$ is not permissible.

$\tau_l(r, r')$ is the shape descriptor for band l and spheres r and r'.

$c_{l,m}(r')$ is the spherical harmonics coefficient for band l, order m and sphere r'.

$c^*_{l,m}(r)$ is the spherical harmonics coefficient complex conjugate for band l, order m and sphere r.

The equation 3.1 can therefore be interpreted as computing the cross-correlation coefficient between the same bands of the spherical harmonics, but from different spheres and with the displacement constant being zero. The advantage of this descriptor τ is that while it does not retain any of the order information directly, it re-introduces some of the order information through having the cross-correlation between different spheres. It is also worth noting that the dimensionality of the coefficient arrays stays three when the τ descriptor is used, as now the results array is parametrised by band (l), sphere 1 (r) and sphere 2 (r'). A very interesting feature of the τ descriptor is that it is rotation invariant. This

property will be proven in section 3.2.3, but firstly the Wigner matrices ($\mathscr{D}$) need to be briefly introduced.

3.2.2 Brief introduction to Wigner $\mathscr{D}$ matrices

The Wigner $\mathscr{D}$ matrices were described by Wigner (1931) and they are an irreducible representations of the SU(2) and SO(3) groups. In other words, the Wigner $\mathscr{D}$ matrices represent rotations in the spherical-harmonics frequency-domain, so that if the spherical harmonics coefficients $c_{l,m}$ are multiplied by the Wigner $\mathscr{D}^{l}_{m,m'}(\alpha,\beta,\gamma)$ matrices as per equation 3.2, the resulting new set of $c^{rotated}_{l,m}$ coefficients are identical to applying a rotation of α,β,γ Euler angles prior to the spherical harmonics expansion. For more details about the Euler angles, see the section 3.4.1.

$$c^{rotated}_{l,m}(r) = \sum_{m'=-l}^{l} \left(\mathscr{D}^{l}_{m,m'}(\alpha,\beta,\gamma)\, c_{l,m'}(r) \right) \tag{3.2}$$

Given that the Wigner $\mathscr{D}$ matrices are rotation matrices, they do have the standard rotation matrix properties shown in equation 3.3. Note that the order m and m' iterators were removed from the equation 3.3; this is to signify that the matrix $\mathscr{D}_l$ of dimensions $m \times m'$ and vector c_l of length m are used in linear algebra sense.

$$\begin{gathered} \mathscr{D}^{*T}_{l}(\alpha,\beta,\gamma) = \mathscr{D}^{-1}_{l}(\alpha,\beta,\gamma) \\ \mathscr{D}^{*T}_{l}(\alpha,\beta,\gamma) \times \mathscr{D}_{l}(\alpha,\beta,\gamma) = I \\ \left(\mathscr{D}_{l}(\alpha,\beta,\gamma) \times c_{l}(r)\right)^{*T} = c^{*}_{l} \times \mathscr{D}^{*T}_{l}(\alpha,\beta,\gamma) \end{gathered} \tag{3.3}$$

where:

* denotes a complex conjugate.

^{T} denotes matrix transpose.

$^{-1}$ denotes matrix inverse.

I is the Identity matrix.

The Wigner $\mathscr{D}$ matrices will be discussed in greater detail in a later section, but the knowledge of them being the representations of rotations in spherical harmonics space will be sufficient to prove that the τ descriptor is rotation invariant.

3.2.3 Cross-correlation τ descriptor is rotation invariant

Rotation invariance has been shown to be true for spherical Bessel cross-correlation measure by Wang et al. (2008b); using a similar approach to prove this feature is also true for the τ descriptor, start by considering the equation 2.13 (for convenience repeated as the first part of equation 3.4 below) and the effect of applying a rotation $\mathscr{R}(\alpha, \beta, \gamma)$ along the Euler angles α, β and γ to the original function. By definition, the equation for the rotation effect is:

$$\begin{aligned} f(\theta,\phi) &= \sum_{l=0}^{\infty}\left(\sum_{m=-l}^{l}\left(\int_{r=0}^{r_{max}} c_l^m(r)\times Y_l^m(\theta,\phi)\, r^2 dr\right)\right) \\ \therefore\ \mathscr{R}(\alpha,\beta,\gamma)\, f(\theta,\phi) &= \sum_{l=0}^{\infty}\left(\sum_{m=-l}^{l}\left(\int_{r=0}^{r_{max}}\left(\sum_{m'=-l}^{l}\left(\mathscr{D}^l_{m,m'}(\alpha,\beta,\gamma)\, c_{l,m'}(r)\right) Y_l^m(\theta,\phi)\, r^2 dr\right)\right)\right) \end{aligned} \tag{3.4}$$

Which can be written in the matrix form as:

$$\mathscr{R}(\alpha,\beta,\gamma)\, f_r(\theta,\phi) = \sum_{l=0}^{\infty}\left(\int_{r=0}^{r_{max}}\left(\mathscr{D}_l(\alpha,\beta,\gamma)\, c_l(r)\times Y_l(\theta,\phi)\, r^2 dr\right)\right) \tag{3.5}$$

From equation 3.5, it is clear that the rotation is applied directly to the spherical harmonics coefficients vector c_l. Applying the rotation to the shape descriptor τ in the same way, it follows that:

$$\begin{aligned}
\mathscr{D}_l(\alpha,\beta,\gamma) \times \tau_l(r,r') &= \left(\mathscr{D}_l(\alpha,\beta,\gamma)\, c_l(r)\right)^{*T} \times \mathscr{D}_l(\alpha,\beta,\gamma)\, c_l(r') \\
&= c_l^*(r)\, \mathscr{D}_l^{*T}(\alpha,\beta,\gamma)\, \mathscr{D}_l(\alpha,\beta,\gamma)\, c_l(r') \\
&= c_l^*(r) \times I \times c_l(r') \\
&= c_l^*(r)\, c_l(r') \\
&= \tau_l(r,r')
\end{aligned} \tag{3.6}$$

Where the bracket expansion from line 1 to line 2 was done using the last part of equation 3.3, then the middle part of the equation 3.3 was used to combine the Wigner $\mathscr{D}$ matrix and its transposed conjugate into a single identity matrix. The identity matrix can then be removed as it has no effect on vector multiplication (resulting in line 4 of equation 3.6) and finally, using the definition of the τ descriptor (equation 3.1), line 5 of equation 3.6 is obtained. Therefore, it can be concluded that any rotation applied to the descriptor τ is irrelevant and thus that the descriptor τ is intrinsically rotation invariant as required. This proof is based on the proof used by Wang et al. (2008b) to show that the spherical Bessel cross-correlation is rotation invariant.

3.2.4 Cross-correlation τ descriptor and Patterson maps

With the rotation invariance feature proven for the τ descriptor, it is very interesting to consider the spherical harmonics feature stemming from Condon-Shortley phase element (Condon and Shortley, 1951) of the spherical harmonics functions (the Condon-Shortley phase is used to treat angular momenta in quantum mechanics), specifically:

$$Y_{l,-m}(\theta,\phi) = (-1)^m\, Y_{l,m}^*(\theta,\phi) \tag{3.7}$$

Now, this relationship between the spherical harmonics function and its complex conjugate is not generally valid for spherical harmonics coefficients as is evident from the following equation.

$$\begin{aligned} c_l^m(r) &= \int_{\theta=0}^{2\pi} \left(\int_{\phi=0}^{\pi} \left(f_r(\theta,\phi) \times Y_{l,m}^*(\theta,\phi) \right) \sin\phi \, d\phi \right) d\theta \\ &= (-1)^m \int_{\theta=0}^{2\pi} \left(\int_{\phi=0}^{\pi} \left(f_r(\theta,\phi) \times Y_{l,-m}(\theta,\phi) \right) \sin\phi \, d\phi \right) d\theta \\ \therefore \quad c_{l,m}^*(r) &= (-1)^m \int_{\theta=0}^{2\pi} \left(\int_{\phi=0}^{\pi} \left(f_r^*(\theta,\phi) \times Y_{l,-m}^*(\theta,\phi) \right) \sin\phi \, d\phi \right) d\theta \end{aligned} \tag{3.8}$$

Where the relationship shown in equation 3.7 cannot be applied to the spherical harmonics coefficients, because it does not hold for them as shown in the last part of equation 3.8. The reason is that the expanded function f also requires the complex conjugate of its expansion to be taken. However, in the specific case where the expanded function f expansion does have no phase (*i.e.* its imaginary parts are all zero), it is possible to use equation 3.7, as the complex conjugate of a real number is the same real number. Noting that this is exactly the case for Patterson maps, it holds for any function f which has real-valued expansion, that:

$$\begin{aligned} c_{l,m}^*(r) &= \int_{\theta=0}^{2\pi} \left(\int_{\phi=0}^{\pi} \left(f_r(\theta,\phi) \times Y_{l,m}(\theta,\phi) \right) \sin\phi \, d\phi \right) d\theta \\ &= (-1)^m \int_{\theta=0}^{2\pi} \left(\int_{\phi=0}^{\pi} \left(f_r(\theta,\phi) \times Y_{l,-m}^*(\theta,\phi) \right) \sin\phi \, d\phi \right) d\theta \\ &= (-1)^m c_{l,-m}(r) \end{aligned} \tag{3.9}$$

Using the result from equation 3.9, it is easy to show that:

$$\begin{aligned}
\tau_l(r, r') &\equiv \sum_{m=-l}^{l} \left(c^*_{l,m}(r)\, c_{l,m}(r') \right) \\
&= \sum_{m=-l}^{l} \left((-1)^m c_{l,-m}(r)\, c_{l,m}(r') \right) \\
&= \sum_{m=l}^{-l} \left((-1)^m c_{l,m}(r)\, c_{l,-m}(r') \right) \\
&= \sum_{m=l}^{-l} \left(c_{l,m}(r)\, c^*_{l,m}(r') \right)
\end{aligned} \tag{3.10}$$

Where the result of equation 3.9 was used to obtain line 2 of equation 3.10, then the summation order was reversed to obtain line 3 of equation 3.10 and finally the result of equation 3.9 was used again to obtain line 4 of the equation 3.10. Therefore, the spherical harmonics coefficients $c_{l,m}$ are equal to their complex conjugates; as this property can only be true for real numbers, it follows that the spherical harmonics expansion of real valued Patterson functions is also real valued.

3.2.5 The spherical harmonics band zero

At this point, it is important to note the meaning of the spherical harmonics coefficient Y_0^0. Specifically, it can be seen from figure 1.5 that the spherical harmonics band zero is a constant function (with value $1/2\sqrt{1/\pi} \approx 0.2821$). The effect of this fact is that this band does not distinguish shapes, but only takes into account the total density of the function defined on the surface of the sphere. In other words, as spherical harmonics are equivalent to the Fourier transform in the spherical co-ordinates, the spherical harmonics band zero can be seen as the first element of the Fourier series, which also gives the average density of the expanded function but no frequency information.

Given this, it seems unnecessary to compute and use the spherical harmonics band zero when comparing two shapes. The advantage of not using the band zero is in increasing the accuracy of the shape description, as shape-invariant

descriptor with relatively high absolute value is removed from the shape distance computation. Furthermore, there is a decrease in the computational cost, albeit it is rather small in the context of all other costs of the computation. Therefore, the spherical harmonics band zero (Y_0^0) and the related cross-correlation descriptor (τ_0) will not be used in any further shape distance computations.

3.2.6 Cross-correlation distance

With the properties of the cross-correlation descriptor τ discussed, it now needs to be seen how the τ descriptor can be used to obtain a numerical distance between two objects shapes. Assuming that the two objects are mapped onto concentric spheres placed as discussed in the previous section (2.2.2), then expanded onto their respective spherical harmonics coefficients and that the τ descriptor has been computed, both objects are now described by three-dimensional arrays ordered by their band (l), sphere 1 (r) and sphere 2 (r').

Dimensionality of the descriptor arrays

In order to compare two such arrays, their dimensionality needs to be made equal. Consequently, if the two arrays have differing number of bands, the bands missing from one object and present in the other object are the high frequency bands supported by the resolution of one object, but not by the resolution of the other. It therefore seems appropriate to use only the bands present in both objects, as the extra bands should not be compared to zeroes - the missing bands are not zeroes, they are simply not supported by the resolution.

Similarly, if the two arrays have different number of concentric spheres, it is possible to assume the missing spheres to have values zero. However, by using only the spheres which are present in both objects, the descriptor would be able to detect similarity in the protein domain core, even when the surface (outer spheres) are different or not existent. Therefore, the optimal approach for the project seems to be using only the shells which are present in both objects.

Distance between two matrices

Now, to obtain a numerical distance between the descriptor arrays, the distance between the same sized matrices of τ_l (with dimensions $r \times r'$) need to be computed. Nonetheless, there are many different approaches to obtain distances between matrices, each with its advantages and disadvantages. Therefore it was decided that the approaches listed in table 3.2 will all be used to compute the distances between the τ_l matrices from different objects and consequently their respective accuracy in detecting similar and distinguishing different shapes will serve as the basis for selecting the optimal approach for this project.

Table 3.2: Matrix distances used in search for optimal protein domain shape descriptor

Matrix Distance	Definition	where:
Sum of differences	$D_{s.o.d.}(A,B) = \sum_{i=0}^{r}\left(\sum_{j=i+1}^{r'}(abs(a_{i,j}-b_{i,j}))\right)$	$a_{i,j}$ and $b_{i,j}$ are values of matrices A and B
Root square distance (Frobenius norm)	$D_{r.s.d.}(A,B) = \sqrt{\sum_{i=0}^{r}\left(\sum_{j=i+1}^{r'}((a_{i,j}-b_{i,j})^2)\right)}$	$a_{i,j}$ and $b_{i,j}$ are values of matrices A and B
Singular values distance	$D_{s.v.d.}(A,B) = \sqrt{\sum_{i=0}^{r}(\mathbf{S}_i^A - \mathbf{S}_i^B)^2}$	$\mathbf{S}^A$ are singular values of matrix A. $\mathbf{S}^B$ are singular values of matrix B.
Pearson's correlation coefficient	$D_{p.c.c.}(A,B) = \frac{\sum_{i=1}^{r}\sum_{j=1}^{r'}\left((a_{i,j}-\mu_a)(b_{i,j}-\mu_b)\right)}{\sqrt{\sum_{i=1}^{r}\left(\sum_{j=1}^{r'}(a_{i,j}-\mu_a)\right)^2}\sqrt{\sum_{i=1}^{r}\left(\sum_{j=1}^{r'}(b_{i,j}-\mu_b)\right)^2}}$	$a_{i,j}$ and $b_{i,j}$ are values of matrices A and B μ_a is the mean value of matrix A μ_b is the mean value of matrix B
Spearman's rank correlation coeff.	$D_{s.c.c.}(A,B) = 1 - \frac{6\sum_{i=1}^{r}\left(\sum_{j=1}^{r'}\left((rg(a_{i,j}-rg(b_{i,j}))\right)^2\right)}{(rr')\left((rr')^2-1\right)}$	$a_{i,j}$ and $b_{i,j}$ are values of matrices A and B $rg(a_{i,j})$ is integer rank in matrix A $rg(b_{i,j})$ is integer rank in matrix B

This table shows different approaches to computing distances ($D(A,B)$) between two matrices A and B and their mathematical definition.

Single distance from a distance vector

Given that the matrix distance approaches listed in table 3.2 can be computed for each pair of matrices, applying any of them will result in a vector $\mathbf{T}$ with length l, which will contain the distances between all l matrix pairs present in both objects.

Nonetheless, in order to obtain a single numeric distance between the two objects, the vector $\mathbf{T}$ will have to be reduced to a single number.

Similarly to the matrix distance, there are many possible approaches for reducing a vector to a single number; these approaches are generally known as vector norms. In an attempt to discover which of the available vector norms works best with the protein domain shapes and the cross-correlation-based shape-descriptor τ, a selection of frequently used vector norms, listed in table 3.3 along with their mathematical definitions, was chosen. These selected vector norms were then all used in combination with all matrix distance approaches listed above, so that the most accurate combination could be selected.

Table 3.3: Vector norms used in search for optimal protein domain shape descriptor

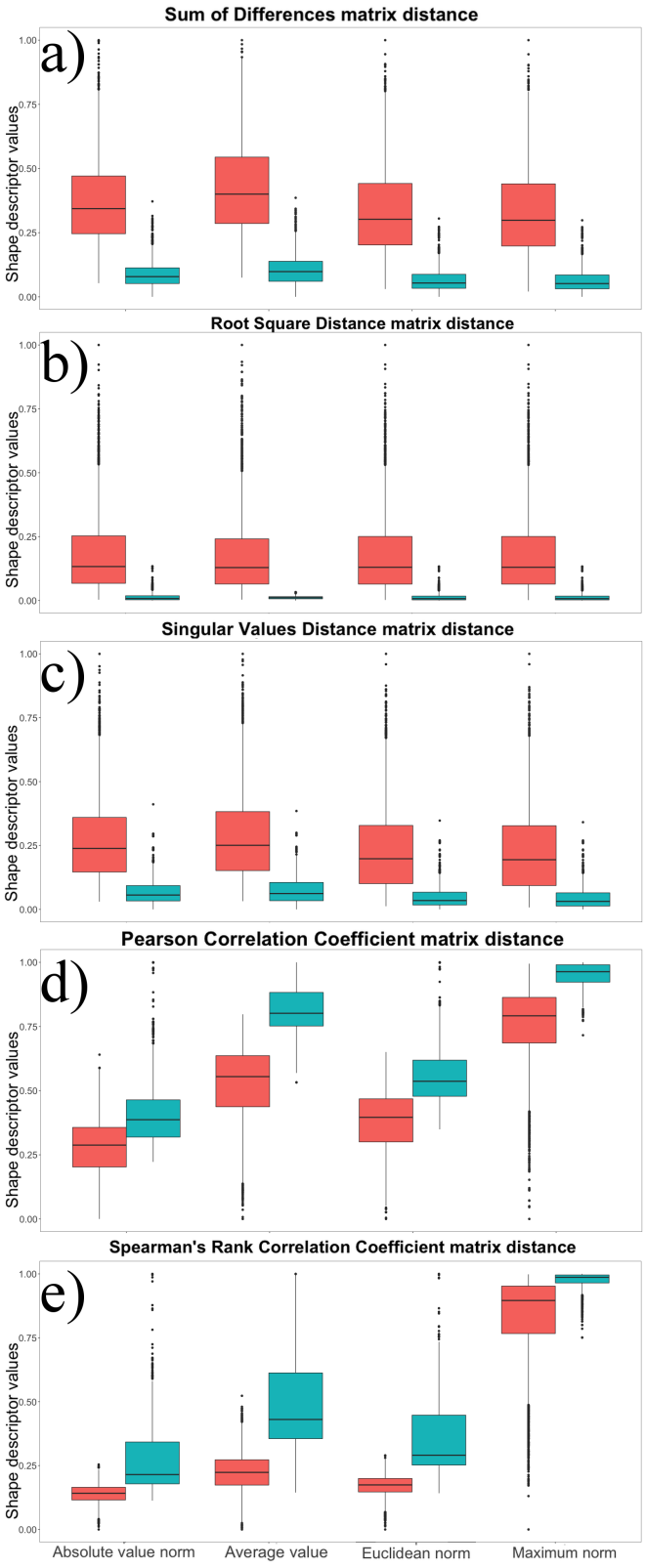
Matrix Distance	Definition	where:
Vector mean	$\|\mathbf{T}\|_{mean} = \frac{1}{n}\sum_{i=1}^{n}(t_i)$	n is the length of vector $\mathbf{T}$ t_i is the i^{th} element of vector $\mathbf{T}$
Absolute value (L_1) norm	$\|\mathbf{T}\|_{abs} = \|\mathbf{T}\|_1 = \sum_{i=1}^{n}(\|t_i\|)$	n is the length of vector $\mathbf{T}$ $\|...\|$ is the absolute value of ... t_i is the i^{th} element of vector $\mathbf{T}$
Euclidean (L_2) norm	$\|\mathbf{T}\|_{Eucl} = \|\mathbf{T}\|_2 = \sqrt{\sum_{i=1}^{n}(t_i^2)}$	n is the length of vector $\mathbf{T}$ t_i is the i^{th} element of vector $\mathbf{T}$
Max norm	$\|\mathbf{T}\|_{max} = \|\mathbf{T}\|_{\infty} = \max(t_1, t_2, ..., t_n)$	n is the length of vector $\mathbf{T}$ t_i is the i^{th} element of vector $\mathbf{T}$

This table shows different approaches to reducing a vector of descriptors $\mathbf{T}$ onto a single distance number.

Comparison of matrix distances and vector norm combinations

With the ability to compute a single distance value for any pair of protein domains, the test dataset can now be used to decide which of the matrix-distance approaches and the vector-norm combination is the most appropriate for this application. Therefore, all combinations of matrix distances and vector norms were used to compute the descriptor distance between any two structures belonging to the same group of the test dataset. Also, the distances were computed between one representative structure of a group to all representative structures of all the other groups. These calculations lead to two distance distributions per combination of matrix distance and vector norm methods, the within groups distances distribution and the between groups distances distribution.

Figure 3.2: Distributions of τ descriptor distances within and between the test dataset groups using different matrix-distance approaches and vector norms



This figure shows the results of computing the cross-correlation based descriptor τ distances between similarly shaped protein domains (defined by being in the same group of the test dataset) and differently shaped protein domains (defined by being in different groups of the test dataset). The resulting two distributions of the descriptor values are shown always with the similar shapes distribution in turquoise colour and different shapes distribution in red colour. The five parts of this figure show these shape descriptor distributions with different matrix distance metric being used for the shape distance computation (see table 3.2); in part a), the Sum of differences measure was used, part b) was produced using the Root square distance method, part c) is based on the Singular values distance approach, part d) uses the Pearson's correlation coefficient, while part e) was produced using the Spearman's rank correlation coefficient. Furthermore, each plot in parts a) to e) is divided into four pairs of distances; each distance pair (distance pair constitutes a similar shape distances and different shape distances distributions) was produced using the same matrix distance method, but using different vector norm - for list of vector norms tested, see table 3.3. All distribution pairs (similar and different shape distributions produced using the same methods) were normalised to have the joint maxima of 1 and minima of 0.

Figure 3.2 shows how the cross-correlation-based shape-similarity descriptor is distributed among the similar protein domains (the turquoise coloured box plots) and dissimilar structures (the red coloured box plots) and more importantly, how these distributions differ when different combinations of the matrix distances and vector norms are applied. In general, it is visible from figure 3.2 that all of the proposed matrix distance approaches do indeed lead to a descriptor which distinguishes between the similar and dissimilar shapes; this can be seen from the fact that the distributions have minimal overlap.

Weighting by the shell distance

However, before making the decision as to which of the matrix-distance and vector-norm methods combination should be implemented into the cross-correlation shape descriptor, it is interesting to consider the possibility of weighting the matrices by the shell distance. The rationale behind this idea is that the different spheres may have slightly different meaning depending on their position; for example, by increasing the weight on the internal spheres, the descriptor would increase the importance of similarity in the core of the protein domain, while being more relaxed about differences in the surface regions. Similarly, by increasing the weight of the shells furthest from the centre, less false positives could be discovered as much closer similarity in the, presumably, most differing regions would be required.

In order to explore this idea, a weighting function was implemented to multiply each element of the τ_l matrices by its shell 1 and shell 2 positions (in other words by the row and column number) raised to the power of the weighting argument or the inverse of this number, that is:

$$\begin{aligned} \tau_l^{weighted} &= \sum_{i=1}^{r} \left(\sum_{j=1}^{r'} \left(\tau_l \left(r_i, r'_j \right) \times i^w \times j^w \right) \right) \\ \tau_l^{weighted} &= \sum_{i=1}^{r} \left(\sum_{j=1}^{r'} \left(\tau_l \left(r_i, r'_j \right) \times {}^1\!/\!_{i^w} \times {}^1\!/\!_{j^w} \right) \right) \end{aligned} \tag{3.11}$$

where:

τ_l is the matrix of $\tau_l(r, r')$ descriptors.

w is the weighting factor.

The resulting distributions of the shape descriptor for similar (within group distances) and different structures (between group distances) are shown in figure 3.3.

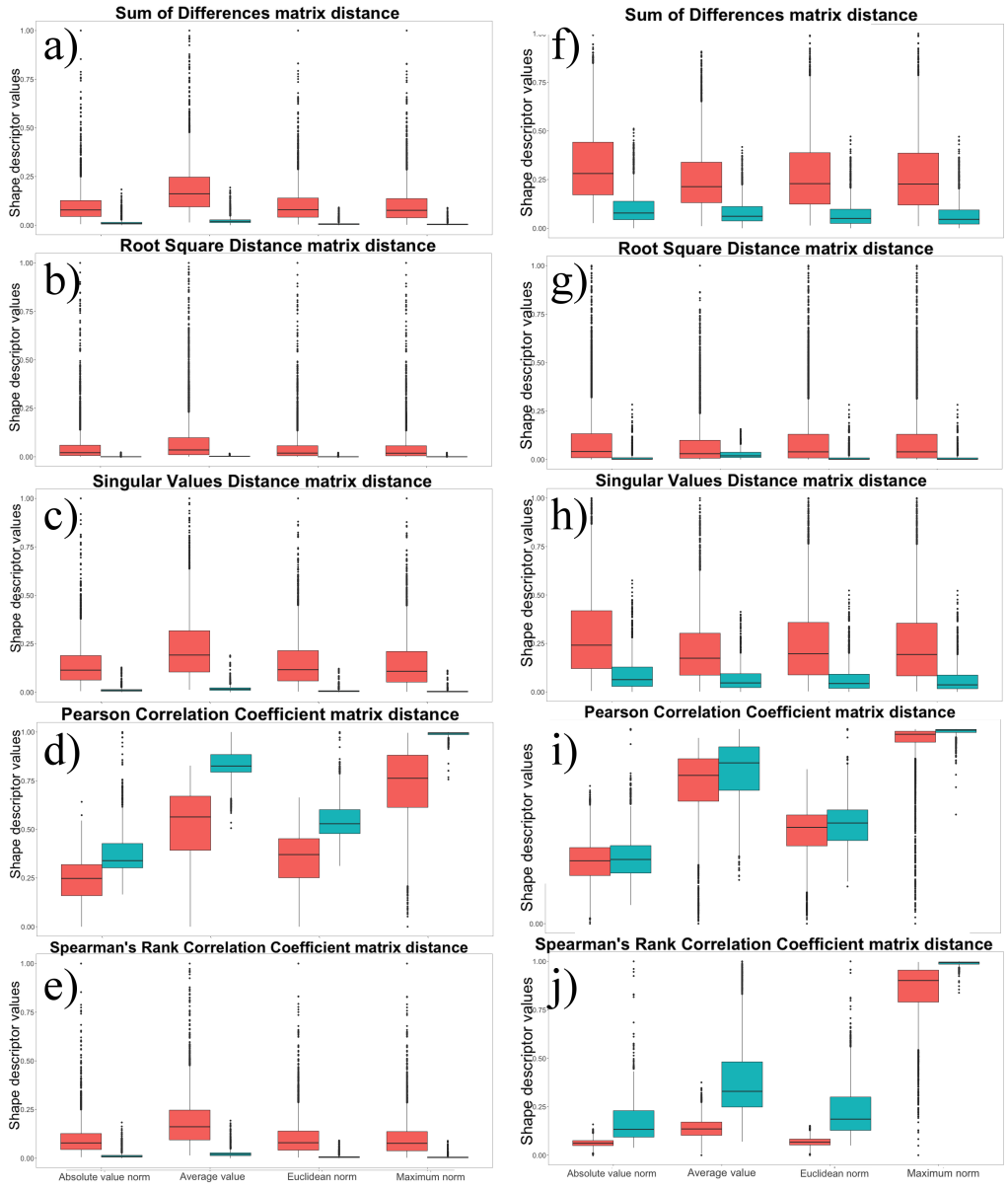
AUROC distribution difference measure

The figures 3.2 and 3.3 are useful for visually inspecting the distributions of the shape descriptor values; however, it would be difficult to determine which of the possible matrix distance method and vector norm combinations, as well as possible weighting approaches should be used to maximise the shape descriptor ability to distinguish between similar and different shapes.

Therefore, the area under receiver operating characteristic (AUROC) measure of predictive power was used to enumerate how efficient the various approaches are. The AUROC measure is based on selecting a numerical threshold (typically just under the lowest measured descriptor value) and computing the false positive rate ($FPR = \sum {}^{False\ Positives}\!/\!_{\sum Condition\ negative}$) as well as the true positive rate ($TPR = \sum {}^{True\ Positives}\!/\!_{\sum Condition\ positive}$). Subsequently, the threshold is increased slightly and the computation of FPR and TPR is repeated. This is done until the threshold value is larger than the highest measured value.

The result of this calculation is a set of FPR and TPR values, which can be plotted against each other (typically, FPR on x axis and TPR on y axis). This

Figure 3.3: Distributions of τ cross-correlation distance measure within and between the test dataset using different matrix-distance approaches and vector norms, as well as using sphere number weighting



This figure shows the same plots as the figure 3.2 for the five different matrix distance methods and four different vector norms. However, in the parts a) to e), the weighting function based on the sphere number (top line of equation 3.11) with weighting parameter 2.0 has been applied, while the parts f) to j) have been computed using the weighting function with inverse sphere position distance weight (the bottom line of equation 3.11) and weight parameter 2.0. All distribution pairs (similar and different shape distributions produced using the same methods) were normalised to have the joint maxima of 1 and minima of 0.

plot is known as the receiver operating characteristic (ROC) and the area under the curve given by the FPR and TPR pairs is then the value of the AUROC measure. The AUROC measure is the same quantity as the Wilcoxon statistic

and is therefore related to the Mann-Whitney U test through it, as discussed in detail by Hanley and McNeil (1982). Moreover, contrary to the statistical tests, the AUROC has easily interpretable value of 0.5 when the two distribution are identical and 1 (or 0) when the two distribution are completely different and can be completely separated by a single threshold value.

Deciding the shape descriptor final implementation

Finally, in order to decide which of the proposed methods for obtaining a single numerical distance for shape similarity using the τ descriptor should be used, the AUROC measure was computed for all methods and weighting combination shown in figures 3.2 and 3.3 as well as some other weighting parameter values. The results are available in table 3.4.

From table 3.4, it is clear that the highest AUROC measure value and consequently the strongest distinguishing power between similar and dissimilar structures is achieved by using the Pearson's correlation coefficient as a distance measure between two matrices, the vector average to reduce it into a single number and not applying any weighting to the sphere positions. It is also worth noting that almost as high value was reached with the Singular Values matrix distance, average as vector reduction and weighting with parameter of 2.0, as well as some other combinations. Nonetheless, the shape descriptor was implemented with the Pearson's correlation coefficient as the matrix distance measure, the vector average as the vector reduction method and with no weighting applied.

Finally, it is also worth noting that while figures 3.2 and 3.3 do show the shape distributions to be in range between 0 and 1, this is not the natural range for most of the matrix distance methods and the pairs of distributions needed to be scaled into this range. Nonetheless, the Pearson correlation coefficient distance does range from -1 to +1 naturally; this feature is not required as scaling can be applied, but it is mathematically pleasing and makes the interpretation of results easier.

Table 3.4: AUROC measure for different matrix distances, vector norms and weighting combinations

Matrix distance method	Vector norm	$w = 2.0$ (inverse)	$w = 0.5$ (inverse)	No weighting	$w = 0.5$	$w = 2.0$
Sum of differences	Average	0.874	0.948	0.970	0.966	0.962
Sum of differences	Absolute	0.876	0.946	0.971	0.968	0.942
Sum of differences	Euclidean	0.871	0.941	0.971	0.974	0.972
Sum of differences	Maximum	0.869	0.939	0.970	0.974	0.971
Root square distance	Average	0.571	0.882	0.964	0.956	0.918
Root square distance	Absolute	0.829	0.872	0.968	0.972	0.973
Root square distance	Euclidean	0.828	0.871	0.966	0.970	0.964
Root square distance	Maximum	0.828	0.812	0.966	0.973	0.974
Singular values distance	Average	0.820	0.816	0.923	0.959	0.976
Singular values distance	Absolute	0.822	0.846	0.928	0.966	0.970
Singular values distance	Euclidean	0.821	0.817	0.903	0.965	0.967
Singular values distance	Maximum	0.821	0.951	0.892	0.953	0.966
Pearson's corr. coefficient	Average	0.621	0.865	**0.977**	0.976	0.970
Pearson's corr. coefficient	Absolute	0.535	0.700	0.785	0.802	0.806
Pearson's corr. coefficient	Euclidean	0.573	0.781	0.882	0.901	0.895
Pearson's corr. coefficient	Maximum	0.779	0.856	0.935	0.969	0.974
Spearman's rank corr. coeff.	Average	0.960	0.965	0.950	0.972	0.973
Spearman's rank corr. coeff.	Absolute	0.918	0.900	0.892	0.884	0.885
Spearman's rank corr. coeff.	Euclidean	0.963	0.960	0.963	0.959	0.961
Spearman's rank corr. coeff.	Maximum	0.962	0.962	0.883	0.961	0.966

This table shows the AUROC measure of descriptor predictive use for five different methods for computing distance between two matrices (defined in table 3.2), four different approaches to vector norms (as shown in table 3.3) as well as four different weights (see equation 3.11) and without any weighting. The bold number is the highest found AUROC value in the table.

3.3 Trace-sigma-based shape descriptor

The best measured value of the AUROC measure for the cross-correlation shape descriptor is 0.977 (table 3.4); a value showing high distinguishing power of the descriptor to separate the similar and dissimilar shapes. Nonetheless, since the

value is not 1.0, it also means that no single threshold can distinguish the similar and different protein domain shapes without introducing at least one false-positive or false-negative result. As this is the case, it seems reasonable to introduce another shape descriptor in an attempt to increase the accuracy, be it on its own or in combination with the cross-correlation shape descriptor.

3.3.1 Derivation of the trace-sigma descriptor

The motivation behind this new descriptor is to find the minimal distance between two three-dimensional objects. There are, however, many different methods for defining distance between three-dimensional objects; nonetheless, it is interesting to consider the distance in terms of their respective spherical harmonics coefficients. Such distance between two objects (f_1 and f_2) can be defined using the spherical harmonics coefficients definition from equation 2.16, to be:

$$d\left(f_1, f_2\right) = \sum_{l=1}^{l_{max}} \left\{ \sum_{m=-l}^{l} \left[\int_{r=0}^{r_{max}} \left(c_{l,m}^{1}\left(r\right) - c_{l,m}^{2}\left(r\right) \right)^2 r^2 dr \right] \right\} \tag{3.12}$$

where:

f_1 and f_2 are the two objects between which the distance is to be computed.

$d\left(f_1, f_2\right)$ is the distance between the objects.

$c_{l,m}^{1}\left(r\right)$ and $c_{l,m}^{2}\left(r\right)$ are the respective spherical harmonics coefficients for objects 1 and 2.

Furthermore, assuming that one of the objects (in this case, let it be object f_2) has been rotated from the position of optimal overlay with object f_1 by some unknown rotation $\mathscr{R}\left(\alpha, \beta, \gamma\right)$ defined by the three Euler angles α, β and γ, thus leading to:

$$d\left(f_1, \mathscr{R}\left(\alpha,\beta,\gamma\right)f_2\right) = \sum_{l=1}^{l_{max}} \left\{ \sum_{m=-l}^{l} \left[\int_{r=0}^{r_{max}} \left(c_{l,m}^{1}\left(r\right) - \sum_{m'=-l}^{l} \left(\mathscr{D}_{l}^{m,m'}\left(\alpha,\beta,\gamma\right) c_{l,m'}^{2}\left(r\right) \right) \right)^2 r^2 dr \right] \right\} \tag{3.13}$$

where:

$\mathscr{R}\left(\alpha,\beta,\gamma\right)$ is a rotation about the Euler angles α, β and γ.

$\mathscr{D}_{l}^{m,m'}\left(\alpha,\beta,\gamma\right)$ is the Wigner $\mathscr{D}$ matrix used as discussed in section 3.2.2.

And by expanding the square of the difference, this can be re-written as:

$$\begin{aligned} d\left(f_1, \mathscr{R}\left(\alpha,\beta,\gamma\right)f_2\right) = \sum_{l=1}^{l_{max}} \Bigg\{ \sum_{m=-l}^{l} \Bigg[& \int_{r=0}^{r_{max}} \left(c_{l,m}^{1}\left(r\right) \right)^2 r^2 dr \\ & + \int_{r=0}^{r_{max}} \Big(\sum_{m'=-l}^{l_{max}} \left(\mathscr{D}_{l}^{m,m'}\left(\alpha,\beta,\gamma\right) c_{l,m'}^{2}\left(r\right) \right) \Big)^2 r^2 dr \\ & - 2 \int_{r=0}^{r_{max}} \left(c_{l,m}^{1} \sum_{m'=-l}^{l_{max}} \left(\mathscr{D}_{l}^{m,m'}\left(\alpha,\beta,\gamma\right) c_{l,m'}^{2}\left(r\right) \right) \right) r^2 dr \Bigg] \Bigg\} \end{aligned} \tag{3.14}$$

From equation 3.14, it is clear that the first two terms (lines 1 and 2) are rotation invariant as mentioned in section 1.4.4. Nonetheless, the cross-term (line 3) depends on rotation and therefore can be minimised by finding the rotation which makes the spherical harmonics coefficients of the two compared objects most similar in terms of their numerical values.

Now, in order to find the minimal distance between two objects, the equation 3.14 needs to be minimised; therefore, focusing on the cross-term of the spherical harmonics distance equation, it can be noted that the Wigner $\mathscr{D}$ matrices are invariant to the radius r and therefore can be moved outside the integral. Also, the factor of -2 can be omitted as it is a constant and therefore does not affect minimisation (except for changing it into maximisation due to sign change). Therefore:

$$\begin{aligned} & \arg\min_{\alpha,\beta,\gamma} \left(d\Big(f_1, \mathscr{R}\left(\alpha,\beta,\gamma\right)f_2 \Big) \right) = \\ & \arg\max_{\alpha,\beta,\gamma} \left\{ \sum_{l=1}^{l_{max}} \left\{ \sum_{m=-l}^{l} \left[\sum_{m'=-l}^{l} \left(\mathscr{D}_{l}^{m,m'}\left(\alpha,\beta,\gamma\right) \int_{r=0}^{r_{max}} \left(c_{l,m}^{1} \times c_{l,m'}^{2*} \right) r^2 dr \right) \right] \right\} \right\} \end{aligned} \tag{3.15}$$

Where the complex conjugate on the spherical harmonics coefficient $c^{2*}_{l,m'}$ comes from the Hermitian inner product definition. Now, the integral can be denoted as:

$$e_{l,m,m'} = \int_{r=0}^{r_{max}} \left(c^{1}_{l,m} \times c^{2*}_{l,m'} \right) r^2 dr \tag{3.16}$$

Thus allowing the equation 3.15 to be written in matrix form as

$$\arg\min_{\alpha,\beta,\gamma} \left(d\Big(f_1, \mathscr{R}(\alpha,\beta,\gamma) f_2 \Big) \right) = \arg\max_{\alpha,\beta,\gamma} \left\{ \sum_{l=1}^{l_{max}} \left[\mathrm{tr}\Big(\mathscr{D}_l(\alpha,\beta,\gamma)\, E_l^T \Big) \right] \right\} \tag{3.17}$$

where:

$\mathscr{D}_l(\alpha,\beta,\gamma)$ is the Wigner $\mathscr{D}$ matrix with dimensions m and m'.

E_l is a matrix consisting of the integral results with dimensions m and m'.

$\mathrm{tr}(...)$ is the trace of matrix given in ... .

The reason for the trace to be used comes from the definition of the trace operator on matrix product (*i.e.* $\sum_i \left[\sum_j \left(A_{i,j} B_{j,i} \right) \right] = \mathrm{tr}(AB)$), which is clearly satisfied when equation 3.15 is written in matrix form. Consequently, it is interesting to consider the result of computing the singular value decomposition (SVD) of the E matrices, that is:

$$\begin{gathered} E_l^T = U_l^T \Sigma_l^T V_l \\ \therefore \quad \arg\min_{\alpha,\beta,\gamma} \left(d\Big(f_1, \mathscr{R}(\alpha,\beta,\gamma) f_2 \Big) \right) = \arg\max_{\alpha,\beta,\gamma} \left\{ \sum_{l=1}^{l_{max}} \left[\mathrm{tr}\Big(\mathscr{D}_l(\alpha,\beta,\gamma)\, U_l^T \Sigma_l^T V_l \Big) \right] \right\} \end{gathered} \tag{3.18}$$

Now, using the fact that matrix multiplication order is irrelevant under the trace operator, the matrix order can be changed to

$$\arg\min_{\alpha,\beta,\gamma} \left(d\Big(f_1, \mathscr{R}(\alpha,\beta,\gamma) f_2 \Big) \right) = \arg\max_{\alpha,\beta,\gamma} \left\{ \sum_{l=1}^{l_{max}} \left[\mathrm{tr}\Big(V_l^T \mathscr{D}_l(\alpha,\beta,\gamma)\, U_l \Sigma_l^T \Big) \right] \right\} \tag{3.19}$$

Then, by realising that the matrices U_l, V_l and $\mathscr{D}_l$ in equation 3.19 are all rotation matrices, it becomes clear that the maximum of the right hand side of the equation 3.19 is when they collapse into the identity matrix I (this is because the diagonal elements of any rotation matrix have maximum of one), and therefore:

$$\begin{aligned} V_l^T \mathscr{D}_l(\alpha,\beta,\gamma)\, U_l &= I \\ \therefore \qquad \mathscr{D}_l(\alpha,\beta,\gamma) &= V_l^T U_l \end{aligned} \tag{3.20}$$

What then leads to the following equation:

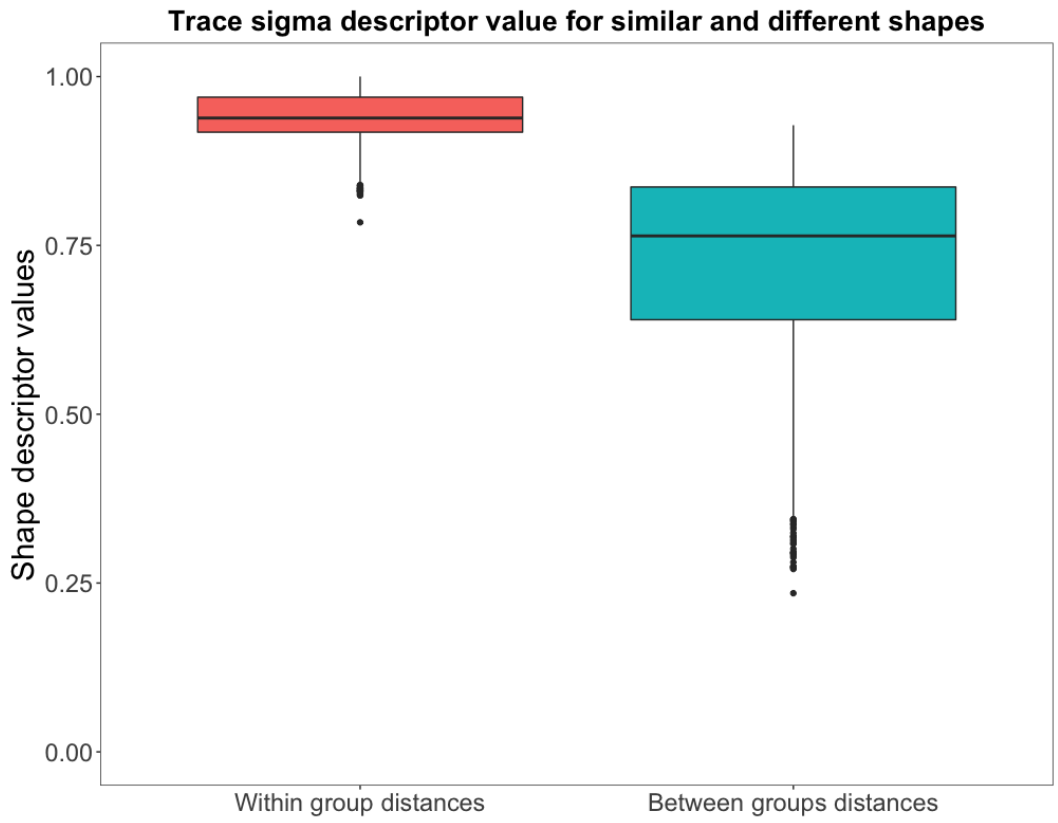
$$\arg\min_{\alpha,\beta,\gamma}\left(d\Big(f_1, \mathscr{R}(\alpha,\beta,\gamma) f_2\Big)\right) \leq \sum_{l=1}^{l_{max}}\left[\mathrm{tr}\Big(\Sigma_l^T\Big)\right] \tag{3.21}$$

It should be noted that the maximisation as per equation 3.20 is done in equation 3.21 for each band l separately. Since it is not possible for the optimal rotation Euler angles α, β and γ to be different for each band of spherical harmonics, it follows that this is not the absolute minima of the shape distance $d(f_1, \mathscr{R}(\alpha,\beta,\gamma) f_2)$, but rather its approximation under the assumption that overlay of each band l between the two objects is optimal with the same Euler angles α, β and γ. For this reason, the right-hand part of equation 3.21 is stated as larger or equal to the minimal distance instead of being equal to it.

3.3.2 Accuracy of the trace-sigma descriptor

In order to evaluate how accurate the trace-sigma shape-distance is in separating the similar and different shapes, it was implemented and used to compute the shape distances within and between the test dataset groups (in identical fashion as in section 3.2.6). The resulting box plot showing the distributions of the trace-sigma shape-descriptor values for similar and different shapes is shown in figure 3.4.

Figure 3.4: The distributions of the trace sigma descriptor for protein domains with similar and different shapes



This figure shows the distribution of the trace sigma descriptor (defined as per the equation 3.21) values for the test dataset. The red box plot on the left describes the distribution of the distances between similar shapes (*i.e.* shapes within a single group), whereas the turquoise box plot on the right shows the distribution between shapes in different groups.

Consequently, to enumerate how much the two distributions differ and therefore how much distinguishing power the trace-sigma descriptor has, the AUROC measure was computed. The value for the trace-sigma descriptor is **0.984**; this is a higher number than the cross-correlation-based descriptor discussed in the previous chapter, but on the other hand it still does not lead to a perfect separation. In order to explore how well the two descriptors agree with each other and therefore whether any extra information can be obtained by their combination, the relative pair ordering needed to be explored.

3.3.3 Comparison of trace-sigma and cross-correlation results

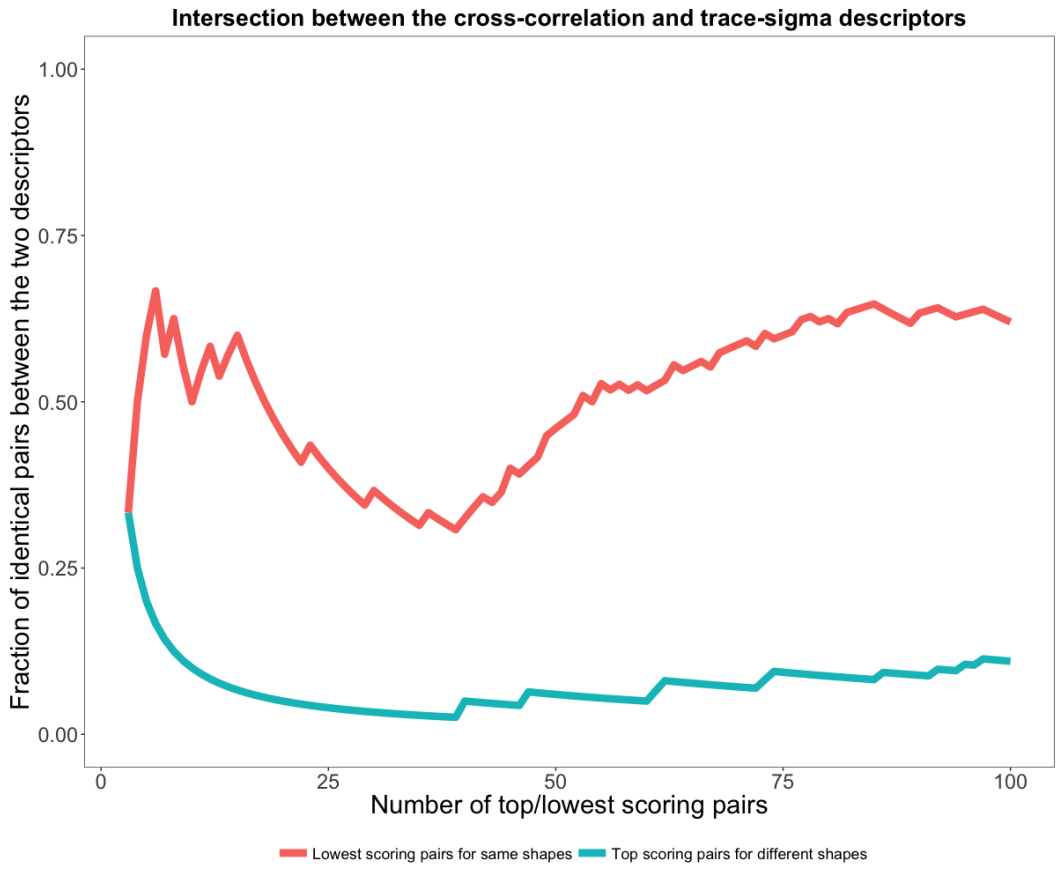
In order to answer this question, is it interesting to calculate the x lowest scoring pairs with similar shape (in other words, taking the lowest x values from the within-group distances distribution) for both shape descriptors. Then, it is possible to compute the intersection of these two vectors and consider the size of the intersection as a fraction of the maximum possible intersection size, x. Similarly, the highest scoring pairs with different shapes can be taken from the between-group distributions of the two descriptors and the same metric of the intersection size as a fraction of maximum possible intersection size can be obtained.

The reason for selecting the lowest scoring pairs with similar shape and the highest scoring pairs with different shape is that these are the most likely false negatives and false positives based on any threshold. Therefore, by selecting these two sets of values, it is possible to see if the two shape descriptors struggle with the same protein domain pairs, or if they struggle with different pairs and therefore that their combination does increase the information content over using just one of them.

The fraction metric devised in the previous paragraph has the property of being one when there is no difference between the two descriptors, while it would have a value of zero if the two shape descriptors would provide completely different information (in terms of failing for different structures). Now, the fraction metric has been implemented and applied to various values of x and the resulting plot is shown in figure 3.5.

As the figure 3.5 shows, only about 10% of the top 25 scoring protein domain pairs with different shapes are shared by the two shape descriptors, while approximately 40% of the lowest scoring protein domains with similar shapes are shared between the two shape descriptors. These results suggest that by combining the results of the two descriptors, better separation of similar and dissimilar shapes could be reached. It is worth noting that by combining the two descriptors, it is not necessarily meant combining their values, but rather combination of their

Figure 3.5: The intersection of the false-positive and false-negative region between the two shape descriptors



This figure was produced by computing the intersection between the cross-correlation and the trace-sigma descriptors for the top x scoring pairs with different shapes (shown in turquoise colour). Similarly, the intersection between the two descriptors was computed for the x lowest scoring pairs with similar shapes (shown in red colour). The value of x used was varied in the range from 3 to 100.

predictions (*i.e.* something akin "if cross-correlation value is higher than x or if trace-sigma value is higher than y..."). For this very reason, it is not possible to compute the AUROC value to demonstrate whether this is indeed the case.

3.3.4 Features of the trace-sigma descriptor

With the trace-sigma descriptor defined and its usefulness discussed, it is worth exploring the mathematical features that the descriptor intrinsically has. Firstly, the trace sigma descriptor is rotation invariant as the singular values matrix Σ and its trace are rotation invariant by definition. This fact can also be seen from

equation 3.19, which shows that the maximum (and therefore the final descriptor value) is reached when all the rotation matrices collapse into a single identity matrix; the only situation when this is true is when the rotation introduced by the singular value expansion matrices U and V reverse the rotation introduced by the Wigner $\mathscr{D}$ matrix, as shown in equation 3.20 (line 2).

On the other hand, the trace sigma descriptor is sensitive to translation. This property of the descriptor is not preferred and its effects need to be reduced or removed completely; this can be done using the centre of density approach as discussed in section 2.3.1; however, there are drawbacks associated with this method as discussed in the same section. Alternatively, the intrinsically centred Patterson maps could be used as input instead of the full density maps, as discussed in section 2.3.2 along with the disadvantages of this approach.

Interestingly, the trace-sigma descriptor does not require any matrix-distance or vector-norm computations, as the reduction of the three-dimensional array of descriptors is done intrinsically and follows from the spherical-harmonics-coefficients distance equation (as shown in section 3.3.1). Nonetheless, the descriptor values can range dramatically and their absolute values can be hard to interpret. To reduce this issue, it is possible to introduce a normalisation procedure based on the Pearson's correlation coefficient equation (shown in table 3.2 and for convenience repeated in equation 3.22), where the numerator is the correlation formula and the denominator is normalisation factor.

$$p = \frac{\sum_{i=1}^{n} \left(\left(a_i - \mu_a \right) \left(b_i - \mu_b \right) \right)}{\sqrt{\left(\sum_{i=1}^{n} \left(a_i - \mu_a \right)^2 \right)} \sqrt{\left(\sum_{i=1}^{n} \left(b_i - \mu_b \right)^2 \right)}} \tag{3.22}$$

where:

n is the length of vectors $\mathbf{A}$ and $\mathbf{B}$ between which the correlation is to be computed.

a_i and b_i are i^{th} elements of the vectors $\mathbf{A}$ and $\mathbf{B}$, respectively.

μ_a and μ_b are the mean values of the vectors $\mathbf{A}$ and $\mathbf{B}$, respectively.

Therefore, by modifying the equation 3.16 as shown in equation 3.23, the E matrix now does not express the cross-term of the distance equation (3.14) in absolute terms, but rather in relative terms with respect to the total sum of frequencies in the given band l. Consequently, the equations following from the modified E matrices to the descriptor are identical to the original equations (3.16 to 3.21), but the resulting normalised trace sigma descriptor now has values between -1.0 for completely opposite shapes and 1.0 for identical shapes.

$$e_{l,m,m'} = \frac{\int_{r=0}^{r_{max}} \left(c_{l,m}^{1} \times c_{l,m'}^{2*} \right) r^2 dr}{\sqrt{\int_{r=0}^{r_{max}} \left(c_{l,m}^{1} \right)^2 r^2 dr} \sqrt{\int_{r=0}^{r_{max}} \left(c_{l,m}^{2*} \right)^2 r^2 dr}} \tag{3.23}$$

Finally, it is worth noting that the normalised trace-sigma descriptor does require the E matrices to be computed for each pair of compared shapes independently. This feature of the descriptor causes a rather high computational cost when compared to the cross-correlation descriptor. The reason for the difference in computational cost is that the cross-correlation descriptor is based on cross-correlation between different spheres of the *same* structure and therefore these can be pre-computed for each structure; consequently, comparison of two structures only requires the comparison of the τ matrices and a vector norm.

On the other hand, the normalised trace-sigma descriptor allows only the spherical harmonics coefficients to be pre-computed, as the calculation of the E matrices does require information from two *different* structures to be completed; *i.e.* it needs to be done for each pair independently. Furthermore, multiple matrix singular value decompositions are required after the computation of the E matrices and this further increases the computational cost of each distance calculation.

3.4 Rotation-function descriptor

While the normalised trace-sigma descriptor is able to differentiate the similar and different shapes well, it cannot distinguish them perfectly. This finding therefore

warrants exploring yet another possible approach to shape-similarity detection, presumably with higher shape-similarity distinguishing power. One possible approach to this task is to re-consider the equation 3.15 (for convenience repeated here).

$$\begin{aligned} &\underset{\alpha,\beta,\gamma}{\arg\min}\left(d\Big(f_1,\mathscr{R}(\alpha,\beta,\gamma)f_2\Big)\right) = \\ &\underset{\alpha,\beta,\gamma}{\arg\max}\left\{\sum_{l=1}^{l_{max}}\left\{\sum_{m=-l}^{l}\left[\sum_{m'=-l}^{l}\left(\mathscr{D}_l^{m,m'}(\alpha,\beta,\gamma)\int_{r=0}^{r_{max}}\left(c_{l,m}^{1}\times c_{l,m'}^{2*}\right)r^2dr\right)\right]\right\}\right\} \end{aligned} \tag{3.24}$$

The equation shows that the minimal distance between two objects is obtained by finding the Euler angles α, β and γ which maximise the right hand side of the equation. By finding the optimal α, β and γ values, it is therefore possible to obtain a shape descriptor without the limitation discussed in section 3.3.1 and presumably with higher distinguishing power between the similar and different shapes.

In order to find the optimal α, β and γ values, the naïve approach of computing the distance from equation 3.24 for multiple α, β and γ values and consequently selecting the highest value (a form of three-dimensional search) could be used. However, should simply the increments of 5° angles be tried for the allowed ranges of α, β and γ, this approach would require $72 \times 72 \times 36 = 186,624$ computations of both the Wigner $\mathscr{D}$ matrices and the integral; the computational cost of this approach is prohibitive for this project even if the algorithm could be optimised. Therefore, another approach is required.

3.4.1 The SO(3) Fourier transform approach

One approach which provides an answer to the required maximisation problem in a shorter time is based on the SO(3) group Fourier transform. However, in order to explain how this is done and why it solves the maximisation problem, the mathematical concepts behind it need to be explored, starting with the Euler angles.

The Euler angles

The Euler angles were introduced by Euler (1776) and they represent three rotations around the axes of the three-dimensional Cartesian co-ordinate system, so that rotation of any object can be unambiguously determined with respect to the co-ordinate system. There are various conventions on the order and the axes to which the rotations are applied; throughout this project, the convention ZYZ (that is α and γ are rotations along the z axis and β is rotation along the y axis) with intrinsic rotations (that is, the axes of the co-ordinate system are attached to the moving system as opposed by using fixed axes for extrinsic rotations) shall be used.

Given that the three Euler angles are rotations along at least two axes of the co-ordinate system, it follows from the Euler's rotation theorem (Euler, 1776) that their combination is a single rotation along some axis passing through the co-ordinate origin - in other words, the Euler angles represent any rotation about the co-ordinate origin. Now, the group of all rotations about the co-ordinate origin in the three-dimensional Cartesian space is known as the SO(3) group (also the 3D Rotation group).

The SO(3) Group

Typically, the SO(3) group is defined as the group of all 3×3 orthogonal matrices ($\mathbf{M}$) with the property that $\mathbf{M}^T\mathbf{M} = \mathbf{M}\mathbf{M}^T = I$ and determinant of one - such matrices are known as the rotation matrices or special orthogonal matrices (thus giving name to the group). From the definitions of the SO(3) group and the Euler angles, it is clear that every matrix in the SO(3) group has associated Euler angles, as noted, for example, by Kostelec and Rockmore (2008). The following equations show the formulae for converting the Euler angles onto the special orthogonal matrix $\mathbf{M}$ and the reverse.

$$\begin{gathered}\mathrm{Sa} = \sin(\alpha) \;\; ; \;\; \mathrm{Sb} = \sin(\beta) \;\; ; \;\; \mathrm{Sc} = \sin(\gamma) \\ \mathrm{Ca} = \cos(\alpha) \; ; \; \mathrm{Cb} = \cos(\beta) \; ; \; \mathrm{Cc} = \cos(\gamma)\end{gathered} \tag{3.25}$$

$$\mathbf{M} = \begin{pmatrix} \mathrm{Ca} * \mathrm{Cb} * \mathrm{Cc} - \mathrm{Sa} * \mathrm{Sc} & \mathrm{Sa} * \mathrm{Cb} * \mathrm{Cc} + \mathrm{Ca} * \mathrm{Sc} & -\mathrm{Sb} * \mathrm{Cc} \\ -\mathrm{Ca} * \mathrm{Cb} * \mathrm{Sc} - \mathrm{Sa} * \mathrm{Cc} & -\mathrm{Sa} * \mathrm{Cb} * \mathrm{Sc} + \mathrm{Ca} * \mathrm{Cc} & \mathrm{Sb} * \mathrm{Sc} \\ \mathrm{Ca} * \mathrm{Sb} & \mathrm{Sa} * \mathrm{Sb} & \mathrm{Cb} \end{pmatrix}$$

$$\begin{aligned} \alpha &= \mathtt{atan2}\left(\mathbf{M}_{3,2}, \mathbf{M}_{3,1}\right) \\ \beta &= \cos^{-1}\left(\mathbf{M}_{3,3}\right) \\ \gamma &= \mathtt{atan2}\left(\mathbf{M}_{2,3}, -\mathbf{M}_{1,3}\right) \end{aligned} \tag{3.26}$$

where:

$\mathbf{M}_{i,j}$ is the rotation matrix $\mathbf{M}$ element row i and column j.

i and j are in range $(1, 3)$.

`atan2` is defined as in Organick (1966).

Moreover, it can be shown that any function f defined on the SO(3) group (*i.e.* $f \in \mathrm{SO}(3)$) can be expanded onto the Euler angles as shown in the following equation. The proof can be found in Kostelec and Rockmore (2008), who further argue that "*the Euler angle expansion provides a natural co-ordinate system for working with functions on SO(3), allowing us to write a function* $f(g)$ *for* $g \in SO(3)$ *as* $f(\alpha, \beta, \gamma)$*, where* $0 \leq \alpha, \gamma < 2\pi$ *and* $0 \leq \beta \leq \pi$".

$$f = f(\alpha, \beta, \gamma) = z(\alpha)\, y(\beta)\, z(\alpha) \tag{3.27}$$

where:

$z(\delta)$ corresponds to a rotation of δ radians along the z axis.

$y(\delta)$ corresponds to a rotation of δ radians along the y axis.

f is a function defined in SO(3).

Wigner $\mathscr{D}$ matrices

At this point, it is worth to return to the Wigner $\mathscr{D}$ matrices, already briefly discussed in the section 3.2.2, as they have interesting properties in regard to the SO(3) group. They form an irreducible representation of the SO(3) group with the formal definition being:

$$\mathscr{D}^{l}_{m,m'}(\alpha,\beta,\gamma) = e^{-im\alpha} d^{l}_{m,m'}(\beta)\, e^{-im'\gamma} \tag{3.28}$$

where:

$d^{l}_{m,m'}(\beta)$ is the Wigner d matrix defined in equation 3.29.

$$d^{l}_{m,m'}(\beta) = \sqrt{\Big[(l+m')!\,(l-m')!\,(l+m)!\,(l-m)!\Big]} \times \sum_{s} \left[\frac{(-1)^{s}}{(l+m-s)!s!\,(m'-m+s)!\,(l-m'-s)!} \Big(cos\frac{\beta}{2}\Big)^{2l+m-m'-2s} \Big(sin\frac{\beta}{2}\Big)^{m'-m+2s} \right] \tag{3.29}$$

where:

the sum is take over all integer values of s that make the factorials non-negative.

this is the formal definition given by Wigner (1931).

Furthermore, as discussed by Kostelec and Rockmore (2008), application of the Peter-Weyl theorem (Peter and Weyl, 1927) to the Wigner $\mathscr{D}$ matrices leads to the following orthogonality relationship:

$$\int_{\alpha=0}^{2\pi} \left\{ \int_{\beta=0}^{\pi} \left[\int_{\gamma=0}^{2\pi} \Big(\mathscr{D}^{l_1}_{m_1,m'_1}(\alpha,\beta,\gamma)\, \mathscr{D}^{l_2 *}_{m_2,m'_2}(\alpha,\beta,\gamma) \Big)\, d\gamma \right] sin\beta\; d\beta \right\} d\alpha = \frac{8\pi^2}{2l_1+1} \delta_{l_1,l_2} \delta_{m_1,m_2} \delta_{m'_1,m'_2} \tag{3.30}$$

where:

$\delta_{a,b}$ is the Dirac delta function centred at point a, b.

This orthogonal relationship means that the Wigner $\mathscr{D}$ matrices could be used as the basis function for Fourier transform on the SO(3) group (in a similar

way the spherical harmonics functions are used as the basis functions for the Fourier transform on the surface of a sphere).

The SO(3) Fourier transform

And indeed, such Fourier transform is discussed, for example, by Kostelec and Rockmore (2008), who show that any square integrable function (function which satisfies $\int_{-\infty}^{\infty} |f(x)|^2 dx < \infty$) defined on SO(3) group can be expanded onto its SO(3) Fourier transform coefficients as follows:

$$f(\alpha,\beta,\gamma) = \sum_{l=0}^{l_{max}} \left\{ \sum_{m=-l}^{l} \left[\sum_{m'=-l}^{m} \left(a^l_{m,m'} \mathscr{D}^l_{m,m'}(\alpha,\beta,\gamma) \right) \right] \right\} \tag{3.31}$$

where:

$a^l_{m,m'}$ is the SO(3) Fourier transform coefficient for band l and orders m and m'.

With the $a^l_{m,m'}$ coefficients given by the following equation:

$$a^l_{m,m'} = \frac{2l+1}{8\pi^2} \int_{\alpha=0}^{2\pi} \left\{ \int_{\beta=0}^{\pi} \left[\int_{\gamma=0}^{2\pi} \left(f(\alpha,\beta,\gamma)\, \mathscr{D}^{l*}_{m,m'}(\alpha,\beta,\gamma) \right) d\gamma \right] \sin\beta \; d\beta \right\} d\alpha \tag{3.32}$$

Finding the maximum correlation

Equipped with the Fourier transform on the SO(3) group, Kostelec and Rockmore (2008) show that if two functions f and g are defined on the surface of a sphere and have the spherical harmonics coefficients $c^1_{l,m}$ and $c^2_{l,m}$ respectively, then by combining the coefficients as per the following equation, a correlation measure is obtained (in other words, a measure of the overlap for a particular rotation represented by Euler angles α, β and γ).

$$C(\alpha,\beta,\gamma) = \sum_{l=1}^{l_{max}} \left\{ \sum_{m=-l}^{l} \left[\sum_{m'=-l}^{l} \left(c_{l,m}^{1} c_{l,m'}^{2*} (-1)^{m-m'} \mathscr{D}_{m,m'}^{l}(\alpha,\beta,\gamma) \right) \right] \right\} \tag{3.33}$$

Moreover, Kostelec and Rockmore (2008) also show that by taking the inverse SO(3) Fourier transform of the function $C(\alpha,\beta,\gamma)$ as defined in the equation 3.33, the overlap (or correlation) of the two functions f and g can be evaluated for all Euler angle values in a single step. Clearly, not all feasible values of the Euler angles are evaluated, but rather all values on a grid given by the sampling of the functions f and g - for more details, see Kostelec and Rockmore (2008).

Now, by noting that the equation 3.24 does combine the spherical harmonics coefficients almost identically to the equation 3.33, this suggests how the SO(3) Fourier transform approach can lead to finding the Euler angles which minimise the distance between two objects in their spherical harmonics coefficients space. More specifically, the only difference between equations 3.24 and 3.33 is the integral over the radial dimension and the $(-1)^{m-m'}$ element; however, it can be seen that once the integral is evaluated, the two equations become identical up to the sign, which can be accommodated easily as well.

Therefore, it follows that the already implemented computations for the E matrix can be simply used to obtain the function C. Consequently, function C could be used to find the Euler angles α, β and γ which maximise the correlation of the two structures and thus minimise the spherical harmonics coefficients distance, leading to a more accurate shape descriptor. This approach has the advantage of finding the minimal distance for all the shells and bands values at the same time and therefore should have increased accuracy as compared to the trace-sigma descriptor discussed above.

The Rotation function

It is worth noting that the rotation function of molecular replacement procedure in crystallography is based on the very same approach as just described, albeit it is typically applied to matching Patterson maps. The rotation function was introduced by Rossmann and Blow (1962) and improved upon in terms of computation cost and accuracy by Crowther (1972).

The rotation function has been implemented by the major molecular replacement software, such as the `ROTING` part of the `AMoRe` software package (Navaza, 1994), the `Phaser` software (McCoy et al., 2007) or the `MOLREP` software (Vagin and Teplyakov, 2010). Nonetheless, all of the aforementioned molecular replacement software uses the spherical Bessel expansion (discussed in section 2.2.1) rather than just the spherical harmonics expansion.

3.4.2 Definition of the rotation-function descriptor

With the approach to finding the minimal distance between two three-dimensional objects in their spherical harmonics coefficients now described, the rotation-function descriptor can be obtained. In order to do this, the Euler angles which minimise the distance between the two three-dimensional objects need to be computed.

Finding the optimal overlay angles

As discussed in the previous section, by taking the inverse SO(3) Fourier transform of the E matrix computed as per equation 3.16, a three-dimensional map with the three Euler angles as its grid co-ordinates and the correlation function $C(\alpha, \beta, \gamma)$ from equation 3.33 as the grid values is obtained. This map is known as the *rotation function*.

Therefore, by finding the highest value in this map and its co-ordinates, the three Euler angles which lead to highest correlation between the two three-dimensional objects are found. This is exactly the information required by the

right hand side of equation 3.24, so the three resulting Euler angles can now be substituted into the maximisation.

Computing the rotation-function descriptor

The result of this substitution is the equation 3.34, which is the formal definition of the rotation-function descriptor.

$$\underset{\alpha,\beta,\gamma}{\arg\min}\left(d\Big(f_1,\mathscr{R}\left(\alpha,\beta,\gamma\right)f_2\Big)\right)=\sum_{l=1}^{l_{max}}\left\{\sum_{m=-l}^{l}\left[\sum_{m'=-l}^{l}\left(\mathscr{D}_l^{m,m'}\left(\alpha_o,\beta_o,\gamma_o\right)E_{m,m'}^{l\ T}\right)\right]\right\} \quad (3.34)$$

where:

α_o, β_o and γ_o are the Euler angles of the highest peak of the inverse SO(3) Fourier transform map of the E matrix.

Normalisation of the rotation-function descriptor

Similarly to the trace-sigma descriptor (see section 3.3.4), when the rotation-function descriptor is computed using the equation 3.34, the resulting distance is the absolute value of this distance. As such, its magnitude will be affected by the total sum of the frequencies, making direct comparison between different pairs of three-dimensional objects difficult.

In order to normalise the rotation-function descriptor value to the same range and make direct comparisons straightforward, the same normalisation as devised for the trace-sigma descriptor can be used; specifically, by applying the Pearson's correlation coefficient normalisation approach to the E matrix (as per equation 3.23), the effect of the total sum of frequencies of each three-dimensional object is removed and the resulting distance between the objects is now in range from -1 to +1. The interpretation of this range is similar to the Pearson's correlation coefficient, that is -1 means that the two shapes are completely opposite, 0 means that the two shapes are not related and 1 means that the two shapes are identical.

3.4.3 Accuracy of the rotation-function descriptor

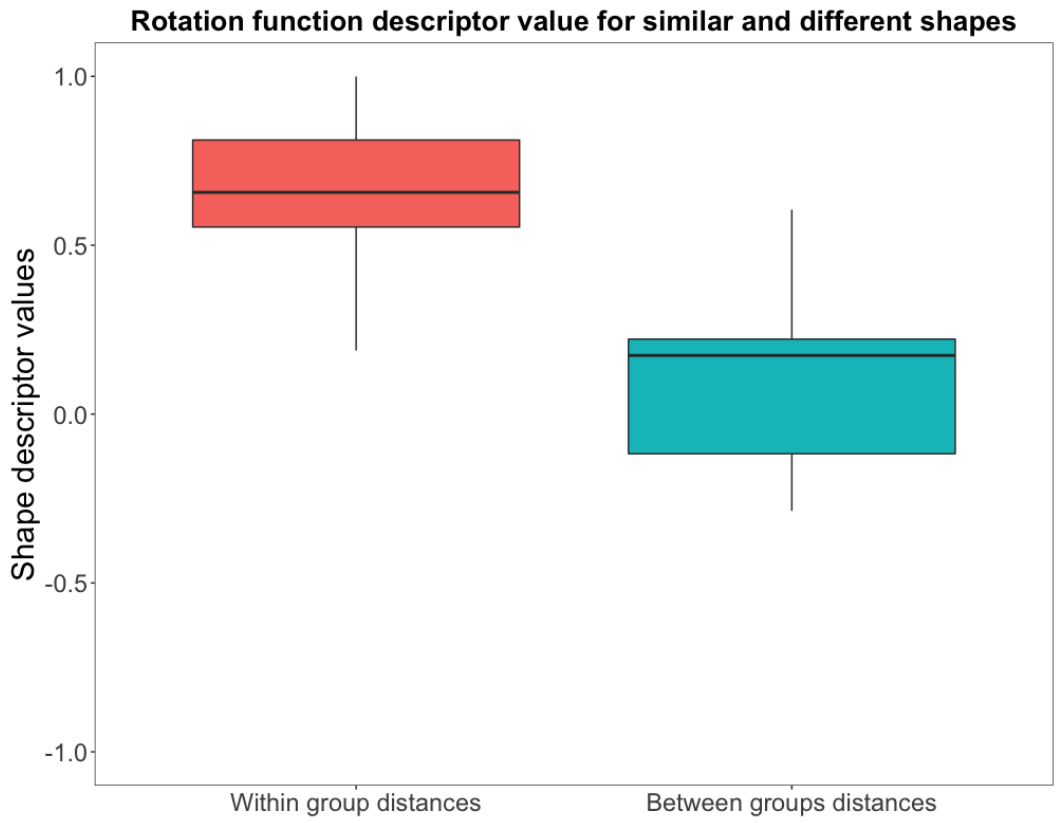
With the definition of the rotation-function descriptor available in equation 3.34, it was implemented using the `SOFT` library written by Kostelec and Rockmore (2003) in the C language. Then, in order to evaluate the accuracy of the rotation-function-based descriptor, it was used to compute the distances between all pairs of protein domains with similar shape (*i.e.* in the same groups in the test dataset) as well as the distances between pairs of protein domains with different shapes (that is, in different groups of the test dataset), using the same approach as for the cross-correlation and trace-sigma descriptors in the above sections. The box plots of the two resulting distributions of distances (distribution of similar shape distances and distribution of different shape distances) are shown in figure 3.6.

The AUROC measure of shape-similarity distinguishing power was consequently computed for these two distribution, yielding the value of **0.985**. This value is higher than the values computed for the trace-sigma descriptor (0.984) and the cross-correlation descriptor (0.977). This result is as expected, that is, the descriptor with least assumptions and simplifications (and costing the most computational resources) is the most accurate one.

3.4.4 Comparing the results of the three shape descriptors

Similarly to the trace-sigma descriptor, it is interesting to consider whether the three shape descriptors have issues with the same structure pairs, or whether they have different strengths and weaknesses and therefore their combination could improve the overall ability to detect similar and dissimilar shapes. Therefore, as discussed in section 3.3.3, it is possible to obtain the similar shape pairs with the lowest distance for the three shape descriptors; the intersection of these pairs from different descriptors shows whether the three descriptors found the same pairs as being 'most dissimilar', or whether the three descriptors differed. The same method can be applied to the highest scoring pairs of structures with different shapes.

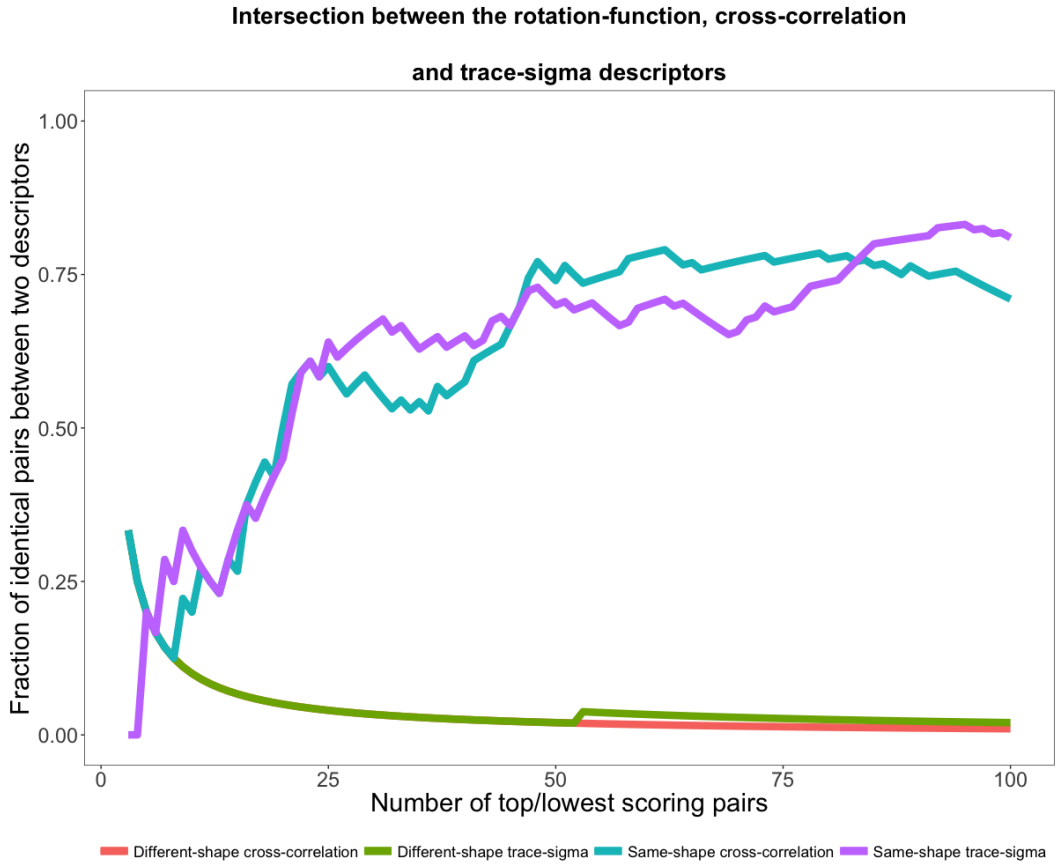
Figure 3.6: The distributions of the rotation-function descriptor for protein domains with similar and different shapes



This figure shows the distribution of the rotation function descriptor values computed by using the inverse SO(3) Fourier transform to find the minimal distance between three-dimensional objects as per equation 3.34. The red box plot on the left describes the distribution of the distances between similar shapes, whereas the turquoise box plot on the right shows the distribution of distances for pairs with different shapes.

From the figure 3.7, it can be seen that while the rotation-function descriptor overlaps both the trace-sigma and the cross-correlation descriptors in relatively high percentage for the low-scoring similar-shape pairs (around 65% overlap for the lowest scoring 25 pairs), there is little overlap (approximately 10% for the highest scoring 25 pairs) for the high-scoring different-shape pairs. This result shows that all the descriptors can be useful in combination with the other descriptors in an attempt to reduce the number of false positives and false negatives.

Figure 3.7: The intersection of the false-positive and false-negative regions between the rotation-function, trace-sigma and cross-correlation shape descriptors



This figure was produced by computing the intersection between the rotation-function, the trace-sigma and the cross-correlation descriptors for the lowest x scoring pairs with similar shapes (shown in turquoise colour for rotation-function versus cross-correlation descriptor and in purple colour for rotation-function versus trace-sigma descriptors). Similarly, the intersection between the descriptors was computed for the x top scoring pairs with different shapes (shown in green colour for rotation-function versus trace-sigma and in orange colour for rotation-function versus cross-correlation descriptors). The value of x used was in the range from 3 to 100.

3.4.5 Features of the rotation-function descriptor

Regarding the mathematical features of the rotation-function descriptor, it should firstly be noted that it is translation sensitive and therefore its accuracy will be affected by the accuracy of the centring technique. This is not a preferred feature, but its effects can be reduced using the approaches discussed in section 2.3.

On the other hand, the rotation-function descriptor is rotation invariant; this follows from the fact that the optimal rotation detection is intrinsically part of the descriptor. Moreover, the ability to compute the optimal rotation angles is rather

useful for the further applications of the protein domain shape descriptors, as it allows overlaying the two compared structures in real space directly. This ability is useful in, for example, the scenario where one of the structures is a fragment of a density map and the other is a domain database entry - in this case, the protein domain database entry can be placed and rotated in the full density map correctly, thus simplifying the task of automatically fitting domains into density maps.

Similarly to the trace-sigma descriptor, the rotation-function descriptor requires the E matrices (defined in equation 3.23) to be computed for each pair of structures independently. This means that only the spherical harmonics coefficients, but not the E matrices can be pre-computed for multiple structure comparisons. This fact in combination with the required SO(3) Fourier transform computation for each structure pair makes the rotation-function descriptor the most computationally costly descriptor discussed here.

Nonetheless, given that the normalised E matrices are required for the trace-sigma descriptor as well, the increase in the computational cost is only the SO(3) Fourier transform. Regarding the transform, Kostelec and Rockmore (2008) show that their algorithm implemented in the `SOFT` library (which is being used to implement the SO(3) Fourier Transform) requires $O(l_{max}^4)$ computations; an acceptable computational cost for this project.

3.5 Further considerations

With the three shape descriptors discussed and implemented, there are several factors which were not discussed so far, but which do have an effect on the shape descriptor computation and accuracy. These factors are considered separately from the other factors discussed in sections 3.2, 3.3 and 3.4, as they are common to all three descriptors.

3.5.1 Co-ordinate data and map data

First factor that should be considered is the difference between the two input data types, the co-ordinate data in PDB format and the density map data in the CCP4 MAP format. As these two input types contain different information, they need to be processed differently in order to arrive to a common data representation, which could subsequently be processed to compute the descriptors.

Co-ordinate data input

When the co-ordinate input type is to be used, the data are read in using the `MMDB` library (Krissinel et al., 2004) and the co-ordinates are translated so that the minimum x, y and z axes co-ordinates are placed at a positive distance (default value is 8 Å) from the co-ordinate origin. The translation is required to make sure the resulting theoretical density map is centred in the grid, while the extra space is added so that any interactions between asymmetric cells are reduced or removed. Also, it is possible to change the B factors of all atoms to a particular value; the rationale behind this option will be discussed more in section 3.5.3.

Consequently, the translated co-ordinate data are submitted to the `Clipper` library (Cowtan, 2002) for computation of a theoretical density map. At this point, the resolution of the map has to be decided; as this decision will be discussed in the section 3.5.3, it will not be discussed further here. Given the resolution up to which the theoretical density map should be computed and sampled, automatic determination of the angular resolution of the spheres on which the spherical harmonics expansion will be performed, as well as the sphere placement can be done as discussed in section 2.2.2.

Once the theoretical map is computed, its COD is moved to the origin of the co-ordinate system. A set of concentric spheres is then placed and grids are computed for each sphere as discussed above. Finally, each angular grid point on each sphere has its value interpolated using the closest eight density map grid points by

applying the trilinear interpolation method. Therefore, a set of concentric spheres with mapped density data are obtained.

Density map data input

A somewhat different approach is required when the input data are density-map values. The input file is initially read in using the `cmap` library (which is part of the CCP4 suite (Winn et al., 2011)) and is immediately checked for containing rectangular cell, as only rectangular (P_1) cells are currently supported. Then, the map is centred in the middle of the grid and checked for having the majority of the density in the middle of the grid instead of in the corners (as this is the case for some maps and the centre of density cannot be used to distinguish these cases).

Once the majority of map density is in the centre of the grid, the resolution can be determined from the map sampling and cell sizes, as discussed in section 3.5.3. However, since the experimental maps may differ in the amount of processing that has been done, many issues may arise at this point. These differences between the map processing include potential sharpening/blurring or masking with different values for the masked out region.

In order to allow uniform processing of the experimental maps, a procedure has been developed to first compute a map mask by increasing the B-factor (or blurring) by 250. From the resulting map, a threshold of 4 interquartile ranges from the third quartile is used to set all values under this threshold to 0. Consequently, the number of independent islands in the mask are detected by the Watershed algorithm (Lindeberg, 1991) and only large islands with high compactness are retained. Once a masked structure is computed, the sphere placement and angular resolution of the spheres can be done automatically as per section 2.2.2. Alternatively, the map masking procedure can be skipped and the original map can be used by supplying a single command line option.

Finally, the map sampling grid is re-indexed so that the map COD is moved to the origin of the co-ordinate system and the concentric spheres are drawn around

it. Then, each of the grid points on the surfaces of all the concentric spheres have their value interpolated using the trilinear interpolation method based on the surrounding eight closest map sampling grid points. Therefore, this procedure results in a set of concentric spheres with mapped density data; the same data representation as is reached for the co-ordinate data; consequently, from this point, both input data types can be processed identically.

3.5.2 Density maps and Patterson maps

When the descriptors computation is applied to co-ordinates or density maps, the phase information is intrinsically known. Nonetheless, it should be considered that the methods could be applied to molecular replacement candidate search and consequently phase information may not be available. Therefore, the methods were developed in such a way to allow for both types of computation, with and without phases. Moreover, in order to allow the methods to be applied to comparisons of structural data with and without phases against each other, Fourier transform followed by removal of phase information and inverse Fourier transform method was implemented. This approach was used to calculate the *phaseless* results in the following sections.

Furthermore, to determine how accurate the descriptors are without using the phase information, the AUROC values for the test dataset were computed with the phase information removed - the results are shown in tables 3.5, 3.6 and 3.7 and discussed in section 3.5.5. Finally, it is worth noting that both the Fourier transform and the inverse Fourier transforms are done using the `FFTW` library of Frigo (1999).

3.5.3 Map resolution limit

As mentioned in the previous sections, the resolution limit to which the map data should be processed is an important parameter which needs to be addressed. From the perspective of the input data, the map resolution limit is a function of

the map sampling, with the higher the resolution limit, the finer the sampling. Now, the extent to which the resolution-limit value can be varied differs between the theoretical maps computed from co-ordinate data and the experimental maps read directly.

Co-ordinate data and resolution limit

In the case of the theoretical maps, these can be computed to an arbitrary resolution limit. However, there is one more factor which should be considered alongside the resolution value - the atomic displacement parameter (ofter referred to as B). The B-factor is proportional to the variance of the Gaussian used to represent any particular atom in the density map; in other words, the smaller the B-factors, the narrower the approximating Gaussian distributions and the closer it resembles a single Dirac δ function.

Therefore, the resolution-limit value (which determines the map sampling for theoretical maps) needs to be proportional to the variance of the Gaussian function approximating the individual atoms (B-factor). This fact can be seen by considering the case of an atom located midway between two sampling grid points; if the B-factor value is too small, the contribution of this atom to the values of the two grid points will be weak and possibly hidden by the background.

Considering this relationship, it seems appropriate to leave the users the freedom of selecting any values they may like. Nonetheless, in order to keep the software as simple and automated as possible, a default value to be used when no resolution limit and B-factor values are supplied by the user need to be determined. To address this issue, the AUROC measure was computed for the test dataset using different values for resolution limit and B-factor; the results are shown in table 3.5, 3.6 and 3.7 and discussed in section 3.5.5.

Map data and resolution limit

A different situation can be seen for the experimental maps, which have the resolution limit and B-factors determined. Clearly, the experimental map resolution limit can be changed by re-sampling the sampling grid, but the "true" resolution (*i.e.* the clarity of map features) cannot be truly improved (although, the number of grid points can be increased, but no extra information can be extracted). Given that the finer the sampling grid, the larger the map and the higher the computational cost of processing it, the software was implemented so that while the user is free to decrease the resolution limit, when higher resolution limit value than that measured from the map data is supplied by the user, such input will be overridden by the maximum resolution limit allowed by the map. Nonetheless, in case the user would not supply a required resolution limit, a suitable default value is determined as per section 3.5.5.

Similarly, the B-factor is already incorporated in the experimental map. While it is possible to blur or sharpen the map by changing the B-factor value globally, it is not possible to set the B-factor to an universal value for the whole map. Therefore, there is no need for B-factor manipulation by the software, unless the user requires sharpening/blurring through a specific parameter.

3.5.4 Computation time

It is also interesting to consider the time required to compute distance in between a single pair of protein domains. The reason for this value being of interest is that while increased accuracy may be available by increasing the computational complexity, it may not be feasible to use very high accuracy settings for multiple comparisons.

In order to obtain an approximation of the range in which the execution time is, a single pair of structures with different shape was selected and submitted to the current shape descriptor implementation 100 times for each settings combination.

The resulting times were then averaged and are reported in tables 3.5, 3.6 and 3.7. While these times may vary on different computers and for different structure pairs, they give a good overview of the order of magnitude in which the computation can be done. The computer used to run all these shape comparisons was a MacBook Pro computer with macOS 10.13.4, 3.1 GHz processor (four cores) and 16 GB RAM.

3.5.5 Varying settings for resolution, B-factor and phase

With the major factors affecting all the three descriptors discussed, their individual effect on the accuracy and time of execution can be seen from the tables 3.5, 3.6 and 3.7. It is worth noting that the AUROC values in the previous sections were computed using the phase information, with resolution 8.0 Å and B-factor value of 80.

Deciding default values

With the information in the tables 3.5, 3.6 and 3.7, the decision as to the default values for the resolution, B-factor and phase information option can be made. However, when deciding the optimal default values for the execution of the code, it should be pointed out that if these default parameters were to be different for the three descriptors, the structure spherical harmonics expansion would have to be repeated for each unique set of options. Therefore, it would be preferable to find default values suitable for all three descriptors.

Firstly, it is worth noting that all the three tables show sharp decrease in the execution time when the phase information is included and the resolution is changed from 7.0 Å to 8.0 Å; in most such cases the execution time is almost halved. This observation by itself does not lead to any particular value which should be set as the default; however, it suggests that if a resolution lower than 7.0 Å could be selected without strongly affecting the accuracy (as measured by the AUROC measure), the execution time could be decreased considerably.

Table 3.5: The AUROC results for the cross-correlation descriptor with different resolution, B-factor and phase usage values

Resolution limit (Å)	B-factor value	AUROC with phase	Average time with phase (ms)	AUROC without phase	Average time without phase (ms)
2.0	40	0.959	78, 793.0	0.665	56, 179.2
2.0	80	0.953	75, 257.9	0.659	57, 077.4
2.0	120	0.935	75, 508.7	0.688	51, 084.3
3.0	40	0.968	12, 102.2	0.640	9, 490.9
3.0	80	0.969	12, 063.1	0.673	11, 372.2
3.0	120	0.966	13, 314.4	0.718	10, 398.6
4.0	40	0.976	3, 792.5	0.612	3, 534.3
4.0	80	0.970	3, 769.6	0.618	4, 126.1
4.0	120	0.969	4, 107.7	0.621	3, 770.8
5.0	40	0.978	2, 123.7	0.661	1, 992.8
5.0	80	0.974	2, 198.4	0.680	2, 403.9
5.0	120	0.971	2, 272.4	0.609	2, 217.1
6.0	40	0.976	1, 538.4	0.608	1, 544.3
6.0	80	0.972	1, 517.5	0.679	1, 654.0
6.0	120	0.968	1, 624.6	0.705	1, 552.5
7.0	40	0.982	1, 316.6	0.630	1, 298.9
7.0	80	0.974	1, 293.0	0.632	1, 431.9
7.0	120	0.976	1, 305.3	0.608	1, 345.3
8.0	40	0.978	626.1	0.576	597.3
8.0	80	0.977	630.6	0.635	720.2
8.0	**120**	**0.974**	**657.2**	0.687	656.5
9.0	40	0.965	560.8	0.637	531.2
9.0	80	0.977	544.5	0.661	606.6
9.0	**120**	0.973	544.1	**0.678**	**552.9**
10.0	40	0.922	478.5	0.670	476.1
10.0	80	0.938	481.1	0.673	539.6
10.0	120	0.940	502.5	0.611	508.5
11.0	40	0.908	447.4	0.702	446.9
11.0	80	0.922	457.1	0.591	490.3
11.0	120	0.950	467.3	0.584	455.9
12.0	40	0.845	433.5	0.579	429.4
12.0	80	0.933	437.7	0.542	474.9
12.0	120	0.960	449.2	0.647	448.0
13.0	40	0.847	425.6	0.590	418.3
13.0	80	0.946	415.7	0.668	451.9
13.0	120	0.946	439.4	0.708	431.4
14.0	40	0.866	428.2	0.619	418.3
14.0	80	0.924	415.0	0.642	456.1
14.0	120	0.932	424.6	0.640	416.1
15.0	40	0.842	426.2	0.716	413.4
15.0	80	0.894	407.3	0.753	456.4
15.0	120	0.910	424.9	0.753	426.0

This table shows the cross-correlation descriptor AUROC values computed using the test dataset for different settings of the resolution, B-factor and phase values. The average time was not computed for the whole dataset, but rather only for a single pair; the pair distance was computed 100 times and the execution time was then averaged. The bold values are the default parameters for distance computation with and without phases.

Table 3.6: The AUROC results for the trace-sigma descriptor with different resolution, B-factor and phase usage values

Resolution limit (Å)	B-factor value	AUROC with phase	Average time with phase (ms)	AUROC without phase	Average time without phase (ms)
2.0	40	0.989	355, 384.8	0.813	206, 296.8
2.0	80	0.987	373, 392.0	0.801	202, 364.7
2.0	120	0.986	384, 791.5	0.842	202, 696.7
3.0	40	0.989	54, 029.8	0.780	32, 605.8
3.0	80	0.988	54, 997.9	0.840	31, 797.7
3.0	120	0.986	55, 802.4	0.852	31, 680.5
4.0	40	0.987	14, 519.4	0.833	9, 185.0
4.0	80	0.986	14, 395.2	0.816	9, 499.4
4.0	120	0.985	15, 739.3	0.809	9, 636.6
5.0	40	0.989	6, 142.1	0.788	4, 214.9
5.0	80	0.986	6, 112.0	0.812	4, 426.3
5.0	120	0.983	6, 462.0	0.808	4, 243.5
6.0	40	0.987	3, 391.2	0.785	2, 563.9
6.0	80	0.983	3, 357.3	0.809	2, 507.0
6.0	120	0.982	3, 462.4	0.836	2, 478.4
7.0	40	0.988	2, 291.0	0.821	1, 860.5
7.0	80	0.983	2, 262.0	0.838	1, 826.0
7.0	120	0.984	2, 283.5	0.789	1, 855.6
8.0	40	0.983	1, 182.2	0.739	905.8
8.0	80	0.984	1, 189.3	0.815	971.6
8.0	**120**	**0.983**	**1, 307.2**	0.802	996.4
9.0	40	0.982	913.2	0.818	737.6
9.0	80	0.980	898.0	0.839	731.2
9.0	**120**	0.978	907.5	**0.854**	**750.9**
10.0	40	0.969	670.3	0.771	610.1
10.0	80	0.972	678.6	0.801	600.8
10.0	120	0.953	695.5	0.734	593.7
11.0	40	0.946	588.0	0.794	535.3
11.0	80	0.914	610.0	0.756	535.4
11.0	120	0.971	633.2	0.762	552.6
12.0	40	0.934	548.7	0.740	497.9
12.0	80	0.947	548.5	0.740	497.4
12.0	120	0.970	555.2	0.808	495.7
13.0	40	0.934	500.8	0.722	474.0
13.0	80	0.971	496.3	0.783	462.4
13.0	120	0.972	512.5	0.827	467.5
14.0	40	0.927	470.0	0.708	460.9
14.0	80	0.955	459.3	0.765	443.1
14.0	120	0.955	463.5	0.784	442.3
15.0	40	0.921	452.7	0.798	456.7
15.0	80	0.939	443.5	0.838	439.9
15.0	120	0.944	458.9	0.853	444.7

This table shows the trace sigma descriptor AUROC values computed using the test dataset for different settings of the resolution, B-factor and phase values. The average time was not computed for the whole dataset, but rather only for a single pair; the pair distance was computed 100 times and the execution time was then averaged. The bold values are the default parameters for distance computation with and without phases.

Table 3.7: The AUROC results for the rotation-function descriptor with different resolution, B-factor and phase usage values

Resolution limit (Å)	B-factor value	AUROC with phase	Average time with phase (ms)	AUROC without phase	Average time without phase (ms)
2.0	40	0.992	358, 636.2	0.761	205, 978.6
2.0	80	0.990	370, 748.0	0.758	212, 515.3
2.0	120	0.989	374, 093.8	0.780	207, 474.3
3.0	40	0.994	55, 395.9	0.744	33, 589.3
3.0	80	0.993	54, 051.4	0.804	34, 930.6
3.0	120	0.992	54, 854.1	0.801	32, 451.4
4.0	40	0.991	14, 617.4	0.786	9, 660.0
4.0	80	0.990	14, 186.6	0.771	10, 295.8
4.0	120	0.989	15, 529.6	0.761	10, 080.1
5.0	40	0.993	6, 106.6	0.756	4, 342.4
5.0	80	0.992	6, 145.1	0.770	4, 706.7
5.0	120	0.992	6, 263.1	0.753	4, 301.8
6.0	40	0.987	3, 476.2	0.752	2, 636.2
6.0	80	0.985	3, 354.1	0.757	2, 728.0
6.0	120	0.985	3, 413.6	0.775	2, 605.5
7.0	40	0.985	2, 359.7	0.782	1, 934.1
7.0	80	0.985	2, 287.1	0.799	2, 003.1
7.0	120	0.984	2, 304.1	0.737	1, 893.8
8.0	40	0.980	1, 221.7	0.712	952.5
8.0	80	0.985	1, 221.0	0.769	1, 073.6
8.0	**120**	**0.987**	**1, 346.6**	0.744	1, 011.5
9.0	40	0.978	924.8	0.783	780.4
9.0	80	0.986	918.2	0.798	804.6
9.0	**120**	0.986	922.0	**0.805**	**760.5**
10.0	40	0.973	681.6	0.730	626.7
10.0	80	0.979	687.6	0.756	641.8
10.0	120	0.973	703.6	0.686	606.2
11.0	40	0.951	598.1	0.771	544.0
11.0	80	0.943	647.8	0.750	589.2
11.0	120	0.965	652.4	0.727	563.9
12.0	40	0.937	555.1	0.719	519.5
12.0	80	0.944	556.7	0.716	525.6
12.0	120	0.967	572.3	0.775	502.8
13.0	40	0.907	499.2	0.685	476.1
13.0	80	0.954	499.2	0.741	483.8
13.0	120	0.963	506.5	0.766	450.9
14.0	40	0.899	478.0	0.687	476.5
14.0	80	0.943	465.9	0.742	468.1
14.0	120	0.957	468.6	0.752	451.3
15.0	40	0.912	461.4	0.781	444.6
15.0	80	0.932	444.8	0.811	463.2
15.0	120	0.944	446.2	0.816	446.2

This table shows the rotation function descriptor AUROC values computed using the test dataset for different settings of the resolution, B-factor and phase values. The average time was not computed for the whole dataset, but rather only for a single pair; the pair distance was computed 100 times and the execution time was then averaged. The bold values are the default parameters for distance computation with and without phases.

Considering the results shown in table 3.5, the highest accuracy with phase is obtained for resolution 7.0 Å and B-factor value 40, however, the difference between the highest AUROC score and the remainder of the scores is rather minimal, even when the resolution is decreased below 7.0 Å. Similarly, the highest AUROC score without the phase information is reached for resolution 3.0 Å and B-factor value 120, but the difference to the AUROC scores with resolution below 7.0 Å is marginal.

Similar results can be seen in tables 3.6 and 3.7, where the highest AUROC scores with phase are at resolution 5.0 Å with B-factor value 40 and resolution 3.0 Å and B-factor value 40, respectively; nonetheless, other parameter combinations have very similar scores. The results for the computation without phase are even more interesting, as the highest AUROC values are not reached for high resolution computations, but rather with resolution 9.0 Å and B-factor value 120.

Therefore, given the marginal differences in the with phase AUROC score for all three descriptors, while having sharp decrease in the execution time with resolution reaching 8.0 Å, the decision was reached to implement the default values for all three shape descriptors when using the phase information to resolution **8.0** Å and B-factor value **120**; the corresponding rows are shown in bold in the tables 3.5, 3.6 and 3.7. The reason for this decision is that this combination of values keeps the execution time shorter than higher resolution-limit values, while being the best available combination for the most accurate descriptor (the rotation-function-based descriptor) and being only minimally sub-optimal for the other two descriptors.

On the other hand, considering the phaseless AUROC scores, there are several considerations worth discussing. Firstly, it is clear that the phaseless AUROC scores are lower than with phase and this should not be surprising, as the phase information is missing from them. However, comparing the phaseless AUROC values for the three descriptors, it can be observed that the normalised-trace-sigma and normalised-rotation-function descriptors perform better than the cross-correlation

descriptor. This is an interesting realisation and it (somewhat *ad hoc*) furthers the argument that these descriptors are valuable additions which should be used in combination with the cross-correlation descriptor, even though they are more computationally expensive.

Considering these observations from the tables 3.5, 3.6 and 3.7, higher weight needs to be attributed to the trace sigma and rotation function descriptors, which are more accurate when phase information is not available. Since both these descriptors have highest accuracy for resolution of **9.0** Å, this resolution will be used as the default value. This resolution limit also leads to relatively high accuracy for the cross-correlation descriptor, allowing for usage of single resolution settings for all three descriptors. Regarding the B-factor value, the default will be set to **120**, as it leads to slightly higher accuracy for all three descriptors in the already decided resolution limit bracket.

Chapter 4

Symmetry detection

As the three-dimensional shape-similarity-distance descriptors were discussed in the previous chapter, the SO(3) Fourier transform and its inverse transform were also explored. The implementation of this method now allows considering the following case: A single structure can be expanded onto its spherical harmonics coefficients and the inverse SO(3) Fourier transform of the E matrix resulting from combining a single-structure's spherical-harmonics coefficients can be obtained. It will then be a three-dimensional map with the Euler angle α, β and γ values as its indices and the self-correlation scores for the single structure as its values. Consequently, it is interesting to consider how such result can be used to derive information about the single structure's symmetry.

This approach is similar to the crystallographic self-rotation function as suggested by Rossmann and Blow (1962) and expanded upon by Tong and Rossmann (1972) and which is typically used to detect non-crystallographic symmetries (NCS) in a crystal and provide constraints on molecular replacement (Tong, 2001). Nonetheless, the crystallographic self-rotation function is typically applied to the Patterson maps, while the approach described here is focused on phased maps obtained by either X-ray crystallography or EM methods.

4.1 The symmetry test dataset

Nonetheless, before the possibility of using the inverse SO(3) Fourier transform map for detecting symmetries in structures can be discussed further, the approach to testing any results and setting of any parameters needs to be explored. Similarly to the discussion of the shape-similarity descriptors in previous chapter, a test set is required to allow fast, consistent and reliable empirical determination of any parameters. The test set should contain examples of all symmetries currently known in the protein shapes as well as both types of the input data, co-ordinates and density maps.

Therefore, the PDB database (Berman et al., 2000) was queried for proteins with particular symmetries and from each symmetry defined by the database, a single example was obtained (except for tetrahedral, octahedral and icosahedral symmetries, for which more examples were obtained). This approach was done for both co-ordinate data and density maps and the PDB accession codes of all the samples, along with the reported symmetry are shown in table 4.1.

After the test-set structures were all obtained from the PDB database, a manual check was done for each of the structures to make sure that the reported symmetry is actually present in the data. The co-ordinate data were reviewed using the `PyMOL` (Schrodinger LLC, 2015) software, while the density map-data were checked using the `Chimera` (Pettersen et al., 2004) software. With the test-data set now available, the possibility of detecting symmetries using the already implemented inverse SO(3) Fourier transform map can be fully explored.

4.2 Finding symmetry peaks in the inverse SO(3) map

The inverse SO(3) Fourier transform map produced by computing a single structure overlay against itself will contain peaks located at any co-ordinates where the Euler angles forming these co-ordinates lead to highly correlated overlay of the structure with itself. One obvious peak must be at co-ordinates 0; 0; 0, as the

Table 4.1: The wwPDB entries forming a test set for symmetry detection method development

Co-ordinate file symmetry	PDB accession code	Map file symmetry	PDB accession code
C2	5NL2	C2	5TV4
C3	5VN3	C3	5UJZ
C4	5VKQ	C4	5VKQ
C5	3JCF	C5	5MCY
C6	5UVN	C6	5LI2
C7	4AAQ	C7	5JZH
C8	5H1Q	C8	5H1Q
C9	5GAQ	C9	5GAQ
C11	3JBL	C11	3JBL
C12	5NBZ	C12	4AV2
C13	4V2T	C13	4V2T
C14	3ZBI	C14	2YPW
C15	5WQ7	C15	5WQ7
C22	5FMW	C24	2Y9J
C24	5TCP	C30	5WC3
D2	5GRS	D2	5VY5
D3	3JBB	D3	5K12
D4	5NV3	D4	5NV3
D5	4AJ5	D5	4BED
D6	5LDF	D6	2J9I
D7	4S0R	D7	4AAR
D8	3J1B	D8	5JUL
D9	3J1F	D9	3J1C
D12	2WCD		
		TET	5MQ3
		TET	5JM9
		TET	4CI0
		OCTA	6EZM
		OCTA	5TRE
		ICOS	6B9Q
		ICOS	5VLY
		ICOS	5XS4
		ICOS	5NED
		ICOS	5UF6

This table shows the complete set of PDB structures selected as a test set for symmetry detection algorithm development. The grey cells are empty and the colour is used to signify this,

correlation of any object with itself (without any rotation being applied) must be 1.0. However, any other high-valued peaks would indicate rotations for which the object shape does not change much, if at all.

By considering that symmetry operations are defined as "*isometric transforms that leave the shape globally unchanged*" (Martinet et al., 2006), it is easy to see that the rotations described by the high-valued peaks in the inverse SO(3) Fourier transform map do conform to this definition and therefore that any struc-

ture with a symmetry group would have to have a set of appropriate peaks in the inverse SO(3) Fourier transform map. Consequently, by using the co-ordinates of these peaks, it should be possible to use the reverse logic and detect all symmetries present in the single structure from which the inverse SO(3) Fourier transform map was computed.

More specifically, the three Euler angles can describe any proper rotation of a three-dimensional object, but not its improper rotations (Morawiec (2004); improper rotations include reflections and are also known as rotoreflections or rotoinversions). Therefore, by using the inverse SO(3) Fourier transform to find rotations which do not change the shape of the object, improper rotations cannot be detected. This is generally a preferred feature for protein domains, as any particular protein domain should be considered separately from its mirror image, because chirality of molecules is generally of interest.

In view of these facts, it seems feasible to develop an algorithm for detecting symmetries present in a single input structure, be it defined by its atomic model co-ordinates or density map. From the previous sections (especially section 3.4), it can be seen that with only minor changes, all the required calculations have already been implemented up to the point of having the results of the inverse SO(3) Fourier transform for a single structure against itself. Therefore, the next step to be discussed is how the high-valued peaks can be reliably found.

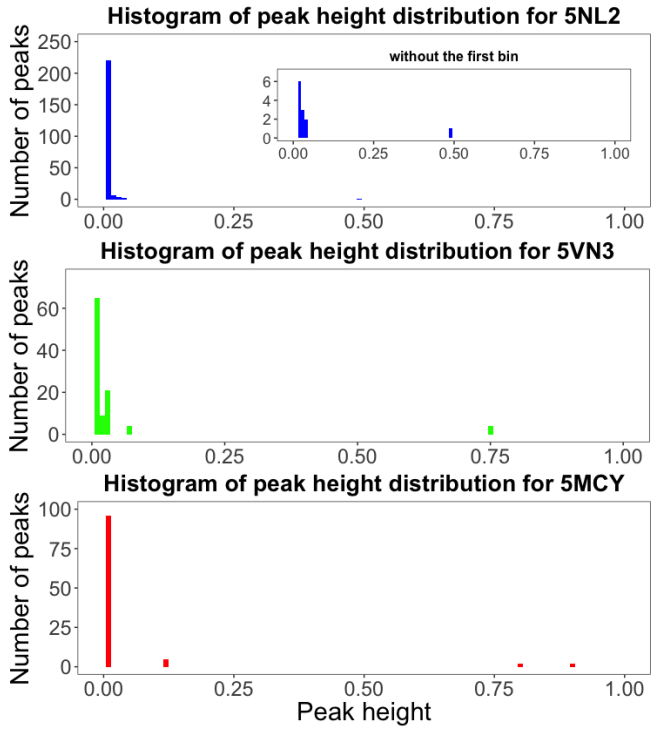
4.2.1 Initial peak searching

The first task for the symmetry detection algorithm is to find all the peaks with high enough value; that is, peaks for which the original and rotated structures are similar enough. In order to do so, first, all the peaks need to be located along with their value (thereafter referred to as *peak height*). To this end, a simple algorithm for checking each co-ordinate in the inverse SO(3) Fourier transform map for having higher peak height than all its neighbouring co-ordinates is used; the only complication of this algorithm being the map edges, where periodicity of

the map is assumed and neighbouring co-ordinates are therefore drawn from the opposite map edge.

With all the peaks detected, it was observed that there are between few hundreds and several thousands peaks conforming to the above algorithm for each of the test set structures. However, most of these peaks are expected to be background noise and as such need to be removed from the peak set. Therefore, the peak distribution needs to be considered, so that background could be determined and removed before subjecting the peak set to further analysis. Figure 4.1 shows the peak height distribution histograms for three selected structures.

Figure 4.1: Peak height distribution histograms for three selected protein structures



This figure shows the histograms of the peak height distributions for the three selected structures; the co-ordinate file with C2 symmetry peak height distribution histogram is shown in blue, the co-ordinate file with C3 symmetry peak height distribution histogram is shown in green and the density map file with C5 symmetry peak height distribution histogram is shown in red. All histograms have the *bin width* parameter set to 0.01. Note that the three plots do have different y-axis as the different structures do have different number of peaks.

The figure 4.1 shows that the peak height distribution tends to have many peaks with heights close to zero with several outliers having much larger heights. Therefore, by detecting the outliers, the possible symmetry peaks could be kept, while most of the background peaks could be removed, thus reducing the computational cost and possibility of false positives.

More specifically, the main outlier detection methods are based on computing the distribution statistics and using these to determine the threshold beyond which all values are considered outliers. One possibility is to use the mean and standard deviation statistics, while another option is to compute the interquartile range and distribution median. Given that the mean and standard deviation are more affected by outliers than the median and interquartile range, it seems appropriate to use the latter for outlier detection. Tukey (1977) suggests using 1.5 interquartile range from the median to detect outliers; however, by empirical testing of different number of interquartile ranges from the median, all structures in the test dataset had the outlier peaks still detectable with 3.0 interquartile ranges, while the number of background peaks was decreased as compared to the 1.5 interquartile ranges results. Therefore, the default value for the peak height outlier detection was set to 3.0 interquartile ranges with the option for the user to change which value if they so wish.

4.2.2 Local peak optimisation

With the peak detection now complete and the peak set reduced to the highest valued outliers, the resolution of the inverse SO(3) Fourier transform map needed to be considered. Each dimension of the inverse SO(3) Fourier transform map has $2l_{max}$ grid points, where l_{max} is the maximum bandwidth of the spherical harmonics expansion. However, the peak maximum is not likely to lie on any particular grid point, but rather in between the grid points and therefore, local optimisation of the peak position could improve the symmetry detection in general. To address this issue, a local optimisation procedure was developed.

Starting from the realisation that direct optimisation of the Euler angles is non-trivial, as the relationship between any one of the Euler angles and the rest is complex, the optimisation in terms of the rotation matrices was considered instead. Regarding the rotation matrices, their individual elements are related to rotations along the x, y and z axes by the trigonometric functions of the angle of the rotation

along that axis. Therefore, if a required rotation is known to be in between two rotation matrices with similar rotation angles along the axes, linear averaging of all the elements of the two rotation matrices and consequent normalisation produces orthogonal matrix with determinant one and the required rotation approximation. This method is described in more details by, for example Curtis et al. (1993) and Sharf et al. (2010).

Applying this method, the grid point with the maximum peak height and all of its immediate neighbours can have their Euler angle co-ordinates converted to rotation matrices (as per equation 3.25). Consequently, a new 3×3 matrix (M_A) can be constructed with each element being the sum of the same elements of each of the highest peak and its neighbours rotation matrices, weighted by the peak height of each of the grid points and divided by the sum length that was used to produce the rotation matrix. In other words, each element of this new matrix M_A is the weighted sum of the same elements of all the rotation matrices divided by the number of these elements.

Nonetheless, the M_A matrix is not necessarily a rotation matrix, as it needs to be normalised to become orthogonal and have determinant one. To achieve this, the M_A matrix can be expanded using the singular value expansion; this results in two rotation matrices (U and V) and the singular values matrix (Σ) as per the following equation. Consequently, by assuming all the singular values to be one, the determinant of the resulting matrix will be normalised to one and the matrix will be orthogonal. This can be seen from the fact that in the case that $\Sigma = I$, two rotation matrices are multiplied and the result must be a rotation matrix as well. Therefore, it follows that:

$$M_A = U\Sigma V^T \tag{4.1}$$

$$M_N = UIV^T = UV^T \tag{4.2}$$

where:

M_N is the normalised version of the matrix M_A.

Finally, the matrix M_N can be converted back to the Euler angles using equation 3.26, with the new Euler angles now being optimised for the peak height.

4.3 Cyclic symmetry detection

With the peaks of the inverse SO(3) Fourier transform map now located and locally optimised, an algorithm is required to check for presence of any symmetries in the peak list. Firstly, the cyclic symmetries are considered; the cyclic symmetries are defined as point groups for which any number of rotations by an angle of $\frac{360}{n}$ degrees along the symmetry axis does not change the shape of the rotated object. These symmetries are denoted as C_n in the Schoenflies notation (Schoenflies, 1891) or simply by n in the Hermann-Mauguin notation (Hermann (1928) and Mauguin (1931)), where n is the number of identical orientations the symmetry group has, also often called the *symmetry fold*. Therefore, in order to demonstrate the presence of this type of point group in the inverse SO(3) Fourier map, a set of peaks with identical symmetry axis and n different, consecutive rotations with their angle being an integer multiple of $\frac{360}{n}$ need to be detected.

4.3.1 Angle-axis representation

The task of finding peaks representing the fold n rotations with the same symmetry axis could be facilitated if the peaks could be grouped by their respective axes

of rotation. Nonetheless, both the Euler angles and the rotation matrices do not have their rotation axis trivially available, as they are both combinations of three rotations along the co-ordinate axes. However, both rotation matrices and the Euler angles representations of rotation can be converted to the angle-axis representation of rotation. The angle-axis representation parametrises any rotation in the three-dimensional space by a vector from the co-ordinate origin, which serves as the rotation axis and an angle, which states how much the object should be rotated along such a rotation axis. The angle-axis representation can be obtained from the rotation matrix M by using the following conversion equations.

$$\begin{aligned} \text{Angle} &= \cos^{-1}\Big(\frac{\text{tr}\,(M)-1}{2}\Big) \\ \text{x} &= \frac{M_{3,2}-M_{2,3}}{N} \\ \text{y} &= \frac{M_{1,3}-M_{3,1}}{N} \\ \text{z} &= \frac{M_{2,1}-M_{1,2}}{N} \\ \text{N} &= \sqrt{(M_{3,2}-M_{2,3})^2+(M_{1,3}-M_{3,1})^2+(M_{2,1}-M_{1,2})^2} \end{aligned} \tag{4.3}$$

where:

$\text{tr}\,(M)$ is the trace of matrix M.

$M_{i,j}$ is the rotation matrix M element row i and column j.

i and j are in range $(1, 3)$.

The conversion described by equations 4.3 is generally valid, but care must be taken when dealing with the singularities at angle values 0 and 180 degrees. In the case of 0 angle rotation, the appropriate rotation matrix would be the identity matrix, which would make the normalisation of the axis (N) 0 and thus cause division by 0 issue. Similarly, rotation by 180 degrees would have the rotation matrix with the non-diagonal elements 0, two diagonal element with -1 and one diagonal element with 1 (depending on along which axis the 180 degrees rotation is to be done). Again, the normalisation (N) would be 0 and division by 0 would ensue.

4.3.2 Finding all rotations for $\mathbf{C_n}$ symmetry

By converting all the symmetry peaks to the angle-axis representation, it becomes easy to group the peaks with similar axes of rotation. The algorithm needs to take into account two considerations; firstly, the axis can be present in either direction and yet these cases should be grouped together into a single group. And secondly, there may be slight numerical inaccuracies in the axis determination and therefore small error (denoted ϵ_0) needs to be accommodated when comparisons for axis similarity are made.

With both these issues cleared, each group of peaks with the same symmetry axis needs to have the possible symmetry fold n estimated. In the optimal case where only the n peaks with consecutive integer multiples of the angle $\frac{360}{n}$ are in a single group, the fold n could be determined as the average distance between the consecutive sorted group member angles. However, the situation may be complicated by several issues: a) some of the angles could be repeated (with small difference) multiple times. This situation arises with high enough resolution of the inverse SO(3) Fourier transform map, where two peaks could be separated by a single grid point. Both such peaks could have similar enough axis to be grouped together (within the small error ϵ_0) and similar angle (increasingly more similar with higher map resolution). And b), a false-positive peak passing the outlier detection could have similar enough axis to be grouped with other peaks just by chance, thus introducing a random angle into the group.

In order to detect any possible C_n symmetry while avoiding the two issues mentioned in the previous paragraph, the possible fold n of the group is sought by firstly finding all the differences between the angles in the groups and taking the smallest angle difference (η) within the symmetry axis group. Then, the smallest angle difference is checked for having the remainder of the division $^{360}/_{\eta}$ close to 0 or 1, again allowing for numerical inaccuracies by accommodating for small error ϵ_1. If such check is not passed, the next smallest angle in the list is considered and checked. This step is repeated until either a passing angle-difference-value η

is found or all distances have been tried. If all angle differences fail, it can be concluded that the symmetry axis group does not contain any C_n symmetry.

On the other hand, if any η passes, the C_n symmetry corresponding to the η value can be searched for. In order to do this, all consecutive integer multiples of the optimal angle given by $^{360}/_{n}$ degrees are computed in the range of -180 to 180 degrees and saved in a vector $\mathbf{V_n}$. Then, the presence of these angles in the group is sought and the longest continuous streak of matches between the group and the vector $\mathbf{V_n}$ is recorded. The comparison of the angles in $\mathbf{V_n}$ and the group does take into account the possible numerical inaccuracies by, again, accommodating for small error ϵ_1. Finally, if the longest continuous streak of matches is shorter than n, the angle difference η is removed from the list of distances between angles in the group and the next smallest value is tried; however, if the longest continuous streak of matches has the length of at least n, the C_n symmetry has been found.

This result follows the fact that the group has the same axis, which is the symmetry axis of the C_n symmetry, and rotation by any integer multiple of $^{360}/_{n}$ degrees along this axis produces a highly correlated structure (*i.e.* a peak with high height in the inverse SO(3) Fourier transform map); in other words, rotation by any integer multiple of $^{360}/_{n}$ degrees does not change the shape. This is the definition of a C_n symmetry. Furthermore, it is worth noting that this approach does identify the highest n symmetries first, as they will (by definition) have smaller angles and therefore the η value required for their detection will be smaller and thus attempted first. Nonetheless, this approach has two weaknesses which now need to be discussed in some detail.

Small η values and accuracy

The first issue that needs to be addressed is that while small values of n have distinct $^{360}/_{n}$ degree angles, which are easily separated from each other, for the large values of n, the difference can become rather small and when numerical inaccuracies are present, it may be difficult to distinguish which n should be associated

with what value of the smallest angle difference η. This can be shown on example of C_{15} symmetry, where $^{360}/_{15} = 24°$, but $^{360}/_{14} \approx 25.71°$ and $^{360}/_{16} = 22.5°$. In this case, the difference between the perfect symmetry angles is 1.5 degrees.

Given that the smallest angle difference η is used to estimate the appropriate value of n for the symmetry, it is easy to see that if the numerical inaccuracies would be as small as 0.75 degree, the wrong n could be estimated in this case. To avoid this issue, the angle error of misplacing the inverse SO(3) Fourier transform map peak by a single grid point is computed (equation 4.4) as well as the optimal angle difference (ϵ_A) between the initially assumed n value and the next $(n+1)$ value; *i.e.* $\epsilon_A = {}^{360}/_{n} - {}^{360}/_{(n+1)}$.

$$\epsilon_P = \frac{360}{2l_{max}} \tag{4.4}$$

where:

ϵ_P is the angle difference between two grid points on the inverse SO(3) Fourier transform map.

Consequently, if the ratio between the map grid point angle difference ϵ_P and the next symmetry angle difference ϵ_A is more or equal to 0.5, or in other words, if the map grid difference is less than double the next symmetry angle difference, the values of $n-1$, n and $n+1$ are all tested instead of just the value n. With this simple test implemented, one more possible source of error in the previously discussed algorithm for detection of the cyclic symmetries needs to be discussed.

Missing peaks

Another possible issue with the algorithm is that some proportion of the angles required for the symmetry (and by extension the peaks defining them) may be missing in the data. This can happen as a result of the interactions between the Euler angles not being linear; for example, if the β angle is 0.0, then the

angles α and γ are not independent as they are both rotations along the same axis (this assumes the ZYZ convention). Therefore, in this particular case, the three dimensional space is reduced to one dimensional space with the dimension being $\alpha + \gamma$.

Now, if the symmetry axis of a group is on the y axis of the co-ordinate system, then exactly this case arises and any particular peak given by such symmetry will be present in all co-ordinates where $\alpha + \gamma$ is equal to the peak angle. It is then easy to imagine that, especially for cyclic symmetries with large fold n, the number of peaks will be large and unless the inverse SO(3) Fourier transform map sampling is very fine, the peaks will merge together. In this case, the peak detection algorithm described above will not find some of the symmetry related peaks due to them having neighbouring peak heights with higher value than the symmetry-related peak position.

To accommodate for this issue, the cyclic symmetry detection algorithm was updated so that if for a particular value of η the number of $\mathbf{V_n}$ vector matches is at least 70% of n (the percentage can be changed when the software is used, it is just the default value), a search for missing peaks is initiated. Once initiated, the missing peak search takes the $\mathbf{V_n}$ vector values which were not matched in the same-symmetry-axis peaks group and searches for their existence in the inverse SO(3) Fourier map. This is done by searching the inverse SO(3) Fourier transform map for grid points which have co-ordinate Euler angles corresponding to the required group symmetry axis and the missing angle. The highest value among all the grid points conforming to these conditions is then compared to the original threshold and if larger, it is added to the same symmetry axis group. It is worth noting that this modification of the algorithm does not decrease the original outlier detection threshold for the peak distribution and therefore it only locates peaks which should have been detected originally.

4.4 Dihedral and polyhedral symmetries

With the ability to detect cyclic symmetries just discussed, the possibility of detecting other symmetries should be considered. In particular, the dihedral symmetries are the second type of point groups that can be detected using the inverse SO(3) Fourier transform map. Moreover, the polyhedral symmetry groups, which are the rotation groups of Platonic solids such as tetrahedron, octahedron and icosahedron should also be amenable to detection using the inverse SO(3) Fourier transform approach.

4.4.1 Dihedral symmetry detection

There are three different types of dihedral symmetry groups, the chiral dihedral symmetry, the achiral prismatic and achiral anti-prismatic dihedral symmetry. However, since both the achiral dihedral symmetry groups do require reflection as well as rotation, they will not be considered further as they cannot be detected using the inverse SO(3) Fourier transform method. Nonetheless, the chiral dihedral symmetries are point groups with a single cyclic symmetry C_n along one symmetry axis and another C_2 symmetry along a symmetry axis perpendicular to the first symmetry axis. They are denoted by D_n in the Schoenflies notation, where the n is the same number as for the C_n symmetry that forms the D_n symmetry, or simply by $n2$ in the Hermann-Mauguin notation.

Following from the definition of the chiral dihedral symmetry, a simple algorithm can be devised to detect them from the already available list of detected cyclic symmetries. Specifically, the list of the cyclic symmetries detected by the algorithm described in the section 4.3 can be searched for any pairs of symmetries with symmetry axes perpendicular to each other. The resulting list of symmetry pairs then, by definition, is the list of all detected chiral dihedral symmetries present in the structure. It should be noted that this approach will detect not only

the dihedral symmetry groups, but also the hexahedron (cube) symmetry, which is composed of three perpendicular C_4 symmetries.

4.4.2 Polyhedral symmetry detection

The polyhedral symmetries are point groups defined as combinations of multiple C_n symmetries which can be found in Platonic solids. Generally, there are five Platonic solids, each associated with its own polyhedral symmetry: tetrahedron, hexahedron (cube), octahedron, dodecahedron and icosahedron. Nonetheless, only three of these symmetries have been observed in the wwPDB database, namely the tetrahedral symmetry, the octahedral symmetry and the icosahedral symmetry. It is worth noting that there are wwPDB entries which state other symmetries - for example the entry 2WQT (Montgomery et al., 2010) named "*Dodecahedral assembly of MhpD*"; however, the wwPDB still designates this entry as icosahedral. Nonetheless, this project will only consider symmetry groups defined in the wwPDB at this stage.

Regarding the detection of the polyhedral symmetry, it is worth mentioning the Schlafli symbols, which describe regular polytopes. In three dimensions, the Schlafli symbols for Platonic solids are written in curly brackets and consist of two numbers, the first number signifies the number of edges that each face has, while the second number is the number of polygons that surround each vertex - for example, the octahedron has Schlafli symbol of $\{3, 4\}$. Regarding the quasi-regular polyhedra, their Schlafli symbols are written as $\left\{ \begin{smallmatrix} p \\ q \end{smallmatrix} \right\} = \left\{ \begin{smallmatrix} q \\ p \end{smallmatrix} \right\}$ where the p and q signify that the quasi-regular polyhedron contains faces from the regular polyhedron $\{p, q\}$ and the double regular polyhedron $\{q, p\}$. For more details on the notation, see for example Coxeter (1973).

The dihedral angle of a regular or quasi-regular polyhedra is the angle between any two adjoining faces of the polyhedra. Moreover, it can also be seen as the angle between the two lines which each connects the centroid (centre) of the polyhedron and the centre of one face of the polyhedron, assuming the two

polyhedron faces are adjoining. By realising that the regular and quasi-regular polyhedra have symmetry axes which go from the centroid to the centre of their faces, it can be seen that the dihedral angles are also the angles between these axes of symmetry, albeit other symmetry axes may not have these angles. Table 4.2 shows the dihedral angles and the Schlafli symbols for all five Platonic solids and the cuboctahedron.

Table 4.2: The dihedral angles for Platonic solids and selected quasi-regular polyhedra

Name	Schlafli symbols	Dihedral angle (radians)	Dihedral angle (degrees)
Tetrahedron	$\{3,3\}$	$\cos^{-1}(1/3)$	≈ 70.53
Hexahedron	$\{4,3\}$	$\pi/2$	90
Octahedron	$\{3,4\}$	$\pi - \cos^{-1}(1/3)$	≈ 109.47
Cuboctahedron	$\left\{\begin{smallmatrix}3\\4\end{smallmatrix}\right\}$	$\pi - \cos^{-1}(1/\sqrt{3})$	≈ 125.26
Dodecahedron	$\{5,3\}$	$\pi - \tan^{-1}(2)$	≈ 116.56
Icosahedron	$\{3,5\}$	$\pi - \cos^{-1}(\sqrt{5}/3)$	≈ 138.19

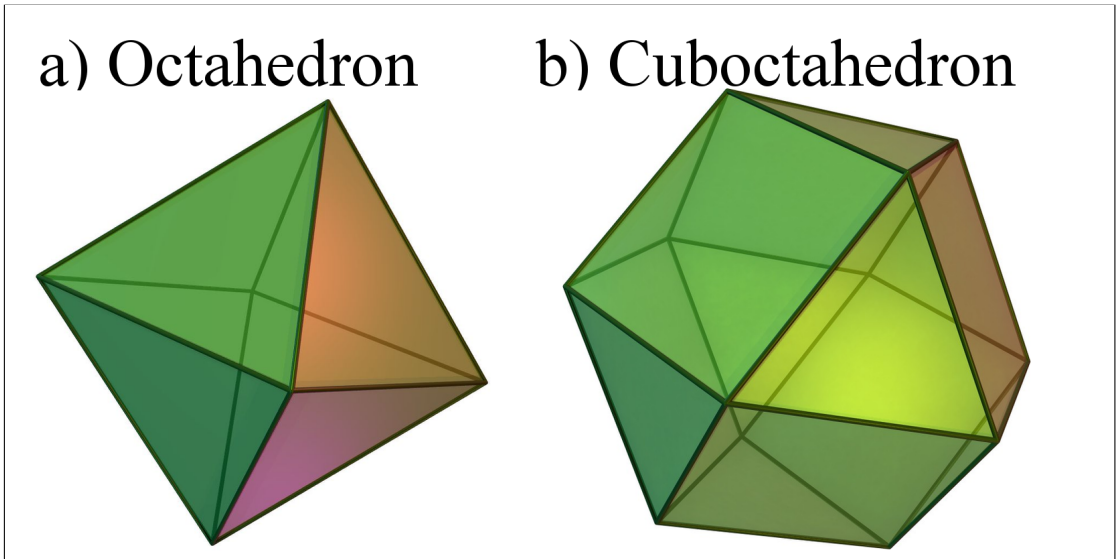
This table shows the Schlafli symbols and dihedral angles for the five Platonic solids and cuboctahedron. The table is adapted from Table 1 of Coxeter (1973).

The reason for including the cuboctahedron in table 4.2 along with the Platonic solids is that while it is a different shape than the octahedron, they do share the same symmetry group. This fact can be explained by exploring the conjugacy classes of polyhedral symmetries; the conjugacy classes are groups of group elements (in the case of symmetry groups, the symmetry elements - *i.e.* all rotations which do not change the shape of the object) which share similar features. Mathematically speaking, group elements a and b are conjugate if there is an element g for which it is true that $gag^{-1} = b$. In other words, if one element of the conjugacy class (a) can be changed into another element of the same conjugacy class (b), then the two elements are in the same class. It therefore follows, that if any two C_n elements of the same symmetry group with the same value for n exist, they must be in the same conjugacy class, as Euler's theorem guarantees the existence of a rotation g which rotates a to b.

Now, the polyhedral symmetry groups are defined by their conjugacy classes, but not by the angles between them. Therefore, since the octahedron and cuboctahedron do have the same composition of the conjugacy classes, they do have

the same symmetry group. However, since the dihedral angles do differ between the two shapes with octahedral symmetry group (see table 4.2), it is necessary to know for which of the two shapes the octahedral symmetry group is being searched for (at least when the inverse SO(3) Fourier transform approach is used). To this end, several of the wwPDB database entries with reported octahedral symmetry were obtained (see section 4.1) and the dihedral angles were detected by firstly using the aforementioned algorithm for detecting cyclic symmetries and then finding the angles between appropriate C_4 and C_3 symmetry axes. This approach has shown that the octahedral symmetry group in macromolecules has the cuboctahedral rather than the octahedral dihedral angles. To demonstrate the difference between the octahedron and cuboctahedron shapes, they are both visualised in figure 4.2.

Figure 4.2: The difference between the octahedron and cuboctahedron



This figure shows example of the Platonic solid octahedron and the quasi-regular polyhedra cuboctahedron. Octahedron picture was adapted from Wikipedia (2018b), while the Cuboctahedron picture was adapted from Wikipedia (2018a).

It then follows that if an object has any of the regular polyhedra symmetry group, the appropriate dihedral angles must be measurable between the symmetry axes of the C_n symmetries which are present for each face of the polyhedra. Consequently, this fact can be used to detect the tetrahedral, octahedral and icosahedral symmetries simply by testing for presence of two C_n symmetry axes of appropriate

n values and correct dihedral angle between them; specifically, two C_3 symmetry axes with angle $\approx$ 70.53 degrees for tetrahedral symmetry, one C_3 and one C_4 symmetry axes with angle $\approx$ 125.26 degrees for octahedral symmetry and one C_3 and one C_5 symmetry axes with angle between them of $\approx$ 138.19 degrees for icosahedral symmetry. For more details about polyhedra, see for example Coxeter (1973).

While detecting a symmetry is understood as finding evidence for the claim that the symmetry exists for a given structure, finding all the individual elements is a slightly different task. The algorithms discussed above are aimed to detect symmetry, that is to find the evidence for the symmetry existence; however, they are not sufficient to obtain all the elements of the polyhedral symmetry groups. Therefore, the algorithms introduced so far are sufficient to determine all the symmetry elements for the cyclic and dihedral symmetry groups, but only a detection of existence is done for the polyhedral symmetry groups.

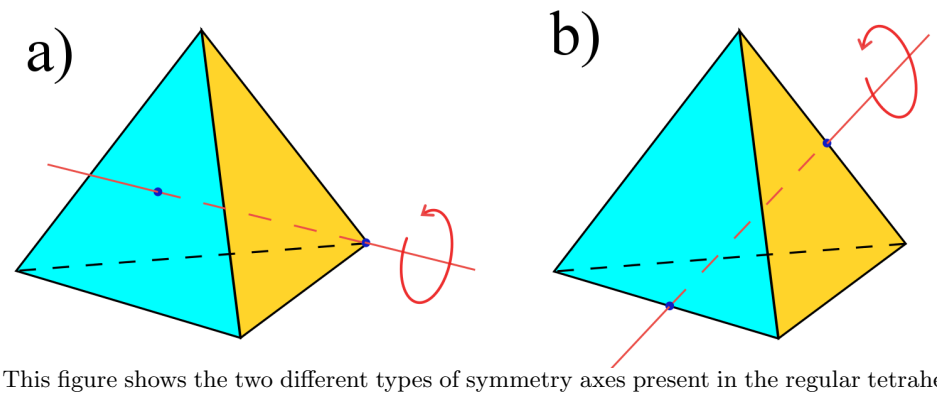
4.4.3 Polyhedral symmetry elements generation

Assuming that the previously discussed algorithm detects the presence of a polyhedral symmetry in the structure of interest, the list of all the symmetry elements could be obtained. In order to do so, the knowledge of conjugacy classes constituting the polyhedral symmetry group could be used, as the conjugacy classes of any polyhedral symmetry specify the list of all C_n symmetries which need to be found in order for the full polyhedral symmetry group to be described; in turn, each such symmetry axis defines a portion of the symmetry elements of the symmetry group it forms. Nonetheless, the conjugacy classes do not include the angles between the symmetry axes, which are required to fully determine the symmetry group elements. Therefore, the symmetry axes of the regular polyhedra specifying the symmetry need to be determined first.

Tetrahedral symmetry elements

The tetrahedron shape has 7 unique symmetry axes, which need to be identified in order to obtain all the 12 symmetry elements. The 7 axes are composed of 4 C_3 axes, each connecting a single vertex with the centre of the opposing face, as well as 3 C_2 axes, each connecting the centre of an edge with the centre of the opposing edge. For visualisation of these two symmetry axes types, see figure 4.3

Figure 4.3: The symmetry axes of regular tetrahedron



This figure shows the two different types of symmetry axes present in the regular tetrahedron. Part a) shows a C_3 symmetry axis connecting a vertex with the centre of the opposing face, while part b) shows a C_2 symmetry axis connecting the centre of an edge with the centre of an opposing edge. Figure adapted from Wikipedia (2007).

In order to locate these seven axes in the shape, the two already known C_3 symmetry axes with the dihedral angle of $\cos^{-1}(1/3)$ radians can be used as a start. An algorithm can be written to search the list of cyclic symmetries already available for the structure for the other two C_3 symmetries which do have the angle to each other and the already known two C_3 symmetries equal to the absolute value of the dihedral angle with some small error ϵ_1 being allowed for.

Subsequently, the three C_2 symmetries can be located by testing all C_2 symmetries already located in the shape for having the angle of $\pm \cos^{-1}(1/2)$ radians to all four C_3 symmetries, while being perpendicular to any already found C_2 symmetries (*i.e.* having the angle of $\pm \cos^{-1}(0)$ radians to any C_2 symmetry axis already associated with the shape).

Now, for the tetrahedral symmetry group, the set of conjugacy classes differs on whether the group is chiral or achiral; that is whether it includes reflections or not. Given that the discussed inverse SO(3) Fourier transform map approach cannot detect reflections, only the chiral tetrahedral symmetry group will be considered here. The list of conjugacy classes of the chiral tetrahedral symmetry group is shown in table 4.3 and it can easily be seen that each symmetry group element is associated with some of the already detected axes, except for the identity element.

Table 4.3: The conjugacy classes of the chiral tetrahedral symmetry group

Description	Symmetry element type	Number of occurrences
Identity	C_1	1
Rotation by $\pm 120°$ at each of the vertices	C_3	8
Rotation by $180°$ along the line connecting the centres of opposing edges	C_2	3

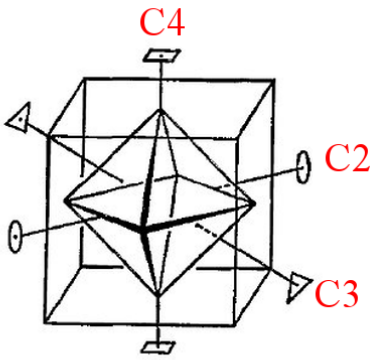
This table shows the conjugacy classes for the chiral tetrahedral symmetry group along with number of symmetry elements each of these classes has. Note that the number of occurrences of the symmetry elements is double the number of symmetry axes for axes with two angles (denoted by the $\pm$ symbol).

Therefore, the twelve chiral tetrahedral symmetry group elements can now be generated by combining the appropriate axis with angle, except for the identity element, where the axis is irrelevant as the angle is zero radians.

Octahedral symmetry elements

The octahedron and cuboctahedron shapes do have the same symmetry axes as discussed above, albeit the angles between the axes do differ. The list of axes that need to be determined in order to obtain all the octahedral symmetry conjugacy classes includes four C_3 symmetry axes, each of which connects two opposing vertices, three C_4 symmetry axes with each of these connecting two opposing centres of faces and also 6 C_2 symmetry axes, each of which connects the opposing centres of edges. For visualisation of where some examples of these axes are located in the regular octahedron, see figure 4.4.

Figure 4.4: The symmetry axes of regular octahedron



This figure shows the three different types of symmetry axes present in the regular octahedron. The C_3 symmetry axis connects two opposing vertices, the C_4 symmetry axis connects the centre of face with opposing centre of face, while the C_2 symmetry axis connects a centre of an edge with the opposing edge's of centre. The figure was adapted from International Union of Crystallography (1962).

Therefore, to locate the thirteen symmetry axes present in the octahedral symmetry group with angles between the axes given for the cuboctahedron shape, the algorithm needs to locate two C_4 symmetries perpendicular (*i.e.* with angle of $\cos^{-1}(0)$ radians) to the already detected C_4 symmetry axis and each other. Next, the algorithm searches the list of already known cyclic symmetries (produced by the symmetry detection algorithm discussed above) for four C_3 symmetries with absolute value angles to all the C_4 symmetry axes equal to the dihedral angle of cuboctahedron, that is $\pi - \cos^{-1}(1/\sqrt{3})$ radians. Finally, the algorithm needs to search for the six C_2 symmetries, four of which will have the angle of $\cos^{-1}(1/\sqrt{2})$ radians to each of the C_4 already found symmetry axes and two of which will have the angle of $\cos^{-1}(1/\sqrt{2})$ radians to two of the C_4 axes and angle of $\cos^{-1}(0)$ radians to the last C_4 symmetry axis.

With all the thirteen unique symmetry axes detected for the octahedral symmetry, the knowledge of the conjugacy classes can be used to determine the symmetry group elements for the octahedral symmetry group. Although there are two types of the octahedral symmetry, the chiral and achiral types, the achiral octahedral symmetry requires reflections as well as rotations and therefore cannot be detected by the inverse SO(3) Fourier transform based approach. Consequently, only the chiral version of the octahedral symmetry will be considered, similarly to

the dihedral and tetrahedral cases. The conjugacy classes of the chiral octahedral symmetry group are listed in the table 4.4.

Table 4.4: The conjugacy classes of the chiral octahedral symmetry group

Description	Symmetry element type	Number of occurrences
Identity	C_1	1
Rotation by 180° at each of the vertices	C_4	3
Rotation by ±90° at each of the vertices	C_4	6
Rotation by ±120° along the line connecting the centres of opposing faces	C_3	8
Rotation by 180° along the line connecting the centres of opposing edges	C_2	6

This table shows the conjugacy classes for the chiral octahedral symmetry group along with number of symmetry elements each of these classes has. Note that the number of occurrences of the symmetry elements is double the number of symmetry axes for axes with two angles (denoted by the ± symbol).

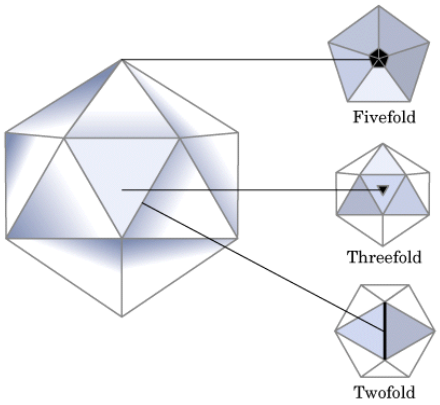
Therefore, the 24 chiral octahedral symmetry group elements can now be generated by combining the appropriate axis with angle, except for the identity element, where the axis is irrelevant as the angle is zero radians.

Icosahedral symmetry elements

The icosahedron shape does have 31 unique symmetry axes, distributed as follows: 6 C_5 symmetries with their axes always connecting two opposing vertices, 10 C_3 symmetries with their axes connecting the centres of two opposing faces and 15 C_2 symmetries with their axes connecting the centre of any edge with the opposing edge's centre. The positions of one example of each of these three axis types is shown in the figure 4.5.

To detect all of these 31 axes, the algorithm first takes the already known C_5 symmetry and searches the list of already detected cyclic symmetries in the structure. It attempts to locate five C_5 symmetries, whose axes have angle of $\pm\cos^{-1}(1/2)$ radians to each other and the already known C_5 symmetry. Next, the algorithm searches for the 10 C_3 symmetries by requiring any potential candidate C_3 symmetry to have the dihedral angle of $\pm\cos^{-1}(\sqrt{5}/3)$ radians to the three closer

Figure 4.5: The symmetry axes of a regular icosahedron



This figure shows the three different types of symmetry axes present in the regular icosahedron. The top inset shows the view from a C_5 symmetry axis, the middle inset shows the location and view from the C_3 symmetry axis and the bottom inset shows the view for a single C_2 axis. Figure adapted from Ziegler (2003).

C_5 symmetry axes and the angle of $\pm\cos^{-1}(1.0 - \sqrt{5}/3)$ radians to the three further C_5 symmetry axes. Finally, to detect the 15 C_2 symmetries, the algorithm requires all candidate C_2 symmetry axes to be perpendicular to two of the C_5 axes (*i.e.* have the angle of $\cos^{-1}(0)$ radians to them), to have the angle of $\pm\cos^{-1}(1/2)$ radians to another two C_5 symmetry axes and to have the angle of $\pm\cos^{-1}(\sqrt{3}/2)$ radians to the last two C_5 symmetry axes.

With the 31 unique axes now determined, the conjugacy classes of the icosahedral symmetry group can be used to obtain the symmetry elements of the symmetry group. As with the tetrahedral and octahedral symmetry groups, there are two icosahedral symmetry groups, the chiral and achiral one; again, as the achiral icosahedral symmetry groups requires reflections, it cannot be considered here. Table 4.5 shows the conjugacy classes for the chiral icosahedral symmetry group.

Therefore, the 60 chiral icosahedral symmetry group elements can now be generated by combining the appropriate axis with angle, except for the identity element, where the axis is irrelevant as the angle is zero radians.

Missing symmetry axes

The approach described above for determining the symmetry elements of the polyhedral symmetry groups has one issue, which should be discussed before proceeding further. As mentioned in the beginning of section 4.3.2, the C_n symmetry searching

Table 4.5: The conjugacy classes of the chiral icosahedral symmetry group

Description	Symmetry element type	Number of occurrences
Identity	C_1	1
Rotation by $\pm 72°$ at each of the vertices	C_5	12
Rotation by $\pm 144°$ at each of the vertices	C_5	12
Rotation by $\pm 120°$ along the line connecting the centres of opposing faces	C_3	20
Rotation by $180°$ along the line connecting the centres of opposing edges	C_2	15

This table shows the conjugacy classes for the chiral icosahedral symmetry group along with number of symmetry elements each of these classes has. Note that the number of occurrences of the symmetry elements is double the number of symmetry axes for axes with two angles (denoted by the $\pm$ symbol).

algorithm starts by grouping all the detected similarity peaks according to their axis of symmetry and making sure two peaks with axes in opposite directions are grouped together by switching the direction of one of the axes. The result of this operation is simpler cyclic symmetry detection, but also a possibility of not finding an axis required by one of the polyhedral symmetries, as the axis with opposite direction could have been used instead. Consequently, the axis with the opposite direction, while being identical to the required axis, may not have the same angles to the other axes as required by the angle tests discussed above.

To remedy this situation, an algorithm was developed to search for a C_n symmetry with given axis (or its opposite) in the inverse SO(3) Fourier transform map. This is done by firstly obtaining a list of all map values which produce the required axis, sorting these by the angle they represent and then finding the combination of n angles separated by distance of $^{360}/_{n}$ (accommodating for small error ϵ_1) which has the largest average peak height. If this average peak height is larger than the minimum peak threshold, the C_n symmetry with the required axis is considered to exist.

With this new ability to search for symmetries which should be present in the shape, but are not either for the aforementioned reason or any other, the algorithms for detecting the tetrahedral, octahedral and icosahedral chiral symmetry

groups were updated to also search for missing axes whenever required. Therefore, the symmetry detection algorithm can now detect cyclic, dihedral, tetrahedral, octahedral and icosahedral symmetries as well as their respective symmetry elements. Finally, it is worth noting that the algorithm as well as the software tool available for applying it has recently been published by the author in Nicholls et al. (2018).

Chapter 5

Clustering of BALBES protein domain database

The chapter 3 concluded with defining three shape similarity measures and discussion of their implementation. It is now interesting to consider how these shape descriptors can be used to reduce the shape redundancy of any set of shapes and particularly a protein domain database. The purpose of doing so is twofold, firstly it allows a large scale testing of the shape descriptors and possible detection of any issues with the implementation or parametrisation of the shape descriptors. And secondly, by reducing the size of the BALBES (Long et al., 2008) protein domain database, any future usage of the database will not be as computationally expensive.

In order to reduce the shape redundancies of the BALBES protein domain database, such redundancies needs to be detected. This task can be seen as finding similar objects given their distance, or in other words, as a clustering problem. More generally, the task of finding clusters in any dimensional data has been studied for some time now and there are many available algorithms to perform it. These algorithms differ on the definition of a cluster and the consequent approach to comparing two clusters, as well as the "strictness" of the clustering, that is, whether a single object can belong to only one cluster, or whether it can belong

to multiple or no cluster at all. Table 5.1 provides an overview of the common clustering algorithms as well as their brief description and references for more information.

Table 5.1: Overview of commonly used clustering algorithms

Name	Cluster definition and comparison approach	Reference
Hierarchical clustering	Uses connectivity models (*e.g. dendrograms*) to detect clusters. It assumes numerical distance measure and may be dependent on the selected method for obtaining distances between clusters.	Rokach and Maimon (2005)
Equivalent classes	Two set members are classified together if they both conform to a predefined equivalence relation - for example have distance more than x.	Avelsgaard (1989)
k-means clustering	Using k centroids, each centroid being the average of the closest objects to it, the algorithm iteratively moves the centroids to minimise the square distance between the centroid and the objects associated to it.	MacQueen (1967)
Maximum-likelihood clustering	Starting with a random model, the likelihood of the model is computed and maximised using the real data. These steps are repeated until convergence.	Dempster et al. (1977)
Density-based clustering	Using the number of neighbours as a measure of compactness, these approaches assign regions with high compactness as clusters and all other regions as outliers.	Ester et al. (1996)
Graph-based clustering	By firstly computing the graph connecting the objects, the algorithm then searches for highly connected subgraphs, assigning these as clusters.	Hartuv and Shamir (2000)
Self-organising maps	This approach is based on artificial neural networks, but instead of using the back-propagation as a learning method, an evolutionary algorithm combining values of high-scoring nodes and their neighbours is used, thus growing better networks.	Kohonen (1982)

This table shows the common clustering algorithms as well as brief description as to how they achieve clustering of a given set of objects. References are given for sources of more information about each of the mentioned algorithms.

When considering the approaches listed in table 5.1 with regards to the intention of clustering the BALBES protein domain database, it seems reasonable to start by considering the nature of the features by which the clustering is to be done. More specifically, the shape descriptors described in chapter 3 should be used as the basis for clustering. However, given that the BALBES database contains $13,719$ protein domains, computing the descriptor values for each unique pair

would require $\frac{13{,}719 \times 13{,}718}{2} = 94,098,621$ distance computations. As the computational cost of computing distances between all unique pairs of domains is rather high, methods for reducing the number of distances need to be considered.

5.1 Reducing number of distances

In order to decrease the computational cost of obtaining the distances between protein domains of the BALBES database, reduction of the number of comparisons is required. In other words, by applying rules specifying which protein domain pairs need to have their distances determined and which pairs can be safely ignored, the total number of distances to be computed will be reduced and thus the computational cost of the distances calculation will decrease proportionally.

5.1.1 Pre-filtering using domain sizes

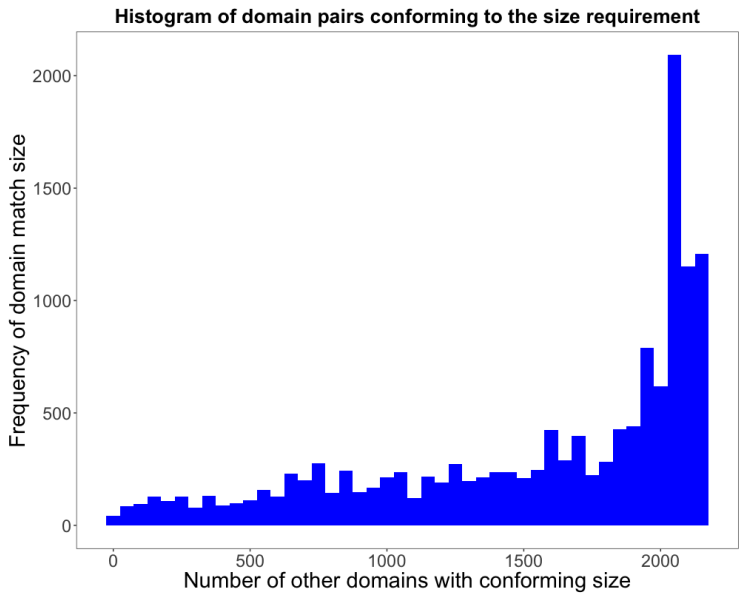
The first pre-filtering approach to be considered is using the size information of the protein domains. The size information is readily available in the form of the number of atoms, while other related measures, such as volume or maximum distances from centre of density could be easily obtained. Given that the intention of clustering the BALBES database protein-domains is to reduce redundancy in shape, it follows that only protein domains with similar number of atoms should be clustered together; although, there needs be some allowance for minor differences in otherwise similar shapes.

An example of similar shapes with minor differences would be protein domains with differently sized loops connecting the secondary-structure elements, or possibly protein domains with identical secondary-structure elements positions, but slightly different lengths of some of the secondary-structure elements. On the other hand, if a pair of protein domains has considerably different sizes, even in the case that the smaller protein domain is a subset of the larger domain, they

should not be clustered together, as this would result in loss of information and not simply removal of redundancies.

Therefore, by selecting a relatively conservative requirement that the smaller domain of the pair should have at least 90% of the number of atoms of the larger protein domain, the number of comparisons could be reduced. Specifically, by finding the number of atoms present in each domain and then, for each domain finding all other domains conforming to the size requirement, the number of pairs for which the distances need to be computed can be reduced to $21,340,790$ or approximately 22.7 % of the original number of distances. Moreover, no domain has more than $2,200$ other domains conforming to the size requirement and $2,840$ domains have less than $1,000$ other domains conforming to the size requirement. The histogram of frequencies of the number of other domains conforming to the size requirement is shown in figure 5.1.

Figure 5.1: The histogram of the number of other domains passing the size requirement



This figure shows the histogram obtained by finding the number of domains which conform to the 90% size similarity requirement in a pair with another domain, for each domain. The histogram was produced using the *bin width* parameter value of 50. The average number of protein domains passing the 90% size similarity requirement is $1,555$.

5.1.2 Hierarchical distance computation

While the protein-domain size-similarity requirement did decrease the number of required pair distance calculations considerably, there is another option which

could be explored. By considering that there are three different shape distances that are to be computed for each pair of structures passing the size requirement just discussed and, furthermore, noting that the three distance measures have different execution times for the same settings (see tables 3.5, 3.6 and 3.7), it is possible to use the fastest descriptor value (the cross-correlation shape descriptor) as a pre-filter deciding whether the rest of the shape descriptors need to be computed or not. Similarly, if the second-fastest shape-distance is to be computed (the trace-sigma descriptor), its value could be used to decide whether the rotation-function shape-distance needs to be computed.

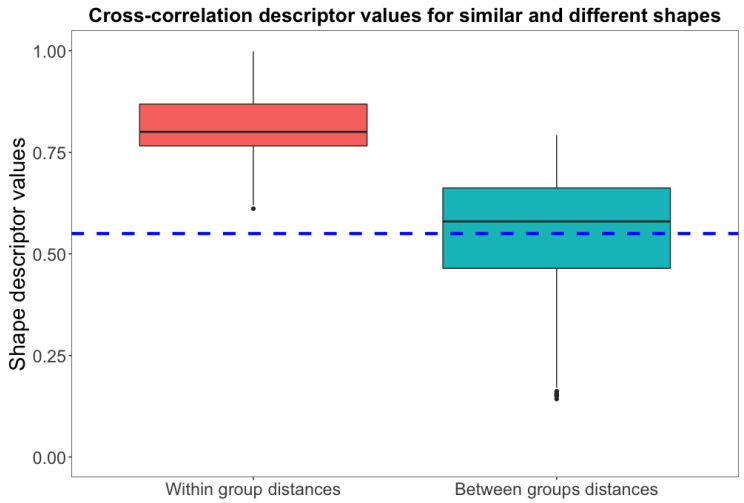
By using the distances computation in this hierarchical way, the time required for each pair distance calculation could be reduced considerably; for example, it can be seen from the tables 3.5, 3.6 and 3.7 that each pair not passing the cross-correlation distance threshold would have its computation time decreased approximately by $1/2$, assuming the default values of resolution 8.0 Å and B-factor value of 80. Nonetheless, to estimate to how many of the distances computations this speed-up would apply, the threshold for the cross-correlation-descriptor distances as well as for the trace-sigma-descriptor distances need to be determined.

Hierarchical distance thresholds

Clearly, the thresholds for the hierarchical distance computation will be dependent on the resolution and B-factors used; however, since the default parameters for the distance computation algorithms were determined in section 3.5.5, it seems logical to start with these values. Now, by observing the distributions of the cross-correlation descriptor with the default settings for similarly shaped and differently shaped pairs of structures in the test dataset, a threshold could be decided. This threshold will, inevitably, be arbitrary as there is no perfect way to determine it. Nonetheless, by requiring the threshold to have minimum (optimally zero) false positives and being conservative about placing such a threshold, that is, not placing the threshold just below the lowest measured distance between similar

structures, but leaving some extra space for any similar structures which may be found later to have yet smaller distance. As shown in figure 5.2, a threshold value conforming to the conservative requirement would be 0.55, albeit it is an arbitrary value and there is no particular reason why slightly different value could not have been chosen.

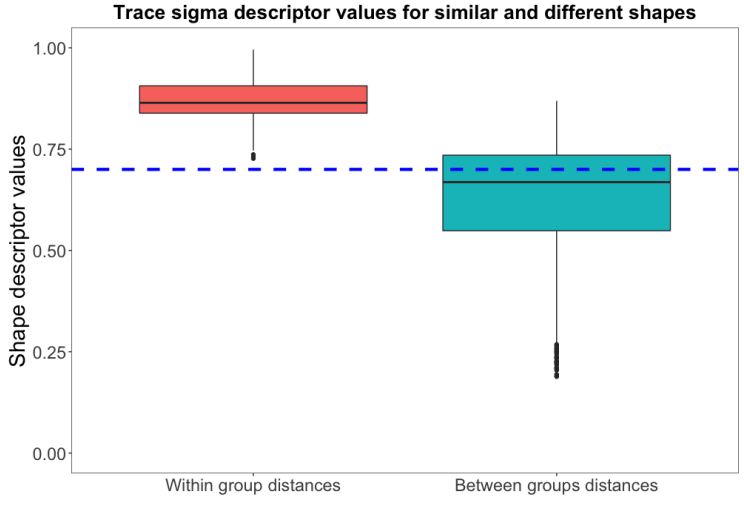
Figure 5.2: The distributions of the cross-correlation descriptor distances for similarly shaped and differently shaped pairs of structures using the default distance computation settings



This figure shows the box plot of distributions of the cross-correlation shape descriptor distances for pairs of similar shapes (in red) and of different shapes (in turquoise) as defined in the test set (see section 3.1). The blue dashed line shows how the threshold value 0.55 cuts both of the distributions.

The threshold for the trace sigma descriptor can be selected using the same criteria and as shown in figure 5.3, one such threshold value is 0.70.

Figure 5.3: The distributions of the trace-sigma descriptor distances for similarly shaped and differently shaped pairs of structures using the default distance computation settings



This figure shows the box plot of distributions of the trace sigma shape descriptor distances for pairs of similar shapes (in red) and of different shapes (in turquoise) as defined in the test set (see section 3.1). The blue dashed line shows how the threshold value 0.70 cuts both of the distributions.

Finally, the idea of hierarchical shape distance computation introduced here was implemented for the previously discussed shape descriptors calculation routine and with this change, it became possible to compute the distances between pairs of protein domains in the BALBES database which passed the size criteria.

5.2 BALBES protein domain clustering approach

With the distances between BALBES protein domains now computed for all pairs of domains which conform to the size criteria and with only having all three descriptor distances for the pairs which conform to the two conservative thresholds, decision on clustering algorithm needs to be made. From the algorithms listed in table 5.1, several can be excluded by not being optimal given the input-data type.

5.2.1 Selection of clustering algorithm

The self-organising maps algorithm requires knowing the targets (*i.e.* the correct answers) in order to optimise the neural network best suited to cluster the inputs. While it would be possible to train a self-organising map on the test dataset used previously, there is no guarantee that the test dataset contains sufficient amount of data to produce a reliable self-organising map and therefore it seems that the self-organising maps algorithm is not easily applicable to the clustering problem addressed here.

Moreover, the maximum-likelihood clustering algorithm cannot easily be applied to the distance data, as it requires the knowledge of the number of clusters that the data contain and this information is not available at the start of the clustering. The algorithm could be applied row-wise to the sparse distance-matrix (that is, to each domain to see which other domains cluster with it) with searching for two clusters, but this would lead to various cluster thresholds for various domains and therefore to non-optimal clustering. Similarly, the k-mean clustering method requires the number of clusters to be known in advance and since this is

not the case for the clustering of protein domains, the method cannot be easily applied.

Regarding the graph-based clustering methods, it is non-trivial to build a graph from a sparse distance-matrix, although it would be possible to build a graph from the complete distance-matrix. Nonetheless, it seems worth exploring the alternative clustering algorithms before deciding to compute all distances in order to build the graph. Similarly, the density based clustering approaches would require converting the distances to a n-dimensional space where each domain has a given co-ordinate position and therefore would also require computing the complete instead of the sparse distance-matrix.

The hierarchical clustering algorithm can be used on sparse matrices, however, it is then non-trivial to decide how the dendrogram should be "cut" in order to obtain clusters. Moreover, it is non-trivial to combine the three different shape-distance values produced by the three shape descriptors in a single hierarchical clustering algorithm, while building three different dendrograms would then require a method for joining the results into a single clustering result, thus raising another host of issues.

Finally, the equivalent-classes clustering approach is a rather general approach, as any equivalence relation can be used for clustering and therefore combining the three results can be done directly by defining an equivalence relation using all three descriptors at once. Furthermore, the equivalence-classes clustering approach does support sparse distance-matrices and does not require deconstructing the distance matrices to co-ordinates. Therefore, the equivalent-classes clustering appears as the most applicable method for the intended purpose of clustering the BALBES protein domain database.

5.2.2 Deciding thresholds

With the clustering algorithm decided, the first step in applying the equivalent classes algorithm is to determine the equivalence relation, that is, the rule by

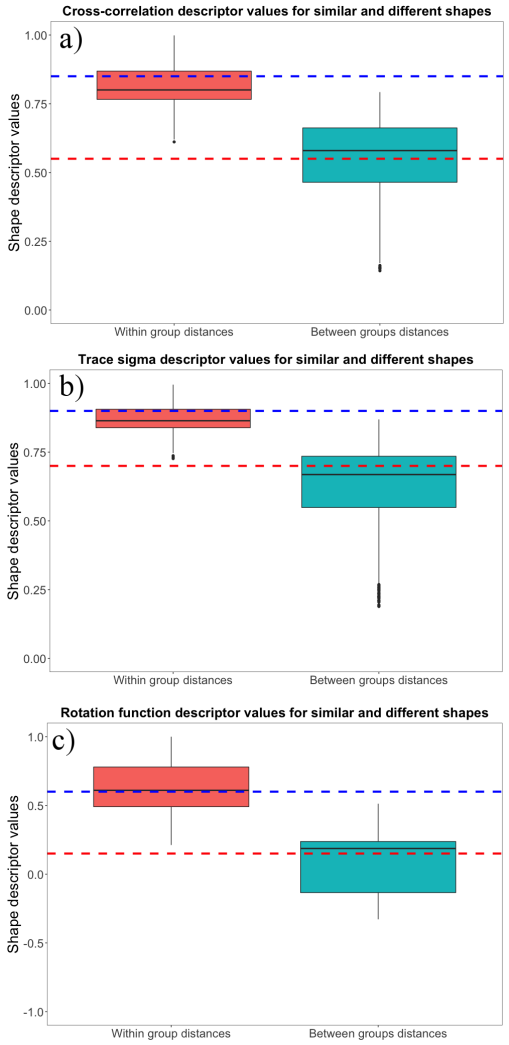
which the algorithm decides whether two protein domains should be clustered together or not. Given that for each pair of domains passing the size requirement, there may be distances available for one, two or all three shape descriptors, the decision rule should include this information. Moreover, the algorithm should be based on some thresholds for the three descriptors, presumably universal to all the domain pairs.

Furthermore, it should be noted that the thresholds decided previously for the cross-correlation descriptor and the trace-sigma descriptor were conservative threshold reducing the number of false negatives (*i.e.* pairs which are similar, but the threshold judges them different); however, for the clustering outcome, the expected behaviour would be exactly opposite, that is the algorithm should conservatively reduce the number of false positives (*i.e.* domain pairs which are different, but the algorithm clusters them together). This requirement comes from the fact that the relative effect of false positives and false negatives is not identical, as not clustering two similar shapes together will only keep the redundancy in the database, while clustering together different shapes will cause issues when the clustered database is used in any further applications.

Therefore, new thresholds need to be decided for each of the shape descriptors; these new thresholds should conservatively reduce the number of false positives as just discussed. To this end, it is still the case that there is no absolute way to determine these thresholds and therefore they will necessarily be somewhat arbitrary. Nonetheless, the thresholds for the clustering were decided using the results from the test set (as discussed in section 3.1) to be 0.85 for the cross-correlation descriptor, 0.9 for the trace-sigma descriptor and 0.6 for the rotation-function-based descriptor. The placement of all of the three thresholds relative to the similar and different-shape-distance distributions is shown in figure 5.4. It is worth noting that the figure also includes the false-negatives reducing thresholds determined previously for the cross-correlation and trace-sigma descriptors as well as newly

determined for the rotation-function-based descriptor. The reason for including these thresholds will become apparent in the following section.

Figure 5.4: The false-positive and false-negative reducing thresholds for the cross-correlation, trace-sigma and rotation-function-based descriptors computed with default settings.



This figure shows the distributions of distances between the similar-shape pairs (in red) and different-shape pairs (in turquoise) of the BALBES protein domains as determined in the test dataset for the cross-correlation descriptor in part a), for the trace-sigma descriptor in part b) and for the rotation-function-based descriptor in part c). The blue dashed lines show how the conservative false-positives reducing threshold relates to these two distributions for each descriptor (the values are 0.85 for the cross-correlation descriptor, 0.9 for the trace-sigma descriptor and 0.6 for the rotation-function-based descriptor), while the red dashed lines show how the conservative false-negatives reducing threshold relates to the two distributions (the values are 0.55 for the cross-correlation descriptor, 0.7 for the trace-sigma descriptor and 0.15 for the rotation-function-based descriptor).

5.2.3 Equivalence relation

As the decision thresholds have been determined, the equivalence relation can now be addressed. The equivalence relation is the basis of the equivalence-classes clustering as this is the rule which is used to decide whether any pair of the clustered objects is to be placed in the same cluster or not. In the case of clustering

BALBES protein domain database, there are several considerations to be discussed before deriving the equivalence relation rule.

The first requirement is that the equivalence relation should reduce the number of false positives, optimally to zero. This is the reason why the conservative false-positive reducing thresholds (shown in blue dashed lines in figure 5.4) should be used. The second requirement is that the equivalence relation should determine how the three descriptor distances should be combined to decide whether to cluster the two domains together or not.

Regarding the second requirement, if there was a single descriptor distance, the conservative false-positives-reducing threshold would be sufficient to formulate the equivalence relation and in the simple case where the pair of domains surpasses the false positives reducing threshold for all three descriptor distances, this remains true. However, in the cases where the domain-pair distances surpass zero, one or two thresholds, but not all three thresholds, there needs to be a clear rule to decide whether to cluster the two domains together or not. Considering that the false-positives are more problematic than the false-negatives, it seems that requiring at least two of the descriptors to be higher than their respective false-positives-reducing thresholds is appropriate.

Considering the case where a domain pair has two descriptor distances higher than the false-positives-reducing threshold and the third descriptor distance does not, it seems that this third descriptor distance should at least be higher than the false-negatives-reducing threshold. The reason for this additional rule is that if a descriptor determines that two domains are different, this information should be considered as much as if the descriptor determined that they are similar. It is akin to the reasoning behind the hierarchical descriptor distance computation approach discussed in section 5.1.2.

Finally, the equivalence relation can be stated as follows: *Two domains are clustered together if at least two of their shape descriptor distances surpass their respective conservative false-positives-reducing thresholds, while the third descriptor*

distance is greater than its respective conservative false-negatives-reducing threshold.

5.2.4 Equivalence classes clustering

Finally, with the equivalence relation determined, the equivalence-classes clustering algorithm can be applied to the computed distances. In order to do so, a simple algorithm was implemented to take iteratively each row of the sparse distance matrix and apply the equivalence rule to each of the matrix entries. Recalling that each row of the sparse matrix are the distances from a single domain to all other domains, this means finding all other domains which should be clustered together with the domain of the particular row according to the equivalence rule.

Consequently, if no other domains passing the equivalence rule are found, the algorithm places the row domain as a singleton (cluster with a single entry) into a list of resulting clusters. However, if there are domains which cluster with the row domain, the algorithm will firstly check the list of already found clusters for containing any of the domains now clustered to the row domain, or the row domain itself. For ease of expression, the domains clustering with the row domain itself together with the row domain will be called *prospective cluster*.

There are three possible outcomes of comparing a prospective cluster to the list of already found clusters: a) no match between the prospective cluster and the list of already found clusters is found and in this case, a new entry in the list of resulting clusters is created to contain the prospective cluster; b) one or multiple matches are found to a single entry in the list of already found domains. In this case, all entries of the prospective cluster are checked by the equivalence relation against each entry of the matching cluster. If all matches pass the equivalence relation, the prospective cluster is added to the existing cluster, while otherwise the prospective cluster is added to the list of resulting clusters as a new entry, creating duplication of the entries which matched between the two clusters. And c), if the prospective cluster matches to multiple different clusters in the list of

resulting clusters. In this case, the solution to outcome b) is applied iteratively to all the matching clusters, that is, if the whole prospective cluster passes the equivalence relation against all members of any of the already found clusters, the clusters are joined together, if no such joining is done, the prospective cluster is added to the list of existing clusters as a new entry, creating a duplication of any matches.

Regarding the duplicate entries in the resulting list of clusters, these are protein domains which pass the equivalence relation to at least two different other domains, but where these two different other domains do not pass the equivalence relation between themselves. This situation can be demonstrated on the following, hypothetical, example: Assuming that there are two different states (A and B) of the same domain in two different physiological conditions, each represented as a different structure in the BALBES protein domain database. Furthermore, assuming that there is a domain representing the intermediate state (I) of the transition from state A to state B. Given this, the equivalent-classes algorithm described here could find domain I clustered with domain A and domain B, but not find domains A and B clustered together. Therefore, in order not to cluster together structures A and B, which could be very different, the algorithm rather allows for some duplication.

5.3 BALBES clustering results

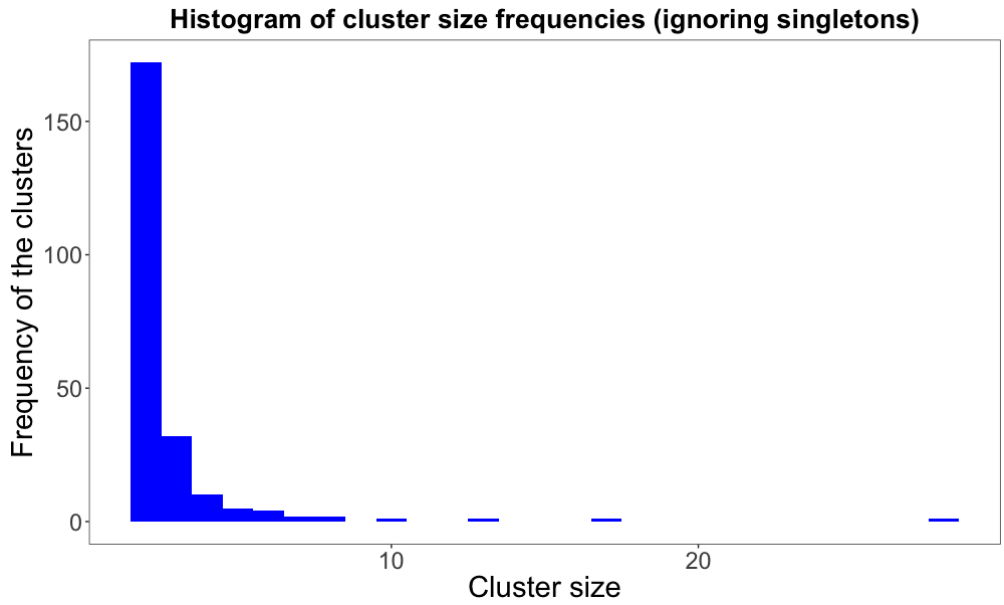
Finally, by firstly computing the distances between BALBES protein domains and then applying the equivalent-classes clustering algorithm, a list of protein domain clusters can be obtained.

5.3.1 First iteration

As mentioned previously in section 5.1.2, the distances between the BALBES protein domain database entries were computed using the default parameters of

resolution 8.0 Å and B-factors 80. The resulting list of clusters contains 13,392 clusters, with the cluster size frequencies as shown in figure 5.5. Moreover, figure 5.6 shows a visualisation of the overlay of the domains clustered together by the algorithm for selected clusters; the reason for visualising these overlays is to allow visually checking that the clustered domains do actually have similar shape.

Figure 5.5: The frequencies of the cluster sizes after first iteration of clustering

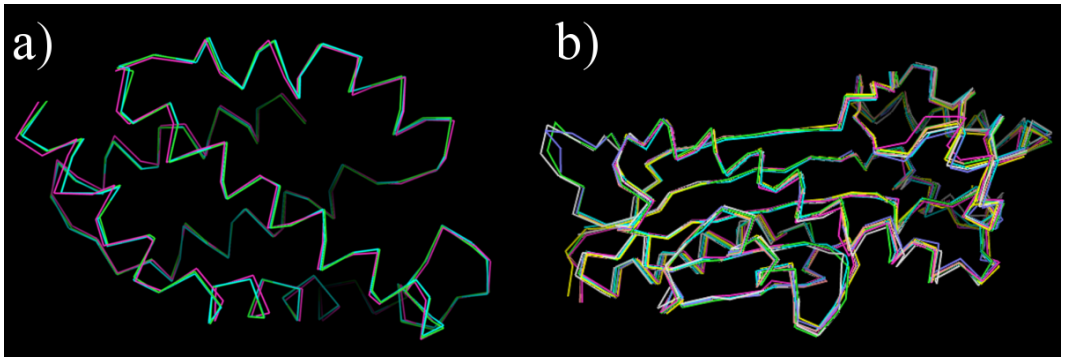


This figure shows the frequencies of clusters with given cluster size in the list of clusters resulting from the equivalent classes clustering of the BALBES protein domain database. Note that the singletons (clusters containing a single domain) are not shown.

Given that the new clusters contain domains with similar shape, by representing each cluster with a single domain, the BALBES database could be reduced to contain only 13,392 domain, leading to a reduction of ≈ 2.4 %. This is not a large difference, nonetheless, it is a decrease in the database size and therefore in the number of calculations required when any structure is being compared against the whole database.

One feasible explanation for why the reduction in BALBES database was not larger is that the resolution settings used for computing the distances between the protein domains did not provide sufficient distinguishing power and consequently, when conservative thresholds were used to cluster the domain using the domain distances, only the clearest cases could be recognised as similar using the distances.

Figure 5.6: Overlay of domains clustered together for selected clusters



This figure shows two different clusters resulting from the equivalent classes clustering of the BALBES protein domain database. The left cluster a) consists of three domains (3O18_A_dom_1, 1CPC_A_dom_1 and 3KVS_A_dom_1), while the right cluster consists of seven domains (1Z1E_A_dom_1, 3A5Q_A_dom_1, 2D3M_A_dom_1, 3AWJ_A_dom_1, 1BI5_A_dom_1, 3OV2_A_dom_1 and 1EE0_A_dom_1). For all structures, only the traces of their backbone atoms are shown. The domains were aligned using `ProSMART` (Nicholls et al., 2014) and visualised using `PyMoL` (Schrodinger LLC, 2015).

Clearly, there are other possible explanations, such as the descriptors not being powerful enough in general or the BALBES domain database not having any more shape redundancies; however, the explanation suggested here can be easily tested by increasing the resolution settings and computing new distances.

Nonetheless, before the change in resolution is attempted, the current progress should be considered. More specifically, computing the distances for the reduced version of the BALBES database does not only reduce the computational cost, but also allows iteratively reducing the redundancy of the database by increasing the resolution of the descriptors. Therefore, if all clusters can be represented by a single domain, then the iterative increase in resolution can be used to improve the redundancy removal.

Typically, clusters of objects in clustering scenarios are represented by the average of the clusters (for example k-means clustering or average linkage hierarchical clustering approach). Nonetheless, in the case of protein domains (and any molecules for that matter), taking an average is not really meaningful, as the average of two chemical states would likely be unstable and possibly not even valid in the sense of chemistry. Therefore, the clusters of protein domains will be repre-

sented by a single domain which has the highest average similarity (as measured by the shape descriptors) to the rest of the domains in the cluster. With this decision made, a simple script can be written to convert the list of clusters onto a list of protein domains, where singleton clusters are represented by the only domain present in the cluster, clusters with two domains are represented by randomly selected one of the two domains, as they have the same distance to each other and any larger clusters are represented by the domain with highest average similarity to the rest. The resulting list of protein domains can now be used instead of the complete BALBES protein domain database entries list for the next iteration of domain clustering.

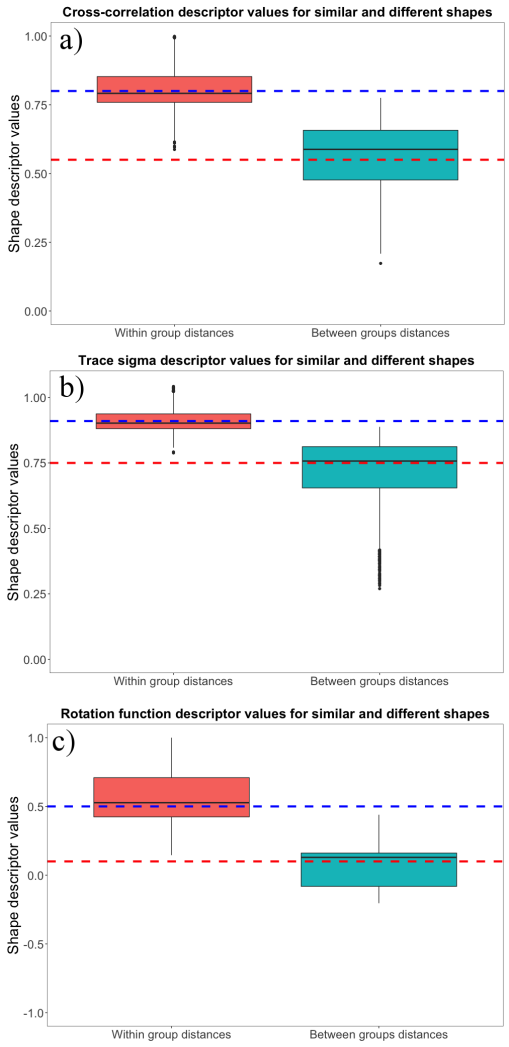
5.3.2 Second iteration

The first step of the second iteration of the clustering algorithm is to decide the values for resolution and B-factors to be used for computing the new distances between the reduced set of BALBES protein domains database. Given that the first iteration of the clustering algorithm used the default parameters, the second iteration could use higher resolution in an attempt to improve the clustering accuracy and therefore to cluster more similar shape structures together. Consequently, by consulting the tables 3.5, 3.6 and 3.7, it is clear that by increasing the resolution to 6.0 Å and changing the B-factor values to 40, the accuracy (as measured by the AUROC measure on the test dataset) can be slightly increased for the trace sigma and rotation function descriptors, albeit at the expense of almost tripling the computation time. Therefore, these values will be used to compute the new distances between the reduced BALBES domain database entries.

It should be noted that the same size requirements and hierarchical distance computation approaches should be used to reduce the computation cost. Therefore, new set of thresholds for the conservative false-negatives reduction will be required to be used in the hierarchical distance computation. Since the clustering using these new distances will be done using the same algorithm as before, which

will require the conservative false-positives-reduction threshold, figure 5.7 shows both of these threshold in terms of their relation to the distributions of the similar shape pairs and different shape pairs as defined in the test dataset and computed using the resolution of 6.0 Å and B-factor value 40.

Figure 5.7: The false-positives and false-negatives-reducing thresholds for the cross-correlation, trace-sigma and rotation-function-based descriptors for the second iteration of the clustering algorithm



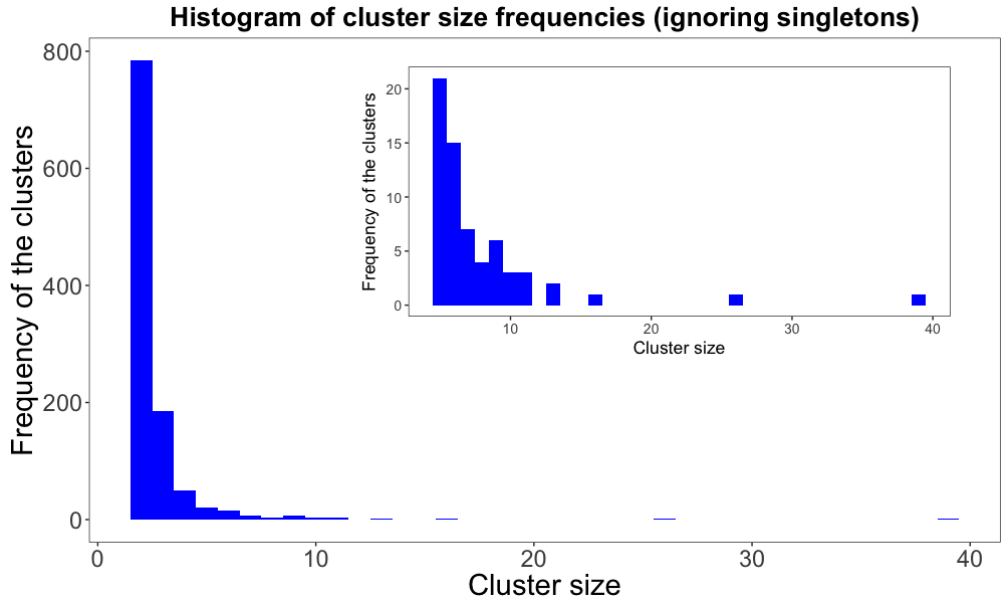
This figure shows the distributions of distances between the similar-shape pairs (in red) and different-shape pairs (in turquoise) of the test dataset of protein-domains computed using the resolution of 6.0 Å and B-factor of 40. The cross-correlation descriptor is shown in part a), the trace-sigma descriptor is shown in part b) and the rotation-function-based descriptor is shown in part c). The blue dashed lines show how the conservative false-positives-reducing threshold relates to these two distributions for each descriptor (the values are 0.8 for the cross-correlation descriptor, 0.9 for the trace-sigma descriptor and 0.5 for the rotation-function-based descriptor), while the red dashed lines show how the conservative false-negatives-reducing threshold relates to the two distributions (the values are 0.55 for the cross-correlation descriptor, 0.75 for the trace-sigma descriptor and 0.1 for the rotation-function-based descriptor).

With the thresholds for hierarchical distances computation determined as 0.55 for the cross-correlation descriptor distance, 0.75 for the trace-sigma descriptor distance and 0.1 for the rotation-function-based descriptor distance, the sparse matrix was computed. Consequently, the same equivalent classes clustering algorithm with the same definition of the equivalence relation as discussed in section

5.2.3, except now using the new thresholds and distances, was used to cluster the reduced BALBES database.

The resulting list of clusters contains 11,868 clusters, thus providing reduction of $\approx 11.4\%$ from the results of the first iteration of the algorithm (or reduction of $\approx$ 13.5 % of the original BALBES database). The distribution of the cluster size frequencies is shown in figure 5.8, while figure 5.9 shows a visualisation of the overlays of structures in the same cluster for two selected clusters.

Figure 5.8: The frequencies of the cluster sizes after the second iteration of clustering



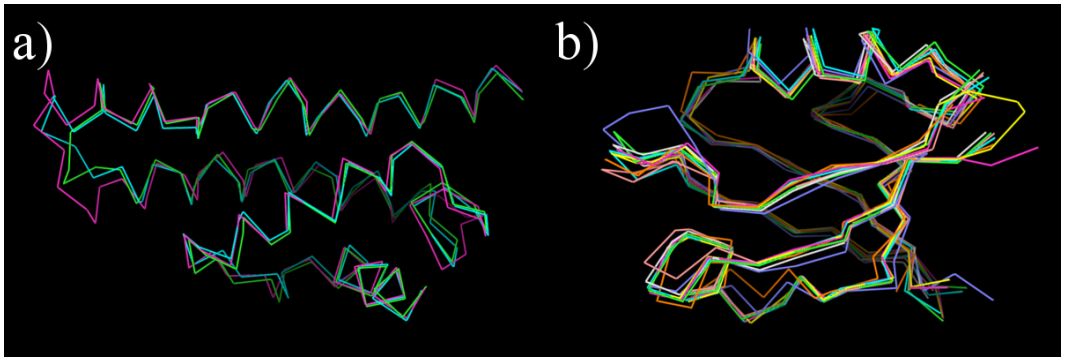
This figure shows the frequencies of clusters with given cluster size in the list of clusters resulting from the second iteration of the equivalent classes clustering of the BALBES protein domain database. Note that the singletons (clusters containing a single domain) are not shown in the main plot, while the inset does not show frequencies of clusters with size less than 5 domains.

Considering that the change of resolution did lead to further reduction of the number of clusters in the BALBES protein domain database, this finding suggests that further iterations of the algorithm with higher resolution could further the clustering accuracy.

5.3.3 Third iteration

The third iteration of the clustering algorithm is, similarly to the second iteration, based on increasing the resolution of the distances computation. Therefore, by

Figure 5.9: Overlay of domains clustered together in the second clustering iteration for selected clusters

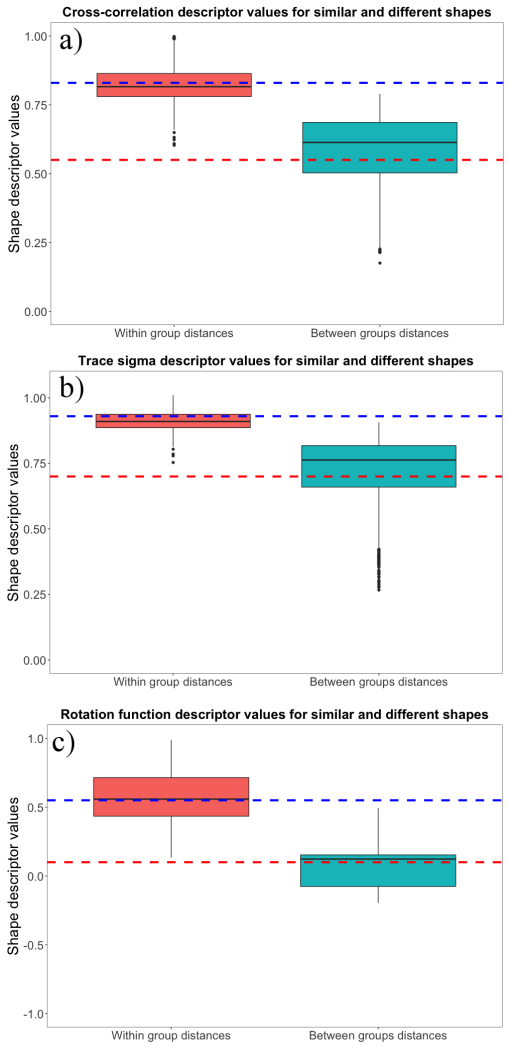


This figure shows two different clusters resulting from the second iteration of the equivalent classes clustering of the BALBES protein domain database. The left cluster a) consists of three domains (1FW1_A_dom_2, 3N5O_A_dom_2 and 2V6K_A_dom_2), while the right cluster consists of eight domains (3GSL_A_dom_2, 2BYG_A_dom_1, 3I4W_A_dom_1, 2HE2_A_dom_1, 2FE5_A_dom_1, 3GSL_A_dom_1, 1W9E_A_dom_2 and 3QE1_A_dom_1). For all structures, only the traces of their backbone atoms are shown. The domains were aligned using `ProSMART` (Nicholls et al., 2014) and visualised using `PyMoL` (Schrodinger LLC, 2015).

consulting the tables 3.5, 3.6 and 3.7, it is clear that increasing the resolution from the previous 6.0 Å to 5.0 Å leads to higher accuracy (as measured by the AUROC measure on the test dataset) when the B-factors are set to 40. Similarly to the previous iterations, the size requirement is applied to the reduced set of domains resulting from the second iteration of the clustering algorithm. Moreover, in order to apply the hierarchical distance computation and later for the use in the equivalent-classes clustering, the conservative false positive and negative thresholds were decided and their relative positions to the distributions of the similar shape pair distances and different shape pair distances for the new settings are shown in figure 5.10.

The hierarchical distances thresholds for computation of the sparse distance matrix were determined as 0.55 for the cross-correlation-descriptor distance, 0.70 for the trace-sigma-descriptor distance and 0.1 for the rotation-function-based-descriptor distance and used to obtain the matrix. Then, the sparse distance matrix was computed using these values and clustered using the same equivalent-classes algorithm with the same equivalent relation as defined in section 5.2.3. The conservative false-positives-reducing thresholds were determined as 0.83 for

Figure 5.10: The false-positives and false-negatives-reducing thresholds for the cross-correlation, trace-sigma and rotation-function-based descriptors for the third iteration of the clustering algorithm



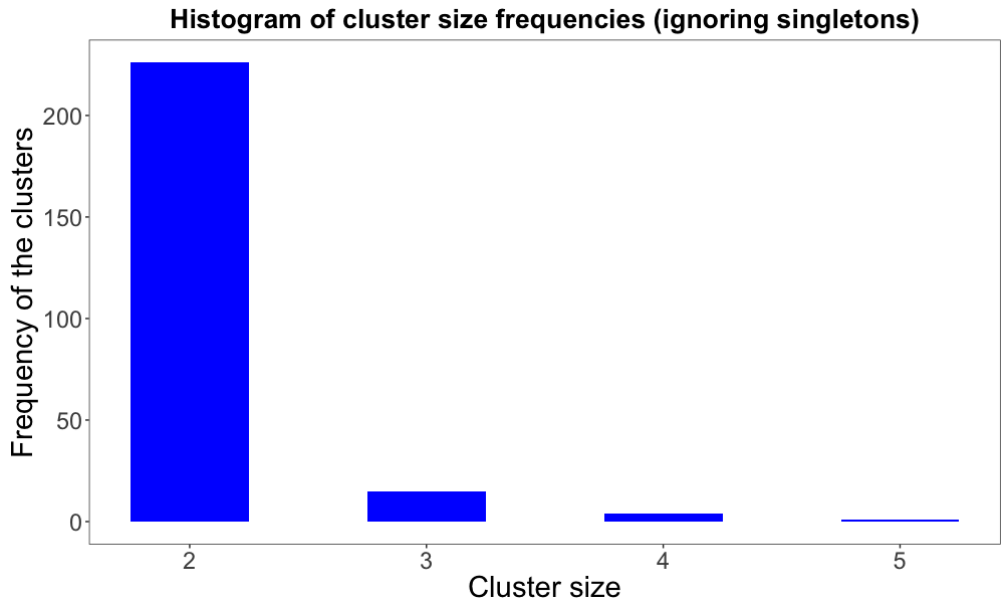
This figure shows the distributions of distances between the similar-shape pairs (in red) and different-shape pairs (in turquoise) of the test dataset structures as computed using the resolution of 5.0 Å and B-factor values 40. The cross-correlation descriptor is shown in part a), the trace-sigma descriptor is shown in part b) and the rotation-function-based descriptor is shown in part c). The blue dashed lines show how the conservative false-positives-reducing threshold relates to these two distributions for each descriptor (the values are 0.83 for the cross-correlation descriptor, 0.93 for the trace-sigma descriptor and 0.55 for the rotation-function-based descriptor), while the red dashed lines show how the conservative false-negatives-reducing threshold relates to the two distributions (the values are 0.55 for the cross-correlation descriptor, 0.70 for the trace-sigma descriptor and 0.1 for the rotation-function-based descriptor).

the cross-correlation descriptor, 0.93 for the trace-sigma shape-descriptor and 0.55 for the rotation-function-based shape-descriptor.

The resulting list of clusters contains 11, 595 clusters and therefore reduces the BALBES database by further $\approx$ 2.3 % from the second iteration results (or by total of $\approx$ 15.5 % reduction of the original BALBES database size). The frequencies of clusters with different sizes in the new list of clusters is shown in figure 5.11, while a visualisation of two of the newly found clusters is shown in figure 5.12.

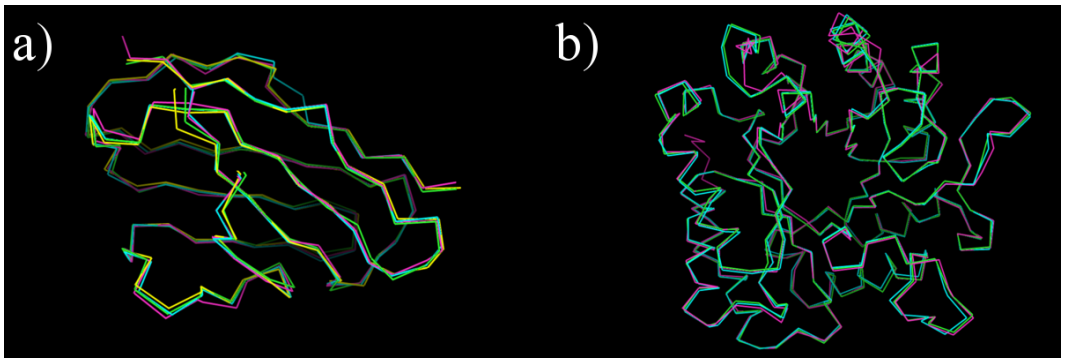
Finally, it is worth noting that several domains clustered together in this

Figure 5.11: The frequencies of the cluster sizes after the third iteration of clustering



This figure shows the frequencies of clusters with given cluster size in the list of clusters resulting from the third iteration of the equivalent classes clustering of the BALBES protein domain database. Note that the singletons (clusters containing a single domain) are not shown.

Figure 5.12: Overlay of domains clustered together in the third clustering iteration for selected clusters



This figure shows two different clusters resulting from the third iteration of the equivalent classes clustering of the BALBES protein domain database. The left cluster a) consists of four domains (1BFO_A_dom_1, 1H8N_A_dom_1, 3D9A_L_dom_1 and 1ZAN_L_dom_1), while the right cluster consists of three domains (3IGU_A_dom_1, 1KTB_A_dom_1 and 3HG3_A_dom_1). For all structures, only the traces of their backbone atoms are shown. The domains were aligned using `ProSMART` (Nicholls et al., 2014) and visualised using `PyMoL` (Schrodinger LLC, 2015).

iteration were found to be the same domain. This has happened as a result of the redundancies in domains discussed in section 5.2.4; generally, when two domain clusters have some identical entries, but the rest of entries differ and not all entries of one cluster do pass the equivalence relation with all entries of the other cluster,

the clusters are kept separate with some entries being present in both clusters. In some cases, it seems that these redundant domains became the representative structures for both clusters and therefore these two clusters are now represented by the same structure (and consequently are clustered together in the next iteration of the algotithm).

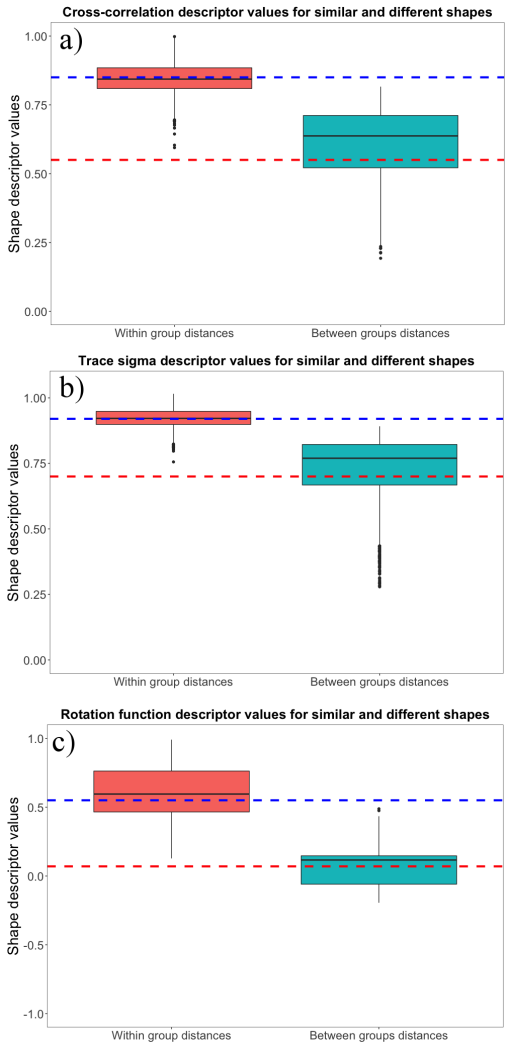
5.3.4 Fourth iteration

In order to increase the resolution further for the fourth iteration, tables 3.5, 3.6 and 3.7 show that the resolution of 4.0 Å and B-factor value 40 do have lower accuracy as measured by the AUROC measure on the test dataset for all three descriptors. However, since the differences in the AUROC values are rather small and produced from a test set two orders of magnitude smaller than the size of the reduced BALBES protein domain database, it may be worth attempting this resolution as the settings for this iteration of the clustering algorithm.

Therefore, the conservative false-positives and false-negatives-reducing thresholds now need to be determined for the new settings. The selected values for the false-positives-reducing threshold are 0.85 for the cross-correlation descriptor, 0.92 for the trace-sigma descriptor and 0.55 for the rotation-function-based descriptor, while the false-negatives-reducing threshold value are 0.55 for the cross-correlation descriptor, 0.70 for the trace-sigma descriptor and 0.07 for the rotation-function-based descriptor. The relative position of these thresholds to the distributions of the similar and different shapes pair distances for the settings are shown in figure 5.13.

The thresholds shown in figure 5.13 were used to compute the sparse distance matrix for all against all protein domains in the reduced BALBES protein domain database resulting from the previous iteration of the clustering algorithm. Subsequently, the equivalence-classes clustering algorithm was used to cluster the resulting distances using the same equivalence relation as before, except using the new thresholds for the descriptor distances. This resulted in a list of $11,210$ clus-

Figure 5.13: The false-positives and false-negatives-reducing thresholds for the cross-correlation, trace-sigma and rotation-function descriptors for the fourth iteration of the clustering algorithm



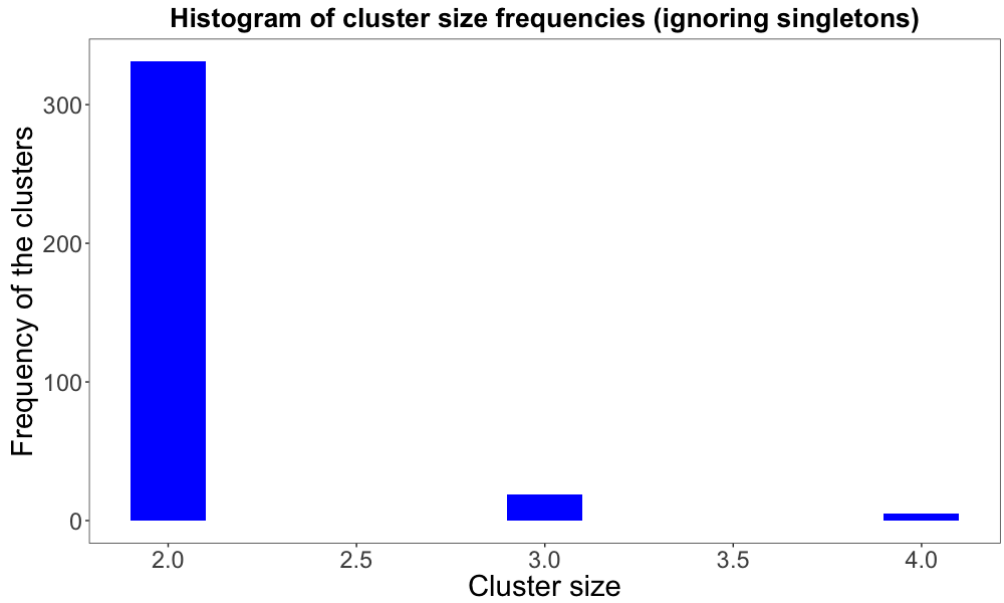
This figure shows the distributions of distances between the similar-shape pairs (in red) and different-shape pairs (in turquoise) of the test dataset domains as determined using the resolution of 4.0 Å and B-factor value of 40. The cross-correlation descriptor is in part a), the trace-sigma descriptor is shown in part b) and the rotation-function descriptor is shown in part c). The blue dashed lines show how the conservative false-positives-reducing threshold relates to these two distributions for each descriptor (the values are 0.85 for the cross-correlation descriptor, 0.92 for the trace-sigma descriptor and 0.55 for the rotation-function-based descriptor), while the red dashed lines show how the conservative false-negatives-reducing threshold relates to the two distributions (the values are 0.55 for the cross-correlation descriptor, 0.70 for the trace-sigma descriptor and 0.07 for the rotation-function-based descriptor).

ters, thus reducing the BALBES protein domain database by further ≈ 3.3 % or a total combined reduction from the original size of ≈ 18.7 %.

There were no clusters resulting from previous clustering iteration clusters being represented by the same structure and, furthermore, there were no duplications (*i.e.* clusters sharing the same domain) in the results of this iteration. The distribution of cluster sizes in this iteration of the clustering algorithm is shown in figure 5.14 and shows that most of the new clusters now rarely consist of more that two structures. Finally, two selected new cluster overlays are shown in figure 5.15 to allow the visual comparison of the clustered shapes. With these results,

the iterative clustering approach was stopped; this decision was reached partly because the goal of the algorithm has been reached - it has demonstrated that the shape descriptor distance can be used to reduce shape redundancy in a dataset of structure; and and partly because increasing the resolution further would increase the computational costs considerably.

Figure 5.14: The frequencies of the cluster sizes after the fourth iteration of clustering

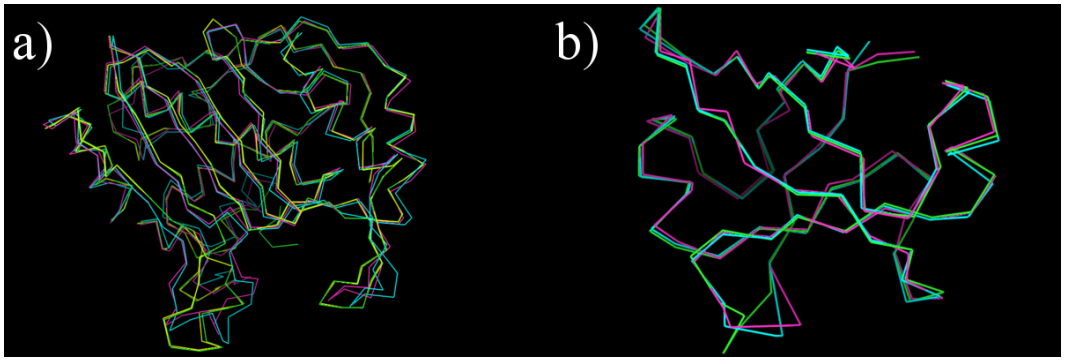


This figure shows the frequencies of clusters with given cluster size in the list of clusters resulting from the fourth iteration of the equivalent classes clustering of the BALBES protein domain database. Note that the singletons (clusters containing a single domain) are not shown.

5.4 Reduced database organisation

By applying the four iterations of the equivalent-classes algorithm with different resolutions, the BALBES protein domain database has now been reduced to $11,210$ protein domains. Recalling that the intention behind the clustering of the BALBES database was partly to allow faster comparisons against the database, it is interesting to consider whether the database could be organised is such a way as to allow faster comparisons.

Figure 5.15: Overlay of domains clustered together in the fourth clustering iteration for selected clusters



This figure shows two different clusters resulting from the fourth iteration of the equivalent classes clustering of the BALBES protein domain database. The left cluster a) consists of four domains (3ED7_A_dom_1, 3IKO_A_dom_1, 3BGX_A_dom_1 and 3EDW_X_dom_1), while the right cluster consists of three domains (3HU3_A_dom_1, 3TIW_A_dom_1 and 3CF2_A_dom_1). For all structures, only the traces of their backbone atoms are shown. The domains were aligned using `ProSMART` (Nicholls et al., 2014) and visualised using `PyMoL` (Schrodinger LLC, 2015).

5.4.1 Pre-computation of the database

The first idea to consider is that when all the distances between the database and a single structure of interest are to be computed, the whole database needs to be read in, one file at a time. Each such file, be it a map file or a PDB file, is then converted onto a centred density map, mapped onto a set of concentric spheres and has its spherical harmonics computed. This makes sense for a single comparison against the database, but when the case of multiple comparisons against the same database is considered, it becomes clear that all the aforementioned steps are repeated many times and are therefore redundant.

Therefore, it would be beneficial to convert the database from a list of PDB files to a list of spherical harmonics coefficients, which can be read in and used directly, bypassing the need for the redundant computations. Moreover, by saving the list of database domains spherical harmonics coefficients in a single binary file with a given structure, the speed of reading the structure files from the hard drive could be increased. Nonetheless, by converting the list of protein domain files to a single binary file with the protein domain spherical harmonics coefficients, there

would be a limit on many of the parameters used for computing the distances between structures.

More specifically, if a binary database of spherical harmonics coefficients were to be built with the default resolution of 8.0 Å, then other structures could only be compared to this database using this value. The other settings, such as the bandwidth (the maximum band to which the spherical harmonics coefficients are computed) could allow for comparisons against a database with lower or equal bandwidth than the database was computed with, but it would again not be possible to compute the distances using higher bandwidth than that which has been used to compute the database. This realisation leads to the need of saving the settings used to produce the database in the binary database file as well as all the spherical harmonics coefficients.

On the other hand, as long as the database was built using the same settings as those used to compare against it, there is considerable computational cost decrease that can be gained. Therefore, the functionality required for saving any list of protein domains into a binary file as well as the settings was written. Furthermore, the functionality required to compare a single structure against the database was developed and a test database containing a random selection of $1,000$ protein domains from the reduced BALBES database was built using the default settings as discussed in section 3.5.5.

The time required to compile the database file was ≈ 21.59 seconds (mean of 100 runs, standard deviation of ≈ 0.59 seconds) on a single core (of the 3.1 GHz Intel Core i7 processor) of a MacBook Pro laptop with 16GB of RAM memory. Subsequently, searching this test database against a single structure took ≈ 15.72 seconds (mean of 100 runs, standard deviation is ≈ 0.58 seconds) on the same machine. This compares with the ≈ 26.31 seconds (mean of 100 runs, standard deviation of ≈ 0.93 seconds) that were required for computing the same $1,000$ distances between the single structure and the other $1,000$ structures which were used to create the database, but this time without pre-computing the database

first and rather using the PDB files directly. These results suggest that using the database functionality reduces the computational cost when at least three or more comparisons against such database are made, although it should be noted that this was a single set of $1,000$ structures and results may vary if differently sized structures were to be used.

5.4.2 Using volume to reduce the number of comparisons

Another idea how the number of comparisons against a database could be reduced is that structures with very different volumes may not need to have their distances computed as the purpose for computing the distances against the whole database is presumably to find similar structures. Therefore, by computing the volume of the structure compared against the database and then only considering database entries with similar volume should provide all required results while reducing the computational cost of comparison against the whole database.

Considering that different amount of dissimilarity in the volume may need to be allowed for different purposes of comparisons against the database, the database reading functionality was implemented as follows: The structure to be compared against the database is loaded and its dimensions are computed. Then, each of the dimensions is reduced by a factor of $1.0-d$, where the parameter d has default value of 0.1, but this value can simply be changed by the user; the minimum volume is then computed as the volume enclosed by the reduced dimensions. Similarly, the maximum volume is computed as the volume enclosed by the dimensions enlarged by a factor of $1.0 + d$. Finally, all database entries are required to have their volume in the range defined by the minimum and maximum volumes and only those database domains passing this requirement have the distances to the structure of interest computed and reported.

With the volume checking applied to the comparison of a structure against a database, it is now also interesting to consider organising the database by sorting the domains by their respective volume and saving them in this order. While such

approach would increase the time required to build the database, the database searching could then be done only until the database entries have volume higher than the maximum allowed for comparing to the query structure. This approach reduces the need to read some portion of the structures from the database at all and therefore should allow faster comparisons against the database.

To test these approaches, the database binary file creation was changed to first sort the structures based on their volume and then save their spherical harmonics coefficients into the file. This new approach took $\approx$ 31.19 seconds on the aforementioned machine (mean of 100 runs with standard deviation of $\approx$ 1.67 seconds) for the test case of $1,000$ structures database. However, when a single structure was searched against the sorted database with the volume check implemented, it only required $\approx$ 8.07 seconds (mean of 100 runs with standard deviation of $\approx$ 0.33 seconds) to search this structure against the database; it should be noted that in the case of this structure, 374 of the database entries did pass the volume check.

5.4.3 File size of the binary database file

Finally, it is worth considering the file sizes of the binary file as well as the original PDB files and consider how the new database may be made available to anyone interested. The final reduced database contains $11,210$ domains, each in its own PDB file; the total disk space required to store these files is 1, 570 MB and these files can be compressed to a single tar container with size of 302.7 MB. However, the binary database file storing the spherical harmonics coefficients and using the default settings requires 7, 670 MB and can be compressed to a tar container with size of 2, 290 MB. These facts show that distributing the binary database would require considerably more space than distributing the PDB files. Furthermore, given that any user of the software can build the database using any preferred settings they may require in the order of minutes, it seems much simpler to build

the binary database on any computer which the users may want to use anew, instead of copying it from any available repository.

Chapter 6

The `ProSHADE` software tool

The previous chapters have described the methods used to derive the three shape descriptors and their use in re-clustering the BALBES protein domain database, as well as how symmetry can be detected and described using the inverse SO(3) Fourier transform approach intrinsic to the rotation-function-based descriptor. Throughout these chapters, all results were obtained using code developed specifically to compute the required information and as a result, this code was developed from simple test scripts to a software tool, which can now be introduced.

6.1 Technical information

The protein shape descriptors and symmetry detection tool named `ProSHADE` (`Pro`tein `SHA`pe `DE`scriptors and symmetry detection) is written in the C++ language and conforms to the ISO C++11 standard (International Organization for Standardization, 2011). The tool is used through a single binary executable, installation of which is currently limited to Unix based system including standard Linux distributions and MacOS. `ProSHADE` is accessed and used through the command line console and does not have any user interface at the current version. Moreover, there is a dynamic library containing the same functionality as the binary

executable, which can be linked to any other projects and allows programmatic access to the results.

6.1.1 Dependencies

The ProSHADE tool makes use of the already developed and well tested computer libraries to accomplish particular tasks required for its proper execution. While these libraries do need to be installed before the ProSHADE tool itself can be compiled and linked on any computer, they provide numerical stability and standardisation, as well as allow this project to focus on its aims instead of re-implementing already available functionality. The complete list of ProSHADE dependencies and their purpose follows:

- **cmap, Clipper and MMDB libraries**

  ProSHADE uses the cmap library available from CCP4 (Winn et al., 2011), the Clipper library of Cowtan (2002) and the MMDB library of Krissinel et al. (2004) to read in and manipulate both the input types (*i.e.* the PDB files as well as the MAP files). All of these libraries are part of the CCP4 suite (Winn et al., 2011).

- **FFTW library**

  In order to execute fast and numerically stable Fourier transforms, ProSHADE uses the FFTW library of Frigo and Johnson (2005) to provide this functionality in the Cartesian co-ordinates as well as the basis for computation of the Fourier transform in spherical co-ordinates and on the SO(3) group.

- **LAPACK library**

  The LAPACK library of Dongarra et al. (1992) is used to compute singular value expansions for complex matrices in reliable and numerically stable manner.

- **SOFT2.0 library**

  Finally, the SOFT2.0 library of Kostelec and Rockmore (2008) is used to

compute the spherical harmonics expansion and SO(3) Fourier transforms by extending the aforementioned FFTW library.

6.1.2 Installation

The current version of the ProSHADE tool is freely available from the FusionForge; the project website is http://fg.oisin.rc-harwell.ac.uk/projects/proshade/ and the stable version source codes can be obtained anonymously using the Bazaar revision control system (http://wiki.bazaar.canonical.com/Bzr) using the online checkout command shown below, from the command line of any Unix based operating system.

```
bzr checkout http://fg.oisin.rc-harwell.ac.uk/anonscm/bzr/proshade/trunk
```

Automatic installation using CMake

The trunk (*i.e.* the stable version) folder obtained by executing the aforementioned command contains the source codes as well as the CMake (Martin and Hoffman, 2007) configuration files. Therefore, by invoking the following commands, automatic installation of the ProSHADE executable, library and examples will be attempted by CMake:

```
cmake CMakeLists.txt
make
make install
```

Upon successful completion, the executable will be installed in the ./bin folder, the library will be installed in the ./lib folder with the required include header file in the ./include folder. It is worth noting that except for the FFTW library, the automatic installation scripts will attempt to install all of the dependencies anew. The reason for this approach is that specific compilation options (*e.g.* -fPIC) are required to be used in the dependency compilation so that it could be used to compile the ProSHADE dynamic library and as a result, it is

non-trivial to be sure that these options were used when the dependencies were installed originally.

Manual installation from source codes

It is also possible to install ProSHADE using manual compilation from source codes; however, this approach is non-trivial. The ProSHADE documentation available in the ./documentation folder in the trunk directory does provide the details as to how this can be done, although at the current version it only supports static library linking.

6.1.3 Documentation

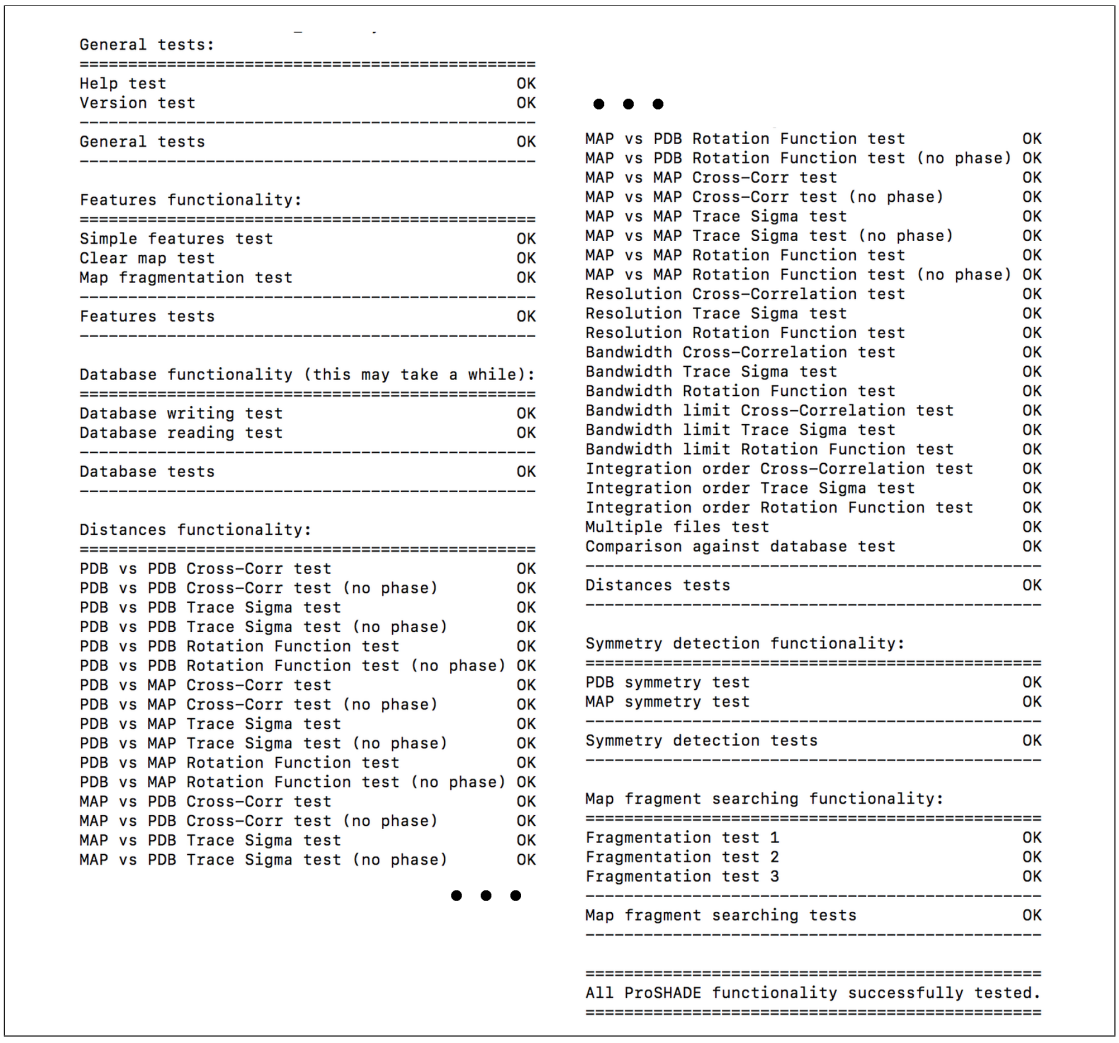
The trunk directory also contains a folder called documentation, which contains a complete HTML documentation for the ProSHADE tool. The user can easily visualise the documentation by opening the index.html file in any web browser. The documentation is automatically generated using the Doxygen software of van Heesch (2016) and it includes a full manual installation guide with the URLs to all dependencies project websites and all command line commands required to install the ProSHADE tool. Moreover, the documentation includes a manual for using the ProSHADE tool as well as examples and their expected output, complete description of all functions and classes used in the code and the code itself.

6.1.4 Testing

Upon successful installation, the user should test the ProSHADE executable on their computer to make sure it works properly. For this purpose, there are over 40 simple tests available in the tests folder of the trunk directory. Each of these tests attempts one simple run of the ProSHADE tool focused on a particular functionality and has hardcoded correct output, which it compares to the real

output on the users computer. In order to run all the tests and see the text overview results, the user can type `make testProshade` after navigating to the ProSHADE folder in the command line, assuming the ProSHADE has been installed using the automated CMake approach; if ProSHADE was installed manually, the user needs to type `make test` instead. A typical view of the tests results screen is shown in figure 6.1.

Figure 6.1: Testing procedure output



6.2 Using the ProSHADE tool

The ProSHADE tool was developed in a modular fashion and so the usage slightly changes depending on the functionality that is required. Nonetheless, care has

been taken to make sure that identical or closely related features are controlled by the same command line arguments in all cases. Moreover, the GNU command line options standard (as described by the GNU Organisaiton (2018)) have been adhered to and therefore the users familiar with other command line tools should find the entering of command line arguments simple. The standard command line options such as `-h` or `--help` for help dialogue, `-v` or `--version` for version number and `--verbose` are all defined and can be used. Figure 6.2 shows the general flow of the tool and selection of the parameters that the user can use to modify its behaviour.

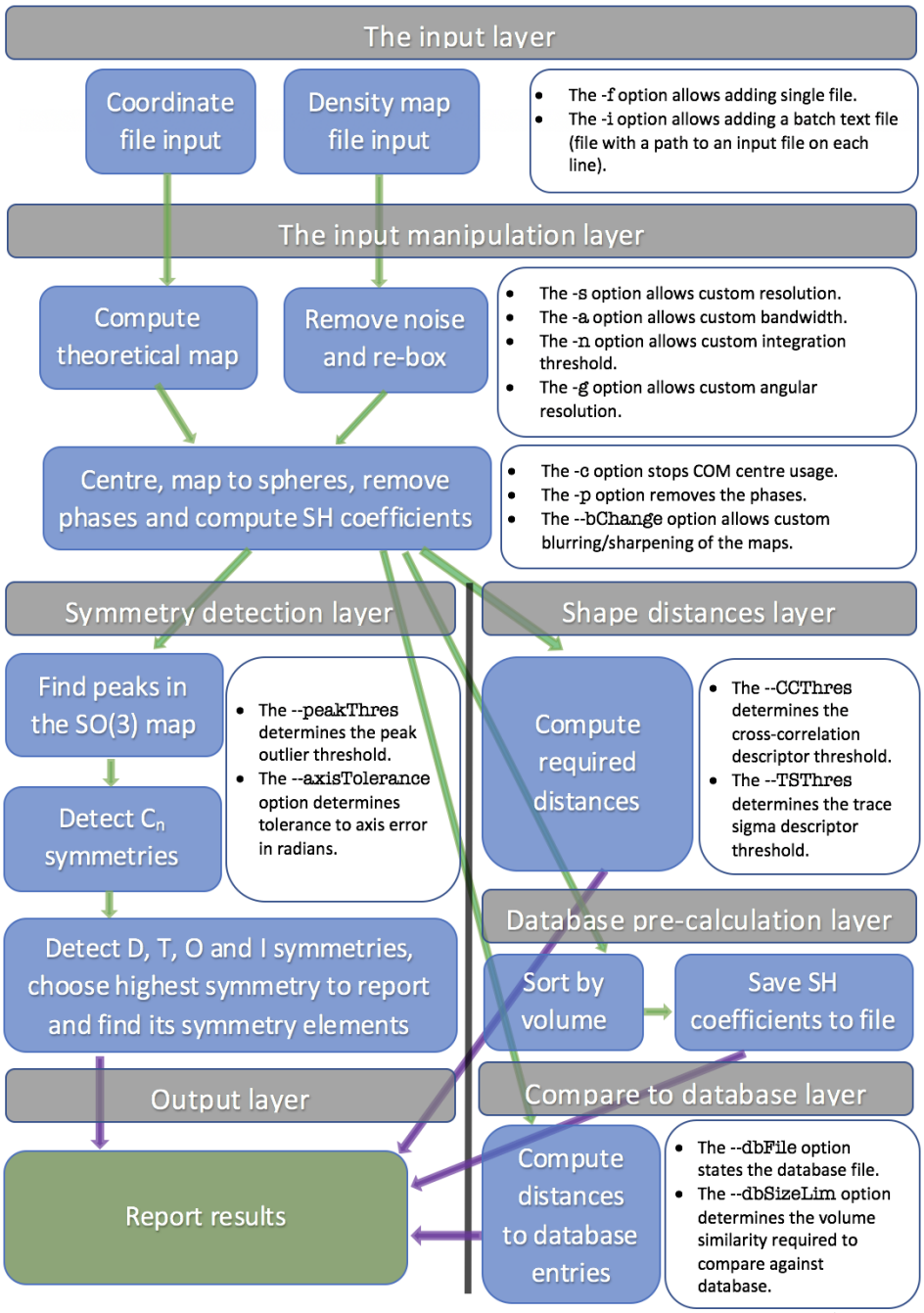
6.2.1 Detecting symmetry using **ProSHADE**

In order to detect symmetry in either a co-ordinate input file, or in a map input file, the `ProSHADE` executable needs to be supplied with the option `-S` or `--symmetry` and it will also require a single input file to be supplied using the `-f` option. These two options are the only mandatory options, although there are many additional values that the user can supply to supersede the default values and therefore modify the operation of the `ProSHADE` executable to fit the particular purpose of the user.

One specific option regarding the symmetry detection mode should be noted; the `--sym` (or `-u`) option allows the user to state which symmetry they believe to exist in the structure. The allowed values for this command line argument are Cx for a cyclic symmetry of fold x, Dx for a dihedral symmetry of fold x, T for a tetrahedral symmetry, O for an octahedral symmetry and I for an icosahedral symmetry. When this option is used, it removes the default behaviour of returning the highest detected symmetry and instead the symmetry requested by the user is returned, if it can be found in the structure.

To demonstrate how the tool can be run and the standard output for the symmetry mode of operation, the current version of the `ProSHADE` executable was used to detect the symmetry of a density map of the bacteriophage T4 portal

Figure 6.2: The flowchart of ProSHADE execution

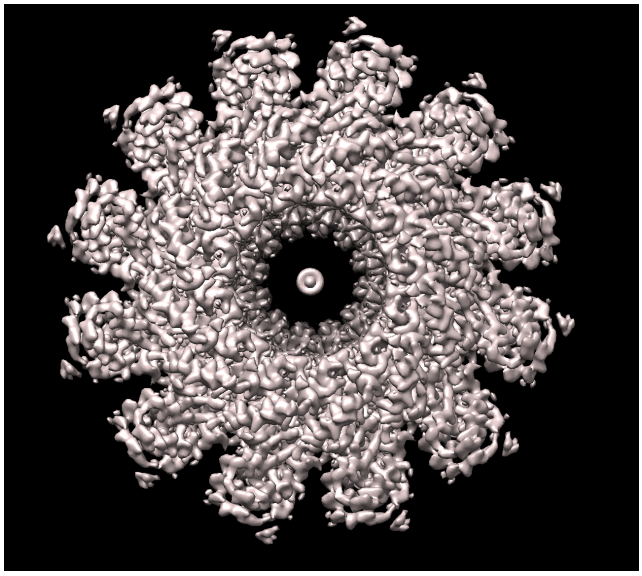


This figure shows the flow of the ProSHADE tool execution for all available functionality. The different layers or modules are shown as grey headings, various computational tasks are shown as blue bubbles and the main command line options used to control their execution are given in the white bubbles. The green arrows show the how the execution of the ProSHADE software tool progresses, while the purple arrows connect the final steps of each mode to the output printing layer.

protein with the PDB accession code 3JA7 (Sun et al., 2015), which has the C_{12} symmetry. The visualisation of the structure is shown in figure 6.3, while the

output of the ProSHADE tool follows. It is worth noting that this example, albeit in less details, has been recently published by the author in Nicholls et al. (2018) along with the algorithm used in ProSHADE software tool.

Figure 6.3: A visualisation of the 3JA7 density map



This figure shows the density map for the bacteriophage T4 portal protein structure with PDB accession code of 3JA7 (Sun et al., 2015) as visualised by Chimera (Pettersen et al., 2004).

```
user$: ./proshade -S -f ./3JA7.map

ProSHADE 0.6.1 (JUN 2018):
==========================


------------------------------------------------------------
|                        MODE: Symmetry                    |
------------------------------------------------------------


Structure loaded.
C symmetries detected.
D symmetries detected.
T, O and I symmetries detected.
>> Generation of T, O and I symmetry group elements complete.

------------------------------------------------------------
|                        COMPLETED                         |
------------------------------------------------------------
```

```
|                                RESULTS                                |

Detected Cyclic symmetry

Symmetry axes table:
```

Symmetry Type	Fold	x	y	z	Angle	Peak height
C	12	+0.02	+0.02	+1.00	2pi / 12	+0.443

Symmetry elements table:

Symmetry Type	x	y	z	Angle (deg)
E	+1.00	+0.00	+0.00	+0.0
C12	+0.02	+0.02	+1.00	−150.0
C12	+0.02	+0.02	+1.00	−120.0
C12	+0.02	+0.02	+1.00	−90.0
C12	+0.02	+0.02	+1.00	−60.0
C12	+0.02	+0.02	+1.00	−30.0
C12	+0.02	+0.02	+1.00	+30.0
C12	+0.02	+0.02	+1.00	+60.0
C12	+0.02	+0.02	+1.00	+90.0
C12	+0.02	+0.02	+1.00	+120.0
C12	+0.02	+0.02	+1.00	+150.0
C12	+0.02	+0.02	+1.00	+180.0

Alternatives:

Symmetry Type	Fold	x	y	z	Angle	Peak height
C	4	+0.01	+0.01	+1.00	2pi / 4	+0.581
C	2	+0.01	+0.01	+1.00	2pi / 2	+0.443
C	12	+0.02	+0.02	+1.00	2pi / 12	+0.443
C	6	+0.02	+0.01	+1.00	2pi / 6	+0.399
C	3	+0.01	+0.01	+1.00	2pi / 3	+0.340

6.2.2 Computing shape distances using **ProSHADE**

The distances computation mode is signalled to the ProSHADE executable by the command line argument -D or --distances. This mode requires two or more structures to be supplied either using the -f command line option, or using the batch option -i, which precedes a text file with a single structure path per line for any number of lines. Note that the results are calculated only for the first structure against all the remaining structures, **not** for all against all distance matrix.

There are multiple command line options that the user has available to specify the particulars of how the distances between the structures should be computed; the main command line options include the -s option for entering the required resolution to which the structures should be described. The -a option allows specifying the maximum spherical harmonics bandwidth, the -n option allows determining the maximum Gauss-Legendre integration limit, while the -g option allows determining maximal angular resolution. The hierarchical distances calculation (discussed in section 5.1.2) are controlled by the --CCThres and the --TSThres options for supplying the cross-correlation and trace sigma thresholds. The user should consult the help dialogue available through the -h option for a full list of command line parameters and their description.

To demonstrate the output of the ProSHADE software tool for computing distances between structure shapes, the distances between the BALBES protein domains 1BFO_A_dom_1 and 1H8N_A_dom_1 (which have similar shape) and the 3IGU_A_dom_1 domain which has a different shape, as can be seen from the figure 5.12 (the first two domains are both in cluster a), while the last domain is from the cluster b)). The output is the ProSHADE software tool is shown below:

```
user$: ./proshade -D -f ./1BFO_A_dom_1.pdb -f ./1H8N_A_dom_1.pdb
-f ./3IGU_A_dom_1.pdb -s 4

ProSHADE 0.6.1 (JUN 2018):
==========================


------------------------------------------------------------
|                      MODE: Distances                     |
------------------------------------------------------------

Finding distances between the structure ./1BFO_A_dom_1.pdb against all other
structures.
Computing the cross-correlation distances.
Computing the trace sigma distances.
Computing the rotation function distances.

------------------------------------------------------------
|                      COMPLETED                           |
------------------------------------------------------------


------------------------------------------------------------
|                      RESULTS                             |
------------------------------------------------------------


Energy Level Descriptor distances      :          +0.85959     +0.4909
Trace Sigma Descriptor distances       :          +0.94150     +0.6537
Rotation Function Descriptor distances :          +0.72780     -0.1663
Matching structure names               :   ./1H8N_A_dom_1.pdb
Matching structure names               :   ./3IGU_A_dom_1.pdb
```

6.2.3 Building structure database using **ProSHADE**

In order to build a database using the ProSHADE tool, the command line option -B needs to be supplied as well as a list of structures to be saved into the database (given by either the -f or the -i options as discussed above) and the name for the database file preceded by the --dbFile command line option. These three elements are mandatory for ProSHADE execution in the database building mode, but the same list of command line options as in the section 6.2.2 can be used to modify how the spherical harmonics coefficients are computed before saving.

It should also be noted that once the database is created, all structures which the user wants to search for in the database will be processed using the same settings as the settings used to build the database; the reason is that once the database is computed, its settings cannot be changed unless a new database is computed again. Furthermore, the database is saved in an sorted order according to the total volume enclosed by the domain and this information is later used when searching against such a database, so that only domains with similar volume would be considered. The `--dbSizeLim` option can be used to specify the threshold for what "similar volume" means when searching against the database.

To demonstrate the database building and searching functionalities, the BALBES protein domain cluster produced by the second iteration of the clustering algorithm in the previous chapter and shown in figure 5.9 part b) will be used to build a database and, subsequently, one of the cluster structures will be searched for in the database. The database can be built as follows (including the `ProSHADE` tool output):

```
user$: ./proshade -B -f ./3GSL_A_dom_2.pdb -f./2BYG_A_dom_1.pdb
-f ./3I4W_A_dom_1.pdb -f ./2HE2_A_dom_1.pdb -f ./2FE5_A_dom_1.pdb
-f ./3GSL_A_dom_1.pdb -f ./1W9E_A_dom_1.pdb -f ./3QE1_A_dom_1.pdb
--dbFile "testDB.bin"
ProSHADE 0.6.1 (JUN 2018):
==========================


-------------------------------------------------------------
|                        MODE: BuildDB                      |
-------------------------------------------------------------

Now detecting sizes of structures for database sorting.
Saving files in sorted volume order.

-------------------------------------------------------------
|                        COMPLETED                          |
-------------------------------------------------------------

Database saved to: testDB.bin
```

and a particular structure can then be queried against the database:

```
user$: ./proshade -D -f ./1W9E_A_dom_1.pdb
--dbFile ./testDB.bin --dbSizeLim 0.5
ProSHADE 0.6.1 (JUN 2018):
==========================


------------------------------------------------------------
|                      MODE: Distances                     |
------------------------------------------------------------


Structure 0 loaded.
Structure 1 read from the database.
Structure 2 read from the database.
Structure 3 read from the database.
Structure 4 read from the database.
Structure 5 read from the database.
Structure 6 read from the database.
Structure 7 read from the database.
Structure 8 read from the database.
Computing the cross-correlation distances.
Computing the trace sigma distances.
Computing the rotation function distances.


------------------------------------------------------------
|                      COMPLETED                           |
------------------------------------------------------------


------------------------------------------------------------
|                      RESULTS                             |
------------------------------------------------------------


Energy Level Descriptor distances      :          +0.76005     +0.7434 +0.7112
+0.6794 +0.7559 +0.6811 +0.6975 +1.0000
Trace Sigma Descriptor distances       :          +0.88031     +0.8885 +0.8580
+0.8734 +0.9085 +0.8301 +0.8753 +1.0000
Rotation Function Descriptor distances :          +0.37403     +0.3777 +0.3014
+0.2518 +0.3886 +0.2208 +0.3160 +1.0000
Matching structure names               :   ./3GSL_A_dom_1.pdb
Matching structure names               :   ./3QE1_A_dom_1.pdb
Matching structure names               :   ./2BYG_A_dom_1.pdb
Matching structure names               :   ./3GSL_A_dom_2.pdb
Matching structure names               :   ./2HE2_A_dom_1.pdb
Matching structure names               :   ./3I4W_A_dom_1.pdb
Matching structure names               :   ./2FE5_A_dom_1.pdb
Matching structure names               :   ./1W9E_A_dom_1.pdb
```

6.3 Final remarks

It should be noted that while the development of the ProSHADE software tool has reached a stage where it is available to the community and can be used, it still does require further development. The future steps intended at this point include, for example, an overlay mode capable of finding the optimal translation and rotation for fitting PDB and MAP files into each other and rotation group optimisation capable of optimising the symmetry group elements positions (*i.e.* the symmetry axis and angle) to better fit the discovered symmetry group. Moreover, while ProSHADE can currently be installed as a standalone software tool, it is in the process of becoming part of the CCP-EM software suite (Burnley et al., 2017); this step will, hopefully, make the software tool more easily available to the community in general. Finally, to further the ease of use of the ProSHADE tool, a development is currently underway to use the Swig (Swig Development Team, 2018) software to create a Python language module from the dynamic library already available.

Chapter 7

Summary, conclusions and future direction

The previous chapters have introduced the mathematical framework for computing the three three-dimensional shape distances with particular focus on protein domain shapes. The usage of this mathematical framework for symmetry detection has been explored as the possibility emerged, along with the limitations of this approach. Furthermore, the direct application of the shape descriptors in clustering of the BALBES protein domain database has been shown and a software tool capable of performing all the discussed tasks was developed and demonstrated. Finally, this chapter discusses these results in a wider context as well as their further applications and possible research directions.

The initial aim of the project was to develop methods and tools for finding macromolecular structures with similar shape in any given dataset, as discussed in section 1.2. To reach this goal, five different shape description approaches were briefly explored with regards to the macromolecular data features that needed to be accommodated for by the descriptors. Moreover, the robustness of these descriptors was considered in terms of the relative descriptor value changes in response to displacement or deletion of a percentage of the residues. This initial analysis has suggested that the spherical harmonics expansion method is well suited for the

purpose of finding shape similarities in macromolecular structures and therefore became the basis of this project.

7.1 Shape descriptors

Consequently, the background of the spherical harmonics expansion method was explored as well as the features it contains. More specifically, the spherical harmonics expansion is not rotationally and translationally invariant, meaning that any shape descriptor based on it needs to introduce these features in itself and several suggestions as to how these features can be obtained in general were mentioned. With this information in mind, three different shape descriptors were developed.

7.1.1 Cross-correlation descriptor

The first shape descriptor is based on the energy level descriptors introduced by Kazhdan et al. (2003), but instead of summing over the individual spherical harmonics frequencies in each band and thus having its dimensionality reduced, it uses the cross-correlation of the identical spherical harmonics bands from various shells. This approach conserves the dimensionality of the data, while introducing the preferred rotation invariance feature. Moreover, the cross-correlation descriptor can be used to show that the imaginary parts of spherical harmonics expansion have to be zero for Patterson maps. This result is general and applies to all functions with real valued Fourier coefficients, although it can be easily shown using the descriptor.

Multiple approaches were explored as to how two three-dimensional arrays of cross-correlation values could be reduced to a single number representing the distance between them. The approach of using Pearson's correlation coefficient to compare two matrices and consequently using arithmetic average to reduce the vector of Pearson's correlation coefficients into a single number has been shown to

be the most accurate on a test dataset with accuracy measured by the AUROC measure.

7.1.2 Trace-sigma descriptor

The second shape descriptor is based on the idea of using the sum of differences in spherical harmonics values as a distance between two structures in the spherical harmonics coefficient space. By attempting to minimise this distance, the problem can be simplified to finding the Wigner $\mathscr{D}(\alpha, \beta, \gamma)$ matrix which maximises the trace of the matrix multiplication of the Wigner matrix and the E matrix (matrix of different spherical harmonics coefficient multiplied and integrated over the radius of their respective shells). Under the assumption that the maximisation for each spherical harmonics band is equal to maximisation for all spherical harmonics at the same time, a faster solution can be found by using the singular value decomposition of the E matrix; the trace-sigma descriptor then becomes the sum of the trace elements of the singular-value matrix Σ. This also makes the descriptor rotation invariant.

Nonetheless, it should be noted that the assumption used above to calculate the trace-sigma descriptor is known not to hold completely. Therefore, the trace-sigma descriptor is not the absolute minimal distance between the two objects in the spherical harmonics coefficients space, but rather its approximation. On the other hand, this approximation is shown to have increased accuracy as compared to the cross-correlation descriptor, suggesting that, at least for the test dataset, the approximation is acceptable.

7.1.3 Rotation-function-based descriptor

The rotation-function-based descriptor returns to the approximation used by the trace-sigma descriptor and finds the complete maximisation solution without any simplifying assumptions. To do this, the descriptor computation requires expansion of the E matrix by the inverse Fourier transform on the SO(3) rotation group

and detection of the largest peak in the resulting rotation-function map. While this descriptor is the most computationally costly from the three descriptors discussed here, it is the most accurate one as measured by the AUROC measure on the test dataset. It should also be noted that by finding the optimal overlay rotation, the rotation invariance is guaranteed up to numerical inaccuracies of the computations.

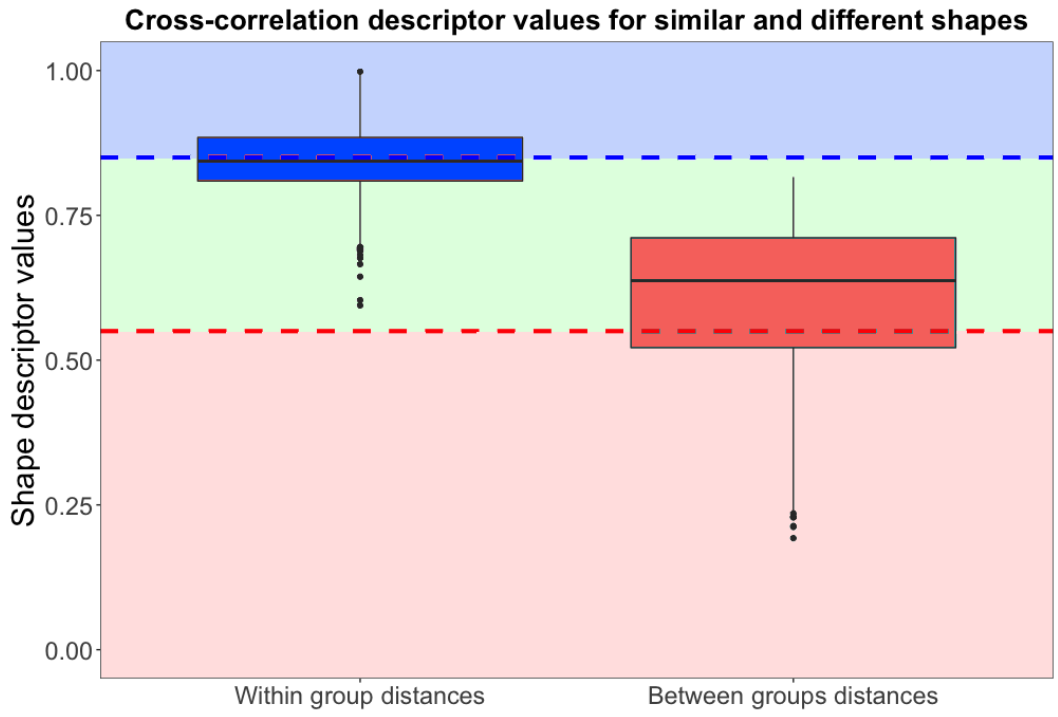
7.2 Detection of similar shapes

The application of the three shape-distance descriptors will dictate the relative importance of the false-positive results, *i.e.* results where the descriptors claim similar shape when in reality the shape is different, and the false-negative results, that is, results where the descriptors claim different shape, but the shape is in reality similar. This difference can be demonstrated by recalling the plots showing the distributions of distances of similarly and differently shaped pairs obtained for a particular descriptor, for example the one shown in figure 5.13 part a) - for convenience replicated here in figure 7.1 with some colouring differences.

It can be seen from the figure 7.1 that any pair of structures scoring above the false-positives-reducing threshold (light blue area) can be considered as having similar shape, while any pair of structures scoring under the false-negatives-reducing threshold (light red area) can be considered to have different shape. Depending on how the *twilight zone* in between the thresholds (shown in light green) is treated, the shape distances can have different applications.

It is worth noting that the thresholds shown in figure 7.1 are arbitrary and there is no perfect way of determining them. The position of the false-positives-reducing threshold needs to be above the highest distance for pairs with different shape, but just how much above is an arbitrary decision. Similarly, the position of the false-negatives-reducing threshold needs to be just below the smallest distance found in the similarly shaped pair distances distribution, but the exact position

Figure 7.1: Cross-correlation distance distributions for similarly and differently shaped datasets



This figure shows the distribution of the cross-correlation-descriptor distances obtained using the resolution of 4.0 Å and 40 *B*-factor settings for similarly-shaped pairs in blue and for differently-shaped pairs in red. The blue dashed line is a false-positives-reducing threshold, while the red dashed line is the false-negatives-reducing threshold. The area above the false-positives-reducing threshold is shaded light blue, the area under the false-negatives-reducing threshold is shaded light red and the area in between the thresholds is shaded light green

is also arbitrary. Nonetheless, it is clear that these two threshold will be different as long as there is overlap between the two distributions and therefore the specific positions of the thresholds are not relevant for this discussion.

7.2.1 Detection of only similar shapes

The first application of the shape descriptors is in the case where only the pairs of structures with high probability of having similar shape are required, that is, where the false-positives removal has higher importance than false-negatives removal. In this case, all pairs with distance score in the twilight zone is treated as different shapes; an example of such an application is the clustering of the BALBES database, where it is more important that the clusters only have similar shapes

than the possibility of not clustering some similar shapes together and instead keeping them as separate clusters.

This type of shape descriptor applications would be expected when a dataset is clustered with intention to reduce the shape redundancies of the dataset or where the resulting clusters will be the basis of further analysis. In the case of removing shape redundancies, this project has shown that the BALBES database can be reduced by over 18.7 % and as a result, any searches against the reduced database are proportionally faster while the diversity of the shapes is mostly conserved.

One possible future application of this approach to the shape descriptors is in the case when automated detection of all protein domains in the whole PDB database would be attempted. Such project could start by first splitting all PDB structures into (presumably overlapping) fragments based on an automated algorithm. This algorithm would only save fragments with high enough number of atoms (to remove empty or almost empty fragments) and fragments with high compactness, definition of which would have to be explored. Then, considering each of the resulting fragments as a potential protein domain, these could be searched against a database of already known protein domains, possibly the reduced BALBES protein domain database. In the case of high probability match (that is, giving higher importance to the false-positives removal), the fragment would be added to the already known protein domain class and its volume completely removed from the local copy of the PDB structure. Consequently, any remaining fragments could be considered as novel protein domains. Clearly, the suggested application would need to be explored in much more details and it is mentioned here only as an example of future application of the shape descriptors method. On the other hand, such an approach seems feasible and would benefit from not being based on manual curation, as well as being amenable to application to any new structures that may be solved in the future.

Another example where clustering similar shapes could lead to very interesting results would be to retrieve all known structures (either as density map or as

co-ordinate file) containing atoms of various metals and then cutting out all fragments with given radius around the metal atom in the centre. By subsequently clustering all such fragments for all metals using the shape-descriptor distances, the resulting clustering classes should be related to the metal coordination number and by extension to the metal co-ordination type. Presumably, a database of the clustering classes for various metals could then be used to predict which metal is present in density map without model or to validate metal types in co-ordinate data files. Again, it should be noted that this suggested application would need to be explored more and would presumably require more work than suggested here. For example, Zheng et al. (2017) suggests that any tools for metal analysis based on the PDB structures need to account for errors in the structures and this element would have to be considered before the suggested application could be successfully developed.

7.2.2 Detection of possibly similar shapes

A very different approach to the shape descriptors applications occurs when removal of false negatives is more important than the removal of false positives; or in other words, when being able to detect all possibly similar structures is more important than allowing some of these to actually have different shape. This approach was used in the hierarchical distance computation when the distances between BALBES protein domain structures were computed. In this case, the computationally cheaper descriptor values were used to determine if the computationally more expensive descriptors need to be computed or not. This approach treats the twilight zone as similar shapes instead of as different shapes and would be expected in applications where the shape descriptors serve as pre-filters for more accurate, but also considerably more computationally expensive method.

One possible application of the shape descriptors based on this principle would be attempting to find molecular replacement candidates using the similarity in shape (more precisely, in the Patterson map shape) between a candidate

database and the structure to be solved instead of the currently used sequence similarity. It is clear that searching for sequence similarity is faster and computationally cheaper than searching for shape similarity, and furthermore, if two structures do not share sequence similarity, the amino acid side chains will differ by definition and this may cause failure of molecular replacement. On the other hand, Lesk and Chothia (1980) have shown that dissimilar sequences can lead to similar structures, while Kosloff and Kolodny (2008) have shown examples where similar sequences lead to different structures. These two results combine to show that sequence is not a perfect predictor of shape similarity and therefore suggest that using shape similarity information, possibly in concert with sequence similarity information may allow for molecular replacement solutions for structures previously inaccessible by this method.

Another possible application of the shape descriptors where the twilight zone is treated as similar rather than different shapes would be automated fitting of protein domain atomic models into (presumably EM) density maps. This application would require splitting of a single density map into fragments, either defined by a sliding box or by some combination of compactness and density detection (possibly using the density-based clustering methods). Nonetheless, the specific approach to fragmenting a map would have to be explored by such a project, but assuming an automated approach to finding clusters of density and cutting these out of the density map can be developed, it would then be possible to search each of the potential domain fragment against a database of protein domains, possibly the reduced BALBES protein domain database. Consequently, any strong match between any fragment and a protein domain would allow placing the protein domain atomic model into the density map in the location of the fragment. Moreover, the inverse SO(3) Fourier transform method discussed in this thesis could be used to determine, at least approximately, the rotation of the atomic model to fit the fragment density; this could be accomplished, for example, by minor modification to the `ProSHADE` software tool introduced in this thesis - it is worth noting that an experimental version of this feature is currently being implemented. The fit

could then be improved by either using the molecular refinement software, such as `phenix.refine` (Afonine et al., 2012) or `REFMAC5` (Kovalevskiy et al., 2018), or by using the available tools such as `ESSENS` (Kleywegt and Read, 1997), `MOLREP` (Vagin and Teplyakov, 1997), `FFFEAR` (Cowtan, 1998), `FOLDHUNTER` (Jiang et al., 2001), `Situs` (Wriggers and Birmanns, 2001), `Modeller` (Eswar et al., 2006), `FOLD-EM` (Saha and Morais, 2012) or the automated approach available in the `Chimera` software (Pettersen et al., 2004).

7.3 Detection of symmetry

The possibility of the symmetry detection became apparent when the rotation-function-based descriptor was being developed and the inverse SO(3) Fourier transform was implemented. The intention for exploring this method was to find the Euler rotation angles which result in the highest structure overlay cross-correlation score so that the optimal rotation can be used in the minimisation problem faced by the descriptor. Nonetheless, it became evident that the overlay of a single structure against itself results in high cross-correlation values in the data when the rotated and original structures have similar shape, or in other words when a symmetry within the structure exists.

Building on this fact, it was possible to develop algorithms capable of confirming the existence of any cyclic (C), dihedral (D), tetrahedral (T), octahedral (O) or icosahedral (I) symmetry groups in the data. However, it needs to be stressed that the algorithms can only detect the chiral variants of the symmetry groups; the achiral variants, which include reflections as well as rotations cannot be detected using this method. While this may be the preferred state, as molecules with different chirality should be treated separately in the case of protein domains, there may be cases where this is not the optimal approach. Moreover, the method was then further extended by adding the ability to report the symmetry axis (or axes) of the detected symmetry group as well as reporting all the symmetry group elements, *i.e.* individual rotations of the asymmetric part of the structure required

to reconstruct the complete structure from it. On the other hand, the detection of helical symmetry has not yet been explored for the inverse SO(3) Fourier transform approach and this is one of the limitations of the current implementation. This limitation should be explored in the future work on this project.

In terms of the applications of this method, the ability to directly determine symmetry in any macromolecule is a useful application on its own. While it is true that most PDB files do contain the symmetry in the header and that the EM software suites such as `EMAN1` (Ludtke et al., 1999), `EMAN2` (Tang et al., 2007) or `RELION` (Scheres, 2012) do detect symmetry in the EM data, the symmetry information is not, for example, available for all EMDB database (Lawson et al., 2015) structures. Therefore, anyone obtaining the structure may not have the symmetry information readily available, a situation remedied by the `ProSHADE` symmetry detection tool. Furthermore, researchers may be interested in knowing the symmetry present in the intermediate maps before their further processing by the EM software suites, at which point the symmetry may not have been determined yet.

Moreover, there are several possible applications for the symmetry determination part of `ProSHADE` tool which would, however, require some extra work to be done before being completely possible. One such application would be to use the symmetry computation to determine the asymmetric parts of both density maps and co-ordinate files. With this information available, it would be possible to save only the asymmetric parts of the structures and a set of the symmetry group elements required to reproduce the complete molecule from the asymmetric part, instead of storing the complete macromolecular structures, thus decreasing the storage requirements for databases such as the wwPDB (Berman et al., 2000) or the EMDB (Lawson et al., 2015). Regarding this application, it should be noted that wwPDB reports $80,643$ structures without any assigned symmetry out of the total $138,678$ structures [*as of 19 March 2018*], meaning that ≈ 41.8 % of wwPDB structures do have some kind of symmetry. Furthermore, it should

also be noted that some structures in the wwPDB have already been deposited as asymmetric fragments only, with the `SCALE`*n* and `MTRIX`*n* fields (where *n* is the transformation parameter) being used to describe the transformations required to produce the full structure. Examples of such structures include the structures with accession codes 2QZV (Anderson et al., 2007), 4HL8 (Casanas et al., 2013) or 5OSN (Roeding et al., 2017). These examples demonstrate that storing only the asymmetric fragments of structures is possible, although it is not yet the standard.

7.4 Further method development

There are several possible improvements in terms of the methods developed as part of this project and the `ProSHADE` software tool in general. One possible improvement would be to implement the translation function search using phased data; an approach similar to the crystallographic translation function described by Crowther and Blow (1967) but using the phase information to find the optimal centring (as described in, for example, Cowtan (1998)). This would allow reducing the inaccuracies resulting from the centre of density centring approach. Furthermore, implementing this functionality would also make matching two input structures simpler as the optimal translation could be found using this approach.

Another possible improvement would be to make `ProSHADE` part of the `CCP-EM` suite (Burnley et al., 2017). This would be advantageous to users as it would allow development of a simple and user-friendly interface. Moreover, if `ProSHADE` was distributed as part of the `CCP-EM` suite, the users could simply obtain a `ProSHADE` binary for their system and avoid the complexities of installing dependencies and linking them properly. Furthermore, by completing the SWIG process for converting the `ProSHADE` library into a Python language module, this would make the usage much simpler for many developers who use the Python language.

Finally, while the aforementioned possible applications and further method developments do require more exploration and thought before any of them can be attempted, they suggest multiple interesting potential applications for the algorithms and the `ProSHADE` software tool in general.

Bibliography

Adrian, M., Dubochet, J., Lepault, J., and McDowall, A. W. (1984). Cryo-electron microscopy of viruses. *Nature*, 308:32–36.

Afonine, P. V., Grosse-Kunstleve, R. W., Echols, N., Headd, J. J., Moriarty, N. W., Mustyakimov, M., Terwilliger, T. C., Urzhumtsev, A., Zwart, P. H., and Adams, P. D. (2012). Towards automated crystallographic structure refinement with phenix.refine. *Acta Crystallographica*, D68:352–367.

Alberts, B. (2014). *Molecular Biology of the Cell*. Garland Science.

Alexandrov, N. N. and Fischer, D. (1996). Analysis of topological and nontopological structural similarities in the PDB: New examples with old structures. *PROTEINS: Structure, Function, and Genetics*, 25:354–365.

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local aligment search tool. *Journal of Molecular Biology*, 215:403–410.

Anderson, D. H., Kickhoefer, V. A., Sievers, S. A., Rome, L. H., and Eisenberg, D. (2007). Draft crystal structures of the vault shell at 9Å resolution. *PLoS Biology*, 5:318–318.

Andreeva, A., Howorth, D., Chandonia, J., Brenner, S., Hubbard, T., Chothia, C., and Murzin, A. (2008). Data growth and its impact on the SCOP database: new developments. *Nucleic Acids Research*, D(36):419–425.

Avelsgaard, C. (1989). *Foundations of Advanced Mathematics*, page 127. Scott Foresman.

Banaszak, L. J. (2000). *Foundations of Structural Biology*. Elsevier, Burlington.

Bartesaghi, A., Sprechmann, P., Liu, J., Randall, G., Sapiro, G., and Subramaniam, S. (2008). Classification and 3D averaging with missing wedge correction in biological electron tomography. *Journal of Structural Biology*, 162:436–450.

Berman, H. M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T. N., Weissig, H., Shindyalov, I. N., and Bourne, P. E. (2000). The protein data bank. *Nucleic Acids Research*, 28(1):235–242.

Blow, D. M. (1958). The structure of heamoglobin VII. determination of phase angles in the non-centrosymmetric [100] zone. *Proceedings of Royal Society London*, 247(1250):302–336.

Blow, D. M. and Rossmann, M. G. (1961). The single isomorphous replacement method. *Acta Crystallographica*, 14:1195–1202.

Borrajo-Pelaez, R. and Hedstrom, P. (2017). Recent developments of crystallographic analysis methods in the scanning electron microscope for applications in metallurgy. *Critical Reviews in Solid State and Materials Sciences*, pages 1–20.

Bragg, W. L. (1913). The diffraction of short electromagnetic waves by a crystal. *Proceedings of Cambridge Philosophical Society*, 17:43–57.

Briggs, J. A. (2013). Structural biology *in situ*–the potential of subtomogram averaging. *Current Opinion in Structural Biology*, 23(2):261–267.

Burnley, T., Palmer, C., and Winn, M. D. (2017). Recent developments in the CCP-EM software suite. *Acta Crystallographica Section D*, D73:469–477.

Cai, W., Shao, X., and Maigret, B. (2001). Protein-ligand recognition using spherical harmonic molecular surfaces: towards a fast and efficient filter for large virtual throughput screening. *Journal of Molecular Graphics and Modelling*, 5301:1–16.

Callaway, E. (2015). The revolution will not be crystallized: a new method sweeps through structural biology. *Nature*, 525(7568):172–174.

Canterakis, N. (1996). 3D Zernike moments and Zernike affine invariants for 3D image analysis and recognition. *Proceedings of the 11th Scandinavian Conference on Image Analysis*, pages 85–93.

Carugo, O. (2007). Statistical validation of the root-mean-square-distance, a measure of protein structural proximity. *Protein Engineering, Design and Selection*, 20(1):33–38.

Carugo, O. and Pongor, S. (2001). A normalized root-mean-square distance for comparing protein three-dimensional structures. *Protein Science*, 10:1470–1473.

Casanas, A., Querol-Audi, J., Guerra, P., Pous, J., Tanaka, H., Tsukihara, T., Vergaguer, N., and Fita, I. (2013). New features of value architecture and dynemics revealed by novel refinement using deformable elastic network approach. *Acta Crystallographica Section D*, 69:1054–1061.

Cheng, H., Schaeffer, R. D., Liao, Y., Kinch, L. N., Pei, J., Shi, S., Kim, B. H., and Grishin, N. V. (2014). ECOD: An evolutionary classification of protein domains. *PLoS Computational Biology*, 10(12):e1003926.

Condon, E. U. and Shortley, G. (1951). *The Theory of Atomic Spectra*. Cambridge University Press, Cambridge, England.

Cowtan, K. (1998). Modified phased translation functions and their application to molecular-fragment location. *Acta Crystallographica Section D*, 54:750 – 756.

Cowtan, K. (2002). The Clipper Project. http://www.ysbl.york.ac.uk/~cowtan/clipper/doc/.

Cowtan, K. (2006). The Buccaneer software for automated model building. 1. tracing protrin chains. *Acta Crystallographica*, D62:1002–1011.

Coxeter, H. S. M. (1973). *Regular Polytypes*. Dover Publications, New York, third edition.

Crowther, R. A. (1972). The molecular replacement method. In Rossmann, M. G., editor, *A Collection of Papers on the Use of Non-Crystallographic Symmetry*, pages 173–178. Gordon and Breach, New York.

Crowther, R. A., Amos, L. A., Finch, J. T., Rosier, D. J. D., and Klug, A. (1970). Three dimensional reconstruction of spherical viruses by fourier synthesis from electron micrographs. *Nature*, 226:421–425.

Crowther, R. A. and Blow, D. M. (1967). A method of positioning a known molecule in an unknown crystal structure. *Acta Crystallographica*, 23:544 – 548.

Csaba, G., Birzele, F., and Zimmer, R. (2009). Systematic comparison of SCOP and CATH: a new gold standard for protein structure analysis. *BMC Structural Biology*, 9(23).

Curtis, W. D., Janin, A. L., and Zikan, K. (1993). A note on averaging rotations. In *Proceedings of IEEE Virtual Reality Annual Symposium*.

Daras, P., Zarpalas, D., Axenopoulos, A., Tzovaras, D., and Strintzis, M. G. (2006). Three-dimensional shape-structure comparison method for protein classification. *IEEE Transactions on computational biology and bioinformatics*, 3(3):193 – 207.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39(1):49–58.

DiMaio, F. P., Soni, A. B., Phillips, G. N., and Shavlik, J. W. (2009). Spherical-harmonic decomposition for molecular recognition in electron-density maps. *International Journal of Data Bioinformatics*, 3(2):205–227.

Dongarra, J., Demmel, J., and Ostrouchov, S. (1992). LAPACK: A linear algebra library for high-performance computers. In Dodge, Y. and Whittaker, J., editors, *Computational Statistics*, pages 23–28. Physica, Heidelberg.

Dubochet, J., Adrian, M., Chang, J. J., Homo, J. C., Lepault, J., McDowall, A. W., and Schulz, P. (1988). Cryo-electron microscopy of vitrified speciments. *Quaterly Review of Biophysics*, 21(2):129–228.

Egerton, R. F., Li, P., and Malac, M. (2004). Radiation damage in the TEM and SEM. *Micron*, 35(6):399–409.

El-Kareh, A. B. and El-Kareh, J. C. J. (1970). *Electron beams, lenses and optics*. Academic Press, Orlando San Diego New York London Toronto Montreal Sydney Tokyo.

Emsley, P., Lohkamp, B., Scott, W. G., and Cowtan, K. (2010). Features and development of Coot. *Acta Crystallographica*, D66:486–501.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In Simoudis, E., Han, J., and Fayyad, U. M., editors, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231.

Eswar, N., Webb, B., Marti-Renom, M. A., Madhusudhan, M. S., Eramian, D., Shen, M. Y., Pieper, U., and Sali, A. (2006). *Comparative protein structure modelling using Modeller*, chapter 5. Wiley.

Euler, L. (1776). Novi commentarii academiae scientiarum petropolitanae. `http://eulerarchive.maa.org//docs/originals/E478.pdf` [Accessed on 30 Jan 2018].

Finn, R. D., Bateman, A., Clements, J., Coggill, P., Eberhardt, R. Y., Eddy, S. R., Heger, A., Hetherington, K., Holm, L., Mistry, J., Sonnhammer, E. L., Tate, J., and Punta, M. (2014). Pfam: the protein families database. *Nucleic Acids Research*, 42:222–230.

Fox, N. K., Brenner, S. E., and Chandonia, J.-M. (2014). SCOPe: Structural classification of proteins - extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Research*, 42(1):304–309.

Frank, J. (2006). *Three-Dimensional Electron Microscopy of Macromolecular Assemblies*. Oxford University Press.

Freeden, W. and Gutting, M. (2013). *Special Functions in Mathematical (Geo-)Physics*. Birkhauser, Basel Hedelberg New York Dordrecht London.

Frigo, M. (1999). A Fast Fourier Transform Compiler. In *Proceedings of ACM SIGPLAN Conference on Programming Language Design and Implementation*.

Frigo, M. and Johnson, S. G. (2005). The design and implementation of FFTW3. *IEEE*, 93(2):216–231.

Gildea, R. J., Waterman, D. G., Parkhurst, J. M., Axford, D., Sutton, G., Stuart, D. I., Sauter, N. K., Evans, G., and Winter, G. (2014). New methods for indexing multi-lattice diffraction data. *Acta Crystallographica Section D*, 70:2652–2666.

Glaeser, R. M., Cosslett, V. E., and Valdre, U. (1971). Low temperature electron microscopy: Radiation damage in crystalline biological materials. *Journal of Microscopy*, 12:133–138.

GNU Organisaiton (2018). Argument Syntax (The GNU C Library). `https://www.gnu.org/software/libc/manual/html_node/Argument-Syntax.html`.

Grandison, S., Roberts, C., and Morris, R. J. (2009). The application of 3D Zernike moments for the description of "model-free" molecular structure, functional motion, and structural reliability. *Journal of Computational Biology*, 16(3):487–500.

Gunasekaran, P., Grandison, S., Cowtan, K., Mak, L., Lawson, D. M., and Morris, R. J. (2009). Ligand electron density shape reconstruction using 3d zernike descriptors. In Kadirkamanathan, V., Sanguinetti, G., Girolami, M., Niranjan, M., and Noirel, J., editors, *Lecture Notes in Computer Science*, pages 125–136. Springer-Verlag, Berlin Heidelberg.

Hanley, J. A. and McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36.

Hartuv, E. and Shamir, R. (2000). A clustering algorithm based on graph connectivity. *Information Processing Letters*, 76(4-6):175–181.

Hasegawa, H. and Holm, L. (2009). Advances and pitfalls of protein structural alignment. *Current Opinion in Structural Biology*, 19:341–348.

Hauptman, H. (1986). The direct methods of X-ray crystallography. *Science*, 233(4760):176–183.

Heger, A. and Holm, L. (2003). Exhaustive enumeration of protein families. *Journal of Molecular Biology*, 328(3):749–767.

Henderson, R. (1990). Cryo-protection of protein crystals against radiation damage in electron and X-ray diffraction. *Proceedings of Royal Society London*, 241:6–8.

Hendrickson, W. and Ogata, C. (1997). Phase determination from multiwavelength anomalous diffraction measurements. *Methods in Enzymology*, 276:494–523.

Hendrickson, W. A. (2014). Anomalous diffraction in crystallographic phase evaluation. *Quaterly Review of Biophysics*, 47(1):49–93.

Hermann, C. (1928). Zur systematischen strukturtheorie i. eine neue raumgruppensymbolik. *Zeitschrift fur Kristallographie - Crystalline Materialsr Kristallographie*, 68:257–287.

Hoenger, A. and Bouchet-Marquis, C. (2011). Cellular tomography. *Advanced Protein Chemistry and Structural Biology*, 82:67–90.

Holm, L. and Rosenstrom, P. (2010). Dali server: conservation mapping in 3D. *Nucleic Acids Research*, 38:545–549.

Holm, L. and Sander, C. (1998). Dictionary of recurrent domains in protein structures. *Proteins*, 33(1):88–96.

International Organization for Standardization (2011). Iso/iec 14882:2011. `https://www.iso.org/standard/50372.html`.

International Union of Crystallography (1962). 50 years of x-ray diffraction. https://www.iucr.org/publ/50yearsofxraydiffraction/full-text/crystallography.

Ioerger, T. R. and Sacchettini, J. C. (2003). Textal system: Artificial intelligence techniques for automated protein model building. *Methods in Enzymology*, 374(12):244–270.

Jiang, W., Baker, M. L., Ludtke, S. J., and Chiu, W. (2001). Bridging the information gap: computational tools for the intermediate resolution structure interpretation. *Journal of Molecular Biology*, 308(5):1033–1044.

Johnson, I., Bergamaschi, A., Buitenhuis, J., Dinapoli, R., Greiffenberg, D., Henrich, B., Ikonen, T., Meier, G., Menzel, A., Mozzanica, A., Radicci, V., Satapathy, D. K., Schmitt, B., and Shi, X. (2012). Capturing dynamics with Eiger, a fast-framing X-ray detector. *Journal of Synchrotron Radiation*, 19(6):1001–1005.

Jones, T. A., Zou, J. Y., Cowtan, S. W., and Kjeldgaard, M. (1991). Improved methods for building protein models in electron density maps and the location of errors in these models. *Acta Crystallographica*, A47:110–119.

Kabsch, W. (1976). A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica*, A(32):922–923.

Kabsch, W. (1988). Automatic indexing of rotation diffraction patterns. *Journal of Applied Crystallography*, 21:67–72.

Kabsch, W. (1993). Automatic processing of rotation diffraction data from crystals of initially unknown symmetry and cell constants. *Journal of Applied Crystallography*, 26:795–800.

Kazhdan, M., Funkhouser, T., and Rusinkiewicz, S. (2003). Rotation invariant spherical harmonic representation of 3D shape descriptors. In Kobbelt, L., Schroder, P., and Hoppe, H., editors, *Eurographics Symposium on Geometry Processings*.

Kleywegt, G. J. and Read, R. J. (1997). Not your average density. *Structure*, 5(12):1557 – 1569.

Kohonen, T. (1982). Self-organised formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–60.

Korenev, B. G. (2002). *Bessel Functions and Their Applications*. Chapman & Hall/CRC.

Kosloff, M. and Kolodny, R. (2008). Sequence-similar, structure-dissimilar protein pairs in the PDB. *Proteins*, 71(2):891–902.

Kostelec, P. J. and Rockmore, D. N. (2003). SOFT: SO(3) Fourier Transforms. Technical report, Department of Mathematics; Dartmouth College, Hanover HN 03755.

Kostelec, P. J. and Rockmore, D. N. (2007). *SOFT: SO(3) Fourier Transforms*. Department of Mathematics, Dartmouth College, Hanover HN 03755.

Kostelec, P. J. and Rockmore, D. N. (2008). FFTs on the rotation group. *Journal of Fourier Analysis and Applications*, 14(2):145–179.

Kourkoutis, L. F., Plitzko, J. M., and Baumeister, W. (2012). Electron microscopy of biological materials at the nanometer scale. *Annual Review of Materials Research*, 42(1):33–58.

Kovalevskiy, O., Nicholls, R. A., Long, F., Carlon, A., and Murshudov, G. N. (2018). Overview of refinement procedures within REFMAC5: utilizing data from different sources. *Acta Crystallographica Section D*, 74(3):215–227.

Kraft, P., Bergamaschi, A., Bronnimann, C., Dinapoli, R., Eikenberry, E. F., Graafsma, H., Henrich, B., Johnson, I., Kobas, M., Mozzanica, A., Schleputz, C. M., and Schitt, B. (2009). Characteristation and calibration of PILATUS detectors. *IEEE Transactions on Nuclear Science*, 56(3):758–764.

Krissinel, E. and Henrick, K. (2004). Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions. *Acta Crystallographica Section D*, 60:2256–2268.

Krissinel, E. and Henrick, K. (2014). Protein structure comparison service PDBeFold at European Bioinformatics Institute. `http://www.ebi.ac.uk/msd-srv/ssm/`.

Krissinel, E. B., Winn, M. D., Ballard, C. C., Ashton, A. W., Patel, P., Potterton, E. A., McNicholas, S. J., Cowtan, K. D., and Emsley, P. (2004). The new CCP4 coordinate library as a toolkit for the design of coordinate-related applications in protein crystallography. *Acta Crystallographica Section D*, 60(12):2250–2255.

Kuhlbrandt, W. (2014). Cryo-EM enters a new era. *eLife*, 3:e03678.

Langlois, R., Pallesen, J., Ash, J. T., Ho, D. N., Rubinstein, J. L., and Frank, J. (2014). Automated particle picking for low-contrast macromolecules in cryo-electron microscopy. *Journal of Structural Biology*, 186(1):1–7.

Lawson, C. L., Patwardhan, A., Baker, M. L., Hryc, C., Garcia, E. S., Hudson, B. P., Lagerstedt, I., Ludtke, S. J., Pintilie, G., Sala, R., Westbrook, J. D., Berman, H. M., Kleywegt, G. J., and Chiu, W. (2015). EMDataBank unified data resource for 3DEM. *Nucleic Acids Research*, D1:396–403.

Lesk, A. M. and Chothia, C. (1980). How different amino acid sequences determine similar protein structures: The structure and evolutionary dynamics of the globins. *Journal of Molecular Biology*, 136:225–270.

Leslie, A. G. W. (2006). The integration of macromolecular diffraction data. *Acta Crystallographica Section D*, 62(1):48–57.

Lindeberg, T. (1991). *Discrete Scale-Space Theory and the Scale-Space Primal Sketch*. PhD thesis, Swedish Royal Institute of Technology, S-100 44 Stockholm.

Liu, H., Hexemer, A., and Zwart, P. H. (2012). The Small Angle Scattering ToolBox (SASTBX): an open-source software for biomolecular small-angle scattering. *Journal of Applied Crystallography*, 45:587–593.

Liu, H., Morris, R. J., Hexemer, A., Grandison, S., and Zwart, P. H. (2011). Computation of small-angle scattering profiles with three-dimensional zernike polynomials. *Acta Crystallographica*, A68:278–285.

Long, F., Vagin, A. A., Young, P., and Murshudov, G. N. (2008). BALBES: a molecular-replacement pipeline. *Acta Crystallographica Section D*, 64:125–132.

Ludtke, S. J., Baldwin, P. R., and Chiu, W. (1999). EMAN: semiautomated software for high-resolution single-particle reconstructions. *Journal of Structural Biology*, 128:82–97.

Lyumkis, D., Brilot, A. F., Theobald, D. L., and Grigorieff, N. (2013). Likelihood-based classification of cryo-EM images using FREEALIGN. *Journal of Structural Biology*, 183:377–388.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1.

Marchler-Bauer, A., Bo, Y., Han, L., He, J., Lanczycki, C. J., Lu, S., Chitsaz, F., Derbyshire, M. K., Geer, R. C., Gonzales, N. R., Gwadz, M., Hurwitz, D. I., Lu, F., Marchler, G. H., Song, J. S., Thanki, N., Wang, Z., Yamashita, R. A., Zhang, D., Zheng, C., Geer, L. Y., and Bryant, S. H. (2017). CDD/SPARCLE: functional classification of proteins via subfamily domain architectures. *Nucleic Acids Research*, 45(D1):200–203.

Martin, K. and Hoffman, B. (2007). An open source approach to developing software in a small organisation. In *IEEE Software*, volume 24.

Martinet, A., Soler, C., Holzschuch, N., and Sillion, F. X. (2006). Accurate detection of symmetries in 3D shapes. *ACM Transactions on Graphics*, 25(2):439–464.

Mauguin, C. (1931). Sur le symbolisme des groupes de repetition ou de symetrie des assemblages cristallins. *Zeitschrift fur Kristallographie - Crystalline Materials*, 76:542–558.

McCoy, A. J., Grosse-Kunstleve, R. W., Adams, P. D., Winn, M. D., Storoni, L. C., and Read, R. J. (2007). Phaser crystallographic software. *Journal of Applied Crystallography*, 40(4):658–674.

McPherson, A. (2004). Introduction to protein crystallization. *Methods*, 34(3):254–265.

Merritt, E. A. (2011). Breaking Friedel's Law. `http://skuld.bmsc.washington.edu/scatter/AS_Friedel.html`.

Milne, J. L. S., Borgnia, M. J., Bartesaghi, A., Tran, E. E. H., Earl, L. A., Schauder, D. M., Lengyel, J., Pierson, J., Patwardhan, A., and Subramaniam, S. (2013). Cryo-electron microscopy: A primer for the non-microscopist. *FEBS*, 280(1):28–45.

Montgomery, M. G., Coker, A. R., Taylor, I. A., and Wood, S. P. (2010). Assembly of a 20nm protein cage by escherichia coli 2-hydropentadienoic acid hydratase (Mhpd). *Journal of Molecular Biology*, 396:1379 – 1391.

Morawiec, A. (2004). *Orientations and Rotations: Computations in Crystallographic Textures*. Springer-Verlag Berlin Heidelberg New York.

Morris, R. J., Najmanovich, R. J., Kahraman, A., and Thornton, J. M. (2005). Real spherical harmonic expansion coefficients as 3D shape descriptors for protein binding pocket and ligand compaisons. *Bioinformatics*, 21(10):2347–2355.

Muscariello, L., Rosso, F., Marino, G., Giordano, A., Barbarisi, M., Cafiero, G., and Barbarisi, A. (2005). A critical overview of ESEM application in the biological field. *Journal of Cellular Physiology*, 205:328–334.

Navaza, J. (1994). AMoRe: an automated package for molecular replacement. *Acta Crystallographica*, A50:157–163.

Needleman, S. and Wunsch, C. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48:443–453.

Nicholls, R. A., Fisher, M., McNichollas, S., and Murshudov, G. N. (2014). Conformation-independent structural comparison of macromolecules with ProSMART. *Acta Crystallographica*, D(70):2487–2499.

Nicholls, R. A., Tykac, M., Kovalevskiy, O., and Murshudov, G. N. (2018). Current approaches for the fitting and refinement of atomic models into cryo-EM maps using CCP-EM. *Acta Crystallographica Section D*, D74:492 – 505.

Orengo, C. A., Bray, J. E., Buchan, D. W. A., Harrison, A., Lee, D., Pearl, F. M. G., Sillitoe, I., Todd, A. E., and Thornton, J. M. (2002). The CATH protein family database: A resource for structural and functional annotatino of genomes. *Proteomics*, 2:11–21.

Organick, E. I. (1966). *A Fortran IV primer*. Addison-Wesley.

Parkhurst, J. M., Winter, G., Waterman, D. G., Fuentes-Montero, L., Gildea, R. J., Murshudov, G. N., and Evans, G. (2016). Robust background modelling in DIALS. *Journal of Applied Crystallography*, 49:1912–1921.

Patterson, A. L. (1934). A Fourier series method for the determination of the components of interatomic distances in crystals. *Physical Reviews*, 46(372).

Patterson, A. L. (1935). A direct method for the determination of the components of interatomic distances in crystals. *Zeitschrift fur Kristallographie - Crystalline Materials*, 90.

Perrakis, A., Sixma, T. K., Wilson, K. S., and Lamzin, V. S. (1997). wARP: Improvement and extension of crystallographic phases by weighted averaging of multiple-refined dummy atomic models. *Acta Crystallographica*, D53:448–455.

Peter, F. and Weyl, H. (1927). Die vollstandigkeit der primitiven darstellungen einer geschlossenen kontinuierlichen gruppe. *Mathematical Annals*, 97:737–755.

Pettersen, E. F., Goddard, T. D., Huang, C. C., Couch, G. S., Greenblatt, D. M., Meng, E. C., and Ferrin, T. E. (2004). UCSF Chimera - a visualization system for exploratory research and analysis. *Journal of Computational Chemistry*, 25(13):1605–1612.

Plaisier, J. R. and Abrahams, J. P. (2007). Cyclops: new modular software suite for cryo-EM. *Journal of Structural Biology*, 157(1):19–27.

Powell, H. R. (1999). The Rossmann Fourier autoindexing algorithm in MOSFLM. *Acta Crystallographica Section D*, 55:1690–1695.

Prakash, G. and Jhanji, V. (2016). Wavefront optics for the non-mathematician. *CRSTEurope*, pages 67–71.

RCSB PDB (2018a). PDB database statistics. `http://www.rcsb.org/pdb/statistics/holdings.do`.

RCSB PDB (2018b). PDB database statistics on number of deposited structures. `http://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=total&seqid=100`.

RCSB PDB (2018c). Redundancy in the protein data bank. `http://www.rcsb.org/pdb/statistics/clusterStatistics.do`.

RCSB PDB (2018d). Yearly growth of structures solved by electron microscopy. `http://www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=explMethod-em`.

RCSB PDB (2018e). Yearly growth of structures solved by X-ray. `www.rcsb.org/pdb/statistics/contentGrowthChart.do?content=explMethod-xray`.

Rentzsch, R. and Orengo, C. A. (2013). Protein function prediction using protein domain families. *BMC Bioinformatics*, 14.

Richardson, J. S. (1981). *The Anatomy and Taxonomy of Protein Structure*, pages 167–339. Elsevier.

Robinson, C. V., Sali, A., and Baumeister, W. (2007). The molecular sociology of the cell. *Nature*, 450:973–982.

Rodola, E., Bulo, S. R., Windheuser, T., Vestner, M., and Cremers, D. (2014). Dense non-rigid shape correspondence using random forrests. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Roeding, P., Ginn, H. M., Pakendorf, T., Sutton, G., Harlos, K., Walter, T. S., Meyer, J., Fischer, P., Duman, R., Vartiainen, I., Reime, B., Warmer, M., Brewster, A. S., Young, I. D., Michels-Clark, T., Sauter, N. K., Kotecha, A., Kelly, J., Rowlands, D. J., Sikorsky, M., Nelson, S., Damiani, D. S.,

Alonso-Mori, R., Ren, J., Fry, E. E., David, C., Stuart, D. I., Wagner, A., and Meents, A. (2017). High-speed fixed-target serial virus crystallography. *Nature Methods*, 14:805–810.

Rogen, P. and Bohr, H. (2003). A new family of global protein shape descriptors. *Mathematical Biosciences*, 182:167 – 181.

Rokach, L. and Maimon, O. (2005). *Clustering methods*, pages 321–352. Springer.

Rossmann, M. G. and Blow, D. M. (1962). The detection of sub-units within the crystallographic asymmetric unit. *Acta Crystallographica*, 15:24–31.

Rupp, B. (2010). *Biomolecular Crystallography: Principles, Practice, and Application to Structural Biology*. Garland Science.

Sael, L., Li, B., La, D., Fang, Y., Ramani, K., Rustamov, R., and Kihara, D. (2008). Fast protein tertiary structure retrieval based on global surface shape similarity. *Proteins: Structural Functional Bioinformatics*, 72(4):1259–1273.

Saha, M. and Morais, M. C. (2012). FOLD-EM: automated fold recognition in medium and low-resolution (4-15a) electron density maps. *Bioinformatics*, 28(24):3265–3273.

Sauter, N. K., Grosse-Kunstleve, R. W., and Adams, P. D. (2004). Robust indexing for automatic data collection. *Journal of Applied Crystallography*, 37:399–409.

Scheres, S. H. W. (2012). RELION: Implementation of a bayesian approach to cryo-EM structure determination. *Journal of Structural Biology*, 180(3):519–530.

Schoenflies, A. (1891). *Krystallsysteme und Krysrtallstruktur*. B. G. Teubner Verlag, Leipzig.

Schonefeld, V. (2005). Spherical harmonics. Available from `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.85.6784&rep=rep1&type=pdf`.

Schrodinger LLC (2015). The PyMOL moleculer graphics system, version 1.8. `pymol.org`.

Sharf, I., Wolf, A., and Rubin, M. B. (2010). Arithmetic and geometric solutions for average rigid-body rotation. *Mechanism of Machine Theory*, 45:1239–1251.

Shindyalov, I. and Bourne, P. (1998). Protein structure alignment by incremental combinatorial extension (CE) of optimal path. *Protein Engineering*, 11(9):739–747.

Sillitoe, I., Cuff, A., Dessailly, B., Dawson, N., Furnham, N., Lee, D., Lees, J., Lewis, T., Studer, R., Rentzsch, R., Yeats, C., Thornton, J., and Orengo, C. (2013). New functional families (FunFams) in CATH to improve the mapping of conserved functional sites to 3D structures. *Nucleic Acids Research*, D(41):490–498.

Steller, I., Bolotovsky, R., and Rossmann, M. G. (1997). An algorithm for automatic indexing of oscillation images using Fourier analysis. *Journal of Applied Crystallography*, 30:1036–1040.

Sun, L., Zhang, X., Gao, S., Rao, P. A., Padilla-Sanchez, V., Chen, Z., Sun, S., Xiang, Y., Subramaniam, S., Rao, V. B., and Rossmann, M. G. (2015). Cryo-EM structure of the bacteriophage T4 portal protein assembly at near-atomic resolution. *Nature Communications*, 6:7548–7548.

Swig Development Team (2018). Executive summary. `http://www.swig.org/exec.html`.

Tang, G., Peng, L., Baldwin, P. R., Mann, D. S., Jiang, W., Rees, I., and Ludtke, S. J. (2007). EMAN2: An extensible image processing suite for electron microscopy. *Journal of Structural Biology*, 157:38 – 46.

Taylor, G. L. (2010). Introduction to phasing. *Acta Crystallographica Section D*, 4(66):325 – 338.

Teh, C. H. and Chin, R. T. (1988). On image analysis by the method of moments. *IEEE Transactions on Pattern Analysis and Machine Learning*, 10(4):196–203.

Terwilliger, T. C. (2004). SOLVE and RESOLVE: automated structure solution, density modification and model building. *Journal of Synchrotron Radiation*, 11:49–52.

Terwilliger, T. C., Adams, P. D., Read, R. J., McCoy, A. J., Moriarty, N. W., Grosse-Kunstleve, R. W., Afonine, P. V., Zwart, P. H., and Hung, L. W. (2009). Decision-making in structure solution using Bayesian estimates of map quality: the PHENIX AutoSol wizzard. *Acta Crystallographica Section D*, 65:582–601.

Tong, L. (2001). How to take advantage of non-crystallographic symmetry in molecular replacement: 'locked' rotation and translation functions. *Acta Crystallographica Section D*, 57(10):1383 – 1389.

Tong, T. and Rossmann, M. G. (1972). The locked rotation function. *Acta Crystallographica Section A*, 46:783 – 792.

Touw, W. G., Baakman, C., Black, J., te Beek, T. A., Krieger, E., Joosten, R. P., and Vriend, G. (2015). A series of PDB related databases for everyday needs. *Nucleic Acids Research*, 43(D):364–368.

Tsukamoto, K., Yoshikawa, T., Yokata, K., Hourai, Y., and Fukui, K. (2009). The development of an affinity evaluation and prediction system by using protein-protein docking simulations and parameter tuning. *Advanced Applied Bioinformatics Chemistry*, 2:1–15.

Tukey, J. W. (1977). *Exploratory Data Analysis*. Addison-Wesley.

Vagin, A. and Teplyakov, A. (2010). Molecular replacement with MOLREP. *Acta Crystallographica Section D*, 66(1):22–25.

Vagin, A. A. and Teplyakov, A. (1997). MOLREP: an automated program for molecular replacement. *Journal of Applied Crystallography*, 30:1022–1025.

van Heel, M. (1987). Angular reconstruction: a posteriori assignment of projection direction for 3D reconstruction. *Ultramicroscopy*, 21:111–123.

van Heel, M. (2012). *Four-dimensional cryo-electron microscropy at quasi atomic resolution: IMAGIC 4D*. Wiley, Chichester, 2nd edition.

van Heesch, D. (2016). Doxygen project. `http://www.stack.nl/~dimitri/doxygen/index.html`.

Vernon-Parry, K. D. (2000). Scanning electron microscopy: an introduction. *III-Vs Review*, 13(4):40–44.

Viksten, F., Forssen, P.-E., Johansson, B., and Moe, A. (2009). Comparison of local image descriptors for full 6 degree-of-freedom pose estimation. In *IEEE International Conference on Robotics and Automation*. IEEE.

Wang, Q., Ronneberger, O., and Burkhardt, H. (2008a). *Fourier Analysis in Polar and Spherical Coordinates*. ALBERT-LUDWIGS-UNIVERSITAT FREIBURG.

Wang, X. F., Huang, D. S., Du, J. X., Xu, H., and Heutte, L. (2008b). Classification of plant leaf images with complicated background. *Applied Mathematics and Computation*, 205(2):916–926.

Wetlaufer, D. B. (1973). Nucleation, rapid folding, and globular intrachain regions in proteins. *Proceedings of National Academy of Sciences of USA*, 70(3):697–701.

White, T. A., Bartesaghi, A., Borgnia, M. J., Meyerson, J. R., de la Cruz, M. J., Bess, J. W., Nandwani, R., Hoxie, J. A., Lifson, J. D., Milne, J. L. S., and Subramaniam, S. (2010). Molecular architectures of trimeric SIV and HIV-1 envelope glycoproteins on intact viruses: strain dependent variation in quaternary structure. *PLoS Pathology*, 6:e1001249.

Wigner, E. P. (1931). *Gruppentheorie und ihre Anwendungen auf die Quantenmechanik der Atomspektren*. Braunschweig. Translated as "Group Theory and its application to Quantum Mechanics of Atomin Spectra" published by Academic Press in 1959, New York and London.

Wikipedia (2007). The proper rotations and reflection plane in the symmetry group of regular tetrahedron. `https://en.wikipedia.org/wiki/Tetrahedron#/media/File:Symmetries_of_the_tetrahedron.svg`.

Wikipedia (2018a). Cuboctahedron, made by cyp using pov-ray. `https://en.wikipedia.org/wiki/Cuboctahedron#/media/File:Cuboctahedron.jpg`.

Wikipedia (2018b). Octahedron, made by cyp using pov-ray. `https://en.wikipedia.org/wiki/Octahedron#/media/File:Octahedron.jpg`.

Winey, M., Meehl, J. B., O'Toole, E. T., and Giddings, T. H. (2014). Conventional transmission electron microscopy. *Molecular Biology of the Cell*, 25(3):319–323.

Winn, M. D., Ballard, C. C., Cowtan, K. D., Dodson, E. J., Emsley, P., Evans, P. R., Keegan, R. M., Krissinel, E. B., Leslie, A. G. W., McCoy, A., McNicholas, S. J., Murshudov, G. N., Pannu, N. S., Potterton, E. A., Powell, H. R., Read, R. J., Vagin, A. A., and Wilson, K. S. (2011). Overview of the CCP4 suite and current developments. *Acta Crystallographica Section D*, 67(4):235–242.

Wriggers, W. and Birmanns, S. (2001). Using *Situs* for flexible and rigid-body fitting of multiresolution single-molecule data. *Journal of Structural Biology*, 133(2):193–202.

Wu, D., Cui, F., Jernigan, R., and Wu, Z. (2007). PIDD: database for protein inter-atomic distance distributions. *Nucleic Acids Research*, 35:202–207.

Xiao, Y. and Yang, G. (2017). A fast method for particle picking in cryo-electron micrographs based on fast R-CNN. *AIP Conference Proceedings*.

Yang, C., Pflugrath, J. W., Courville, D. A., Stence, C. N., and Ferrara, J. D. (2003). Away from the edge: SAD phasing from the sulfur anomalous signal measured in-house with chromium radiation. *Acta Crystallographica Section D*, 59(11):1943–1957.

Yang, Z., Fang, J., Chittuluru, J., Asturias, F. J., and Penczek, P. A. (2012). Iterative stable alignment and clustering of 2D transmission electron microscope images. *Structure*, 20(2):237–247.

Ye, Y. and Godzik, A. (2003). Flexible structure alignment by chaining aligned fragment pairs allowing twists. *Bioinformatics*, 19:246–255.

Yotter, R. A., Dahnke, R., Thompson, P. M., and Gaser, C. (2011). Topological correction of brain surface meshes using spherical harmonics. *Human Brain Mapping*, 32(7):1109–1124.

Young, P. (2009). Helmholtz's and Laplace's Equations in Spherical Polar Coordinates: Spherical harmonics and spherical Bessel functions. `http://physics.ucsc.edu/~peter/116C/helm_sp.pdf`.

Zernike, F. (1934). Beugungstheorie des schneidenverfahrens und seiner verbesserten form, der phasenkontrastmethode *in German*[*Diffraction theory of the cut procedure and its improved form, the phase contrast method*]. *Physica*, 1(8):689–704.

Zettl, A. (2005). *Sturm-Liouville Theory*, volume 1. American Mathematical Society.

Zheng, H., Cooper, D. R., Porebski, P. J., Shavalin, I. G., Handing, K. B., and Minor, W. (2017). CheckMyMetal: a macromoleculer metal-binding validation tool. *Acta Crystallographica Section D*, 73(3):223–233.

Ziegler, M. (2003). Proteins: Quaternaty structure. `http://cbc.chem.arizona.edu/classes/bioc462/462a/NOTES/Protein_Structure/quatern_structure.htm`.