Next Generation Software for Placing Atoms into Electron Density Maps

Paul S. Bond

Doctor of Philosophy

University of York
Chemistry

February 2021

Abstract

X-ray crystallography is the most common method used to determine the three-dimensional structure of biological macromolecules, with proteins being of particular interest. Both the amplitudes and phases of the diffracted X-ray waves are needed to construct an electron density map, which is then interpreted by building an atomic model. However, the phases cannot be directly measured and must be estimated using either experimental phasing or molecular replacement.

BUCCANEER is a program for automatically building protein models into electron density maps, which is iterated in a pipeline with global refinement to refine the model and update the map. The amount of time-consuming manual completion required depends on the success of automated building, which may fail in difficult cases with low resolution data or poorly estimated phases. The aim of this work was to improve automated building and therefore make structure solution quicker and easier.

Potential developments to BUCCANEER were explored, but it was changes to the pipeline that proved to be most effective. The pipeline control system was updated and the following steps were added: shift-field refinement, classical density modification, addition of water and dummy atoms, pruning, and final rebuilding of side chains. The new pruning steps delete chains, residues and side chains using two neural networks, which were trained to predict main-chain and side-chain correctness by combining many validation metrics. The set of 54 experimental phasing cases previously used for testing BUCCANEER was expanded to 202 experimental phasing and 1351 molecular replacement cases. The combined pipeline changes substantially improved performance, increasing the mean completeness of the experimental phasing cases from 85% to 91% and the molecular replacement cases from 40% to 74%. The updated pipeline was released as a new program called ModelCraft.

List of Contents

List of Tables

List of Figures

Acknowledgements

Firstly, I am extremely grateful to my supervisors, Professor Kevin Cowtan and Professor Keith Wilson, for giving me the opportunity to do this project with them and for all their help and guidance along the way. I would also like to thank Professor Rod Hubbard for supervising me prior to my PhD and introducing me to computational structural biology, and Professor Fred Antson for providing support as my Independent Panel Member.

My gratitude extends to the White Rose BBSRC Doctoral Training Partnership in Mechanistic Biology for funding the project. Additionally, I would like to thank Emad Alharbi for producing results for other model building programs and Dr Melanie Vollmar and Dr Navraj Pannu for sharing test data.

I would like to thank to CCP4 for inviting me to teach at multiple crystallography schools. They were incredible opportunities to learn from both speakers and students. My thanks as well to numerous members of the crystallographic community for insightful discussions at the CCP4 Study Weekend and CCP4 Annual Review.

Special thanks to my parents for supporting me and always believing in me; my family, friends and colleagues for the much-needed laughter; Yvette for her support and patience and Loki for the ball.

Declaration

I declare that this thesis is a presentation of original work and I am the sole author, with the exception of the published or collaborative work listed below.

- Bond, P. S. *et al.* Predicting protein model correctness in *Coot* using machine learning. *Acta Crystallographica Section D* **76,** 713–723 (2020) was published as part of this PhD project and is reproduced in Sections 3.1.1 to 3.1.8.
- The supplementary material 'Automatic creation of molecular-replacement test sets' from Bond, P. S. *et al.* Predicting protein model correctness in *Coot* using machine learning. *Acta Crystallographica Section D* **76,** 713–723 (2020) is reproduced in Sections 2.3.1 to 2.3.7.
- The ARP/wARP and PHENIX AutoBuild results in Section 4.2 were provided by Emad Alharbi at the University of York.
- A total of 221 experimental phasing datasets from the Joint Centre for Structural Genomics were provided by Navraj Pannu at Leiden University.
- A total of 180 molecular replacement datasets were provided by Melanie Vollmar at Diamond Light Source.
- Figure 1.1 is derived from Cowtan, K. *Automated phase improvement and model building with Parrot and Buccaneer* Presentation. `https://www.ccp4.ac.uk/schools/APS-2011/tutorials/buccaneer/Pannu.parrot_buccaneer_aps_11.ppt`. Accessed: 2021-02-19.
- Figure 1.2 is reproduced from Cowtan, K. *Model Building* Presentation. Icknield Workshop: Model Building and Refinement for High Resolution EM Maps. 2017.
- Figure 1.3 is reproduced from Perrakis, A. *et al. ARP/wARP* and molecular replacement. *Acta Crystallographica Section D* **57,** 1445–1450 (2001).
- Figure 1.4 is reproduced from Terwilliger, T. C. *et al.* Iterative model building, structure refinement and density modification with the *PHENIX AutoBuild* wizard. *Acta Crystallographica Section D* **64,** 61–69 (2008).

- Figures 5.15 and 5.16 are reproduced from Cowtan, K. Completion of autobuilt protein models using a database of protein fragments. *Acta Crystallographica Section D* **68,** 328–335 (2012).

This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as References.

Chapter 1

Introduction

1.1 Protein Crystallography

X-ray crystallography is a technique used to determine the structure of molecules. X-rays can be generated with wavelengths of a similar order to bond lengths, and so can be used to elucidate atomic structure. Electrons in the molecule scatter X-rays through a process known as Thomson scattering, where the scattering is elastic, i.e. the scattered beam has the same wavelength as the incident beam. When the beam is scattered by a molecule it produces an image in reciprocal space but, unlike visible light in an optical microscope, X-rays cannot be refracted by a lens to invert the image into real space, so the scattering pattern must be detected in reciprocal space. Scattering from a single molecule is too weak, so a single crystal form is used to amplify the signal. Because of the repeating crystal lattice, the scattered waves interfere constructively or destructively at different angles, leading to strong spots in the diffraction pattern at the angles where the waves interfere constructively according to Bragg's law [7]:

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

where d is the distance between planes in the crystal, λ is the X-ray wavelength and n is a positive integer. For a three-dimensional crystal, there are three Miller indices (h, k and l) that define a set of planes with a certain orientation and spacing. The amplitude and phase of the wave reflected from each plane give information about the electron density in the crystal. The measured intensity of each reflection can be used

to derive the amplitude, but the phase cannot be directly measured. Instead, the phases must be independently derived to produce an initial electron density map. An atomic model is then built, which can be assessed using two methods: the agreement between the calculated and observed diffraction patterns, and the similarity to other solved structures.

X-ray crystallography can be used to solve structures of both small molecules and macromolecules such as proteins. Crystals of small molecules are more ordered and diffract well at high resolution. Additionally, there are fewer atoms in the model so the observation to parameter ratio is much higher than in protein crystallography. This allows the phases to be routinely obtained using direct methods, i.e. using phase relationships between reflections and the knowledge that the resulting electron density should be positive and show discrete spherical atoms. Protein crystals have more disorder, more atoms and more solvent, so they diffract less well at high resolution and other techniques must be used to solve the phase problem. Figure 1.1 shows an overview of the structure solution process for protein crystallography.

Once crystals have been obtained, the first step is to collect the diffraction data. Many laboratories have in-house X-ray tube sources, which produce X-rays by accelerating electrons towards a metal anode, commonly made from copper or molybdenum. Much more intense X-rays are produced at national synchrotron facilities, which accelerate electrons to high speeds in a ring using magnets. Often, crystals are tested on a home source to check their identity and quality before they are sent to a synchrotron for the best possible data collection. The X-ray beam damages the crystal, so a collection strategy must be decided on prior to the experiment. Many factors need to be considered, including beam size, crystal centring, detector distance, rotation angles, beam intensity, exposure time and wavelength.

Next is data processing in order to obtain accurate intensities for each of the reflections, along with accurate uncertainties of the intensities. The first step is indexing to determine the Miller indices for each spot and the unit cell parameters, which can be refined by minimising the difference between the predicted and observed spot positions. Then the pixels that correspond to the reflections are integrated and the surrounding background level is subtracted. Symmetry can be

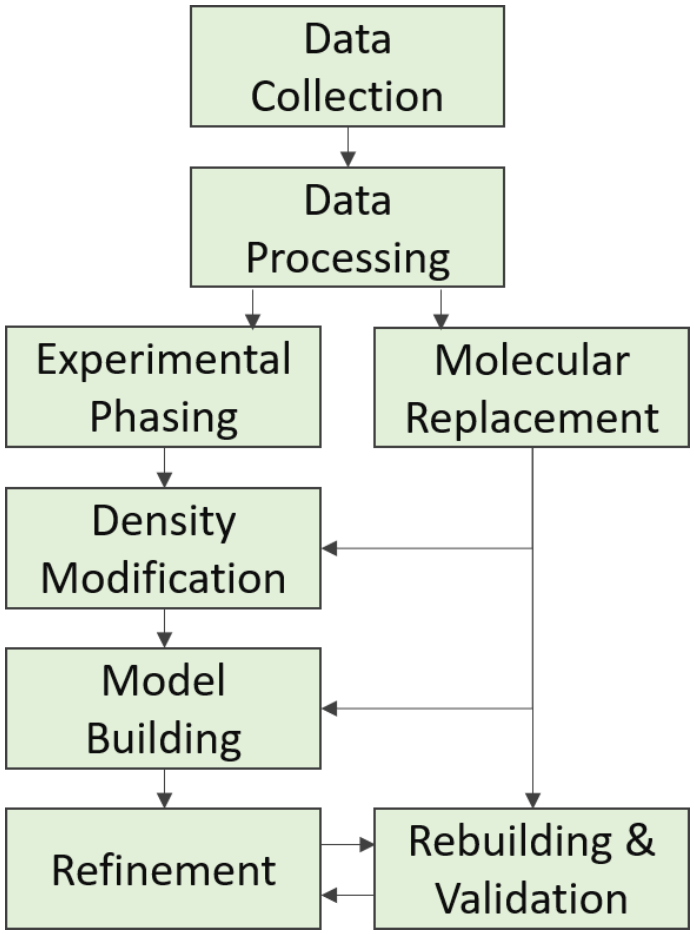

Figure 1.1: A general structure solution pipeline for protein crystallography. Figure derived from a diagram in a presentation by K. Cowtan [2]

determined from the angles of the unit cell and by looking at the correlation between potentially equivalent reflections. Once the symmetry is chosen, the integrated intensities are scaled to account for differences in the path through the crystal, radiation damage, etc., and the intensities of the equivalent reflections are merged.

As previously mentioned, the phases cannot be measured experimentally so initial estimates are needed. There are two general methods for solving this problem: experimental phasing, which uses data available from the experiment itself, and molecular replacement, which uses similar structures that are already solved. A popular experimental method in the early days of crystallography was isomorphous replacement, which requires two crystals: one of the native protein and one of a heavy atom derivative. The crystal could be soaked in a solution of heavy atoms,

and if the two crystals are isomorphous, then the differences in the reflection amplitudes can be used to locate the heavy atoms and thus provide some correct phase information as a starting point. Unfortunately, the addition of heavy atoms may causes a change in cell parameters or rotation/translation of the protein structure to result in non-isomorphism. With one derivative the technique is called single isomorphous replacement (SIR) and with multiple derivatives it is multiple isomorphous replacement (MIR).

These days, a more common experimental phasing method is anomalous dispersion, which only requires a single crystal. Anomalous scattering is wavelength dependent and occurs when the energy of the X-rays is close to the absorption edge of an atom, i.e. the energy required to promote a core electron. Anomalous scattering is stronger in heavy atoms. If none are already present in the crystal then a selenomethionine derivative is usually produced. Friedel's law states that opposite reflections should have the same amplitude but the reverse phase. However, anomalous scattering occurs at 90° to normal scattering, and this creates differences in amplitude between Friedel pairs of reflections that, like differences between crystals in MIR, can be used to find positions of the heavy atoms. A fluorescence scan can quantify the anomalous signal at different wavelengths. Single-wavelength anomalous dispersion (SAD) uses one wavelength, often with maximum anomalous signal, and multi-wavelength anomalous dispersion (MAD) uses multiple sweeps of the same crystal at different wavelengths, though each one needs to be less intense to minimise radiation damage. Anomalous scattering can also be exploited in SIR and MIR experiments, which are then called SIRAS and MIRAS.

The most common method for solving the phase problem is molecular replacement. This is where the structure of a similar protein is rotated and translated to search for positions in the asymmetric unit where it overlaps with the unknown structure, and hence shows agreement with the observed data. Exploring rotations and translations together in a six-dimensional search is computationally expensive, but it is possible to perform two separate three-dimensional searches by searching for the rotation first using the Patterson map, which is translation-independent.

The success of molecular replacement depends on the similarity of the homologue to the unknown structure, where there is a trade-off between the size and the RMSD of the model. As structural similarity cannot be calculated, sequence similarity is

used to find potential homologues. Molecular replacement also depends heavily on the resolution of the data available. In most cases individual chains or domains are used as models, but at high resolution it is possible to use small fragments such as α-helices [8, 9], or with very high resolution even single atoms [10]. Quaternary assemblies may need to be used to find a correct solution with very low resolution data, especially if there are many copies in the asymmetric unit. If a homologous structure is isomorphous, i.e. has the same crystal form, then molecular replacement is not needed and the model can simply be refined against the new data. This is common when solving mutant or ligand-bound structures.

A problem with molecular replacement is that it introduces model bias. The structure factor phases provide more information about the electron density than the amplitudes, so using phases from an incorrect model causes the incorrect features to appear in the map. If only part of the model is incorrect, e.g. a loop or a small domain, then it can be removed and rebuilt after refinement. Automated model building will usually fix this problem. Other times however, the molecular replacement solution may be completely incorrect, even when refinement reduces the R-factor below 50% and the model fits the map. These cases might be identified through poor molecular replacement scores, implausible crystal packing, noisy solvent regions, and the inability of automated model building to improve the model. Another good test is to delete a small part of the model. After further refinement, a correct solution should show positive difference density for the missing region.

The phase estimates obtained from the initial phasing step may be poor, so density modification can be used to improve the phases and produce a map that is more easily interpreted. Classical density modification works by altering the map in real space to ensure the solvent regions are flat, the protein regions have the expected distribution of high and low density, and NCS related regions are consistent. Amplitudes and phases for the modified map are then used to update the original phases. Statistical density modification is a different method that works by creating a density probability distribution for each point in the map. These are transformed into a probability distribution for a structure factor in reciprocal space, which is used to update the phase for that structure factor. Density modification is especially effective in the case of SAD experimental phasing, which

produces bimodal phase probability distributions. It can also be helpful for low-similarity molecular replacement solutions, although a large number of molecular replacement cases will be able to skip this step.

The next step is to build a model that both fits the density map and has expected geometry based on prior knowledge from high resolution structures, although the relative importance of these two factors depends on the resolution and quality of the map. Model building is the focus of this thesis and methods are discussed in detail in the next section. In short, a number of programs are available for automated model building that iteratively combine model building with refinement, which attempts to minimise the difference between the amplitudes computed from the atomic model and the experimental amplitudes and produces an updated map. The data at the start of model building differs depending on the phasing method used. Phases from molecular replacement will have model bias, while experimental phases are less biased and can be used as phase restraints in refinement. In addition, there is either a heavy atom model or a molecular replacement model available to use as a starting point. Automated model building can be skipped if a molecular replacement model has a very high similarity, such that only a few changes are needed.

The purpose of refinement is to improve the model to best represent the data. It does this by changing the model parameters, e.g. the coordinates and B-factors of each atom, to minimise a target function that includes the fit to data as well as restraints. Most commonly, refinement refers to global reciprocal-space refinement that minimises the difference between the calculated and observed structure factor amplitudes, using programs such as REFMAC [11] or *phenix.refine* [12]. With an isotropic B-factor model, there are 4 parameters (X, Y and Z coordinates and a single B-factor) to optimise per atom and so, because proteins have many atoms, the observation to parameter ratio is low and restraints need to be used to reduce the degrees of freedom and prevent overfitting. Geometry restraints ensure the model has realistic bond lengths and angles, and B-factor restraints prevent large differences between bonded atoms that should have similar disorder. More weight needs to be given to the restraints at low resolution when there are fewer experimental X-ray observations. Refinement can also be performed in real space by altering the model to fit a map, which has the advantage of being able to refine only part of the model. Shift-field refinement is a newer refinement method that

avoids overfitting by calculating parameter shifts using large spherical regions of the map [13, 14]. Shift-field refinement is fast as it is performed at low resolution, and is most useful when large concerted changes to the model are needed.

Automated model building often reaches a point where the protein model is largely correct, but further work still needs to be done to finalise the model. Residues that were not built automatically may need to be added, along with other components such as nucleic acids, sugars, ligands and water. At high resolution, alternative conformations may be resolved. The process of improving the model involves cycles of validation, rebuilding using a program such as COOT [15] and refinement. MolProbity is a popular validation suite [16]. Validation can identify residues that fit the density poorly or have uncommon geometry, but it is the responsibility of the crystallographer to decide whether the model needs to be modified accordingly. Ideally, all parts of the model should be examined visually before deposition to the PDB [17], as some errors may be missed by validation metrics.

1.2 Model Building

The first protein structures solved were those of myoglobin [18] and haemoglobin [19]. The models built of these structures were not atomic as only low resolution data (6 Å and 5.5 Å, respectively) were used in the Fourier syntheses. Instead, areas of high electron density were used to report the tertiary structure of the proteins. A higher resolution structure of myoglobin was published soon afterwards [20]. In this case an atomic model was built into a large number of steel rods, each with coloured clips attached to represent electron density levels. Later, models were built using a device known as a Richards Box [21]. These used stacks of transparent plates with electron density contours printed on them. The plates were then optically superimposed with wire models using a half-silvered mirror.

The practice of fitting models into electron density without building a physical model started becoming common in the late 70s. Programs such as GRIP [22], FRODO [23] and BILDER [24] allowed virtual models to be built using a command interface. Initial implementations were simple replacements for manual model building, but more automated building tools were gradually introduced. One of the first such tools

was the calculation of skeletonised map connecting likely Cα positions [25]. This was incorporated into FRODO [26] and combined with a database of known fragments to produce an initial map. The program O [27] took this a step further with a workflow that included automated main chain building, side chain building and real space refinement. Other programs were available around this time such as TURBO FRODO [28], XTALVIEW [29], QUANTA [30, 31] and MAIN [32]. These later programs provided more automated tools aimed at increasing speed and reducing user bias during model building, but the building process was still mainly led by the user. A popular modern equivalent of these graphical building programs is COOT [15, 33], which contains many model building, completion and validation tools as well as providing a scripting interface. Another modern program, ISOLDE [34], uses interactive molecular dynamics simulations to build models into low resolution maps.

Today, the usual process for building an initial model is to use automated model building software instead of a graphical building package. Automated building can often produce a very good model, but additional iterations of validation, manual corrections and refinement are needed to finalise a model for deposition. Completely unsupervised workflows are also used but are more common in industry for repeated experiments on similar structures, for example during ligand screening, where very few changes are needed to the protein model. Four widely used model building programs: BUCCANEER, ARP/wARP, Phenix AutoBuild and SHELXE are discussed below in more detail.

1.2.1 BUCCANEER

A fundamental part of the BUCCANEER program is the Fast Fourier feature recognition (FFFEAR) method [35]. This uses a Fourier-based search function to locate protein fragments, which proved to be useful for lower resolution maps and larger fragments. Carrying out the search in Fourier space also has the advantage that search targets can be constructed to be applicable at varying resolutions [36]. In order to apply this to model building in BUCCANEER, a simulated electron density map for a known reference structure is produced. The map simulation ensures it has similar features, such as scale and noise level, to the map that is to be interpreted. After map simulation, the main model building cycle is iterated a

number of times specified by the user, often only two or three. This consists of ten steps:

1. Finding initial oriented Cα positions.
2. Growing each initial Cα into its own chain fragment.
3. Joining overlapping fragments into the longest possible chains.
4. Linking chains with termini in close proximity.
5. Sequencing the chains, i.e. assigning residue types.
6. Correcting any insertions or deletions identified through sequencing.
7. Filtering chains that are too short or in poor density.
8. Extending chains through superposition of NCS copies.
9. Pruning residues to resolve any clashes between chains.
10. Building side chains.

Initial Cα positions are found with an oriented electron-density likelihood function using FFFEAR [37]. The closest method previously, CAPRA [38], used an orientation independent likelihood function produced by a neural network. The search target in BUCCANEER is constructed using a 4 Å sphere about the Cα atoms in the reference structure using well conserved regions of both high and low density in the simulated map. The search function involves an exhaustive six-dimensional rotation and translation search in Fourier space. This is followed by simplex minimisation of the most likely locations, this time using a simpler summation calculation. If an existing model is passed to the finding step, the search is modified to prioritise positions away from the current model.

The new Cα positions are then grown into longer chain fragments by adding Cα atoms in both directions. Two residues are built simultaneously using an exhaustive search over allowed Ramachandran angles, with steps of 20° for the first residue and 30° for the second residue. Extensions are scored by combining the FFFEAR scores for both residues. The five best scoring pairs then undergo simplex minimisation of

the φ and ψ angles. A 'look-ahead' approach is used where only the first Cα position is taken from the best scoring pair. Growing in each direction is repeated until the score falls below a cutoff value.

The result of the previous finding and growing steps is usually a large number of chains, many of which overlap with each other. The joining step collects these consistent chain fragments together by first splitting them into overlapping three residue fragments. Fragments where all three residues overlap are merged by averaging the coordinates, then subsequent joins are chosen such that the longest possible chains are produced. After joining, the linking step tries to connect the termini of nearby chain fragments by inserting one or two residues. This step may link chains incorrectly, but errors introduced at this point can be corrected after the sequence has been assigned.

Sequence docking uses a similar FFFEAR target function to that used in the initial Cα finding step [39]. This time 20 different search targets, one for each residue type, are produced using a 5.5 Å sphere about the Cβ atom. A score is obtained for each search target at each Cβ in the model and converted to a Z-score to account for different amounts of flexibility between residue types. The known sequence is then moved along the chain to find the best scoring match. Sub-sections of the chain are tested independently to allow for the case of an incorrectly traced chain. The best scoring match is compared with the second best scoring match and if there is a significant difference the match is assumed to be correct. It is interesting that the sequence assignment works using targets that are averaged over all rotamers in the reference structure.

After sequencing, extra information is available about the correctness of the structure. Any insertions and deletions identified at this stage are rectified by rebuilding to delete or add a residue. Addition or deletion is tested at multiple positions around the error to identify the chain that fits best. Next, a relatively simple filtering step removes chains shorter than 6 residues and chains in poor density. If present, an NCS building step subsequently uses chain copies to aid building. For each chain, all matching chains are superposed with the sequences removed and used as input to the joining, sequencing, filtering, and tidying steps. The resulting chain is kept if it is longer and has more residues sequenced than the original.

The penultimate step is to prune clashing chains. Chains that can be merged or linked have already been addressed in previous steps, but there may be cases such as two chains built in the same density in opposite directions. Pruning is performed by removing residues from one of the chains. Sequenced chains and longer chains are assumed to be correct in this operation. A recent addition to BUCCANEER also assesses the fit to density of the chain when deciding which residues to remove.

Finally, carbonyl oxygens and side chains are built. Side chains are built using the Penultimate Rotamer Library [40]. Each rotamer is scored using the mean density Z-score of the γ, δ, ε, and ζ atom coordinates. Large side chains are also rotated about the Cα-Cβ bond slightly to allow for their flexibility. Clashes between pairs of residues are then corrected by testing different rotamer combinations and truncating both residues if a suitable pair cannot be found. After the model building cycle, a final tidying step groups chain fragments together by deciding which belong to the same chain.

A loop building algorithm was also written for BUCCANEER [6], which uses a protein fragment database constructed from 500 high resolution structures [41]. Wrongly traced residues are pruned away from the edges of the loop and the database is searched for fragments of the correct length that overlap with the two residues at either side. This step was initially placed in the model building loop before side chain building. However, it caused too many errors at the start of the building process when the model only consists of short fragments, so the BUCCANEER version including this loop building step was not released. Instead, the loop building method was incorporated into both COOT and a separate program, SLOOP.

BUCCANEER was first published in 2006, but at that point it only included main chain tracing using a single cycle of the finding, growing, joining, and pruning steps. This initial release was tested on 58 structures from the Joint Centre for Structural Genomics [42]. The results showed BUCCANEER to be fast and not very sensitive to resolution, but with a strong dependence on the quality of the initial phases. The BUCCANEER program itself contains no global refinement step but has been implemented in a number of pipelines, such as the one shown in Figure 1.2. Internal cycles of BUCCANEER are iterated with cycles of a refinement program such as REFMAC [43], after which the updated model and

map are passed back to BUCCANEER.

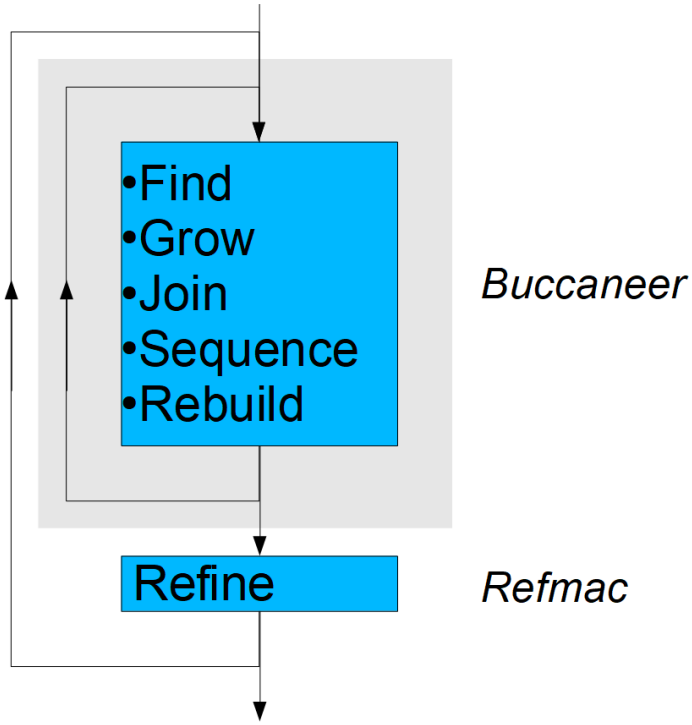

Figure 1.2: An overview of the BUCCANEER pipeline. Only some of the steps within BUCCANEER are shown. Figure reproduced from a presentation by K. Cowtan [3].

1.2.2 ARP/wARP

The origins of ARP/wARP lie in the Automated Refinement Procedure (ARP) published by Lamzin and Wilson in 1993 [44]. ARP works by refining a model as a series of free atoms. Least-squares refinement is iterated with removing atoms in low difference density and addition of new free atoms in high difference density. Once the iterations converge, as measured using R factor and high difference peaks, a new protein model is built by renaming atoms in the free atom model.

ARP was later applied in the wARP (weighted ARP) procedure used to improve the quality of the phases before a model is built [45]. Initial models are produced by iteratively adding free atoms in the highest density until 1.5, 2 and 3 times the expected number of atoms are placed. These are subsequently reduced to 1.2 times the number of expected atoms and three more models are produced by shifting

these models slightly. All six models are then refined using an external program for reciprocal space refinement, for example REFMAC, followed up by real-space refinement using ARP. Structure factors from the models are weighted by how close they are to the average and a single set of updated phases is produced from a weighted average.

ARP/wARP was the first fully automated procedure that can go from an electron density map to a complete model without user intervention [46]. The first step is to run density modification and build a free-atom model, which is refined and used in an auto-tracing step to produce a hybrid model containing some peptide chains and some free atoms. The hybrid model is given to REFMAC and ARP for refinement, and the auto-tracing step is repeated until tracing is complete.The final step is to dock the sequence and build side chains to produce a complete model. Figure 1.3, reproduced from Perrakis *et al.* [4], shows an overview of the procedure for molecular replacement. Individual steps are discussed in more detail below.

In the auto-tracing step, free atoms are assigned a score based on the atomic displacement factor and the height of the electron density. Pairs of atoms with a high score separated by 3.8 ± 0.5 Å are marked as possible successive Cα atoms. Trans peptide bonds are then built between candidate Cα pairs and are kept only if the carbonyl density is good. All possible chains are constructed from connected peptides traced in the same direction keeping the Cα-1 and Cα+1 distances within 4.6–7.8 Å. At this point branch points are resolved by eliminating the peptide in the lowest density. Any chains overlapping with the longest chain are removed, then the second longest and so on, and finally chains shorter than 5 residues are removed. Main chain tracing was later updated using more intelligent graph searching algorithms [47]. These view the tracing of likely Cα atoms as a constrained integer programming problem that maximises expected geometry in the model.

Sequence docking is performed by expressing each side chain as a vector of atom counts from Cβ onwards, e.g. leucine is 112 and phenylalanine is 11221 [48]. For each residue in the main chain, an observed vector of free atom counts is used to calculate probabilities of it being each of the 20 residue types. A vector of probabilities for the built chain is then moved along the known sequence and a score calculated for each position. A confidence score is calculated using the difference between the first

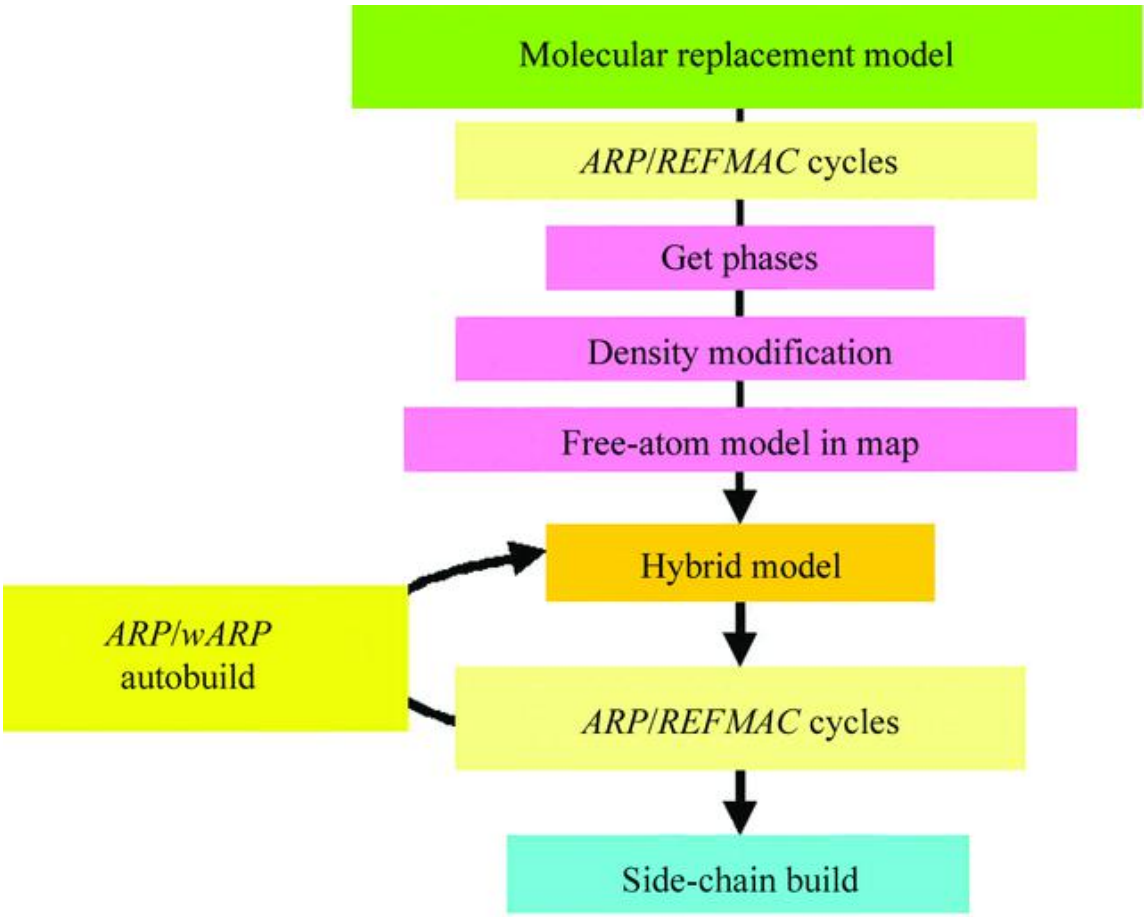

Figure 1.3: An overview of the ARP/wARP procedure. Figure reproduced from Perrakis *et al.* [4]

and second best positions. The chain with the highest confidence score is chosen and that part of the known sequence is no longer available for other chains.

As in BUCCANEER, side chain building uses the Penultimate Rotamer Library [40]. The best rotamer is built before being refined using simplex minimisation. The simplex algorithm varies the χ angles in the side chain, along with the φ angle to move the Cβ position. The scoring function changes from a quicker summation of density at atomic positions that is used initially to a slower but more accurate real-space correlation function. Clashing side chains are avoided by fitting into a 'real space residual map', which is a copy of the density map from which existing model density has been subtracted. Well-ordered residue types are placed first to give them priority, but the ideal order may vary with map quality. Multiple high-scoring conformations can be marked as alternates at high resolution.

ARP/wARP contains a loop building step that uses a database of pentapeptides from known structures [49]. A large number of possible loops conformations are built between two anchors by extending in both directions. Initial building is based on geometrical restraints from the pentapeptide database and electron density is only used to stop clashes with built residues. Loops are then filtered to only keep those with the best geometry that end close to the opposite anchor. The contribution of the density score is gradually increased to select the best loop. Finally, the loop's main chain and side chains are built from the Cα positions and the loop is refined in real-space.

Iterations in ARP/wARP differ between running the program in classic or expert mode. The classic system uses a predefined number of cycles for each of the steps. Expert mode uses a control system called FLEX-WARP that will alter the protocol based on the results of the previous step [48, 50]. There are three points where a decision must be made. Firstly, from a free atom model, whether to keep refining or to trace the main chain. Secondly, with a hybrid model, whether to keep building the main chain or move onto sequence docking and side chain building. Thirdly, after side chain chains have been built, whether to accept the model and move onto completion or use it as an input in earlier steps.

A more recent addition to ARP/wARP was to use NCS copies to help extend existing models [51]. Partially built fragments of a fixed length are superposed to see if they match. Pairs of superposed fragments with similar rotations are used to identify symmetry copies of chains. The matching fragments are then extended to contain the residues common to both chains, subject to an RMSD cutoff. Atoms beyond the common fragments are then superposed onto other copies provided they do not clash. Extensions of the model found this way are weighted by their estimated accuracy and only the top scoring extensions are used. This PNS (Protein NCS-based Structure) Extender module is turned on by default for medium to low resolution structures.

The initial implementation of warpNtrace required data to a resolution of 2.3 Å or higher [46]. This is likely owing to the reliance on unrestrained free-atom refinement causing too much inaccuracy in atomic positions at lower resolution. The improvements in Version 6, such as the Cα graph searching strategies, were shown to extend the effective resolution limit to 2.5 Å [52]. Version 7 extended this limit slightly further still and was able to build partial models at 2.7–2.8 Å,

although building still performs much better at higher resolution [53].

The latest version of ARP/wARP (8.0) is capable of building at even lower resolutions, including for cryo-EM maps at better than 4.0 Å resolution [54]. It contains a new sequencing method (SEQQY) that uses the changing volume of the side chain density at various contouring thresholds to distinguish different residue types, which gives improved performance at mid to low resolution as it does not rely on free atom positions [55]. Another change is to aid chain tracing decisions with the use of DipCheck [56], which assesses the backbone conformation using the extension, twist and bend of dipeptides, derived from an internal distance matrix. Finally, a method for improving low resolution models using homologous structures has been developed [57]. This searches for structures of sequence homologues, then superposes fragments of these models over the backbone built by ARP/wARP and merges them to create a single consistent model.

1.2.3 Phenix AutoBuild

Model building in Phenix AutoBuild relies on an earlier program called RESOLVE, which started off as a likelihood-based density modification program [58]. This approach was successful due to combining experimental information and prior knowledge about expected density distributions. Initially, the prior knowledge only included flat solvent regions and the protein region but was extended to include pattern recognition and NCS [59, 60].

RESOLVE started implementing model building as well as density modification soon after these extensions [61]. Main chain building starts by locating likely α-helices and β-strands in the density. This is done using an FFT-search of idealised hexapeptide fragments as described in a previous paper [59]. Libraries containing α-helices and β-strands of different lengths are then placed into the resulting locations, truncated, and scored by length and fit to density. The best scoring fragment for each position, provided its score is greater than a threshold value, is kept and extended using a library of tripeptide fragments. A look-ahead approach is used where two tripeptide fragments are docked and scored but only the first fragment from that extension is used. After extension, overlapping chain fragments are merged into single chains, again prioritising length and fit to density.

Sequence docking uses a custom rotamer library derived from 574 structures with a resolution of at least 1.8 Å and an R-factor of 20% or better [62]. A template was produced for each rotamer by averaging the electron density of amino acids with that conformation. At each residue in the chain, a correlation coefficient is calculated for each rotamer. The best scoring rotamer for each residue type is listed and Z-scores are determined using the mean and standard deviations of correlation coefficients. The Z-scores are translated into probabilities and combined with prior probabilities using the total number of residues of that type in the structure. Sub-fragments for the main chain are then tested against the known sequence using the previously determined probabilities. Once the sequence has been assigned, the most probable rotamer for that side chain type is built.

Iterating RESOLVE model building and density modification with REFMAC refinement led to increased performance [63]. However, the modern pipeline is provided by the Phenix AutoBuild wizard [5] in the PHENIX software suite. Figure 1.4, reproduced from Terwilliger *et al.* [5], shows an overview of the pipeline. Phenix AutoBuild combines RESOLVE model building and density modification with refinement using phenix.refine. Multiple models are built into the density-modified map using RESOLVE. Each model is scored, initially by the number of residues built, number of residues sequenced and number of chains but later using R-factors. The best model is refined using phenix.refine and all models are combined by cutting them into fragments, extending and merging. From the merged model, additional model building is performed on a masked map followed by loop building. Missing loops are found in both a sequence independent way by looking for nearby termini and also by finding short fragments missing from the built sequence. Once two termini of a possible loop are identified the chains are extended to try and make them overlap by at least one amino acid.

In addition to the standard building procedure, a model can be rebuilt such that no atoms are added or removed. The rebuilding process involves using the loop building algorithm to rebuild fragments (usually hexapeptides) along the chain to produce a new model. The new model is then combined with the old model, choosing the best fragments from each. This procedure is useful for altering the main chain at very high completion or for rebuilding very similar molecular replacement models.

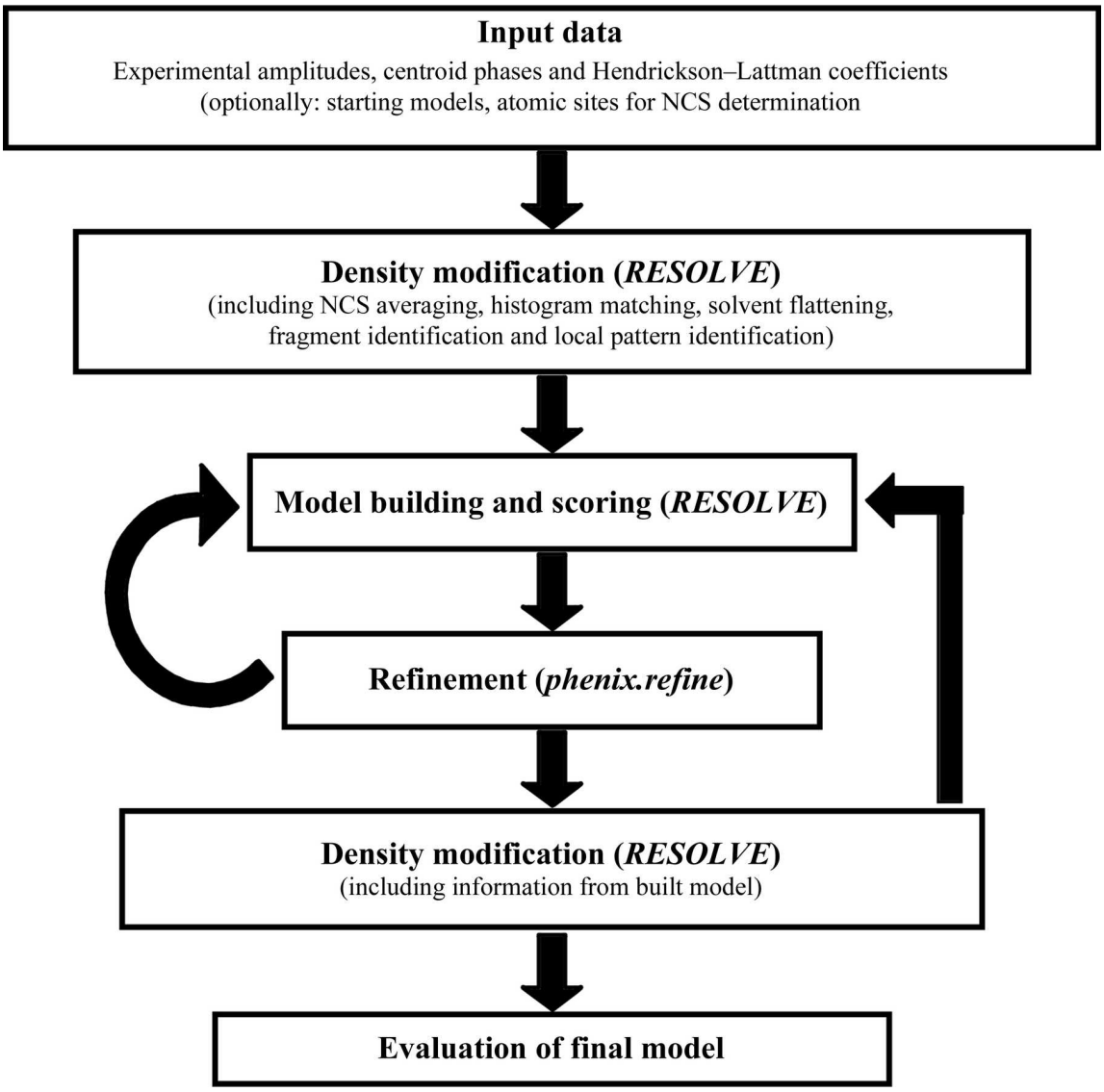

Figure 1.4: An overview of the Phenix AutoBuild pipeline. Figure reproduced from Terwilliger *et al.* [5]

1.2.4 SHELXE

The SHELXC/D/E suite is a used for experimental phasing of macromolecules. However, the SHELXE program uses protein main chain tracing for automated density modification [64]. Tracing starts by finding seven residue α-helices and common tripeptide fragments. A weighted density score is used where the weights of Cα and C are 6, N is 7, O is 8, Cβ is 4 and 'holes' around the residue are −2. Initial positions are found using a template search [65], placing carbonyl groups on

density peaks. These initial chains are extended by minimising 15 φ/ψ pairs using a two-residue look-ahead similar to the one used in BUCCANEER. A difference from the implementation in BUCCANEER is that extension into symmetry related space is not allowed. Extended chains are accepted based on their density score, length, Ramachandran outliers, secondary structure, and possible hydrogen bonding from N atoms. Overlapping chains are joined by cutting the chains at the closest point, which is assumed to be correct, and choosing the best of the options for the N and C terminal parts.

In contrast to BUCCANEER, a polyalanine chain is built without sequencing and tracing is performed cautiously to avoid errors that may deteriorate the phase quality. Although SHELXE was written for structures solved by experimental phasing, it is often used for chain tracing after molecular replacement to extend a partial model [66], for example in ARCIMBOLDO [8] and AMPLE [67].

More recent developments improve the performance of SHELXE at lower resolution and with poorer starting phases [68]. Tripeptide parallel and anti-parallel β-strands have been added to the search templates, along with longer helices of up to 14 residues. To reduce errors, extensions of helical templates using the original look-ahead approach are required to have helical geometry in the initial tracing stages when phases are poorest. In addition, helices can be extended by translating and refining an another helical template.

1.2.5 Summary

Although the four programs covered in this section have some similarities, each has a different approach to the problem of model building at its core, and because of these differences each program must possess its own strengths and weaknesses relative to the others. BUCCANEER is a very fast program that still performs well at low resolution. Although it initially searches for individual Cα positions, the target is actually a 4 Å sphere that still works at sub-atomic resolution. However, BUCCANEER has less advanced model completion steps such as loop fitting and rotamer building. ARP/wARP performs well at high resolution as its free atom model can represent the structure very accurately, including atoms with unknown identities that are missed from other programs. The downside is that classic

ARP/wARP is less likely to build an initial model into a low resolution map. Phenix AutoBuild appears to provide a more rounded pipeline as the initial search for fragments make the technique applicable to lower resolution data and the model completion steps mean less work is needed subsequently by the user. However, the number of steps involved, and the fact they are performed on multiple models, means the pipeline is much slower than others. As it was the original aim of the program, the strength of SHELXE is in its ability to build into poorly phased maps to improve the phases, especially at high resolution. The obvious downside of SHELXE is that it is not a complete model building solution as it only provides chain tracing for the backbone without side chains.

There are not many quantitative comparisons between model building programs. One was performed by van den Bedem *et al.* in 2011 using Xsolve [69]. Xsolve is a pipeline from the Joint Centre for Structural Genomics that provides automatic data processing, phase solution and model building. Combinations of MOSFLM and XDS for data processing, SOLVE and AUTOSHARP for phasing, and BUCCANEER, ARP/wARP and RESOLVE for model building were ran in parallel with default settings. Resulting traces with more than 40% of their residues docked into the sequence were combined using a program called ConsensusModeler to produce a final model for refinement. A test set of 36 structures ranging from 1.3 Å to 3 Å resolution was used to look at the pipeline in more detail, including a comparison of model building programs. BUCCANEER produced the most complete trace for 33 out of the 36 structures. It also performed similarly well at both high and low resolution. For the 12 structures below 2.5 Å resolution, ARP/wARP did not build any above 40% completion and RESOLVE showed a 23% drop-off in completion from structures above 1.8 Å resolution. It was also noted that although BUCCANEER models were more complete, they also had more wrongly built fragments than RESOLVE at low resolution.

Unfortunately, a comparison of the programs in 2011 will not be as relevant today. All of the programs undergo continuous development, not all of which is published in journal articles. A more recent comparison was performed by Alharbi *et al.* in 2019 [70]. BUCCANEER, ARP/wARP, PHENIX AutoBuild and SHELXE were tested using default settings on 148 experimentally phased datasets between 1.2 and 3.2 Å resolution. Phenix AutoBuild models had the highest completeness and

ARP/wARP models had the best R-factors. Lower resolution datasets were also produced by artificially inflating the B-factors of the structure factor amplitudes and truncating the data to 3.2–4.0 Å resolution. BUCCANEER produced the best models for this simulated low-resolution test set.

It is important to note that all the programs are highly configurable, such that an expert user can tailor them to work in a wider range of scenarios. In some cases it might be expected that the user will change the default settings to suit their data. Some customisation of settings may also be provided automatically though a graphical user interface, as this is how most users interact with the programs. For well-phased high-resolution datasets, all programs are likely to perform well, so it is less important which is used. In more difficult cases, it is advised to try every method to find the best result.

1.3 Aims and Overview

The broad aim of this thesis was to improve automated protein model building for X-ray crystallography. Based on feedback from users, BUCCANEER can perform less well than other model building programs with high resolution data and structures phased by molecular replacement, as well as producing less complete models that need more manual rebuilding (K Cowtan, 2021, personal communication). A recent study tested a number of programs on 148 experimentally phased structures and found that the BUCCANEER pipeline in CCP4i produced models with higher R-factors and more incorrect residues than ARP/wARP and PHENIX AutoBuild [70]. The work presented in this thesis focuses on addressing these weaknesses.

The first chapter is on the creation of test sets, i.e. sets of example datasets used for testing methods, because they underpin the rest of the work in the thesis. In order to assess the performance of a model building program, each test case must provide input data from either experimental phasing or molecular replacement. Experimental phases are sometimes deposited in the PDB along with the final structure, but associated heavy atom and molecular replacement models are not routinely available so the creation of model building test sets is not

straightforward. In addition, it is important to have test cases with varied data, e.g. both experimental phasing and molecular replacement examples, and covering a range of data resolutions and initial phase qualities.

Next is a chapter introducing a new approach for quantifying the 'correctness' of protein residues. Identifying poorly-built residues is a fundamental problem that occurs during model building and validation. Crystallographers use multiple validation metrics when analysing suspicious parts of a model, but when a decision needs to be made by an automated program, for example whether the backbone has been correctly traced or which side chain conformation to build, then usually only a single metric is used. BUCCANEER assesses residues with one of two scores, depending on the context: a simple mean of the density at the atomic centres or a C-alpha log-likelihood target function [37]. This chapter presents the creation and application of a new score that combines multiple sources of information through machine learning.

The penultimate chapter examines improvements to the BUCCANEER pipeline. Running BUCCANEER by itself is useful for quickly assessing a map, for example to determine the correct hand after SAD phasing, but usually a user wants the most complete model possible to minimise the amount of time spent building manually. The original pipeline used five iterations of BUCCANEER with REFMAC [11] to perform global refinement and update the map, but further testing found many molecular replacement examples where this was insufficient. The pipeline has been updated with a more intelligent control system and new steps for phase improvement and model completion using SHEETBEND [14], PARROT [71] and COOT [15].

The final chapter covers developments made internally to BUCCANEER. Changes were made to the side chain building algorithm including a new rotamer library, simplex minimisation, and a new scoring function. Additionally, a new backbone rebuilding step was introduced that rebuilds overlapping two-residue fragments along each chain. This work was performed first in the project, with the aim of increasing accuracy at high resolution through more flexible building methods, but it is presented last because the changes did not provide an immediate benefit and have not yet been released.

Chapter 2

Test Sets

Changes to BUCCANEER are assessed by running the program on a series of test cases. Looking at individual cases in detail is useful for identifying unexpected behaviour and potential weaknesses in the method, but it does not provide statistically useful information about general performance. It is likely that a change will lead to better models in some cases but worse models in others. It may be possible to predict which cases are likely to be better and which are likely to be worse in advance, in which case the new method can be applied to a subset of the cases for maximum performance. However, there may be no significant pattern to deduce and the differences may be due to noise, in which case the average difference in performance can be used to decide if a change should be accepted.

The size of the test set is important for determining the significance of performance changes. It is common for a new development to make little difference in a lot of cases. If changes are only seen in a small fraction of cases then the mean change is small and the test set needs to be large to reduce the standard error of the mean. The difficulty of the structures included in the test set is also important. If a structure can already be built completely then this only leaves room for speed improvements. At the other end of the spectrum, poor input data does not help to differentiate between methods that cannot even build a partial structure.

The difficulty of automated model building is greatly affected by resolution and phase error. Having a test set that samples a range of resolutions and phase qualities gives

the best chance that a new development will change at least some of the cases. This chapter details improvements made to the test sets used for testing BUCCANEER to increase their size and coverage of resolution and phase quality. Structures phased by experimental phasing and molecular replacement are discussed separately.

2.1 Experimental Phasing

Historically, developments to BUCCANEER have been tested on a small number of structures from the Joint Centre for Structural Genomics (JCSG) [42]. The initial BUCCANEER main chain tracing method was tested on 58 of these datasets [37]. At the start of my PhD, the number of test cases had been reduced to 54, excluding a few cases with very poor initial phases. The datasets had been solved by SAD/MAD phasing using the Xsolve automated structure solution pipeline [69]. The output from the phasing step is an MTZ file with mean structure factor amplitudes and initial phases in the form of Hendrickson-Lattman coefficients. However, multiple files may be present from phasing steps within different branches of the pipeline. If multiple phasing results were present for a structure, they were ranked using the RMSD of local map RMSD in a 4 Å radius. A map with better phases is expected to have a higher RMSD due to differences between the protein and solvent regions, where protein regions will have a high local RMSD and solvent regions will have a low local RMSD.

To prepare these datasets for use in automated model building with BUCCANEER, density modification was first performed on the experimental phases using PARROT [71]. Then the coordinate file deposited in the PDB was downloaded and the protein sequence was taken from the built model. In some cases there were differences in the cell dimensions between the MTZ file and deposited PDB file. This may be because a different branch of processing was used in the automated pipeline or because of changes made manually before deposition. Unknown ligand (UNL) residues were removed from the deposited structures and the structures were refined using 10 cycles of REFMAC [43]. The refinement uses the cell described in the reflection file and is able to proceed since the differences in the cell dimensions are small.

A larger test set was desired so that changes in performance could be measured more

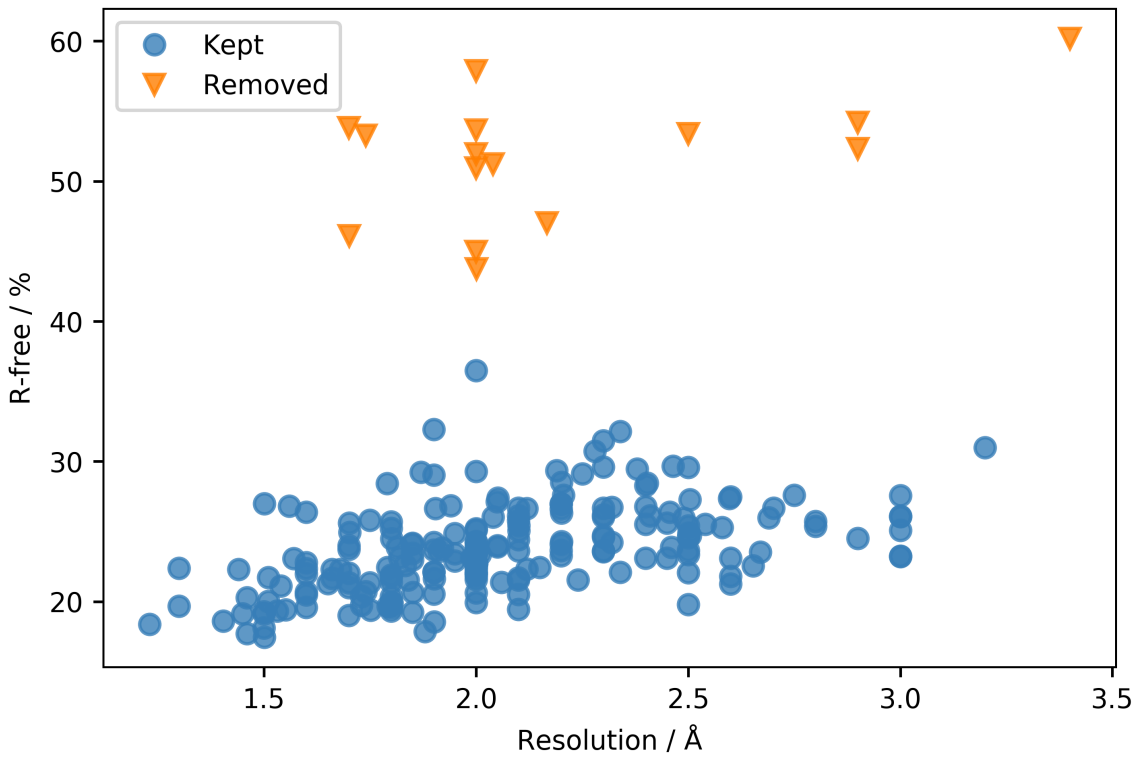

Figure 2.1: Resolution against final R-free for 217 structures from the JCSG after refining the deposited structure with the chosen MTZ from the automated phasing step. 15 structures were removed from the set due to very high R-free values.

accurately. The automated data processing results are no longer publicly available from the JCSG, but 221 datasets were provided by N. Pannu at Leiden University. There were 9 structures in our existing test set that were not present in this new set, giving a total of 230 structures available. The same procedure was used to rank phasing results by the RMSD of local map RMSD. One dataset failed at this stage due to having no readable MTZ files from phasing. The deposited structures were refined against the data using 10 cycles of REFMAC. Refinement failed for 11 structures owing to large differences between the cell definitions in the coordinate and reflection files. In some cases this was due to a difference in the space group. One further structure gave an error due to a serine residue with a residue name of UNK. Figure 2.1 shows the final R-free value for the refined structure and the high resolution limit of the data for the remaining 217 structures. A further 15 datasets were removed as the deposited structures had very high R-factors after refinement. This does not mean the deposited structures were bad quality. The R-factors may be high because the data have been processed incorrectly and are not the same

data used to solve the deposited structure, or it could be due to differences in the refinement procedure.

Figure 2.1 shows there is a need for more high and low resolution test cases. Out of the 202 structures kept in the test set, 169 (84%) are between 1.5 Å and 2.5 Å resolution. There are 6 structures at 3.0 Å and one at 3.2 Å resolution. It was decided to fill this gap at low resolution by simulating lower resolution data using the existing datasets. The first step in doing this was to artificially inflate the B-factors of the structure factor amplitudes by an amount that would be expected from the difference between the real resolution and chosen lower resolution.

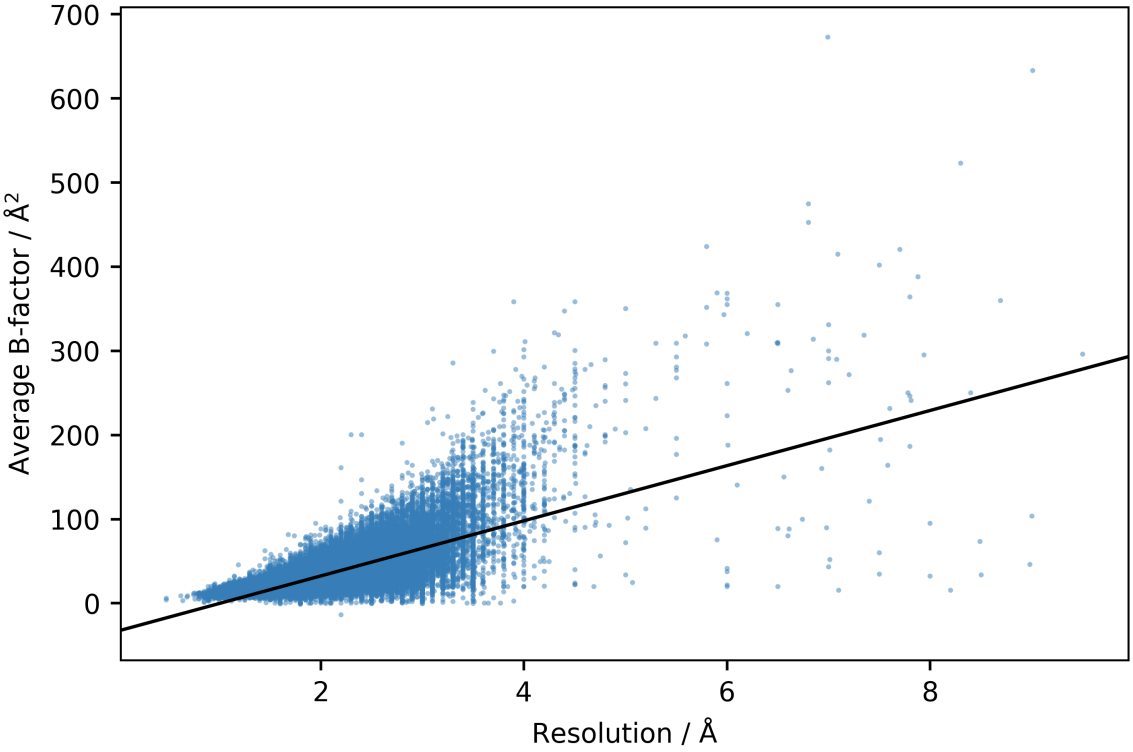

Figure 2.2: Blue points show resolution and average B-factor for 85204 X-ray structures in the PDB. The black line shows a linear fit of these points with the equation $y = 32.8x - 33.3$.

Figure 2.2 shows a linear fit of average B-factor against resolution for 85204 X-ray structures in the PDB. The fit has a low coefficient of determination (R^2) of 0.51 owing to the noisy data but is highly significant with a p-value less than 2.2×10^{-16}. The gradient of the fit is 32.8 ± 0.1 Å, so if the original resolution was 2 Å and the simulated lower resolution was 4 Å, the difference in resolution is 2 Å and the

B-factors would be expected to increase by 65.6 Å^2 (i.e. 2 Å $\times$ 32.8 Å).

After inflating the B-factors, the data were truncated to the new high resolution limit. Each of the 202 datasets were altered to 3.2, 3.4, 3.6, 3.8 and 4.0 Å resolution, apart from one dataset that was already at 3.2 Å and did not need to be altered for this resolution. Although increasing the B-factors increases the spread of the peaks in the density, it does not affect the peak positions because the phases were unchanged. In reality, larger errors in the phases would also be expected for lower resolution datasets. Additionally, reducing structure factor amplitudes reduces both the signal and the noise of the data when only the signal should be reduced.

For each dataset (i.e. the original and the simulated low resolution datasets) density modification was carried out with PARROT using three different options: once without any NCS averaging, once with the NCS operators being determined from the deposited methionine S or Se atomic positions, and once determining the NCS operators from the full deposited model. This test set was used by Alharbi *et al.* [70] for comparing different model building pipelines.

2.2 Molecular Replacement

Previously, there was no set of structures available that had been phased using molecular replacement for testing BUCCANEER other than demo datasets such as γ-Adaptin and β-Lactamase, for which search models are provided but molecular replacement must first be carried out. The first molecular replacement test set was formed from a set of 180 structures from M. Vollmar at Diamond Light Source, each with an MTZ file containing structure factor amplitudes, a sequence of the target and a PDB file of a potential homologue. I created the search model by aligning the sequences of the target and homologue and using the sequence alignment to trim and mutate the homologous chain with CHAINSAW [72]. Molecular replacement was then carried out using MOLREP [73] and the placed model was refined with REFMAC [43]. Most of the models failed to produce a correct molecular replacement solution. The following thresholds were used when deciding which cases to keep:

- CHAINSAW estimated sequence identity $> 20\%$

- MOLREP MR Score > 0.35
- MOLREP MR Z-score > 3
- REFMAC final R-free $< 0.55\%$

The MOLREP MR score is the product of the correlation coefficient and the packing function, which is 1 if no molecules overlap and -1 if molecules overlap completely. The MR Z-score, also known as contrast, is the number of standard deviations between the top MR score and the mean MR score. There were 63 structures remaining in the set after applying these thresholds, which were deliberately chosen to be lenient to keep some poor starting models that a model building program will struggle with. If all the models can be built easily then the test set has very little discriminating power to identify improvements. Figure 2.3 shows model R-free against resolution for the 63 structures remaining. The R-free values are not comparable with the values in Figure 2.1 as the latter are for a deposited structure instead of a molecular replacement model. A subset of these processed datasets was given to Burla *et al.* [74] to test an automated molecular replacement pipeline using REMO09 [75], SYNERGY [76] and CAB [77].

As this molecular replacement test set was not very large, a new test set was created from 310 randomly chosen structures in the PDB between 1 Å and 4 Å resolution. For each structure, multiple sequence homologues were identified using PHMMER [78]. Trimmed and mutated models were made for each alignment with the homologous chain using SCULPTOR [79]. However, instead of performing molecular replacement, the models were superimposed over the target using GESAMT [80]. If multiple copies of the target chain were present, the model was superimposed over each copy. Finally, the superimposed models were refined using REFMAC [43].

This led to a much larger test set, but the approach had some flaws. Although homologues were chosen so that no two homologues had a sequence identity of more than 95%, some targets ended up with thousands of models. There were more than 40,000 models between all 310 target structures, most of which were poor quality. The models were superposed over the deposited structures because it was much quicker than performing molecular replacement, but this also creates an

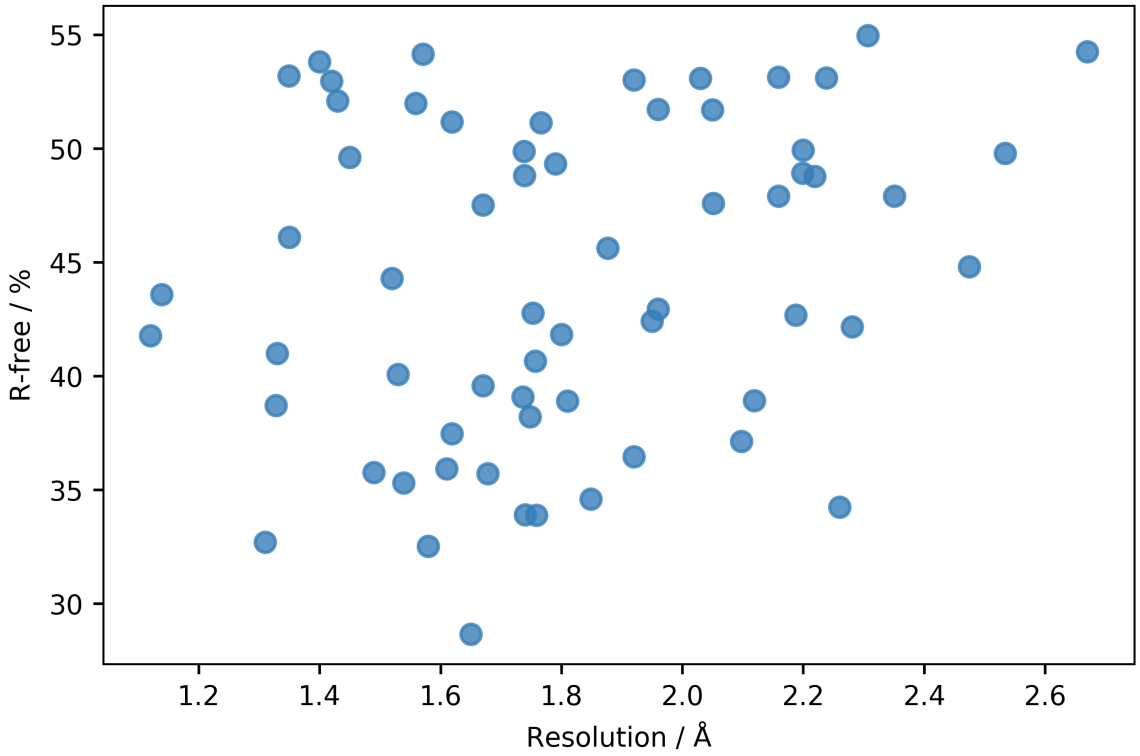

Figure 2.3: Data resolution against R-free of the refined model for 63 molecular replacement structures.

unrealistic starting point for model building. Real molecular replacement solutions will have larger errors from the rotation and translation steps, especially at lower resolution. To address these issues, a new method was developed that used a GESAMT archive search [81] to find structural homologues and PHASER [82] to perform molecular replacement. The new method is described in the supplementary material 'Automatic Creation of Molecular Replacement Test Sets' from Bond *et al.* [1], which is reproduced below.

2.3 Automatic Creation of Molecular Replacement Test Sets

2.3.1 Abstract

When testing software, it is important to use a large number of test cases so that the significance of performance improvements can be assessed. Molecular replacement (MR) is the most commonly used structure solution technique for X-ray diffraction data, and this has led to the development of a number of automated MR solution pipelines. Since different software tools often focus on different problem classes, for example data resolution, it is useful to be able to generate a custom test set for a given task. A program was written for the automatic creation of MR test sets to address this problem. In addition to this, a large example test set was prepared using the program. Two thousand structures, evenly spread between 1 Å and 3.5 Å resolution, were chosen at random from the PDB. The structures had to meet quality thresholds, measured using wwPDB validation percentiles, and not contain any chains with 50% or more sequence identity to chains in other chosen structures. After checking the structure factor data and refining the deposited structure, 1800 of the structures were deemed suitable. A search for structural homologues was carried out for the 2100 unique chains in these structures and 15532 MR models were made from the homologues, 11183 of which led to a refined solution. Two reduced test sets were produced with only one MR model per structure. The full reduced test set contains 1351 structures for the full resolution range and a wide range of initial phase qualities. The easy reduced test set contains 639 structures with better than 2.5 Å resolution, all with good quality phases. The BUCCANEER model building pipeline from CCP4i built models with a mean completeness of 39% for the full set and 87% for the easy set. The test set creation program and the example test sets are available as a resource for the community.

2.3.2 Introduction

Large test sets are required to determine whether new program developments lead to statistically significant performance improvements. It is likely that a change will

lead to better performance in some cases and worse performance in others, so it is not sufficient to test only one structure and assume that the same improvement will be seen across the board. Testing more independent structures will lead to a smaller standard error in the mean improvement. There is often a trade off between the speed and accuracy of a program, and being able to reliably measure differences means a more informed decision can be made about whether a change should be implemented.

The best way to proceed with structure solution in X-ray crystallography depends on the information currently available. A high resolution structure with only a single helix placed and a low resolution structure that requires some large scale domain movements present very different problems that need different methods to solve. As an example, ACORN [83] is a very powerful phase refinement procedure but it is only applicable to high resolution data. Having a large test set with a range of resolutions and initial phase qualities makes it possible to predict where the program would be beneficial, allowing for expert pipelines to be created that use different approaches depending on factors such as resolution.

Most program developers will already have their own test sets but these are usually not publicly available. For instance, the automated model building program ARP/wARP [46, 53] is available via a web service and the data submitted to that can be used to train ARP/wARP algorithms. The user may also give permission for their data to be shared with a wider audience, in which case it may be available on request. However, without further curation, user submitted data may not form a uniform or representative sample and so may not be suitable for any specific research question.

The work presented here has two main aims. Firstly, to create a large, publicly available test set with structures that are evenly spread across a range of resolutions and phase qualities. The independence of individual tests is controlled by ensuring the target structures do not have high sequence homology to each other. The second aim is to release the program used to make the test set so it can be repeated to create new test sets with varied parameters, such as a set containing only low resolution structures.

A source of initial phases is needed to test model building programs and phase

refinement programs. If the structure contains heavy atoms, and anomalous data has been deposited at suitable wavelengths, then SAD/MAD phasing could be used. However, it is a minority of structures in the PDB [17] that meet these requirements. It was decided to use molecular replacement (MR) as a source of initial phases because only mean structure factor amplitudes are needed and an MR model can be created from any homologous structure. A range of test cases can be produced by varying the similarity of the homologues, as well as the fraction of the target structure that they represent. This also means that the resulting test sets could be used to assess the performance of molecular replacement programs.

2.3.3 Methods

Calculations were performed on a Scientific Linux 7.7 server with two AMD EPYC 7451 CPUs and 256 GB RAM. Programs were sourced from CCP4 7.0.076 [84].

2.3.3.1 Test Set Creation

2.3.3.1.1 Choosing Target Structures A list of all deposited chains was downloaded from the RCSB PDB [17] and filtered to contain only L-polypeptide chains of at least 20 residues from structures solved by X-ray diffraction. In order to ensure an even spread across a range of resolutions, structures were placed into 10 bins between 1 Å and 3.5 Å resolution. Two hundred structures were chosen at random from each bin to give 2000 targets in total. Structures were only chosen if they were of suitable quality. Five statistics from the PDB validation report were used as overall quality indicators: R-free, calculated by DCC [85]; clashscore, Ramachandran outliers and sidechain outliers, calculated by MolProbity [16]; and real-space R-value Z-score (RSRZ) outliers, calculated by EDS [86]. R-free had to be in at least the 50th percentile relative to similar resolution structures and the other statistics had to be in at least the 40th percentile.

To stop common protein families being represented multiple times, structures were also rejected if they contained a chain with $\geq 50\%$ sequence identity to a chain in an already chosen structure. This was assessed using cluster numbers available from the PDB, which are pre-calculated using BLASTCLUST [87] at various sequence

identity thresholds. Resolution bins with fewer structures were considered first to avoid running out of qualifying structures.

Other than needing to contain a protein chain with at least 20 residues, there were no restrictions on the content or size of the structures. Structures can be hetero-multimers and may contain other entities that make model building more complicated, such as nucleic acids, cofactors and glycosylation.

2.3.3.1.2 Preparing Structure Data A FASTA format sequence file was downloaded from the RCSB PDB with entries for all 2000 structures. For each structure, sequences of protein chains with at least 20 residues were extracted and written to a separate sequence file. A second sequence file was written containing only the unique sequences as some programs require a file without duplicate entries.

Reflection data were converted from CIF format to MTZ format using CIF2MTZ. The files may contain amplitudes or intensities and data for Friedel pairs may be combined or held separately for anomalous phasing. In order to make the files contain the same type of structure factor data, they were all processed with CTRUNCATE [88], which converts intensities to amplitudes, anomalous data to mean data and also performs anisotropy correction. A new free-R flag was then assigned using the CCP4 utility FREERFLAG and column labels were standardised.

Unknown ligand (UNL) residues were removed from the deposited coordinates, which were then refined with REFMAC [11, 43] for 10 cycles. By default, the program exits if it encounters a new ligand, but this behaviour was altered to avoid too many failures. Instead, REFMAC was told to proceed with refinement using a dictionary description it creates from the ligand coordinates. The final R-factor was compared to the R-work reported in the PDB and the structure was rejected if it was more than 5% higher. Structures were also rejected if the overall data completeness was less than 90%.

2.3.3.1.3 Choosing Homologues For each unique chain in the selected structures, a search for structural homologues was performed using GESAMT [80] on a local copy of the PDB, updated on the 31st July 2019, containing 153578 structures. This search is very time consuming, but was sped up considerably by

using a pre-constructed GESAMT archive and searching in parallel over 96 threads. The GESAMT archive consists of compressed binary files containing only protein Cα coordinates that can be read very efficiently [81].

Homologues with $\geq 70\%$ sequence identity were removed from the results as these are too similar to make challenging MR models. At the other end of the scale, the search results also contain large numbers of chains that are too distant to be suitable for MR, so only chains with a C-alpha RMSD less than 3 Å and Q-score more than 0.2 were considered. For each target chain, up to 10 homologues were chosen from the filtered list in order of descending Q-score. If a homologue had a sequence identity more than 70% or a C-alpha RMSD less than 1.5 Å to a previously chosen homologue then it was eliminated.

2.3.3.1.4 Preparing MR Models Each homologue chain was superposed onto its target chain using GESAMT to produce a sequence alignment. The sequence files from GESAMT were converted to CLUSTAL [89] format alignment files for SCULPTOR [79], which was used for preparing the MR models. Alternate conformations were removed from the input model and default parameters were used for pruning.

2.3.3.1.5 Molecular Replacement Molecular replacement was carried out for each SCULPTOR model using PHASER [82]. This was done for each model individually, so there will be no solutions containing multiple components other than multiple copies of the same model. In a real MR scenario the target structure is not known, so only sequence identity from the GESAMT alignment was given to PHASER, which then made its own estimate of the model RMS error. The composition of the asymmetric unit was defined using the counts of each atom type in the deposited coordinates. The number of copies to search for was also known from the deposited structure. In order to speed up cases that lacked a clearly significant solution, the solution list was purged to keep only the top solution at the rotation function, translation function and refinement stages.

The placed MR models were refined for 10 cycles with REFMAC using default parameters. Phases from the refined MR model were compared to phases from the

refined deposited structure using the CCP4 utility CPHASEMATCH, which will correct for alternate origins chosen during molecular replacement.

2.3.3.2 Test Set Reduction

The full test set contains multiple placed MR models for each structure, up to 10 individual models for each unique chain. This could be useful to see whether a model building program can produce a correct structure from a variety of starting points, but it is not optimal in many circumstances. It takes much longer to run the whole test set and results obtained for separate models are less independent. To address these issues, two smaller test sets were constructed by choosing a single model for each target structure.

The first, named the full reduced test set, aims to have a broad range of phase qualities. Phase quality was measured using F-map correlation, which is the correlation coefficient between the structure factor amplitudes of the map from the refined MR model and the map from the refined deposited model, weighted by the cosine of the phase difference. Seven F-map correlation bins were created between 0.2 and 0.9. The models for each structure were checked in a random order to see if they belong in the bin with the least number of structures. If no model was found, then the bin with the next least structures was considered until a suitable model was found.

The second is named the easy reduced test set for cases that should be easily solved by automatic model building. It was created using the same method as the full reduced test set, but only with structures where the resolution is 2.5 Å or better and using 5 F-map correlation bins between 0.7 and 0.95.

2.3.3.3 Model Building with BUCCANEER

Automated model building was carried out on both reduced test sets using the BUCCANEER pipeline from CCP4i [37, 39] with default options for molecular replacement, including model seeding, which adds every third residue in the MR model to the input model. In some cases the target structure was a

Table 2.1: Reasons for 200 out of the 2000 structures being rejected.

Count	Reason
114	Data completeness below 90%
45	Error during refinement
34	Refined R-work more than 5% higher than reported
4	Error in the deposited coordinate file
1	No symmetry information in the structure factor data
1	No standard deviations in the structure factor data
1	Error during the least squares fit when converting amplitudes

selenomethionine derivative and MSE residues were built instead of MET. The final models from the BUCCANEER pipeline were superposed onto the refined deposited structure using CSYMMATCH, which searches for the best fit using symmetry operations and allowed origin shifts.

2.3.4 Results and Discussion

2.3.4.1 Test Set Creation

The initial structure selection can be altered by changing the minimum and maximum resolution, number of resolution bins, number of structures per resolution bin, maximum sequence identity, and validation thresholds. The maximum sequence identity may need to be increased and the validation thresholds decreased if looking for large numbers of structures at high or low resolution where there are less structures available.

When choosing the number of structures, it should be taken into account that not all will be suitable. Out of the 2000 structures initially selected, only 1670 (83.5%) had one or more refined MR models, which may be correctly or incorrectly placed. 200 structures were rejected at some stage during data preparation, reasons for which are shown in Table 2.1.

The most common reason for rejection was low data completeness. It was decided that a threshold of 90%, rejecting 5.7% of structures, was acceptable but this can

be changed by the user. The next most common reason was errors occurring during refinement, which were mainly due to ligand atoms being absent in the library. The deposited structures had to refine to within 5% of the reported R-work using default parameters in REFMAC. If the structure was originally refined using non-default procedures, such as twinned refinement or anisotropic B-factor refinement, then it will have a much higher chance of being rejected at this stage.

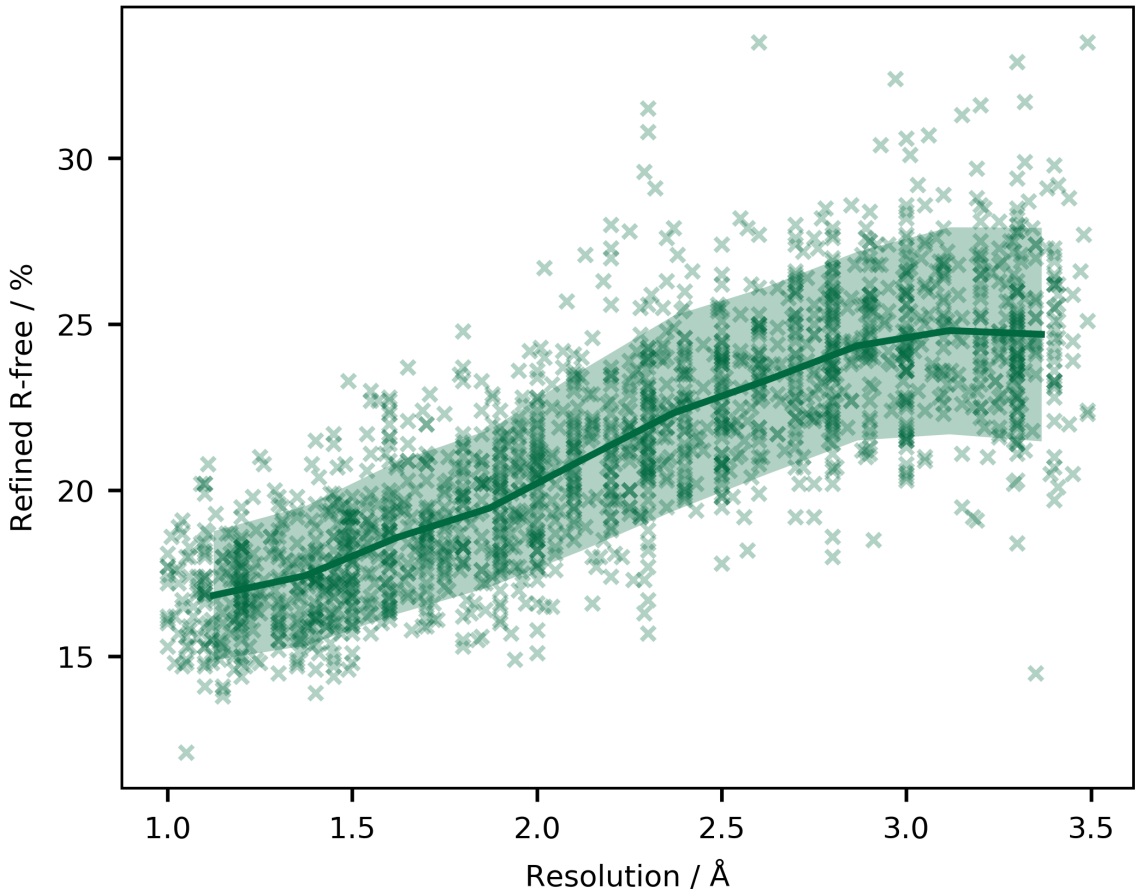

Figure 2.4: Resolution and refined R-free for the 1800 structures that passed the data preparation stage. Raw data are shown as crosses. Mean values for 10 resolution bins are shown as a solid line. The shaded area shows one standard deviation above and below the mean.

Resolution and refined R-free for the 1800 structures that passed are shown in Figure 2.4. As expected, lower resolution structures generally have higher R-factors. There were structures with much higher R-free values but these were rejected.

The 1800 chosen structures had 2100 unique chains between them with 15551

structural homologues chosen from the GESAMT archive searches. The maximum number of homologues to choose for each target chain can be modified, along with how similar the homologues can be to the target and how similar they can be to each other. Searching for structural homologues instead of sequence homologues should lead to a lower failure rate during molecular replacement. In a real molecular replacement scenario this is obviously not possible because the target structure is not known.

SCULPTOR made an MR model for 15532 of the homologues. In the other 19 cases an empty coordinate file was produced. PHASER rejected the input for 8 of the models. In one case this was due to a poor ensemble model from a homologous structure containing two models that did not correlate well with each other. The other 7 errors were from two structures that had some reflections with negative structure factor amplitudes. PHASER failed to find a solution for 4341 out of the 15524 runs that terminated successfully, leaving 11183 placed MR models that were refined.

The GESAMT Q-score is a measure of alignment quality that takes into account both C-alpha RMSD and the length of the alignment [80]. It increases from 0 to 1 as the structural similarity of the two structures increases, so models with higher Q-scores should be more successful during molecular replacement. PHASER provided a Log Likelihood Gain (LLG) and an estimated main-chain RMSD for 10896 of the models it placed. Figure 2.5 shows GESAMT Q-score and F-map correlation for these models.

The GESAMT Q-scores in Figure 2.5 are from the superposition step and not the structural homologue search. There are occasional differences between these values, hence a few homologues have Q-scores below 0.2 despite that being the minimum during selection. More than half (56%) of the placed MR models have F-map correlations below 0.15. It is likely that this cluster is formed mostly of incorrect solutions.

Increasing the minimum Q-score between the homologue and target chains should increase the success rate of molecular replacement. As expected, Figure 2.5 shows a positive correlation between Q-score and F-map correlation, but the structural similarity of the homologue is not the only factor to take into account. One model

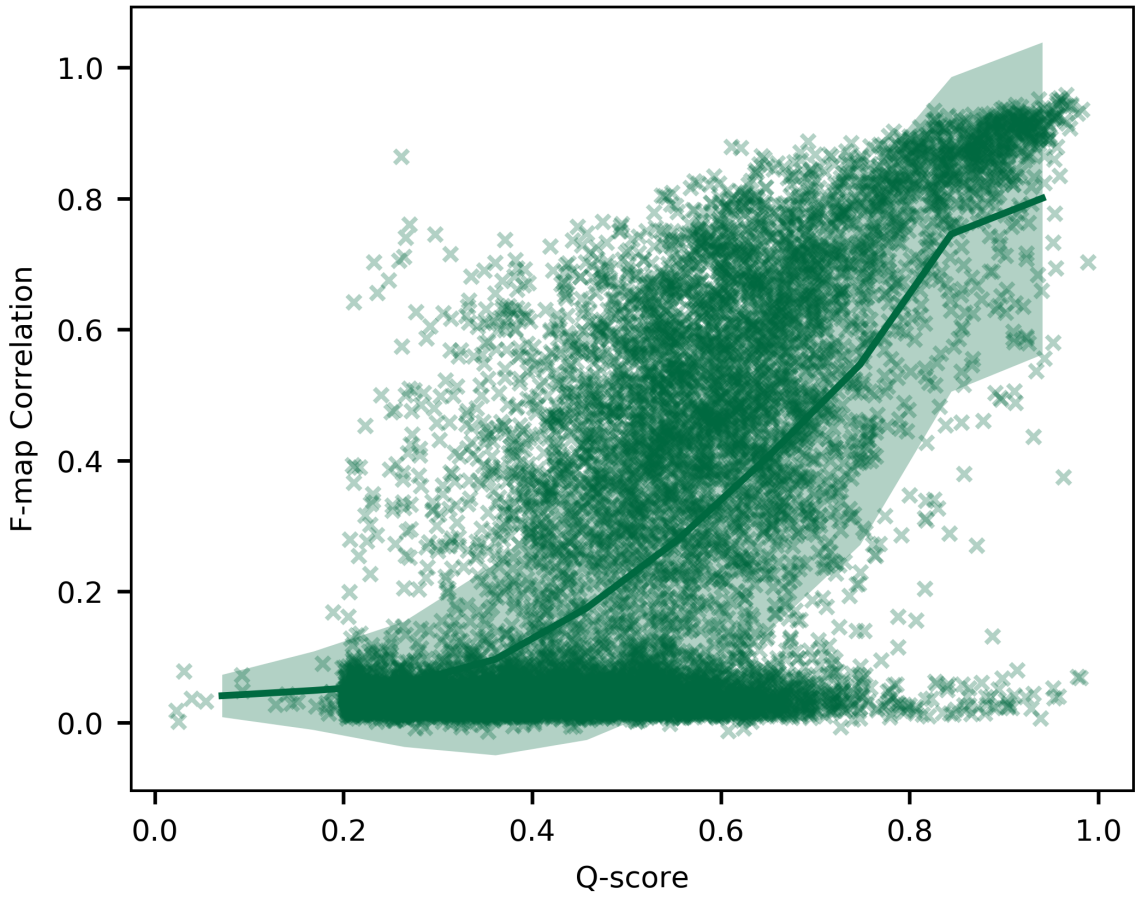

Figure 2.5: GESAMT Q-score and F-map correlation for 10896 models placed with PHASER and refined using REFMAC. Mean and standard deviation for 10 Q-score bins are overlaid.

has a Q-score of 0.94 but an F-map correlation of only 0.01. The target structure is 6HV7 at 3.4 Å resolution, which has 6602 residues comprising two copies of 14 unique chains. Both the target, chain I, and the homologue, chain J of 4R3O, have 204 residues built. The GESAMT superposition aligns 203 residues with a C-alpha RMSD of 0.70 Å and a sequence identity of 54%. Although the model is very similar, it is small in comparison to the full structure and the resolution is low so PHASER could not produce a correct solution.

In another example, a model has a Q-score of 0.26 but this leads to an F-map correlation of 0.86. The target structure is 5MN7 at 3.3 Å resolution. It has two copies of a 305 residue chain with 303 residues in the deposited model. The homologue is chain A from 2Q1Y with 305 residues, but the GESAMT

superposition only aligns 160 residues with a C-alpha RMSD of 0.75 Å and a sequence identity of 64%. The Q-score is low because it is calculated for just over half of the full length. However, the sequence alignment produced is for the full length and SCULPTOR is still able to produce a good model despite the alignment containing an incorrect gap.

Not all of the models with low F-map correlations are incorrect. Correctly placed solutions can still lead to low F-map correlations if the model is dissimilar or makes up a small fraction of the complete structure. In these cases model building will be challenging but it might be possible to improve the phases using density modification or further molecular replacement with other parts of the structure. Log Likelihood Gain (LLG) is often used to judge the correctness of a solution. Table 2.2 shows the percentage of correct solutions for different LLG ranges, assuming that solutions with F-map correlations greater than 0.15 are correct.

When determining whether a solution is correct, it is best to look at LLG in combination with other statistics, such as the refined Translation Function Z-score (TFZ) and the number of packing clashes. The R-factors of the refined MR model are also useful. Figure 2.6 shows how F-map correlation varies with R-work. Solutions with F-map correlations less than 0.15 generally have R-work values above 47%. However, most of the correct solutions also have R-factors in this region. A refined R-work of 50–55% does not give much information about the

Table 2.2: Percentage of correct solutions for different LLG ranges. Solutions with F-map correlations greater than 0.15 are classed as correct.

Min LLG	Max LLG	Number of Solutions	Correct / %
0	20	718	1.4
20	40	3784	8.4
40	60	1795	29.4
60	80	822	50.1
80	100	474	70.3
100	120	346	80.6
120	140	259	92.3
140	160	211	93.4

correctness of the solution, but values less than 45% are very likely to be correct solutions.

PHASER produces an estimate of the main-chain RMSD between the model and the true structure. This can be compared to the C-alpha RMSD from GESAMT, which is calculated for the aligned region of the homologue and target chains. Figure 2.7 shows RMSD from both PHASER and GESAMT. There is a positive correlation but PHASER overestimates low RMSD values and underestimates high RMSD values compared to the actual values from GESAMT.

The purpose of performing molecular replacement was to get some placed models with realistic errors for testing different model building strategies. A less realistic route would have been to superpose the MR model onto all the copies of the target chain using GESAMT. If the goal is to create a test set for assessing molecular

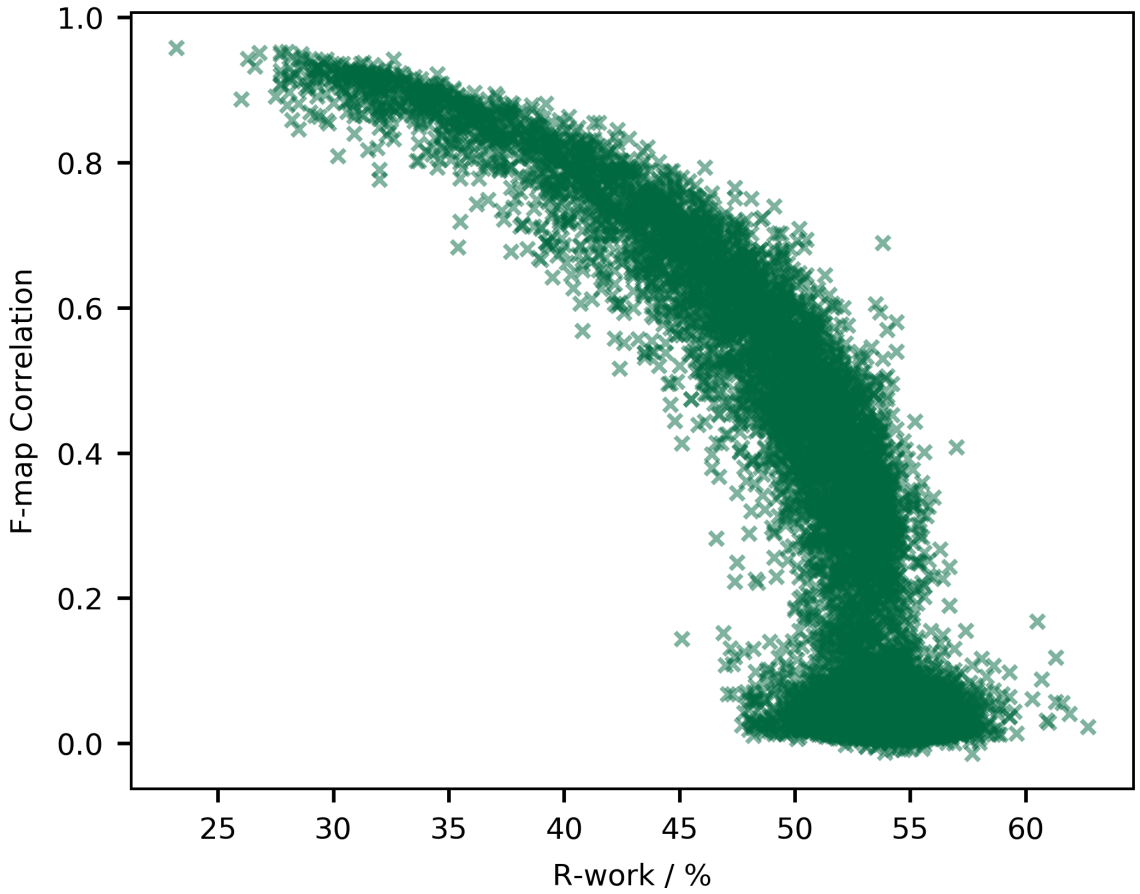

Figure 2.6: R-work and F-map correlation for 10896 models placed with PHASER and refined using REFMAC.

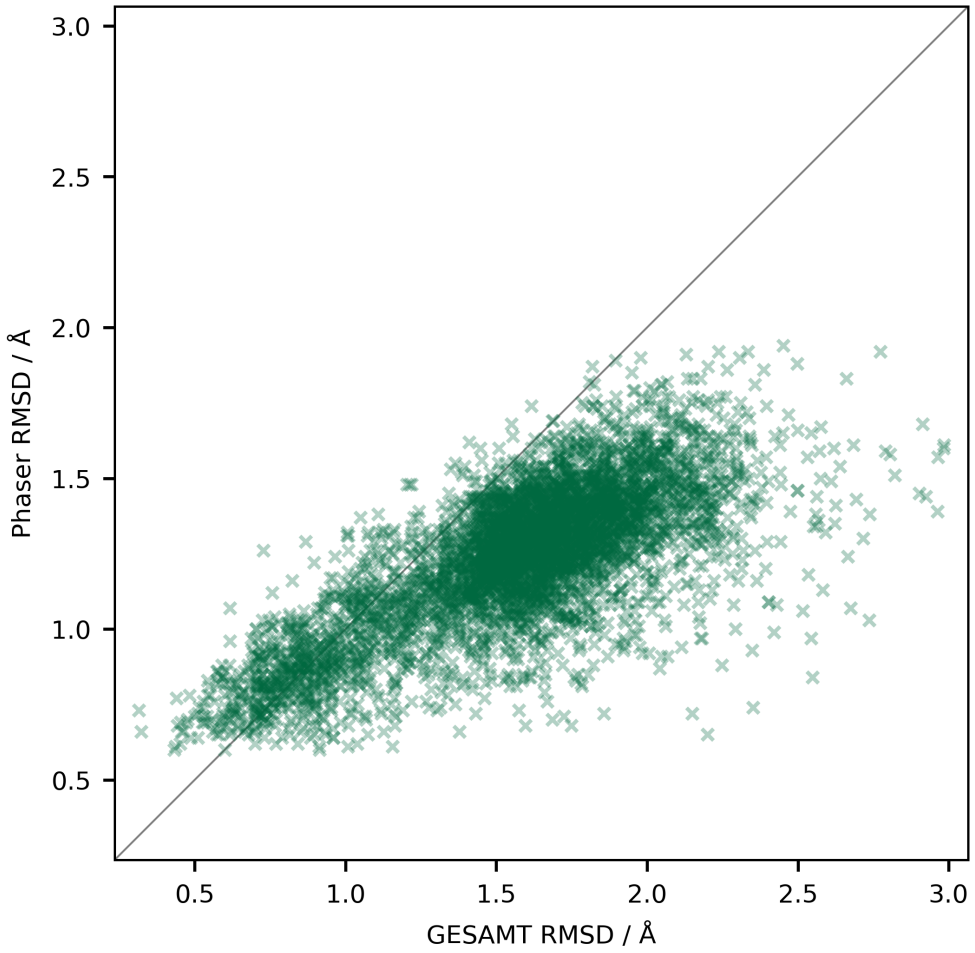

Figure 2.7: GESAMT C-alpha RMSD vs PHASER estimated main-chain RMSD for 4773 models with F-map correlation greater than 0.15.

replacement then the creation script can be stopped after SCULPTOR creates the models.

Because the aim was not to solve as many cases as possible, neither PHASER nor REFMAC were used to their full potential. Intensities, if originally available, were converted to amplitudes despite intensities being preferred for the LLGI function [90], as it was decided that having standardised data provided a more useful comparison. Purging all but the top solution also severely limited the performance of PHASER, but had to be done to save time when running thousands of jobs without clear solutions. It is usually preferable to first refine MR models in REFMAC using rigid body refinement or to include jelly body restraints for many cycles, especially when there are large scale differences between the model and the true structure, but this

was also not done in order to save time.

2.3.4.2 Test Set Reduction

There are 1351 structures in the full reduced test set with resolutions between 1.0 and 3.5 Å and F-map correlations between 0.2 and 0.9. The easy reduced test set has 639 structures with resolutions between 1.0 and 2.5 Å and F-map correlations between 0.7 and 0.95. In both test sets, cases are spread evenly across the resolution and F-map correlation ranges.

2.3.4.3 Model Building with BUCCANEER

Table 2.3: Overall performance of the CCP4i BUCCANEER pipeline on both the full and easy reduced test sets. Values shown are the mean ± one standard error.

	Full Set (1351 structures)	Easy Set (639 structures)
Completeness / %	39.5 ± 0.8	86.7 ± 0.5
R-work / %	43.1 ± 0.2	30.6 ± 0.2
R-free / %	48.9 ± 0.2	34.5 ± 0.3

The overall performance of the CCP4i BUCCANEER pipeline on the reduced test sets is shown in Table 2.3. Completeness is the percentage of residues in the refined deposited structure that have a matching residue in the model. Two residues were only considered matching if the N, CA and C positions were all within 1 Å. As expected, performance is much better on the easy reduced test set. Performance on some structures will be limited due to the presence of non-protein components, such as nucleic acids, that BUCCANEER is not able to build.

Figure 2.8 shows how completeness varies with resolution for 389 cases with a starting F-map correlation of 0.7 or more. There is more of a drop in performance at low resolution than was observed for simulated low resolution experimentally phased datasets, which still had a mean completeness higher than 50% at 3.4 Å resolution [70]. There are many factors contributing to this difference. Firstly, the simulated datasets have better phase information than would normally be obtained at low resolution. The maps produced by experimental phasing and molecular

replacement are also quite different. Even at a similar level of F-map correlation, an experimentally phased map will likely have more uniformly distributed errors. Molecular replacement maps contain model bias that makes model building more challenging. BUCCANEER also uses different options depending on the initial source of phases. If an MR model is available it will be used for Cα seeding. If experimental phases are provided they will be given to REFMAC for MLHL refinement. Lastly, the completeness metric used in this study is slightly stricter as it matches residues using N and C positions as well as Cα positions. Both metrics used a tolerance of 1 Å for correct atomic positions. A drop in performance at low resolution is expected due to this as two similar quality models are less likely to be within a fixed tolerance of each other.

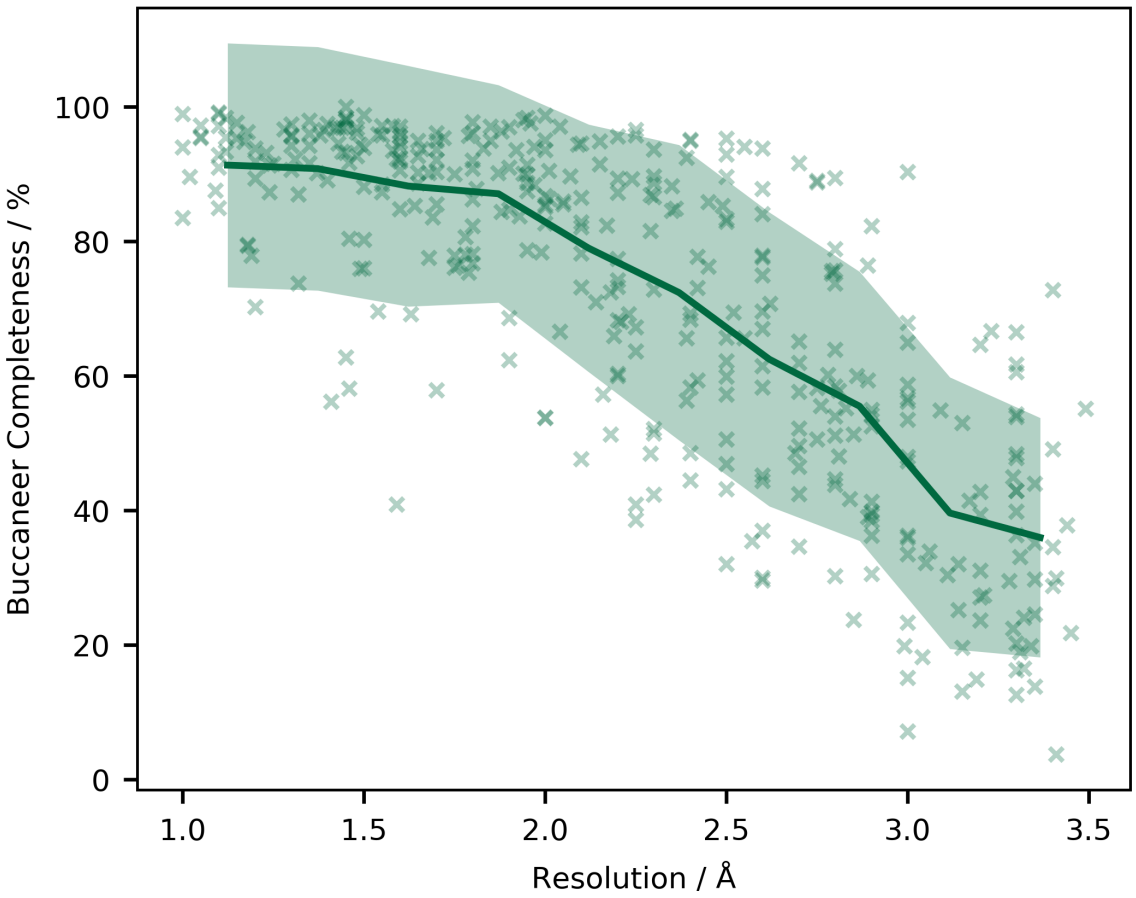

Figure 2.8: Resolution and completeness of the BUCCANEER model for 389 structures in the full reduced test set with F-map correlation 0.7 or more. Mean and standard deviation for 10 resolution bins are overlaid.

Figure 2.9 shows how completeness varies with phase quality for 911 cases with resolutions 2.5 Å or better. BUCCANEER is known to be sensitive to phase quality [37]. Below an F-map correlation of 0.4 there is only one model with more than 50% completeness. However, all of the cases contain some correct phase information, from which it would hopefully be possible to bootstrap a correct solution.

2.3.5 Conclusion

A program has been developed for the automatic creation of molecular replacement test sets. It starts by choosing good quality target structures with diverse protein sequences that span a range of resolutions. A search for structural homologues is then

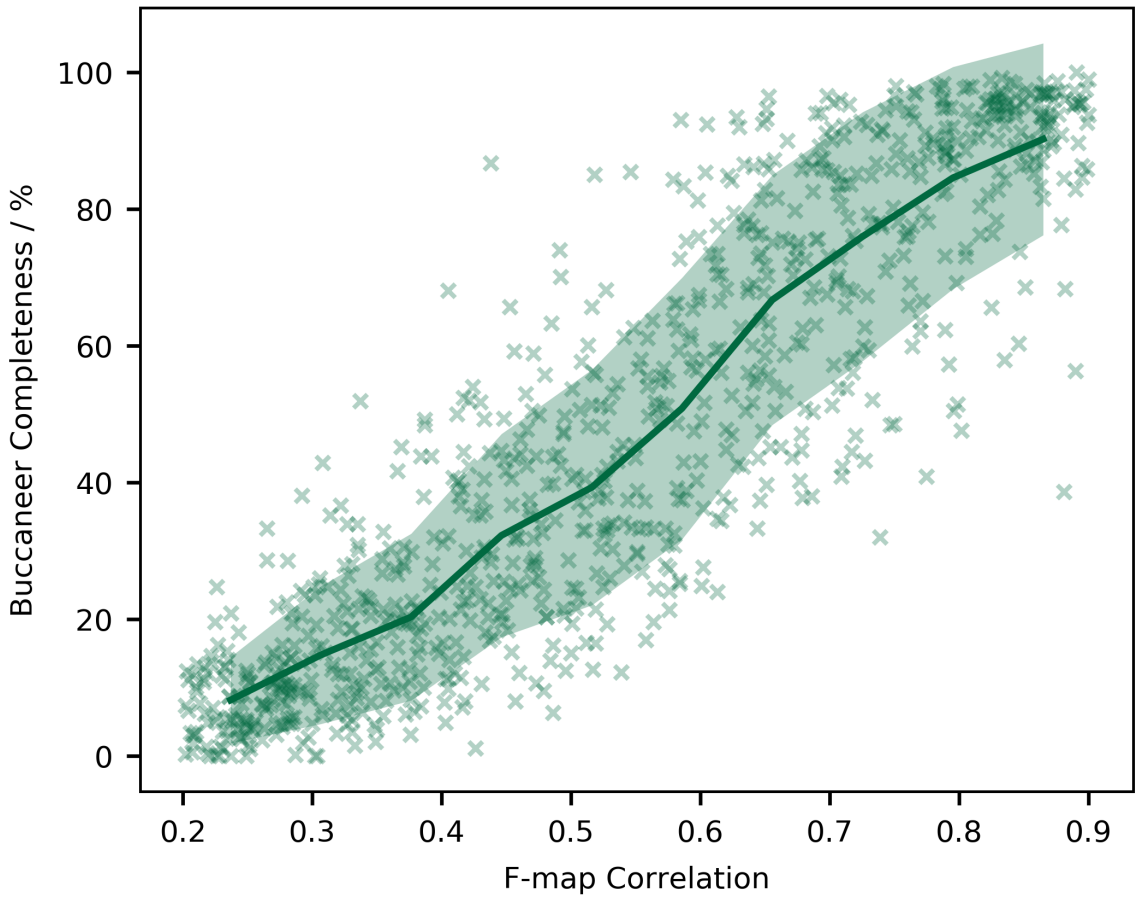

Figure 2.9: F-map correlation and completeness of the BUCCANEER model for 911 structures in the full reduced test set with resolutions 2.5 Å or better. Mean and standard deviation for 10 F-map correlation bins are overlaid.

carried out and molecular replacement models are prepared from the homologues. The program can be terminated at this point, or it can continue to perform molecular replacement and refine the placed solutions when a source of initial phases is needed. Many parameters can be adjusted to suit the needs of the user, such as the number of structures, resolution range, validation thresholds, number of models and the similarity of models.

An example test set was created that contains 15532 MR models across 1800 structures with resolutions between 1 Å and 3.5 Å. Hopefully this set is large enough that the program only needs to be repeated for more specialist needs. The full test set could be useful for assessing the performance of molecular replacement programs. However, most of the models do not produce correct solutions that are needed for testing model building and phase refinement programs.

Two reduced test sets were derived by selecting one model per structure. The easy reduced test set has 639 structures with resolutions between 1 and 2.5 Å where the model has an F-map correlation between 0.7 and 0.95. These should all be cases where automated model building works well so they will be useful for comparing model completion algorithms, where the aim is to replace routine manual model building tasks that need to be done after automated building has finished. The full reduced test set contains 1351 structures between 1 and 3.5 Å resolution with F-map correlations between 0.2 and 0.9, which is useful for assessing the performance of automated model building in more challenging cases.

The BUCCANEER pipeline from CCP4i was tested on both reduced test sets. As expected, performance was better at high resolution and with good quality phases. The pipeline runs for 5 cycles by default and, although this is sufficient for most of the easy cases, performance on some of the more difficult cases will be improved with more cycles. This issue has been addressed by recent improvements to the CCP4i2 [91] pipeline, which will be discussed in a future publication.

2.3.6 Availability

All the data relating to this publication are available at:

https://doi.org/10.15124/44145f0a-5d82-4604-9494-7cf71190bd82

This includes the test set creation program and other scripts, the unreduced test set with multiple models per structure, and both reduced test sets. The test set creation program is also available on GitHub at https://github.com/paulsbond/create-mr-set.

2.3.7 Acknowledgements

The authors would like to thank Ronan Keegan for his advice on searching for structural homologues using GESAMT and Huw Jenkins for help with PHASER.

This work was supported by the White Rose BBSRC DTP in Mechanistic Biology (BB/M011151/1) and BBSRC grant BB/S005099/1.

2.4 Summary

The number of structures used to test BUCCANEER has been greatly expanded from 54 experimental phasing cases to 1351 molecular replacement cases and 202 experimental phasing cases. The larger test sets allow much more thorough testing of proposed changes to the protocols to be carried out. The resulting changes in performance can then be measured with greater statistical significance and errors that occur infrequently have a greater chance of being identified. A comparison of resolution and F-map correlation for the cases in the old experimental phasing test set and the new experimental phasing and molecular replacement test sets is shown in Figure 2.10.

The molecular replacement test set is much larger and it would be good to increase the experimental phasing test set to a similar size with a similar coverage of resolution and phase quality. The simulated low-resolution datasets help to some degree, but the resulting datasets are not independent because they use the same structures at each resolution. In addition, the B-factors of the structure factor amplitudes were inflated but the phases were left unchanged, which will give unrealistically good phases for lower resolution data where phasing is more difficult.

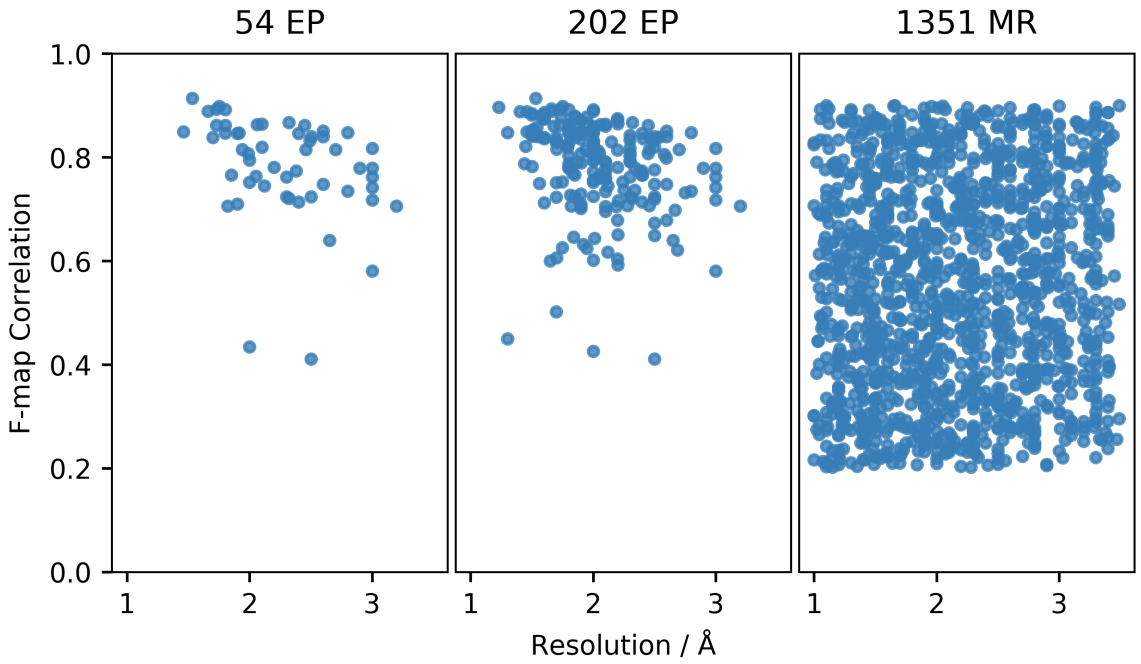

Figure 2.10: Resolution and F-map correlation for three test sets: 54 experimental phasing cases (left), 202 experimental phasing cases (centre) and 1351 molecular replacement cases (right). F-map correlation is the correlation coefficient between the amplitudes of the starting map and the map from the refined deposited structure, weighted by the cosine of the phase difference.

It should be possible to create a new experimental phasing test set from information deposited in the PDB. It may be possible to perform SAD/MAD phasing for those structures with anomalous intensities or amplitudes deposited for one or more wavelengths. The deposited structure can be examined to see what elements could be used for substructure determination and phasing could be performed using an automated pipeline such as CRANK2 [92]. Alternatively, some entries may contain experimental phases in the structure factor data and these could provide a more direct starting point. A drawback of this approach will be the lack of a heavy atom substructure model to aid phase improvement and model building. One could be constructed from the heavy atoms in the deposited model, but this is unrealistic as there will be no errors. It also might not be possible to determine the source of the phases in an automated fashion, since they may come from a number of programs and may or may not include density modification.

The MR test set is large enough for many purposes but it contains a number of

structures that could be considered unrealistic starting points for model building. For example, some structures contain multiple copies of a protein but not all the copies have been found by molecular replacement. If automated model building cannot complete this partial solution then the user would repeat the molecular replacement step until a more complete solution is produced. There are also many structures with multiple components, but molecular replacement was only carried out for one component. In reality, a user would prepare models for each component for which homologous structures are available in the PDB and search for the largest component or the component with the best model first.

Chapter 3

Model Correctness

Validation is important to determine the correctness of residues within a model. In addition to aiding manual model building by identifying areas in need of correction, it can be used in automated procedures to remove incorrect residues introduced by imperfect model-building methods. Within BUCCANEER, a Cα likelihood function is used to score the protein backbone and a mean density Z-score to score side chains. These scores are useful but they do not give the full picture. Parts of the structure with more disorder will have worse scores despite being built correctly. In COOT, plots of Ramachandran scores and rotamer scores are often used for identifying incorrect residues. Lovell *et al.* [41] state "A residue with good fit to density, low B-factor, favored ϕ, ψ values, a rotameric sidechain, no atomic clashes, and ideal covalent geometry is almost certain to be modeled correctly.", but is it possible to combine these separate validation metrics into a single score that measures the 'correctness' of a residue? This chapter covers an initial attempt at solving this problem using machine learning.

The content below is taken from "Predicting protein model correctness in Coot using machine learning" by Bond *et al.* [1].

3.1 Predicting Protein Model Correctness in Coot Using Machine Learning

3.1.1 Abstract

Manually identifying and correcting errors in protein models can be a slow process, but improvements in validation tools and automated model building software can contribute to reducing this burden. This paper presents a new correctness score that is produced by combining multiple sources of information using a neural network. The residues in 639 automatically built models were marked as correct or incorrect by comparing them with the coordinates deposited in the PDB. A number of features were also calculated for each residue using *Coot*, including map-to-model correlation, density values, B factors, clashes, Ramachandran scores, rotamer scores and resolution. Two neural networks were created using these features as inputs: one to predict the correctness of main-chain atoms and the other for side chains. The 639 structures were split into 511 that were used to train the neural networks and 128 that were used to test performance. The predicted correctness scores could correctly categorize 92.3% of the main-chain atoms and 87.6% of the side chains. A *Coot* ML Correctness script was written to display the scores in a graphical user interface as well as for the automatic pruning of chains, residues and side chains with low scores. The automatic pruning function was added to the *CCP*4*i*2 *Buccaneer* automated model-building pipeline, leading to significant improvements, especially for high-resolution structures.

3.1.2 Introduction

Manual completion of a model is a very time-consuming step in macromolecular structure solution. Initial models from homologues or from automated model-building programs will contain errors that must be identified and corrected. The primary method for identifying errors is visual examination of the model, the $2mF_o - DF_c$ map and the $mF_o - DF_c$ map by the crystallographer, using a model-building program such as *Coot* [15, 33]. Errors can often be identified by visual examination alone. However, other validation metrics become more

important in guiding decisions when the density is less obvious, for example in less ordered regions or lower resolution structures. *Coot* provides validation tools to identify Ramachandran outliers, unusual rotamers, and other potential errors, as well as an interface to some tools from *MolProbity* [16]. The job of the crystallographer is to combine all of these sources of information and decide whether the model is acceptable or whether it needs to be changed. The work presented here aims to emulate this decision-making process by using machine learning to predict the correctness of protein residues. Machine learning is well suited for this problem as expected patterns in the data are not written into the model in advance but can be found through analysis of the training data. A recent example from the field of crystallography is the use of initial data-processing statistics to predict whether the data are suitable for successful structure determination through SAD/MAD phasing [93].

The correctness of a model is not something that is easy to define. If the coordinates of an atom are altered gradually, there is no definitive point at which the position becomes correct. The model needs to fit both the experimental data and previously acquired knowledge of atomic structures, especially at lower resolution when it is not possible to distinguish individual atomic peaks. In this space it is likely there are multiple local minima, the positions of which will vary depending on the refinement procedure. However, alternate conformations aside, usually only one minimum is considered to be correct within an individual refinement procedure.

Predicting the correctness of residues can be formulated as a supervised machine-learning problem, where each data point has several feature attributes that are used to predict another target attribute. In this application, a data point is a residue, the features are pieces of information about the residue, for example the Ramachandran score and a score of the fit to density, and the target is correctness. The prediction could be performed using either classification, where each residue is labelled as correct or incorrect, or regression, where a numerical correctness score is assigned. It was decided to use regression as the score would be useful for graphical validation tools and for automated procedures to select badly scoring residues at various thresholds.

The amount of manual model-building work that needs to be performed can be drastically reduced by having better automated model-building programs that lead

to models with fewer errors. *Buccaneer* [37, 39] is a fast model-building program that works well at a range of resolutions and is distributed with the *CCP*4 software suite [84]. It does not perform any global refinement of coordinates or B factors, so it is most effective when combined with a refinement program such as *REFMAC* [11, 43] in an iterative pipeline. The refinement program improves the model geometry and fit to density and produces an updated map that can be passed to the next building cycle. There are *Buccaneer* pipelines available in *CCP*4*i* [94] and *CCP*4*i*2 [91]. *Buccaneer* is also used in other pipelines such as *CRANK* [95, 96], *CAB* [77] and *CCP*4*Build*, which is a new model-building pipeline available in *CCP*4*Cloud* [97].

It has been observed that although *Buccaneer* is good at building complete structures at low resolution, it can build more incorrect residues than other programs [69, 70]. The incorrect residues are mostly small unsequenced chains built into the solvent that need to be removed by the user at the end of the pipeline. There are already some existing steps within *Buccaneer* for removing chains: the filter step removes chains shorter than six residues and the pruning step solves clashes between chains by truncating the chain with the most unsequenced residues or the shorter chain. However, if the chain contains at least six residues and does not overlap with another chain, then it will be kept. It would also be useful to have a method for deleting individual residues and side chains identified as incorrect. Errors such as peptide bonds that need flipping and side chains built with the wrong rotamer are not uncommon. If pruning these errors is followed by refinement, then the resulting likelihood-weighted maps will be less biased towards the error and future automated building cycles are more likely to correct the issue. A pruning step has already been implemented in *CCP*4*Build* that uses real-space difference density Z-scores (RSZDs) from *EDSTATS* to identify residues and side chains to delete. The RSZD metric is calculated separately for main-chain atoms and side chains and is useful for determining how accurately parts of a structure fit the electron density, but the calculation can be slow for high-resolution structures. A new pruning step is presented here that uses the machine-learned correctness scores to delete whole chains, individual residues, and side chains. We show that this pruning step enhances the ability of the *Buccaneer* pipeline to self-correct mistakes and produce better models that need less manual correction.

3.1.3 Methods

Calculations were performed on a Scientific Linux 7.7 server with two AMD EPYC 7451 CPUs and 256 GB RAM. Programs were sourced from *CCP*4 7.0.076 [84].

3.1.3.1 Structure-set curation

A program was written for choosing sets of target structures and creating molecular-replacement models using existing structures in the PDB [17]. The goal was to choose diverse, good-quality target structures that cover a range of resolutions and to produce a range of molecular-replacement models, some leading to good-quality phases and some leading to poor-quality phases. Using this program, 1800 target structures at 1–3.5 Å resolution were chosen with 11183 molecular-replacement models between them. This set was reduced by choosing a subset of the target structures with only one molecular-replacement model per structure. Two reduced sets were created: a full reduced set with 1351 structures at 1–3.5 Å resolution with a wide range of initial phase qualities and an easy reduced set with 639 structures at 1–2.5 Å resolution with only good-quality phases. The program and structure sets are documented in detail in the supporting information and are available to other developers.

3.1.3.2 Neural network target

For the 639 structures in the easy reduced set, models automatically built with the *CCP*4*i Buccaneer* pipeline were used to provide examples of both correct and incorrect residues. Refined versions of the models deposited in the PDB were used as references that are assumed to be wholly correct. As detailed in the supporting information, target structures were only chosen if they had good overall quality indicators, *i.e.* R_{free}, clashscore [16] and percentage outliers, so only a small minority of residues should have errors. The target correctness values of residues were assigned by comparing them with the reference structure. An alternative would be to label residues manually, which could be more accurate but would be very time-consuming and many samples are needed for higher coverage of the feature space. The *Buccaneer*

models were first moved onto the reference using *CSYMMATCH*, which searches for the best fit using symmetry operations and allowed origin shifts, and refined again using *REFMAC*. For an individual residue, if all of the main-chain atoms, including C^{β}, are within 1 Å of an equivalent atom in the reference, then the main chain of that residue is given a correctness score of 1. However, if one of the atoms is more than 1 Å away from the reference then the main chain of the residue is given a correctness score of 0. The same calculation is performed for the side-chain atoms from the γ position onwards. Asparagine, glutamine and histidine have side chains that still fit the density well if the terminal χ angle is rotated by 180° so these are classed as correct if built either way round.

3.1.3.3 Neural network features

The features used are summarized in Table 3.1. There are 12 features for predicting main-chain correctness and nine features for predicting side-chain correctness. Eight features are used for both but, other than resolution, they are calculated separately

Table 3.1: Summary of the features used to predict main-chain and side-chain correctness.

Features	Main/side chain	Section
Map-to-model correlation	Both	3.1.3.3.1
Mean best density Z-score	Both	3.1.3.3.2
Minimum best density Z-score	Both	3.1.3.3.2
Minimum difference density Z-score	Both	3.1.3.3.2
Maximum B-factor Z-score	Both	3.1.3.3.3
Maximum B-factor change Z-score	Both	3.1.3.3.3
Maximum atom overlap	Both	3.1.3.3.4
Resolution	Both	3.1.3.3.5
Ramachandran score	Main	3.1.3.3.6
Maximum peptide twist	Main	3.1.3.3.7
Pepflip peak	Main	3.1.3.3.8
Difference density Z-score at the next C^{α}	Main	3.1.3.3.2
Rotamer score	Side	3.1.3.3.9

for the main-chain atoms (N, C^{α}, C^{β}, O and C) and side-chain atoms from the γ position onwards. *Coot* 0.8.9.2 was used to calculate all features using functions described in the *Coot* user manual [98]. Explanations of individual features can be found in Sections 3.1.3.3.1–3.1.3.3.9.

3.1.3.3.1 Map-to-model correlation Correlation coefficients were calculated using the *map-to-model-correlation* function. Two different masks were used to calculate this separately for the main-chain atoms (atom mask mode 1) and the side-chain atoms excluding C^{β} (atom mask mode 3).

3.1.3.3.2 Density Z-scores Values of the $2mF_o - DF_c$ (best) density map and the $mF_o - DF_c$ (difference) density map were measured at the atomic positions of each atom. The raw map values were normalized by dividing them by the atomic number of the atom, and they were then converted to modified Z-scores using (3.1) [99], where $\tilde{x}$ is the sample median:

$$\begin{aligned} \text{MAD} &= \text{median}_i\{|x_i - \tilde{x}|\}, \\ Z &= \frac{0.6745(x_i - \tilde{x})}{\text{MAD}}. \end{aligned} \tag{3.1}$$

This uses the median of absolute deviations from the median (MAD) as a replacement for standard deviation as it should be more robust in skewed distributions. Z-scores were calculated separately for main-chain and side-chain atoms over the whole structure. Three features were used to predict both main-chain and side-chain correctness: the mean best density Z-score, the minimum best density Z-score and the minimum difference density Z-score. In addition, the difference density Z-score at the C^{α} position of the next residue was used as a feature to predict main-chain correctness.

3.1.3.3.3 B-factor Z-scores Isotropic B factors were recorded for each atom, as well as the maximum percentage increase from the B factors of bonded atoms. B factors and the maximum change in B factors were converted to modified Z-scores for main-chain and side-chain atoms as described in section 3.1.3.3.2. The maximum B-factor Z-score and maximum B-factor change Z-score were used to predict both main-chain and side-chain correctness.

3.1.3.3.4 Atom overlap To measure the extent to which a residue clashes with its neighbours, a list of atom-overlap volumes was obtained using the *molecule-atom-overlaps* function. This was used to calculate the maximum overlap for the main-chain atoms and side-chain atoms of each residue.

3.1.3.3.5 Resolution The high-resolution limit of the data does not vary per residue, but it was included as a feature as it should be useful for adjusting the weights of other features.

3.1.3.3.6 Ramachandran score The main-chain conformation of each residue is assigned a probability based on how often the combination of its ϕ and ψ angles are observed in high-quality protein structures. This information was obtained using the *all-molecule-ramachandran-score* function, which uses three probability distributions derived from the Top500 database [41]: one for glycine, one for proline, and one for other residue types.

3.1.3.3.7 Peptide twist The twist of a peptide bond was measured as the minimum deviation of the ω angle from either 0° or 180°. For residues connected by two peptide bonds the largest twist is used.

3.1.3.3.8 Pepflip peak This is a binary feature that indicates whether there is a positive peak in the difference map at a position that the N or O atoms of a residue could move to if the peptide bond was rotated. A list of positive difference-map peaks was generated using the *map-peaks-around-molecule* function. Each main-chain N and O atom was then examined to see if it could be rotated to any of these peaks by checking distances and angles between the peak and the main-chain atoms. Initial estimates were made for the r.m.s.d. threshold used for peak picking and acceptable ranges for distances and angles. The estimates were then refined using Nelder–Mead minimisation [100] on the full set of 639 structures. The function being minimised was $-\mathrm{TP}+5\mathrm{FP}$, where TP is the number of true positives, *i.e.* residues that have a pepflip peak and a main-chain target correctness of 0, and FP is the number of false positives, *i.e.* residues that have a pepflip peak and a main-chain target correctness of 1. With the minimized parameters, only positive difference-map peaks above 4.45

r.m.s.d. were considered. For the peak to be attributed to the O atom, the distance between the peak and the C atom had to be 0.89–2.75 Å, the distance between the peak and the C^{α} atom had to be 1.01–3.71 Å, the distance between the peak and the C^{α} atom of the next residue had to be 1.84–3.87 Å and the angle between the peak the C atom and the O atom had to be greater than 60.9°. For the peak to be attributed to the N atom the distance between the peak and the C^{α} atom had to be less than 2.09 Å, and the distance between the peak and the O atom of the previous residue had to be less than 1.46 Å.

3.1.3.3.9 Rotamer score These were obtained using the *rotamer-score* function, which uses data from the MolProbity Top 500 database [41]. The most commonly observed rotamer is assigned a score of 100. Other conformations are scored relative to this based on their observed frequencies within the database.

3.1.3.4 Neural network training

The 639 structures were randomly split, using a 4:1 ratio, into a training set of 511 structures with 305594 residues and a test set of 128 structures with 76891 residues. Only residues with side chains longer than C^{β} were used in the side-chain neural network, of which there are 229967 residues (75.3%) in the training set and 57522 (74.8%) in the test set. This excludes glycines and alanines, as well as unknown residues that are built as alanine by *Buccaneer*.

The pre-processing and training procedure was the same for both main-chain and side-chain correctness. If a residue had a missing feature, because it depends on neighbouring residues that may not be present, it was assigned the median value of that feature in the training set. The features in the training set were then transformed to have a mean of 0 and a unit variance. The same transform was applied to the features in the test set using the means and standard deviations from the training set.

Regression was carried out using a multi-layer perceptron (MLP) neural network from *scikit-learn* version 0.21.2 [101], which trains using back-propagation with the square error as a loss function. Both networks had one hidden layer with 10 neurons

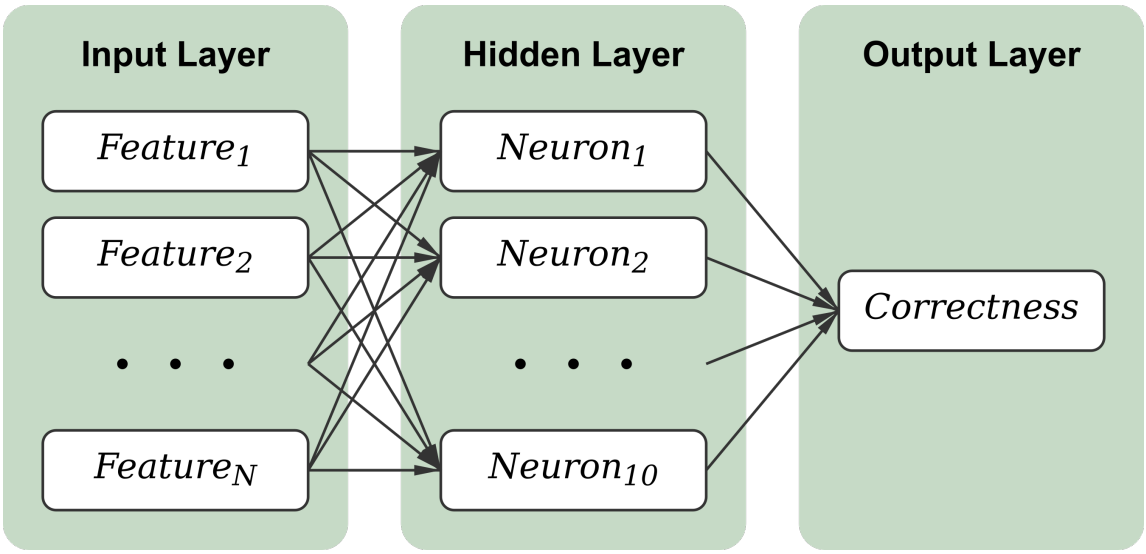

Figure 3.1: Diagram of the neural network. The input layer contains N scaled features (12 for the main-chain network and 9 for the side-chain network), the hidden layer contains ten neurons and the output layer contains only one output with the correctness value. Each arrow has an associated coefficient and intercept that are modified during training.

using the hyperbolic tan function as an activation function, and a single output giving the correctness value without an activation function. Default values were kept for all other parameters, for example the α regularisation term was 0.0001 and optimization was carried out using *Adam* [102] for a maximum of 200 iterations. A diagram of the neural network is shown in Fig. 3.1 and an equation for calculating Correctness from the input features is shown in (3.2), where w_{nk} and c_{nk} are the coefficient and intercept between Feature$_n$ and Neuron$_k$, and w_{ko} and c_{ko} are the coefficient and intercept between and Neuron$_k$ and the output node:

$$\begin{aligned} \text{Neuron}_k &= \tanh\left[\sum_{n=1}^{N}(w_{nk}\cdot\text{Feature}_n + c_{nk})\right], \\ \text{Correctness} &= \sum_{k=1}^{10}(w_{ko}\cdot\text{Neuron}_k + c_{ko}). \end{aligned} \tag{3.2}$$

The trained neural networks were scored on both the training and test sets using the coefficient of determination (COD), which assesses the fit between the predicted and target correctness values. The coefficient of determination is usually referred to as

R^2, but this was avoided owing to confusion with the crystallographic R factor. It varies between 0, where the model is no better than the mean of the target values, and 1, where the model perfectly predicts all target values. Training was repeated 100 times with different random-number seeds and performance was assessed using the mean and standard error in the COD over the test set. The first trained network, with a random seed of 0, was used as the final predictor. To test whether all the features should be included in the network, features were removed one at a time and the training repeated, again using 100 different seeds, to establish the change in the COD.

The final predictor was also assessed on its ability to classify residues in the test set by converting the correctness score to a binary class, where a score of ≥ 0.5 is predicted to be correct. The residues in the test set were then split into true positives (TP) that are actually correct and predicted to be correct, true negatives (TN) that are actually incorrect and predicted to be incorrect, false positives (FP) that are actually incorrect but predicted to be correct, and false negatives (FN) that are actually correct but predicted to be incorrect. Equations (3.3) to (3.10) show a number of quality metrics that were derived from these counts.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{3.3}$$

$$\text{Error} = \frac{\text{FP} + \text{FN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} = 1 - \text{Accuracy}, \tag{3.4}$$

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{3.5}$$

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}, \tag{3.6}$$

$$\text{False-negative rate} = \frac{\text{FN}}{\text{TP} + \text{FN}} = 1 - \text{Sensitivity}, \tag{3.7}$$

$$\text{False-positive rate} = \frac{\text{FP}}{\text{TN} + \text{FP}} = 1 - \text{Specificity}, \tag{3.8}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{3.9}$$

$$F_1 \text{ score} = \frac{\text{TP}}{\text{TP} + \frac{\text{FN}+\text{FP}}{2}}. \tag{3.10}$$

3.1.3.5 *Coot* ML Correctness script

A *Coot* ML Correctness script was created that calculates the features and uses the trained neural networks to obtain the main-chain and side-chain correctness scores for each residue. Machine-learning data were incorporated into the script through an object containing the medians for each feature, the means and variances used for scaling features, and the coefficients and intercepts used by the neural networks. Running this script creates two new menu items in the *Coot* user interface under the heading 'ML Correctness'. The first is a graphical user interface (GUI) that has lists of all the residues along with their correctness scores. Clicking on a residue will move the view in the main window to that location. Owing to the time that it takes to calculate some of the features, the GUI does not update as the model changes, but check boxes are provided so the user can keep track of which issues have been addressed.

The second menu item is an automatic pruning function that deletes whole chains, whole residues and side chains with low correctness scores. Whole chains of up to 20 residues in length are deleted if the mean main-chain correctness for that chain is less than 0.2 times the median main-chain correctness in the full structure. Individual residues and side chains are deleted if the main-chain and side-chain correctness scores, respectively, are less than half of the median for the full structure. After the low-scoring residues have been deleted, isolated residues are also removed. A maximum of 20% of the residues or side chains are deleted at each stage. The pruning function is also available via a scripting interface, where it can be called with custom parameters.

3.1.3.6 *Buccaneer* Pipeline

As described in Section 3.1.3.5, the *Coot* ML Correctness script contains an automatic pruning function that deletes chains, individual residues and side chains with low completeness scores. This function was incorporated into two new versions of the *CCP*4*i*2 *Buccaneer* pipeline that are summarized in Table 3.2. The chain-pruning pipeline has an additional step that prunes whole chains at the end of each iteration, followed by a further five cycles of refinement using *REFMAC*. The full pruning pipeline also starts each iteration, other than the first, by deleting chains, residues and side chains in the model from the previous cycle, running five cycles of *REFMAC*, and passing the updated model and map to *Buccaneer*.

Table 3.2: Summary of the *CCP*4*i*2 of the *Buccaneer* pipeline versions that were tested.

Pipeline	Initial full pruning	Final chain pruning
Released (*CCP*4 7.0.076)	No	No
Chain pruning	No	Yes
Full pruning	Yes	Yes

All three pipelines were tested on 867 structures between 1 and 3.5 Å resolution from the full reduced set. The full reduced set contains 1351 cases but 483 were excluded because the target structures were part of the neural network training set. Another structure, PDB entry 5da8, was excluded because the noncrystallographic symmetry in this case leads to very long run times using the version of *Buccaneer* in *CCP*4 7.0.076; this issue has been addressed in *CCP*4 7.1. The pipelines were run using default parameters starting from the molecular-replacement model.

3.1.4 Results and Discussion

3.1.4.1 Neural network training

The COD for the trained neural network models is shown in Table 3.3 for both the training set and the test set. Values are given as the mean with an uncertainty of one standard error after repeating the training 100 times with different random-number

Table 3.3: Trained neural network COD for the training and test sets. Values are the mean COD after training with 100 different random-number seeds with one standard error in parentheses.

Network	Training-set COD	Test-set COD
Main chain	0.6534(2)	0.6665(2)
Side chain	0.6004(2)	0.6073(2)

seeds. If the COD was much higher for the training set than the test set this could indicate overfitting, but in this case the values for the test set are higher. Overfitting is unlikely due to the large number of residues and small size of the neural network, but there could be some differences between the training and test sets depending on the random split of the 639 structures. The COD is lower for the side-chain network, but this is heavily dependent on the proportion of correct residues. The main-chain sets contain a higher proportion of correct examples so a higher COD is expected.

Although regression was used instead of classification, two classes were obtained using a threshold correctness of 0.5. Confusion matrices, which show the relationship between target (true) correctness and predicted correctness are presented in Fig. 3.2.

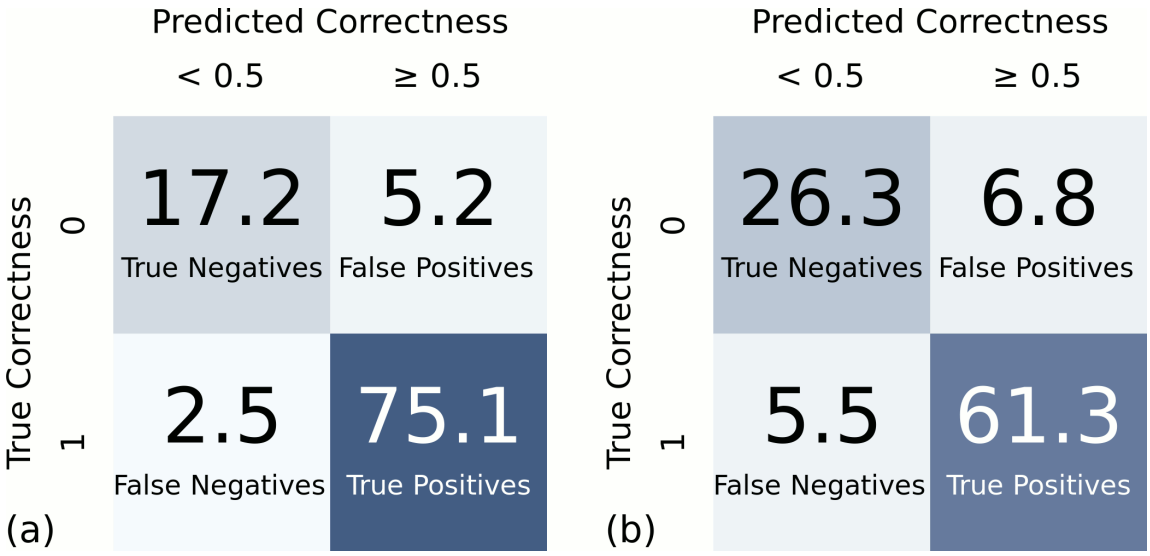

Figure 3.2: Confusion matrices for (a) the main-chain and (b) the side-chain network. Values shown are percentages of residues in the test set.

Table 3.4 shows various quality metrics derived from the number of true positives, true negatives, false positives and false negatives in the test set. Both networks do a good job at identifying correct residues but are less good at identifying incorrect residues, as shown by the difference in the sensitivity (true-positive rate) and specificity (true-negative rate) or, equivalently, by the false-positive rate being much higher than the false-negative rate. This is a symptom of the training data, especially the main-chain data, containing mostly correct residues, so the networks are more likely to assume that a residue is correct. The correctness threshold of 0.5 could be increased for a higher specificity at the cost of lower sensitivity.

Table 3.4: Quality metrics for the main-chain and side-chain neural networks on the residues in the test set, assuming that residues with correctness scores of ≥ 0.5 are predicted to be correct. Equations for these metrics are given in (3.3)–(3.10).

Network	Main chain	Side chain
Accuracy (%)	92	88
Error (%)	8	12
Sensitivity (%)	97	92
Specificity (%)	77	79
False-negative rate (%)	3	8
False-positive rate (%)	23	21
Precision (%)	94	90
F_1 score (%)	95	91

The simplistic method of determining the target correctness needs to be taken into account when comparing the true and predicted correctness values. This was performed by comparing each residue with the deposited structure. If any atom was more than 1 Å away it was marked as incorrect. Firstly, the cutoff was not chosen based on any analysis of existing data. It was just assumed that at both high and low resolution the same conformation is usually closer than 1 Å and different conformations are usually further apart than 1 Å after refinement. Another issue is that not all acceptable conformations will be modelled in the deposited structure, especially for flexible side chains at low resolution, when it is hard to distinguish multiple conformations. In addition, the deposited model may also contain errors. Structures were filtered based on overall quality indicators from the wwPDB validation report, but local problems may still exist. However, the

Table 3.5: Test-set COD for the main-chain neural network after it has been trained with individual features removed. Values are the mean COD after training with 100 different random-number seeds with one standard error in parentheses.

Missing main-chain feature	Test-set COD	Decrease
No missing feature	0.6665 (2)	0.0000
Pepflip peak	0.6646 (3)	0.0019
Maximum B-factor Z-score	0.6642 (2)	0.0023
Difference density Z-score at the next C^{α}	0.6624 (2)	0.0041
Maximum B-factor change Z-score	0.6621 (2)	0.0044
Minimum best density Z-score	0.6613 (2)	0.0052
Maximum peptide twist	0.6604 (3)	0.0061
Minimum difference density Z-score	0.6598 (2)	0.0067
Maximum atom overlap	0.6592 (2)	0.0073
Mean best density Z-score	0.6570 (3)	0.0095
Ramachandran score	0.6563 (2)	0.0102
Resolution	0.6377 (3)	0.0288
Map-to-model correlation	0.6087 (3)	0.0578

Table 3.6: Test-set COD for the side-chain neural network after it has been trained with individual features removed. Values are the mean COD after training with 100 different random-number seeds with one standard error in parentheses.

Missing side-chain feature	Test-set COD	Decrease
No missing feature	0.6073 (2)	0.0000
Minimum difference density Z-score	0.6038 (2)	0.0035
Maximum atom overlap	0.6027 (2)	0.0046
Maximum B-factor change Z-score	0.6021 (2)	0.0052
Minimum best density Z-score	0.6000 (2)	0.0073
Mean best density Z-score	0.5968 (2)	0.0105
Maximum B-factor Z-score	0.5901 (2)	0.0172
Resolution	0.5874 (2)	0.0199
Rotamer score	0.5835 (2)	0.0238
Map-to-model correlation	0.5566 (2)	0.0507

training and test sets are still useful for machine learning, and a larger, noisy data set can even produce a better predictive model than a smaller, less noisy one.

For both neural networks, the input features were removed one at a time and the training was repeated to establish the magnitude and significance of the change in the COD. Table 3.5 shows the results for the main-chain features and Table 3.6 shows the results for the side-chain features. However, the change in the COD depends both on how much useful information a feature has and how well it correlates with other features. If removing a feature leads to no decrease in the COD then it either does not provide information that is useful for identifying incorrect residues or the information is duplicated in another feature. In either case the feature can be removed. If removing a feature causes a large reduction in the COD then it is both useful and independent. All of the features give a significant reduction in the COD when removed, so they are all providing some useful information.

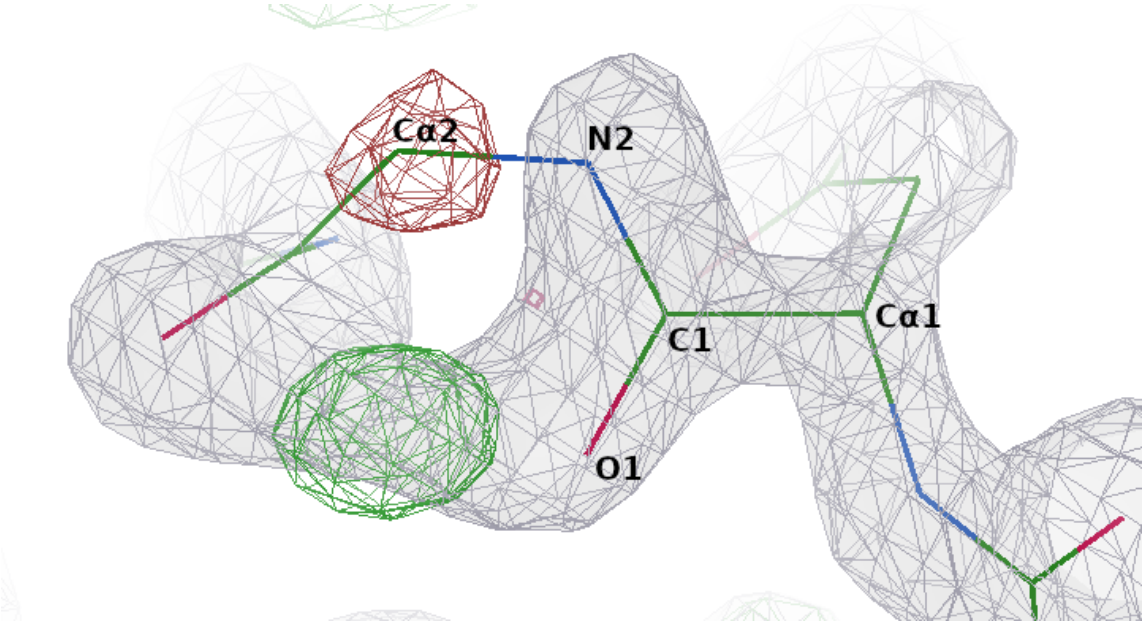

Figure 3.3: A reversed amide bond where negative difference density at the next C^{α} suggests an error in the previous residue. The example is a peptide bond between asparagine and glycine in a 1.86 Å resolution structure built by *Buccaneer* that was not used in this study. The $2mF_o - DF_c$ map is shown in grey. The positive and negative contours of the $mF_o - DF_c$ map are shown as green and red, respectively.

The pepflip peak and next C^{α} difference density features in the main-chain neural network are quite unusual. They are not general validation metrics, but are designed

to highlight specific errors that may occur during model building. The parameter minimization for the pepflip peak feature, as described in Section 3.1.3.3.8, resulted in a score of −3574, meaning there are at least 3574 residues (0.93%) with a pepflip peak and a target correctness of 0. Fig. 3.3 shows an example where it is useful to look at the density at the next residue. The amide oxygen and nitrogen need to swap positions, but both still fit the density well. However, the negative difference density at the next C^{α} suggests that there is something wrong with the previous residue.

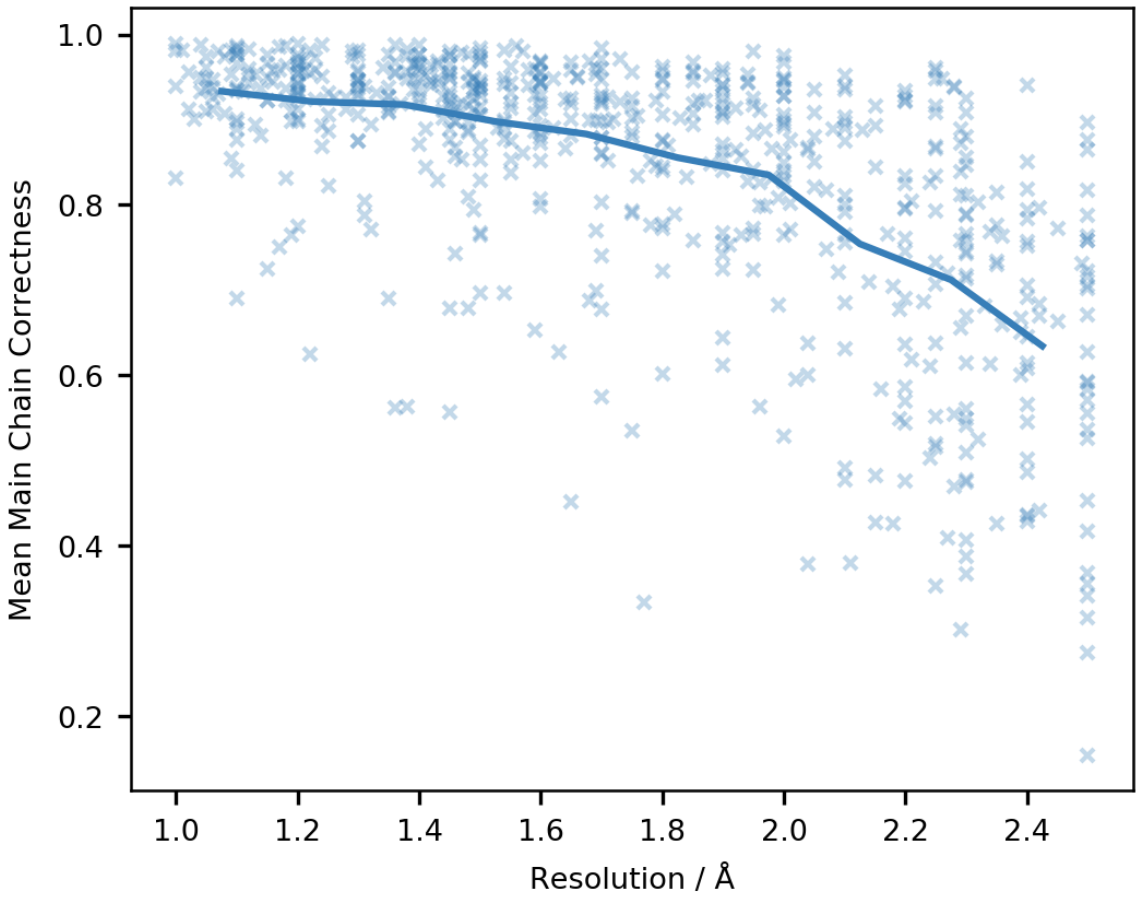

Figure 3.4: Resolution and mean main-chain target correctness for 639 structures in the training and test sets. The mean value for 10 resolution bins is shown as a line.

Resolution is an interesting feature because it varies per structure and not per residue so, within a structure, it does not give any information about which residues are correct if used by itself. It was included to adjust the weights of other metrics; for example, at low resolution it is harder to distinguish side-chain positions and it is expected that rotamer score will be given more weight as uncommon conformations

should only be built if the evidence for them is sufficient. However, the performance of *Buccaneer* is resolution dependent. Fig. 3.4 shows that there is a higher proportion of incorrect residues at lower resolution, so the resolution feature will likely penalize the scores of residues in lower resolution structures. This is compensated for during automatic pruning by deleting residues with correctness values less than a fraction of the median value in the structure.

3.1.4.2 *Buccaneer* pipeline

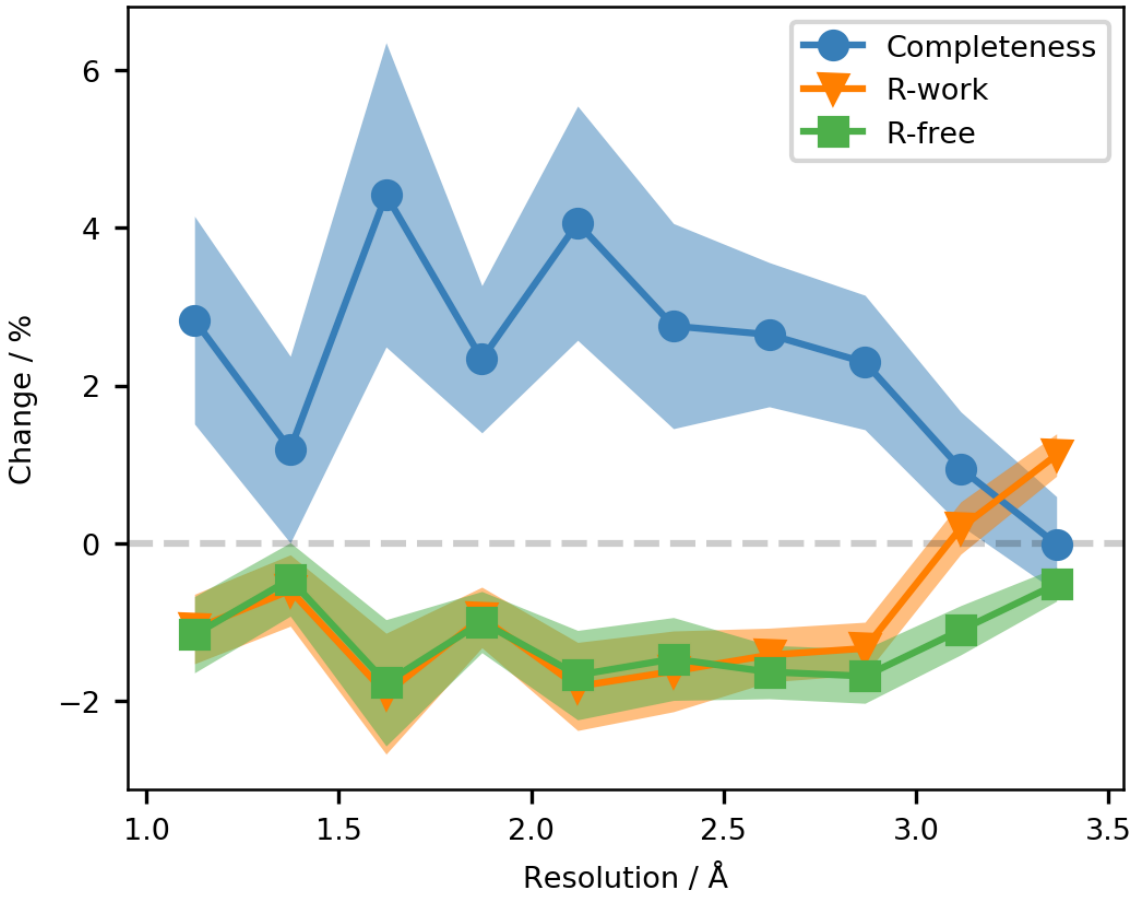

Figure 3.5: Change in completeness, R_{work} and R_{free} between the released pipeline and the chain-pruning pipeline. The 867 structures were divided into 10 resolution bins and the mean and standard error of the change for each bin is shown.

Fig. 3.5 shows the change in completeness, R_{work} and R_{free} of the models produced by the *Buccaneer* pipeline on the addition of a chain-pruning step at the end of each iteration. Completeness is the percentage of residues in the refined deposited

structure that have a matching residue in the model. Two residues were only considered to match if the N, C^{α} and C positions were all within 1 Å. At a resolution of 2.8 Å or better, completeness improves by 2–3% and R factors improve by 1–2%. Performance may be slightly less at very high resolution but it is hard to tell due to the noise in this region. At lower resolutions there is less improvement but R_{free} still decreases. The gap between R_{free} and R_{work} widens at low resolution, which suggests that deleting some of the less correct chains is reducing the overfitting.

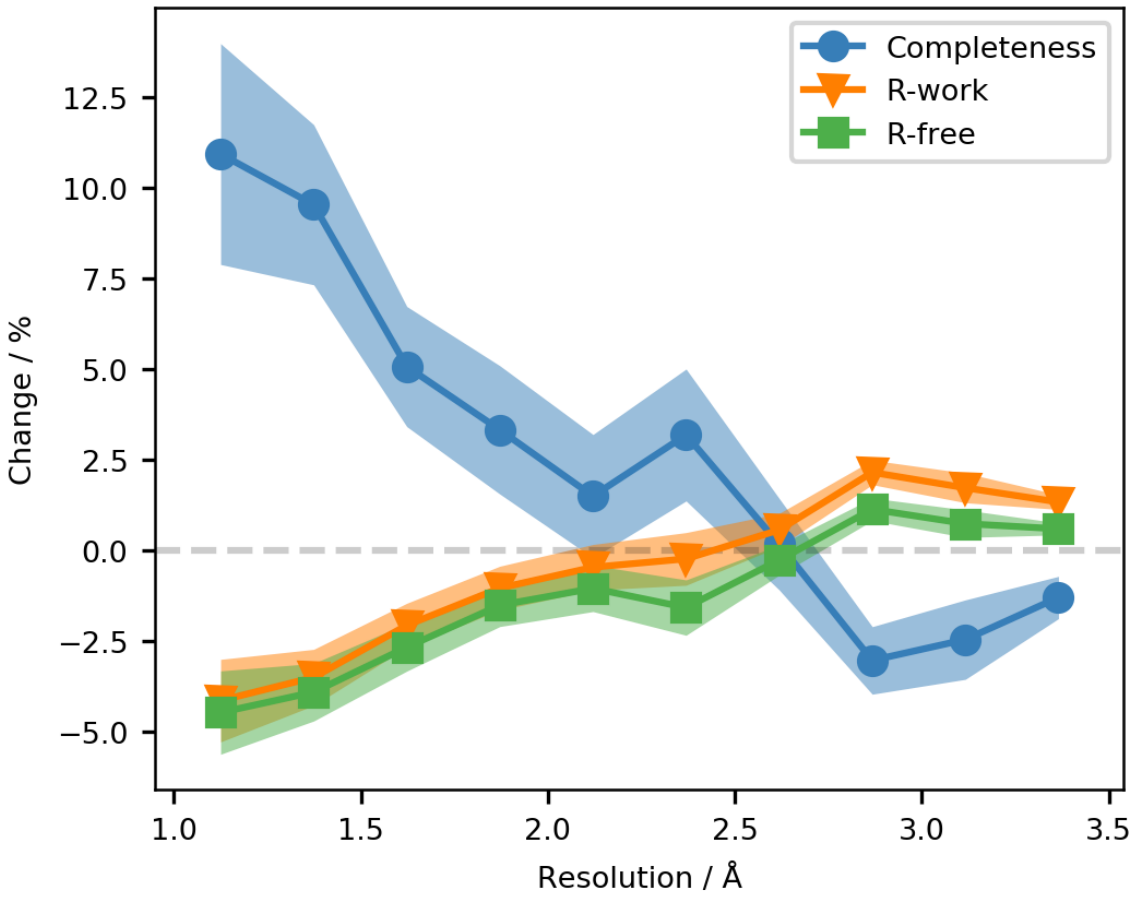

Figure 3.6: Change in completeness, R_{work} and R_{free} between the chain-pruning pipeline and the full pruning pipeline. The 867 structures were divided into 10 resolution bins and the mean and standard error of the change for each bin is shown.

Fig. 3.6 shows the change in completeness, R_{work} and R_{free} of the pipeline models if an additional pruning step is added at the start of each iteration, other than the first, that prunes chains, residues and side chains. The effect of this change varies dramatically with resolution. The greatest improvement is seen at high resolution,

where completeness improves by around 10% and R-factors decrease by around 4% on average. The improvement quickly drops off at lower resolutions, with the full pruning step leading to worse pipeline performance below 2.6 Å resolution. Again, there is a difference between the R_{work} and R_{free} that shows pruning reduces overfitting.

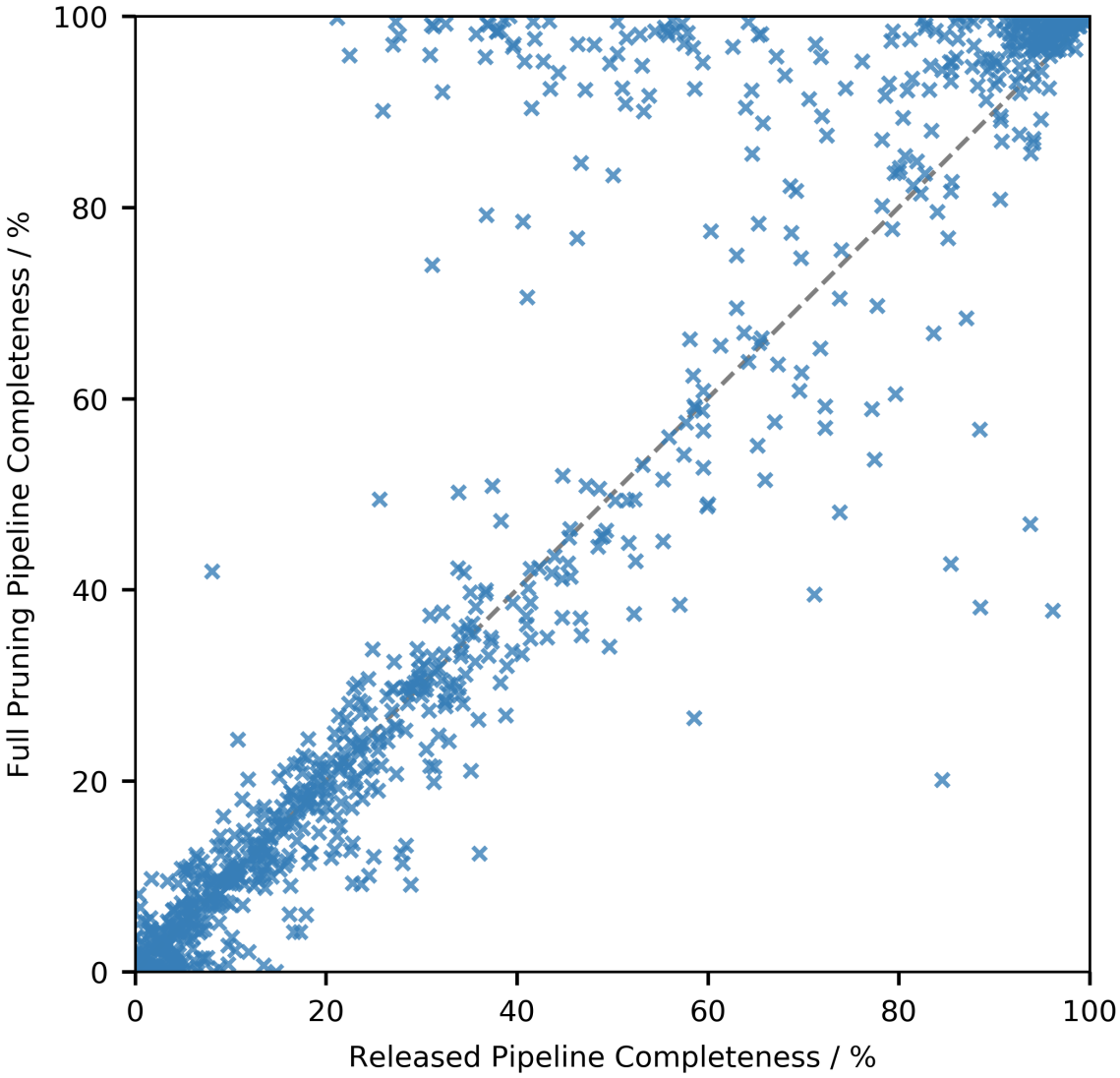

Figure 3.7: Completeness of the models from the released pipeline and the full pruning pipeline for the 867 structures tested.

Fig. 3.7 compares the completeness of the models from the released pipeline and the full pruning pipeline. There are 336 structures (39%) where the model from both pipelines had $< 20\%$ completeness. Out of these structures which performed badly

in both pipeline versions, 173 (51%) were more complete in the released pipeline and 135 (40%) were more complete in the full pruning pipeline. At the other end of the scale, there are 183 structures where both pipelines produced a model with $> 80\%$ completeness. Of these relatively complete structures, only 23 (13%) were more complete in the released pipeline while 153 (84%) were more complete in the full pruning pipeline. There are also 63 structures (7%) at the top of Figure 3.7 where the model from the full pruning pipeline has $> 90\%$ completeness and the model from the released pipeline has $< 70\%$ completeness, including an extreme example where the completeness increases from 21% to 100%.

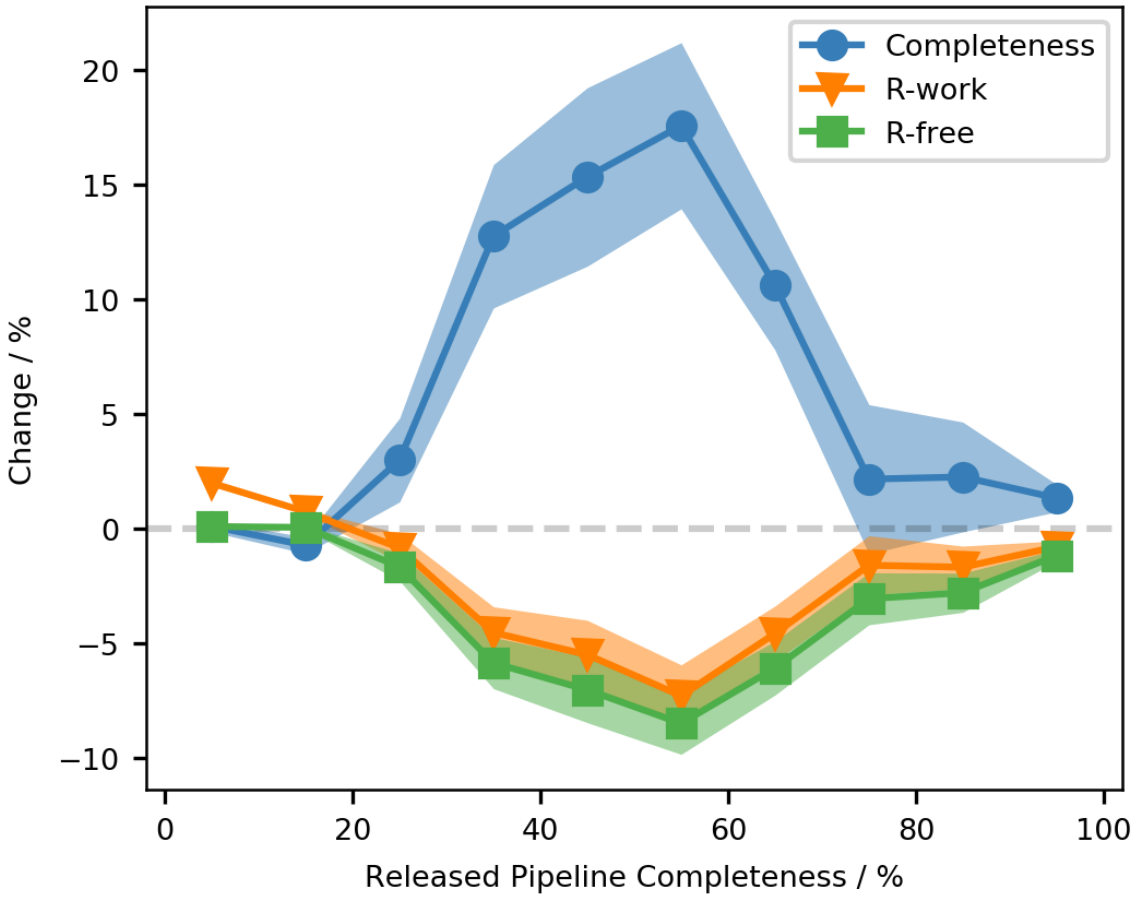

Figure 3.8: Change in completeness, R_{work} and R_{free} between the released pipeline and the full pruning pipeline against the completeness of the model from the released pruning pipeline. The 867 structures were divided into 10 completeness bins and the mean and standard error of the change for each bin is shown.

An overview of the effect of the new pruning steps at different levels of completeness is shown in Fig. 3.8. For structures where the released pipeline

produced models with around 50% completeness, the full pruning pipeline produced models with substantially higher completeness and lower R_{free} values on average. At higher levels of completeness there is much less room for improvement, but a small increase in completeness and decrease in R-factors is still observed.

An example with high completeness in both pipeline versions is PDB entry 4wn5 [103] at 1.15 Å resolution. The model produced by the released pipeline has a completeness of 90.14% and the model produced by the full pruning pipeline has a completeness

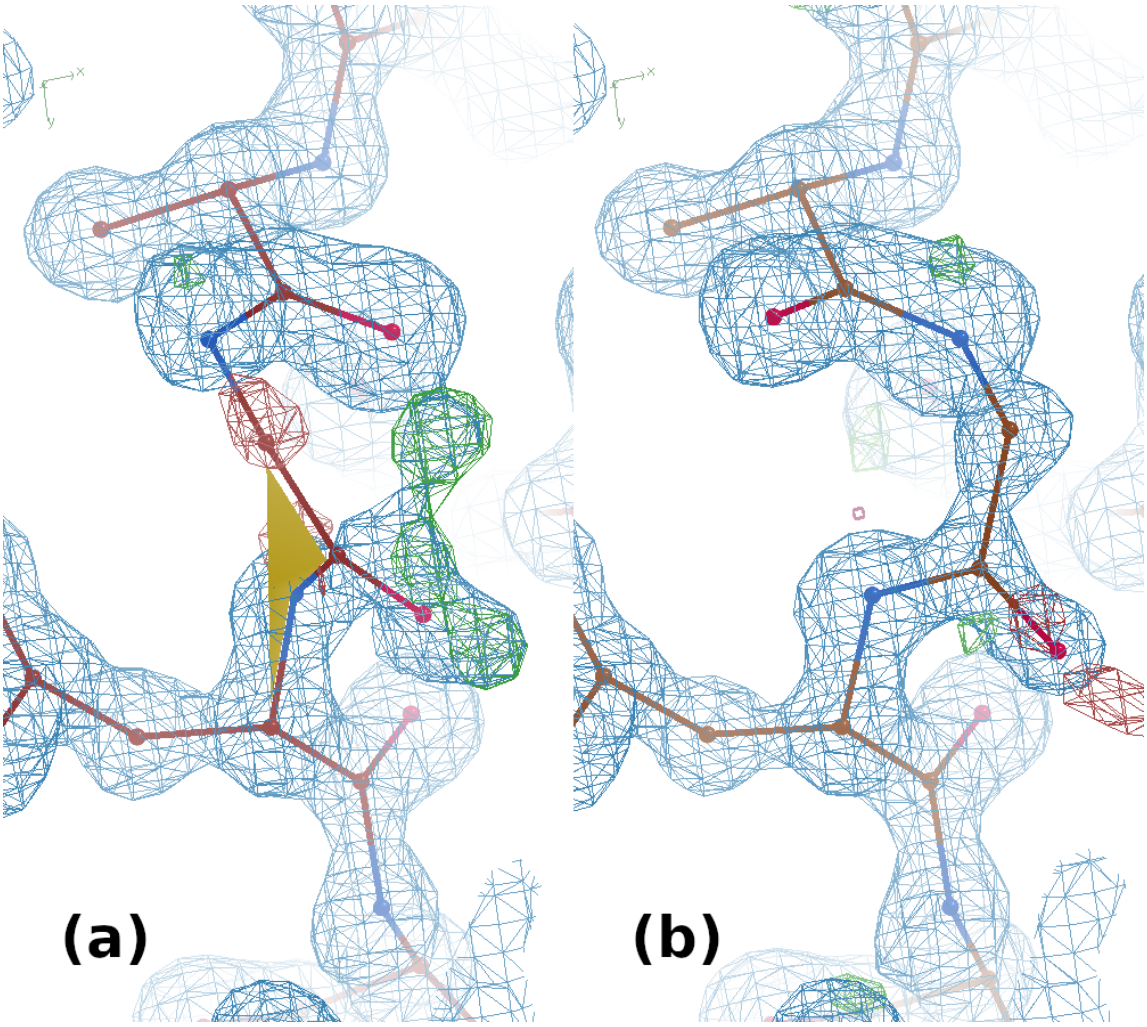

Figure 3.9: A section of PDB entry 4wn5 in (a) the model built by the released pipeline and (b) the model built by the full pruning pipeline. The $2mF_o - DF_c$ map is shown in blue. The positive and negative contours of the $mF_o - DF_c$ map are shown as green and red, respectively. The yellow shaded area shows that the peptide bond is twisted, *i.e.* the ω angle is between 30° and 150°.

of 98.59%. Much of the improvement in completeness is not owing to new parts of the structure being built, but because errors in the backbone conformations have been corrected. A section of both models is shown in Fig. 3.9. The peptide between alanine and glycine at the top of the Fig. 3.9(a) is reversed, similar to the example shown in Fig. 3.3, so the glycine C^{α} atom is out of the density. The next peptide bond after glycine is also twisted, as indicated by the yellow shaded area. Both of these factors will contribute to a low correctness score. Deleting these residues allows *Buccaneer* to build the model correctly.

When using the predicted correctness scores for pruning, a decision needs to be made about the threshold used for selection. Because the scores cannot predict correctness with 100% accuracy, any chosen threshold will prune some correct residues and leave some incorrect ones. The thresholds tested were 0.2 times the median for whole chains and 0.5 times the median for residues and side chains. Other thresholds have not yet been tested, but the optimum value is likely to depend on the stage of model building. More caution needs to be taken at the end of the pipeline because it is usually easier for the user to fix an incorrect conformation than to build a missing feature. If pruning is done during the pipeline, before further cycles of *Buccaneer*, then it can be less cautious because correct residues that are mistakenly deleted should be automatically rebuilt. However, a balance is still required because deleting more correct residues than incorrect residues can reduce the quality of phases and make building more challenging.

3.1.5 Conclusion

The correctness of 382485 residues in 639 *Buccaneer* models was assigned by automatic comparison with the models deposited in the PDB for those structures. Residues were given correctness values of either 0 or 1, which was performed separately for the main chain and side chains. This method of producing target correctness values is not perfect, but the vast majority of residues will be labelled correctly. Manual labelling of each residue is too slow and it is important to have a large number of data points for the machine learning to work well.

Regression was carried out for 511 of the structures using two neural networks to predict the correctness by combining many features of each residue. The input

features include map-to-model correlation, density values, B factors, clashes, Ramachandran scores, rotamer scores and resolution. Using regression instead of classification means intermediate correctness scores can be obtained, hopefully for residues where it is not obvious if the conformation is correct or not. If scores of less than 0.5 are classed as incorrect, the trained networks correctly categorise 92.3% of main chain atoms and 87.6% of side chains in the set of 128 structures that were not used for training. The correctness predictions show no sign of overfitting, but they are expected to work best on structures similar to the ones used in the training set, i.e. mostly complete structures with resolutions better than 2.5 Å.

A *Coot* ML Correctness script was written to calculate the predicted correctness values and show them to the user as a validation tool. This helps to quickly identify the worst parts of a structure for further examination. The aim is not to have high correctness scores for the whole structure as, owing to the reliance on Z-scores in the input features, the score is relative to the whole structure. Deleting poor parts of the structure will decrease the correctness scores for the remaining model. The script also contains an automatic pruning function for deleting whole chains, residues and side chains with low correctness scores. It can be called with default parameters from the *Coot* graphical user interface or with custom parameters via the scripting interface.

The pruning function was incorporated into the *Buccaneer* pipeline in *CCP*4*i*2 to prune whole chains at the end of each cycle and also individual residues and side chains at the beginning of each cycle. The pipeline changes were tested on 867 structures at 1–3.5 Å resolution. The final pruning of whole chains leads to improved models and the improvement is not very dependent on resolution. In contrast, initial pruning of residues and side chains gives large improvements at high resolution but often leads to worse models at low resolution. Hence, it is only recommended to include residue-level pruning when the resolution is better than 2.6 Å. There are many structures that have changed from being partially built to almost fully built with the addition of the new pruning steps.

3.1.6 Future Work

Although the addition of the pruning step leads to improvements in the *Buccaneer* pipeline, the correctness score is far from optimal. One of the main problems is that machine learning was carried out as a mixture of classification and regression. Regression was used in order to obtain a continuous correctness score instead of a binary classification. However, as the target data were categorical, *i.e.* all samples had a target correctness of 1 or 0, it would have been better to use a classifier and obtain continuous values in the form of the predicted probabilities for each class. Another option would be to perform regression against a different, continuous target; for example, the r.m.s.d. between the atoms of the query structure and the reference structure. This has the advantage that no cutoff has to be chosen, although it may also have difficulties in that a residue built into the solvent 5 Å away from the structure is no different to one 10 Å away. Classification using the r.m.s.d. could be a solution to this, but it does not have to be binary: for example, the classes could be an r.m.s.d. of < 0.5 Å, < 1 Å, < 2 Å and ≥ 2 Å.

After choosing the training target and either classification or regression, the model should be examined in more detail. For this study a neural network model was used, and hyperparameters such as the learning rate and the regularisation term were kept at their default values. However, other models such as a decision tree or a random forest should also be explored as they may produce better results, and hyperparameters should be tuned for optimum performance.

The structures built by *Buccaneer* in the easy reduced set contain mostly correct residues and side chains. This imbalance means that the networks will be better at identifying correct residues than incorrect ones and explains the high false-positive rate. Incorrect residues are identified, but these are likely to be obvious errors such as residues built into the solvent. Resampling should be considered to either undersample the correct residues or oversample the incorrect residues. More difficult cases could be included, but these need to be chosen carefully. Models built by *Buccaneer* are often either largely correct or composed of small fragments built into noise, and the incorrect residues in these two extremes will have very different features. The correctness score was not intended to help in the latter case, where better initial phases may be required.

As mentioned in Section 3.1.5, owing to the use of Z-scores in the features, the correctness of a residue is not only dependent on its immediate environment but on the whole structure. This is counterintuitive and should be changed. Map values will still be needed in the features, but dependencies on the absolute scale of the map or the solvent content of the structure may be introduced depending on how they are measured.

It would also be beneficial to have a correctness score using features that can be calculated quickly for an individual residue for the purpose of providing feedback during model building. This could be provided in addition to a more accurate score that is only calculated after refinement. For the quick score, difference-map values should not be used as they would need to be recalculated after the model changes. It may also be necessary to remove B factors from the features unless they can be obtained quickly, for example using shift-field refinement [13]. Other features that are missing from the current implementation should be investigated. It is likely that more generic geometric scores would be helpful, such as the χ^2 values of the bond and angle restraints displayed in *Coot* after real-space refinement.

3.1.7 Availability

The *Coot* ML Correctness script and scripts used for training the neural networks are available at https://doi.org/10.15124/44145f0a-5d82-4604-9494-7cf71190bd82. *Coot* version 0.8.9.2 or later is required for the script to work. The new pruning steps added to the *Buccaneer* pipeline in *CCP*4*i*2 will be available in *CCP*4 version 7.1. They can be turned on and off from the Options tab on the Input page of the task.

3.1.8 Acknowledgements

The authors would like to thank Paul Emsley for help and advice with *Coot*. This work was supported by a White Rose BBSRC DTP in Mechanistic Biology (BB/M011151/1 to Paul S. Bond) and a BBSRC grant (BB/S005099/1 to Kevin D. Cowtan).

3.2 Summary

The work presented in this chapter was a first attempt at combining multiple per-residue validation metrics into composite main-chain and side-chain correctness scores using machine learning. The scores have already proved useful for pruning models built by BUCCANEER, leading to better models being produced by the BUCCANEER pipeline in CCP4i2. To improve this work, the current binary correctness target should be changed to a continuous target that is suitable for regression or more classes should be added to distinguish between residues that are partially correct and residues that are built into noise. The input features should also be modified to include bond and angle deviations and to create absolute scores that are dependent only on the residue and its immediate environment instead of relative scores that are dependent on the whole structure. Finally, two separate scores should be developed: one that can be calculated quickly for immediate feedback during model building and a more accurate score that requires global refinement to be performed first.

Chapter 4

Pipeline Developments

BUCCANEER [37] is rarely used in isolation as it only builds a protein model into an input map. It does not perform any global refinement of coordinates or B-factors and it does not update the map as the model improves. This is especially important in crystallography as the initial map may have poor phases that make model building difficult. Therefore, in most cases BUCCANEER is used as part of a model building pipeline that performs cycles of model building and refinement, which improves the model geometry and fit to density and produces an updated map that can be passed to the next building cycle.

BUCCANEER and its pipelines are distributed with the CCP4 software suite [84]. The oldest pipeline is the one available via the CCP4i graphical user interface (GUI) [94], which performs a fixed number of iterations of BUCCANEER and REFMAC5 [11]. Until recent versions, the pipeline in the newer CCP4i2 GUI [91] was largely a re-implementation of this original pipeline if default settings were used. The CCP4i2 pipeline also has more options for advanced functionality, for example to include a COOT [15] real-space operation or to use ProSMART [104] restraints from a high resolution structure during refinement. BUCCANEER is also used in other software pipelines such as CRANK2 [92], CAB [77] and CCP4Build, which is a new model building pipeline available in CCP4Cloud [97].

This chapter covers improvements made to the BUCCANEER pipeline in CCP4i2 and the development of ModelCraft, a new GUI-independent model building pipeline.

4.1 CCP4i2

The addition of model pruning steps to the CCP4i2 pipeline was discussed in the previous chapter on model correctness. This section covers developments made prior to this. There have been additional subsequent changes to the pipeline. In the most recent version, the COOT real-space operation to add waters is turned on by default and SHEETBEND has been included at the start of the pipeline when refining an input molecular replacement model [14]. These recent developments are discussed in more detail in the next section on ModelCraft because they were implemented in that pipeline before the changes were introduced to CCP4i2.

4.1.1 CCP4i vs. CCP4i2

Before this PhD project, the BUCCANEER pipelines in CCP4i and CCP4i2 were broadly similar when run with default parameters except for some small differences that were introduced over time. However, at that time testing the impact of the changes had been limited to small scale testing using the interface. Recent developments mean it is now possible to run the CCP4i2 pipeline on the command line so testing on a large number of structures is much easier. To measure the difference in performance between the two pipelines, both were tested on 202 experimental phasing cases and 63 molecular replacement cases using CCP4 version 7.0.059. Table 4.1 compares the pipeline results using R-factor, R-free and completeness by residue, which is the percentage of built residues that are sequenced.

Table 4.1: R-factor, R-free and completeness of the BUCCANEER pipeline in the CCP4i and CCP4i2 GUIs. Values are the mean ± one standard error over all 265 test cases.

	R-factor / %	R-free / %	Completeness / %
CCP4i	30.00 ± 0.41	33.68 ± 0.47	88.96 ± 1.25
CCP4i2	31.00 ± 0.49	35.77 ± 0.52	88.18 ± 1.22
Difference	1.00 ± 0.24	2.09 ± 0.23	-0.77 ± 0.66

R-factor and R-free were significantly worse in CCP4i2. Completeness was also worse

but the difference is not as significant. It is important to use the standard error of the difference when comparing results. For example, the mean completeness of the CCP4i models was 88.96% and the mean completeness of the CCP4i2 models was 88.18%, giving a difference of 0.77%. This difference may seem small when compared to the uncertainty in the means themselves, which are around 1.2%. However, the uncertainty in mean completeness is large because some cases in the test set have high completeness and some have low completeness. Using the standard error of the differences for individual test cases corrects for this and shows the uncertainty in the mean difference between the pipelines.

After looking at the log files for differences between the pipelines, it was noted that the first BUCCANEER job in the pipeline had two options that were different between the interfaces. Firstly, CCP4i passed a free-R flag to BUCCANEER so that the free reflections are not used when generating the density map, but CCP4i2 let BUCCANEER generate maps using all the reflections. Secondly, CCP4i2 passed the *model-filter* keyword to BUCCANEER to remove residues in poor density and protein chains with less than 6 residues, although this will not affect building on the first cycle unless an input model is supplied. A change was made to the CCP4i2 pipeline to pass the free-R flag to BUCCANEER. This version is labelled CCP4i2-free. Table 4.2 shows the difference in pipeline performance owing to this change. There may be a slight worsening of R-factor and completeness when the free reflections are omitted in map generation, but this does not seem to be the major difference between CCP4i and CCP4i2.

Table 4.2: R-factor, R-free and completeness of the BUCCANEER pipeline in CCP4i2 and CCP4i2-free. Values are the mean ± one standard error over all 265 test cases.

	R-factor / %	R-free / %	Completeness / %
CCP4i2	31.00 ± 0.49	35.77 ± 0.52	88.18 ± 1.22
CCP4i2-free	31.17 ± 0.49	35.85 ± 0.53	87.47 ± 1.26
Difference	0.17 ± 0.20	0.09 ± 0.19	-0.71 ± 0.63

The REFMAC jobs in the pipeline had more options that were different between CCP4i and CCP4i2:

- CCP4i2 used 20 cycles of refinement and CCP4i only used 10 cycles.
- CCP4i2 started with bulk scaling without solvent and changed to using simple scaling with explicit solvent once the R-factor fell below 30%. CCP4i used simple scaling with explicit solvent throughout the pipeline.
- CCP4i2 generated riding hydrogen atoms for use during refinement and CCP4i did not use hydrogen atoms.

A new version of the CCP4i2 pipeline was written that changed all of these options to be the same as CCP4i. This is labelled CCP4i2-mimic. Results comparing CCP4i and CCP4i2-mimic are shown in Table 4.3. The performance of the two pipelines is now more comparable. R-factor is lower in the new CCP4i2 version but R-free is slightly higher. Completeness has also increased slightly. The gap between R-work and R-free is larger in CCP4i2, meaning that there is more overfitting, but this was also the case before the changes were made, as can be seen in Table 4.1.

Table 4.3: R-factor, R-free and completeness of the BUCCANEER pipeline in CCP4i and CCP4i2-mimic. Values are the mean ± one standard error over all 265 test cases.

	R-factor / %	R-free / %	Completeness / %
CCP4i	30.00 ± 0.41	33.68 ± 0.47	88.96 ± 1.25
CCP4i2-mimic	29.26 ± 0.43	34.04 ± 0.47	89.64 ± 1.16
Difference	-0.74 ± 0.14	0.36 ± 0.15	0.68 ± 0.55

The performance of CCP4i2-mimic was significantly improved compared to the previous version and the changes were released. However, there are some remaining differences that mean the two pipelines do not give identical results. For cases starting from experimental phasing, the initial phases are passed to REFMAC for MLHL refinement. This happens throughout the pipeline in CCP4i, but in CCP4i2 the phase restraints are released when the R-factor falls below 35%. There are some additional differences caused by the handling of MTZ files. In CCP4i2, the structure factor amplitudes passed to BUCCANEER are always the same as the input amplitudes, but in subsequent cycles of CCP4i, BUCCANEER is given the amplitudes from the last REFMAC job, which differ due to scaling.

In order to see the effect of some of the changes individually, two new versions of CCP4i2 were derived from CCP4i2-mimic. One reversed the change to riding hydrogen atom generation, which will be called CCP4i2-hydrogen. The other reversed the change to the solvent and scaling options, which will be called CCP4i2-scaling. Comparisons of these two versions with CCP4i2-mimic are shown in Tables 4.4 and 4.5. This shows that removing the hydrogen atom generation gave some improvement to the pipeline, but changing the scaling options to always use simple scaling had the most impact. The changes made in CCP4i2-mimic have since been released and are used in the latest version of CCP4i2.

Table 4.4: R-factor, R-free and completeness of the BUCCANEER pipeline in CCP4i2-hydrogen and CCP4i2-mimic. Values are the mean ± one standard error over all 265 test cases.

	R-factor / %	R-free / %	Completeness / %
CCP4i2-hydrogen	30.03 ± 0.46	34.59 ± 0.49	89.40 ± 1.18
CCP4i2-mimic	29.26 ± 0.43	34.04 ± 0.47	89.64 ± 1.16
Difference	-0.77 ± 0.15	-0.56 ± 0.17	0.23 ± 0.45

Table 4.5: R-factor, R-free and completeness of the BUCCANEER pipeline in CCP4i2-scaling and CCP4i2-mimic. Values are the mean ± one standard error over all 265 test cases.

	R-factor / %	R-free / %	Completeness / %
CCP4i2-scaling	31.01 ± 0.48	35.73 ± 0.52	87.98 ± 1.24
CCP4i2-mimic	29.26 ± 0.43	34.04 ± 0.47	89.64 ± 1.16
Difference	-1.76 ± 0.22	-1.69 ± 0.22	1.66 ± 0.56

4.1.2 Automatic Stopping

The original CCP4i pipeline runs for 5 cycles by default and outputs the model from the final cycle. Some cases may benefit from running more cycles than this. However, there is a trade off between the quality of the model produced by the pipeline and the length of time it takes to run, and a more intelligent way of deciding when to stop the pipeline is required. The pipeline should run longer in difficult cases where the

model is still improving and finish earlier in cases where extra cycles are not needed. In addition, due to the stochastic nature of the model building process, there is no guarantee that the model produced in the final cycle will be the best model produced during the pipeline.

Some other BUCCANEER pipelines have already implemented methods to tackle these problems. The CAB pipeline by Burla *et al.* from the Institute of Crystallography, Bari, Italy [77] runs for a maximum of 20 iterations, with each iteration consisting 5 cycles of the CCP4i pipeline followed by a novel phase combination procedure. It terminates once there is an increase in R-free, subject to R-free being less than 40% and sequence coverage being greater than 85%. The model with the lowest R-free is chosen for final refinement. The whole pipeline is repeated with modified input phases if the final model does not have sufficient sequence coverage, meaning up to 200 runs of BUCCANEER can be performed. The CCP4Build pipeline runs for fewer cycles than CAB, but each cycle has many steps using PARROT, BUCCANEER, EDSTATS, COOT and REFMAC. It will terminate if R-free, number of residues built and electron density correlation coefficient (EDCC) have not improved in a set number of cycles. Up to four output models are produced with the lowest R-free, the highest EDCC, the most residues and the least number of fragments.

4.1.2.1 Method

The BUCCANEER pipeline in CCP4i2 was modified to run for a maximum of 25 cycles and to stop automatically if the model is not improving. The model from the final REFMAC run at the end of each cycle is analysed using the following metrics:

- R-work
- Number of built residues
- Number of sequenced residues
- Number of chain fragments
- Length of the longest fragment

The model is marked as a possible improvement if any of these metrics have improved by more than 2% of their previous best value. If the pipeline goes 4 cycles without such an improvement it will stop. The size of the improvement required was tested at 2% and 5% and the number of cycles without improvement at 3, 4 and 5. Values of 2% and 4 cycles were chosen as they seemed to provide the best compromise between model completeness and computational cost. Another change was to output the model from the cycle with the lowest R-free instead of simply using the model from the last cycle. These changes were released in CCP4 7.0.069.

The CCP4i2 pipeline in CCP4 7.0.073 was tested on 548 datasets, of which 195 were from experimental phasing and 353 from either molecular replacement or superposition of a homologue using GESAMT. The pipeline was tested using three different settings:

- Up to 25 cycles with the option to stop automatically (default).
- Exactly 5 cycles without the option to stop automatically (5-cycle).
- Exactly 25 cycles without the option to stop automatically (25-cycle).

4.1.2.2 Results

Two out of the 548 datasets gave an error during the pipeline and so have been excluded from this analysis. Figure 4.1a compares the completeness of the models produced by the 5-cycle and 25-cycle pipelines. Completeness is the percentage of residues in the deposited model that have a matching residue in the pipeline model, where a residue is considered matching if it is the same type and the N, Cα and C positions are all within 1 Å of the deposited position. The majority of points are very close to the diagonal line where the extra cycles did not lead to a more complete model. These occur in the bottom left, where both pipelines produced an incomplete model, and in the top right, where both pipelines produced a complete model. However, there are also a considerable number above the diagonal where the 25-cycle pipeline gave an improvement. A small number of points are below the diagonal, which are datasets where a less complete model with a lower R-free was generated after the first 5 cycles, although cases where completeness is noticeably lower are already very incomplete.

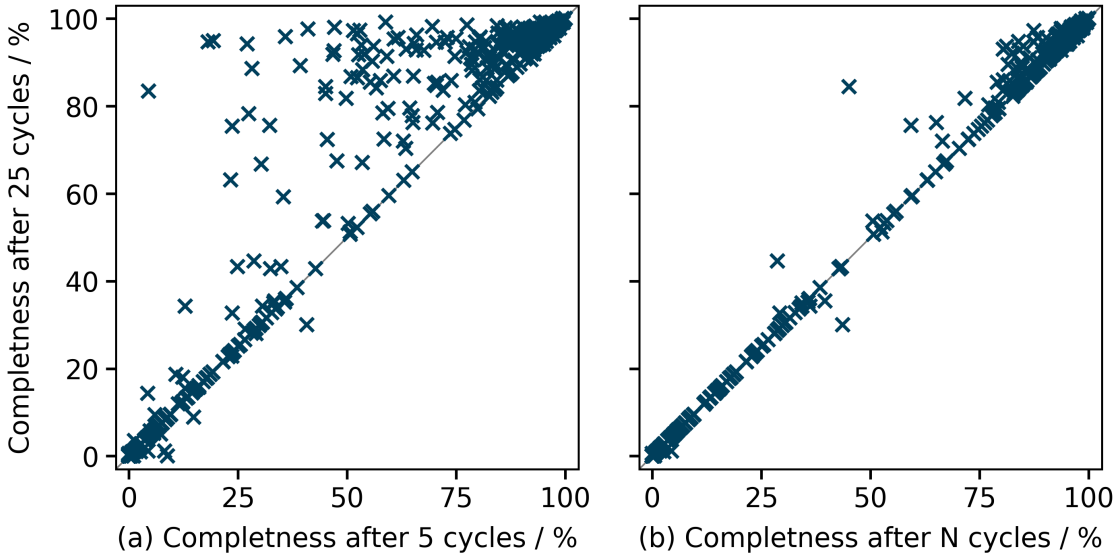

Figure 4.1: Completeness of the models produced by the CCP4i2 7.0.073 pipeline (a) after 5 cycles compared to after 25 cycles and (b) after N cycles compared to after 25 cycles, where N is the cycle (between 5 and 25) where the pipeline stopped automatically.

Figure 4.1b shows the completeness of models from the default pipeline with automatic stopping compared to running the full 25 cycles. There are fewer points far from the diagonal compared to Figure 4.1a, although there are still some datasets where a more complete model would be obtained if the pipeline continued for more cycles. The pipeline with automatic stopping runs for a minimum of 5 cycles and a maximum of 25 cycles. If the automatic method never stopped the pipeline early then all of the points would be on the diagonal, so it must also be checked that the pipeline is not performing too many unnecessary cycles.

Figure 4.2 shows the number of datasets that stopped at each cycle when automatic stopping was turned on. Results are categorised into those where the 25-cycle pipeline produced a model with $>$ 50% completeness and those where completeness was $\leq$ 50%. Overall, 23% of the datasets continued to the full 25 cycles, but 80% of these did not produce very complete models. Looking at the datasets for which $>$ 50% completeness can be achieved within 25 cycles, a large proportion finished early from cycle 6 onwards. The theoretical minimum number

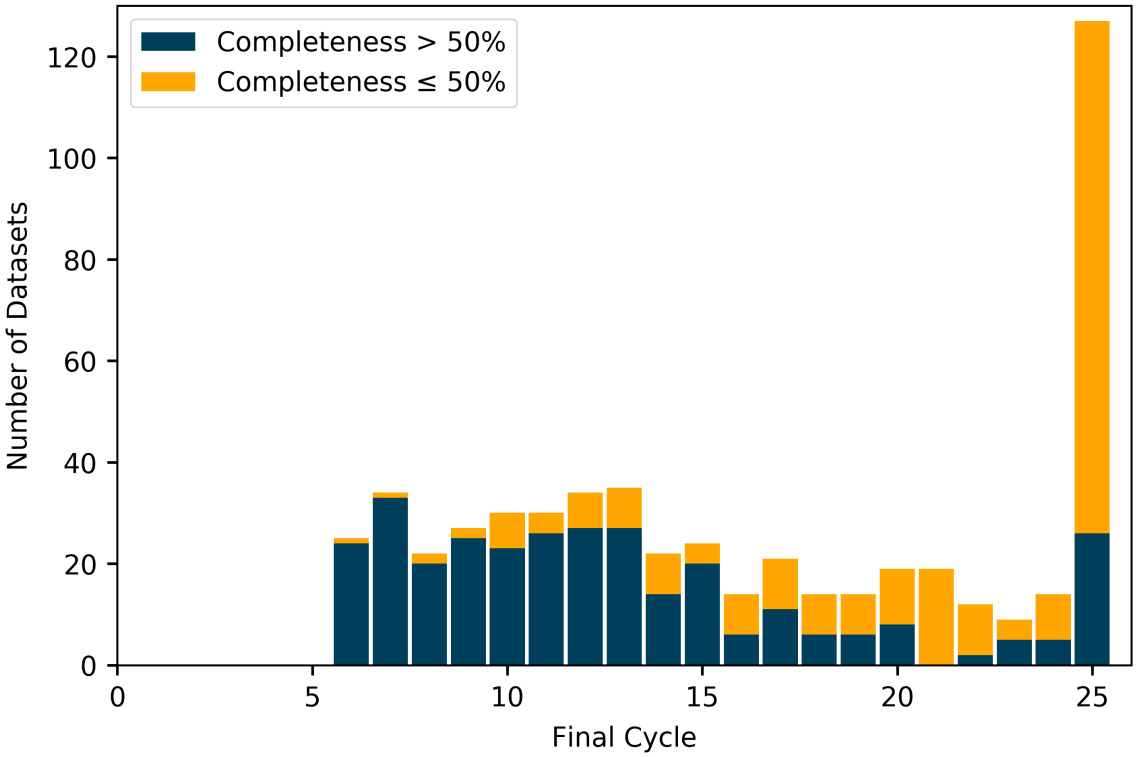

Figure 4.2: Number of datasets that finished at each cycle of the CCP4i2 pipeline with automatic stopping turned on. Datasets are categorised using the completeness of the model from the full 25-cycle pipeline.

of cycles is 5 if no possible improvement is seen after cycle 1, but there were no datasets for which this was the case.

The structure with the largest improvement in completeness is 2A9V [105], which is a structure from the Joint Centre for Structural Genomics (JCSG) with four copies of a 212 residue GMP synthase at 2.24 Å resolution. Initial phases come from a combination of molecular replacement and MAD and have an F-map correlation of 0.643 and a mean phase error of 53° after PARROT density modification.

Figure 4.3 shows R-work and R-free during the 25-cycle pipeline for 2A9V. After 5 cycles, the output model (from cycle 4) has an R-free of 54.6% and a completeness of 4.5%. The automatic stopping criteria are not met in this run and the pipeline continues to the maximum 25 cycles, after which the output model (from cycle 25) has an R-free of 34.6% and a completeness of 83.4%. The model from cycle 4 is still very fragmented without well defined secondary structure elements and the R-factors

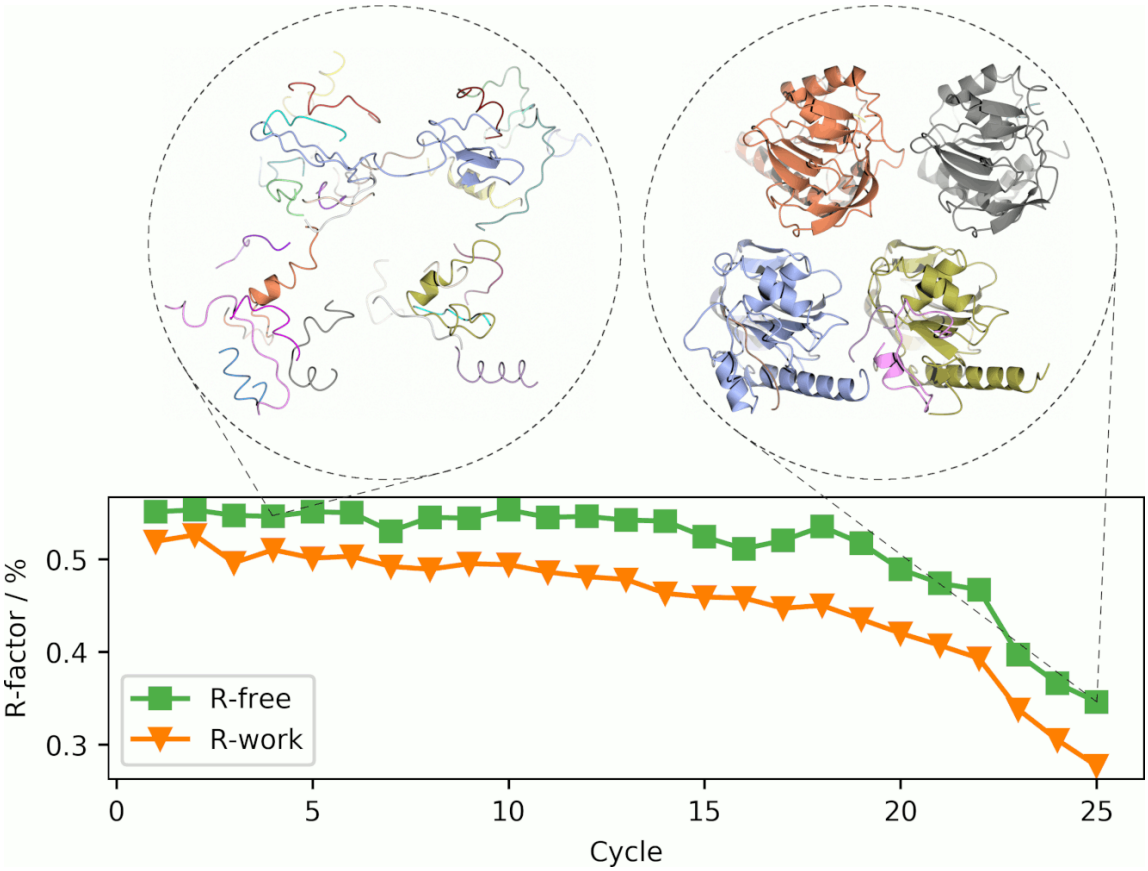

Figure 4.3: R-factors during the 25-cycle CCP4i2 pipeline for the 2A9V test case. Ribbon diagrams (coloured by chain) of the models from cycles 4 and 25 are shown above.

do not change dramatically until the end of the pipeline. In this case the model would likely be improved further by continuing for more than 25 cycles.

Because the pipeline outputs the model with the lowest R-free, there were not many cases with a decrease in completeness after running more cycles. One example is 5NBP [106], which is a 1.80 Å structure with two copies of a 272 residue glycosyl hydrolase. Initial phases were produced using chain A from 3ILN [107], which was prepared using SCULPTOR [79] and superposed over each chain using GESAMT [80]. Refinement of the resulting model in REFMAC [11] gave an R-work of 48%, an R-free of 50%, an F-map correlation of 0.535 and a mean phase error of 59.9°.

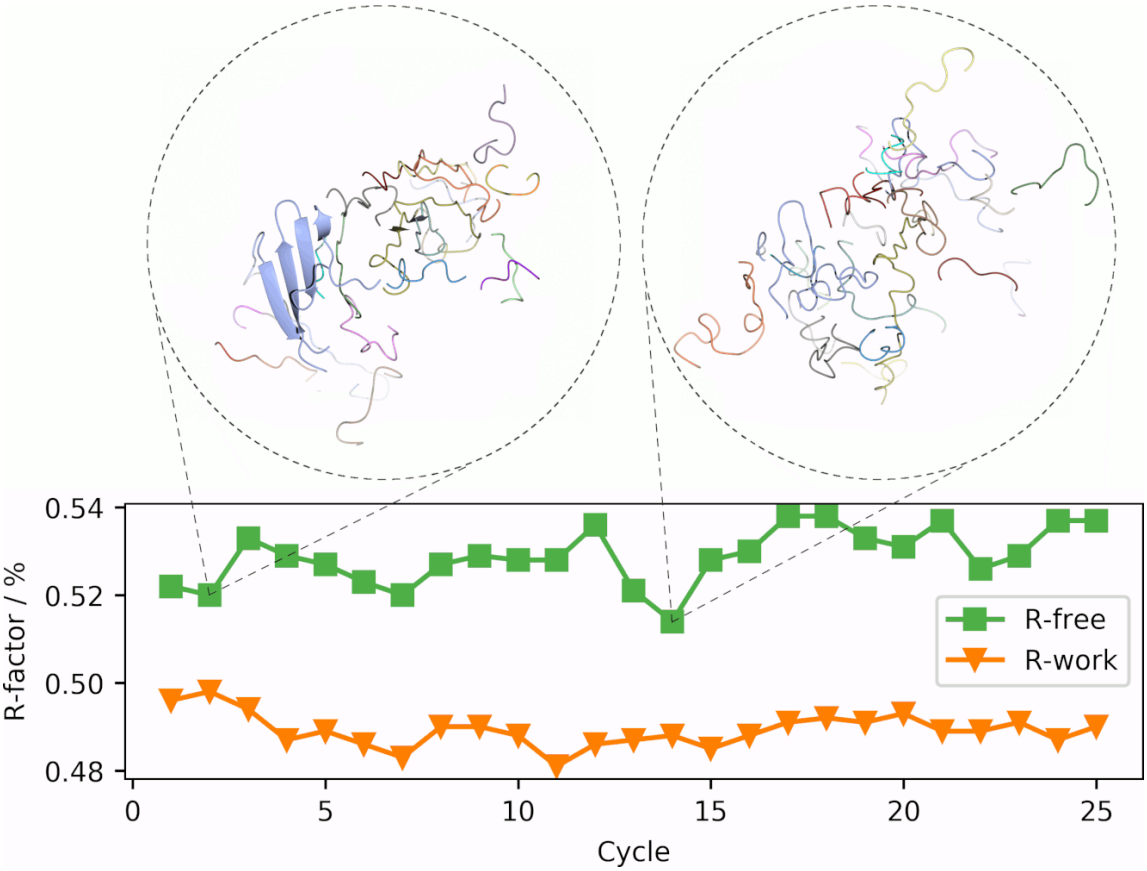

Figure 4.4: R-factors during the 25-cycle CCP4i2 pipeline for the 5NBP test case. Ribbon diagrams (coloured by chain) of the models from cycles 2 and 14 are shown above.

Figure 4.4 shows the R-factors during the 25-cycle pipeline for 5NBP. In this test, BUCCANEER starts by building into an empty map without using the model for initial Cα locations. The output model after 5 cycles (from cycle 2) has an R-free of 52.0% and a completeness of 8.9%. The automatic stopping criteria halts the pipeline after cycle 18. The output model (from cycle 14) has a slightly lower R-free of 51.4% but the completeness has dropped to 0.0%. Ribbon diagrams show that the model becomes even more fragmented and loses the β-sheet that was present in cycle 2. Better results might be achieved by using the starting model in BUCCANEER. This was the structure with the biggest decrease in completeness (8.9%), but the model was already very incomplete. There were no examples where a complete model had a similar drop in completeness.

4.1.2.3 Discussion

Previously, the BUCCANEER pipelines in both CCP4i and CCP4i2 performed 5 cycles by default. This was chosen as a good compromise between speed and model quality for early test sets, but after the test set was expanded a number of structures were found for which more cycles are needed. However, changing the default number of cycles to 25 would make the pipeline run much longer than necessary in most cases, so a new automatic stopping method was introduced that helps tackle this problem. The CCP4i2 pipeline now runs for a maximum of 25 cycles, but will stop early if the model does not show any signs of improvement. The ability to run for more cycles if needed, combined with choosing the model with the lowest R-free, enhances the quality the models produced by the pipeline.

BUCCANEER approaches model building by growing many fragments, some of which may be in poor density, before merging, linking and pruning them to produce a single consistent model. This lets the pipeline explore a large amount of space and means it can run for many cycles with a changing fragmented model before the phases improve enough and the correct structure is found. These cases can be identified by the R-factors remaining high but unstable before decreasing rapidly. It was for this reason that the pipeline was allowed to continue for four cycles without any noticeable improvement in the model. The approach of building and pruning means that a correct chain fragment may occasionally be deleted by mistake, for example it is difficult to choose between a shorter chain that is fully sequenced and a longer chain that contains unsequenced residues. BUCCANEER version 1.6.10 contains improvements to the pruning step that reduce the chance of pruning correct residues. Although it can still occur, the problem is mitigated by choosing the model from the cycle with the lowest R-free. The new automatic stopping method makes it more likely that the final cycle will not have the best model in the pipeline so choosing the best model is even more important.

The stopping method could be improved. The majority of datasets that produced a mostly complete model did so within the first 10 cycles but the user has to wait another 4 cycles before the pipeline stops. Being able to recognise that the model cannot be improved further would save a large proportion of the time in these cases. In other cases the maps given to the pipeline are so poor that BUCCANEER is

unlikely to succeed. In which case the pipeline often runs for the full 25 cycles due to different fragmented models being classed as possible improvements. If very poor input maps can be identified and model building is not succeeding, then the pipeline could be stopped early with a message to the user or density modification could be used to try improve the phases.

4.2 ModelCraft

The two current-generation CCP4 graphical user interfaces, CCP4i2 and CCP4Cloud, both have a BUCCANEER pipeline. Both pipelines are presented as 'BUCCANEER', so it would be reasonable for a user to assume they are the same. However, CCP4Cloud calls the older CCP4i pipeline, which was the only one that could run on the command line at the time of development. As covered in the previous section, many improvements have been made to the CCP4i2 pipeline since then. It is now also possible to run the CCP4i2 pipeline on the command line using i2run [108]. CCP4Cloud could be changed to use the newer CCP4i2 pipeline, but instead it was decided to make a new pipeline called ModelCraft that is independent of any GUI framework. The aim is to have a single pipeline with command line, CCP4i2 and CCP4Cloud interfaces so that pipeline developments only need to be made once. It should also be possible to use the ModelCraft framework for the BUCCANEER pipeline in CCP-EM [109] to reduce the duplication of effort between X-ray crystallography and cryo-EM.

As in the CCP4i2 pipeline, ModelCraft runs for a maximum of 25 cycles with the option to stop early if the model does not improve for 4 cycles. However, the stopping criteria have been simplified so that a cycle is only marked as an improvement if R-free decreases by 0.1% from the previous best value. More steps have been added to each cycle of the pipeline. A single cycle now consists of the following steps:

1. Prune chains, residues and side chains using COOT followed by 5 cycles of REFMAC. This step is not performed on the first cycle or if the resolution is 2.3 Å or worse.

2. Density modification using PARROT.

3. Flood the structure with dummy atoms using COOT followed by 10 cycles of REFMAC, but only accept the result if R-free improves. This step is not performed on the first cycle of experimental phasing cases when there is no model.

4. BUCCANEER followed by 10 cycles of REFMAC.

5. Prune chains using COOT followed by 5 cycles of REFMAC.

6. Find waters using COOT followed by 10 cycles of REFMAC, but only accept the result if R-free improves.

Extra steps have also been added to the start and end of the pipeline. If starting phases are being determined from a molecular replacement model, SHEETBEND is used to refine the model before REFMAC. At the end of the pipeline, COOT is used to try and improve side chains in the final model if R-work is below 30% and the resolution is better than 2.5 Å. Individual steps will be discussed in more detail in sections 4.2.1 to 4.2.6. First, the overall performance of ModelCraft will be compared with the CCP4i BUCCANEER pipeline, as this shows the combined effect of all the pipeline developments made during the PhD project. Testing was performed on 1348 molecular replacement cases and 193 experimental phasing cases using CCP4 7.1.000. For comparison, model building results from ARP/wARP 8.0 and PHENIX 1.14 AutoBuild are also presented. The runs of ARP/wARP and PHENIX AutoBuild were performed by E. Alharbi at the University of York using default command line parameters. The experimental phasing results were published by Alharbi *et al.* [70] but the molecular replacement results have not yet been released (E Alharbi, 2019, unpublished).

Figure 4.5 shows completeness against resolution for ModelCraft, ARP/wARP, PHENIX AutoBuild and CCP4i. Molecular replacement cases are shown on the left and experimental phasing cases on the right. Data are only included for the 1324 molecular replacement cases and 191 experimental phasing cases for which all pipelines produced a model and terminated normally. Completeness is the percentage of residues in the refined deposited structure that have a matching residue in the pipeline model. A residue is deemed to be matching if N, Cα and C are all within 1 Å.

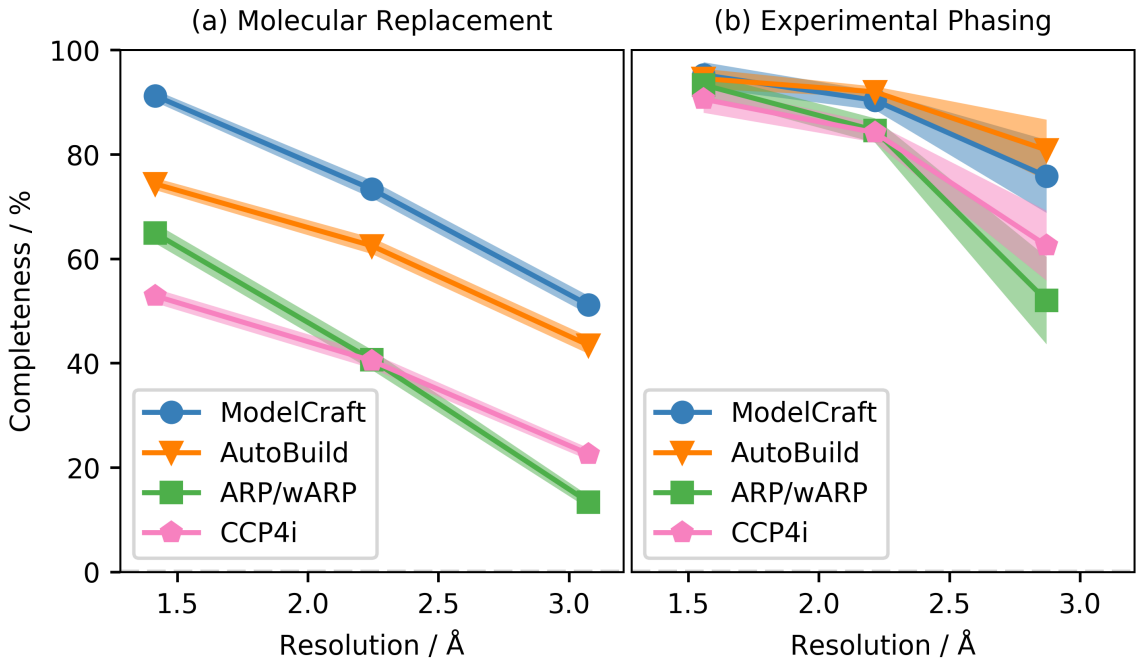

Figure 4.5: Model completeness as a function of resolution for ModelCraft, PHENIX AutoBuild, ARP/wARP and CCP4i over 1324 MR and 191 EP cases. Points show the mean completeness in 3 bins and the shaded area shows one standard error above and below the mean.

For molecular replacement, mean completeness is similar for CCP4i and ARP/wARP, although ARP/wARP is better at high resolution and CCP4i is better at low resolution. PHENIX AutoBuild gives more complete models than both CCP4i and ARP/wARP and has a similar resolution dependency to CCP4i. However, ModelCraft has the highest mean completeness at all resolutions. The mean ModelCraft completeness in the low resolution bin is similar to mean CCP4i completeness in the high resolution bin. Most of the experimental phasing cases have good phases so the mean completeness is high for all pipelines, especially in the highest resolution bin where they all perform similarly well. At lower resolutions, PHENIX AutoBuild and ModelCraft give better models than CCP4i and ARP/wARP.

Figure 4.6 shows completeness against F-map correlation for the pipelines. F-map correlation is the correlation coefficient between the structure factor amplitudes of the starting map and a map of the refined deposited structure, weighted by the cosine of the phase difference. Again, ModelCraft has the highest mean completeness for

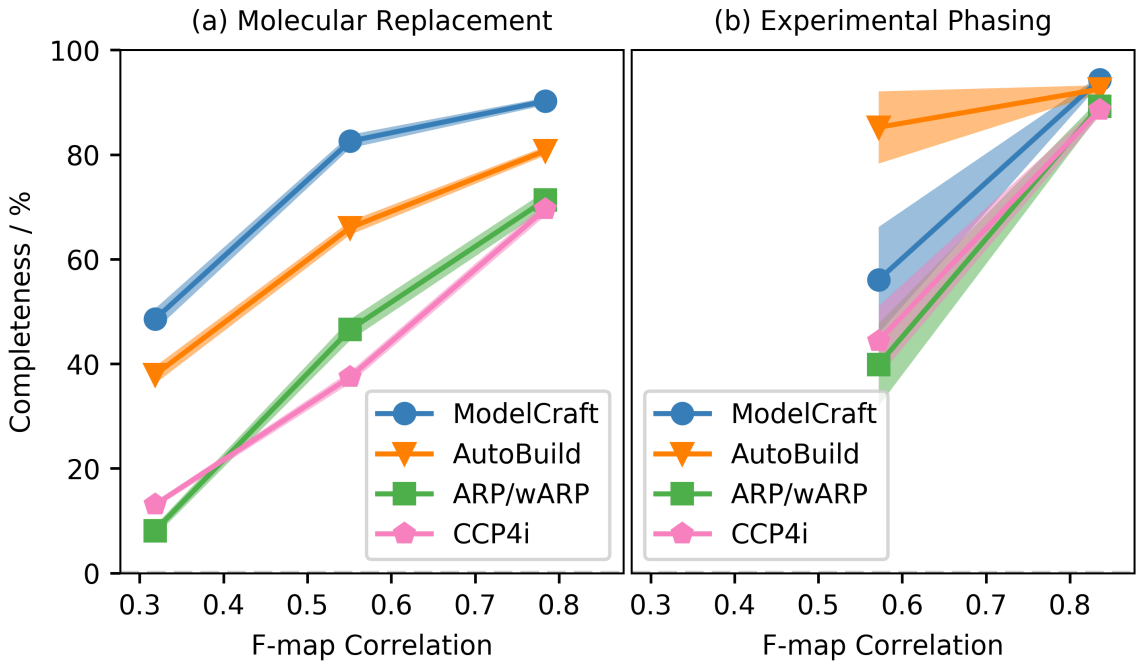

Figure 4.6: Model completeness as a function of F-map correlation for ModelCraft, PHENIX AutoBuild, ARP/wARP and CCP4i over 1324 MR and 191 EP cases. Points show the mean completeness in each bin and the shaded area shows one standard error above and below the mean.

all bins, followed by PHENIX AutoBuild. ARP/wARP and CCP4i struggle with very poor molecular replacement models. The pipeline developments have made ModelCraft less sensitive to the quality of the initial model. Comparing the middle F-map correlation bin to the high F-map correlation bin, ModelCraft completeness only drops slightly but CCP4i completeness almost halves.

For the experimental phasing test set, the data are split into two bins because there are not many examples with low F-map correlation, unlike the molecular replacement test set where F-map correlation is evenly distributed between 0.2 and 0.9. All pipelines perform well at high F-map correlation, with ModelCraft and PHENIX AutoBuild giving more complete models than ARP/wARP and CCP4i. There is a sharp drop in completeness as the starting phases become worse for ModelCraft, ARP/wARP and CCP4i. However, PHENIX AutoBuild still has high completeness in the lower F-map correlation bin. It would be useful to increase the number of experimental phasing cases with low F-map correlation for a more accurate comparison and to help find improvements in this area.

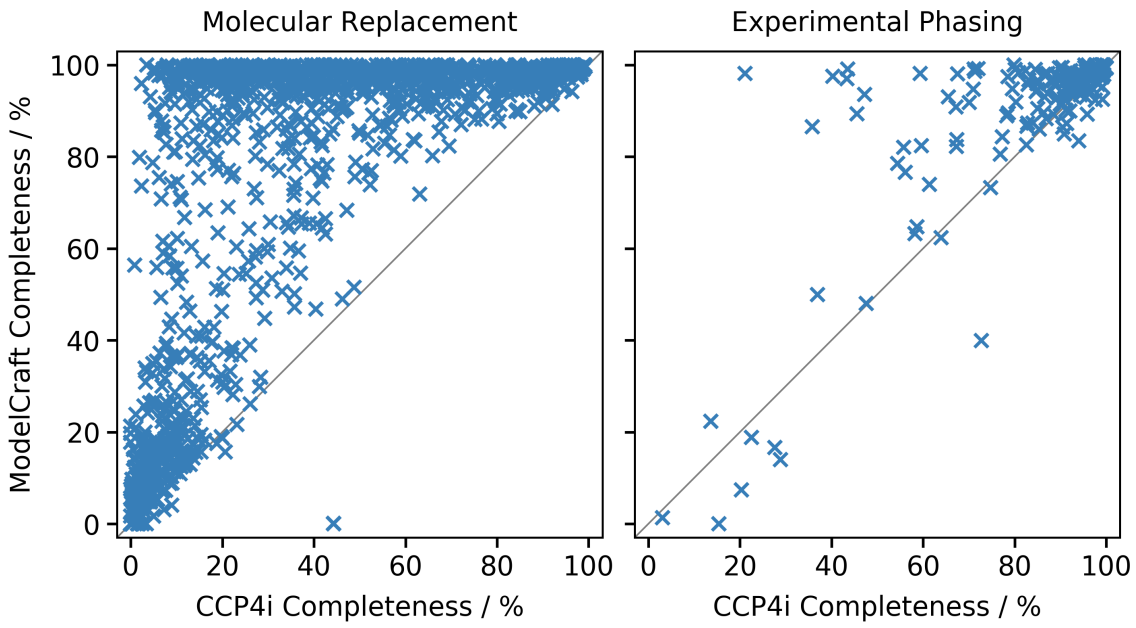

Figure 4.7: Scatter plots of model completeness for the CCP4i and ModelCraft pipelines for 1348 MR and 193 EP cases.

Figure 4.7 shows the completeness of the CCP4i and ModelCraft models for each test case. For molecular replacement, a lot of cases are built to near 100% completeness by ModelCraft, even when CCP4i produces a very incomplete model. There are still many difficult cases that cannot be built well by either pipeline, but ModelCraft nearly always gives an improvement. The experimental phasing test set does not contain many difficult cases, but even so the underlying distribution looks slightly different to molecular replacement, with CCP4i performing better in most of the incomplete models and a few of the complete models.

Figure 4.8 shows the extra time it takes ModelCraft to run compared to CCP4i against the extra completeness achieved. The 1348 molecular replacement and 193 experimental phasing cases are shown together. The running times are only estimates as they are measured differently for the two pipelines. CCP4i only includes BUCCANEER and REFMAC and the pipeline time is the total of the CPU time read from the log files. ModelCraft includes other programs that do not report CPU time, so the real time was measured instead.

For CCP4i, the average time over all 1541 cases is 20 minutes. It is very fast as it

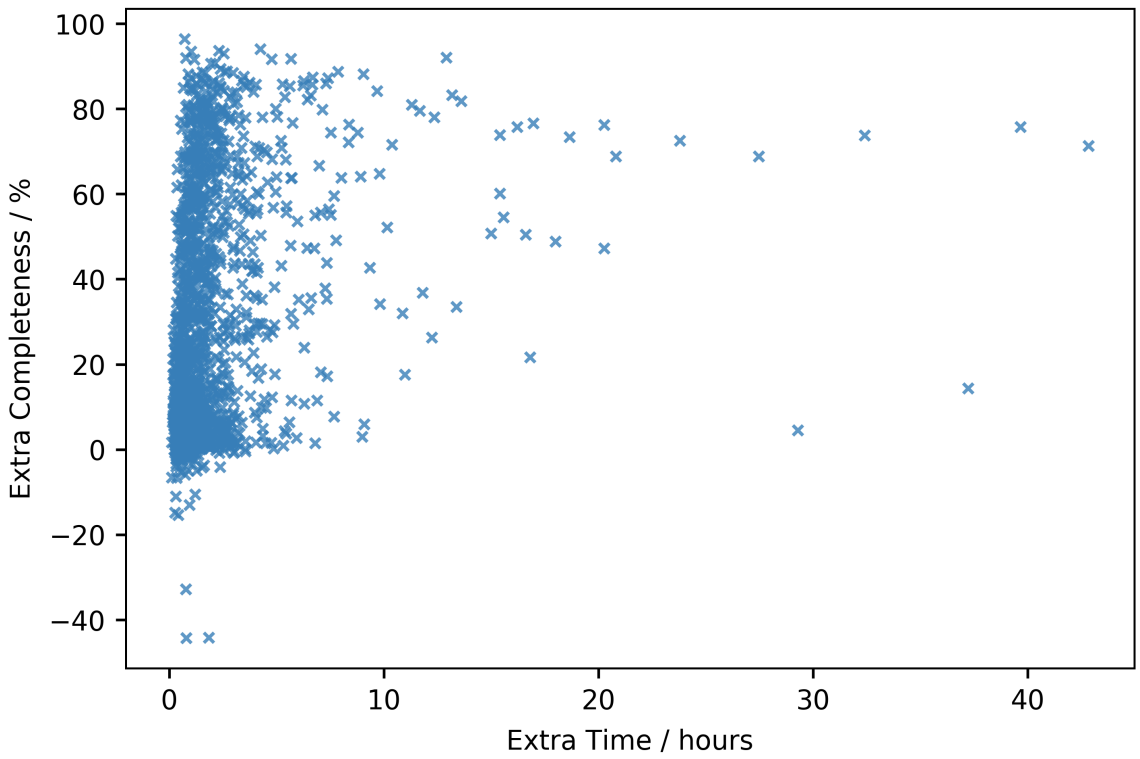

Figure 4.8: The extra completeness gained by using the ModelCraft pipeline instead of the CCP4i pipeline against the extra time it takes for the pipeline to finish for 1541 test cases.

only runs for five cycles and each cycle is composed of one call to BUCCANEER and REFMAC. The average time for ModelCraft is 2 hours 21 minutes. ModelCraft runs for up to 25 cycles and each cycle includes additional steps of COOT, PARROT and REFMAC. ModelCraft finishes most cases in less than 1.5 hours but a few cases take a very long time.

Figure 4.9 shows a subset of Figure 4.8, excluding 102 cases where ModelCraft takes more than 5 hours longer than CCP4i. There are many cases where both pipelines produce a model with similar completeness, but ModelCraft is always slower than CCP4i so CCP4i may be preferred in these cases. It is rare that CCP4i produces a more complete model than ModelCraft. In quite a few cases, ModelCraft produces a substantially better model than CCP4i. The user will have to wait longer for the more complete model but perhaps still less time than it would take to improve the model manually. Even if ModelCraft takes three times as long as CCP4i and only builds 5–10% more of the structure it will often be preferable.

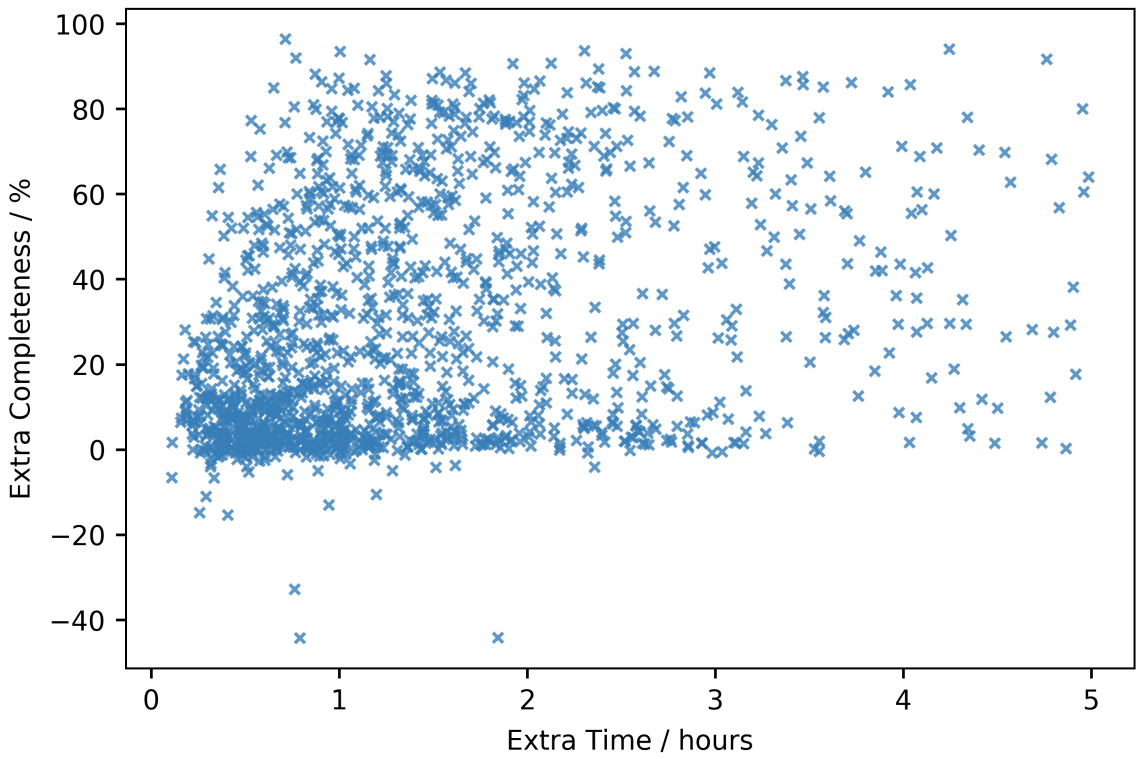

Figure 4.9: The extra completeness gained by using the ModelCraft pipeline instead of the CCP4i pipeline against the extra time it takes for the pipeline to finish for 1439 test cases where the extra time is less than five hours.

Figure 4.10 shows the total running time for each program within ModelCraft against the number of residues in the deposited model. The majority of test cases are in the first bin so this has small standard errors. The running time of the pipeline increases as the size of the structure increases. The program that takes the most time is REFMAC as it is called many times on each cycle. For small structures, REFMAC takes up a large proportion of the overall pipeline time. As the size of the structure increases, the time taken by BUCCANEER and PARROT increases greatly. They take up a big proportion of the total time for large structures even though they are only called once per cycle. The running time of PARROT is more variable than BUCCANEER as can be seen by the difference in the standard error. COOT is used for pruning, adding waters and dummy atoms, and rebuilding side chains at the end of the pipeline but it is relatively quick for large structures. SHEETBEND is fast and is only used once at the start of the pipeline when determining phases from a molecular replacement model.

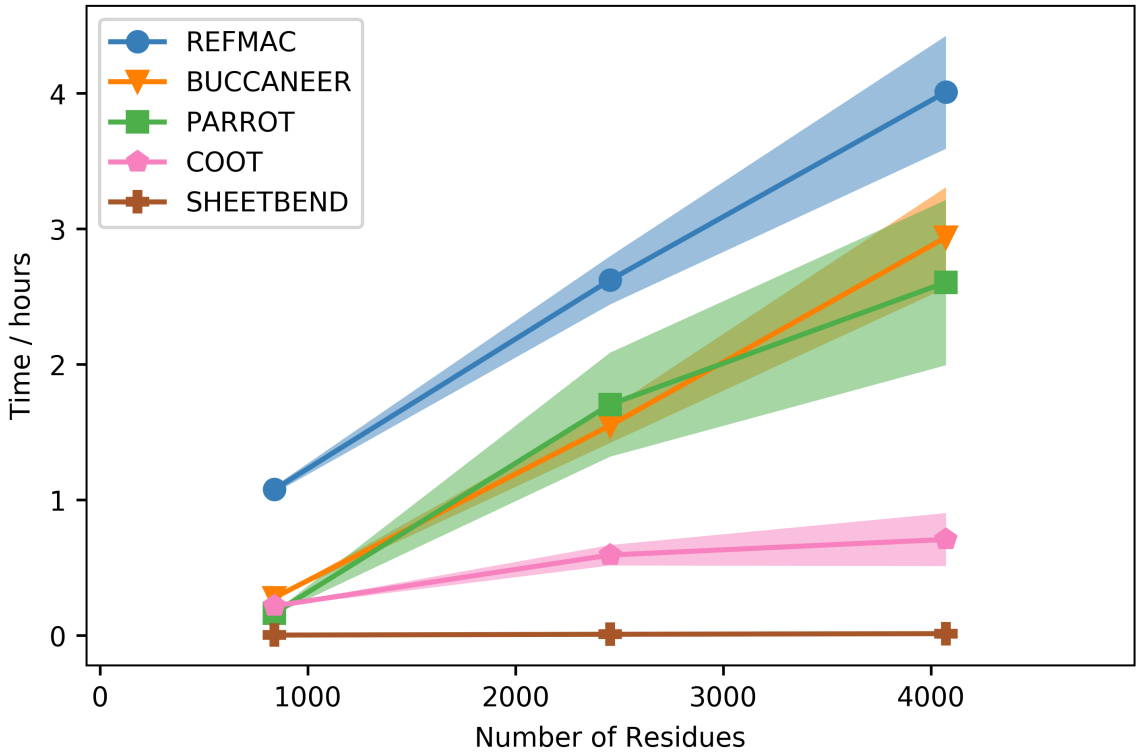

Figure 4.10: Total running time of each program in the ModelCraft pipeline as a function of the number of residues in the deposited structure. Only cases with less than 5000 residues are included in the data for this plot. Points show the mean time in 3 bins and the shaded area shows one standard error above and below the mean.

Pipeline changes were assessed using the change in completeness and R-factors of the models produced. Performance was examined as a function of resolution as it was expected some steps would only be beneficial within certain resolution ranges. For example, it was found that pruning individual residues using the machine-learned correctness score was beneficial at high resolution but detrimental at low resolution. A pipeline change was accepted if it gave an improvement then the next change was assessed. This method of sequential development has the drawback that it does not examine the interdependence between steps. A change to the pipeline that gives an improvement may also make other steps obsolete. In addition, it is possible that two steps only give an improvement when used together, or even only when used separately. The process of developing a pipeline is therefore quite organic as not every possible combination of steps can be explored.

The following sections will look at individual developments within ModelCraft in

more detail. For each step, the change in completeness and R-factors when the step is removed from the pipeline will be presented. As stated previously, the removal of a step may have less of an effect than its original addition owing to the subsequent inclusion of other steps. An alternative would be to present the effect of adding individual steps to a basic pipeline that does not have any of the new developments. This will likely show larger changes, but it is less representative of the current state of the pipeline.

4.2.1 Shift-field Refinement

Shift-field refinement is a new refinement method where parameter shifts are calculated using large regions of the map instead of individual atomic positions. The shifts therefore vary smoothly across the map and do not require external restraints to preserve small scale atomic features such as bond lengths. The technique was originally demonstrated for the refinement of B-factors [13], but it has since been validated as a useful technique for refining the coordinates of molecular replacement models [14]. Shift-field refinement for both coordinates and B-factors is available in the program SHEETBEND. The calculation is quick as it only requires a small number of cycles and is performed at low resolution. Similar to jelly-body refinement [43], it has a larger radius of convergence than conventional refinement and provides the most benefit when large shifts in the model are needed.

When starting from a potentially-unrefined molecular replacement model, the first step of the pipeline is to refine the model to improve it and produce a set of starting phases. Shift-field refinement is performed first using 12 cycles of SHEETBEND with the resolution increasing from 6 Å to 3 Å in the last 6 cycles. This is followed by 10 cycles of conventional refinement using REFMAC. The lack of restraints in shift-field refinement means the model can become distorted over many cycles so conventional refinement is needed to correct this [14].

Figure 4.11 shows that removing the shift-field refinement step (so that the input molecular replacement model is only refined using REFMAC) makes both completeness and R-factors worse. There does not seem to be much of a dependence on resolution as performance is worse in all three resolution bins.

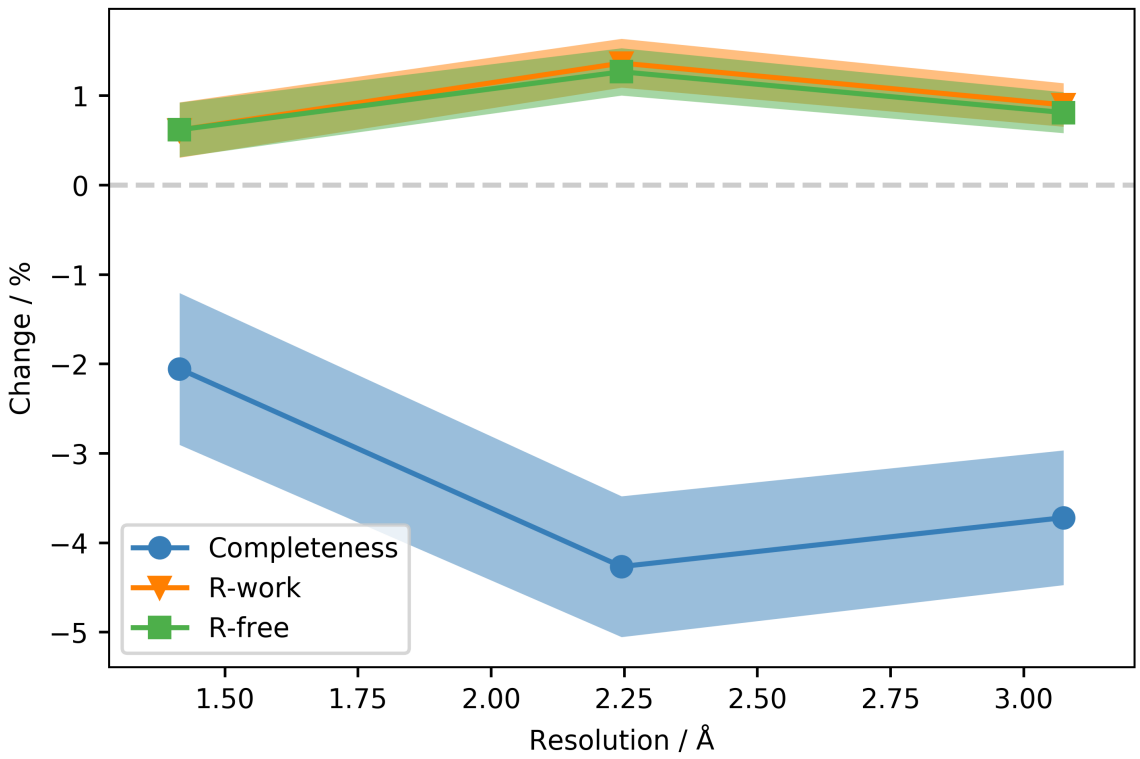

Figure 4.11: Change in completeness, R-work and R-free on the removal of the SHEETBEND shift-field refinement step for 1348 MR cases. Points show the mean change in 3 bins and the shaded area shows one standard error above and below the mean.

R-work and R-free change by a similar amount, which suggests that the shift-field refinement step is not causing any overfitting. The mean change is small because SHEETBEND only has a big effect when large shifts to the molecular replacement model are required. It will make little difference to most cases in the test set. The same is true for all of the new steps, i.e. they occasionally give big improvements but, in most cases, complete models are still complete and incorrect models are still incorrect.

4.2.2 Pruning

The new pruning steps were described in detail in Chapter 3. The pruning depends on a machine-learned correctness score that is calculated using a neural network. Two pipeline steps were developed: one to prune whole chains at the end of each cycle

before adding waters, and one that also prunes individual residues and side chains at the start of each cycle before PARROT density modification. The final chain pruning step was found to be beneficial at all resolutions, but pruning individual residues and side chains was only beneficial at high resolution so the step is turned off when the resolution 2.3 Å or worse.

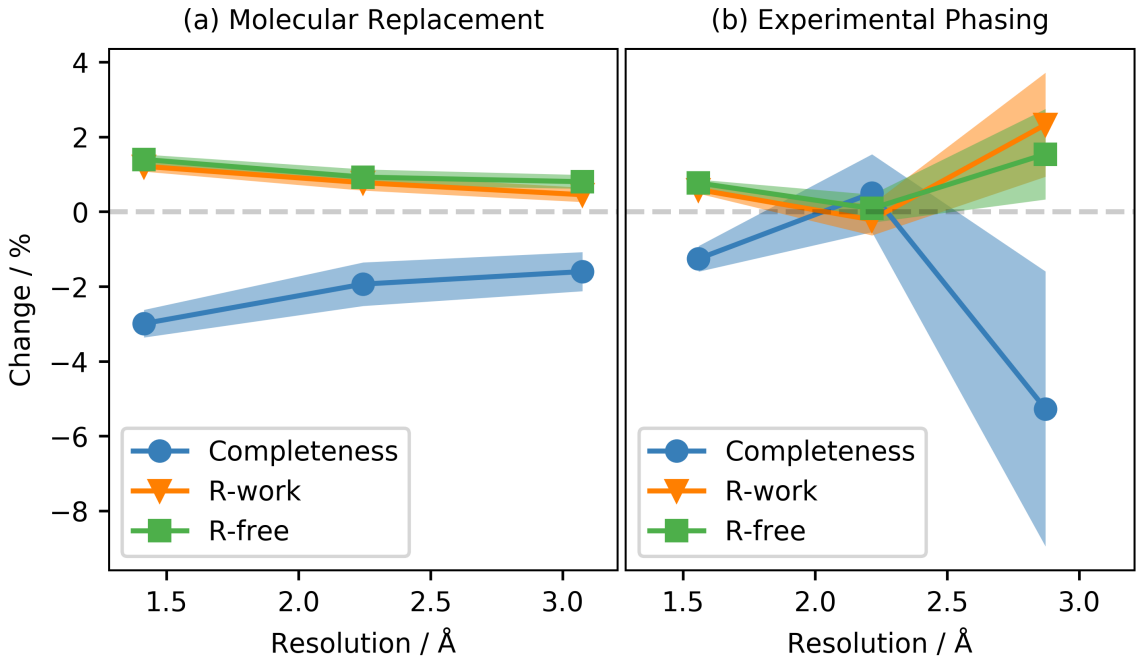

Figure 4.12: Change in completeness, R-work and R-free on the removal of the pruning steps for 1347 MR and 193 EP cases. Points show the mean change in 3 bins and the shaded area shows one standard error above and below the mean.

Figure 4.12 shows the change in completeness and R-factors when both pruning steps are removed from the pipeline. For molecular replacement cases, pruning improves R-factors and completeness at all resolutions, although slightly more at high resolution. Pruning gives less improvement in the experimental phases test set, but there is still a statistically significant improvement at high resolution.

4.2.3 PARROT Density Modification

PARROT [71] is classical density modification program that updates the phase probability distribution using a single maximum-likelihood calculation that includes prior phase information. In contrast, previous programs such as DM [110],

SOLOMON [111] and CNS [112] use two steps by first estimating the error in the modified phases and then combining the resulting phase distribution with the experimental phases. Although PARROT is not included in the previous BUCCANEER pipelines, it is expected that density modification will already have been carried out when starting from experimental phases. However, PARROT is included in the CRANK2 experimental phasing pipeline [92], which runs density modification on each cycle and feeds the results into a combined experimental phasing, phase combination and model refinement step.

Another new development in PARROT is the use of multiple pairwise weighted NCS averaging masks, which is better for handling cases where there are three or more NCS copies that are not all equally similar. As the mask calculation is fast it is performed on each cycle, using the correlation between NCS regions in the current map, instead of once using the starting map. For NCS averaging to be effective the NCS operators need to be correctly determined from the model, which will hopefully be more likely when using a pruned model compared to a model with extra chains

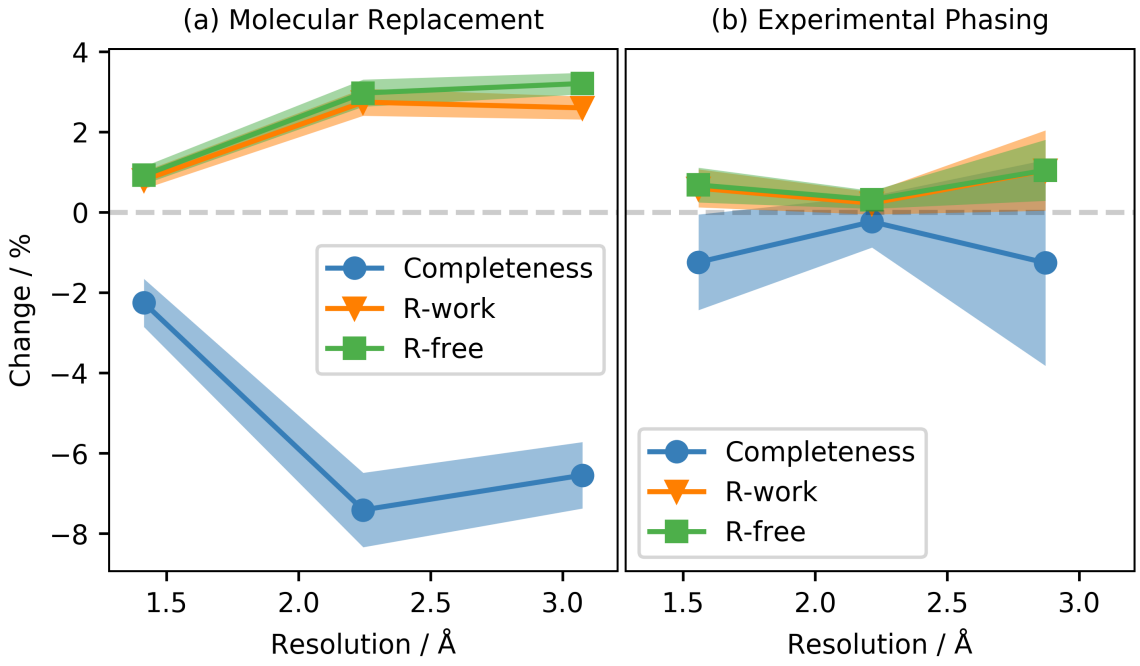

Figure 4.13: Change in completeness, R-work and R-free on the removal of the PARROT density modification step for 1347 MR and 193 EP cases. Points show the mean change in 3 bins and the shaded area shows one standard error above and below the mean.

built into the solvent.

Five cycles of PARROT were added to the start of each cycle of the ModelCraft pipeline, after the pruning step and before adding dummy atoms. PARROT is provided with the protein sequence for solvent content estimation and the current model for the determination of NCS operators. However, the model is not used as a solvent mask as this leads to poor performance with partial and fragmented models. Instead, the solvent mask is recalculated during each PARROT cycle using the current map.

Figure 4.13 shows how completeness and R-factors change when the PARROT step is removed. For the molecular replacement test set, both completeness and R-factors are worse without PARROT, especially at mid to low resolution. R-free increases slightly more than R-work at low resolution, which suggests that PARROT is reducing overfitting, perhaps by reducing the differences between NCS copies. The change in performance is much smaller for the experimental phasing test set, perhaps because the starting phases have already been modified using PARROT. Further testing should be carried out on the molecular replacement test set to see whether similar improvements can be achieved by running PARROT only at the start of the pipeline, or only when there is a change in the determination of NCS operators.

4.2.4 Dummy Atom Addition

Dummy atoms are pseudo atoms that are added to an atomic model to represent unknown scattering matter. They do not have a chemical identity but are assigned an element, often oxygen, so their contribution to the structure factor equation can be calculated. As dummy atoms do not represent real atom sites with known bonding, they are refined without geometry restraints. However, at high resolution, it is often possible to determine individual atomic positions even with unrestrained refinement. At lower resolution, the unrestrained atoms will become over-fitted, but it could still be beneficial to include them even without accurate parameters.

ARP/wARP uses dummy atoms for phase improvement [45], main chain tracing [47] and side chain identification [48]. In ModelCraft, they are only used a tool

for phase improvement. Dummy atoms are added to an existing model in peaks that are unaccounted for then the hybrid model is refined using REFMAC and the resulting structure is only accepted if R-free is lower with the dummy atoms. It is important to use R-free instead of R-work for this check because R-work will almost certainly decrease due to overfitting while R-free provides an unbiased measure of improvement. The next step in the pipeline is BUCCANEER, which will discard any dummy atoms in the input model.

Dummy atom positions are chosen using the *flood* option of the COOT *findwaters* program. This places dummy atoms in peaks above 2σ that are within a 1.9–10 Å distance of the model, with a minimum contact distance between dummy atoms of 1.4 Å. Dummy atom addition and refinement is performed after the PARROT step, using the modified map from PARROT and the pruned model. It is hoped that residues in incorrect conformations will be pruned, PARROT will reduce the phase error, then the dummy atoms will emphasise the density for the correct conformation.

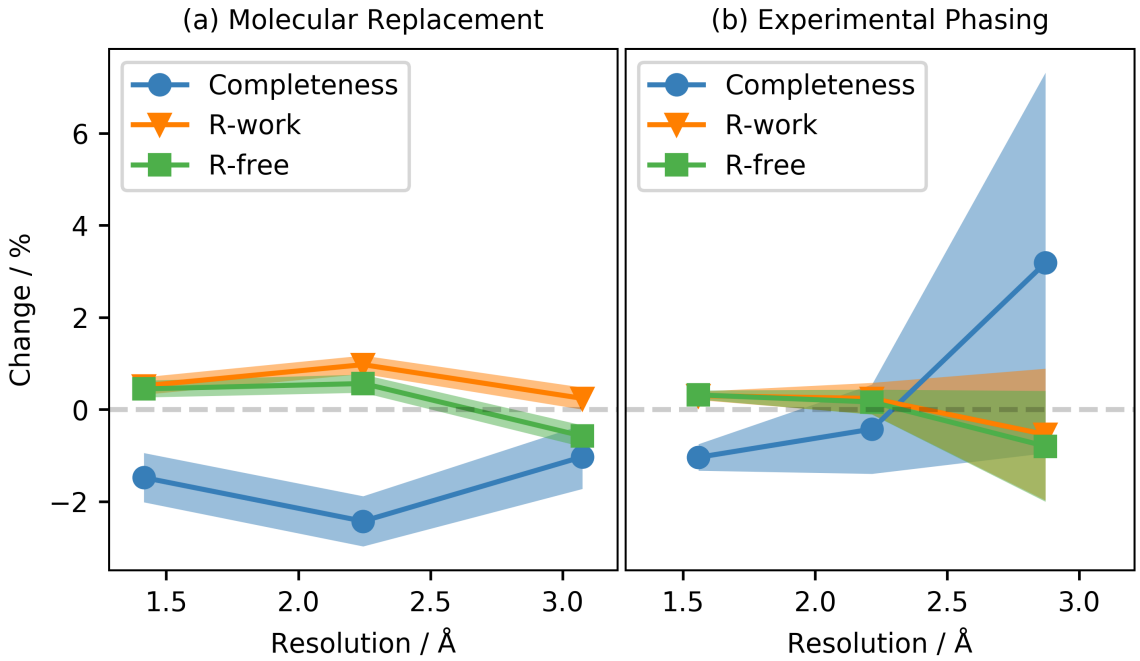

Figure 4.14: Change in completeness, R-work and R-free on the removal of the dummy atom addition step for 1347 MR and 193 EP cases. Points show the mean change in 3 bins and the shaded area shows one standard error above and below the mean.

Figure 4.14 shows the change in completeness and R-factors on the removal of the

dummy atom addition step. For the molecular replacement test set, adding dummy atoms improves completeness at all resolutions. However, at low resolution, R-free now seems to improve without the step and the difference between R-work and R-free shows that using dummy atoms for density modification is still causing overfitting. This is despite only accepting the dummy atom model if R-free improves. Perhaps the step should be turned off for low resolution cases or the dummy atom model should only be accepted if R-free improves by a larger amount. For the experimental phasing cases, there is an improvement at high resolution, but at lower resolution the standard errors are too high to draw firm conclusions.

4.2.5 Water Addition

A water addition step was added to the ModelCraft pipeline that is very similar to the dummy atom addition step. It also uses the COOT *findwaters* program, but without the *flood* option. Peaks above 2σ are only accepted if the volume of the peak is less than 15 Å^3 (corresponding to a sphere with a radius of 1.53 Å). The water building runs in 3 cycles where each potential water position is checked to be chemically sensible, i.e. more than 2.4 Å away from any other atom and within 3.2 Å of either a nitrogen or oxygen atom of the model or a water added in a previous cycle.

Water addition is the very last step in each cycle. It is performed after pruning whole chains from the model so that chain fragments built into the solvent do not block ordered water positions. As in the dummy atom addition step, the model with waters is only accepted if it has a better R-free than the model without waters.

Figure 4.15 shows how completeness and R-factors change when the water addition step is removed. Similar results are observed for the molecular replacement and experimental phasing test sets. At high resolution, adding waters leads to much better R-factors. Unlike dummy atoms, the added waters are kept in the output model so they have a direct impact on the R-factors. The difference between R-work and R-free increases at lower resolutions, meaning there is more overfitting, but there is still an improvement to both metrics. There is not much of a difference in completeness, except possibly at mid to low resolution in the molecular test set where adding waters leads to an improvement, presumably due improved phases being used

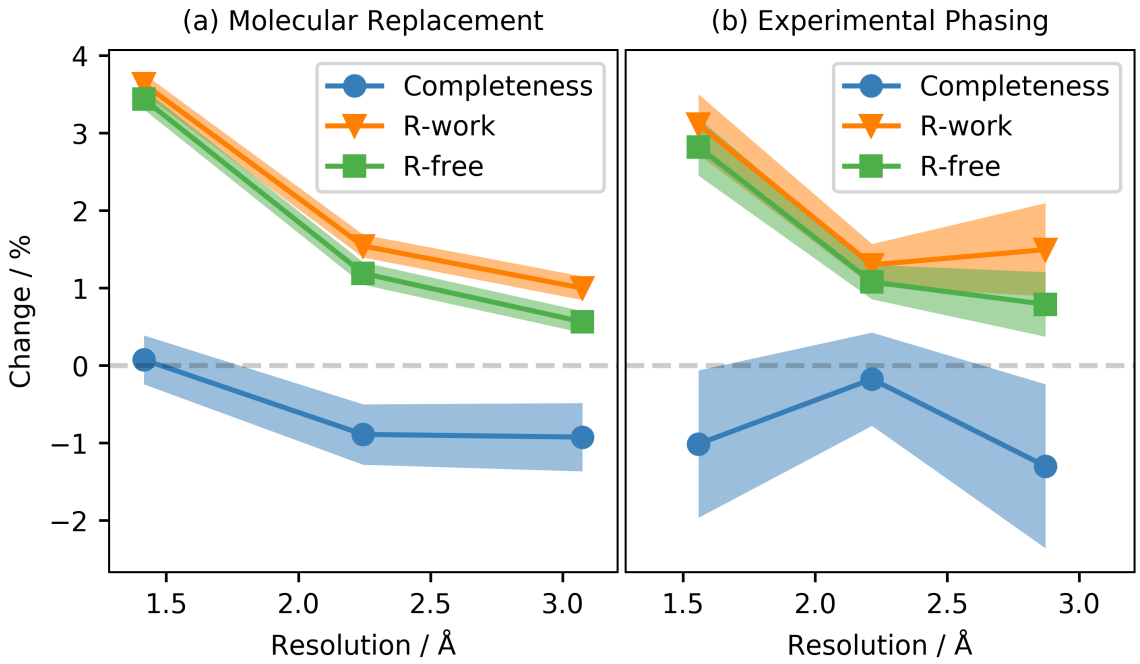

Figure 4.15: Change in completeness, R-work and R-free on the removal of the water addition step for 1346 MR and 193 EP cases. Points show the mean change in 3 bins and the shaded area shows one standard error above and below the mean.

in subsequent cycles.

The inclusion of a water addition step in the BUCCANEER pipeline is not new. The CCP4i2 pipeline has the option to perform a COOT real-space operation, for example water addition, once R-work drops below a threshold value (40% by default). However, this has not been used by default in the past. After it was found to provide an overall benefit to ModelCraft, more testing was done to show this was also the case in CCP4i2 and the step was made a default in CCP4 7.1.004.

4.2.6 Side Chain Rebuilding

The side chain rebuilding step is performed once at the end of the pipeline to fix side chains that are either missing or predicted to be in incorrect conformations. The reason side chains may be missing is due to the clash resolution function in BUCCANEER, which penalises the score of rotamer pairs that clash and will truncate both side chains to Cβ if the best scoring pair still clashes. Side chains

with a machine-learned correctness score (discussed in chapter 3) less than 0.25 times the median for the model are marked for rebuilding. For both missing and low scoring side chains, rebuilding is only carried out if the main chain correctness score is higher than 0.25 times the median. This is to avoid refining the side chain into the wrong position when the main chain has large errors.

COOT is used for the rebuilding procedure, which starts by truncating the side chain to Cβ then performing real-space refinement on the residue and the residues either side. This hopefully fixes any errors in the main chain that make it harder to identify the correct rotamer. Truncating the side chain before refinement ensures that the backbone is not being held in the wrong place by the side chain. The side chain is then re-added and the *auto-fit-best-rotamer* function [98] is used to find the best rotamer, which works by rigid-body refining each rotamer and scoring it against the density. Only rotamers with probabilities above 0.1% are considered, provided that they do not clash with other atoms in the model. Once the rotamer is built, a final refinement is carried out on the residue and its neighbours.

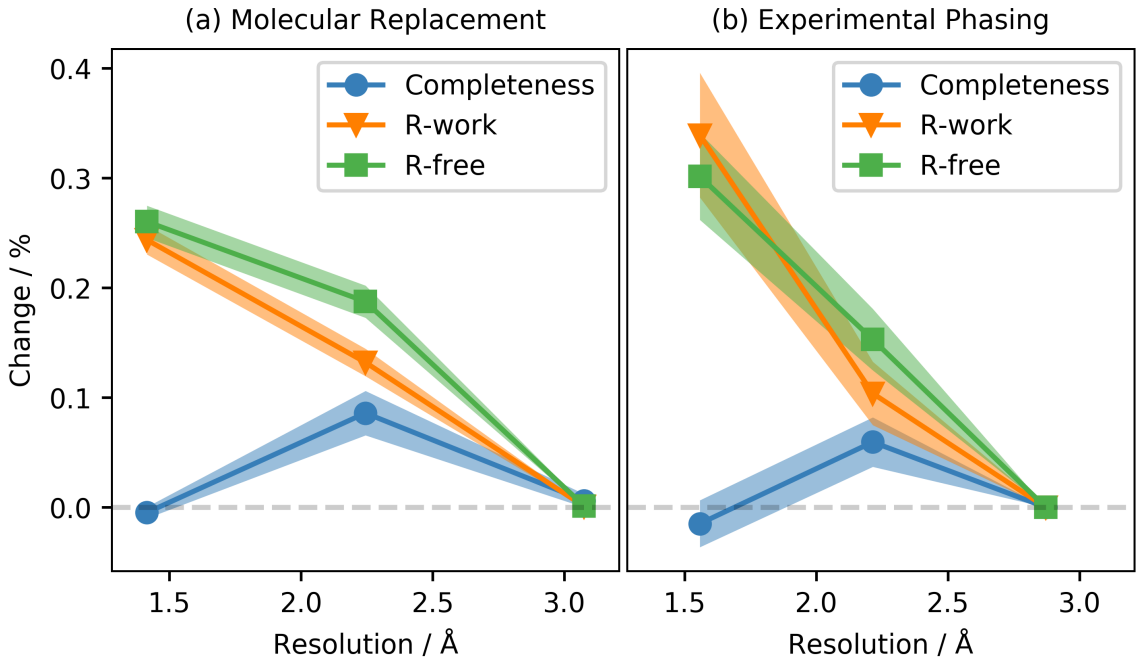

Figure 4.16: Change in completeness, R-work and R-free on the removal of the final side chain fixing step for 1348 MR and 193 EP cases. Points show the mean change in 3 bins and the shaded area shows one standard error above and below the mean.

The step operates on the output model, i.e. the model from the cycle with the lowest R-free, but only if R-work is less than 30% and the data resolution is better than 2.5 Å. After the side chains are rebuilt, the model is refined with 5 cycles of REFMAC but, as with the water and dummy atom addition steps, the modified structure is only accepted if it gives an improvement in R-free.

Figure 4.16 shows the change in completeness and R-factors when the final side chain rebuilding step is removed. Again, there is a similar change for the molecular replacement and experimental phasing test sets. This is expected as the step is only applied to final models where R-work is less than 30%, which should not be affected by the source of initial phasing. As the step is only used on cases with resolution better than 2.5 Å, i.e. all the cases in the high-resolution bin and most of the cases in the mid-resolution bin, no change is seen in the low-resolution bin and the mean change in the mid-resolution bin lessened slightly. The R-factor improvement from the step is small but as only a fraction of side chains in the model are being changed a large difference was not anticipated. Interestingly, the side chain rebuilding step reduces completeness in the mid-resolution bin due to real-space refinement of the main chain. To try and counter this, the threshold for main chain correctness of 0.25 times the median could be increased so more problematic areas of the main chain are not refined using this procedure.

4.3 Summary

A number of developments have been made to the BUCCANEER model building pipeline in CCP4i2:

- Solvent, scaling and hydrogen atom generation options in REFMAC.
- An increased number of cycles with automatic stopping criteria.
- Selection of the output model with the lowest R-free.
- Machine-learned pruning of chains, residues and side chains.
- Initial refinement of molecular replacement models using SHEETBEND.

- Default addition of waters using COOT.

However, as of CCP4 7.0.009, the new pruning options are not used by default. They will become default in a future release after more testing is done to ensure compatibility with uncommon pipeline options. Additionally, a new GUI independent pipeline called ModelCraft has been developed with further changes:

- Simplified automatic stopping criteria.
- Default use of machine-learned pruning.
- Density modification using PARROT.
- Addition of dummy atoms using COOT.
- Final rebuilding of side chains using COOT.

ModelCraft is not yet distributed with CCP4 but is available to install as a separate command line utility [113]. ModelCraft is much better than the previous BUCCANEER pipeline at solving molecular replacement cases, at the cost of taking more time to run. This has been shown for structures with a range of resolutions and initial model qualities. ModelCraft also gives improved results for structures starting from experimental phases, but a larger test set with more difficult cases is needed to explore this difference further.

Figure 4.17 shows the change in completeness and R-factors on removing each of the ModelCraft steps discussed in this chapter for the molecular replacement test set. As discussed earlier, the steps are likely to be interdependent, so removing multiple steps may have a different effect on performance. The step that is giving the biggest independent improvement is PARROT density modification. The mean improvement comes from a distribution where the majority of test cases are unaffected but some have improved greatly, i.e. from a very poor fragmented model to one that is mostly correct. Removing the side chain rebuilding step gives the smallest difference, although this is expected as it is only performed once at the end of the pipeline and only a small number of atoms are affected.

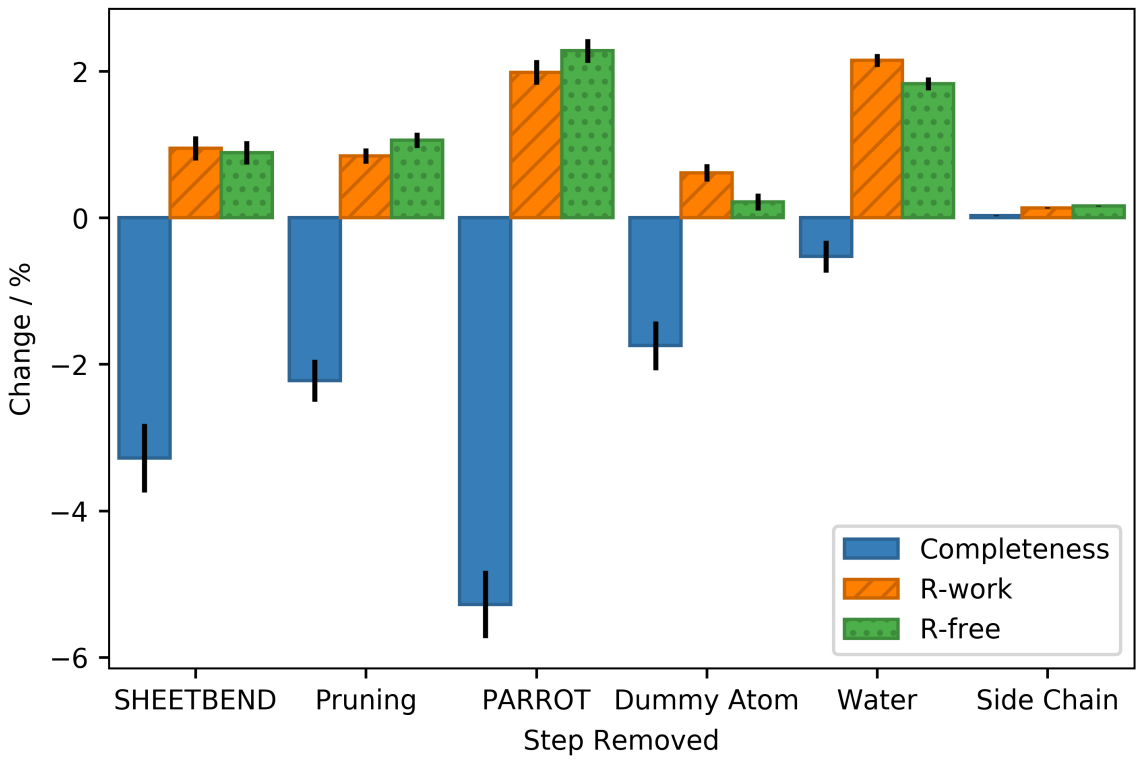

Figure 4.17: The mean change in completeness, R-work and R-free on removing various steps in ModelCraft for 1343 molecular replacement test cases. Error bars show one standard error either side of the mean.

The speed of ModelCraft could be improved by reducing unnecessary steps. The original pipeline only used a small number of BUCCANEER and REFMAC cycles. For many structures this is sufficient to produce a complete model and the additional steps slow down the process without providing much benefit. However, for some structures the extra steps are needed for model building to be successful. A simple approach would be to try the quickest method first and only use slower steps if they do not produce satisfactory results.

A more complicated, but potentially more useful, approach would be to predict the improvement that will be gained by each step as well as how long it will take to run, and use this prediction to decide whether to include the step. For example, the resolution and current R-factors could be used to predict how the R-factors will change after adding dummy atoms. This could even be taken a step further to move away from the traditional cyclic control system to one that chooses the next step by examining both the current state of the model and a history of previous steps and

how successful they were.

So far, ModelCraft only builds protein molecules using data from X-ray crystallography. It should be extended to work with data from cryo-EM experiments as well. The steps in the pipeline may be quite different, e.g. phase improvement techniques are not needed in cryo-EM because the data are measured in real space, but using the same pipeline framework will mean there is less duplication of development effort in other areas such as the input parsing, file handling and reporting. The pipeline should also build non-protein components such as nucleic acids, carbohydrates and ligands, for example using NAUTILUS [114], SAILS [115] and COOT [15].

Chapter 5

Buccaneer Steps

The previous chapter covered developments that have been made at the pipeline level, i.e. outside of the BUCCANEER binary. This chapter covers internal changes to BUCCANEER that have been tested. Making changes at this level has the advantage that fast C++ code can be used without the need for potentially costly input/output operations or changing the molecule or map representations. However, there are limitations as it is not possible to perform global model refinement or use functions from other programs such as COOT.

5.1 Side Chain Building

A protein model is often separated into the main chain (or 'backbone') and amino acid side chains, and the model is built starting with the main chain. This makes sense because the main chain conformation, i.e. the secondary and tertiary structure of the protein, is a larger scale feature that can be determined even at very low resolution. Therefore there is more confidence that the main chain atom positions are correct. Side chains are smaller scale features that cannot be seen at very low resolution, but are crucial for enzyme activity or the binding of drug molecules so it is essential they are built correctly.

Side chains make up a significant proportion of the atoms in a protein, although there is quite a lot of variation in how large and how frequent the different residue

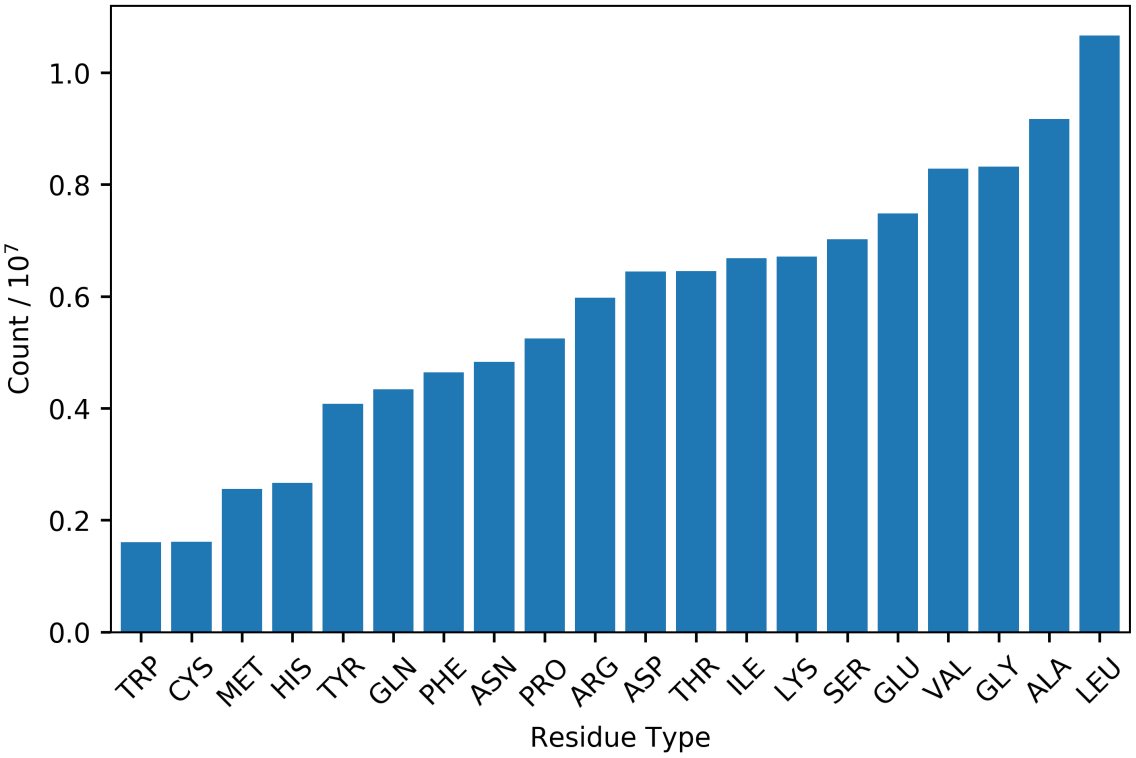

Figure 5.1: The number of occurrences of each residue type in the PDB.

types are. Figure 5.1 shows the number of occurrences of each residue type in the PDB [17]. The totals were obtained from the annotated sequences of protein chains, but only for residues that are also present in the model. Leu is the most common residue, occurring 6.6 times as often as Trp or Cys. This distribution may not be representative of all proteins. The PDB contains mostly well-ordered, soluble proteins of scientific interest, often repeated in multiple entries. For example, membrane proteins are likely to be under-represented because they are less soluble and hence more difficult to crystallise. It is expected they will have a different proportion of hydrophobic and hydrophilic residue types. Water-soluble proteins have a hydrophobic core and hydrophilic surface to aid folding and stability in an aqueous environment. There may also be some artefacts in the occurrences of certain residue types, for example histidine may be slightly over-represented due to structures with his-tags for purification that have not been removed.

Figure 5.2 shows the number of electrons in the side chain from the γ position onwards for each residue type. It assumes that Arg, His and Lys are positively charged, Asp

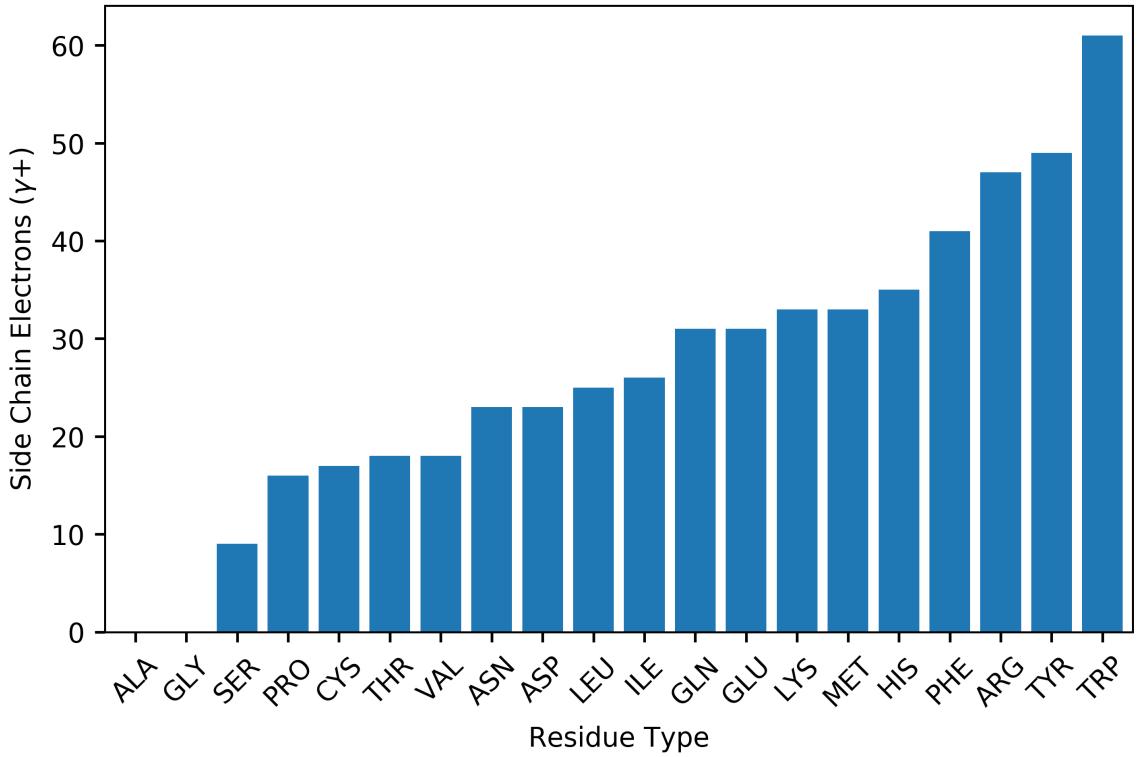

Figure 5.2: The number of electrons in the side chain from the γ position onwards for each residue type. Assuming Arg, His and Lys are positively charged, Asp and Glu are negatively charged, and the other residue types are neutral.

and Glu are negatively charged, and the other residue types are neutral. If the positions of the backbone atoms (N, Cα and C) are known then the position of Cβ is also known, subject to very small differences in the bond lengths and angles. However, the conformation of the side chain atoms from the γ position onwards can vary greatly due to rotations around the Cα-Cβ bond (χ1) and angles further along the side chain (χ2, χ3 and χ4). Tryptophan has the most electrons in the side chain, followed by the other residues with aromatic rings and arginine.

Table 5.1 shows the raw values from Figures 5.1 and 5.2 along with the percentages to which these values correspond. The frequency percentage assumes that all side chains are one of these 20 types, i.e. ignoring the 285207 residues with a code of X in the protein sequence. The percentage of electrons in the side chain assumes that the residue is in the middle of a peptide chain, i.e. not an N-terminal or C-terminal residue. Summing the product of these two percentages for each residue type reveals that the rotatable atoms of side chains account for 35% of the scattering matter

Table 5.1: Frequency in the PDB, electrons in the side chain from the γ position onwards, and number of rotatable bonds for each residue type.

Residue Type	Frequency	γ+ Electrons	Rotatable Bonds
ALA	9171422 (7.99%)	0 (0%)	0
GLY	8322972 (7.25%)	0 (0%)	0
SER	7024726 (6.12%)	9 (20%)	1
PRO	5249559 (4.57%)	16 (31%)	?
CYS	1612386 (1.40%)	17 (31%)	1
THR	6454819 (5.62%)	18 (33%)	1
VAL	8280563 (7.21%)	18 (33%)	1
ASN	4829809 (4.21%)	23 (38%)	2
ASP	6443342 (5.61%)	23 (38%)	2
LEU	10662337 (9.29%)	25 (40%)	2
ILE	6682619 (5.82%)	26 (42%)	2
GLN	4338463 (3.78%)	31 (46%)	3
GLU	7480114 (6.52%)	31 (46%)	3
LYS	6710140 (5.85%)	33 (47%)	4
MET	2559097 (2.23%)	33 (47%)	3
HIS	2661740 (2.32%)	35 (49%)	2
PHE	4641388 (4.04%)	41 (53%)	2
ARG	5972795 (5.20%)	47 (56%)	4
TYR	4078459 (3.55%)	49 (57%)	2
TRP	1607652 (1.40%)	61 (62%)	2

of a protein on average. This is a significant proportion, therefore a protein model built as polyalanine will have much worse phases than a model with the correct side chains.

Table 5.1 also shows the number of rotatable bonds for each residue type. Residues with aromatic side chains (His, Phe, Tyr and Trp) have large side chains, but they are not very flexible with only two rotatable bonds. Arginine and lysine are the most flexible side chains with 4 rotatable bonds. The rigid guanidinium group at the end of arginine means the conformation is usually visible in the density, but lysine does not have this and is often found at the surface of proteins without a well defined

conformation. Proline is unique as the Cδ atom is bonded to the N atom to form a 5-membered ring, leaving the side chain with very little flexibility.

The final step in BUCCANEER is to build the side chains using only the position of the backbone atoms and the density map. The algorithm starts with a list of rotamers from CLIPPER [116]. Each rotamer from the library is built, then residues with large side chains (Arg, Gln, Glu, His, Lys, Met, Phe, Trp and Tyr) are rotated by ±18° and ±36° about the Cα-Cβ bond (χ1). Each of these conformations, i.e. the base rotamers and the rotated variants, are scored using the mean density Z-score at the γ, δ, ε and ζ atom coordinates, with a higher score indicating a more favourable conformation. The best scoring conformation is built for each residue and then clashes between residues are considered. Two residues are marked as clashing if one of the γ+ side chain atoms is within 1.25 Å of any atom in the other residue. Clashes are fixed by generating conformations as described above and iterating through all possible combinations. The scores of the individual conformations are added together and a penalty is applied for combinations that clash. The combination with the best score is chosen unless it has a clash, in which case both residues are truncated to Cβ. The modifications I have made to the current side chain building algorithm by changing the rotamer library, the search function and the scoring function are discussed in Sections 5.1.1 to 5.1.3.

5.1.1 Rotamer Library

A rotamer library is a list of commonly occurring side chain conformations. An individual rotamer can be represented in several different ways. A simple description would be a list of χ angles, which are the rotatable dihedral angles of a side chain. This description is usually sufficient. Information about other bond angles and bond lengths might not be included and reference values would be assumed. A full description of the rotamer would involve either specifying all these parameters using an internal coordinate representation or providing Cartesian coordinates for the atoms.

A rotamer library used by many programs is the 'Penultimate' rotamer library, derived from the Top240 structure database [40]. This was expanded to the Top500 structure database soon after, but the side chain rotamers were not updated [41]

from this expansion. BUCCANEER retrieves its rotamers from CLIPPER [116], which uses a subset of the Penultimate rotamers available in COOT. There is a more recent successor to this library called the 'Ultimate' rotamer library, which is based on the much larger Top8000 structure database where stricter filtering has been applied both at the chain level and residue level [117]. It is important to only look at residues from well-built high resolution structures where the data clearly support the side chain conformation, otherwise the distribution of conformations will be biased by the building tools used. The number of rotamers for each residue type in different libraries is shown in Table 5.2.

Table 5.2: The number of rotamers for each residue type in different rotamer libraries.

Type	Penultimate	COOT	CLIPPER	Ultimate
ARG	34	33	19	60
ASN	7	7	7	5
ASP	5	5	5	4
CYS	3	3	3	3
GLN	9	9	7	13
GLU	8	8	7	9
HIS	8	8	8	8
ILE	7	7	5	7
LEU	5	5	4	8
LYS	27	24	13	47
MET	13	13	12	23
PHE	4	4	4	4
PRO	3	3	3	2
SER	3	3	3	3
THR	3	3	3	3
TRP	7	7	7	7
TYR	4	4	4	4
VAL	3	3	3	3

The COOT library contains nearly all the rotamers from the Penultimate library. Only one uncommon Arg rotamer and three uncommon Lys rotamers are missing using the default probability cutoff. CLIPPER uses a subset of the rotamers from

COOT but the probability cutoff is much stricter and so contains fewer rotamers. The Ultimate rotamer library contains many more rotamers than the previous Penultimate library but the increases are mainly for the amino acids with the most flexible side chains. The number of rotamers nearly doubles for Arg, Lys, and Met, while Gln, Glu, and Leu have small increases. Other residues have the same number of rotamers between the new and old libraries, other than Asn, Asp and Pro which have less. Individual residue types should be examined in more detail to ensure performance is not affected by over or under sampling.

For the Top500 library, Lovell *et al.* also calculated smoothed rotamer score distributions [41]. The same was done by Hintze *et al.* for the Top8000 library [117]. Each residue in the database provided an individual observed data point in the multi-dimensional χ space for that residue type. The number of dimensions is the same as the number of χ angles in the side chain. A kernel density estimation was performed in two steps: firstly using a cosine function with a fixed width and then a cosine function with a variable width that depends on the initially estimated local density, with a larger width used for more sparse areas. This ensured smooth contours separate the allowed, favoured and outlier regions, whilst keeping steep transitions in regions of rapidly changing frequency. The density was normalised to a rotamer score that represents the fraction of rotamers with worse scores, i.e. the most observed conformation in the distribution has a score of 1 and unobserved conformations have a score of 0. The number of bins for each angle depends on the number of dimensions. For example, cysteine has one χ angle divided into 360 (1°) bins and arginine has four χ angles divided into 36 (10°) bins. As with Ramachandran scores, $< 0.3\%$ is considered an outlier, $< 2\%$ is allowed and $\geq 2\%$ is favoured [41, 118].

Figure 5.3 shows a comparison between the Leu rotamers in the Penultimate and Ultimate rotamer libraries as well as a comparison between the Top500 and Top8000 rotamer score distributions. Because Cβ and Cγ are both sp^3 hybridised, staggered conformations are preferred for both χ1 and χ2. Individual rotamers are named based on the staggered conformations, i.e. p (plus) for 60°, t (trans) for 180° and m (minus) for 300° (-60°). For example, the pp rotamer is in the bottom left and the mp rotamer is in the bottom right. The most common rotamer is mt, followed by tp, which occurs roughly half as often. The other rotamers are much rarer.

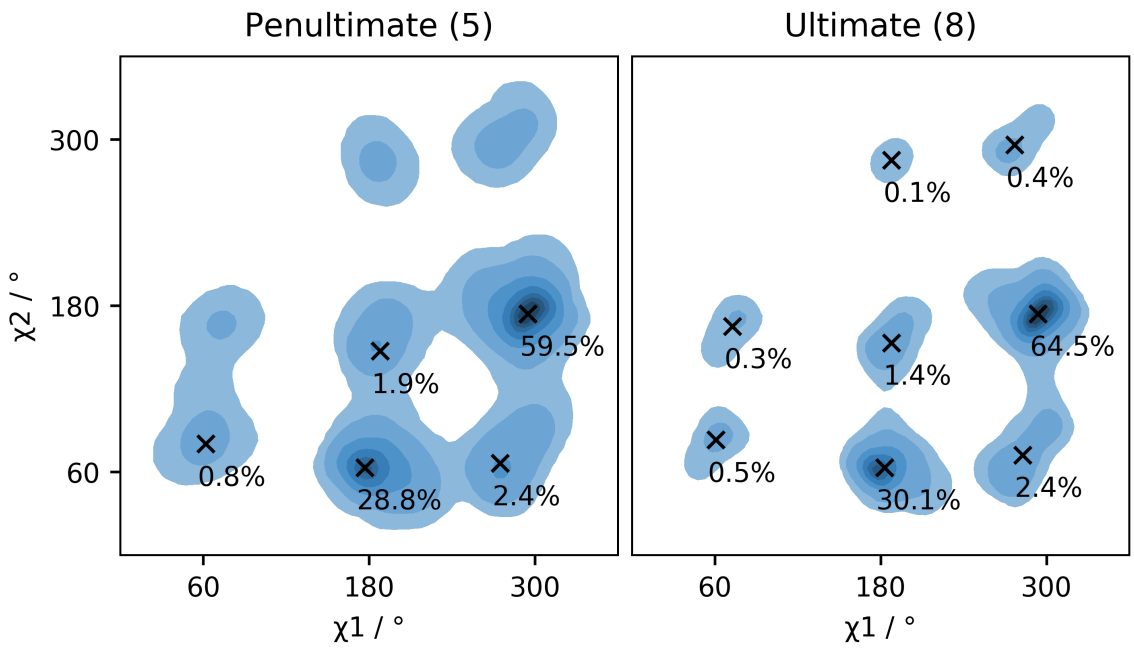

Figure 5.3: Comparison of the LEU rotamers in the Penultimate and Ultimate rotamer libraries. Penultimate rotamers are on the left over the Top500 rotamer score distribution. Ultimate rotamers are on the right over the Top8000 rotamer score distribution. Rotamers are shown as crosses labelled with the observed frequency in the database. The rotamer score distributions are contoured at 0.3%, 2%, 10%, 20%, 40%, 60% and 80%.

The Penultimate library contains 5 Leu rotamers, but the pp rotamer is missing from CLIPPER due to the frequency cutoff used. In the Ultimate library, there are additional pt, tm and mm rotamers that are not present in the Penultimate library but can still be seen in the Top500 rotamer score distribution. The Top500 and Top8000 rotamer score distributions have a similar pattern, but the 0.3% and 2% areas are much larger in the Top500 data. This could be due to both the paucity of observations in these regions in the smaller database and differences in the smoothing functions used.

All 9 possible staggered conformations of Leu are shown in Figure 5.4. The most common rotamers, mt and tp, are favoured because both Cδ atoms are pointing away from the backbone. The pm rotamer is extremely unfavourable, and not visible in the allowed rotamer score distributions, because both Cδ atoms are close to the backbone.

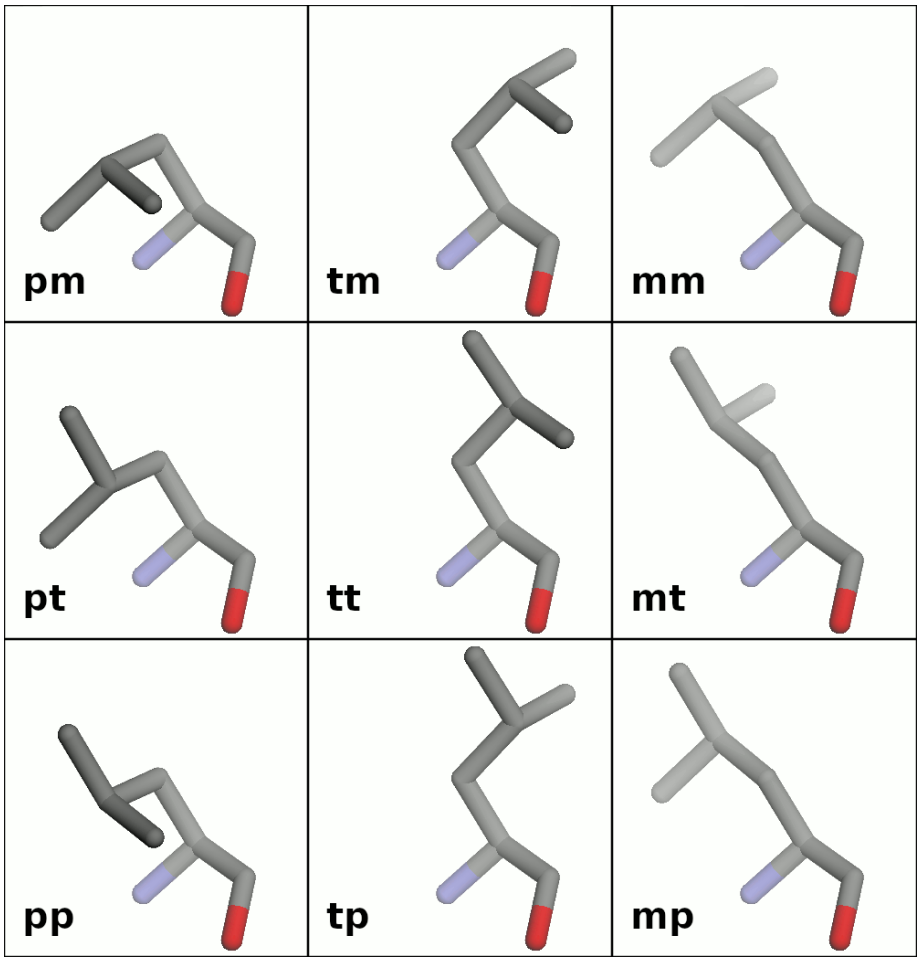

Figure 5.4: All 9 staggered LEU conformations with idealised χ angles of 60° (p), 180° (t) and −60° (m).

Asn is an example where the Ultimate library contains fewer rotamers than the Penultimate library. A comparison between the Asn Penultimate and Ultimate rotamers, and the Top500 and Top8000 rotamer score distributions is shown in Figure 5.5. Staggered conformations are still preferred for $\chi 1$ but, because the Cγ atom is sp^2 hybridised, it is possible for $\chi 2$ to rotate more freely. The naming scheme for the rotamers then changes to represent the final χ angle as a number. The Penultimate library contains 7 rotamers: p30, p-10, t30, t-20, m120, m-80 and m-20. However, the Ultimate library only contains 5 discrete rotamers. Some pairs of Penultimate rotamers are now represented by a single rotamer, e.g. p30 and p-10 are now represented by p0. Similarly, t30 and t-20 have merged into t0 and m-80 and m-20 have merged into m-40. In addition, the Ultimate library contains an uncommon t160 rotamer that was not present in the Penultimate library and the

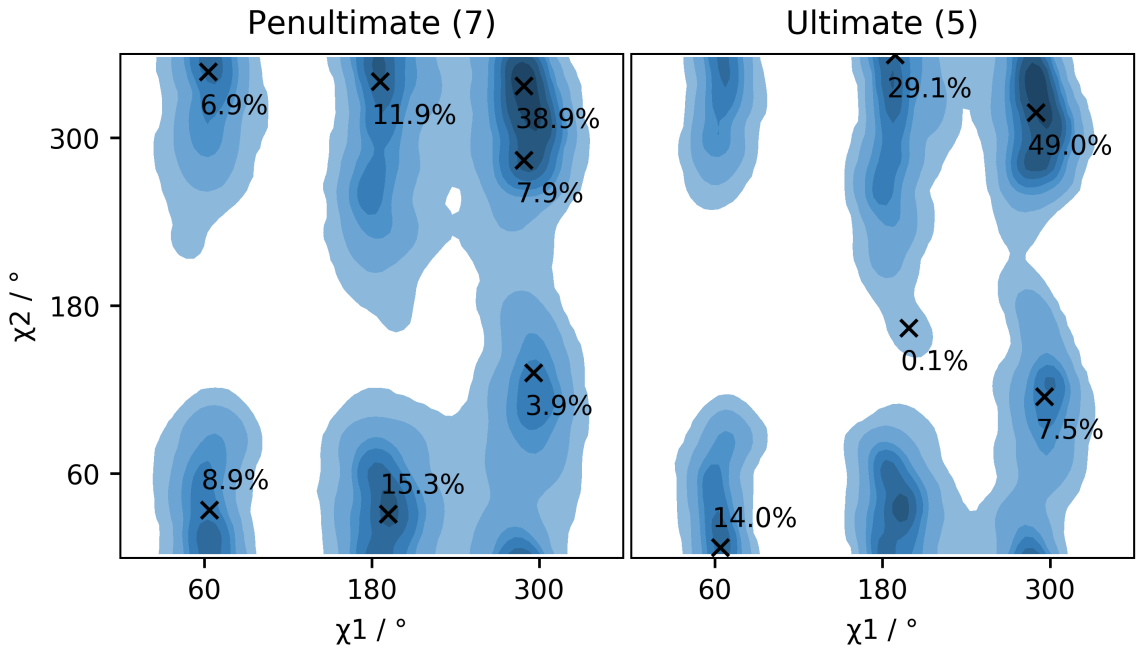

Figure 5.5: Comparison of the ASN rotamers in the Penultimate and Ultimate rotamer libraries. Penultimate rotamers are on the left over the Top500 rotamer score distribution. Ultimate rotamers are on the right over the Top8000 rotamer score distribution. Rotamers are shown as crosses labelled with the observed frequency in the database. The rotamer score distributions are contoured at 0.3%, 2%, 10%, 20%, 40%, 60% and 80%.

m120 rotamer has shifted slightly to m110. As with Leu, the rotamer score distributions are similar but the outer contours in the Top500 distribution cover a larger area.

5.1.1.1 Method

In order to test whether BUCCANEER performs better or worse using the Ultimate rotamer library, the Ultimate library rotamers were incorporated into CLIPPER. CLIPPER already contains two rotamer libraries: the subset of the Penultimate rotamer library discussed earlier, which is the default, and the Dunbrack and Karplus rotamer library [119], so very few code changes were required to add a third library. The main difficulty encountered was the translation between data formats. CLIPPER stores rotamers using XYZ coordinates for each

Table 5.3: Data for cysteine in the Ultimate rotamer library, where n is the number of occurrences in the Top8000 database, frequency% is the percentage occurrence, and mean and standard deviations are provided for each χ angle and bond angle.

rotamer	p	m	t
n	2962	9301	4399
frequency%	17.73	55.67	26.33
chi1_mean	65	-65	-178
chi1_esd	9.0	8.2	8.2
CA_CB_SG_mean	114.6	113.6	113.7
CA_CB_SG_esd	1.68	1.86	1.94
CA_C_O_mean	120.5	120.4	120.5
CA_C_O_esd	0.98	0.87	0.86
CB_CA_C_mean	110.5	109.5	110.3
CB_CA_C_esd	1.50	1.63	1.30
N_CA_C_mean	111.3	111.5	109.7
N_CA_C_esd	2.78	2.42	2.44
N_CA_CB_mean	110.9	110.6	110.0
N_CA_CB_esd	1.23	1.07	1.37

Table 5.4: Rotamer data for cysteine converted from the values in Table 5.3 into the CLIPPER format, where num_rota is the number of rotamers; rota is the index of the rotamer; rota_prob is the occurrence as a fraction; num_atom is the number of non-hydrogen atoms in the side chain; atomname is the name of the atom; and x, y and z are atom coordinates.

num_rota	rota	rota_prob	num_atom	atomname	x	y	z
3	0	0.5567	2	CB	-0.932	-1.213	0.017
3	0	0.5567	2	SG	-2.165	-1.198	-1.308
3	1	0.2633	2	CB	-0.915	-1.226	-0.005
3	1	0.2633	2	SG	-2.045	-1.296	1.408
3	2	0.1773	2	CB	-0.958	-1.193	0.006
3	2	0.1773	2	SG	-0.149	-2.809	0.098

atom from Cβ onwards, assuming a standard position and orientation of the backbone. The Ultimate library describes rotamers using mean and standard deviations of χ angles and bond angles, including those of the main chain. To perform the conversion, I wrote a program that reads the mean angles from the Ultimate library and combines them with bond lengths from COOT to build up the rotamer. It also moves the rotamer into a standard orientation and outputs the coordinates for CLIPPER. An example of the rotamer data before and after conversion is shown in Tables 5.3 and 5.4.

Testing was done on 53 structures from the Joint Centre for Structural Genomics (JCSG) [42]. This is a subset of the structures used by Cowtan (2006) [37] with five structures removed (1VJO, 1VKW, 1VPJ, 1VR9 and 1ZEJ). These use experimentally phased data that were processed as described in Section 2.1 to produce an initial density modified map from PARROT [71] and a deposited model that has been refined against these data using REFMAC [43].

Two tests were performed. The first was a single iteration of the BUCCANEER side chain building step on the refined deposited model using the density modified map. This isolates the side chain building performance as there is no error in the main chain but the phases are not ideal. The second test was to run the CCP4i BUCCANEER pipeline starting from the density modified phases without a model.

5.1.1.2 Results and Discussion

For the isolated side chain building step, side chain RMSD (from Cγ onwards) was calculated between the built model and the refined deposited model for each residue. Symmetry was accounted for by flipping Arg, Asn, Asp, Gln, Glu, His, Phe, and Tyr residues and selecting the conformer with the lower RMSD value. Figure 5.6 shows the change in side chain RMSD for each residue type when the rotamer library is changed from the CLIPPER Penultimate library to the Ultimate library. A more negative value shows an improvement where the Ultimate library builds rotamers closer to the deposited structure.

Proline shows the largest improvement by far, with a very small standard error. The side chain is made up of a five membered ring where the Cδ atom attaches to the

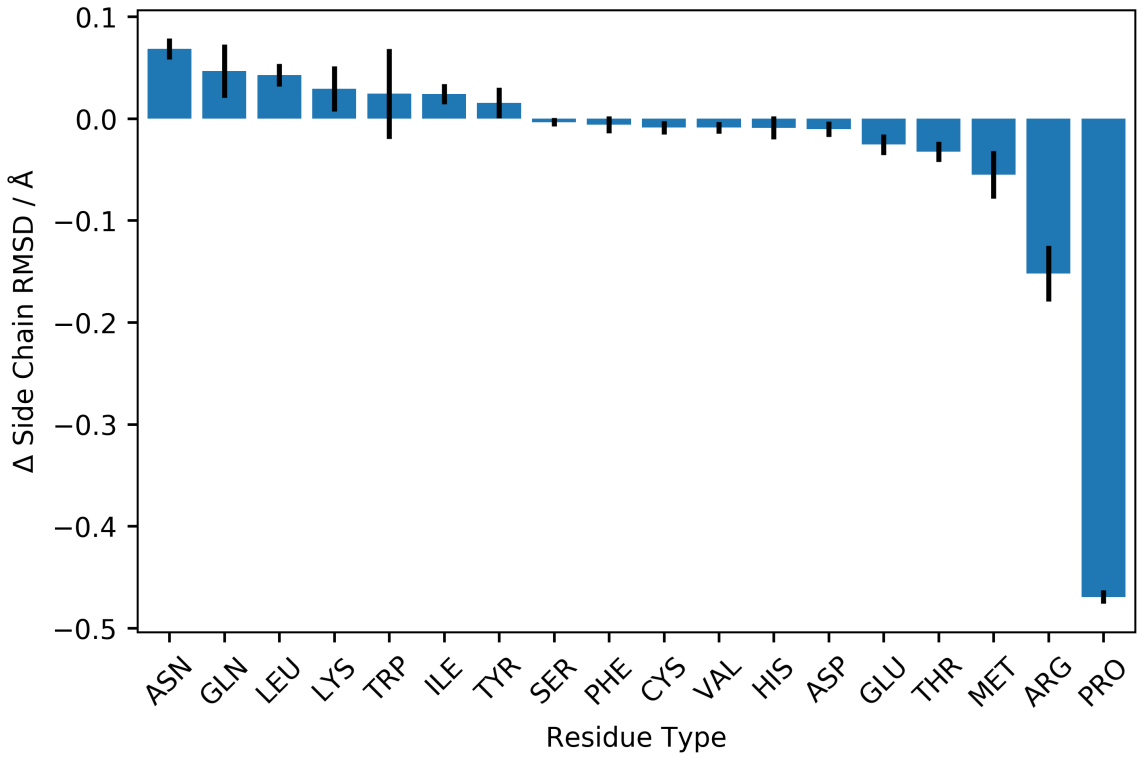

Figure 5.6: The change in side chain RMSD for each residue type when changing the side chain building step in BUCCANEER to use the Ultimate rotamer library instead of the CLIPPER Penultimate library. A single side chain building step was performed on 53 structures from the JCSG. Bars show the mean change ± one standard error.

backbone N, so there is very little room for flexibility. The pyrrolidine ring, as with other unsaturated 5-membered rings, has two ring puckering modes. For proline, these are characterised by the position of the Cγ atom: if χ1 is slightly negative it is an exo (or up) rotamer and if χ1 is slightly positive it is an endo (or down) rotamer. These two rotamers are present with comparable frequencies overall, although the exo rotamer is much more common in α-helices and the endo rotamer is much more common in β-strands [40]. The Ultimate rotamer library only contains these two rotamers but the Penultimate library contains a third rotamer, which is an endo proline rotamer for a cis-peptide.

Figure 5.7 shows the proline rotamers from the CLIPPER library. The difference in RMSD is not due to the number of rotamers because the two Penultimate endo rotamers are almost identical. The difference is mainly in the χ2 angles, which are

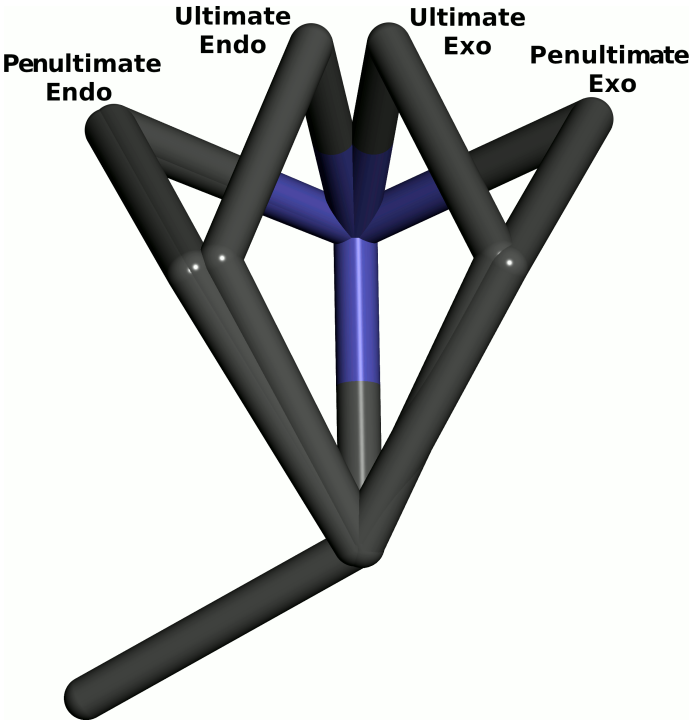

Figure 5.7: The proline rotamers in CLIPPER viewed along the Cβ-Cα bond. Carbon is drawn in grey and Nitrogen in blue. The CLIPPER Penultimate library has two endo rotamers that are almost identical: one for a trans-peptide and one for a cis-peptide.

0° for all the Penultimate rotamers. This is likely a mistake as the average χ2 angles are missing in the Penultimate rotamer descriptions of proline (although a range is provided) [40]. The Ultimate rotamer descriptions of proline include average values for χ1, χ2 and χ3 [117] so this was not a problem when translating the Ultimate rotamers for CLIPPER.

Arginine has the next biggest RMSD improvement in Figure 5.6. The number of arginine rotamers has increased from 19 in the CLIPPER Penultimate library to 60 in the Ultimate library. Methionine also has an large RMSD improvement with the number of rotamers increasing from 12 to 23. However, for lysine the number of rotamers increased from 13 to 47 and this leads to a worse RMSD, although only at a significance of around one standard error. This could be owing to the frequent occurrence of disordered lysine residues where the density does not obviously favour one rotamer over another. It could also be because the new rotamers do not sample the conformational space as effectively. Asparagine has the biggest increase in

RMSD where the older Penultimate rotamers (shown in Figure 5.5) perform better. Unfortunately, the higher coverage of rotamers for leucine (shown in Figure 5.3) also leads to a higher RMSD. Each residue type should be considered individually to see which rotamers work best. Another point to note is that this test did not include any refinement after the BUCCANEER side chain building step, and it is likely many of the RMSD differences would be resolved by this.

For the CCP4i pipeline test, R-free fell by $0.5\% \pm 0.3\%$ after changing the CLIPPER Penultimate rotamer library to the Ultimate rotamer library. This is a relatively small improvement, but a large change in R-free is not expected for a small algorithm change such as modifying the rotamer library. The change is also not very significant and a larger test set is needed to reduce the standard error. The isolated side chain building step showed that having many more arginine rotamers is an improvement when the main chain is correct, but early on in the pipeline when the phases are poor and there are errors in the main chain it is probably better to build common rotamers instead of over-interpreting the density.

5.1.1.3 Tryptophan Sampling

Tryptophan residues often have very obvious density for the rigid indole side chain so it is frustrating for users when automated building gets the conformation wrong. In many cases the $\chi 1$ angle is correct but $\chi 2$ is flipped by 180°, which is relatively understandable as the first ring still fits well in the density. An example of this can be seen in Figure 5.8. These cases can be easily fixed by flipping the side chain. In other examples however the side chain is neither correct nor flipped by 180°. Because the side chain is very large, refinement may be less likely to move an incorrect rotamer to a correct one because of steric hindrance from other residues. This is especially true for residues that are not on the protein surface.

The tryptophan rotamers in the Penultimate and Ultimate rotamer libraries are shown in Figure 5.9, along with the Top500 and Top8000 rotamer score distributions. Like the Asn rotamer score distributions in Figure 5.5, the $\chi 1$ angle (sp^3 to sp^3) is roughly constrained to the three staggered regions, but the $\chi 2$ angle (sp^3 to sp^2) has a much larger range of allowed values. Both libraries have seven rotamers that are very similar. The biggest change is between the Penultimate t90 rotamer and the

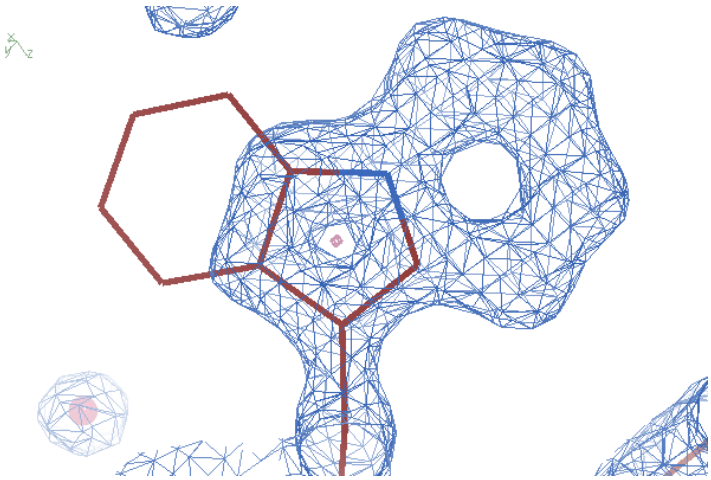

Figure 5.8: A tryptophan side chain in a 1.36 Å resolution structure. The density is for the correct conformation. The χ2 angle has been rotated by 180° using the Sidechain 180° Flip tool in COOT to show that one of the rings still fits the density well.

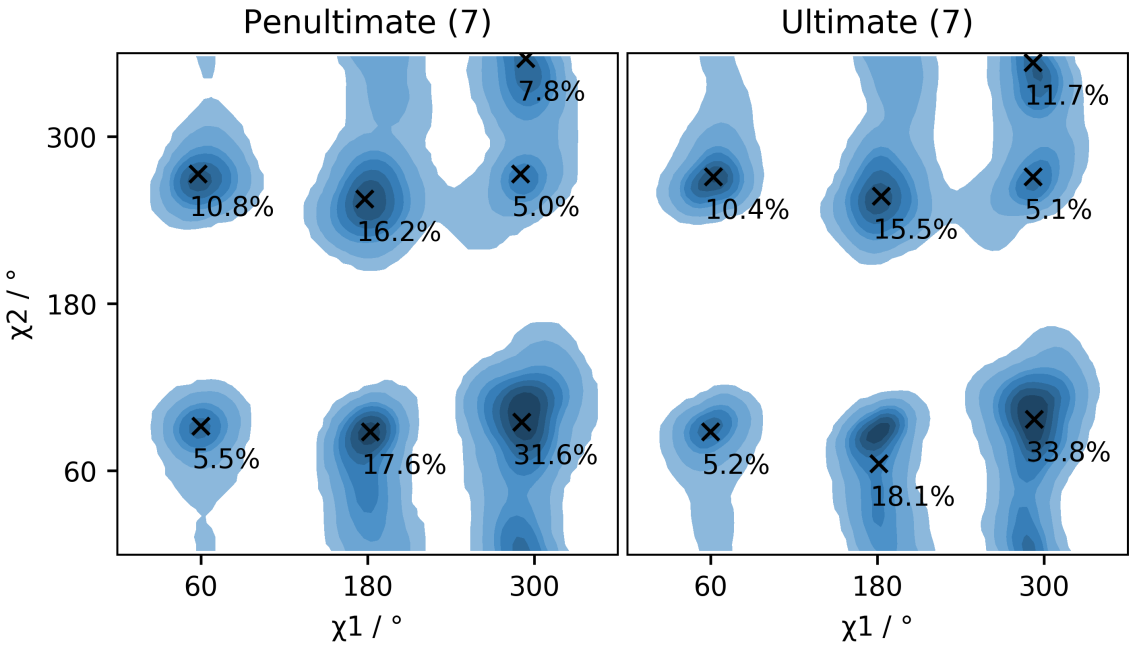

Figure 5.9: Comparison of the TRP rotamers in the Penultimate and Ultimate rotamer libraries. Penultimate rotamers are on the left over the Top500 rotamer score distribution. Ultimate rotamers are on the right over the Top8000 rotamer score distribution. Rotamers are shown as crosses labelled with the observed frequency in the database. The rotamer score distributions are contoured at 0.3%, 2%, 10%, 20%, 40%, 60% and 80%.

Ultimate t60 rotamer, where the t60 rotamer is further from the rotamer score peak.

As can be seen in Figure 5.6, changing libraries does not lead to a significant difference in the accuracy of tryptophan building because the Penultimate and Ultimate rotamers are very similar. A new rotamer library was tested that sampled $\chi 1$ at 60°, 180° and 300°, and $\chi 2$ every 20°. In total this gives 54 conformations, which can be seen on the left of Figure 5.10 over the Top8000 rotamer score distribution. This was tested on a single high resolution crystal structure using an isolated side chain building step. The input model for this was the output from a previous BUCCANEER pipeline, so the main chain was not entirely correct. The new rotamer library managed to correctly build more tryptophan residues than the previous library, but still not all. The examples where it did not build the correct conformation had their Cβ positions displaced from the deposited model by around 0.3 Å.

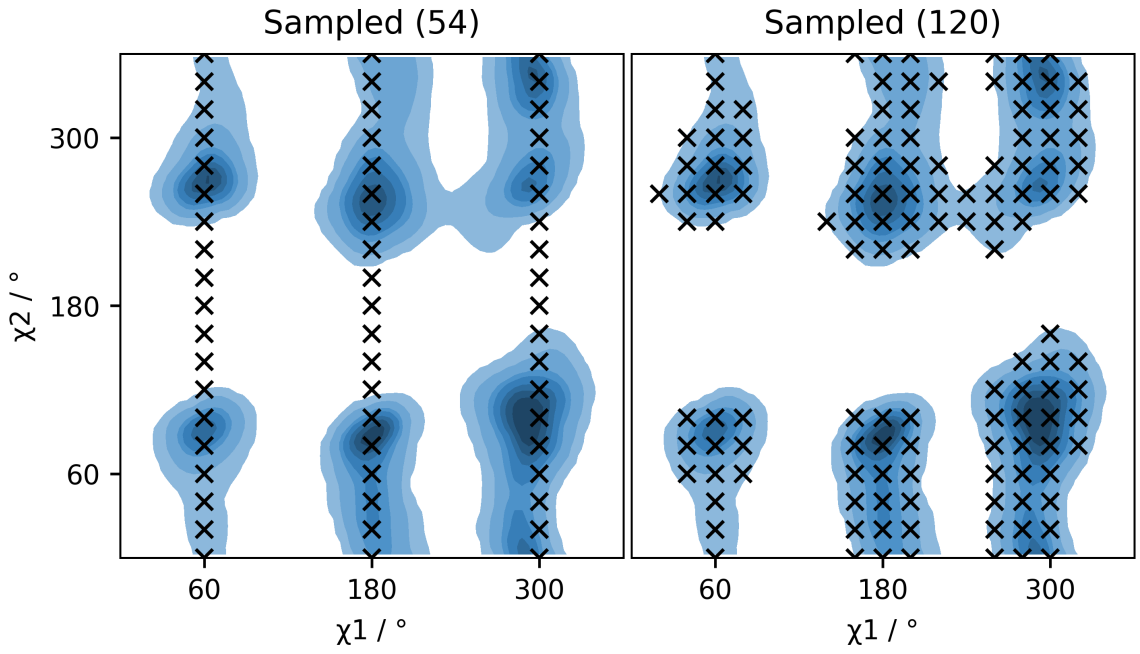

Figure 5.10: Regularly sampled tryptophan conformations over the Top8000 rotamer score distribution. Conformations are shown as crosses. The rotamer score distributions are contoured at 0.3%, 2%, 10%, 20%, 40%, 60% and 80%.

Some of the 54 conformations are very unlikely as a $\chi 2$ angle of around 180° is rarely observed, so another rotamer library was produced that sampled both angles every 20° but only combinations with a rotamer score greater than 0.3% were selected. This

left 120 conformations that can be seen on the right of Figure 5.10. Other options to having these regularly sampled libraries would be to manually select rotamers or to use a greedy algorithm that chooses a rotamer at the highest rotamer score peak before masking the distribution around that rotamer. These new libraries need to be tested on a larger scale to see how useful they are. However, the best rotamer library for any residue will depend on the algorithm used for building and when it is used.

5.1.2 Search Function

The responsibility of the search function is to effectively sample the conformational space of the side chain. If we only consider the χ angle representation of a rotamer then the conformational space has the same number of dimensions as the number of χ angles. Some sampling is already provided by the rotamer library. The only other sampling in the released side chain building algorithm is to rotate the $\chi 1$ angle of residues with large side chains (Arg, Gln, Glu, His, Lys, Met, Phe, Trp and Tyr) by $\pm 18^\circ$ and $\pm 36^\circ$.

Instead of altering the rotamers using discrete rotations, they could be modified through a procedure that optimises the conformation based on the scoring function. A very simple form of optimisation is the Nelder-Mead method, also known as simplex minimisation [100]. A simplex is a shape with $N + 1$ vertices, where N is the number of dimensions. A two-dimensional simplex is a triangle and a three-dimensional simplex is a tetrahedron. Simplex minimisation is a type of optimisation algorithm that uses a simplex to find a local minimum. Given a point in an N dimensional space, the first step is to produce N more points by incrementing in each dimension. All points are then passed to the scoring function, the worst scoring point is identified and the centroid of all other points is calculated. If gradient minimisation is used then the calculation of this centroid is weighted towards better scoring points. A shift vector is then calculated from the worst point to the centroid, and this is used to find a replacement for the worst point. In a normal step, a point at the centroid plus the shift vector is taken. If the normal step scores better than the worst point then an extension step is tested which adds the shift vector again. If the extension step is better then this is used,

otherwise the normal step is kept. If the normal step was worse then a contraction step uses a point halfway between the centroid and the worst point. In the uncommon scenario when both the normal and contraction steps are worse, all points are contracted towards the best scoring point. Convergence of this algorithm indicates that a minimum has been found. Whether this is truly a local minimum depends on the size of the initial increments. If they are large then there is more chance of skipping over a minimum.

5.1.2.1 Method

Two new versions of BUCCANEER were produced that use the CLIPPER Penultimate rotamer library but with altered search functions. The first version simply removed the discrete $\chi 1$ rotations so that only the rotamers in the library were scored and built. The second version performed simplex minimisation on each rotamer in the library and the minimised rotamers were scored. In the released implementation, the function to build a rotamer takes two integers: one is the index of the rotamer in the library and the other is the index of the discrete rotation to be applied. This second parameter was replaced with array of doubles representing rotations about the rotatable bonds in the side chain. The length of the array is the same as the number of rotatable bonds and an array of zeros will simply give the rotamer as described in the library. For each rotamer, an initial simplex was constructed by adding 0.1 rad (5.7°) to each rotatable bond in turn. A difference from the previous discrete search function is that only one conformation is returned per rotamer.

5.1.2.2 Results and Discussion

Figure 5.11 shows the change in RMSD for each residue type when removing the discrete $\chi 1$ rotations for long side chains. There is no change in the building algorithm for Asp, Asn, Leu, Thr or Ile, but these have occasional changes in RMSD due to instances of clashes with longer side chains. For Gln, Glu and Met the rotations to $\chi 1$ do not make a significant difference. For all the other long side chains there is a higher RMSD without the $\chi 1$ rotations so it is recommended to include them for isolated side chain building. However, this does not translate to a big difference

in the performance of the BUCCANEER pipeline. Removing the discrete rotations changes the pipeline R-free by $0.0\% \pm 0.3\%$.

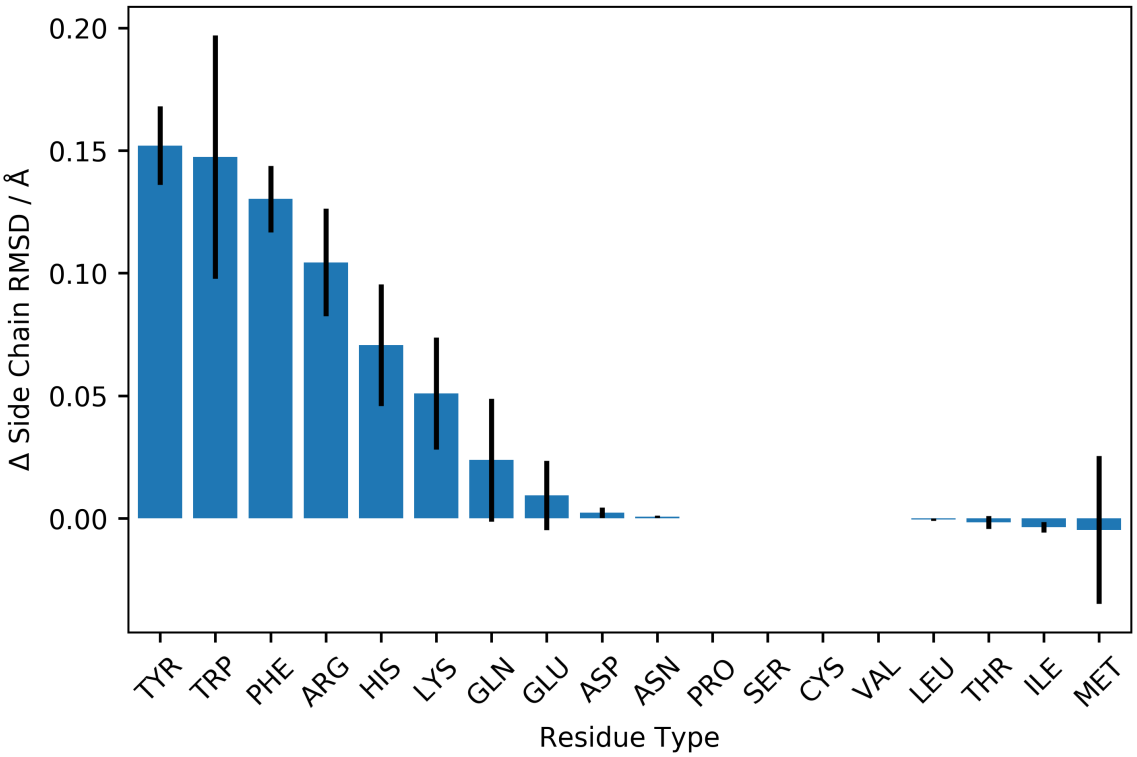

Figure 5.11: The change in side chain RMSD for each residue type when removing the $\chi 1$ rotations of $\pm 18^\circ$ and $\pm 36^\circ$ for long side chains. A single side chain building step was performed on 53 structures from the JCSG. Bars show the mean change $\pm$ one standard error.

Figure 5.12 shows the change in side chain RMSD when modifying the building algorithm from discrete $\chi 1$ rotations of only long side chains to simplex minimisation of all χ angles. Simplex minimisation is performed for all side chains other than Pro. The biggest improvement is Trp, even though this residue only has two rotatable bonds and already had $\chi 1$ rotations. Its rigid, bulky side chain is easy to spot in density, meaning a deeper energy well that has more chance of being chosen if the sampling is able to find it. As can be seen in Figure 5.9, the $\chi 2$ angle of Trp has a wide range of allowed angles, so sampling this angle is important. Tyr, Phe, and His would be expected to show similar results to Trp as they all have two rotatable bonds followed by an aromatic ring. They all still improve with the increased sampling but not as much.

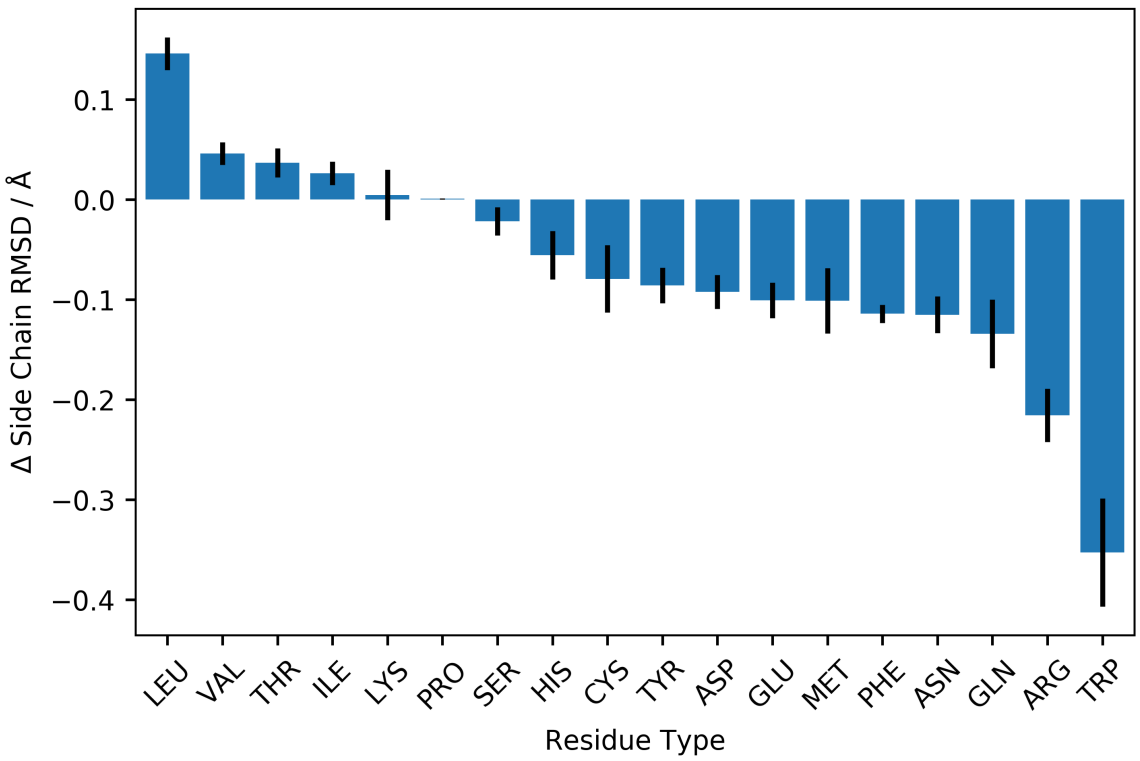

Figure 5.12: The change in side chain RMSD for each residue type on changing to simplex minimisation of each rotamer in the rotamer library. A single side chain building step was performed on 53 structures from the JCSG. Bars show the mean change ± one standard error.

Arg has the second biggest improvement. It and Lys are the most flexible residues, both having four rotatable bonds, but Lys does not improve with the new building algorithm. Perhaps the difference can be explained by Arg, like Trp, having a bulky end group that is easier to place in the density. All four residue types with sp^3 branched side chains have a worse RMSD, especially Leu. This could be because the angles are more constrained to staggered positions and minimisation is moving the library rotamers to be more eclipsed. The density for these rotamers may also be more ambiguous, because the side chains could be rotated by 120° while still partially fitting the density. In the released side chain building algorithm, Glu and Gln both have $\chi 1$ rotations but Asp and Asn do not. They all show similar improvements from the simplex minimisation method.

In summary, this shows that simplex minimisation would be preferred for an isolated side chain building step as most of the residue types perform better. However,

as seen previously, improvements to the the side chain building algorithm for a model with a correct main chain do not mean the same building algorithm will lead to improvements in the model building pipeline. In this case, using simplex minimisation of side chains increases the pipeline R-free by $0.2\% \pm 0.4\%$.

5.1.3 Scoring Function

The scoring function allows the comparison of different side chain conformations by giving them a score reflecting how likely they are to be correct. Usually, the most important part of this score will be a measure of how well the model fits the map. In BUCCANEER, this is done using the mean map Z-score at the γ, δ, ε, and ζ atom coordinates. This is shown in Equation 5.1 where s is the score, Z is the map Z-score at the position of the ith atom and n is the number of atoms.

$$s = -\frac{1}{n}\sum_{i=1}^{n} Z_i \tag{5.1}$$

The negative mean is used so that a lower value indicates a better fit. A more thorough metric would be to analyse regions of both high and low density similar to the fingerprint method used by NAUTILUS [114]. By also looking for low density in the regions immediately surrounding the conformation where voids in the map are expected, the score takes into account the shape of the density and becomes more sensitive.

Although this has not been implemented in BUCCANEER, a scoring function could also include information about the prior likelihood of a conformation, or its energetic favourability. One form of this is a molecular mechanics style forcefield, which describes how the potential energy varies as bond lengths, angles or torsions are changed. It would also be possible to include intermolecular forces such as van der Waals and electrostatic interactions, but in practice these are not often used for model building. Forcefields are widely used in restraint dictionaries of refinement packages but not in crystallographic model building or validation. The forcefield parameters could be determined using physics-based methods, such as quantum mechanics simulations, or they could be knowledge-based, i.e. based on

experimental observations.

Experimental observations may also be used on a larger scale, i.e. not just using the observed distributions of bond lengths and angles but the observed distributions of whole rotamers. The rotamer libraries discussed previously all provide occurrence data but these can only be used with static rotamers. If one of the χ angles of the rotamer is altered then the listed occurrence is no longer applicable. However, the rotamer score distributions from the Top8000 library, which are described in Section 5.1.1, can be used to produce a score for any set of χ angles.

5.1.3.1 Method

The simplex minimisation attempted so far is performed using the density score alone. This may cause a problem where rotamers are being moved into unrealistic conformations. To test this theory, a version of BUCCANEER was written that incorporates the rotamer score into the scoring function used during minimisation. This is shown in Equation 5.2, where r is the rotamer score.

$$s = -\frac{1}{n}\sum_{i=1}^{n} Z_i - \ln r \tag{5.2}$$

The rotamer score is a value between 0 and 1 representing the fraction of rotamers with worse scores, so the logarithm is used to combine it with a Z-score. Very unlikely rotamers will give $\ln r$ a large magnitude used as a penalty in the scoring function. After minimisation has been carried out, the best rotamer is chosen without the rotamer score using Equation 5.1.

Another version of BUCCANEER was produced that combines all the of the new developments to the side chain building algorithm, i.e. it uses the Ultimate rotamer library instead of the CLIPPER Penultimate library and performs simplex minimisation of each rotamer using Equation 5.2. As in previous sections, the new BUCCANEER versions were tested using both an isolated side chain building step and a 5-cycle pipeline where BUCCANEER is iterated with global refinement using REFMAC.

5.1.3.2 Results and Discussion

Figure 5.13 shows the change in side chain RMSD for each residue type on including the rotamer score during simplex minimisation, i.e. using Equation 5.2 instead of Equation 5.1. The biggest improvement is for leucine. Figure 5.12 showed that simplex minimisation of leucine was worse than simply building the rotamers in the library, but adding the rotamer score has counteracted this. The other residues with sp^3 branching (Ile, Thr and Val) that gave a higher RMSD with simplex minimisation using density alone all show improvement as well.

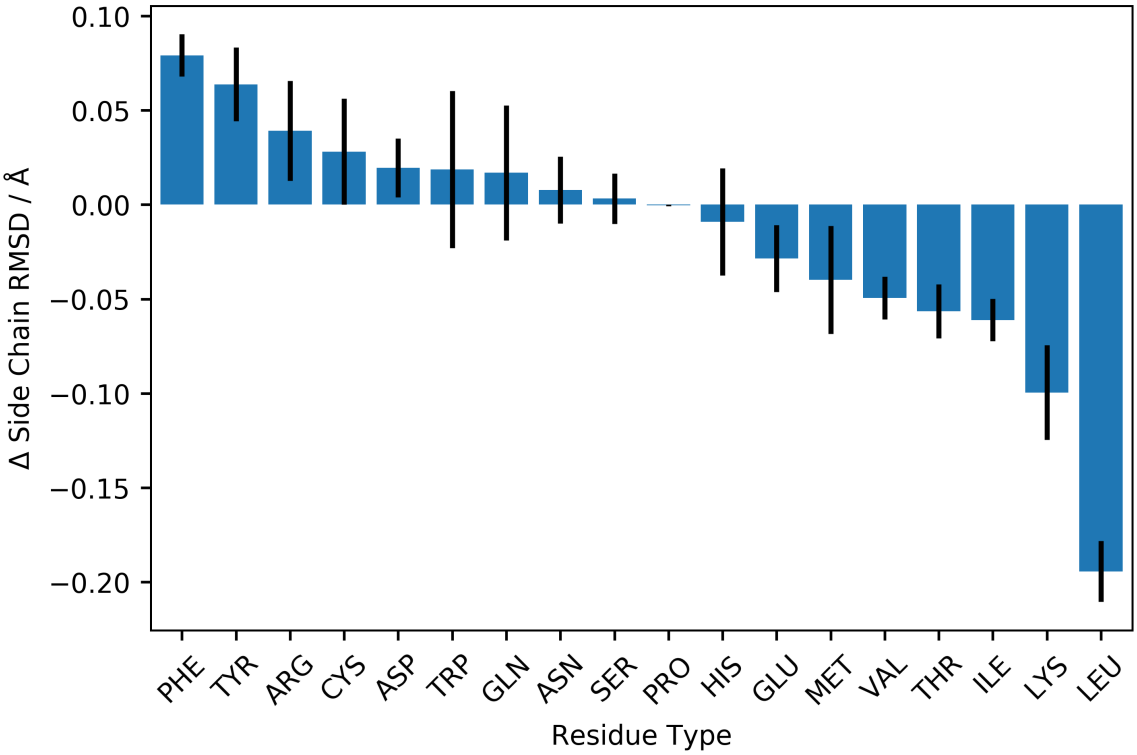

Figure 5.13: The change in side chain RMSD for each residue type on adding a rotamer score to the simplex minimisation scoring function. A single side chain building step was performed on 53 structures from the JCSG. Bars show the mean change ± one standard error.

Lysine had a similar RMSD with both the discrete $\chi 1$ rotations in the released algorithm and simplex minimisation using density score alone, but the addition of the rotamer score gives an improvement. This may reflect how often the side chain of lysine is poorly defined in the density. The same is not true for arginine where the building performs slightly worse on adding the rotamer score. Phenylalanine and

tyrosine also give a worse RMSD. This could be caused by the rotamer score being too heavily weighted, which would force side chains out of the density into more commonly observed conformations. As with bond and angle restraints used in other programs, the ideal weight is expected to be dependent on the quality of the map.

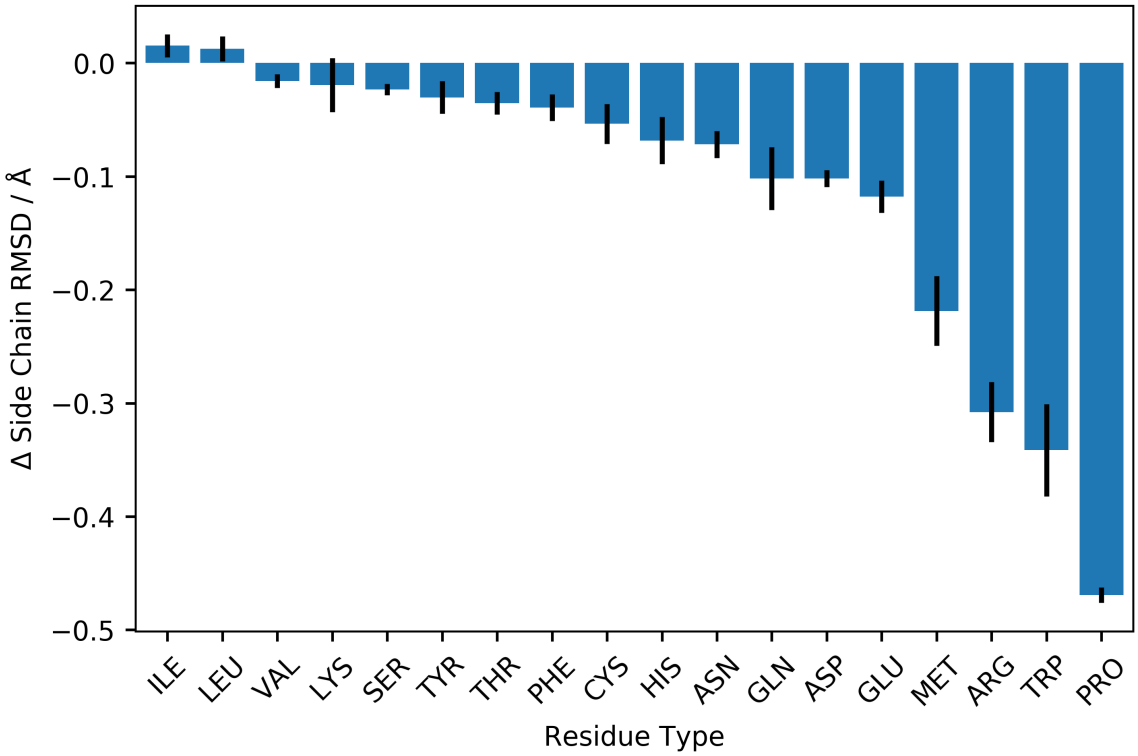

Figure 5.14: The change in side chain RMSD for each residue type on changing the released algorithm to use the Ultimate rotamer library with simplex minimisation including a rotamer score. A single side chain building step was performed on 53 structures from the JCSG. Bars show the mean change ± one standard error.

Figure 5.14 shows the difference in side chain RMSD between the released building algorithm and an algorithm with all the developments combined. Nearly all of the residue types give a better RMSD, other than Leu and Ile, which are slightly worse. Despite this, when the new algorithm is used in the 5-cycle pipeline, R-free for the models produced increases by 0.2% ± 0.4% compared to using the released side chain building algorithm. Again, this shows that the improvements to a single side chain building step performed with a correct main chain do not lead to a significant difference in pipeline performance. One reason for this could be that the global refinement performed by REFMAC is providing a similar improvement to more flexible side chain building algorithm within BUCCANEER. Refining the side

chains early in the pipeline when there is significant error in the main chain could also be detrimental to improving the model by trapping it in the wrong minimum.

5.2 Main Chain Rebuilding

As described in Section 5.1, I have done a lot of work to try and improve side chain fitting, but the correctness of the main chain is equally, if not more, important. BUCCANEER builds the protein main chain first before attempting to build the side chains, so the success of side chain building is highly dependent on the main chain conformation. If the Cα and Cβ positions are wrong, even by a small margin, then choosing the correct side chain rotamer will be difficult. To try and improve the accuracy of the main chain, a new rebuilding step was considered that modifies sections of the backbone without changing the overall fold. This involves starting at one end of the chain, rebuilding a small fragment, then moving along the chain and repeating the process until the other end of the chain is reached. A similar 'rebuild-in-place' function is used in PHENIX AUTOBUILD where a loop building algorithm iteratively builds overlapping hexapeptide fragments along the chain [5].

BUCCANEER already contains functionality for rebuilding one or two residues, which is used in the chain linking and sequence correction steps. The function *rebuild8atoms* takes the coordinates of two Cα atoms that are three residues apart, as well as two backbone atoms either side, and returns possible conformations for the eight atoms in between using an exhaustive search over allowed Ramachandran angles. A similar function called *rebuild5atoms* does the same for two Cα atoms that are only two residues apart.

Another possibility for rebuilding small sections of chain is to use the protein fragment database currently used in the loop building program, SLOOP [6]. The database contains the N, Cα and C positions for 106295 residues in 1327 fragments from 500 high resolution structures [41]. It can be queried using a fragment that contains residues with known positions, which are used to search the database, and residues with unknown positions, for which possible positions are provided by the matching database fragments.

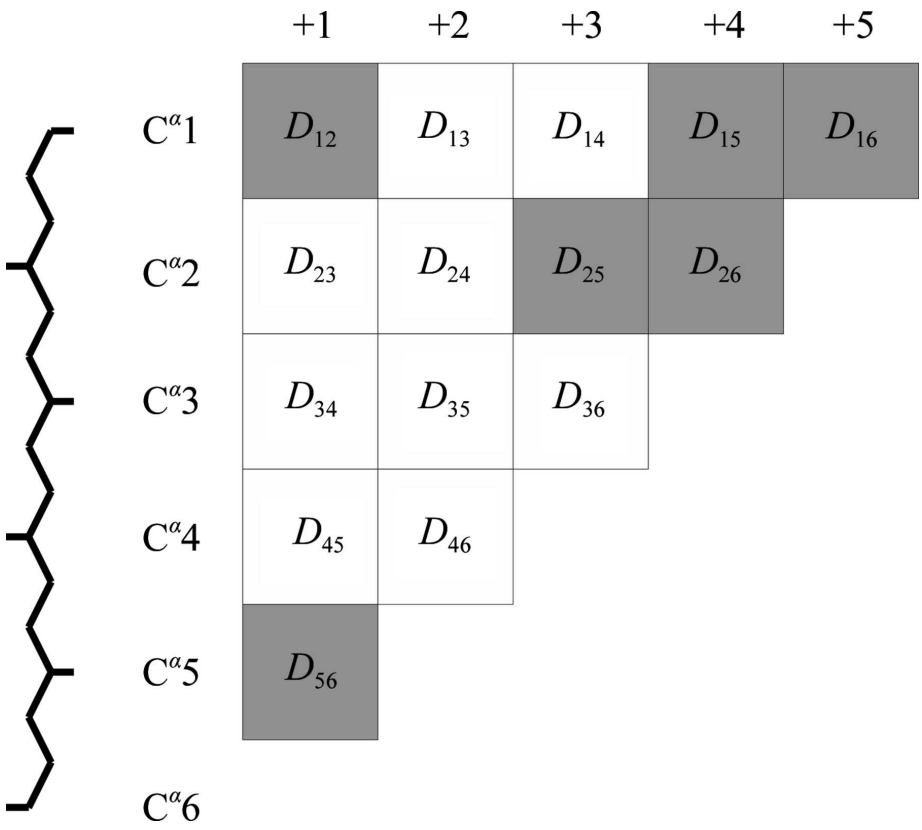

Figure 5.15: Cα distance matrix for a hexapeptide query fragment where the position of the middle two residues is unknown. D_{ij} is distance between the ith and jth Cα atoms. The shaded cells are the distances used in the initial search of the fragment database. Figure reproduced from Cowtan, 2012 [6].

Fast searching is made possible using a pre-computed Cα distance matrix. Figure 5.15 shows the triangular Cα distance matrix for a hexapeptide query fragment, where the position of the middle two residues is marked as unknown. The shaded cells are the distances between the Cα atoms of the residues with known positions that are used to find potential hits in the database.

Figure 5.16 shows part of the rectangular Cα distance matrix for the fragments in the database. Only six columns are shown but the full matrix is 20 columns wide, which could support query fragments up to 21 residues in length. Fragments are scored by comparing distances in the query matrix and the database matrix using a sum of squared differences. The dark shaded cells show the distances used to check if the hexapeptide fragment starting from C^{α}_{n+2} matches the query fragment in Figure 5.15. However, using only distances means that the search is blind to inversion, so

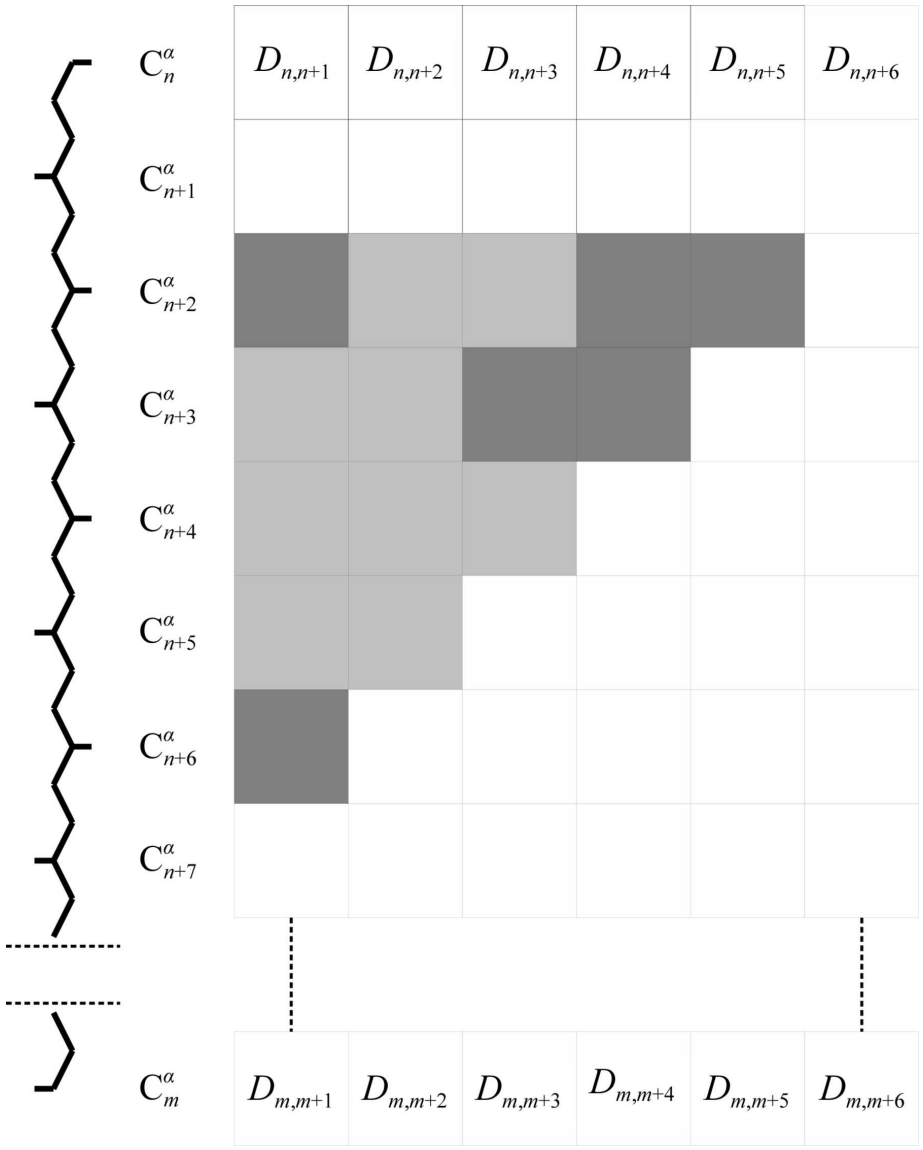

Figure 5.16: Running Cα distance matrix for the fragments in the database. D_{ij} is the distance between the ith and jth Cα atoms. The dark shaded cells are the distances used for comparison with the hexapeptide query fragment in Figure 5.15. Figure reproduced from Cowtan, 2012 [6].

a final step is done to superpose each fragment from the database onto the query using a least-squares fit of the Cα positions. The fragments are then ordered by the RMSD after superposition.

5.2.1 Method

Two new versions of BUCCANEER were produced that include a main chain rebuilding step. Using BUCCANEER 1.6.3 as a starting point, the rebuilding step was added as a penultimate step after building NCS copies and pruning clashing chains but before side chain building. Both versions proceed from the N-terminus to the C-terminus of each chain one residue at a time, building overlapping two residue fragments using the residues either side as anchor points.

The first version rebuilds the chain using the *rebuild8atoms* function to search over allowed Ramachandran angles with a torsion sampling of 12°. Each possible conformation is scored using the Cα likelihood target that is used in the finding and growing steps [37]. Importantly, the best scoring conformation is only accepted if it has a better score than the existing conformation.

In the second BUCCANEER version, the protein fragment database is searched for hexapeptide fragments where the first two residues and the last two residues match the residues at either side of the two residues to rebuild. An example of this is shown in Figure 5.17. Figure 5.17a shows six residues in a loop that make up the query fragment. Residues 3 and 4 are marked as unknown, so only residues 1, 2, 5 and 6 will be used to find matching fragments in the database. A maximum of 100 fragments are returned from the database, in ascending order of RMSD, from up to 1000 fragments chosen in the initial distance-based search. Figure 5.17b shows the top 100 fragments from the database superposed on the query fragment, again only using the Cα atoms of residues 1, 2, 5 and 6. Each fragment with an RMSD less than 0.75 Å is built into the chain by taking the coordinates of residues 3 and 4 then rebuilding the peptide bonds to residues 2 and 5. Figure 5.17c shows the fragments merged into the original conformation. As in the Ramachandran rebuilding step, merged fragments were scored using the Cα likelihood function and only accepted if they have a better score than the original.

BUCCANEER 1.6.3 and both new versions with rebuilding steps were tested on the JCSG202 experimental phasing and MR63 molecular replacement test sets (described in Chapter 2) using five iterations of BUCCANEER and REFMAC in CCP4 7.0.045. This is very similar to the defaults used in the CCP4i pipeline, except that the MR63 molecular replacement models were not passed to BUCCANEER.

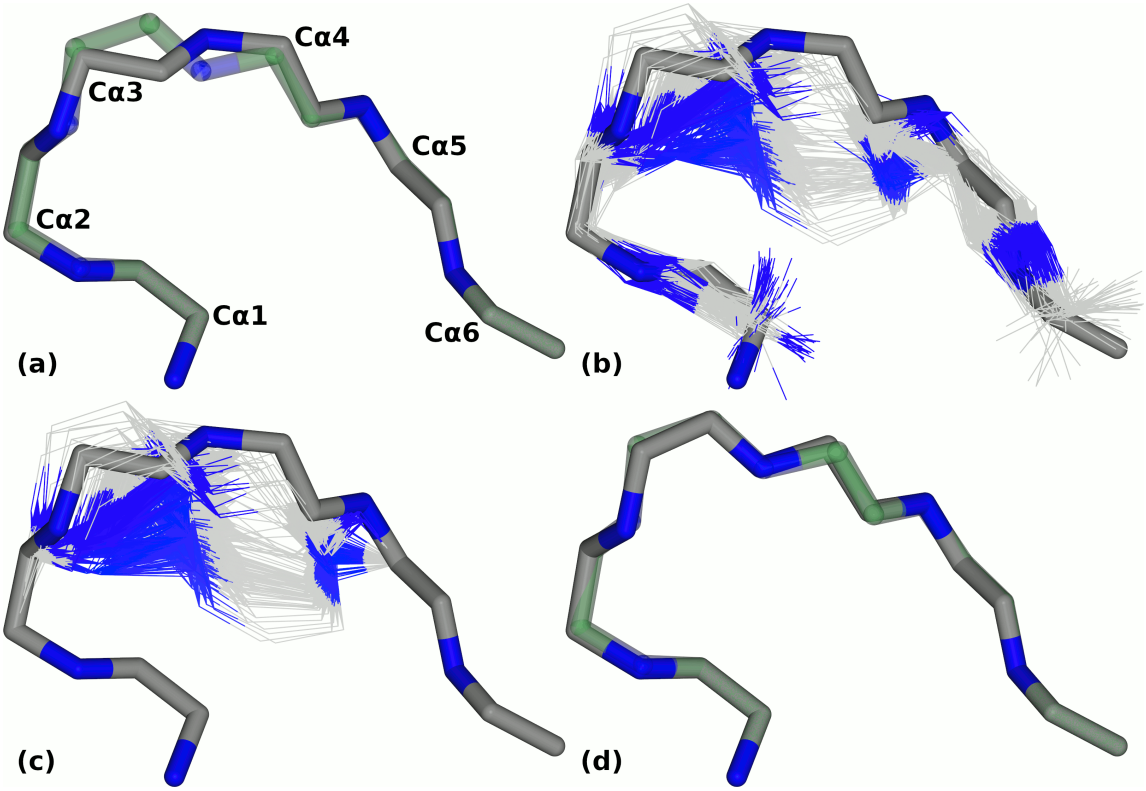

Figure 5.17: Example search of the fragment database. (a) shows a loop in opaque grey that contains an error in residues 3 and 4. The correct conformation is shown in translucent green. (b) shows 100 fragments from the database where the RMSD of Cα1, Cα2, Cα5 and Cα6 is $\leq$ 0.75 Å. (c) shows the fragments merged into the original conformation by copying residues 3 and 4 and tidying the peptide bonds to residues 2 and 5. (d) shows that the best scoring conformation is close to the correct one.

5.2.2 Results

For each of the 265 test cases, the final R-free value reported by the last REFMAC job was recorded for all three BUCCANEER versions. Table 5.5 shows the mean difference in R-free on addition of the main chain rebuilding step, along with the time it takes the rebuilding step to run. The Ramachandran rebuilding step is very fast, taking only 3.5 seconds for a single cycle, but it does not lead to any significant difference in R-free. However, rebuilding using the protein fragment database improves R-free by 0.78% on average, significant to 6 standard errors. Unfortunately, it is also much slower. A single step taking 23.6 seconds adds up to

Table 5.5: The change in R-free after 5 iterations of Buccaneer and REFMAC on the addition of a main chain rebuilding step and the time a single step takes to run. Values shown are the mean and one standard error over 265 test cases.

Rebuilding Step	R-free Change / %	Time / s
Ramachandran	-0.05 ± 0.14	3.5 ± 0.2
Fragment DB	-0.78 ± 0.13	23.6 ± 1.7

4.3 minutes for a 5 iteration pipeline with 11 internal BUCCANEER cycles. This seems to contradict previous findings where using the protein fragment database was quicker than performing an exhaustive Ramachandran search [6], but this could be due to different parameters being used, for example the torsion sampling size or the number of fragments to search for.

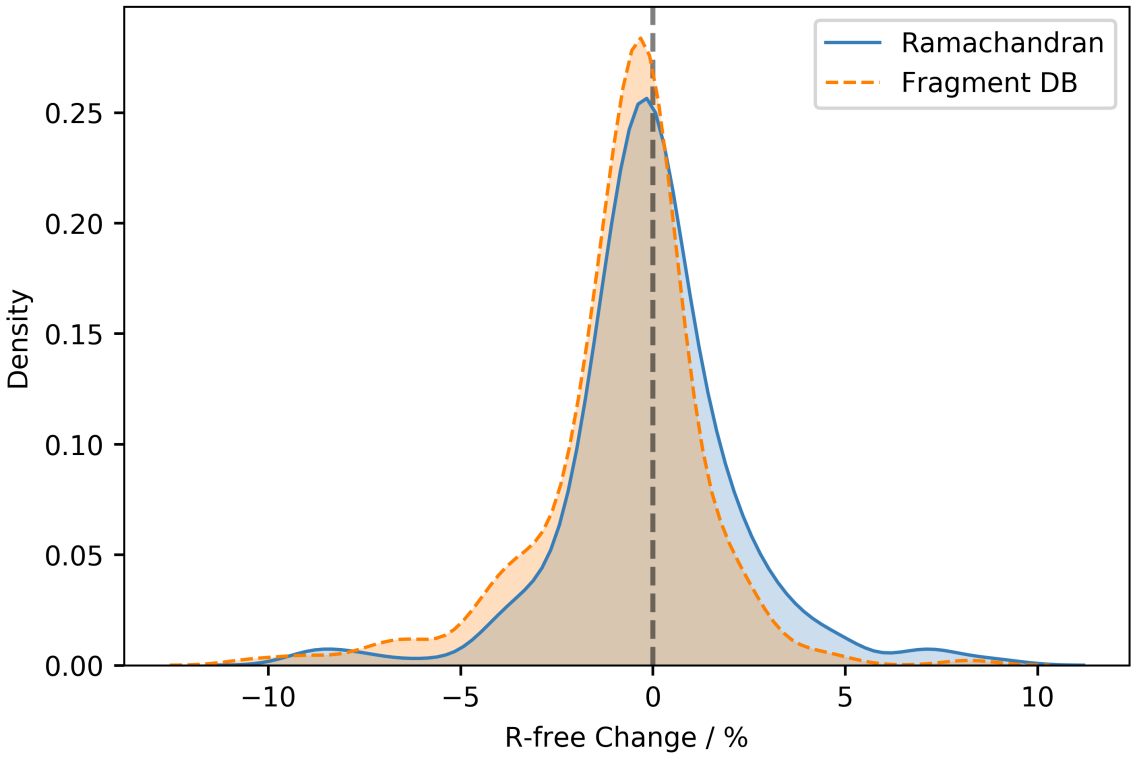

Figure 5.18: Kernel density estimation of the change in R-free after 5 iterations of Buccaneer and REFMAC on the addition of a main chain rebuilding step.

Figure 5.18 shows a kernel density estimation of the changes in R-free for each rebuilding step. To the left of the figure are cases where R-free improves. To the right of the figure are cases where R-free is made worse. The kurtosis of the distribution

depends on the stochasticity of the model building program and the difficulty of the cases in the test set, but for a random change that did not affect overall accuracy we would expect the distribution to be symmetric around a change of 0%. The changes due to the Ramachandran rebuilding step indeed look to be quite symmetrical, but the distribution for changes due to the fragment database rebuilding step is shifted to to the left of this.

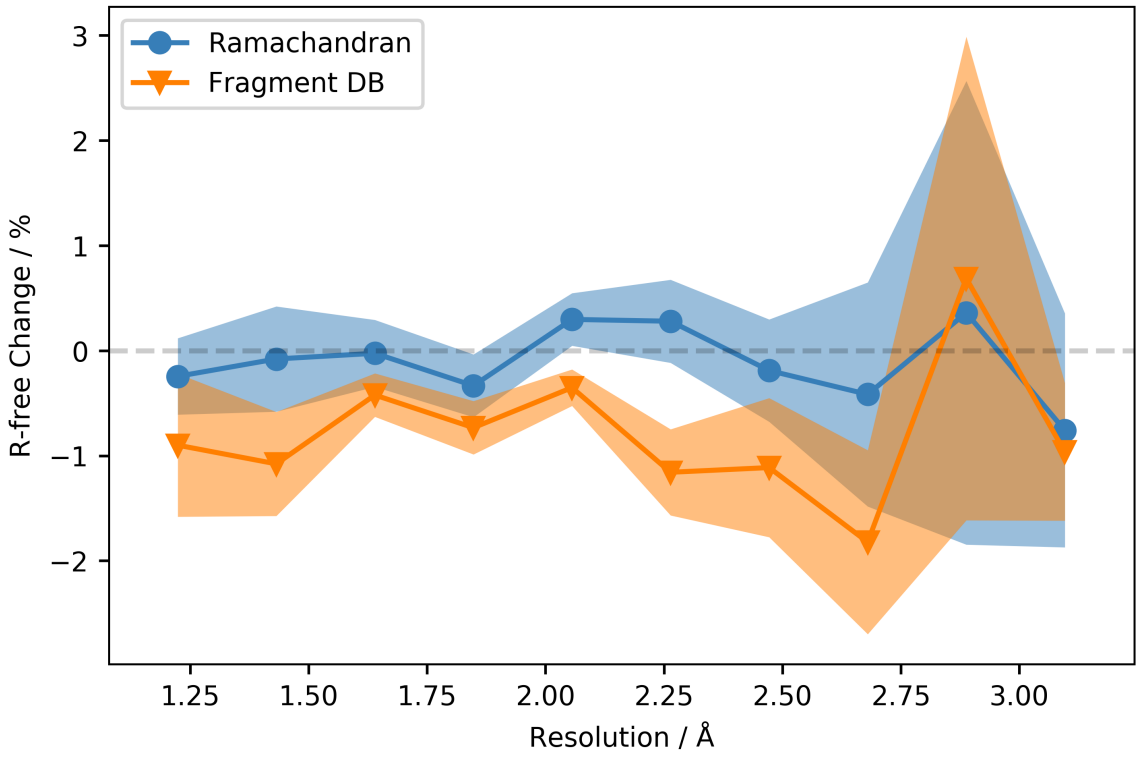

Figure 5.19: The change in R-free on the addition of the main chain rebuilding step as a function of resolution. The 265 test cases are split into 10 bins based on the resolution. The points show the mean for each bin and the shaded area shows one standard error either side of the mean.

Figures 5.19 and 5.20 show the change in R-free on addition of the main chain rebuilding step as a function of resolution and F-map correlation respectively. Resolution is the high resolution limit of the observed data. F-map correlation is the correlation coefficient between the structure factor amplitudes of the starting map used by BUCCANEER and the map from the refined deposited model, weighted by the cosine of the phase difference. In some of the test cases, the reflection data do not match the entry deposited in the PDB, so F-map correlations could only be calculated for 241 test cases where a refined deposited structure was

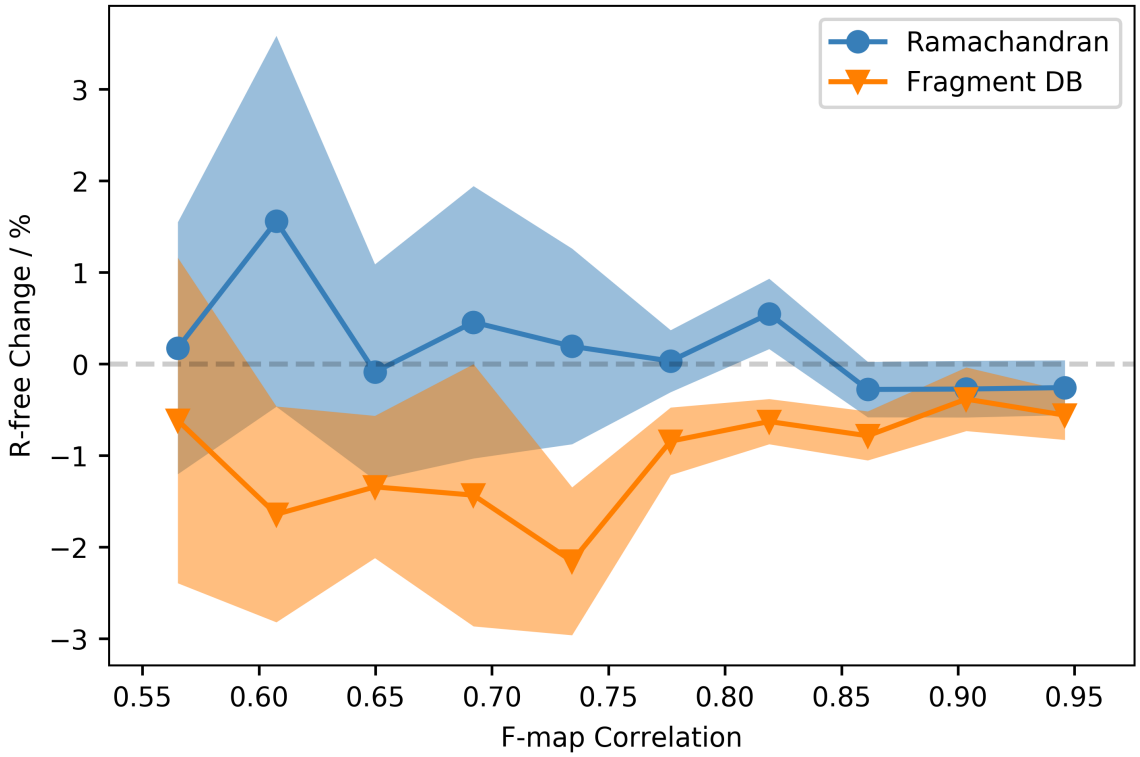

Figure 5.20: The change in R-free on the addition of the main chain rebuilding step as a function of F-map correlation. 237 test cases with F-map correlations > 0.5 are split into 10 bins based on the F-map correlation. The points show the mean for each bin and the shaded area shows one standard error either side of the mean.

available. Not many of the test cases have resolutions worse than 2.5 Å or F-map correlations worse than 0.75 so it is hard to draw conclusions for these regions. No strong correlation is apparent between the improvement of the rebuilding step and either resolution or F-map correlation.

Figure 5.21 shows how the time needed for the rebuilding step varies with the number of residues in the model. There are fewer structures in the bins to the right of the figure so the standard errors are higher, but the increase seems roughly linear as expected. For reference, the length time it takes a single cycle of BUCCANEER 1.6.3 to run is also shown.

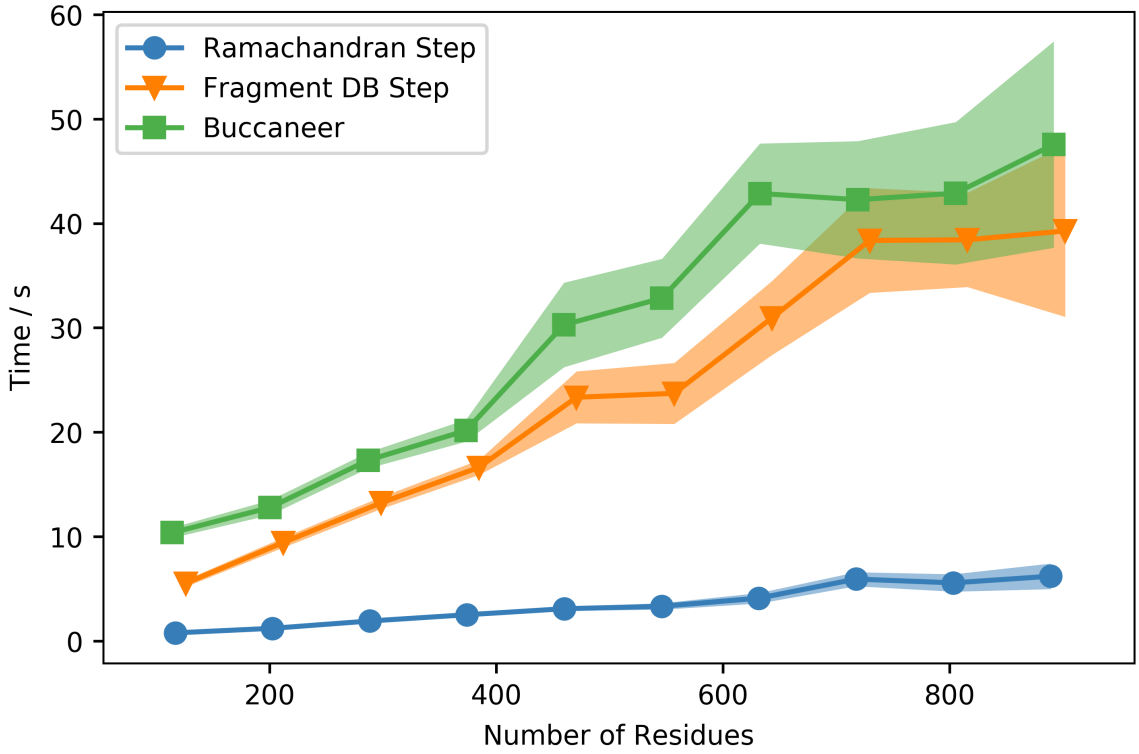

Figure 5.21: The time it takes the main chain rebuilding step to run and the time it takes a cycle of Buccaneer to run as a function of the number of residues in the model. Only 238 cases with less than 1000 residues are included. Cases are split into 10 bins based on the number of residues. The points show the mean for each bin and the shaded area shows one standard error either side of the mean.

5.2.3 Discussion

Main chain rebuilding using the *rebuild8atoms* function is very fast but unfortunately shows no sign of improving the model. Perhaps the method is too similar to how the chains are constructed in the first place, as the growing step in BUCCANEER also searches for new Cα positions by searching over allowed Ramachandran angles.

In contrast, rebuilding the main chain using the protein fragment database gives a definite improvement in model quality. This could be due to the step fixing common building problems, such as peptides that should be flipped, or it could be that the step is providing a more general regularisation of the model. Current geometric bond, angle and torsion restraints, used for regularisation during refinement in both reciprocal space and real space, are derived from observations of high resolution

structures, so is not unreasonable that information from high resolution structures could also provide a benefit on a larger scale, i.e. over a whole residue or multiple residues.

Unfortunately, the step has a drawback of increasing the run time of BUCCANEER by around 76%. It is still not very slow in absolute terms, as BUCCANEER is much faster than other model building programs [70], but some speed improvements can definitely be made. The existing database is designed to be queried by fragments of different lengths. It will contain many similar fragments, for example from α-helices and β-strands, and this redundancy can be removed for a given fragment length and RMSD threshold.

The possible speed increase can be estimated using the frequency of secondary structure elements. The RCSB PDB provides secondary structure information for protein residues, calculated from the deposited coordinates using DSSP [120]. Out of the 115 million residues annotated on 05 May 2021 [121], 79% are part of an α-helix, an isolated β-bridge, a β-strand, a 3_{10}-helix, a π-helix, a hydrogen bonded turn, or a bend. The remaining 21% are part of a loop or other irregular structure. If we consider all the possible hexapeptide fragments from these chains, 25% have the same secondary structure element for all six residues. Therefore, assuming the secondary structure is ideal, searching a non-redundant fragment database should take 25% less time. However, this estimate also assumes there is no redundancy in the conformations of other fragments so it is expected the actual speed increase will be greater than this.

Depending on the performance improvement gained by decreasing redundancy, the step could also be sped up by performing a preliminary assessment of the backbone to identify regions where rebuilding might help, for example using the Cα likelihood target or the main chain correctness score presented in Chapter 3. Residues with poor scores connected to residues with good scores would be ideal targets. This might also help to address cis-peptide bonds, which are not built by BUCCANEER but occur relatively frequently at proline residues. It is also untested whether rebuilding should be done throughout the whole pipeline or whether it is sufficient to only perform this step as the model approaches completion. The extreme case of this would be to only perform the main chain rebuilding step once at the end of the pipeline. If this provided a similar improvement to the quality of the model then the

speed of the step would be much less of an issue. More tests should be carried out using the larger molecular replacement test sets described in Chapter 2.

The speed of searching the database is one important factor. Another is whether the database actually contains the fragment being searched for. The MOLPROBITY Top500 database [41] has since been superseded by the much larger Top8000 database, which was curated using updated filtering criteria [117]. Switching to this newer source of structures should further enhance conformational coverage of the database.

Updates to the protein fragment database will have many uses outside of main chain rebuilding. An obvious use is loop building. The database is already used for this purpose in both SLOOP and COOT, but this has not as yet been incorporated into a model building pipeline.

5.3 Summary

New additions to the steps within BUCCANEER have been explored. For the final side chain building step, a new rotamer library was added and simplex minimisation of the library rotamers was tested, both using density alone and with the addition of a rotamer score. This work was performed very early in my PhD and could be improved in many ways. Firstly, the changes did not produce a similar effect for all residue types, e.g. simplex minimisation was better for building tryptophan but worse for building leucine. Instead of using a single rotamer library and building algorithm for all residue types they should be considered separately to determine what works best for each.

As testing was only performed on 53 structures, the results of the pipeline tests were not conclusive. More accurate results could now be obtained with the larger test sets that have been produced. However, despite the low significance, having a more flexible side chain building algorithm that, given a correct main chain, can build the side chain correctly more often does not seem to have much of an effect on the overall pipeline performance and indeed may even make it worse. The new building algorithm is also much slower, especially for arginine when the large number

of rotamers in the Ultimate library is combined with simplex minimisation of all four χ angles.

Therefore I propose that it is better to only build commonly occurring rotamers with very little flexibility until the final stages of the pipeline. This will run quickly and will build the majority of side chains correctly. Once the main chain has been completed then poor side chains can be identified and a slower and more accurate building algorithm can be used to correct them. This approach has since been adopted in ModelCraft, which uses the original BUCCANEER side chain building algorithm throughout the pipeline and a final side chain building step using COOT at the very end of the pipeline. The COOT step has the advantage that the main chain is refined at the same time as the side chain using a well-established real-space refinement procedure.

Other changes to the side chain building algorithm could be made. For example, if the algorithm cannot resolve a clash between two residues then the side chains of both residues are truncated to Cβ. However, it could happen that one of the residues is built correctly and the other cannot avoid clashing due to an error in the main chain. In this instance it would make sense to only truncate the less correct side chain. Another change that should be implemented is to update the proline rotamers in CLIPPER as they currently have incorrect $\chi 2$ angles.

This chapter also covered the addition of a new main chain rebuilding step as a penultimate step in BUCCANEER before side chain building. Using the protein fragment database to iteratively build overlapping two residue fragments improved the model produced by the pipeline but the step is slow. Before this step is implemented, the protein fragment database needs to be further developed by expanding the pool of source structures and removing redundancy to increase the searching speed.

Currently, rebuilding of the main chain arbitrarily starts from the N-terminus of the chain. As the termini are often flexible with poorly-defined density, it may be beneficial to rebuild outwards from more well-defined regions such as those with secondary structure. Tests should then be performed to determine whether the main chain rebuilding step needs to be included throughout the pipeline or only as the model nears completion. The fragment database is also used for loop building and

this should be revisited as is not yet included in the model building pipeline.

Chapter 6

Conclusions and Future Work

Perhaps the biggest challenge in this PhD has not been creating new methods, but evaluating them. The performance of BUCCANEER is subject to random variability. The result is exactly reproducible only when the same input is used with the same random seed. With a different random seed (or an ineffectual method change) there will be a distribution of performance changes where some examples get better and some get worse, as seen in Figure 5.18. This includes occasional extreme changes owing to the 'butterfly effect'. Because of regression toward the mean, repeating examples that performed poorly with a new method is more likely to give an improvement, even if the new method is worse. Therefore, method changes must be assessed by examining the average difference over large test sets that are not chosen based on past performance.

The aim of the project was to improve the models produced by the BUCCANEER pipeline. Most time was spent on the internal BUCCANEER steps presented in Chapter 5. Unfortunately, the changes to the side-chain building step did not lead to improvements in the pipeline, and the main-chain rebuilding step was too slow to be released without upgrades to the protein fragment database. The focus changed after the test set was expanded with more structures, including a large number of molecular replacement examples, some with relatively poor solutions. This suggested working on large-scale modifications to the pipeline, so that it could build more of the examples to near completion, instead of methods to alter small-scale features of models that are already fairly complete.

The original BUCCANEER pipeline simply performed 5 iterations of BUCCANEER and REFMAC, but it has now been enhanced with the following developments:

- The number of cycles was increased to 25, the output model is taken from the cycle with the lowest R-free, and the pipeline stops automatically if there has been no improvement for 4 cycles. These simple changes make it more likely that running with default settings will produce the best result. The challenge is predicting whether the model will improve given further cycles.

- A new machine-learned correctness score was added to prune chains at the end of each cycle, with the aim of reducing the number of incorrect chain fragments that BUCCANEER builds. It is also used to prune individual residues and side chains at the start of each cycle if high resolution data are available, which helps the pipeline to automatically correct mistakes.

- Steps to find waters and dummy atoms were added. In addition to the expected R-factor decrease, adding waters at the end of each cycle was found to improve the completeness of the protein, even with low resolution data and models with high R-factors. Producing and refining a hybrid dummy-atom model before BUCCANEER provides a similar benefit to map interpretation at high resolution.

- Classical density modification using PARROT was added. The experimental phasing examples had already undergone PARROT density modification, but this gave a big improvement to the molecular replacement test set, especially for those lower resolution structures where the density could be averaged between NCS copies.

- Shift-field refinement was added at the start of the pipeline to refine input molecular replacement models. This increases the radius of convergence and gives the most benefit when large regions of the model require a concerted movement.

- Finally, a side chain rebuilding step was added to the end of the pipeline for well-built models with high resolution data. This identifies side chains that are missing or predicted to be incorrect and rebuilds them using main chain refinement and automatic rotamer fitting in COOT.

These changes were released in a new GUI-independent pipeline called ModelCraft. Combined, they give an enormous improvement to the completeness of the models produced. Out of 1348 molecular replacement examples, the CCP4i pipeline only built 139 (10%) to above 90% completeness, while ModelCraft built 823 (61%) to above 90% completeness. For 193 experimental phasing examples, the CCP4i pipeline could already build 113 (59%) to above 90% completeness and ModelCraft increased this further to 152 (79%). ModelCraft is currently only available as a command-line program installed separately to CCP4, but it is planned to be released as part of CCP4 in the future, with interfaces in both CCP4i2 and CCP4Cloud.

There are two challenges for automated model building that it may be more helpful to separate: being able to build a rough model even with low resolution data or poor starting phases, and finalising models to reduce the amount of manual building required subsequently. If a user encounters the first problem where a model cannot be produced automatically then they can run the program again with different options, use a different program, improve the input phases, or build the model manually. The focus of the pipeline changes was to minimise the chance of this happening, which is why completeness of the protein backbone was the main metric used for comparison. Additionally, it needs to be assessed using a test set that contains many challenging starting points from both experimental phasing and molecular replacement.

However, the second problem is just as important because manual model completion is time consuming and required even when starting from a high-similarity homology model. At this stage, the protein is mostly correct and the phases are good, so more flexible model building methods can be used, including the addition of other components such as carbohydrates and ligands. Model completion should be assessed differently by starting from models where the protein backbone is already largely complete and using a metric that takes the whole model into account, such as R-free.

For future work, I believe that focusing on model completion will have the biggest impact. Specifically, to create a program that aims to build a complete model of all components starting from a well-phased map, with or without an initial model. Setting the problem up in this way means the resulting program should be useful for model building in cryo-EM as well as in the later stages of a crystallographic pipeline. Additionally, a test set can be easily produced using maps from deposited structures

either directly or after interpretation by the current model building pipeline. Local refinement of newly-added or rebuilt residues would be preferred to global refinement for speed, although global reciprocal-space refinement should still be used as a final step. For crystallography, the model completion program could be used once at the end of the pipeline, or iterated with global refinement as BUCCANEER is currently.

Of the future work plans presented in each chapter, the following are deemed to provide the most benefit:

- Creating a new residue correctness score that is absolute instead of relative to other residues in the structure. Ideally, it should be applicable to all residue types and not just protein residues. Using residues in automatically-built structures for training and test samples, a first attempt could be to combine real-space correlation coefficient and restraint deviations to predict RMSD to the deposited position. Distinguishing correct and incorrect parts of the model is important for model completion as it makes it possible to build outwards from the correct parts of the structure using more time-consuming methods.

- Developing the control system in ModelCraft to assess progress using both R-factors and completeness. This could be used to decide when to move on to model completion and to increase speed by starting with simple cycles of BUCCANEER and REFMAC and only using extra steps if they are needed. Additionally, the pipeline should be modified to build nucleic acids as well as protein. In structures with a mixture of protein and nucleic acid, the nucleic acid may form a large proportion of the ordered scattering matter so good phases will not be achievable until they are modelled.

- Refactoring BUCCANEER to be more modular so that the individual steps can be performed separately and shared with other programs. For example, a model completion program may benefit from functions within BUCCANEER such as the growing step, but a more flexible interface would be useful to call the step by itself and alter parameters such as the φ/ψ sampling and the Cα likelihood cutoff.

- Increasing the number of source structures in the protein fragment database and speeding up queries by removing redundancy for pre-determined fragment

lengths and RMSD thresholds. The improved database could then be used for main chain rebuilding, extension and loop building.

- Producing a larger experimental phasing test set. Repeating the phasing step using deposited anomalous amplitudes or intensities would be preferable, but starting from deposited experimental phases without realistic heavy atom models would still be useful.

However, more immediate progress towards model completion may be achieved by creating a program using the scripting interface in COOT. The first step would be to remove parts of the model that are highlighted as potential problems by the validation tools and map the remaining residues onto the known sequence. The protein model can then be completed by iteratively adding terminal residues and side chains through real-space refinement of a number of potential conformations and assessment of the resulting correlation coefficients and geometry deviations. Finally, building of glycans and ligands can be attempted using the automated tools that are already available in COOT. This procedure should work well for finalising a model but perhaps not for earlier building when the density is less well defined. Additionally, the input model should be as complete as possible for accurate mapping onto the sequence and to save time adding new residues.

In summary, the major development of this PhD has been to improve the BUCCANEER pipeline so that it is much more likely to build a good protein model with few changes needed, especially when starting from a molecular replacement model. The process of improving a model through trials of pruning, density modification and model building is now more automated, and therefore requires less time and expertise.

Although the number of new structures deposited in the PDB each year is increasing, the number solved by experimental phasing has stayed roughly the same and the increase is mainly due to molecular replacement structures [122]. This may suggest that the demand for automated model building after molecular replacement will also increase. However, part of the rise in molecular replacement structures is likely due to better automation when collecting data for the same protein with different ligands [123], for example recent fragment screening of the SARS-CoV-2 main protease [124]. Additionally, as molecular replacement models

improve with the number of homologues available, and even with new *ab-initio* methods such as AlphaFold [125], it will become more common that only minor rebuilding and completion is required.

References

1. Bond, P. S., Wilson, K. S. & Cowtan, K. D. Predicting protein model correctness in *Coot* using machine learning. *Acta Crystallographica Section D* **76,** 713–723 (2020).
2. Cowtan, K. *Automated phase improvement and model building with Parrot and Buccaneer* Presentation. `https://www.ccp4.ac.uk/schools/APS-2011/tutorials/buccaneer/Pannu.parrot_buccaneer_aps_11.ppt`. Accessed: 2021-02-19.
3. Cowtan, K. *Model Building* Presentation. Icknield Workshop: Model Building and Refinement for High Resolution EM Maps. 2017.
4. Perrakis, A., Harkiolaki, M., Wilson, K. S. & Lamzin, V. S. *ARP/wARP* and molecular replacement. *Acta Crystallographica Section D* **57,** 1445–1450 (2001).
5. Terwilliger, T. C. *et al.* Iterative model building, structure refinement and density modification with the *PHENIX AutoBuild* wizard. *Acta Crystallographica Section D* **64,** 61–69 (2008).
6. Cowtan, K. Completion of autobuilt protein models using a database of protein fragments. *Acta Crystallographica Section D* **68,** 328–335 (2012).
7. Bragg, W. H. & Bragg, W. L. The reflection of X-rays by crystals. *Proceedings of the Royal Society of London. Series A* **88,** 428–438 (1913).
8. Rodríguez, D. D. *et al.* Crystallographic ab initio protein structure solution below atomic resolution. *Nature Methods* **6,** 651–653 (2009).
9. Jenkins, H. T. *Fragon*: rapid high-resolution structure determination from ideal protein fragments. *Acta Crystallographica Section D* **74,** 205–214 (2018).
10. McCoy, A. J. *et al.* Ab initio solution of macromolecular crystal structures without direct methods. *Proceedings of the National Academy of Sciences* **114,** 3637–3641 (2017).

11. Kovalevskiy, O., Nicholls, R. A., Long, F., Carlon, A. & Murshudov, G. N. Overview of refinement procedures within *REFMAC*5: utilizing data from different sources. *Acta Crystallographica Section D* **74,** 215–227 (2018).

12. Afonine, P. V. *et al.* Towards automated crystallographic structure refinement with *phenix.refine*. *Acta Crystallographica Section D* **68,** 352–367 (2012).

13. Cowtan, K. & Agirre, J. Macromolecular refinement by model morphing using non-atomic parameterizations. *Acta Crystallographica Section D* **74,** 125–131 (2018).

14. Cowtan, K., Metcalfe, S. & Bond, P. Shift-field refinement of macromolecular atomic models. *Acta Crystallographica Section D* **76,** 1192–1200 (2020).

15. Emsley, P., Lohkamp, B., Scott, W. G. & Cowtan, K. Features and development of *Coot*. *Acta Crystallographica Section D* **66,** 486–501 (2010).

16. Chen, V. B. *et al. MolProbity*: all-atom structure validation for macromolecular crystallography. *Acta Crystallographica Section D* **66,** 12–21 (2010).

17. Berman, H. M. *et al.* The Protein Data Bank. *Nucleic Acids Research* **28,** 235–242 (2000).

18. Kendrew, J. C. *et al.* A three-dimensional model of the myoglobin molecule obtained by x-ray analysis. *Nature* **181,** 662–666 (1958).

19. Perutz, M. F. *et al.* Structure of hæmoglobin: a three-dimensional Fourier synthesis at 5.5-Å. resolution, obtained by X-ray analysis. *Nature* **185,** 416–422 (1960).

20. Kendrew, J. C. *et al.* Structure of myoglobin: A three-dimensional Fourier synthesis at 2 Å. resolution. *Nature* **185,** 422–427 (1960).

21. Richards, F. M. The matching of physical models to three-dimensional electron-density maps: A simple optical device. *Journal of Molecular Biology* **37,** 225–230 (1968).

22. Foley, J. D. & Wright, W. V. *An interactive molecular graphics system with a satellite terminal closely coupled to its host* in *Proceedings of the 1975 annual conference* (Association for Computing Machinery, 1975), 88–89. ISBN: 9781450374811.

23. Jones, T. A. A graphics model building and refinement system for macromolecules. *Journal of Applied Crystallography* **11,** 268–272 (1978).

24. Diamond, R. in *Biomolecular Structure, Conformation, Function, and Evolution* (ed Srinivasan, R.) 567–588 (Pergamon, 1981). ISBN: 978-1-4832-8364-7.

25. Greer, J. Three-dimensional pattern recognition: An approach to automated interpretation of electron density maps of proteins. *Journal of Molecular Biology* **82,** 279–301 (1974).

26. Jones, T. A. & Thirup, S. Using known substructures in protein model building and crystallography. *The EMBO Journal* **5,** 819–822 (1986).

27. Jones, T. A., Zou, J.-Y., Cowan, S. W. & Kjeldgaard, M. Improved methods for building protein models in electron density maps and the location of errors in these models. *Acta Crystallographica Section A* **47,** 110–119 (1991).

28. Roussel, A. & Cambillau, C. in *Silicon Graphics Geometry Partner Directory* 77–78 (Silicon Graphics, 1981).

29. McRee, D. E. XtalView/Xfit—A Versatile Program for Manipulating Atomic Coordinates and Electron Density. *Journal of Structural Biology* **125,** 156–165 (1999).

30. Oldfield, T. J. *A Semi-Automated Map Fitting Procedure* in *Proceedings from the 1996 meeting of the International Union Of Crystallography Macromolecular Computing School* (1996).

31. Oldfield, T. Pattern-recognition methods to identify secondary structure within X-ray crystallographic electron-density maps. *Acta Crystallographica Section D* **58,** 487–493 (2002).

32. Turk, D. *MAIN* software for density averaging, model building, structure refinement and validation. *Acta Crystallographica Section D* **69,** 1342–1357 (2013).

33. Emsley, P. & Cowtan, K. *Coot*: model-building tools for molecular graphics. *Acta Crystallographica Section D* **60,** 2126–2132 (2004).

34. Croll, T. I. *ISOLDE*: a physically realistic environment for model building into low-resolution electron-density maps. *Acta Crystallographica Section D* **74,** 519–530 (2018).

35. Cowtan, K. Modified Phased Translation Functions and their Application to Molecular-Fragment Location. *Acta Crystallographica Section D* **54,** 750–756 (1998).

36. Cowtan, K. Fast Fourier feature recognition. *Acta Crystallographica Section D* **57,** 1435–1444 (2001).

37. Cowtan, K. The *Buccaneer* software for automated model building. 1. Tracing protein chains. *Acta Crystallographica Section D* **62,** 1002–1011 (2006).

38. Ioerger, T. R. & Sacchettini, J. C. Automatic modeling of protein backbones in electron-density maps *via* prediction of C^{α} coordinates. *Acta Crystallographica Section D* **58,** 2043–2054 (2002).

39. Cowtan, K. Fitting molecular fragments into electron density. *Acta Crystallographica Section D* **64,** 83–89 (2008).

40. Lovell, S. C., Word, J. M., Richardson, J. S. & Richardson, D. C. The penultimate rotamer library. *Proteins: Structure, Function, and Bioinformatics* **40,** 389–408 (2000).

41. Lovell, S. C. *et al.* Structure validation by Cα geometry: ϕ, ψ and Cβ deviation. *Proteins: Structure, Function, and Bioinformatics* **50,** 437–450 (2003).

42. Elsliger, M.-A. *et al.* The JCSG high-throughput structural biology pipeline. *Acta Crystallographica Section F* **66,** 1137–1142 (2010).

43. Murshudov, G. N. *et al. REFMAC*5 for the refinement of macromolecular crystal structures. *Acta Crystallographica Section D* **67,** 355–367 (2011).

44. Lamzin, V. S. & Wilson, K. S. Automated refinement of protein models. *Acta Crystallographica Section D* **49,** 129–147 (1993).

45. Perrakis, A., Sixma, T. K., Wilson, K. S. & Lamzin, V. S. *wARP*: Improvement and Extension of Crystallographic Phases by Weighted Averaging of Multiple-Refined Dummy Atomic Models. *Acta Crystallographica Section D* **53,** 448–455 (1997).

46. Perrakis, A., Morris, R. & Lamzin, V. S. Automated protein model building combined with iterative structure refinement. *Nature Structural Biology* **6,** 458–463 (1999).

47. Morris, R. J., Perrakis, A. & Lamzin, V. S. *ARP/wARP*'s model-building algorithms. I. The main chain. *Acta Crystallographica Section D* **58,** 968–975 (2002).

48. Cohen, S. X. *et al.* Towards complete validated models in the next generation of *ARP/wARP*. *Acta Crystallographica Section D* **60,** 2222–2229 (2004).

49. Joosten, K. *et al.* A knowledge-driven approach for crystallographic protein model completion. *Acta Crystallographica Section D* **64,** 416–424 (2008).

50. Cohen, S. X. *et al. ARP/wARP* and molecular replacement: the next generation. *Acta Crystallographica Section D* **64,** 49–60 (2008).

51. Wiegels, T. & Lamzin, V. S. Use of noncrystallographic symmetry for automated model building at medium to low resolution. *Acta Crystallographica Section D* **68,** 446–453 (2012).

52. Morris, R. J. *et al.* Breaking good resolutions with ARP/wARP. *Journal of Synchrotron Radiation* **11,** 56–59 (2004).

53. Langer, G., Cohen, S. X., Lamzin, V. S. & Perrakis, A. Automated macromolecular model building for X-ray crystallography using ARP/wARP version 7. *Nature Protocols* **3,** 1171–1179 (2008).

54. Chojnowski, G., Heuser, P., Pereira, J. & Lamzin, V. Building atomic models into electron microscopy maps with *ARP/wARP* version 8.0. *Acta Crystallographica Section A* **74,** e151 (2018).

55. Chojnowski, G., Pereira, J. & Lamzin, V. S. Sequence assignment for low-resolution modelling of protein crystal structures. *Acta Crystallographica Section D* **75,** 753–763 (2019).

56. Pereira, J. & Lamzin, V. S. A distance geometry-based description and validation of protein main-chain conformation. *IUCrJ* **4,** 657–670 (2017).

57. Chojnowski, G. *et al.* The use of local structural similarity of distant homologues for crystallographic model building from a molecular-replacement solution. *Acta Crystallographica Section D* **76,** 248–260 (2020).

58. Terwilliger, T. C. Maximum-likelihood density modification. *Acta Crystallographica Section D* **56,** 965–972 (2000).

59. Terwilliger, T. C. Maximum-likelihood density modification using pattern recognition of structural motifs. *Acta Crystallographica Section D* **57,** 1755–1762 (2001).

60. Terwilliger, T. C. Statistical density modification with non-crystallographic symmetry. *Acta Crystallographica Section D* **58,** 2082–2086 (2002).

61. Terwilliger, T. C. Automated main-chain model building by template matching and iterative fragment extension. *Acta Crystallographica Section D* **59,** 38–44 (2003).

62. Terwilliger, T. C. Automated side-chain model building and sequence assignment by template matching. *Acta Crystallographica Section D* **59,** 45–49 (2003).

63. Terwilliger, T. C. Improving macromolecular atomic models at moderate resolution by automated iterative model building, statistical density modification and refinement. *Acta Crystallographica Section D* **59,** 1174–1182 (2003).

64. Sheldrick, G. M. Experimental phasing with *SHELXC/D/E*: combining chain tracing with density modification. *Acta Crystallographica Section D* **66,** 479–485 (2010).

65. Kleywegt, G. J. & Jones, T. A. Template Convolution to Enhance or Detect Structural Features in Macromolecular Electron-Density Maps. *Acta Crystallographica Section D* **53,** 179–185 (1997).

66. Thorn, A. & Sheldrick, G. M. Extending molecular-replacement solutions with *SHELXE*. *Acta Crystallographica Section D* **69,** 2251–2256 (2013).

67. Bibby, J., Keegan, R. M., Mayans, O., Winn, M. D. & Rigden, D. J. *AMPLE*: a cluster-and-truncate approach to solve the crystal structures of small proteins using rapidly computed *ab initio* models. *Acta Crystallographica Section D* **68,** 1622–1631 (2012).

68. Usón, I. & Sheldrick, G. M. An introduction to experimental phasing of macromolecules illustrated by *SHELX*; new autotracing features. *Acta Crystallographica Section D* **74,** 106–116 (2018).

69. Van den Bedem, H., Wolf, G., Xu, Q. & Deacon, A. M. Distributed structure determination at the JCSG. *Acta Crystallographica Section D* **67,** 368–375 (2011).

70. Alharbi, E., Bond, P. S., Calinescu, R. & Cowtan, K. Comparison of automated crystallographic model-building pipelines. *Acta Crystallographica Section D* **75,** 1119–1128 (2019).

71. Cowtan, K. Recent developments in classical density modification. *Acta Crystallographica Section D* **66,** 470–478 (2010).

72. Stein, N. *CHAINSAW*: a program for mutating pdb files used as templates in molecular replacement. *Journal of Applied Crystallography* **41,** 641–643 (2008).

73. Vagin, A. & Teplyakov, A. *MOLREP*: an Automated Program for Molecular Replacement. *Journal of Applied Crystallography* **30,** 1022–1025 (1997).

74. Burla, M. C., Carrozzini, B., Cascarano, G. L., Giacovazzo, C. & Polidori, G. How far are we from automatic crystal structure solution via molecular-replacement techniques? *Acta Crystallographica Section D* **76,** 9–18 (2020).

75. Caliandro, R. *et al.* Molecular replacement: the probabilistic approach of the program *REMO09* and its applications. *Acta Crystallographica Section A* **65,** 512–527 (2009).

76. Burla, M. C., Cascarano, G. L., Giacovazzo, C. & Polidori, G. Synergy among phase-refinement techniques in macromolecular crystallography. *Acta Crystallographica Section D* **73,** 877–888 (2017).

77. Burla, M. C., Carrozzini, B., Cascarano, G. L., Polidori, G. & Giacovazzo, C. *CAB*: a cyclic automatic model-building procedure. *Acta Crystallographica Section D* **74,** 1096–1104 (2018).

78. Finn, R. D. *et al.* HMMER web server: 2015 update. *Nucleic Acids Research* **43,** W30–W38 (2015).

79. Bunkóczi, G. & Read, R. J. Improvement of molecular-replacement models with *Sculptor*. *Acta Crystallographica Section D* **67,** 303–312 (2011).

80. Krissinel, E. Enhanced fold recognition using efficient short fragment clustering. *Journal of Molecular Biochemistry* **1,** 76–85 (2012).

81. Krissinel, E. & Uski, V. Desktop and Web-based Gesamt Software for Fast and Accurate Structural Queries in the PDB. *Journal of Computer Science Applications and Information Technology* **2,** 1–7 (2017).

82. McCoy, A. J. *et al. Phaser* crystallographic software. *Journal of Applied Crystallography* **40,** 658–674 (2007).

83. Foadi, J. *et al.* A flexible and efficient procedure for the solution and phase refinement of protein structures. *Acta Crystallographica Section D* **56,** 1137–1147 (2000).

84. Winn, M. D. *et al.* Overview of the *CCP*4 suite and current developments. *Acta Crystallographica Section D* **67,** 235–242 (2011).

85. Yang, H. *et al. DCC*: a Swiss army knife for structure factor analysis and validation. *Journal of Applied Crystallography* **49,** 1081–1084 (2016).

86. Kleywegt, G. J. *et al.* The Uppsala Electron-Density Server. *Acta Crystallographica Section D* **60,** 2240–2249 (2004).

87. DW. *Using BLASTClust to Make Non-redundant Sequence Sets* NCBI News. `https://www.ncbi.nlm.nih.gov/Web/Newsltr/Spring04/blastlab.html`. Accessed: 2021-02-19.

88. French, S. & Wilson, K. On the treatment of negative intensity observations. *Acta Crystallographica Section A* **34,** 517–525 (1978).

89. Higgins, D. G. & Sharp, P. M. CLUSTAL: a package for performing multiple sequence alignment on a microcomputer. *Gene* **73,** 237–244 (1988).

90. Read, R. J. & McCoy, A. J. A log-likelihood-gain intensity target for crystallographic phasing that accounts for experimental error. *Acta Crystallographica Section D* **72,** 375–387 (2016).

91. Potterton, L. *et al. CCP*4*i*2: the new graphical user interface to the *CCP*4 program suite. *Acta Crystallographica Section D* **74,** 68–84 (2018).

92. Skubák, P. & Pannu, N. S. Automatic protein structure solution from weak X-ray data. *Nature Communications* **4,** 2777 (2013).

93. Vollmar, M. *et al.* The predictive power of data-processing statistics. *IUCrJ* **7,** 342–354 (2020).

94. Potterton, E., Briggs, P., Turkenburg, M. & Dodson, E. A graphical user interface to the *CCP*4 program suite. *Acta Crystallographica Section D* **59,** 1131–1137 (2003).

95. Ness, S. R., de Graaff, R. A., Abrahams, J. P. & Pannu, N. S. Crank: New methods for automated macromolecular crystal structure solution. *Structure* **12,** 1753–1761 (2004).

96. Pannu, N. S. *et al.* Recent advances in the *CRANK* software suite for experimental phasing. *Acta Crystallographica Section D* **67,** 331–337 (2011).

97. Krissinel, E., Uski, V., Lebedev, A., Winn, M. & Ballard, C. Distributed computing for macromolecular crystallography. *Acta Crystallographica Section D* **74,** 143–151 (2018).

98. Emsley, P. *The Coot User Manual* `https://www2.mrc-lmb.cam.ac.uk/personal/pemsley/coot/web/docs/coot.html`. Accessed: 2021-02-19.

99. Iglewicz, B. & Hoaglin, D. C. in *How to Detect and Handle Outliers* (ed Mykytka, E. F.) 11–13 (ASQC Quality Press, 1993). ISBN: 0-87389-247-X.

100. Nelder, J. A. & Mead, R. A Simplex Method for Function Minimization. *The Computer Journal* **7,** 308–313 (1965).

101. Pedregosa, F. *et al.* Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* **12,** 2825–2830 (2011).

102. Kingma, D. P. & Ba, J. L. *Adam: A Method for Stochastic Optimization* 2017. arXiv: `1412.6980 [cs.LG]`.

103. Fala, A. M. *et al.* Unsaturated fatty acids as high-affinity ligands of the C-terminal Per-ARNT-Sim domain from the Hypoxia-inducible factor 3α. *Scientific Reports* **5,** 2045–2322 (2015).

104. Nicholls, R. A., Long, F. & Murshudov, G. N. Low-resolution refinement tools in *REFMAC*5. *Acta Crystallographica Section D* **68,** 404–417 (2012).

105. Joint Center for Structural Genomics (JCSG). *Crystal structure of a putative gmp synthase subunit a protein (ta0944m) from thermoplasma acidophilum at 2.45 A resolution* PDB ID: 1CI0. 2005.

106. Tamura, K. *et al.* Molecular Mechanism by which Prominent Human Gut Bacteroidetes Utilize Mixed-Linkage Beta-Glucans, Major Health-Promoting Cereal Polysaccharides. *Cell Reports* **21,** 417–430 (2017).

107. Bleicher, L. *et al.* Molecular Basis of the Thermostability and Thermophilicity of Laminarinases: X-ray Structure of the Hyperthermostable Laminarinase from Rhodothermus marinus and Molecular Dynamics Simulations. *The Journal of Physical Chemistry B* **115,** 7940–7949 (2011).

108. CCP4. *i2run* CCP4i2 documentation. `https://ccp4i2.gitlab.io/rstdocs/i2run/i2run.html`. Accessed: 2021-02-19.

109. Burnley, T., Palmer, C. M. & Winn, M. Recent developments in the *CCP-EM* software suite. *Acta Crystallographica Section D* **73,** 469–477 (2017).

110. Cowtan, K. Error estimation and bias correction in phase-improvement calculations. *Acta Crystallographica Section D* **55,** 1555–1567 (1999).

111. Abrahams, J. P. & Leslie, A. G. W. Methods used in the structure determination of bovine mitochondrial F_1 ATPase. *Acta Crystallographica Section D* **52,** 30–42 (1996).

112. Brünger, A. T. *et al. Crystallography & NMR System*: A New Software Suite for Macromolecular Structure Determination. *Acta Crystallographica Section D* **54,** 905–921 (1998).

113. Bond, P. *ModelCraft* `https://paulsbond.co.uk/modelcraft`. Accessed: 2021-02-19.

114. Cowtan, K. Automated nucleic acid chain tracing in real time. *IUCrJ* **1,** 387–392 (2014).

115. Mihaela Atanasova, K. C. & Agirre, J. *Sails - Software for the Automated Identification of Linked Sugars* `https://github.com/glycojones/sails`. Accessed: 2021-02-19.

116. Cowtan, K. *Clipper Libraries* `http://www.ysbl.york.ac.uk/~cowtan/clipper/clipper.html`. Accessed: 2021-02-19.

117. Hintze, B. J., Lewis, S. M., Richardson, J. S. & Richardson, D. C. Molprobity's ultimate rotamer-library distributions for model validation. *Proteins: Structure, Function, and Bioinformatics* **84,** 1177–1189 (2016).

118. Laskowski, R. A., MacArthur, M. W., Moss, D. S. & Thornton, J. M. *PROCHECK*: a program to check the stereochemical quality of protein structures. *Journal of Applied Crystallography* **26,** 283–291 (1993).

119. Dunbrack, R. L. & Karplus, M. Backbone-dependent rotamer library for proteins. Application to side-chain prediction. *Journal of Molecular Biology* **230,** 543–574 (1993).

120. Kabsch, W. & Sander, C. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers* **22,** 2577–2637 (1983).

121. RCSB. *ss_dis.txt.gz* `https://cdn.rcsb.org/etl/kabschSander/ss_dis.txt.gz`. Accessed: 2020-05-02.

122. McCoy, A. *The value of in silico predictions for molecular replacement and crystallography* Presentation. CCP4 Study Weekend. 2021.

123. Patel, D., Bauman, J. D. & Arnold, E. Advantages of crystallographic fragment screening: Functional and mechanistic insights from a powerful platform for efficient drug discovery. *Progress in Biophysics and Molecular Biology* **116,** 92–100 (2014).

124. Douangamath, A. *et al.* Crystallographic and electrophilic fragment screening of the SARS-CoV-2 main protease. *Nature Communications* **11,** 5047 (2020).

125. Senior, A. W. *et al.* Improved protein structure prediction using potentials from deep learning. *Nature* **577,** 706–710 (2020).