DEPARTMENT OF COMPUTER SCIENCE

Molecular Forcefield Optimisation via Machine Learning

James Price

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Master of Engineering in the Faculty of Engineering

Declaration

This dissertation is submitted to the University of Bristol in accordance with the requirements of the degree of Master of Engineering in the Faculty of Engineering. It has not been submitted for any other degree or diploma of any examining body. Except where specifically acknowledged, it is all the work of the Author.

James Price, May 2012

Executive Summary

This thesis investigates the use of machine learning as a tool to optimise molecular forcefields.

In molecular mechanics, forcefields can be used to approximate the potential energy of a system of particles. Derivation of these parameters is performed either by computationally expensive quantum mechanical methods, or by a lengthy process involving manual experimentation and tuning, after which it is often still difficult to achieve an accurate approximation. Forcefields have a vast number of practical applications, including those in the areas of drug discovery and design, understanding the processes underlying cell-signalling and in protein structure prediction and design.

This work focuses on the free energy forcefield employed by the Bristol University Docking Engine (BUDE) to predict the binding affinity of protein-ligand complexes. This forcefield has already been hand-tuned to achieve reasonable results, but still requires significant improvement in order to reach a competitive level of accuracy.

The main contributions of this thesis are:

- An automated forcefield optimisation framework has been developed
- Multiple optimisation algorithms and forcefield evaluation metrics have been implemented and compared
- BUDE's free energy forcefield has been successfully optimised to improve the structure prediction accuracy from 64% to 79%
- The use of the optimisation framework for generating new forcefields from scratch has been investigated

Contents

1 Context **4**
- 1.1 Molecular Modelling with Forcefields . . . 4
  - 1.1.1 Overview . . . 4
  - 1.1.2 Molecular Mechanics . . . 4
  - 1.1.3 Forcefield Development . . . 5
  - 1.1.4 Applications . . . 6
- 1.2 Ligand-Protein Docking with BUDE . . . 6
  - 1.2.1 Molecular Docking Overview . . . 6
  - 1.2.2 BUDE: Empirical Free Energy Forcefield Approach . . . 7
  - 1.2.3 Scope for Improvement . . . 8
- 1.3 Project Aims . . . 8

2 Technical Background **10**
- 2.1 BUDE . . . 10
  - 2.1.1 Docking Algorithm . . . 10
  - 2.1.2 Acceleration . . . 10
  - 2.1.3 Forcefield Design . . . 10
  - 2.1.4 Forcefield Optimisation . . . 11
- 2.2 Related Work . . . 13
  - 2.2.1 GROW . . . 13
  - 2.2.2 Parmscan . . . 13
  - 2.2.3 Simplex Approach . . . 14
  - 2.2.4 QM/MM Force-Matching . . . 14
- 2.3 Optimisation Algorithms . . . 14
  - 2.3.1 Overview . . . 14
  - 2.3.2 Hill Climbers . . . 16
  - 2.3.3 Genetic Algorithms . . . 17

3 Requirements **19**
- 3.1 Software Requirements . . . 19
- 3.2 Hardware Requirements . . . 19

4 Execution **20**
- 4.1 Design & Implementation . . . 20
  - 4.1.1 Overview . . . 20
  - 4.1.2 Forcefield Evaluation . . . 20
  - 4.1.3 OpenCL & GPUs . . . 21
  - 4.1.4 Optimisation Framework & Configuration . . . 22
  - 4.1.5 Forcefield Manipulation . . . 23
  - 4.1.6 Hill Climber . . . 24
  - 4.1.7 Genetic Algorithm . . . 24
- 4.2 Experiments & Results . . . 25
  - 4.2.1 Overview . . . 25
  - 4.2.2 Considerations . . . 26

4.2.3 Forcefield Constraints 27
4.2.4 Best Results 28
4.2.5 Analysis of Learning Algorithms 31
4.2.6 Generation from Scratch 32

5 Conclusion **34**
5.1 Achievements & Critique 34
5.1.1 Automated Forcefield Optimisation Tool 34
5.1.2 Improved Free Energy Forcefield 34
5.1.3 Forcefield Generation 34
5.2 Future Work 35
5.2.1 Further Forcefield Improvements 35
5.2.2 Further Development of Optimisation Algorithms 35
5.2.3 Energy Prediction 36
5.2.4 Modifications to Parameterisation 36
5.2.5 Other Forcefields 36

A Example Configuration File **38**

References **40**

1 Context

1.1 Molecular Modelling with Forcefields

1.1.1 Overview

Molecular modelling refers to the set of methods used to model systems of particles and their behaviour. For all but the simplest systems computers are required to carry out the often intensive computations defined by the model. There are generally two main approaches to molecular modelling. The quantum chemistry approach acts at a sub-atomic level, directly modelling the properties of individual electrons. The molecular mechanics approach works at a higher level, whereby the atoms themselves are considered to be the smallest particles in the system.

Although it often yields a high degree of accuracy, modelling a system fully with quantum chemistry is inherently a very computationally intensive task. Molecular mechanics on the other hand provides a coarser approximation for the system which, at the cost of reduced accuracy, is typically much faster to compute. This results in a trade-off between accuracy and speed for the two approaches, for which the desired level of precision and amount of available computing resources must be taken into account when deciding which method to use.

1.1.2 Molecular Mechanics

The molecular mechanics approach to modelling a system describes the physical interactions between atoms as forces, in a similar manner to Newtonian mechanics. Each atom is considered to be a single physical particle with a fixed radius, and is assigned a coordinate to identify its location in the system (often with respect to a 3D Cartesian space). Forces are primarily produced by the covalent and non-covalent bonds between neighbouring atoms. In addition, there are more subtle acts of both attraction and repulsion caused by factors such as polarisation and electrostatic interactions, which are often dependant on the distance between the atoms [1]. These non-bonded terms are often collectively referred to as wan der Waals forces [3].

A common requirement of molecular mechanical applications is the ability to estimate the potential energy of a system [2]. Although the precise method by which this is done is specific to the problem being solved, the underlying principle is similar for all models. The total potential energy of a system is considered to be the summation of the potential energies for each pair of atoms in the system. These individual energies are computed using a set of potential energy functions, which are themselves a summation of distinct terms that each contribute some energy to the system. These potential energy functions are parameterised on a per-atom basis, by a set of values that describe various properties of the atom, such as polarisation, steric hardness or electrostatic charge. The collective set of the parameters used to describe a system is known as a molecular forcefield.

1.1.3 Forcefield Development

The parameters of a molecular forcefield are intended to best approximate the interactions of the system whilst simultaneously keeping the required calculations reasonably simple. In particular, the van der Waals terms of the potential energy function, which often account for the majority of the calculations in the model, are typically approximated by modelling the fall-off rate of the force due to distance with a mathematically simple function and an absolute cut-off distance. While this results in a function that is efficiently computable, it isn't necessarily an accurate representation of the underlying interactions, which may exhibit more complex behaviours.

Derivation of forcefield parameters is typically done either via quantum mechanical calculations or empirically using an experimental data set. Quantum mechanical techniques for ascertaining these parameters are still maturing, and are again computationally expensive to perform [2]. As such, the empirical derivation methods are more commonly adopted when the creation of a new forcefield is required, and a substantial amount of research has been performed with regards to producing accurate approximations for the potential energy functions used in a variety of applications.

Traditional techniques used to derive forcefield parameters empirically involve a high degree of manual work, usually undertaken by an experienced chemist or bio-engineer. This is an iterative process, in which the forcefield is continually evaluated against a set of experimental data and refined until the amount of error between the model and the reference data set has been reduced to a satisfactory level. As well as being severely time-consuming, the complexity of the models mean that it can be difficult to predict the effects of altering any particular forcefield parameter. This can lead to the derivation process becoming reduced to a tedious 'trial and error' approach, without any clear direction.

No two molecular modelling problems are identical, and so it is extremely rare for any single forcefield to be effective for more than one specific application. This means that there is no such thing as a 'perfect forcefield', and that there will be a continuing need to develop new forcefields for new applications in the future. There do exist some forcefields that have been developed with a view to provide a more general parameterisation of the potential energy function that may be applicable to a range of similar problems, namely those in packages such as AMBER [4], CHARMM [5] and GROMOS [6]. These forcefields have become very popular and have been widely used in a range of molecular mechanics applications, however they still require the developer to tune the parameters to the specific problem before use. This is largely due to the fact that they focus on the way in which the model is parameterised rather than the values of the parameters themselves, and it is the latter problem which is application specific.

1.1.4 Applications

There are a vast range of molecular mechanical applications that utilise force-fields to model the molecular systems that they involve, such as molecular docking, protein folding, protein design, enzyme catalysis and analysis of cell-signalling mechanisms. This means that any progress towards addressing the challenges described in the previous section could potentially have a very high impact on future research in molecular modelling.

One particular application that has a clear commercial standpoint is that of drug discovery and design. The global pharmaceutical industry generates hundreds of billions of pounds in revenue each year, but bringing a new drug to market is an extremely lengthy process and incurs very high costs. Recent estimates suggest that the process typically takes well in excess of 10 years and the average cost of bringing a completely new drug to market is around $1.8 billion, with 46% of the capitalised cost occurring during the design and development stages [8]. As such, advancements in the technologies used for pharmaceutical research and development may produce significant financial savings for the company involved, as well as potentially decreasing the response time against new diseases.

1.2 Ligand-Protein Docking with BUDE

1.2.1 Molecular Docking Overview

Molecular docking is a particular application within molecular modelling which is used to predict the structure of two molecules that have bound together into a stable complex (see Figure 1). Knowledge of the resulting structure allows the prediction of the binding affinity of the molecules, which is a measure of how tightly they have bound.

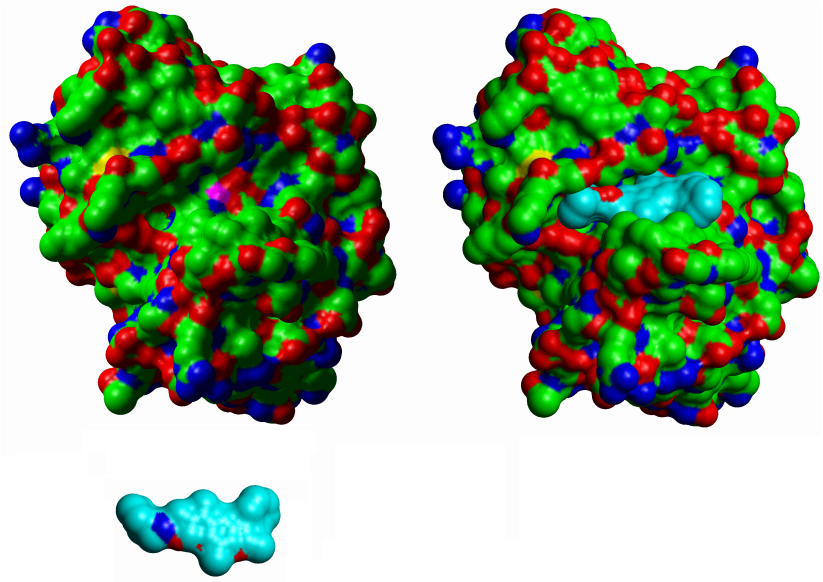
Docking has many practical applications in both industry and academic research, but one that is of particular interest is its use as a method of screening for candidate drug compounds. Given a target protein molecule that requires inhibition, a database of potential drug molecules (ligands) can be virtually screened with a docking program to assess which candidates are most likely to inhibit the target (i.e. which have the highest binding affinity). These top candidates can then be taken forward to physical screening tests and clinical trials.

An approach to molecular docking that is often (though not exclusively) employed by such programs is to treat the process as an optimisation problem. The target molecule (receptor) is held rigidly whilst the ligand is translated and rotated freely [7]. The problem is represented with a 6-dimensional search space, where each point in the domain corresponds to a position and orientation (collectively known as a pose) for the ligand. This grid of poses can be explored with a search algorithm, where the goal is to find the pose which minimises the binding free energy (or Gibbs free energy), thus resulting in a higher binding affinity.

This approach to docking is often very effective, and draws on extensive research in the areas of search and optimisation algorithms. However, an obvious side-effect of these algorithms is the requirement to evaluate the energies of a large number of poses, and so this evaluation must be efficient in order for the algorithm to be effective.

One simple and fast method of estimating the strength of the ligand binding in a particular pose is by using a scoring function, which simply counts the number of favourable and unfavourable atoms in contact with each other. The binding affinity is estimated to be the ratio of these counters. Although very fast to compute, this is arguably an over-simplification of the problem and the accuracy of dockings performed with this method are questionable. In addition, this method clearly provides no means to predict the actual free energy of the interaction, only the structure of the bound molecules [9].

Figure 1: A protein molecule and ligand, before and after docking



1.2.2 BUDE: Empirical Free Energy Forcefield Approach

As described in Section 1.1.2, traditional molecular mechanics forcefields yield the potential energy of a molecular system. In the case of molecular docking, it is the *free* energy we are interested in, which is equivalent to the difference between the enthalpy and entropy of the system. In molecular mechanics, the enthalpy is often considered to be approximately equal to the potential energy, and so this component can be estimated with a typical forcefield. However, computing the entropy component generally requires lengthy molecular dynamics simulations to be performed, which are very computationally expensive. Accuracy is another

issue with these calculations, as the free energy itself will be the difference between two very large numbers, from which it can be difficult to obtain a high precision result.

The School of Biochemistry at the University of Bristol have developed the Bristol University Docking Engine (BUDE), which uses an alternative approach to calculating the binding free energy in order to circumvent the issues described above. BUDE attempts to predict the free energy of interaction *directly*, as the sum of individual atom-to-atom energies in a manner similar to that of a traditional potential energy forcefield. This has been dubbed an empirical free energy forcefield by the authors, in order to distinguish it from classical potential energy forcefields. The aim of this approach is to provide an efficient method of predicting the binding free energy of a ligand-receptor pair in a specific pose, without excessively compromising the accuracy of these predictions. Unlike the scoring approaches, this also provides the capability to predict the free energy of the interaction, not just the structure.

1.2.3 Scope for Improvement

As with traditional molecular mechanics forcefields, BUDE's forcefield currently requires manual tuning by an experienced bio-chemist in order to achieve an acceptable level of accuracy. The current functions and parameterisation that BUDE uses are reasonably good at predicting the structure of protein-ligand complexes, and have generated promising results when screening for candidate drugs to treat the NDM-1 enzyme recently. However, the overall accuracy of the docking predictions still requires further improvement, as tests against a set of docked complexes only result in 64% of the structures being predicted correctly to within an acceptable error tolerance.

The current state-of-the-art docking programs quote their accuracy against ligand-protein test sets as around 80% of the structures predicted correctly, within a threshold of 2 Ångstroms. Although BUDE is not too far behind this mark, in order to become a viable alternative commercially it would have match or exceed this target.

1.3 Project Aims

This thesis investigates the use of machine learning techniques to automatically improve the accuracy of BUDE's free energy forcefield, using the current functions and parameterisation as a starting point.

The primary goals of this thesis are:

- Investigate existing approaches to molecular forcefield optimisation, both manual and automated
- Implement a framework for optimising BUDE's free-energy forcefield in an automated matter

- Compare and contrast multiple machine learning algorithms for their effectiveness in forcefield optimisation, in terms of speed and accuracy
- Investigate the use of these algorithms to generate an effective forcefield from scratch

2 Technical Background

2.1 BUDE

2.1.1 Docking Algorithm

The energy minimisation algorithm employed by BUDE is based on the evolutionary Monte Carlo approach described in [10]. This search comprises a sequence of generations, each of which involves evaluating the binding energy between the receptor and the ligand in a number of poses (the population). The first generation evaluates a population of poses generated uniformly at random over the 6D search space, and each subsequent generation uses a population containing poses which are randomly translated and rotated from the best poses in the previous generation. The algorithm terminates after a fixed number of generations, which is usually large enough for the population to converge.

It is worth noting that this process is currently aimed at predicting the structure of the bound complex, not the absolute value for the free energy of interaction. The binding energy reported by evaluating any single pose is therefore a *relative* binding energy, used only for comparison with other poses. In some cases, these relative energies can also be used to rank the binding affinities of different ligand molecules docked to the same protein target, which allows BUDE to be used to virtually screen for potential drug candidates. An eventual goal would be for BUDE to be able to predict both the structure and free energy of a complex accurately, as this would allow more concrete assertions to be made about the strength of potential drug candidates.

2.1.2 Acceleration

In order to accelerate the computation of the binding free energies, BUDE has been ported to OpenCL, allowing it to utilise the extensive parallelism available in GPUs and multi-core CPUs [11]. This allows BUDE to perform docking operations several times faster than the original Fortran-only implementation when running on a high-end GPU, and this will be crucial for evaluating the effectiveness of large numbers of forcefields during an automated optimisation approach. Further optimisations to the OpenCL kernel may be required in order to improve the overall performance of forcefield evaluation.

2.1.3 Forcefield Design

The forcefield that BUDE uses is a 'heavy-atom' forcefield, in which all atoms excluding hydrogen are modelled as spheres with a fixed radius, and hydrogen atoms contribute to the volume of the atom to which they are attached [11]. Each atom type is assigned a van der Waals radius and electrostatic type, which are taken from publicly available data. There are six more forcefield parameters assigned to atoms which are determined empirically, using a set of reference data. Although each atom has its own value for all of these parameters, some of the parameters are currently kept globally consistent for all atoms.

Hardness The hardness parameter is used to model a steric energy between two atoms that overlap, resulting in a repulsive force. This force is proportional to the amount of overlap that occurs and is multiplied by the sum of the hardness values for the two atoms. This parameter is currently set to be equal for all atom types.

Radius Scaling This parameter provides a means to scale the radius of an atom by a certain factor. Although radii are considered have a 'correct' value in terms of the atoms' physical properties, this parameter allows for interactions to take into account a radius that isn't strictly physical. This parameter is currently set to be equal for all atoms.

NPNP-Distance This parameter defines a cut-off distance for estimating the solvation energy of non-polar to non-polar interactions. This parameter is equal for all atoms.

NPP-Distance This parameter defines a cut-off distance for estimating the solvation energy of non-polar to polar (and vice-versa) interactions. This parameter is equal for all atoms.

Hydrophobic Potential This parameter represents the desolvation potential of an atom, and is used to model the solvation energy of a pair of atoms. Each atom type has an independent hydrophobic potential. For non-zero potentials, the polarity of the atom is known, and so the sign of the potential is fixed.

Electrostatic Potential This parameter is used to model the electrostatic energy of two atoms, which can either be an attractive or a repulsive force depending on the charge of the atoms. Each atom type has an independent electrostatic potential. As with the hydrophobic potential, the charges and therefore signs of these potentials are known, and in some cases the magnitudes are also known.

Not all possible atom types are covered by the current forcefield, and there is a 'catch-all' case for any types not explicitly covered elsewhere in the forcefield. The functional forms in which the parameters are applied isn't necessarily final, and it is possible that further development of this would be required in order to improve overall accuracy. Adding new atom types to the forcefield and manipulating the functional form of the interaction calculations are beyond the scope of this work and will not be further explored in this thesis.

2.1.4 Forcefield Optimisation

The effectiveness of BUDE's forcefield is assessed using a set of experimentally docked molecules from BindingDB, a publicly available database [12]. This data set provides the measured positions of atoms for the docked ligand along with

the binding affinity of the complex. These reference structures can be docked with BUDE, and then the predicted docking pose can be compared against the actual pose using the root-means-square deviation (RMSD).

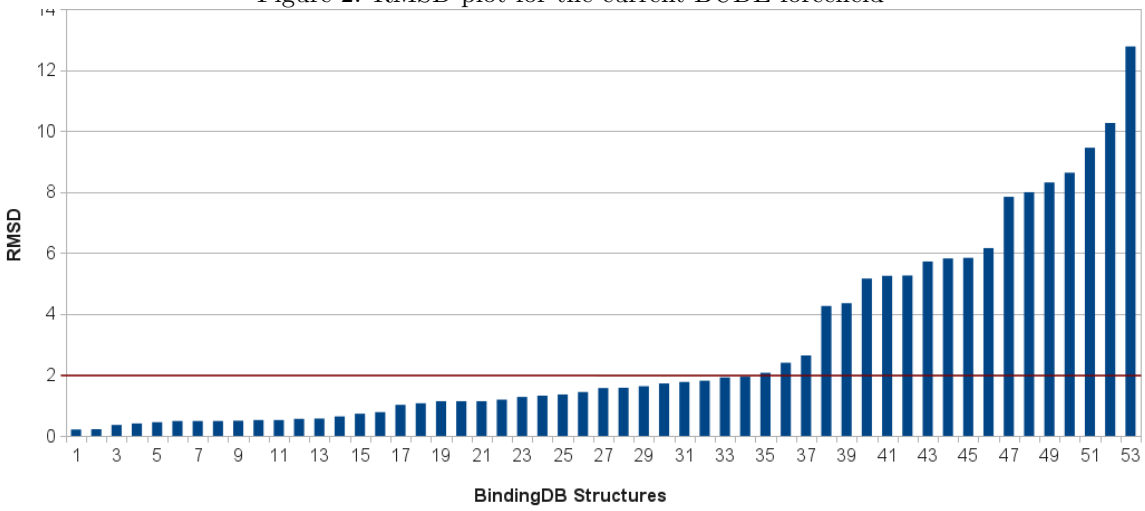
Given predicted positions p and reference positions r for n atoms in the docked ligand, the RMSD is computed as follows:

$$RMSD(r,p) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r_{ix}-p_{ix})^2 + (r_{iy}-p_{iy})^2 + (r_{iz}-p_{iz})^2}$$

A set of 53 suitable structures from BindingDB have been selected for evaluating BUDE's forcefield. A common target for declaring a single docking to have been predicted correctly is to achieve an RMSD of 2Å or less. Therefore, the overall accuracy of a molecular docking program with respect to a particular data set can be defined as the proportion of complexes that are predicted to within 2Å of the actual structure.

The current procedure for modifying the forcefield against such a data set is difficult to define. Although an experienced chemist may be able to analyse to some extent the cause of certain mispredicted structures or predict the effects of changing one particular forcefield parameter, for the most part the sheer number of parameters and the complexity of the energy calculations means that the optimisation process is often reduced to little more than trial-and-error. This is one of the primary reasons that manual forcefield tuning is so time-consuming.

Figure 2: RMSD plot for the current BUDE forcefield



The developers of BUDE have spent a considerable amount of time and effort refining the current forcefield parameters to maximise the accuracy of the docking predictions. When tested against the BindingDB data set, BUDE

predicts 34 out of the 53 structures to within an RMSD of 2Å, yielding an effective accuracy of 64%. A plot of the RMSDs for each structure is shown in Figure 2, sorted by RMSD. This plot shows that while there are many structures being predicted with a high degree of accuracy, there is a noticeable drop in the accuracy of the predictions for the remaining structures. It is worth noting that the search grid used by BUDE for docking these ligands is only 14Å across, and so the highest RMSDs seen here are essentially no better than a pose picked at random.

The forcefield currently has separate parameterisation for the atoms in ligands and for each type of residue in the protein. This means that there are over 300 individual atom types that require parameterisation, which drastically increases the difficulty of generating and optimising the forcefield. In addition, assigning atoms of the same type different potentials isn't an ideal model; in theory atoms of the same type should have the same behaviour regardless of which residue or molecule they are in.

2.2 Related Work

2.2.1 GROW

Hülsmann et al investigate the use of gradient-based numerical algorithms for the purposes of optimising molecular forcefields to simulate physical properties in [13]. The algorithms utilise a quadratic loss function to both evaluate a forcefield with respect to experimental data, but also as a means to manipulate the current parameterisation. The latter assumes a smooth dependency between the loss function and the forcefield parameters; i.e. following the direction of the descent will result in further improvement. While this assumption may hold true for the simulation of certain physical properties of a system, it is unlikely to be the case when approximating a full set of molecular interactions, as in docking.

A further requirement of the presented methodologies is that the initial parameter set must be "reasonably close to the minimum", and achieving this starting condition is stated to be an unsolved problem that is left up to the user. This task however is often the most troublesome, requiring tedious manual tuning in order to produce reasonable results. It is precisely this problem that one would hope to eliminate, at least to a significant extent, by introducing automated optimisation approaches. The authors close by recognising that the work is only a presentation of the methodologies under analysis as a basis for further work, and has not been used to realise high-quality forcefields for practical use.

2.2.2 Parmscan

Wang and Kollman compare the use of systematic search and genetic algorithms to reproduce physical properties such as vibrational frequencies for certain molecules [14]. A prevailing observation in the work is that while systematic search is effective for very small parameter sets, the genetic algorithm based

approach is far more competent at optimising multiple parameters simultaneously. Although certainly an interesting and useful result, the problem which the forcefields are being applied to is the simulation of physical properties of a single molecule. This is a very different problem to that involving the estimation of interaction energies between two or more molecules, in which the functional form of the parameters is significantly more complex.

2.2.3 Simplex Approach

Faller et al used an implementation of the Simplex optimisation algorithm in order to optimise forcefield parameters for molecular dynamics simulations [16]. Although often an efficient algorithm, Simplex is generally only effective for problems with a small dimensionality, and so in this work the authors had to limit the number of parameters which they chose to optimise, rather than parameterising the whole forcefield. As with GROW, this work requires that the initial parameter set is close to the optimum, so the algorithm is only required to perform final tuning.

2.2.4 QM/MM Force-Matching

Maurer et al introduce the concept of force-matching in conjunction with quantum mechanical simulations to optimise a bio-molecular forcefield [15]. This work is interesting in that the forcefield being parameterised is used for complex molecular mechanical simulations, including a direct set of interactions between two molecules in solvation. The authors conclude by briefly proposing extensions to the work that would allow the prediction of binding free energy for receptor-ligand docking. Unlike the work proposed in this thesis however, the parameters used to estimate van der Waals forces are excluded from the optimisation approach.

The motivation behind this work is that the novel approach allows parameterisation with accuracy comparable to that of quantum mechanical methods, but at a significantly lower computational cost. Unfortunately, it is unclear exactly what the magnitude of this saving is; although they acknowledge the 'generous allocation of CPU time' by a sponsoring body, the authors neglect to provide any information about the runtime required for these parameterisation experiments.

2.3 Optimisation Algorithms

2.3.1 Overview

Mathematical optimisation algorithms are an important concept that have uses spanning many disciplines. In particular, they are often applied to fields within computational science to either solve or approximate problems that are too large or difficult to solve with alternative approaches. There are a huge number of different optimisation algorithms each with their own properties and uses, and as such it can often be difficult to predict which will perform best when applied

to a particular problem. Indeed, the 'No Free Lunch Theorems' suggest that when averaged over all possible problems, all search and optimisation algorithms perform equally well [17, 18].

The selection of an optimisation algorithm relies on many different factors, and so when deciding which to use it is useful to ask several questions about the problem being solved:

- How much information about the problem is available?
- Should the solution should be exact or approximated?
- Is the problem deterministic or probabilistic?
- Are there any constraints on runtime or computational resources?

The work presented in Section 2.2.1 utilises gradient descent to optimise a forcefield. Gradient descent and other numerical approaches to solving optimisation problems are popular tools for a large number of well-defined problems, as they often present useful convergence properties. A common requirement between these techniques however is that there needs to be a strong relationship between the characteristics of solutions and their resulting fitness. For gradient descent algorithms the requirement is that the function being optimised is differentiable. This is because these methods draw on the fitness of solutions to formulate the next candidate, and this breaks down for probabilistic or ill-defined problems.

Although these requirements might hold true for many problems, molecular mechanical applications tend to be invariably complex and it is very difficult to formulate a relationship between the parameters and the fitness of the solutions that meets these criteria. The works presented in [13, 14, 16] are based around the estimation of physical properties of a system, which often *does* have such a relationship. The process of molecular docking however, if approximated by a molecular mechanical forcefield, cannot be so easily characterised.

BUDE, like many molecular mechanics based docking programs, is built on an energy minimisation approach. The process of finding the pose with the lowest binding free energy is performed by a non-deterministic search algorithm (a genetic algorithm, in the case of BUDE). The complexity of the calculations being performed, coupled with the non-deterministic nature of the process, mean that linear programming or iterative optimisation techniques that are successful to optimise parameters for simpler molecular modelling systems cannot be effectively used for accurately approximating the interactions carried out in docking.

An alternative to deterministic and iterative optimisation methods are metaheuristic approaches. Like the approaches mentioned above, these methods also optimise a problem with respect to a quality function that can evaluate candidate solutions. However, few assumptions are made about the problem and the algorithms are often probabilistic, which allows them to circumvent some of the issues encountered with traditional numerical optimisation techniques. The downside of these methods is that they do not guarantee that an optimal or near-optimal solution will be found.

In this thesis, the use of the hill climber and the genetic algorithm for force-field optimisation are analysed. These two approaches were selected as they provide very different properties within meta-heuristics, as so allow for interesting comparisons to be drawn. Both of these algorithms require the definition of a fitness function, which can take a candidate solution and produce a numeric value representing how 'good' that solution is.

2.3.2 Hill Climbers

Hill climbing is a local search based optimisation technique. This is a single-point search algorithm, in which only one solution at a time is considered during the optimisation process. At each iteration of the algorithm the 'neighbourhood' of the current solution is considered, where the neighbourhood is defined as all the solutions that can be reached by making a single change to one parameter of the solution. The algorithm will replace the current solution by one of its neighbours that is an improvement with respect to the fitness function. There are a variety of different ascent types that give the search different properties.

Simple hill climbing In this method, the hill climber will move to a new solution as soon as one is found that improves on the current solution.

Steepest ascent hill climbing This method evaluates all the solutions in the neighbourhood and moves to the one which offers the greatest improvement.

Stochastic hill climbing Here neighbours are evaluated at random, and a probabilistic decision is made whether or not to move to them, based on the amount of improvement they offer.

All of the hill climbers suffer the problem that they will stop at local optima, which may not be close to the global optimum. The non-determinism offered by the stochastic hill climber allows it to be run multiple times for the starting solution to find multiple different local optima, from which the best can be selected. The simple and steepest ascent methods can somewhat emulate this by incorporating a random-restart approach, in which the algorithms are restarted with a different starting solution, often mutated at random from the original.

As they only perform one change per iteration, it can take a long time for a hill climber to find a local optima, particularly if the problem has many different parameters to optimise. Additionally, if there is a high degree of dependence between parameters, a hill climber may not be the best approach; some improvements may only be possible by adjusting two or more parameters together.

It can also be considered a useful property that hill climbers only make one change at a time. This can be important if the user wishes to learn something about the problem from the optimisation process, as they can observe the changes that are being made and what effect they have on the fitness.

2.3.3 Genetic Algorithms

Genetic algorithms are search methods that belong to the branch of algorithms that draw on inspiration from biological concepts. First proposed by Holland in the 1970s [19], they essentially mimic the process of evolution and natural selection.

There are a large number of variants to the genetic algorithm, but for the most part they are all based around the same building blocks. A population of solutions is maintained throughout the search, often initialised with solutions generated at random. New solutions are generated by combining or mutating solutions from the current population. This process repeats until a stopping criteria is observed; for example if a satisfactory solution is found or a certain number of iterations are exceeded.

The primary method of evolving new solutions is via the crossover operator, which combines two or more 'parent' solutions to produce a 'child' solution. Parents are selected from the population non-deterministically, but typically biased in some way towards solutions that have a higher fitness. Many different crossover operators have been developed, and the choice of which one to use largely depends on that way in which solutions are represented and the problem being optimised [20].

Solutions are often represented as strings of parameters, or 'genes'. At a high level, the crossover process involves selecting some of these genes from each parent, and so the child will inherit characteristics from both, similar to the process of reproduction in biology. The simplest of crossover techniques involves segmenting the genes at one or more points and then combining these segments from each parent to form a child (these are termed 'point crossovers'). A uniform crossover treats all genes independently, with each parent having a chance of contributing to each particular gene [20].

Another common method of introducing diversity into the population is by applying random mutation to solutions, either after or instead of crossover. Mutation generally occurs with a low probability, and often only a few of the genes in the child are altered. For binary genes the bit can simply be flipped, whereas for numeric genes a new value can be chosen within some boundaries. Mutation is an effective method of preventing the population from converging to the same solution and becoming stuck.

The classical genetic algorithm involves producing an entire population of new children and then replacing the old population in one movement. An alternative approach is to perform incremental replacement, in which new children are generated one at a time, and immediately inserted into the population if they exhibit a satisfactory fitness. This method is interesting because it allows a new child with a high fitness to contribute immediately to reproduction, potentially allowing improvements to propagate more quickly [21].

As with all meta-heuristic optimisation algorithms, there is no guarantee that a genetic algorithm will find a solution that is near the global optimum. This method of search typically involves evaluating a large number of different solutions, and so can become a lengthy process if evaluating a single solution is

time-consuming.

Any particular implementation of a genetic algorithm has a large number of parameters that ultimately affect how well the algorithm will perform. Characteristics like population size, mutation rate, parent selection approach and choice of crossover operator all impact on the overall effectiveness of the search, but it is often not obvious exactly what these parameters should be. The problem of choosing optimal parameters for a genetic algorithm is an open question, and much research is being carried out to automate parameter selection with further machine learning algorithms [22, 23].

Unlike with the hill climbing optimisation approaches, it is often very difficult to analyse the changes that are made by a genetic algorithm in order to learn something about the optimisation process. Because of this, they are often considered to be black-box optimisers [24]. This is arguably a significant drawback for the use of genetic algorithms to solve problems in which the developer wishes to learn about the changes that were made in order to reach a good solution.

3 Requirements

3.1 Software Requirements

The third-party software components used for the implementation in this thesis are:

- BUDE - provided by Dr Richard Sessions from the School of Biochemistry
- BindingDB - a publicly available database of measured binding affinities (www.bindingdb.org)
- The Boost C++ libraries, specifically the program options and random number libraries
- OpenCL - a standard and API for expressing high-level parallelism for execution on parallel architectures (specifically GPUs in this implementation)

3.2 Hardware Requirements

Aside from during development, all of the experiments run for this thesis were performed on GPUs on BlueCrystal, the supercomputer provided by the Advanced Computing Research Centre at the University of Bristol. Specifically, the hardware used was:

- Four independent nodes comprising 2x NVIDIA Tesla M2050 GPUs (BlueCrystal Phase 2)
- Six nodes comprising 2x NVIDIA Tesla M2050 GPUs, used both independently and as a cluster (BlueCrystal Phase 1, available from May 4th)

4 Execution

4.1 Design & Implementation

4.1.1 Overview

The primary aim of this implementation is to provide a highly flexible framework for optimising molecular forcefields in an automated environment. Although initially targeted at optimising forcefields for BUDE, the framework should be extensible enough to be easily applied to other forcefields and programs, potentially with very different structures. Since it is not clear which learning algorithm will be most effective at optimising these forcefields (and it may even depends on the program or problem being targeted), the framework should be able to support multiple optimisation techniques, and allow easy implementation of additional methods in the future.

In order to achieve these goals, the implementation will be broadly divided into two main components. One component will provide a means to evaluate an arbitrary forcefield in terms of some performance metric. The other will apply an automated learning algorithm to optimise a forcefield, using the evaluation component as a fitness function. The interface between the two will be well defined, and will allow alternative evaluation methods to be substituted as needed.

4.1.2 Forcefield Evaluation

The source code for BUDE has been provided by the developers, along with the BindingDB test set in a BUDE readable format and some tools for extracting relevant details from BUDE's output data.

Evaluating a forcefield is done by running BUDE for each of the 53 complexes in the BindingDB data set. This involves preparing input files for each test case, initiating BUDE processes, and then extracting the RMSDs. This process has been wrapped up into a single BASH script, allowing the evaluation of a forcefield to be performed with the execution of a single command. There are a number of different configurations available for this process, allowing a variety of evaluation metrics.

BUDE itself is highly configurable in terms of the search strategy that it employs to minimise the binding free energy, and so it is important to consider the strategy used when evaluating a forcefield. The evaluation script implemented here allows the user to specify a template input file and search configuration file for BUDE as command-line arguments.

Given a set of RMSDs for the structures predicted by BUDE, the simplest metric for describing the forcefield's overall accuracy would be to take the average of these RMSDs. More formally, given the predicted structures P and reference structures R for N complexes, the fitness function F for a forcefield is defined as:

$$F(forcefield) = \frac{1}{N} \sum_{i=1}^{N} RMSD(R_i, P_i)$$

An alternative metric for assessing a forcefield is to instead focus on achieving the maximum number of structures with RMSDs below some acceptable threshold. This is more representative of the typical goal for docking programs of predicting structures to within 2Å of the actual structure. This is implemented by taking the sum of the RMSDs that are above the threshold. For a given threshold T, this fitness is defined as:

$$F(forcefield) = \sum_{i=1}^{N} G(RMSD(R_i, P_i),\ T)$$

where:

$$G(D,T) = \begin{cases} D - T, & if D > T \\ 0, & \text{otherwise} \end{cases}$$

Another approach to determining the fitness of the forcefield is to consider the RMSDs of a range of poses predicted to have a high binding affinity, rather than just the best pose. Although in a practical application of the docking procedure in which no reference data is available this has little meaning, this can be a useful method of analysing the reasons why the structure was mispredicted. For example, if the best pose reported by the docking program has a high RMSD but the correct pose appears near the top of the list as well, then this might be an indicator of some geometric symmetry in the ligand. To consider the pose with the lowest RMSD out of the top m poses predicted, the calls to the $RMSD$ function in the above definitions can be replaced with:

$$\min_{1 \leq j \leq m} RMSD(R_i, P_{ij})$$

4.1.3 OpenCL & GPUs

It is important that evaluating a forcefield against the full data set is fast, as the speed at which this can be done will affect the amount of solutions that can be explored by the learning algorithm. Although BUDE had been previously ported to OpenCL to allow execution on GPUs, further optimisation was required in order to improve the overall runtime of evaluating a forcefield.

After profiling BUDE with NVIDIA's Compute Visual Profiler, it was apparent that although the free energy calculations were running significantly faster on the GPU, the host code was now accounting for a significant portion of the overall runtime. In order to alleviate this, some more computation was moved into the OpenCL kernel, namely the generation of the transformation matrices used to manipulate the ligand. In addition to this, an inefficient implementation of a partial sorting algorithm written in Fortran was identified, and replaced by a more efficient alternative. Finally, a work-padding scheme was introduced to allow the parallel computations to be more effectively mapped onto the GPU architecture.

BUDE achieves the best performance when screening a large number of different ligands against a single protein target, as this results in enough GPU

computation to amortise the overhead of initialisation. The complexes provided in BindingDB relate to different targets however, and so must be processed by individual BUDE processes. In order to improve the performance of forcefield evaluation under these conditions, the scripts implemented here are designed to process several complexes in parallel, using multiple OpenCL enabled devices at once.

To achieve this, BUDE was modified to allow target OpenCL devices to be selected via command-line arguments. The evaluation script can receive a list of OpenCL device identifiers, and partitions the complexes evenly between them. A group of threads is spawned to manage these sets of complexes asynchronously and launch instances of BUDE targeting each device. Using one of the dual-GPU nodes available on Phase 2 of BlueCrystal, launching 3 instances of BUDE per GPU was found to achieve the highest GPU utilisation, and thus the best performance overall.

The 6 GPU nodes on Phase 1 of BlueCrystal are available for use as a cluster. The evaluation scripts were extended in order to accept a list of OpenCL device identifiers under specific hosts, allowing them to reap the benefits of using up to 12 GPUs concurrently. These nodes were not managed by any queueing system, as so were being shared by multiple users. In order to remain flexible in this environment, the usage of this cluster was implemented in such a way that hosts could be dynamically added or removed during a long optimisation run. This allowed for more GPUs to be added as and when they became available, or freed up for use by another process if necessary.

4.1.4 Optimisation Framework & Configuration

The implementation for the optimisation framework is written in C++, and makes use of object-oriented concepts to provide extensibility. An abstract base class `AbstractOptimiser` defines an interface for the implementation of an optimisation algorithm. New algorithms can be added to the framework by extending this class and overriding the pure virtual `run` method. A small amount of additional code needs to be inserted into the `loadOptimiser` method to make the algorithm accessible from the configuration file.

The Boost C++ `program_options` library is used to provide a high-level configuration file interface to control the optimisation parameters for a particular run of the program. The purpose of a configuration file is to control the following aspects of an optimisation run:

- The command to use when evaluating a forcefield
- The constraints of the parameters in the forcefield being optimised (see Section 4.1.5)
- The type of optimisation algorithm to use
- Any algorithm specific parameters

A full example of a configuration file is given in Appendix A. The evaluation command supplied in the configuration file presents the interface between the optimisation algorithm and the forcefield evaluation component. This command will be run as a subprocess whenever the optimisation algorithm needs to evaluate a forcefield, with the filename of that forcefield appended to the command-line string. This is where any options to be passed to the evaluation script can be set, and also allows for alternative evaluation commands to be substituted without any changes to the source code. There are two requirements of an evaluation command specified with this option:

1. The filename of the forcefield to be evaluated will be specified at the *end* of the command-line string
2. The command should output a single, decimal floating point number as the fitness for the forcefield, on the standard output stream

The fitness output by the command is assumed to be ≥ 0 and that lower is better, i.e. a fitness of 0 is a perfect forcefield for the evaluation metric being used. If a negative fitness is returned or the command exits with a non-zero return code, an error is assumed to have occurred and the optimisation run will fail.

4.1.5 Forcefield Manipulation

So that the forcefields output by the optimisation process are not considered to be 'physically unreasonable', constraints are imposed on each parameter, and are defined in the configuration file. A constraint for any particular parameter consists of:

- A minimum value for the parameter
- A maximum value for the parameter
- An increment value defining the smallest change that can be made to the parameter
- A standard deviation to use when randomly mutating the parameter with a normal distribution

For parameters that are global to all atom types, the constraints are specified as absolute values. For per-atom parameters, a single constraint is given for all the atom types, and is specified as a percentage of the starting value for any particular atom type.

The `Forcefield` class encapsulates all of the parameters in a given forcefield, and provides a set of methods for manipulating these parameters, with respect to the current set of constraints.

Forcefield::crossover() This function implements a version of the uniform crossover operator typically used by genetic algorithms. This is applied to two forcefields, and can optionally take a bias parameter which defaults to 0.5. Each parameter in the resulting forcefield is selected independently from one of the two 'parents'. The parent which contributes that particular parameter is decided by generating a random number uniformly between 0 and 1 and comparing it to the bias value. Thus, a bias of 0.5 causes each parent to contribute equally to the child.

Forcefield::generateNeighbourhood() This function generates a list of new forcefields, where each has a single incremental change from the original. For each parameter in the forcefield, two new forcefields are created by adding and subtracting the increment amount to or from the current value. This is useful for optimisation algorithms that explore incremental changes to the current solution, such as hill-climbers.

Forcefield::mutate() This function mutates a forcefield by randomly varying parameters with respect to some mutation rate. A parameter p is provided denoting the percentage of the parameters which should be altered. For each parameter in the forcefield, a random number is generated between 0 and 1 and the parameter will be mutated if the number is less than or equal to p.

Mutation can either be performed by selecting a value uniformly at random between the upper and lower bounds it can take, or by applying some Gaussian variation. If the latter approach is used, a random number is generated from a normal distribution with a mean equal to the current parameter value and standard deviation equal to that specified in the constraint for that parameter. This value is clamped to within the constraints and snapped to a multiple of the increment.

4.1.6 Hill Climber

This optimisation algorithm relies on the function that can generate a neighbourhood of stepwise changes for a particular forcefield, described in Section 4.1.5. Two approaches to hill climbing have been implemented here: steepest ascent and stochastic. The steepest ascent hill climber evaluates every forcefield in the neighbourhood and moves to that with the best fitness, while the stochastic approach traverses the neighbourhood in a random order and moves to the first forcefield it finds that improves on the current one (a slight variation on the traditional stochastic hill climber). The configuration file is used to specify which ascent type to use, and optionally the maximum number of iterations to perform.

4.1.7 Genetic Algorithm

A highly configurable genetic algorithm has been implemented that uses the crossover and mutation functions described in Section 4.1.5. The algorithm is

characterised by the following parameters, which can all be set via the configuration file:

- A population size
- A limit on the number of iterations to perform
- A mutation rate for generating the initial population
- A flag to enable/disable uniform crossover
- A bias to use when performing crossover
- A probability of mutating a child after crossover
- A mutation rate to use when performing mutation

The population is initialised by applying uniform mutation to the initial forcefield, and is maintained sorted by fitness. The algorithm employs incremental replacement, in which new forcefields are generated one at a time. If the new forcefield is better than than the worst forcefield in the population, it is inserted into the population and the worst forcefield removed. This allows a new forcefield with a high fitness to immediately contribute to the next forcefield generated, potentially accelerating improvements.

In order to avoid premature convergence, new forcefields are only inserted into to the population if they have a unique fitness. This strict notion of uniqueness was deemed necessary after preliminary experimentation indicated that the population of forcefields tended to quickly converge to a set of forcefields that achieve the same fitness but differ only very slightly, causing further improvement to become very difficult.

If crossover is enabled, parents are selected from the population at random biased towards the fitter forcefields, using a normal distribution. For biased crossovers, the fitter of the two parents is granted the bias. When applying mutation to children, the Gaussian variation method is used.

4.2 Experiments & Results

4.2.1 Overview

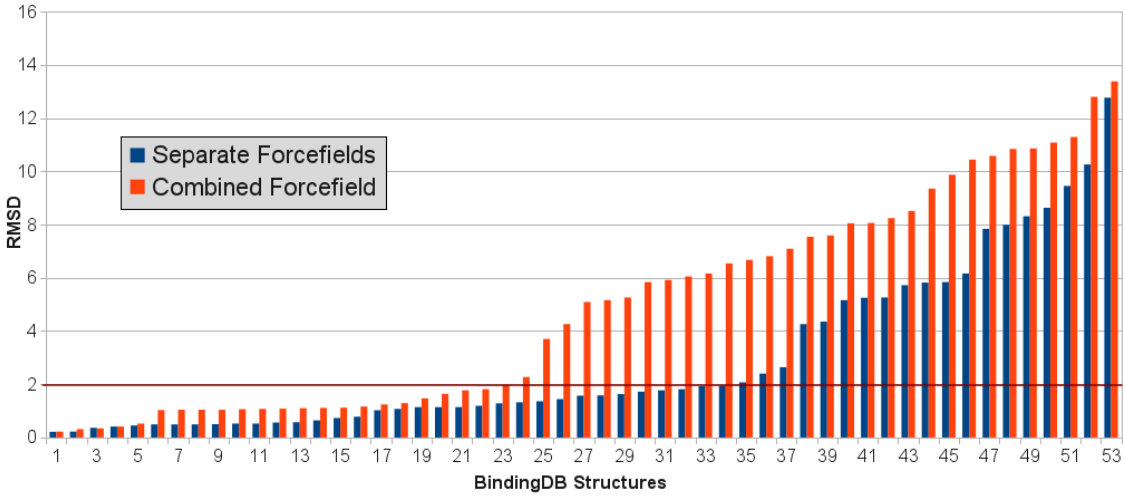
The two main goals of the experiments carried out in this thesis are:

1. Produce a free energy forcefield that improves the accuracy of docking predictions performed by BUDE
2. Assess the effectiveness of using machine learning as a tool for molecular forcefield optimisation

As mentioned in Section 2.1.4, BUDE is currently using separate parameterisations for atoms in different protein residues and for the ligand. A model that better represents molecular mechanical systems is one that simply parameterises

atoms by their type, regardless of which molecule or residue they reside in. The necessary implementation in BUDE to allow it to use a forcefield with this structure has already been performed by its developers. The forcefield parameters have not yet been optimised for this model however, and so the current accuracy of BUDE using this approach is poor; only 43% of the BindingDB structures are predicted correctly. Figure 3 shows a comparison between the separate and combined forcefield approaches. As such, an additional goal of the work here is to produce a forcefield which achieves a satisfactory level of accuracy in this combined forcefield approach, allowing the developers of BUDE to migrate fully to this approach for future research they undertake.

Figure 3: Comparison of initial forcefield for separate and combined approaches



The primary metric used for comparing forcefields in terms of their accuracy will be the percentage of the complexes in the BindingDB test set whose structures were predicted to within an RMSD of 2Å of the actual structure. This metric is fairly coarse however, and so this figure will be frequently supplemented with plots that demonstrate the RMSD achieved for individual complexes, as in Figure 3.

4.2.2 Considerations

Although forcefield evaluation is performed using multiple GPUs to accelerate the docking computation, the amount of time required to evaluate a single forcefield is still very significant. When running on the dual-GPU systems made available on BlueCrystal, the runtime of a single evaluation was just under two minutes. The number of forcefields evaluated by the experiments run here was typically a few thousand, and so the runtime of these experiments was generally a few days. With the limited amount of hardware available, this severely reduced

the total amount of full experiments that could be carried out in a reasonable time-frame.

It is worth noting that the evaluation function is somewhat non-deterministic, in that BUDE relies on a search algorithm to perform the dockings. This is arguable an undesirable property of the fitness function, as it is possible that a mispredicted docking might be a shortcoming of the docking process as opposed to an inaccurate forcefield. A partial solution to this problem would be to perform multiple runs of BUDE for each complex and average the RMSDs. This approach would drastically increase the runtime of the overall optimisation process however, and so seems unviable without the aid of additional computational resources.

4.2.3 Forcefield Constraints

As mentioned in Section 4.1.5, constraints are imposed on the forcefield parameters during optimisation in order to ensure that their final values are deemed to be 'sensible' in terms of the molecular mechanical behaviours they represent. However, it is not at all obvious what these constraints should be. The constraints were initially set to be fairly restrictive, with the expectation that any parameters needing more freedom could be identified by checking for values that have reached either their upper or lower limits during optimisation. Although this did allow for some over-constrained parameters to be identified and dealt with, there were other cases that didn't become apparent so easily. On several occasions there were parameters that hadn't hit their limits after the optimisation algorithm had converged, but re-running the experiment with looser constraints saw the very same parameters go far beyond the limits they had been assigned previously.

This behaviour is likely attributed to the highly epistatic nature of the problem being optimised. The parameters are ultimately used in a set of calculations that are applied to a *pair* of atoms. Whenever the two atoms are of different types (which is clearly the common case), each of these calculations will depend on more than one forcefield parameter. Even in the few cases where the atoms are both of the same type, many of the calculations are related, for example the hydrophobic potentials are used in conjunction with the solvation cut-off distances. This high degree of inter-parameter dependence makes it difficult to assert whether any single parameter is required to exceed its current constraint, as it could equally well be that the apparent restriction is caused by an inaccuracy in a different parameter.

After performing several optimisation experiments, it became clear that there were several complexes that were consistently failing to achieve RMSDs at all close to the 2Å target. In order to determine whether this was simply the result of a badly parameterised forcefield or caused by a more fundamental issue, a set of further experiments were run in which forcefields were optimised for each complex individually. The results indicated that almost 10% of the structures still failed to achieve satisfactory RMSDs, even in this greatly simplified optimisation scenario. This may be indicative of certain atom types missing

from the forcefield which are significant to these particular complexes, or that the functional form of the forcefield needs some adjustment.

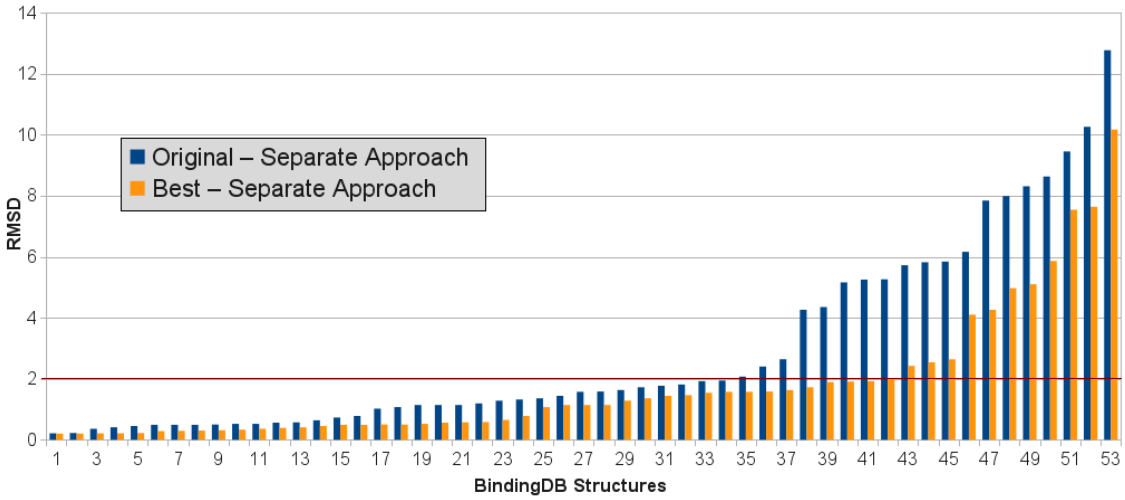
4.2.4 Best Results

Table 1 shows a comparison between the original and best forcefields for both the separate and combined parameterisation approaches, in terms of their overall prediction accuracy, the total RMSD still above 2Å and the average RMSD. These results were achieved after running genetic algorithms on the original forcefields and applying a steepest ascent hill climber to the outputs. The evaluation metric used during optimisation was the thresholded RMSD formula.

Table 1: Comparison between original and best results

	Prediction Accuracy	**Total RMSD Above 2Å**	**Average RMSD**
Original - Separate Approach	64%	78.92	2.9082
Original - Combined Approach	43%	172.04	4.9087
Best - Separate Approach	79%	35.24	1.7811
Best - Combined Approach	74%	43.98	1.8468

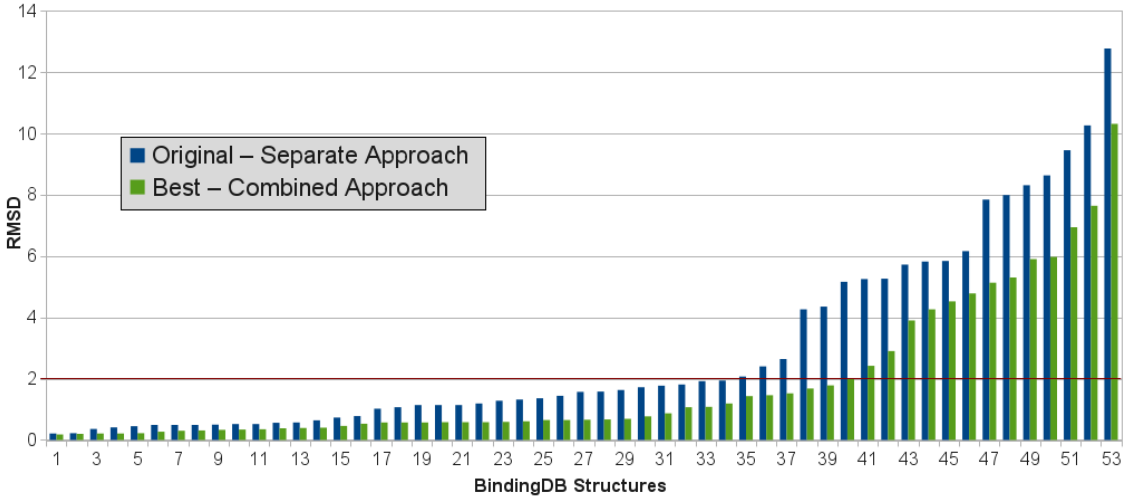
Figure 4: Best forcefield achieved for separate parameterisation approach, compared to original



The very best forcefield achieved throughout the experiments run here gives an overall accuracy of 79% against the test set. This is a forcefield optimised for the separate protein/ligand parameterisation approach, and is compared against the original in Figure 4. This is a very significant improvement from the original 64%, and is very comparable to the accuracy of commercial docking solutions.

Although there is still a noticeable drop in the accuracy of the worst complexes, there are 3 more complexes very near the target of 2Å, suggesting that with further optimisation an accuracy of around 85% may be achievable.

Figure 5: Best forcefield achieved for combined parameterisation approach, compared to original separate forcefield



The accuracy achieved after thoroughly optimising the forcefield targeting the combined parameterisation approach was 74%, and is shown in Figure 5. Although this is not as accurate as the best forcefield achieved for the separate approach, this is still a significant improvement from the original hand-tuned forcefield. It is also worth remembering that the initial forcefield for this approach was much worse that for the separate approach, and so the effective improvement achieved by the optimisation algorithm is actually much greater, as is shown in Figure 6.

Another interesting observation from these RMSD plots is that even though the evaluation metric used was only interested in reducing RMSDs that were above 2Å, both of the optimised forcefields presented here also show general improvements to the other complexes as well. This gives some confidence that the changes made to the forcefield parameters were actually sensible, rather than just coincidental.

Comparing the best forcefields for the two parameterisation approaches shows that the two forcefields generally perform similarly across the test set, shown in Figure 7. The separate approach achieves a noticeably higher accuracy on the complexes in positions 41-45, but interestingly the combined approach makes several improvements for complexes with RMSDs already below 2Å. This is perhaps an indicator that the more general 'by-atom-type' parameterisation approach is the more sensible of the two, although this requires further analysis to confirm.

Figure 6: Best forcefield achieved for combined parameterisation approach, compared to original combined forcefield

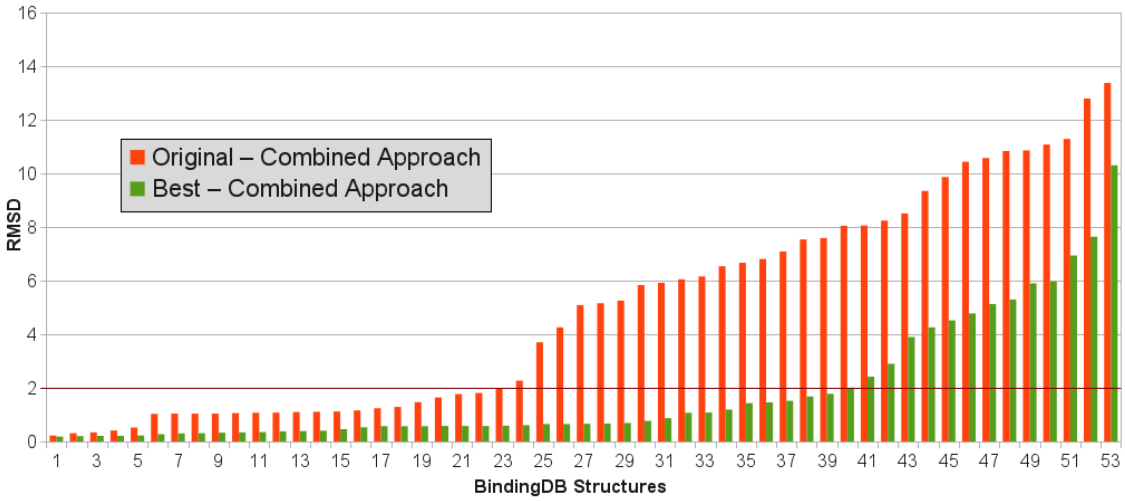
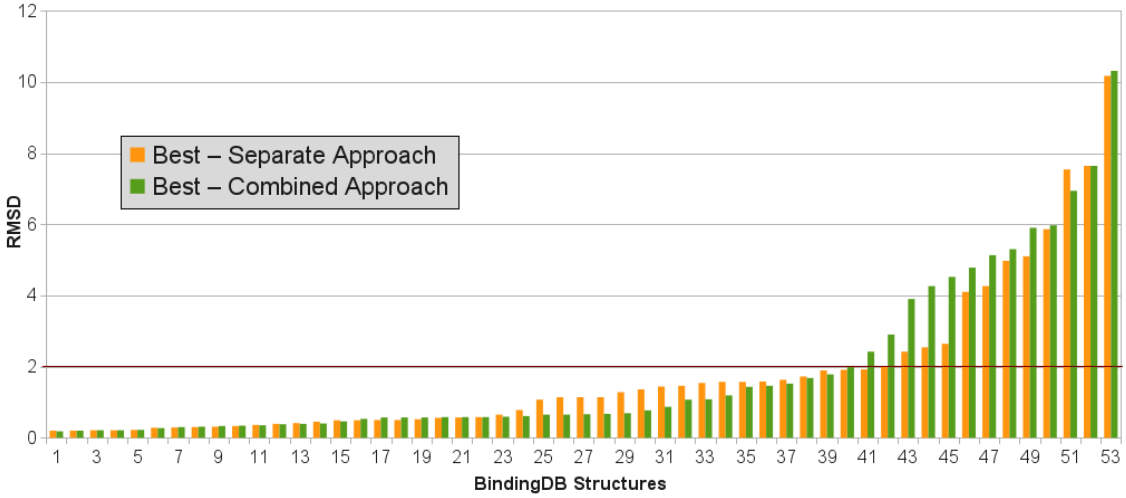


Figure 7: Comparison of best forcefields achieved for separate and combined approaches



4.2.5 Analysis of Learning Algorithms

Overall, the genetic algorithm produced significantly better forcefields than the hill climber. This was almost to be expected, as the hill climber suffers the problem of becoming stuck in local optima. This improvement comes at a cost though, with the runtime of the genetic algorithms used to achieve these results typically at least 5 times that of the hill climbers. Interestingly, the final solutions produced by the genetic algorithms often appeared to not be local optima, and applying a hill climber remedied this within one or two iterations.

Using the average RMSD as an evaluation criteria did allow the optimisation algorithms to bring the average down significantly, however in terms of overall prediction accuracy this method didn't have as much impact. The hill climbers for example managed to bring they average RMSD down by 0.5Å or more in relatively few iterations, but the number of complexes with RMSDs below 2Å didn't change. This seems to be caused by the fact that many changes are able to improve the accuracy for complexes which are already good, without making enough improvement to the complexes which are mispredicted.

It is clear that the different learning algorithms and evaluation metrics used affect not only the performance of the forcefields that are produced by the optimisation experiments, but also the *way* in which forcefields are modified. The latter is an important aspect of the optimisation process, as although the forcefield itself is only approximating the behaviour being simulated and there is no 'right' answer for any of the parameters, the values that these parameters take do have some actual meaning in terms of the molecular interactions that are occurring, and so there are clearly some values that make sense and others that don't. There is an interesting question that arises from this observation: can we analyse the output of an automated forcefield optimisation process in order to better understand the behaviour of the system we are trying to simulate?

This is one metric by which the hill climber is arguably a more suitable algorithm. Upon termination, generally between 5–15 iterations have been performed. Since each iteration performs a single, stepwise alteration to one forcefield parameter, the individual changes that have been made to produce the final forcefield are generally visible simply by comparing with the original. Analysing the trace of the hill climbing process also yields the order in which these changes happen, along with the impact that any particular change had on the fitness.

In contrast, the results produced by the genetic algorithm don't give away as much information. The forcefields output by this algorithm generally showed that every single parameter had changed, by various amounts. Even between two forcefields output that had similar or identical fitnesses there were often drastic changes to a large number of parameters, and so it is difficult to reason about which changes were significant and which were just byproducts of the highly randomised search method.

There are however some common observations to be made about the forcefields produced by the genetic algorithm and the hill climber. In all of the high performing forcefields produced by these algorithms the hardness parameter had been greatly increased from the original, indicating that the forces produced by

steric interactions play a more significant role in the overall binding free energy than previously estimated. The radius scaling parameter has remained unchanged at 1.0, implying that the van der Waals radius is effective without any adjustment.

Interestingly, when optimising with an evaluation function that selects the best RMSD out of the top 100 poses reported by BUDE, the genetic algorithm was able to produce a forcefield for which all but 2 of these RMSDs were below 2Å. Although this is certainly an easier problem to optimise for, there is some indication here that a few of the molecules might exhibit some geometric symmetry, as mentioned in Section 4.1.2. Whether or not this is truly the case requires further investigation, but if so would potentially explain many of the mispredicted structures in the main optimisation experiments. Automatically detecting and dealing with (semi-)symmetrical ligands is a non-trivial problem that would warrant extensive research.

It is worth noting that the search space of all the possible forcefields is incredibly large, as each parameter is real-valued. Even heavily constraining parameters and only allowing them to explore a set of 20 discrete values within an acceptable range yields a total of 2.8×10^{62} different forcefields in the search space. Due to runtime constraints even the most thorough searches performed here examined a few thousand forcefields at most, which is a minute fraction of the overall search space. That the algorithms made as significant improvements as they did under these conditions gives confidence that they are effective tools for optimising forcefields, and with further computational resources the overall improvements may be even greater.

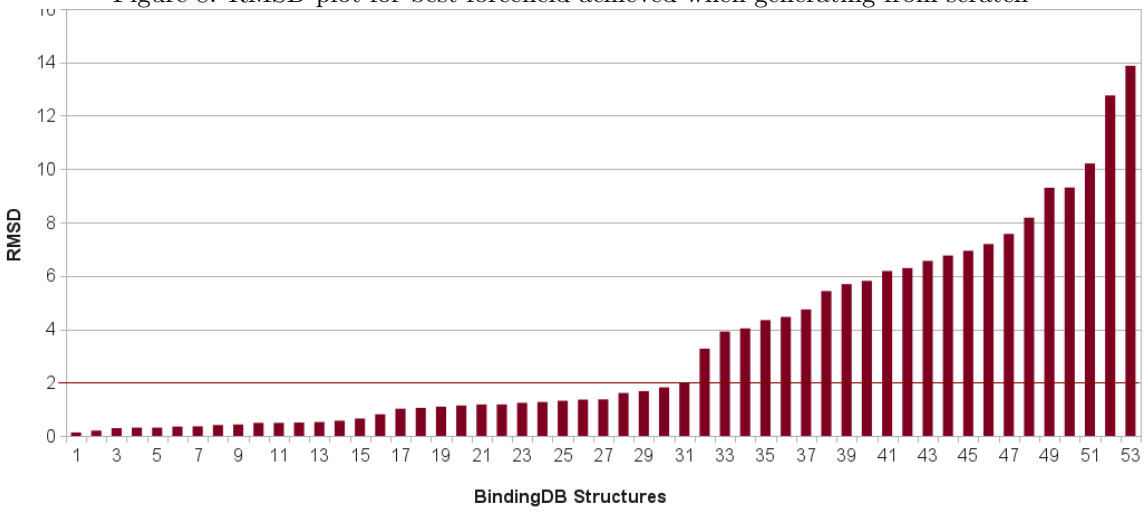
4.2.6 Generation from Scratch

The experiments thus far have focused on improving a forcefield which is already fairly well parameterised. However, although this is a useful method of improving parameters, relying on the existence of a well-tuned forcefield to start with limits the scope for applying this technique. It is arguably a much more powerful solution if accurate forcefields can be produced *without* this requirement; i.e. generated from scratch. This would potentially result in a much more significant saving in terms of cost and time by removing the need for any amount of manual tuning.

In order to keep the forcefield somewhat sensible, a small amount of direction was given to the initial parameterisation. In particular, the signs of the potentials were maintained, as these are known from the types of the atoms. Otherwise, parameters were all given fairly arbitrary starting values. Limits were still applied to the parameters, but were greatly relaxed. This initial forcefield predicted 16% of the BindingDB structures correctly, and even these can almost certainly be attributed to 'lucky guesses'.

For this problem, the hill climbing approaches were unable to produce a forcefield that performed at all well, outputting a forcefield with an accuracy of just 26%. The genetic algorithm on the hand did manage to make significant headway towards a reasonable forcefield, managing to reach an accuracy of

Figure 8: RMSD plot for best forcefield achieved when generating from scratch



58%. This is certainly not a reliable forcefield, but does show promise for using machine learning techniques to generate this kind of forcefield with little manual assistance. The RMSDs for this forcefield are plotted in Figure 8.

An interesting point of investigation with forcefields generated from scratch would be to examine whether the parameters in the output forcefields are at all similar to those selected by a human during manual tuning. Although some of the global parameters appear to be heading towards similar values seen in the other forcefields, it is difficult to say with confidence that this is genuinely the case. Until a high-quality forcefield is generated with this method, this comparison will likely remain inconclusive.

As mentioned in Section 4.2.5, the search space of all possible forcefields is exceptionally large. That the optimisation algorithms weren't able to produce high-quality forcefields when starting from nothing isn't necessarily a failing on their part, but more likely a consequence of the sheer size of task at hand. Clearly the optimisation techniques work well when given a good starting point, and so perhaps all that is needed to produce an effective forcefield from scratch is to allow them to perform further exploration. This obviously comes at the cost of a much greater computational requirement, but the experiments performed here were run on a fairly modest amount of hardware and it may be that a more powerful machine may make this possible.

5 Conclusion

5.1 Achievements & Critique

5.1.1 Automated Forcefield Optimisation Tool

This work has produced a framework for optimising molecular forcefields in an automated manner. The framework is extensible enough to allow the easy implementation of additional optimisation algorithms and forcefield evaluation functions.

More specifically, a hill climber and genetic algorithm have been implemented which can be used to optimise a forcefield against an arbitrary evaluation metric. These algorithms can be easily configured to modify the ways in which optimisation is carried out. These algorithms have been shown to produce good forcefields whilst only testing a very small portion of the total domain, when supplied with a starting forcefield that is already moderately well parameterised.

5.1.2 Improved Free Energy Forcefield

The free energy forcefield used by BUDE to predict the structure of protein-ligand dockings has been vastly improved by the use the automated optimisation tool. When tested on a set of measured docked complexes, BUDE now predicts 79% of the structures to within 2Å of the reference data, compared to 64% in the original forcefield. For comparison, the widely used GOLD docking suite, developed in collaboration with pharmaceutical giant GlaxoSmithKline, quotes an accuracy of 81% against reference data [25].

In addition, BUDE has been migrated to use a single 'by atom-type' parameter set, instead of using separate parameterisation for the ligand molecule and different residues within the protein. This model is both simpler to manage and more representative of the underlying mechanics of the docking procedure. Although this approach is currently only achieving an accuracy of 74%, with further optimisation it is expected to eventually surpass the accuracy of the separate parameterisation method.

The integrity of these forcefields with respect to the physical properties that the parameters represent needs to be confirmed to ensure that all of the values are sensible. These forcefields should also be tested against additional data sets, to ensure that the forcefields have not been over-fitted to the complexes provided in BindingDB.

5.1.3 Forcefield Generation

In addition to using machine learning techniques to optimise a forcefield which has already been manually parameterised, this work has also investigated the use of such learning algorithms to generate an effective forcefield *without* requiring this starting condition. Initial results demonstrate that the genetic algorithm can make significant progress towards a reasonably effective forcefield when

starting from 'scratch', but the forcefields produced are still much less accurate than the current best parameters.

The constraining factor is the limitations on runtime, which mean that the optimisation algorithms can't explore enough of the search space. With more computational resources, it may well be that this approach achieves satisfactory results. This would be an important result, as it would potentially remove a large amount of manual effort from the process of generating new forcefields in the future.

5.2 Future Work

5.2.1 Further Forcefield Improvements

The optimisation framework implemented here has been handed over to the developers of BUDE, who have begun further optimisation experiments to improve the accuracy of the forcefield. They will also analyse the parameters of the most effective forcefield produced by this work, to determine if the values that have been assigned are sensible.

In the near future, a 247 teraFLOP system comprising 372 NVIDIA Tesla M2090 GPUs will become available to the author. With further modifications to the framework to allow this amount of computing power to be efficiently exploited, optimisation experiments that explore the search space much more thoroughly could be be performed. In particular, this resource may provide enough computing power to allow accurate forcefields to be generated from scratch.

In order to more thoroughly test the accuracy of the forcefields, the use of the Astex Diverse Set [27] is currently being investigated. If the parameterisation achieved when optimising against the BindingDB set, then the forcefield should also be effective against other similar complexes.

5.2.2 Further Development of Optimisation Algorithms

Although the results demonstrated here show that genetic algorithms are effective tools for forcefield optimisation, further development of these algorithms provided by this framework will be undertaken in the future. Adding other heuristic and stochastic search algorithms would be interesting in performing further comparisons between these different approaches, and may be necessary if genetic algorithms prove ineffective for other forcefield optimisation problems.

Another more specific target for implementation would be a hybrid genetic algorithm, which have been shown to be more effective than classical genetic algorithms for many problems [26]. This would involve employing a local search algorithm, such as a hill climber, as part of the mutation operator. Although this would be straightforward to implement within the current framework, the runtime implications of this method would mean that optimisations using this algorithm would be impractical without the availability of more computational resources.

5.2.3 Energy Prediction

The optimisation attempts performed here have focused on producing a forcefield which accurately predicts the structure of protein-ligand complexes. The binding free energies reported by BUDE are currently only relative energies; i.e. they aren't comparable to the actual free energies of interaction as reported by experimental measurements. The accurate prediction of the absolute free energy would be another valuable capability for BUDE to have, as it would allow for further analysis of potential drug candidates.

As well as some additional modifications to estimate the configurational entropy, this would require a forcefield to be produced that can perform these predictions. Initially this forcefield is expected to be maintained separately to the structure prediction forcefield, however in the long term attempts would be made to unify the two and allow BUDE to accurately predict both structure and energetics from a single parameter set.

In order to tune a forcefield to predict these energies, the proposed approach would be to simply evaluate the pose which is already known to be correct from the reference data, and compare the resulting free energy to the measured energy in the data set. As this only involves a single pose calculation, this would be around 100,000 times faster that the method used to evaluate a forcefield for structure prediction.

This is an interesting property for the evaluation function to have, especially when considering the idea of tuning a single forcefield to predict both structure and energy concurrently. A possible method of exploiting this would be to evaluate large numbers of forcefields with this fast energy-only method, and then take the highest performing through to evaluate against the more thorough structure approach.

5.2.4 Modifications to Parameterisation

The results clearly show that while a large portion of the test set are being predicted accurately, several of the complexes are still failing to achieve RMSDs at all close to the satisfactory threshold. If further analysis confirms that this is caused by a shortcoming in the parameterisation approach rather than a deficiency in the forcefield parameters themselves, then alterations may be required to the parameterisation approach itself.

This may involve adding new atom types to the forcefield, or changing the functional form in which the parameters are applied. The forcefield would then need to be re-optimised using this framework to take into account these new modifications, and the resulting forcefield would hopefully exceed the accuracy that the current best forcefield exhibits.

5.2.5 Other Forcefields

This work has focused on optimising the free energy forcefield used by BUDE. It would also be of interest to be able to optimise forcefields used in other applications. The starting point for this would be to investigate the use of this

framework to optimise the forcefields provided in popular molecular modelling packages such as AMBER, CHARMM and GROMOS. This will likely require further implementation work to be performed, as most forcefields have a unique structure.

A Example Configuration File

```
# The type of optimiser to use
# Can be 'rand', 'hillclimber' or 'ga'
optimiser = ga

# The evaluation command to use
evaluator = evaluation/pose_rmsd --threshold 2

# The constraints enforced on forcefield parameters.
# <parameter>.stdev is the standard deviation of the normal
# distribution used when mutating values at random.
# <parameter>.inc is the increment which values are snapped to.

# Global parameters
constraints.hardness.min = 10
constraints.hardness.max = 30
constraints.hardness.inc = 1
constraints.hardness.stdev = 3

constraints.npnpDistance.min = 2
constraints.npnpDistance.max = 12
constraints.npnpDistance.inc = 1
constraints.npnpDistance.stdev = 2

constraints.nppDistance.min = 2
constraints.nppDistance.max = 4
constraints.nppDistance.inc = 0.5
constraints.nppDistance.stdev = 0.5

constraints.radiusScaling.min = 0.9
constraints.radiusScaling.max = 1.1
constraints.radiusScaling.inc = 0.1
constraints.radiusScaling.stdev = 0.05

# Per-atom parameter constraints are percentages of the starting value
constraints.hphbPotential.margin = 0.20
constraints.hphbPotential.inc = 0.05
constraints.hphbPotential.stdev = 0.05

constraints.esPotential.margin = 0.20
constraints.esPotential.inc = 0.05
constraints.esPotential.stdev = 0.05
```

```
############################
# Hill-Climber parameters #
############################

# Type can be 'steepest' or 'stochastic'
hillclimber.type = steepest

# The number of iterations to perform, or 0 to run until converged
hillclimber.numIterations = 0


################################
# Genetic Algorithm parameters #
################################

# The number of forcefields to keep in the population
ga.populationSize = 64

# The percentage of mutation to use whilst generating initial population
ga.pMutateInitial = 0.75

# The probability of mutating a new child
ga.pMutateChild = 0.1

# The rate at which to mutate a child's parameters (percentage)
ga.pMutateParam = 0.05

# The number of iterations to perform
ga.numIterations = 1024

# Whether to use crossover
ga.useCrossover = true

# The amount of bias in the crossover
ga.crossoverBias = 0.5
```

References

[1] A. Rahman and F. H. Stillinger, 'Molecular Dynamics Study of Liquid Water', *Journal of Chemical Physics*, vol. 55, no. 4, pp. 3336–3359, 1971

[2] A. D. MacKerell, 'Empircal Force Fields', *Computational Methods for Protein Structure Prediction and Modeling*, pp. 45–69, 2007

[3] I. E. Dzyaloshinskii, E. M. Lifshitz and L. P. Pitaevskii, 'The general theory of van der Walls forces', *Advances in Physics*, vol. 10, no. 38, pp. 165–209, 1961

[4] AMBER website
ambermd.org

[5] CHARMM website
www.charmm.org

[6] GROMOS website
www.gromos.net

[7] T. D. Kuntz, J. M. Blaney, S, J, Oatley, R. Langridg and T. E. Ferrin, 'A geometric approach to macromolecule-ligand interaction', *Journal of Molecular Biology*, vol. 161, no. 2, pp. 269–288, 1982

[8] S. M. Paul, D. S. Mytelka, C. T. Dunwiddie, C. C. Persinger, B. H. Munos, S. R. Lindborg and A. L. Schacht, 'How to improve R&D productivity: the pharmaceutical industry's grand challenge', *Nature Reviews Drug Discovery*, vol. 9, pp. 203–214, Mar. 2010

[9] G. L. Warren, C. W. Andrews, A. M. Capelli, B. Clarke, J. LaLonde, M. H. Lambert, M. Lindvall, N. Nevins, S. F. Semus, S. Senger, G. Tedesco, I. D. Wall, J. M. Woolven, C. E. Peishoff and M. S. Head, 'A critical assessment of docking programs and scoring functions', *Journal of Medical Chemistry*, vol. 49, no. 20, pp. 5912–5931, Oct. 2006

[10] N. Gibbs, A. R. Clarke and R. B. Sessions, 'Ab initio protein structure prediction using physicochemical potentials and a simplified off-lattice model', *Proteins: Structure, Function, and Bioinformatics*, vol 43. no. 2, pp. 186–202, May 2011

[11] S. McIntosh-Smith, T. Wilson, J, Crisp, A. A. Ibarra and R. B. Sessions, 'Energy-aware metric for benchmarking heterogeneous systems', *ACM SIGMETRICS Performance Evaluation Review*, vol. 32, no. 4, pp. 88–94, Mar 2011

[12] T. Lui, Y. Lin, X. Wen, R. N. Jorrisen and M. K. Gilson, 'BindingDB: a web-accessible database of experimentally determined protein-ligand binding affinities', *Nucleic Acids Research*, vol. 35, 2007

[13] M. Hülsmann, T. Köddermann, J. Vrabec and D. Reith, 'GROW: A gradient-based optimization workflow for the automated development of molecular models', *Computer Physics Communications*, vol. 181, no. 3, pp. 499–512, Mar. 2010

[14] J. Wang and P. A. Kollmann, 'Automatic parameterization of force field by systematic search and genetic algorithms', *Journal of Computational Chemistry*, vol. 22, no. 12, pp. 1219–1228, Jun. 2001

[15] P. Maurer, A. Laio, H. W. Hugosson, M. C. Colombo and U. Rothlisberger, 'Automated Parametrization of Biomolecular Force Fields from Quantum Mechanics/Molecular Mechanics (QM/MM) Simulations through Force Matching', *Journal of Chemical Theory and Computation*, vol. 3, pp 628–639, Jan. 2007

[16] R. Faller, H. Schmitz, O. Biermann and F. Müller-Plathe, 'Automatic Parameterization of Force Fields for Liquids by Simplex Optimization', *Journal of Computational Chemistry*, vol. 20, pp. 100–109, 1998

[17] D. H. Wolpert and W. G. Macready, 'No Free Lunch Theorems for Search', *Technical Report - Santa Fe Institute*, 1995

[18] D. H. Wolpert and W. G. Macready, 'No Free Lunch Theorems for Optimization, *IEEE Transactions On Evolutionary Computation*, vol. 1. no. 1, pp. 67–82, 1997

[19] J. H. Holland, 'Adaptation in Natural and Artificial Systems', *MIT Press*, 1975

[20] T. D. Gwiazda, 'Crossover for single-objective numerical optimization problems', *Genetic Algorithms Reference - Self-published E-book*, vol. 1, 2006

[21] F. Vavak and T. C. Fogarty, 'Comparison of Steady State and Generational Genetic Algorithms for Use in Nonstationary Environments', *IEEE International Conference on Evolutionary Computation*, pp. 192–195, May 1996

[22] J. Rees and G. J. Koehler, 'Learning genetic algorithm parameters using hidden Markov models', *European Journal of Operational Research*, vol. 175, no. 2, pp. 806–820, Dec. 2006

[23] M. A. Addicoat and Z. E. Brian, 'Using a Meta-GA for parametric optimization of simple gas in the computational chemistry domain', *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pp. 823-824, 2010

[24] H. Kargupta and D. E. Goldberg, 'SEARCH: An Alternative Perspective Toward Blackbox Optimization', *IEEE International Conference on Evolutionary Computation*, 1995

[25] M. L. Verdonk, J. C. Cole, M. J. Hartshorn, C. W. Murray and R. D. Taylor, 'Improved protein-ligand docking using GOLD', *Proteins: Structure, Function, and Bioinformatics*, vol. 52, no, 4, pp. 629–623, Sept. 2003

[26] T. El-Mihoub, A. A. Hopgood, L. Nolle and A. Battersby, 'Hybrid Genetic Algorithms: A Review', *Engineering Letters*, vol. 13, pp 124–137, 2006

[27] M. J. Hartshorn, M. L. Verdonk, G. Chessari, S. C. Brewerton, W. T. M. Mooij, P. N. Mortenson and C. W Murray, 'Diverse, High-Quality Test Set for the Validation of Protein-Ligand Docking Performance', *Journal of Medicinal Chemistry*, vol. 50, pp. 726–741, 2007