University of Southampton

Faculty of Engineering Science and Mathematics

School of Chemistry

The Development of Novel Scoring Methods for Virtual Screening

by

Luca Antonio Fenu

A thesis submitted for the qualification of

Doctor of Philosophy

June 2007

This thesis was submitted for examination in June, 2007.

To my father

University of Southampton

Faculty of Engineering Science and Mathematics

- School of Chemistry –

Doctor of Philosophy

The Development of Novel Scoring Methods for Virtual Screening

by Luca Antonio Fenu

Abstract

Protein Ligand docking is a valuable technique in the field of Structure Based Drug Design. One of its applications is to select one or few candidates from a set of thousands or million compounds, often known as Virtual Screening. The choice of the correct scoring function is of paramount importance , and currently available scoring function are most often implemented to optimise reproduction of known binding modes, or experimental free energies. The vast amount of information contained in the fact that most ligand do not, indeed bind to a given protein is discarded, building a considerable bias in these functions. The parameterisation method proposed in this work attempt to overcome said issue, by using a set of decoys among which the known ligand is hidden. The parameters of the scoring function are chosen by a Genetic Algorithm so to maximize the known ligand' ranking. The generality of the function is maintained by the constraint of simultaneously optimising several of these sets: the influence of the training set's size on the quality and robustness of the function is assessed. Also, the effect of using different treference poses for the known ligand, either experimental or re-generated by docking, is investigated. Finally, the immediate applicability of the method is shown by reparameterising commercially available scoring functions (GOLDSCORE, CHEMSCORE); a final attempt is made at creating scoring functions tailored to specific protein families', interpreting the results in terms of what is known of their ligan-protein interactions. The results show that such a function is able to discriminate between known actives and decoys, and would therefore be a valuable addition to the computational chemist bag of tricks to select those molecules able to turn into successful drugs.

The Development of Novel Scoring Methods for Virtual Screening

Table of Contents

1 DRUG DESIGN AND DEVELOPMENT 2

1.1 THERMODYNAMICS OF PROTEIN/LIGAND INTERACTION 2
1.2 RATIONAL DRUG DESIGN 3
1.3 COMPUTERS AND DRUG DESIGN 4
1.3.1 Visualisation 4
1.3.2 Simulation 5
1.3.3 Enumerating the innumerable 5
1.4 COMPUTER-AIDED DRUG DESIGN 6
1.5 STRUCTURE-BASED DRUG DESIGN 7
1.5.1 Enter Docking 7
1.6 DOCKING BASICS 8
1.7 SEARCH AND SELECTION 9
1.7.1 Search Engines 9
1.8 SCORING 10
1.9 DOCK 11
1.10 GOLD 12
1.11 GLIDE 13
1.12 SUMMARY 13

2 SCORING 14

2.1 INTRODUCTION 14
2.2 HOW TO SCORE 14
2.3 FORCE FIELD SCORING FUNCTIONS 16
2.3.1 Problems and Solutions 16
2.4 EMPIRICAL SCORING FUNCTIONS 17
2.4.1 Problems and Solutions 18
2.4.2 Recent Advances 19
2.5 KNOWLEDGE-BASED SCORING FUNCTIONS 21
2.5.1 Recent Advances 24
2.6 E PLURIBUS UNUM 25
2.6.1 Recent Advances 26
2.7 COMPARISONS BETWEEN SCORING FUNCTIONS 27
2.8 NEW TEST AND TRAINING SETS 28
2.9 CONCLUSIONS 29

3 THE PDB AND OTHER DATABASES 32

3.1 INTRODUCTION 32
3.2 PROTEIN DATABASES 34
3.2.1 Brookhaven Protein Databank – PDB 34
3.2.2 PDBSum 35
3.2.3 Giving 'Structure' to Structural Databases 36
3.2.4 SCOP 36
3.3 SMALL MOLECULES DATABASES 39
3.3.1 Available Chemical Directory- ACD 39
3.3.2 World Drug Index – WDI 39
3.3.3 National Cancer Institute NCI-Open3dMol 39
3.4 AD HOC DATABASES 40
3.4.1 CCDC/Astex protein dataset 40
3.4.2 PDBbind 41
3.5 CONCLUSIONS 41

4 DATA PREPARATION 42

4.1 INTRODUCTION 42
4.2 LIGAND DATASET(S) 43
4.2.1 NCI dataset 45

4.2.2 Rognan's dataset 46
4.3 PROTEIN DATASET 47
4.3.1 Further reduction of the set 48
4.3.2 Classification by family 49
4.3.3 Subset generation 52
4.4 DOCKING EXPERIMENTS 55
4.4.1 Why Use GOLD? 55
4.4.2 Why Use Distributed Computing / Why Condor? 56
4.4.3 How does distributed computing work? 57
4.5 DOCKING SETUP 59
4.6 DOCKING RUNS 60
4.6.1 Specifics of Gold on Condor 61
4.7 CONCLUSIONS 62

5 SCORING FUNCTION DEVELOPMENT 63

5.1 INTRODUCTION 63
5.2 FRAMEWORK STRUCTURE 63
5.2.1 Why object oriented? Why C++? 64
5.2.2 Basic Data Structure 66
5.2.3 Universe 67
5.2.4 Atom 67
5.2.5 Bond 67
5.2.6 AtomList 68
5.2.7 Residues 68
5.2.8 BondList 69
5.2.9 Molecules 69
5.2.10 Ancillary classes: i.e. Grid, Dictionary, etc. 69
5.3 SCORING FUNCTION IMPLEMENTATION 70
5.3.1 Hydrogen Bonds 70
5.3.2 Hydrogen Bonds: Penalties 72
5.3.3 Surface calculations 74
5.3.4 Electrostatics and Solvation 76
5.3.5 Van der Waals 80
5.4 SCORING FUNCTION USE 81
5.5 PARAMETERISATION PROGRAM 84
5.6 INTRODUCTION TO GENETIC ALGORITHMS 86
5.6.1 Terminology 86
5.7 CROSSOVER AND MUTATION 87
5.8 CODING OF THE GA 89
5.8.1 GA Implementation Details 89
5.8.2 Getting the right sign 90
5.8.3 GA internal parameters 91
5.8.4 Fitness Function 91
5.8.5 Fitness Scaling 92
5.9 CONCLUSIONS 93

6 SCORING FUNCTION PARAMETERISATION 94

6.1 PARAMETERISATION PROCEDURE 94
6.2 CORRELATION BETWEEN TERMS 94
6.2.1 COLO: references and decoys 100
6.3 SELECTING THE RIGHT TERMS 103
6.3.1 Angular distance between solutions 111
6.3.2 Good performers and bad performers 115
6.3.3 Are bad functions really that bad? 119
6.3.4 How Genetic Algorithms really work 125
6.3.5 Checking the good combinations 128
6.3.6 GA versus Exhaustive Sampling 131
6.4 DIFFERENT FUNCTIONS FOR DIFFERENT REFERENCES 132
6.4.1 Closest and Goldest 133
6.4.2 Crystallographically Observed Ligand Orientations simplexed in GOLDSCORE 137

6.5 CONCLUSIONS 138

7 SCORING FUNCTION TESTING 140

7.1 TESTING THE ROBUSTNESS 140
7.1.1 Nomenclature 140
7.1.2 27-complex sets 141
7.1.3 54 protein sets 145
7.1.4 81 protein sets 146
7.2 TESTING THE PERFORMANCES 147
7.2.1 Performances of the 27P parameterised functions 149
7.2.2 Testing the 54p 152
7.2.3 Testing the 81p 156
7.3 TRAINED VERSUS COLO, TESTED VERSUS GOLDEST 158
7.3.1 Training and testing against Goldest 161
7.4 CONCLUSIONS 166

8 FURTHER PARAMETERISATION AND TESTING 168

8.1 INTRODUCTION 168
8.2 GOLDSCORE+CHEMSCORE FUNCTIONS 169
8.2.1 Results 170
8.2.2 Training on subsets 175
8.2.3 Conclusions 177
8.3 SINGLE FAMILY PARAMETERISATION 177
8.4 FAMILIES AND SUBSETS 177
8.5 TRAINING 178
8.5.1 Choose your own terms 178
8.5.2 Choose your coefficients 181
8.6 CONCLUSIONS 186

9 CONCLUSIONS AND FUTURE WORK 188

9.1 SUMMARY OF CONCLUSIONS 188
9.2 FUTURE WORK 189
9.3 FINAL REMARKS 189

***REFERENCES* 190**

List of Figures

Figure 3-1: An example output of PDBSum for PDB entry 1AAW. On the left, the sequence with secondary structure and some interaction flags assigned; on the right, a LigPlot depiction of the ligand in the binding pocket. 35
Figure 4-1: An example figure of clustering; a series of data-points (*e.g.*, molecules represented as descriptors) is partitioned in four diverse clusters (plus a singleton, in blue). A single or reduced set of representatives can be then chosen for each cluster. 44
Figure 6-1: Correlation plot for the two terms GOLDEXVDV and 6/9ExtVdW, for the 108 Crystallographic reference poses (COLOs) - *r=-0.02*. 100
Figure 6-2: Correlation plot for the two terms GOLDEXVDV and 4/8ExtVdW, for the 108 Crystallographic reference poses (COLOs) - *r=-0.67*. 101
Figure 6-3: Correlation plot for the two terms GOLDHBOND and GoodHBond, for the 108 Crystallographic reference poses (COLOs) - *r=-0.69*. 101
Figure 6-4: GBSA and Coulombic with 8Å cutoff are strongly anti-correlated - *r=-0.91*. 102
Figure 6-5: Correlation Plot between the two terms GBSA and Cl8AnGBSA (sum of Coul8A and GBSA) – *r = 1*. 103
Figure 6-6: Graphs showing the solutions from Run A (purple square) and B (yellow triangle) projected within the 2-sphere (circle) of all possible solutions. Their normalised solution (red circle) sits on the blue circle line. 110
Figure 6-7: Graphical depiction of the angle between two solutions A and B. 111
Figure 6-8: The graphs shown illustrate the convergence of a GA towards its final solution. The population's distribution is shown at the 0^{th} (upper left), 150^{th} (upper right), 300^{th} (lower left) and 600^{th} generation (lower right). 113
Figure 6-9: Plot showing a measure of 'enrichment', namely how many COLOs have been found as a function of the number of decoys selected out of a ranked list. 120
Figure 6-10: Reference ligands of the 22 entries ranked first by the scoring function detailed in Table 6-16. 122
Figure 6-11: The ranking power variation with the varying contribution of Coul8A'n'GBSA. The small square represents, in greater detail, the region close to the zero contribution. 124
Figure 6-12: These graphs show how the solution chosen by the GA for the combination of terms detailed in Table 6-18 actually sits in a minimum position for the fitness function. The curves show the fitness trends with varying contributions along the given axis. The black crosses in the various graphs correspond to the computed optimum values, the red crosshairs are instead the values selected by the GA. All are almost overlapping. 125
Figure 6-13: Representation of the three and four terms solutions in the terms space. Similar functions generate similar shapes. 129
Figure 6-14: The ranking power variation with the varying contribution of GOLDITORS. Black is the optimal three-term solution. Coefficients for the red line are taken from the optimal four-term solution. The cross is where the GA has converged – that is, the absolute minimum. 130
Figure 7-1: Best solutions for the combination LigHPhobB – 4/8 ExtVdW when parameterised against COLO reference poses. 143
Figure 7-2:The ability to first-rank a training set's truly active compound increases with the number of terms used, but after four terms, the performance on the test set lags behind. Results shown are for functions trained using COLOs as references. 148
Figure 7-3: Performance on the four 27 proteins COLO sets of the best function parameterised on set 27-A (COLO) with various number of terms. 152
Figure 7-4: Performances on function parameterised on COLO 54-A on its own set and other 54-protein sets. 155
Figure 7-5: Performance on the four 27 protein Goldest sets of the best function parameterised on set 27-A (COLO) with various number of terms. 160
Figure 7-6: Ranking performances of functions parameterised on set 54-A using COLO, when applied to Closest and Goldest sets 54-A and B. 161
Figure 8-1: Plot showing the projection on the CS Hbond / CS Clash space of the original and normalised solutions for *COLO*'s best function, and *GOLDEST* second best. 173
Figure 8-2: Plots showing the best 2-term and 4-term solutions in the CHEMSCORE / CS Lipo space: left, when all components for the 4-terms are accounted; right, neglecting contributions from components other than those represented. 176

Figure 8-3: on the left side, three of the lipocalins used in the work, showing only one or two Hydrogen bonds per ligand; on the right side, an oxido-reductase (1aoe), a protease (1apt), and an exopeptidase (2ctc) - interaction diagrams obtained with MOE (Chemical Computing Group). 185

List of Tables

Table 4-1: the PDB entries corresponding to the complexes in the CCDC/Astex dataset, divided in the subsets defined by its authors. 49
Table 4-2: Number of complexes present in our set for each SCOP superfamily represented. 51
Table 4-3: Protein superfamilies subject of single-family scoring function parameterisation (shown in Section Error! Reference source not found.): for each family considered, the related PDB entries are shown. 52
Table 4-4: The four 54-protein sets by PDB entries. 53
Table 4-5: The four 54-protein sets: percentile overlaps between them. Sets A and C (and B - D) overlap 32 out of 54 entries. Sets A and D (and B - C), 22 out of 54. 54
Table 4-6: The four 81-protein sets: PDB entries. 54
Table 4-7: The four 81-protein sets: percentile overlaps between them. 55
Table 5-1: How the Atomlist's ElectrStat method calls the corresponding atom's method in a cascade. 68
Table 5-2: a general example of input file for the scoring function parameterisation. 83
Table 5-3: a more specific example of input file for the scoring parameterisation. 83
Table 6-1: The scoring terms' numerical IDs, names and brief descriptions (the aliases in the VdW terms from 19 to 24 are the actual name in the code). 95
Table 6-2: Correlation matrix for the 28 terms shown in Table 6-1: above the diagonal the values of the correlation coefficient r relative to a random 1% selection of decoy poses, and below it for the COLO references: highlighted in light yellow are values larger than 0.3 (or smaller than -0.3); in orange are values higher than 0.7, or lower than -0.7. 98
Table 6-3: Determination matrix for the 28 terms shown in Table 6-1: above the diagonal the values of the determination coefficient r^2 relative to a random 1% selection of decoy poses, and below it for the COLO references: highlighted in light yellow are values larger than 0.3 (or smaller than -0.3); in orange are values higher than 0.7, or lower than -0.7. 99
Table 6-4: Best performers among the two-term combinations: An example outcome from the first ten combinations selected from two-term runs, parameterised against COLOs. 108
Table 6-5: Best performers for the three best runs of the two-term combinations. 109
Table 6-6: All three solutions in Table 6-5 correspond to this scaled solution. 110
Table 6-7: Data corresponding to the angular spread of individuals from three example GA runs. 112
Table 6-8: Best solution (normalised) from the three GA runs whose average angle variation along time is shown in Table 6-7; the final solutions are very, very close. 113
Table 6-9: First and last ten rankings for the LigIPhobB + 4/8ExtVdW combination. 114
Table 6-10 Full COLO's rankings for the combination of terms LigIPhobB - 4/8ExtVdW. 115
Table 6-11 Best performers between the four-term combinations: refer to Table 6-1 for translation between numerical IDs and terms' names. 115
Table 6-12: Solution for the three runs of the best four-term combination selected by our protocol. Ranking performances are near-identical. 116
Table 6-13: First and last ten rankings for the LigHPhobS + 4/8ExtVdW + GOLDHBOND + GOLDITORS combination. 117
Table 6-14: a mediocre performer in the four-term combinations. 118
Table 6-15: Best solutions for each run presented in Table 6-14. 118
Table 6-16: The worst performer amongst the four-term combinations. 119
Table 6-17: Proteins for which the scoring function detailed in Table 6-16 ranks the COLOs first: those in bold and above the horizontal line are parts of subsets for which we have parameterised single-family scoring functions; the rest are not. 121
Table 6-18: Results for combination 1-16-19 (grey background) and its *daughter* combination 1-12-16-19 (pink background). The *mother*'s performances (*scorerank*) are better. 123
Table 6-19: Coefficient values for the best two terms selected by the GA parameterisation routine when using three different reference structures: COLO is the experimentally observed ligand orientation; Closest is the re-docked pose most similar to COLO, Goldest is the pose with the highest GOLDSCORE value. 134
Table 6-20: performances of functions trained on different kind of references against each other training set – the functions used are presented in Table 6-19. 135
Table 6-21: Performances of the best 2-,3-,4-,5-terms functions trained on the COLO, *Goldest* and *Closest* training set. 139

Table 7-1: Best performers between the 2-term combinations trained on four different 27 strong complex sets: the performance shown for each combination is on its own training set. Only one result per combination is shown (i.e. 14-19/C but not A or B). 141
Table 7-2: Optimal solutions for the combination *LigHPhobB – 4/8ExtVdW* when trained on the full training set and its four subsets. 142
Table 7-3: Angles relative to the best 2-term combinations for the various 27 complexes training subsets. 144
Table 7-4: Best 2-term functions trained on the four different 81-protein sets. 147
Table 7-5: The best 4-term functions for each set – using COLOs as references. Different colours indicate different interaction classes 150
Table 7-6: Results for the functions detailed in Table 7-5 on the various sets – COLOs as references. 151
Table 7-7: Results for functions parameterised against 54 complexes – COLOs as references. 153
Table 7-8: Best 4-term functions for each 54-protein set – using COLOs as references. As before, different colours flag different classes. 154
Table 7-9: Results for the functions detailed in Table 7-8 on the various sets – using COLOs as references. In *bold+italic*, the performance on the test against which each function has been parameterised. 154
Table 7-10: Best 2-terms functions for each 81-protein set – using COLOs as references. 157
Table 7-11:Two term functions for 81-p sets. The last column shows the performance of the function trained against set *i* on its own complement, *i.e.* the remainder of the full 108 complex training set after selection. 157
Table 7-12: Best 4-terms functions for each 81-protein set. Again, each set independently chooses the same 4 terms. 157
Table 7-13: Ranking performances for 4-term combinations trained on the 81-protein sets. The last column (Complement) shows their performances against the complement sets. 158
Table 7-14: 4-term functions trained against the various 81-protein sets: performances against complement set using COLOs and *Goldest* poses. 159
Table 7-15: Best 2-term functions for each 27-protein set, trained using *Goldest* as references. 162
Table 7-16: Results for the functions detailed in Table 7-15 on the various sets. In *Bold+Italic*, the performance on the test against which every function has been parameterised (intrinsic performances of the functions shown in Table 7-15. 163
Table 7-17: Best 4-term functions parameterised on the four 27-protein sets, using *Goldest* as references. 164
Table 7-18: Results for the functions detailed in Table 7-17 on the various sets. In *Bold+Italic*, the performance on the test against which every function has been parameterised. 165
Table 7-19: Percentage average ranking on all *Goldest* 27-protein sets of two 2-term functions, respectively best and second best from training set 27-A. 165
Table 8-1: Performances upon re-parameterisation of scoring functions term-by-term: the value "Before Retraining" for GS+CS assumes an equal weight for both – performances are expressed as percentage average rankings of reference ligand poses (COLOs). 169
Table 8-2: Performances of the best 2-term function trained on the full *COLO* set, when tested on the same and *Goldest* references. 170
Table 8-3: Performance of the second best 2-term function trained on the full *COLO* set, when tested on the same and *Goldest* references. 170
Table 8-4: Performances of the best 2-term function trained on the full *Goldest* set, when tested on the *COLO*s and *Goldest* references. 171
Table 8-5: Performances of the second best 2-term function trained on *Goldest* set, when tested on the *COLO*s and *Goldest* references. 172
Table 8-6: Best 3-term function trained on the full *COLO* set – its performances in ranking of both *COLO*s and *Goldest* reference poses are shown. 172
Table 8-7: Best 4-term function trained on *COLO*s – its performances in ranking both *COLO*s and *Goldest* reference poses are shown. 174
Table 8-8: Best 4-terms function trained on *Goldest* – its performances in ranking both *COLO*s and *Goldest* reference poses are shown. 174
Table 8-9: *Correct* best 4-term function trained on *Goldest* – its performances in ranking both *COLO*s and *Goldest* reference poses are shown. 175
Table 8-10: Best 2-term scoring function trained against set 27-A, and its performance on sets 27-A to D. 175

Table 8-11: Best 4-term scoring function trained against set 27-A, and its performance on sets 27-A to D. 176
Table 8-12: Best two-term functions selected by the GA. Reference poses are COLOs. Term names and coefficient values are shown for each protein's superfamily. The first line refers to the equivalent result for the full 108 protein set. 179
Table 8-13: Best three-term functions selected by the GA. Reference poses are COLOs. Terms names and coefficient values are shown for each protein's superfamily considered. First line refers to the equivalent result for the full 108 protein set. 180
Table 8-14: Table showing the weights chosen (and performances obtained) for the CHEMSCORE terms trained on nine different families of proteins. 182
Table 8-15: Average CS IIB score for both decoys and reference poses in lipocalins, kinases, immunoglobulins and serine protease protein families – in black, absolute values – red or blue, values scaled by the coefficient obtained from Table 8-14 (*e.g.* 1.06*16.5=17.49). 183

List of Accompanying material

This being a thesis in the subject of Computational Chemistry, various software programs developed by the authors are to be considered integral part of it. However, owing to their length, they are not directly inserted in the thesis' main body. Therefore, an electronic copy of all original programs discussed in this thesis can be requested to the author at the e-mail address: luca.fenu@gmail.com.

Acknowledgements

This is my attempt at a PhD thesis. It's taken so long to complete, that I've racked up a pretty long list of person to thank... However, since I would surely offend someone leaving him or her out, I am not going to name anyone explicitly, and I will refer to them by their role in my life: so let's start.

I want to thank my supervisors, pasts and presents, from academia and industry, for introducing me to such a fascinating field of science, at the border between chemistry, physics and biology, and with both solid theories(!) and strong applications($). Looking back to when I was a child, I send a big thank to my godfather, who sparked my scientific interest, and to my parents who gave me the opportunities and support necessary to make it grow into a fire. And a wink (and a thumb up) to my friends and colleagues, old and new, who kept me company and believed that I could pull it off. A big thank to all the writers who churned out the books I love, from science to science fiction. Some of you played more than one role, as mentors and friends, or supervisors and writers. In the case, double thanks.

Kudos to all of you, and please keep me company. This isn't the end, rather the beginning :-).

1 Drug Design and Development

Drug Design is one of chemistry's most important practical applications. Before the rational application of cheminformatics and receptor structures was well developed, new drugs were usually found by trial and error on chemical compounds isolated from plants, whereas nowadays drugs are more properly ***'designed'*** so as to play their particular role within the target organism, which could be the human body, that of another animal, or a bacteria whose growth must be inhibited. Discovery of natural bio-active compounds still plays an important role, with 60% of drugs traceable to naturally occurring compounds[1]. However, this figure falls to 25% if mimics of natural compounds such as serotonin and histamine, with significative synthetic modification, are considered as 'synthetics'. Either way, there is a clear and strong effort towards newly generated (***ex novo***) drugs.

1.1 Thermodynamics of Protein/Ligand interaction

When looking for a new drug, we are primarily interested in a molecule with a very high "affinity" for the target. Chemically, the affinity is measured as the "inhibition" constant, the inverse of the "dissociation" constant associated with the equilibrium:

$$P_{aq} + L_{aq} \rightleftharpoons PL_{aq}$$
$$K_D = \frac{[P][L]}{[PL]} \tag{1.1}$$

This inhibition constant can be related to the free energy of binding for the ligand-protein system by the following equation:

$$\Delta G^0 = -RT \ln K_D \tag{1.2}$$

Here, K_D stands for dissociation constant, since the reaction is usually considered in the reverse direction. Its inverse value is often referred to as Inhibition Constant, K_i.

Tipically, K_i ranges from 10^{-2} to 10^{-12} l/mol, with associated energy in the range -10 to -70 kJ/mol at standard condition.

Using the Gibbs-Helmholtz equation (not shown), the free energy of binding can be decomposed into its components:

$$\Delta G = \Delta H - T\Delta S \tag{1.3}$$

Usually, the process is enthalpically favoured, while the entropy change usually works against binding, as degrees of freedom (translational, and some orientational/vibrational) are frozen during the process, resulting in a diminished entropy for the system. It is often observed that the enthalpy contribution is favourable to binding, as water is displaced from the active site, and a more hydrophobic ligand is placed in it. As a general rule, water displaced usually does form tighter bonds with bulk water than with the protein's surface. The entropy change is usually unfavourable since, for the ligand to get into the confined space, it has to forfeit its rotational and some vibrational modes, which can be partially offset by those gained back by water freed. The balance between the terms is often tight, and being temperature dependent it can switch between positive and negative depending on the environment[2, 3].

1.2 Rational Drug design

Finding new, effective drugs is difficult. When only looking at natural compounds, finding a new lead molecule is unpredictable at best. The new paradigm, which tries to go beyond this and develop a step-wise approach to lead identification, goes under the name of ***'rational'*** drug design[4, 5].

Rational drug design is a more focussed approach, which uses information about the structure of a drug receptor or one of its natural ligands to identify or create candidate drugs. The other stages of development are left unchanged, so that the molecules generated are then tested against the said targets (***in vitro***), and variations on working themes are tried to further improve potency and/or selectivity. Further down the line, biological (***in vivo***) and clinical tests assess the effective therapeutic activity of the drug candidates. Similar ***in vitro/in vivo*** tests also assess problems related to Drug

Methabolism and PharmacoKinetics (DM-PK), often identified by the acronym ADME-Tox (from Adsorption, Distribution, Metabolism, Elimination, and Toxicity)[6-10]. *In silico* predictions play their role in this field too[11].

Synthesizing all these compounds can be really expensive, and requires skilled and highly trained organic and medicinal chemists. Once synthesized and tested, both successes and failures end up stored in huge repositories - compound collections kept at hand should some suitable target be identified.

Still, especially for the first few phases of drug design, when we try many different variations on a common theme, the selection process is very wasteful. You may spend months to synthesize a class of compounds, and it all goes to waste because while it works wonderfully ***in vitro***, it turns out that ***in vivo*** it inhibits other essential functions. Alternatively, because of the modifications made to improve potency, it will not pass the blood/brain barrier to get where it is needed. It is therefore unsurprising that industries have been looking to minimise costs.

1.3 Computers and Drug Design

Computers can help to uncover new routes to drug design. As we are going to show now, they play an increasingly important role in modern drug design, not just by helping to classify and order huge databases of compounds, but during the screening processes itself. Unfortunately, there is not enough space in this thesis to detail all these applications, and we only have time to discuss some briefly.

"***Non ragioniam di lor, ma guarda e passa***" [12 (1)]

1.3.1 Visualisation

Computers can help us to visualise complex interactions using molecular graphics. This allows us to grasp three-dimensional structures much more easily than previous chemists, limited by what they could visualise in their mind. But visualization is not, by any means, the only use of computers in drug design. It has, in fact, come quite late on the scene, with the development of real-time 3D and video-processing capabilities on personal workstations.

---

[1] "Don't care about them, just have a quick glance and move on", as Virgil told Dante in Hell.

1.3.2 Simulation

Computers can also compute physical values: techniques such as molecular dynamics (MD) and Monte Carlo (MC)[13, 14] allow us to perform simulations of the behaviour of complex systems starting from first principles (Newton's law, for MD, Boltzmann distribution for MC). Such simulation methods act as a bridge between the macroscopic and microscopic visions of the world: they provide microscopic details on phenomena otherwise unobservable, such as the uncoiling of a DNA strand in water. At the same time, with the combined use of tools sourced from statistical mechanics[15], they enable us to retrieve experimentally accessible quantities, such as the free energy change associated with the phenomenon under consideration. Depending on the complexity of the problem to be solved, and on the accuracy of the physical value required, simulation methods can rely on Quantum Mechanics[16], or use a simplified approach, such as in Chemoinformatics[17] and other fields relevant to structure-based drug discovery[18].

1.3.3 Enumerating the innumerable

As said before, a common trend in drug design is to screen more and more compounds extracted from ***in house*** or commercially available databases, to find some promising leads to be brought forward in the development process. In this, computers can play an important role, for example by suggesting modifications to a lead which would improve its value as a drug. Thanks to empirical methods developed using (part of) the existing ***pharmacopoeia*** as knowledge base, we are able to estimate the influence of various features present in a molecule upon its interaction with the target protein, or its pharmacokinetic/pharmacodynamic (often written as PK/PD)[11] properties, equally important for its use as a drug. Using similar techniques, new molecules can also be generated ***ex-novo*** without having an empirical starting point, thereby enormously enlarging the region of chemical space accessible during the first phases of drug design.

The whole body of techniques used in this field goes under the name of **Computer-Aided Drug Design** (often abbreviated as **CADD**), which we are going to discuss in more detail in the next few pages.

1.4 Computer-Aided Drug Design

In Computer-Aided Drug Design, a model is built that reproduces the molecule under consideration, either isolated or in its surrounding environment, be this solvent or a receptor. The model does this with different accuracy depending on our aims, sometimes using quantum-mechanics, or alternatively with approaches well below atomistic detail, such as in meso-scale simulations[19, 20]. So, to what level of detail must we go to simulate accurately the interactions between proteins and drugs?

Interactions between bio-molecules, often termed as ***molecular recognition***, are essentially a matter of electrostatic, expressed as the shape of the molecules involved and charges they bear on their surfaces[21]. Protein and ligand must fit one into each other, and have complementary shapes and properties, according to the lock/key paradigm proposed by Fischer in 1894[22]; a lock which can often adapt itself to better fit the key, in the phenomenon known as ***induced fit***[23-25].

Once we know which protein is our target, two ways of proceeding are possible: first, if a few known ligands are available, we can compare, or ***screen***, putative new ligands against the known, looking for similarities in shape, chemical behaviour and properties. Alternatively, we can use information implicitly contained in the target's structure to achieve the same results. After all, the key and lock all bear the same information, albeit in complementary fashion. In fact, using the protein structure, when available, could unlock new modes of action from a limited sampling of known ligands, especially if they all belong to related classes. As an example, ATP-competitive binding is not the only way to inhibit a kinase[26].

The first approach is called ligand-based drug design, and it is usually pursued when a structure for the target protein is not present [17]. Unfortunately, this is the case for many interesting medicinal targets, such as G-protein coupled receptors (GPCRs)[27], cytochrome P450 (CYP450s)[28] and other membrane-bound proteins, for which crystal structures are very difficult to obtain. Ligand-based are usually very quick, since they only have to compute properties for small organic molecules with average molecular weights of less than 500 Daltons. They can be ideally used as a pre-filtering for large databases even when the target structure is known. As a general rule, when large structures such as proteins or large number of ligands are involved, the model is limited

to classical concepts. Empirical models are parameterised to fit more refined QM models or experimental data. An example are atomic partial charges used to model the Coulomb interaction, such as AM1-BCC[29, 30], where Amber atom-types and semi-empirical Bond Charge Corrections (BCC) are used to assign charges which are designed to agree with more accurate QM/MM derived HF/6-31G* RESP results[31, 32].

1.5 Structure-based Drug Design

Structure-based drug design (SBDD) is based, as the name says, on the 3D structure of the protein. Such a structure is obtained through crystallization, possibly with related cofactors and substrates, followed by X-ray diffraction. The structure can also be obtained through solid-state or solution-phase NMR. An important requirement to ascertain the quality of a crystallised enzyme is that it should be able, even in the crystallised form, to perform its biological action. This ensures that what we are resolving is the actual biological conformation, and not some weirdly denatured form arising due to crystal packing effects. When the protein has not been crystallised, we can obtain a theoretical model based on homology modelling[33, 34], a technique which builds a structure based on those of other proteins with a high degree of overlap (homology) with the primary structure; from similarities in the amino-acids sequence we infer similarity in the further structural levels.

Protein structures are often deposited in public domain databases such as the Brookhaven Protein Databank (PDB)[35, 36]. Other focused protein databases are described later (section 3.2). Pharmaceutical industries will also have proprietary structures for targets of interest, unpublished and undisclosed to the general public.

1.5.1 Enter Docking

Structure-based drug design (SBDD) exploits various techniques, ranging from methods of general applicability such as molecular dynamics (MD) to extract the quantities we need, to other more SBDD-specific, such as protein-ligand docking [21, 37], The foundations of this particular method are based upon the concept of complementarity between ligand and target. Based on the ***lock & key*** paradigm, we attempt to find a ligand (key) which fits the active site for a given protein well enough to offer a useful starting point (hit). Because of its central role in this study and the whole SBDD field, we will now talk in detail about docking.

1.6 Docking Basics

The term "docking" reminds us of a ship slowly entering a marine port, and anchoring itself within a narrow space between concrete banks. In the case of protein-ligand docking, the ligand plays the role of the ship, while the 'docks' are the exposed surface of the protein in the active site.

As described by Sotriffer et al.[21], docking is a computational procedure to investigate the binding between potential ligands and a macromolecular target (a protein) whose structure is known. Accordingly, it plays an essential role in structure-based drug design. Any automated docking program relies on a search algorithm to explore potential binding modes, and a scoring method that translates computationally determined protein-ligand interactions into approximate estimates of binding affinity. Accurate scoring is essential for both the determination of the correct binding geometry and the ranking of different ligands with respect to their putative affinity.

The "docking problem" is easily put like this: given the structure of a ligand and a protein, and limiting ourselves to the case of reversible, non-covalent binding, the task is to predict the structure of the resulting complex. An accurate measure for the relative free-energy of the binding process, as expressed by equations (1.1) through (1.3), is highly desirable.

The "ligand" can, in fact, be another protein, such as in cases where multi-domain assemblies are studied. However, owing to the diversity of the phenomena, we will limit ourselves to complexes between a protein and small organic molecules. Since the native geometry of the complex can generally be assumed to reflect the global minimum of the binding free energy, docking is in fact a free energy minimisation problem, albeit a highly complex one. Regardless of the nature of the two interacting partners, computational docking always requires two components, which may briefly characterised as "searching" and "scoring". Some time will be spent now on detailing exactly these two phases, presenting known problems and recent trends to solve them. Finally, a rapid overview of some (either freely or commercially available) docking programs will be presented, which, in our opinion, represent the current state of the art.

1.7 Search and Selection

The first stage of docking is the so called 'sampling' of the configurational space of the protein/ligand complex. The receptor structure is generally kept mostly rigid, to reduce the dimensionality of the search space. Therefore the sampling is actually executed in the ligand's conformational space, but within the constraints imposed by the receptor's shape. The sampling can be performed in any of a number of ways, but all may be decomposed into a two-step approach: First, the generation of a ligand conformation and its placement within the active site; next, the 'constraints' are enforced by the use of a scoring function, which takes on the role of discarding poses causing clashes between protein and ligand, or other unlikely arrangements of the two.

Scoring functions, therefore, enter both stages of docking, that being described now, and the next, aptly named 'scoring'. Their role is, however, different in the two phases. During the sampling, those configurations most likely to reflect the true protein-ligand binding geometry need to be selected from the many thousands of trial conformations. This is usually achieved by means of a combination of shape complementarity and directional (hydrogen bonding and such) terms, with little regard to whether or not the scores correlate with free energies of binding. These functions will tend to disregard refined measures of solvation or entropy, since they take too long and their efficacy in determining a realistic binding pose is unclear [38-40].

1.7.1 Search Engines

Generally, scoring functions must respect the physical constraints of real protein-ligand interaction, whereas the sampling engines used for the conformational searching can vary significantly. New conformations can be generated randomly, or exhaustively, or with a taboo search [37]. Different internal representations of the molecules can be used, from bit-strings to real-3D structures. Subsequent to the score evaluation, the selection and improvement of good performers can be done *via* Genetic Algorithms, whether Lamarckian[41] or not[40, 42, 43], or purpose-built "***filters***", as in Glide[44, 45], or again with computationally expensive methods such as MonteCarlo "***annealing***"[46]. Hybrid approaches of every kind are possible, and there are always new docking programs coming out, improving on existing algorithms or implementing new ones.

1.7.1.1 Receptor and ligand flexibility

As said before, the receptor structure is kept mostly rigid, to avoid having to sample its huge conformational space. In some cases, an exception is made for a few side chains and/or H-Bond donor/acceptor groups within the putative active site. GOLD [40, 42, 43] allows restricted receptor movement of the polar hydrogen atoms. This allows for a limited degree of induced fit of the active site as a response to the ligand binding, at the price of an increased conformational space to be explored. Hopefully, the possibility of better poses offsets the increase in computational cost. Some very quick programs (FRED[47]) still rely on the rigid ligand approach, where multiple ligand conformations, if needed, are generated beforehand by specialised programs[48], and then rigidly docked into the active site. This makes for very quick docking, but leaves open the criticism of whether the preferred conformation(s) ***in vacuo*** (or water) is still favoured within a very different environment such as a protein. This can however be partially addressed by the use of a scoring function to rescore the poses generated.

1.7.1.2 Incremental construction

Ligands are not even always docked 'as a whole'; some programs break them down into fragments, dock them quickly, thanks once again to the reduced complexity of the search space, then join the fragments to re-generate the input ligand in the new conformation. Others instead, anchor a fragment of the ligand to the protein surface, then subsequently attach the remaining parts[39]. Of course, with a little more flexibility, the same approach can be adapted to generate *de-novo* ligands, using a set of "functional" fragments.

1.8 Scoring

Scoring is the second phase of docking. Since scoring functions are used during the sampling too, it partially overlaps with the first. However, having selected your best poses, a list of estimates of their binding free energies is needed. This is a task for a scoring function, and often one quite different from that used to select the poses at first.

In the first phase, to explore as much configurational space as possible, we want a quick scoring function, forgiving minor clashes. For the scoring part, instead, we want a harsher function, because 'unlikely' configurations, useful before to overcome high energy states in the configurational landscape, are no longer needed. We now want low-

energy, physically realistic conformations, and the scoring function will have to act accordingly.

So, a van der Waals term, instead of a close contact count, and more elaborate electrostatic treatments may be necessary. Ligands with high specificity often form a network of hydrogen-bond interactions with the receptor surface and the water, so hydrogen bond terms are very important. Also, the conformation of the ligand itself must not be strained, unless adequately compensated by an increase in the favourable intermolecular interactions; a torsional potential is therefore often used. No matter how elaborate, this function still has to be quick, of the order of one second per ligand, so elaborate entropy considerations are ruled out once again, and only the conformational entropy is considered, with rough counters such as number of rotatable bonds constrained by the receptor's proximity.

Some particular cases of search engines and scoring functions will be introduced now *via* the description of some docking programs. We instead leave a more detailed treatment of scoring to the next chapter.

1.9 DOCK

DOCK is the first general purpose protein/ligand docking program, built by Kuntz and coworkers in the early eighties[39]. It is also freely available to academics, and this helped it to sustain more than twenty years of particularly fierce competition with commercial programs. Now at version 5, DOCK offer docking of rigid ligands, or fragment building, and scoring *via* three different built-in functions plus many more "***unofficial***" ones often available upon request from the developers. Example of this class is the scoring function resulting from the work of Smith at York University[49], the direct precursor of this project. DOCK's docking engine works through an inverse mapping of the active site shape with a minimal set of spheres, whose centres are then matched to atom positions in the ligand. Once a clique (a set of three or more matching point) has been found, the user-selected scoring function is called to evaluate the corresponding ligand orientation within the receptor. Available scoring functions have grown in number from version 4 to 5: the grid-based bump, contact and molecular mechanics (MM therein) functions, plus the continuous MM, and pairwise GBSA[50], are now

available. Ligand flexibility during docking is addressed in DOCK (since version 4) by an anchor-and-grow algorithm. The molecule is partitioned at the single, acyclic, rotatable bonds, and once a first fragment is docked into the site the rest of the molecule is 'grown' accounting for different possible orientations (conformationally expanded, in the word of the authors). A supplementary internal energy evaluation is available for use with the anchor-and-grow algorithm. This specialised module excludes all close-neighbour interactions up to the third interconnecting bond, and is otherwise identical to the continuous MM.

1.10 GOLD

The Genetic Optimisation Of Ligand (GOLD) program [40, 42, 43] is nowadays one of the most widely used commercial docking program. Developed by Jones et al. at the University of Sheffield, in collaboration with Wellcome (now part of Glaxo Smith Kline), it is marketed by the Cambridge Crystallographic Data Centre and was recently revised and updated by Astex Therapeutics. Its distinguishing feature is the ability to perform flexible ligand docking using a Genetic Algorithm (GA, see section 5.6) to optimise the mapping between hydrogen bond donors and acceptors of the two interacting parts. Ligand and receptor H-Bond interaction points and dihedrals are mapped onto two strings, whose internal members are then paired to each other. Each possible pairing corresponds to a given ligand orientation of the ligand within the active site. By maximising the number of pairings, the GA implicitly obtains the *snuggest* of orientations, while at the same time keeping a sensible strain on the ligand, since a torsional potential keeps the dihedrals in check. GOLD's scoring function, GOLDSCORE, consists of an H-Bond term, a torsional term based on quick evaluation *via* look-up tables, and a van der Waals term. The peculiarity of this (empirical) force-field-like function is that its parameters have been optimised to allow GOLD to reproduce known binding modes, and not binding affinities. As a consequence, GOLD was, until recently, the best program at this task, and consistently ranked first in benchmarks where multiple programs were put to work on the same set of complexes – see next chapter for a more detailed overview. Of course, this performance comes at a price, which is GOLDSCORE's very limited ability to give a reliable measure of free-energy. This is probably the reason behind the decision to implement CHEMSCORE [51, 52] as a second scoring function in the new version of GOLD made available by CCDC/Astex [53].

1.11 Glide

Glide, the docking program part of Schrödinger's Maestro Molecular Modelling suite, uses a different approach for the generation of poses. The authors say that their method "*approximates a complete search of the orientational, conformational, and positional space of the docked ligand*" [44, 45]. In fact a series of poses are generated and scored, allowing the program to select a restricted search space. A first selection of ligand poses sees their internal conformation optimised with an OPLS-AA[54, 55] force field. The best selected poses are then minimised using Monte Carlo sampling, and the final scoring is done with a modified ChemScore function. The authors claim almost twice GOLD's performance (in terms of RMSD deviation) for the reproduction of known structures.

1.12 Summary

Of course, this brief presentation of docking methods does not pretend to be exhaustive. The reader is however referred to a number of widely known reviews of the subject [21, 37, 56]. Also, benchmarks papers presenting the performances of many different programs performed on common test sets are now available [57-61]. The metrics used to express the performance vary significantly, but can usually be reduced to a few classes, namely reproduction of known binding modes, reproduction of known binding affinities, and simulated virtual screening experiments where enrichment is used to assess the performance. Undoubtedly, protein-ligand docking has shown itself to be a valuable technique in drug discovery whenever the protein structure was present, helping to select new leads during the first phases of lead identification, and to rationalise experimental results, or improve the binding properties of new candidates during the early lead optimisation phase. Some examples of drugs where the docking input has been determinant are: *zanamivir* from GSK, a neuraminidase inhibitor sold under the brand name of Relenza; and *amprenavir*, an *HIV* protease inhibitor sold by GSK under the brand name of Agenerase.

2 Scoring

2.1 Introduction

Scoring functions, as previously shown, play an important role in both phases of docking. In the first, they drive the conformational sampling toward the minimum of the underlying energy surface. A function able to select the right binding modes should allow us to explore efficiently the search space without imposing too heavy restraints. Ideally, in the second phase, to obtain a correct ranking, we would like the scores returned to correlate well with experimental free energies of binding, as defined in the previous chapter in equation (1.1) and (1.2). Unfortunately, although the sampling and selection of poses is nowadays handled with a remarkable degree of accuracy, the ranking has turned out to be more difficult than expected. This is partly due to the fact that whereas the insertion of a ligand can be easily modelled with simple functions that take into account shape complementarity and some directional interactions, the actual free energy value strongly depends on the interaction of the ligand with the solvent surrounding it, and also requires more elaborate consideration of entropy. Hence, the use of two different scoring functions during the two phases is often advocated.

2.2 How to Score

The components required for a scoring function can be developed by considering the fundamental physics of intermolecular interactions [62-64]. The important electrostatic interactions between the charges on the protein and ligand [21] may be modelled using a simple Coulomb expression, moderated by dielectric screening arising from any intervening molecules. For more accurate work, a full multipole expansion of the electrostatics may be included. Repulsion/dispersion interactions may be treated using either a simple Lennard-Jones 12-6 potential, with the option of the more accurate Buckingham potential that incorporates an exponential repulsion term. Hydrogen bonding is the result of the interaction between an electronegative atom (acceptor) and a hydrogen atom covalently bound to another electronegative atom (donor). The hydrogen

atom possesses a considerable positive partial charge, favouring electrostatic interactions with the acceptor. There may, however, be an additional covalent contribution to the hydrogen bond. Hydrogen bonds are often attributed as giving specificity to the binding process, since in general, all hydrogen bonding sites in a protein-ligand complex should be satisfied for optimum binding to be observed [65]. To date, and to the author's knowledge, no hydrogen bond function considering the number of unsatisfied hydrogen bond as a penalty term for binding has been presented: this is something we did try to address in this project (see section 5.3.2). Interactions involving π electron systems, including π-stacking and those involving cations or hydrogen bond donors, may be modelled using electrostatic and repulsion-dispersion approximations, although charge transfer effects may also need to be incorporated. The hydrophobic effect is commonly invoked to explain the preferential association between non-polar molecules, or areas of molecules, to minimize water contact. It reflects a complicated interplay of enthalpic and entropic effects involving the molecules and the aqueous solvent. This list of intermolecular forces is not of course exhaustive, as many other forces can play a role in binding, such as the internal energy of the ligand, which in order to bind, may have to assume a different conformation from the one it holds in solution. There are examples of studies that identify the most stable conformation of free and bound species to evaluate the free energy of binding, via a statistically-mechanically correct weighting of the conformations' ensemble [66, 67]. This approach is too expensive for SVS, so scoring functions usually approximate this contribution with rule-based dihedral counts or internal van der Waals-like potentials [40, 68].

Different approaches to scoring have led to different kinds of scoring functions, which are often classified in a tree-like structure, with three principal branches, plus one more 'hybrid' type. The first three branches are those of force-field, empirical and knowledge-based scoring functions. For each of these, we will give a rapid explanation of their rationale, problems encountered up until now, and the most recent attempts at solving these problems. For detailed information on the actual implementation of these approaches, the reader is referred to the original papers or a number of excellent reviews [21, 56] or comparisons [38, 69].

2.3 Force Field Scoring Functions

This class of scoring function exploits currently existing molecular mechanics force fields to estimate the enthalpy of binding. Components of the AMBER [70], CHARMM [71] and other force fields are routinely used as scoring functions in various docking programs [39, 72]. As the calculation of non-bonded term is expensive when applied to ligand/protein docked poses, it is often the case that values on a grid are pre-computed and then interpolated (both linearly or with higher degree polynomial expressions) to obtain the correct value at the site where the guest atom is located [72]. The van der Waals interactions are a commonly used component [39, 40, 73], as well as methods to take into account the desolvation energy and the long-range shielding effect of water [74], either using a distance dependent dielectric constant [75] or a complete Poisson-Boltzmann treatment [76]. This last term is very expensive to compute, and has been substituted by faster approaches [77], such as GBSA [78], recently recast in a pairwise approximation by Liu and coworkers [50], which brings the time needed for a ligand GBSA evaluation to well within the time frame for virtual screening (0.5 s/ligand).

Another important term often included is one that takes into account the internal conformational energy of the ligand. Although experiments on ligand strain energy have been rather rare and sparse until now [79], it is commonly accepted that when a ligand binds inside a binding site it will do so in a low energy conformation [80]. The implementation of this term is usually based on look-up tables for dihedral angles compiled from small molecule datasets.

A fresh approach to force field scoring has come in recent year from Pearlman and Charifson [81, 82], who developed and tested **OWFEG** (**O**ne-**W**indow **F**ree **E**nergy **G**rid), a grid approximation of the free-energy perturbation method. In **OWFEG**, a short MD simulation is used to build a grid of free energy values relative to some probes (neutral, charged, methyl) around the active site. The score is then obtained by linear interpolation of these values for the docked pose. Both K_i prediction[81] and SVS experiments[82] have been performed using this method.

2.3.1 Problems and Solutions

The main problem with force-field scoring functions is that their computation is quite time consuming. Also, parameters that are initially derived for other simulation

methods, such as MD or MC calculations, may not be suited for single-point energy estimates; correlations with experimental data may be poor, especially when the compounds ranked do not belong to the same class. Also, poor charge modelling seems to have a detrimental effect. The obvious solution to these problems seems to be the development of *ad hoc* potentials and charge models for specific use in scoring functions. Another palliative could be the introduction of new non-bonded terms, to estimate interactions until now neglected or 'merged' with others. As an example, we cite some recent papers [83-86] where a quantum-mechanically-based semi-empirical hamiltonian approach to scoring functions is used to estimate the metal-ligand interaction contribution to free energy of binding. Although currently too slow for virtual screening, there is the clear opportunity for such methods to be more widely used, if they provide information that simpler scoring functions cannot.

2.4 Empirical Scoring Functions

Another possible approach to scoring is exemplified by the class of so-called empirical scoring functions. Their rationale is that the free energy of binding can be decomposed into a series of contributions, each and every one with its own clear and intuitive chemical explanation. This assumption is, strictly speaking, incorrect [87], since whereas the free energy of binding is a function of state, the energy components into which it may be divided are not. However, the approximation is useful to a certain extent. By using terms with clear and intuitive explanations, we may be able to model specific interactions, such as π-stacking, that are difficult to capture using alternative approaches. Probably because of their lower computational requirements, the very many terms that can be used, and the different ways in which they can be implemented, empirical scoring functions are present in many SVS methods. First attempts at empirical scoring involved the building of a knowledge base from the analysis of group-to-group interaction in 200 complexes of known experimental binding affinity [88]. These first methods did not require any knowledge of the three-dimensional structure of the complex, as they simply were counts over interactions that were presumed to exist in the complexes used as reference.

Later, atom-to-atom approaches were used in the same fashion [89]. Adding knowledge of the three-dimensional structures of the training set leads to a greater level of detail in the

analysis of interactions between host and ligand [90, 91]. A partitioning scheme can be developed in which the free energy of binding is given by a sum of interaction terms, each multiplied by its own coefficient. In mathematical terms, we can write:

$$\Delta G = \sum_i f_i \Delta G_i \tag{2.1}$$

Here, the f_i are the coefficients, and the ΔG_i are the free energies associated with each term, each reflecting an interaction thought to be important for binding. Other, non-linear, functional forms have been investigated [68], but results seem to show that non-linear terms do not sufficiently improve predictive power to justify the computational overhead and loss of physical meaning on moving away from familiar terms. The coefficients are generally obtained through optimisation/regression techniques in which the scores given by the equation are fitted to known experimental binding affinities. A notable exception to this procedure comes from the work by Smith et al. [49], in which the scoring function is instead parameterised using enrichment of true binders interspersed among a number of decoy molecules.

Common terms in empirical scoring functions include those for hydrogen bonds, ionic interactions, hydrophobic interactions, and the internal energy of the ligand. This makes modern empirical scoring functions similar to a non-bonded force field potential, with parameters provided by fitting to known experimental binding affinities. As these functions are additive by design, larger ligands tend to score better than small ones. In reality, however, larger ligands will suffer from the fact that, to bind, more degrees of freedom are frozen by confinement in the receptor pocket. For this reason, the entropy change usually disfavours binding. Terms that try to estimate this entropy change, predominantly through rotatable bonds counts or similar quick methods, are therefore added. As an example, Pan et al. suggested multiplying the score by the square root of the number of heavy atoms, or the ligand's molecular weight [92].

2.4.1 Problems and Solutions

Despite their successes in modelling such complex interactions with relatively simple equations, empirical scoring functions have their limitations. A major drawback of any regression-based scoring function is the dependence on the size, composition and

generality of the training set used to derive the weights. This is commonly recognized by workers in the field, and to counter this, attempts are being made to create common training and validation sets, containing diverse proteins and ligands. A section of this review is devoted to the description of these efforts. It must also be noted that creating scoring functions 'tailored' for a certain class of protein or ligand is becoming more common [93]. An example of this is Laederach's work with carbohydrates [94] focused on affinity prediction. The other problem with empirical scoring functions is that they, in common with force field scoring functions, often lack terms to account for some interactions. For example, metal-ion interactions are often neglected or approximated by simple coulombic terms, without any reference to charge transfer and coordination. Unfortunately, to gather sufficient information to parameterise such a term can be difficult, so their contribution to binding is merged into other terms. For example, if metal ion interactions are neglected, and we do not have proteins that use metals to bind the ligand in our training set, then our function will obviously fail when applied to such a protein. Alternatively, if we have a few cases in our training set, but no term to account for them, the fitting algorithm may try to compensate for this lack by increasing the coulombic contribution. This will lead to an overall overestimate of coulombic interaction. Even in the case of metal-proteins, this function would not behave correctly, as this modest overestimate is not sufficient to address the missing metal ion term. The answer, in this case, is to develop and add new terms, and once again to select accurately the training set so to reflect the application field. Finally, it is of the utmost importance to remember that our models are just that, and as such limited in scope and applicability.

2.4.2 Recent Advances

A number of novel empirical scoring function have appeared recently: some are simple re-incarnations of earlier functions, for example, Verdonk et al. [53] re-implemented ChemScore [52] within GOLD [40]. At the same time they conducted studies on the influence of using different scoring functions for the sampling and ranking parts of the docking process. The CCDC/Astex docking test set was divided into fragment-like and drug-like ligands. Overall, they confirmed that using GoldScore for both sampling and ranking gives the more accurate results, in contrast to the previous assumption that using two different scoring functions for the two phases could improve the quality of the poses. However, the optimum compromise between speed and accuracy was indeed

to use ChemScore for sampling, and re-rank the outcomes with GoldScore. In this way, sampling was up to three times faster, and the results were nonetheless purged of false positives thanks to the better selectivity of the GoldScore function. Verdonk et al. [95], in a more recent publication, allowed for water molecule to be switched on and off at their experimentally determined position during docking. GOLD's GoldScore and ChemScore functions were modified to consider the loss of the water's rigid-body entropy upon binding, with a constant penalty term σ^p, obtained from training against 58 complexes, and tested against 225. The method managed to predict correctly water displacement/mediation for over 90% of cases in both training and test sets.

Mancera et al. [96, 97], in their program EasyDock [98], used PLP [99] to drive sampling, and ScreenScore (PLP+FlexX's own score) [100] to re-rank the outcomes. The new program GEMDOCK [101], uses a GA for docking, and a novel empirical scoring function. Composed of three terms, it contains coulombic interactions with simplified charges, PLP-like hydrogen bond terms and internal energy. This final term is in fact a constant penalty to restrict poses to a pre-defined docking box. The program has been tested for binding-mode recognition against 100 protein/ligand complexes selected from the PDB.

A new version of GLIDE has also been presented [44, 45]. This program uses a series of filters, of increasing complexity, to select good binding modes. It includes a scoring function derived from ChemScore [52], extended with a few more terms. Both re-docking of known inhibitors [44] and SVS [45] experiments are presented. The authors claim that, overall, GLIDE's performances are equal or superior to those of GOLD and FlexX.

An opportunity to improve the performance of scoring functions lies in the method by which a function is parameterised. Smith et al. [49] have built a novel scoring function, selecting terms from Gschwend's TEC, a toolkit containing a variety of 2 and 3-D descriptors for protein-ligand interactions. The novelty of the method lies not in the terms selected, namely hydrogen bonds, coulombic interactions, a contact/clash count, and a buried hydrophobic surface term. Here, the parameterisation method attempts to mimic a SVS experiment, with a training set of twenty proteins and 1000 ligands selected from the WDI. The ligands were used as decoys to hide the known ligand, while a Genetic Algorithm adjusted the parameters to optimize the enrichment factors.

The functional form of an empirical scoring function can be changed as in, for example, Giordanetto et al. [68]. They assembled and tested a series of scoring functions combining different terms obtained from QXP [46] descriptors, solvent accessible surface area and protein-side chain entropy. The different combinations were then trained against 100 PDB protein-ligand complexes (actually, their affinity/inhibition values) using a variety of methods (partial least squares, Genetic Algorithms, neural networks), and tested against 24 diverse complexes. Other forms of the scoring function, other than the simple linear combination of terms, were examined by the Genetic Algorithm, but with no significant improvement. Moreover, as in the case of neural networks, the non-linear terms do not always have a clear physical meaning.

2.5 Knowledge-based Scoring Functions

The third family of scoring function has been developed thanks to the huge increase in high resolution structures [35] obtained via X-ray diffraction, solid state NMR, and a number of other related methods. These scoring functions are called knowledge-based, as their fundamental idea is to extract statistical information about the ligand/protein binding modes and to correlate these to the free energy of binding using statistical mechanics. Knowledge based pseudo-potentials are usually derived by accumulating radial distribution functions for selected pairs of protein-ligand atoms. The inverse formulation of the Boltzmann equation (see below), or a derivative thereof, is then used to extract the corresponding potentials of mean force from these distributions. To apply these potentials to scoring, the distances between the protein and ligand atoms in each pose is calculated, and the total score obtained by summing each of the energies derived from the potential curves using these distances.

$$E_{ij} = -kT \ln p_{ij}(r) - kT \ln Z \qquad (2.2)$$

Here, E_{ij} represents the interaction energy between atom *i* on the protein and atom *j* on the ligand; $p_{ij}(r)$ is the radial distribution function between the two atom-types at distance r, and Z refers to the partition function. A very good and detailed explanation of the rationale behind this method can be found in the recent review by Sotriffer et al.[21].

By not imposing any particular functional form to the pseudo-potential curves, we are implicitly accounting for any possible kind of interaction that could occur, irrespective of whether it is enthalpic or entropic.

Besides the functional form used to derive the potentials from the populations, the differences between various incarnations of knowledge-based function are in the amount of data used for parameterisation, and in the number of atom types. There is no doubt that the more protein-ligand complexes used in parameterisation, the better. With a larger knowledge-base, the statistical information extracted will be of better quality; one would expect, for example, smoother curves, and improved sampling for rare interactions. The same holds true for the diversity of the parameterisation set, which makes for a more robust scoring function. The number of atom-types, however, does not follow the same premiss "the more, the better". Add too many, and the data is needlessly fragmented – there may not be sufficient information to build pseudo-potential curves between certain atom pairs, with the data therefore effectively being wasted. There is also the risk of implicitly over-fitting the function. Conversely, if there are insufficient atom-types, different interactions will be incorrectly grouped, giving potential curves that, although smoother, may have unphysical features: For example, a single pseudo potential curve with two minima could arise from the superposition of two single-minima curves. Alternatively, it may be signalling the presence of some angular-dependent phenomena, which radially-symmetric potentials cannot incorporate effectively. A balance must therefore be struck, and inevitably different researchers will make different decisions. Good practice is arguably to choose well-known and well-defined atom-types, as Gohlke and coworkers did with DrugScore [102, 103], which uses 17 atom-types taken from the Sybyl mol2 file format. This is ideal as it reduces the difficulties of reliable ligand atom-typing. However, others [104, 105] used more than 17 atom types as their choice to represent the diversity of pairwise interactions in their parameterisation set, or adopted custom-defined atom-types: for example, Muegge et al. chose atom-types based on chemical features and not the atom's hybridisation state for their PMF scoring function [106]. All these scoring functions were parameterised with protein-ligand coordinate sets extracted from the PDB database [35], although DrugScore used Relibase [107, 108] to identify high-quality independently validated structures.

The first applications of knowledge-based scoring functions to drug design were strictly focused to HIV protease [109, 110], the only proteins at the time for which there were sufficient structural data to develop a scoring function. Although promising, these early attempts did not return a generally applicable function. SMoG [105, 111] was the first drug-design program to include a knowledge-based pseudo-potential. Later attempts introduced new terms to overcome some limitations of the methods due to insufficient sampling, or the neglect of solvent exposure. Mitchell et al. proposed BLEEP [112, 113], where the functional form for the shell density is different, and the knowledge-based function is complemented by a van der Waals term to compensate for the low occurrence of short range interactions in crystals. The PMF scoring function [106, 114-117] from Muegge, included a term to take into account the volume occupied by the ligand. This function was tested against the FKBP protein [115], and it was also modified to improve virtual screening performance, by substituting the knowledge-based curves with a van der Waals terms to account for short-range steric effects otherwise neglected. The PMF function contains a volume correction factor, to account for the volume around each ligand atom occupied by other ligand atoms. Its effect on the predictive power was also investigated [114] by the authors; apparently it only helps the function to fit to experimental affinities, but does not affect relative ranking of the ligands. In DrugScore [103], from Gohlke and coworkers, a knowledge-based one-body potential scaled to the size of the solvent-accessible surface (SAS) of the protein and the ligand that becomes buried upon complex formation, was added to the PMF-like pairwise pseudo-potentials. Weighting coefficients scaled the two terms to optimise K_i reproduction. This results in DrugScore effectively being a mixed knowledge-based/empirical scoring function. Notably, the DrugScore potentials were *smoothed* to avoid spurious spikes in the pseudo-potential curves due to misplacements in the crystal structures. The maximum radius considered was limited to 6 Å (12 in PMF), under the justification that only direct contacts, and not water-mediated ones, were important. Recently Ishchenko et al. released a new version [118] of SMoG (SMoG2001), parameterised on 725 structures, and tested against 119, reporting performances similar to DrugScore and better than PMF and SCORE1 (LUDI [90]). They attribute improvements mainly to a better description of the reference state used to normalise the radial distribution function.

The use of the inverse Boltzmann distribution is not, however, the only way that information can be extracted from known structures. A very different approach is shown by Deng et al.[119], who created the Structural Interaction Fingerprint method (SIFt). This technique, based on the conversion of 3D structure into 1D binary fingerprints, not only makes for improvements in organisation and analysis of the structures, either experimental or generated by docking, but more importantly for our interest, it also returns superior enrichment performances when used to screen a database of ligands. The same authors in a follow-up paper [120] show how p-SIFt, a probabilistic interpretation of the SIFt method, can be used to generate target-specific knowledge-based filters for virtual screening.

2.5.1 Recent Advances

Recently, an additional scoring function was added to GOLD, when Verdonk et al. [121] implemented a derivative of DrugScore. This implementation of the scoring function was tested for its ability to drive the docking algorithm, and found to be unsatisfactory compared to ChemScore and GoldScore. This is probably due to the lack of directional terms, which make it difficult for the scoring function to select the correct binding mode. DrugScore (CSD), a re-implementation of DrugScore using distance-dependent pair potentials obtained from small molecule crystal structures, was also unveiled [122]. The higher resolution of these structures compared to protein-ligand complexes led to substantial improvement in recognition of near-native binding modes being observed by the authors. Furthermore, the new scoring function shows a better correlation with binding affinities.

In addition to new implementations of existing scoring functions [118], Feher et al. [123] proposed a new knowledge-based scoring function (BHB), based on descriptors for Buriedness, Hydrogen-bonding, and Binding energy. The approach is very different from traditional knowledge-based scoring functions, because the knowledge-base in this case is harvested not to build radial distribution function-like potentials, but is instead looking for characteristic binding modes and contacts. A list of 'good' interactions with the protein's residues must be compiled prior to running the virtual screening experiment, and the scoring function can also be tailored specifically for the target in a few hours, if sufficient structural data from known actives is available. This new BHB

function claims a 12-fold enrichment improvement over GoldScore when re-scoring GOLD generated poses on a set of PDB protein-ligand complexes.

Murishev et al. [124] developed a three-stage, mixed knowledge-based/empirical parameterisation protocol: First, pairwise pseudo-potentials are extracted from a set of complexes using the inverse Boltzmann law; then, the potentials obtained are approximated by smooth functions *via* multivariate regression; finally, a re-calibration against known affinity data corrects some of the adjustable parameters, such as the energy and radius of hydrophobic interactions, introduced during the second stage. The function generated this way shows better performances than DOCK, GOLD, and FlexX in predicting binding modes.

The strongest criticism of knowledge-based scoring functions, apart from doubts on their statistical-mechanical meaningfulness, is that not all interactions can be approximated efficiently by pairwise terms. Moreover, many interactions are highly directional, whereas radial distribution function-derived pseudo-potentials have spherical symmetry. Although a certain degree of directionality comes from the interplay of different pairwise interactions, the issue has not yet been fully addressed. However, there seems to be little concern in the literature, since these functions are cheap to evaluate, and already work reasonably well within the lead-identification phase of drug-design. With the constant increase in resolved structures, the reliability and general applicability of these functions is likely to improve, and the possibility (re)opens for functions optimised on certain protein-ligand classes, or even for different alternative non-pairwise representations of the intermolecular interactions.

2.6 E Pluribus Unum

Since none of the scoring function classes presented above is clearly superior to the others, a more pragmatic approach to obtain good results is to combine their separate judgements. This approach, called 'consensus scoring', was pioneered by Charifson et al. [125]; two or more scoring functions are applied to the same set of poses, and only structures that perform well in more than a pre-defined share of the said scoring functions are retained. Consensus scoring, analysed in detail by Wang et al. [126], is effective in reducing the number of false positives, which may be able to trick one function, but not all, into believing they are actives. Because of this, performances are

particularly good when the functions included in the jury are different, in the sense that they describe different aspect of the binding. For example, a scoring function oriented towards the careful detection of hydrogen bonds will complement a more hydrophobic function, whereas the union with a more similar function may not be so productive. An example of this is given by Stahl and Rarey [100] with ScreenScore, built from the consensus of PLP and FlexX. However, as good as consensus functions are at getting rid of false positives, they do nothing to enable us to recover false negatives which slip through most or all of our scoring nets and lie at the bottom of our ranked list. In these cases, the description of the protein-ligand interactions is most likely deficient, so there is still room for new terms to be added.

2.6.1 Recent Advances

The consensus scoring field has been very active recently, with a number of interesting papers exploring new solutions and extending and/or confirming other workers' ideas and suggestions. Recently [121], Verdonk et al. tested a number of different consensus combinations and confirmed Wang's [126] insight that rank-by-number outperforms rank-by-rank and rank-by-vote strategies in consensus scoring. Wang [127] and coworkers again, using three scoring functions built upon their previous work (SCORE [128]) created a new consensus scoring method (X-CSCORE), calibrated to reproduce binding affinities of 200 complexes, and tested against thirty more.

Guo et al. [129] built a consensus method using five commercially available functions, previously cited and all available within Sybyl: ChemScore, G-score, F-score, PMF-score and DrugScore. The five were combined not as a simple jury, but using multi-linear regression. The training set was composed of 53 inhibitors' of *Torpedo Californica* Acetyl Cholin Esterase (AChE) from PDB entries with known affinity re-docked into a Human AChE. 16 (different) compounds were used as a test set.

Similarly, Jacobsson [130] applied different training methods to 'multidimensional' consensus scoring. Partial least squares, discriminant analysis, bayesian classification, and rule-based methods were applied, to improve discrimination between active and inactive compounds. Their analysis goes beyond the 'crude' measure of enrichment, by explicitly defining four more measures for assessing scoring function performance, in addition to the usual enrichment factor.

Naive bayesian machine-learning algorithms are used by Klon and coworkers [131] to improve enrichment in high-throughput docking of databases, by selecting the important features from the top ranked structures with extended connectivity fingerprints, and correlating these to their high-scoring compounds. The method works by re-ranking the docking outcome and retrieving compound similar to those in the top of the list from the remainder. It is to us unclear, though, if this would have a detrimental effect on identifying 'novel' leads, unlike any known inhibitor. Does this approach inhibit scaffold hopping?

2.7 Comparisons between Scoring Functions

Bissantz et al. [57, 132] first had the idea of comparing different scoring functions on the same test set. From then on, the issue of comparison between different functions has gained importance. Recently a number of papers tackled the issue from different viewpoints. These are not just sterile comparisons head-to-head, but instead a useful method to identify common deficiencies in modern scoring functions. Wang et al. [38] tested 11 freely and commercially available scoring functions by reproducing the affinity of 100 protein-ligand complexes. Recently, the same authors [69] extended the test, by using 14 different scoring functions and binding affinities of 800 complexes from the PDBBind [133] database. Simulated virtual screening experiments were also performed, on HIV-1 protease, carbonic anhydrase, and trypsin complexes. Ferrara et al. [134] assessed nine scoring functions for their ability to recognise the correct ligand orientation in 189 complexes from the LigandPDB [135]. Krovat and coworkers [136] applied LigandFit to the test case of renin: 10 inhibitors were mixed with 990 drug-like compounds, docked and then analysed with seven scoring functions: LigScore1, LigScore2, PLP1, PLP2, JAIN, PMF, LUDI. The consensus combination PLP1+PLP2+PMF performed the best, presenting all known inhibitors within the top 8% of the list.

Other papers, although not strictly focused on scoring functions, but on docking in general, still offer useful insights: Kontoyanny et al. [60] investigated the sensitivity of five docking programs (FlexX, GOLD, DOCK, LigandFit, and GLIDE) to the nature of the active site. 69 receptors (belonging to 14 families) were used as targets. Perola and coworkers [61] compared three docking programs, GLIDE, GOLD and ICM. The test

focused not on reproduction of binding affinities or ranking of known inhibitors, but instead on the recognition of the correct binding mode; a mixed test set, composed of PDB and Vertex *in-house* complexes (100 of each) was used. The influence of the nature of the active site, and the effect of energy minimization was assessed. A further test compared the three programs on HIV-1 protease, p38 MAP Kinase and IMPDH. GLIDE was deemed the top-performer. DOCK, FlexX, GLIDE, GOLD, Slide, Surflex, and QXP have been tested by Kellenberger et al. [59] in a 100 X-ray small-ligand structure reproduction experiment, and in a fictitious SVS experiment. The three programs (GOLD, GLIDE and Surflex) performed well at both tasks. The authors criticised docking performance on the basis of active site features, and identified common failures. In Vigers et al. [137], once again, it is shown how FlexX and GOLD correctly select the right ligand orientation, but fail to consistently rank a series of ligands. A statistical multiple active site correction (MASC) is presented. This correction lowers scores for ligand which are found to score well not only with the target of interest, but another selected set of seven to nine diverse protein structures, on the assumption that these ligands are 'tricking' the scoring function into believing that they are 'universal binders'. The correction is shown to improve ranking, but unfortunately it is specific to the docking and scoring algorithm used, and so must be re-generated for any new combination.

Concerning structure-based virtual screening techniques, Verdonk et al. [121] advocate the use of 'focused' libraries, instead of general 'drug-like' libraries, when setting up SVS test beds, to avoid ' artificial' enrichment due to diversity between the 'true' ligand and decoys.

2.8 New Test and Training Sets

With an ever increasing number of scoring functions, and comparisons between them, the need has emerged for a common test set. The ideal sets should include many diverse proteins, be representative of the 'protein space', independent of the bias inherent in the current PDB. This bias is due to a combination of some proteins being more readily crystallized, while other proteins are simply more topical, for example HIV-1 protease. The ligands to which the proteins are complexed should also be diverse, and arguably all should be drug-like [138, 139] molecules. This has been an issue until recently, as many scoring functions were parameterised against complexes containing cofactors, or

peptidic ligand, or sugars. In the past, many scoring function authors simply published the selection procedure or just the PDB entries for the proteins used during the training or testing phases, leaving the reader to repeat the work in order to gather similar data. With the increasing size of the PDB, knowledge-based scoring function have become possible, and much effort has been put into selecting PDB entries with suitable features for such training sets. The need for a standard 'set' against which to test new scoring function is however clear. This need was fulfilled by CCDC and Astex Therapeutics who in 2002 published a set of 305 proteins [140], containing all the PDB entries from the GOLD [40] and ChemScore [52] test-sets, plus others. This was not just a re-presentation of data though, as the original structures had been completely revised starting from the experimental electron density data. Moreover, tautomeric states for ligands and proteins have been assessed. This set was welcomed by users and developers, and in recent publications [53, 97] it has been used in the way its authors hoped, namely as a test or training set [141]. The second, more recent attempt to produce a standard dataset is that of PDBbind (http://www.pdbbind.org), by Wang et al. [133, 142] This set consists of 800 protein-ligand complexes culled from the PDB. The complexes all share consistent features, such as high-resolution structure, drug-like ligands, and availability of experimental affinity data. Ligand and proteins have been separated and saved in different files, and all are searchable through a web interface, upon (free) subscription. Also, submission of new affinity data collected from literature is possible, so this set is likely to grow and become a digest of the PDB for drug-developers. The PDBbind set has been used by its own authors for the comparative evaluation of 14 scoring functions [69].

2.9 Conclusions

As shown, the field is certainly active and still facing a large number of issues. During this research project, attempts have been made to address only a few of them: First, the dependence of scoring functions on experimental measured affinities, which forcefully bias the training sets toward the tight binding side of the chemical space, has been assessed. The approach in this case followed the work of Smith et al. [49], in that the scoring function is optimised by bringing the true ligands to the top, amongst a selection of drug-like decoy compounds. In parallel, another kind of bias was tackled, depending on the particular choice of terms used in the scoring function – albeit extensive, this choice may not be the optimal to describe the interactions, and depending on the

frequency of the interactions composing the training set, the scoring function may be inappropriately configured. A QSAR-like approach was chosen in this case, where an automated routine would select the best combination of terms, in order to optimise the resulting scoring functions' performances. Thirdly, an attempt was made at addressing the issue of 'robustness' of the scoring function, cross-validating the performances of functions parameterised against different data sets. The influence of both the training sets' size and composition on the scoring function performance was considered. Through the combination of these approaches, we are looking to develop a robust scoring function suited to yielding enrichment in virtual screening experiments.

BLANK IN ORIGINAL

3 The PDB and other databases

3.1 Introduction

Computer-aided drug design, in many of its areas, is a very data-intensive field; there is always the need for new protein structures, or new versions of the old ones with better resolution or new ligands. In addition, new small molecules need to be catalogued for use as tentative leads. This wealth of data has been categorised in various databases, both of small molecules, such as the Cambridge Structural Database (CSD)[143], or the World Drug Index (WDI) [144], and biological macromolecules, such as the Protein Data Bank (PDB) [36].

However, needs change over time, and after the creation of these databases they have often been searched/used in ways not anticipated by their creators. Purpose-built databases have therefore been compiled, often using the more generic ones as starting points. One such example is the Relibase, a ligand-interaction oriented slice of the Protein Data Bank [36].

For proteins, we have a mixed situation, where the largest structural database of all, the already mentioned Protein Data Bank, is public and freely accessible by anyone. Conversely, pharmaceutical industries resolve structures for their own projects, keeping the outcomes confidential.

While the protein structures are generally wild-type, and of course pharmaceutically relevant, the situation for small-molecules is different: They are not constrained to resemble anything already in existence. Ideally, new chemical classes could prove very useful to diversify the current ***pharmacopoeia***: for example by solving issues of known drugs, or by simply moving the research outside spaces already covered by some competitor's intellectual property. It is not difficult to envisage a program able to generate as many diverse structures as one could want [105]. However, even if one were to use this wealth of newly-born structures in a virtual screening experiment, the hits

would then need to be actually synthesised, and this is likely to be a problem. The suggested structures may not be synthetically accessible. The answer to this impasse is, quite obviously, to use only easily available compounds, for example only those that can be purchased from chemical suppliers such as Aldrich or Acros. The other alternative is to explore the *synthetic accessibility* of your compounds while they are generated [145].

The structures of these compounds that may be purchased are largely available 'for free' in a much dispersed form through the suppliers' electronic catalogue. Many companies, however, have taken the effort to compact all this information into a more uniform format, tailoring it more or less for specific uses, and then selling the result on the market. Examples of this is the Available Chemical Database (ACD)[144]. The World Drug Index (WDI)[146], instead, initially attempted to gather together all known drugs, then opened the doors to the more generic class of 'drug-like' compounds, initially defined by Lipinsky [138, 139] using one of the first releases of WDI itself.

Another common source of ligands is the National Cancer Institute catalogue, resulting from the testing for carcinogenity of chemical components of paints, crop protectants, and everyday items. The publicly accessible part of the NCI database will be discussed further in section 3.3.2.

Another common way of generating small molecule databases has been to scour the PDB itself searching for ligands and cofactors present in the resolved structures. The PDBSUM [147-149] exemplifies this class. Finally, an enormous number of small molecules with drug-like properties can be retrieved by proper filtering of the Cambridge Structural Database (CSD)[143]. This is the central repository of small organic and organometallic molecule structures determined by X-ray and neutron diffraction, either recovered from literature or directly submitted by the authors. Maintained by the Cambridge Crystallographic Data Centre, it is distributed as part of a software suite which enables efficient searching and information extraction – an example is the software IsoStar/SuperStar [150-153], a knowledge base of angular and distance information from both the CSD and PDB.

3.2 Protein Databases

Structural chemistry has provided a wealth of information suitable to act as a starter for computational chemists. Resolved structures can be used as starting points for MD/MC and QM/MM simulations, and docking studies.

3.2.1 Brookhaven Protein Databank – PDB

All resolved structures of proteins ultimately end up, upon publication, in the Brookhaven Protein Data Bank [35, 36]. Previously maintained by the National Institute of Standards and Technology (NIST), the database is now under the responsibility of the Research Collaboratory for Structural Bioinformatics (RCSB). The PDB now contains well over 20 thousands proteins and is freely available to the scientific community. This allowed its file format (.pdb or .ent) to become a standard, and the database itself a widely used pool for gathering smaller data sets with more specific properties/relevance.

New databases, more tailored towards specific uses, are being compiled: they often extract information from the PDB. Since many scientific journals require submission of the resolved structure to PDB, prior to acceptance of the paper, the PDB will most likely continue to be the largest public collection of protein structures available, and a useful source of general information, to complement more specific projects.

A final mention must be made of the particular file format which was born with the PDB, and from which it takes its name. This format, defined by the RCSB, allows authors to submit their structure to the PDB with all relevant information, such as, their names and the paper where the structure was or will be announced, sequence and secondary structure, domain and chain subdivision, resolution, experimental origin, symmetry operations and other crystallographical information, and of course the coordinate for every single (heavy) atom. As important as it has been, this old-style, Fortran77-friendly format is now obsolete, and is now being replaced by MMCIF [154-157], a specialisation of the CIF (Crystallographic Information File) commonly used in Crystallography. The MM stands here for molecular mechanics. The change will hopefully bring some standards to the jungle of different extensions to pdb or completely new file formats which molecular modelling program authors have devised to overcome the pdb format's inadequacies (especially in the atom-typing and property areas). This will also allow the retention in the PDB database of important information

regarding the way the structure has been obtained, without having to use user-defined remark fields, often off the standard. It will also allow better treatment of atom-types and other atomic properties, which did not find place in the fixed-column, Fortran-friendly formatting of the old pdb.

3.2.2 PDBSum

PDBsum [147-149] is a web-based database providing a largely pictorial summary of the key information on each macromolecular structure deposited at the Protein Data Bank (PDB). Schematic diagrams of the molecules in each structure are accessible, and

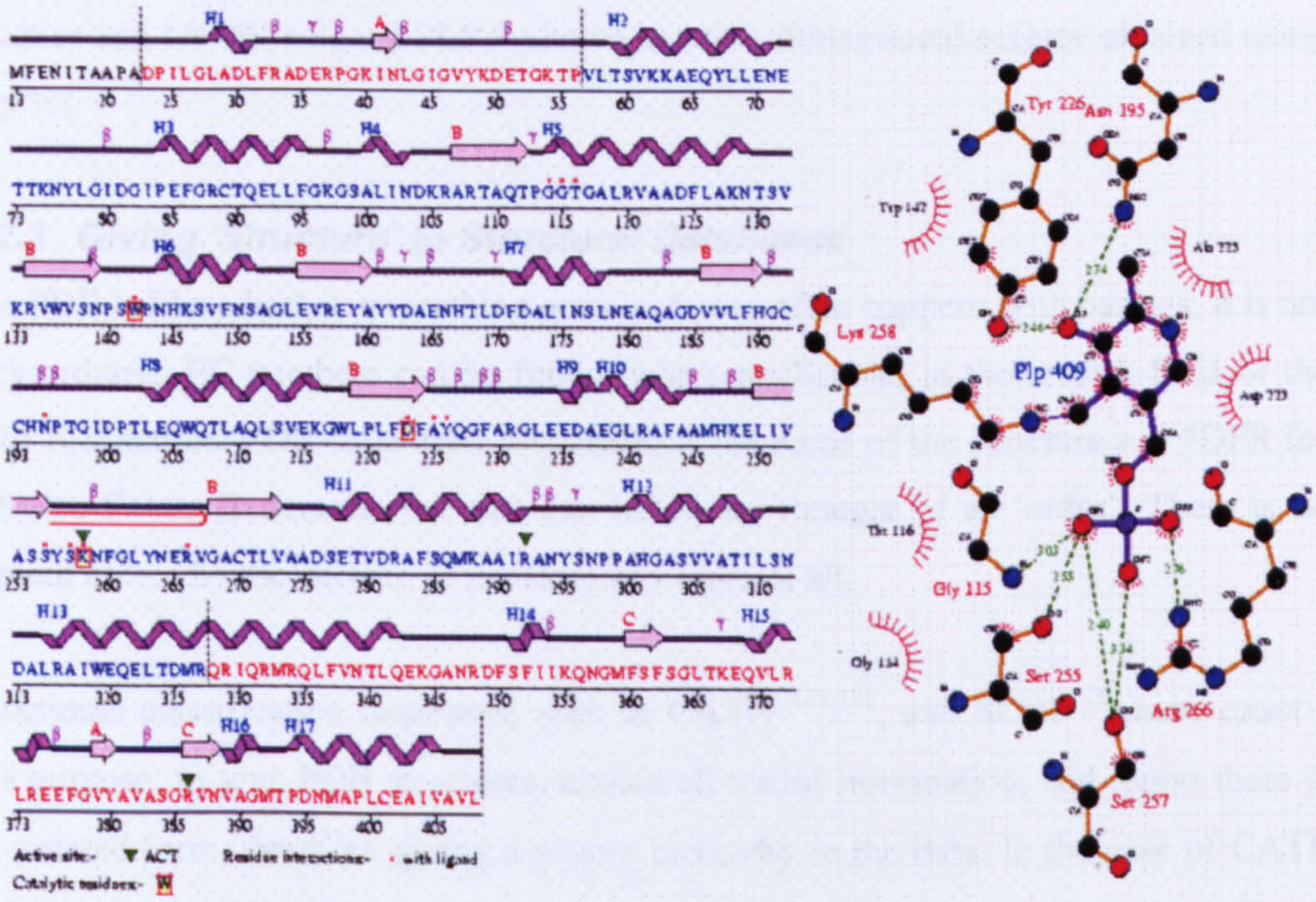


Figure 3-1: An example output of PDBSum for PDB entry 1AAW. On the left, the sequence with secondary structure and some interaction flags assigned; on the right, a LigPlot depiction of the ligand in the binding pocket.

the interactions between them are shown with images generated by LigPlot [158] – see Figure 3-1. The PDB code or a text search is used to look-up a particular protein, or the database can be browsed through its classification tree, such as what ligands or HET groups are in the structure, or the enzyme's classification.

PDBsum includes images of the structure, annotated plots of each protein chain's secondary structure, detailed structural analyses generated by the PROMOTIF [159]

program, summary PROCHECK [160] results, and schematic diagrams of protein-ligand and protein-DNA interactions. RasMol scripts highlight key aspects of the structure, such as the protein's domains, PROSITE patterns, and protein-ligand interactions, for interactive viewing in 3D. PDBsum is updated whenever any new structures are released by the PDB and is freely accessible via http://www.biochem.ucl.ac.uk/bsm/pdbsum. Figure 3-1 shows an example output obtained from the PDBSum for PDB entry 1AAW: on the left a schematic representation of the protein's sequence, with secondary structure information, and different representation (red) for aminoacid residues involved in the active site or playing a catalytic role (boxed); on the right the interaction between some of the flagged residues and 1AAW's ligand PLP is shown in a two-dimensional scheme obtained using LigPlot.

3.2.3 Giving 'Structure' to Structural Databases

The PDB is like a basket: everything goes in, but as often happens with baskets, it is not very ordered. EC numbers can be found, where applicable, in the remark field of the PDB file, and the PDB code itself often reflects the name of the structure *e.g.* *DFR for Dihydro-Folate Reductase, but that can hardly be thought of as 'order'. There is no explicit effort, by the curator, to maintain any order at all.

Functional classification databases, such as CATH [161-172], and SCOP[173], have exactly this purpose: to scan PDB structures, extract all useful information, and report them in an ordered form, therefore giving a proper hierarchy to the data. In the case of CATH and SCOP, the information sought is that of structural similarity between different structures. The attempt is to build a tree-like classification of proteins by architecture, families, and so on, showing the ultimate structural reasons for the difference in biological function of different proteins.

3.2.4 SCOP

The Structural Classification of Protein database[173-180], commonly known as **SCOP**, is a database where all proteins of known structure are ordered accordingly to their evolutionary and structural relationships. The main focus of SCOP is to place the proteins in a coherent evolutionary framework, based on their conserved structural features. Classification is made by domains, and the said entities are grouped into

species and then hierarchically into families, superfamilies, folds and classes. A *domain* is here defined as an evolutionary unit, in the sense that is either observed isolated in nature (as a protein in its own terms) or associated with others in what are known as multi-domain proteins. Small and medium sized proteins will consist of a single domain, and will therefore be treated as a whole. Further classification is, as said, hierarchical, and based on the common structural features shared at different levels, established mainly visually and with help from automatic procedures. Classification attempts to highlight the parallel between evolutionary and structural relationships.

Here follows a brief description of every classification tier:

1. **Class:** This groups is introduced for historical reasons, and assigned on the basis of the secondary structure and its assembly; five classes are presents, depending on the ratio of alpha beta structures and their interspersal/segregation:
   - **All α:** domains whose structures are formed essentially (if not exclusively) by α helices;
   - **All β:** domains whose structures are mainly composed of β sheets;
   - **α & β:** domains with both α and β structures interspersed between each other
   - **α + β:** domains with both α and β features, but kept largely segregated
   - **Multi-Domains:** Other folds or domain for which a single case is known, and whose origin is still object of debate.
2. **Common Fold:** In here go all domains with a given assembly of tertiary structure, regardless of their divergence of sequence. Often, this 'fold' recognises only a small part of the protein, and the new pieces added to tailor the protein to its function may constitute more than half the sequence. Sometimes, the divergence in structure is large enough to hide the common evolutionary origin, but the core fold is retained, or the two different cores may be put in relation between them when intermediate structures are revealed. On the other hand, domains with different origins may share the same fold because their functions require it.
3. **Super-Family:** Families which may or may not have sequence similarities, but share structures and functional features that suggest a common evolutionary origin are placed within the same super-family. As an example, actin, and the

ATPase domains of heat-shock proteins are in the same super-family because both fall within the same RiboNuclease H-like Motif (*3 layers: a/b/a; mixed beta-sheet of 5 strands, order 32145; strand 2 is antiparallel to the rest*).

4. **Family:** This is the tier more amenable to automatic treatment, as the grouping is essentially done on the basis of sequence identity, 30% or more, or at least 15% but a clear similarity of function and structures - for example, many globins form a family though some members have sequence identities of only 15%.

The advantage of SCOP over other similar databases (*e.g.* CATH, described later) is that the classification is built upon the 'human' theory of protein evolution, and not simply on similarity values returned by nevertheless efficient and exact bio-informatics tools. Computational support is used in SCOP, but only to extend the ability of a human to analyse and interpret an otherwise unwieldy wealth of data. The first official SCOP release dates back to 1995[173], and contained 3179 single protein domains (1182 excluding trivial mutations and other non-modifying changes, such as same protein but different ligands in different PDB entries). Since then, it has twice undergone major changes, in 2002[179] and 2004[180], and in this latest release the number of domains has risen to more than 40 000.

The enormous increase parallels that of the PDB itself, which is growing exponentially thanks to the advancing of crystallisation techniques, and the increased availability of instruments such as X-ray diffraction and powerful computers to perform structure refinement.

Extensive re-classification of fold and super-families has also been undertaken. In 2002[179], the old SCOP classification page number was superseded by the SCOP UNique ID (SUNID), together with the Set of Concise Classification Strings (SCCS); whereas the first depended on the order in which the domain appeared in SCOP, the second is a more absolute pointer, not depending on further insertion, and would only change in the case of an actual re-classification of the domain's fold or super-family. The pair SUNID-SCCS is now unique and briefly descriptive of every single entry, and is never re-assigned even after re-classification. Rather, new SUNIDs are created, and the old ones are left; they will be obsolete, but will not be picked up by other folds/families.

2004[180] saw the beginning of a process of merging of SCOP with the information contained in other protein structural databases such as CATH[161-172], and Interpro, Pfam[181-187]. Other than this, the usual, never-ending process of re-classification to take into account new structural and evolutionary relationship is in progress, and will probably continue as new entries are added to the PDB and new knowledge pours in from neighbouring fields[JW5].

3.3 Small molecules databases

3.3.1 Available Chemical Directory- ACD

The Available Chemical Directory[144] is a huge collection of all (roughly half a million, excluding duplicates) chemical products present in the online and paper catalogues from (670) providers such as SIGMA, CIBA etc. Because of its origins, it is not focused on drug-like molecules, so library compilation from it requires the use of purpose-built filters. However, it offers the possibility of searching the content by a few drug-likeness discriminating properties (in Lipinski's sense), such as number of hydrogen bond donors and acceptors, rotatable bond and CLogP. In addition to being searchable by name and CAS registry number, substructure queries can be used to retrieve molecules.

3.3.2 World Drug Index – WDI

The world drug index[146] is a database compiled and marketed by Thomson Scientific, a commercial entity, initially enveloping all known drugs on the market. Later on, the index was extended to drug-like and drug-precursors compounds commercially available, and has been often as primary source of three-dimensional structures for library assembly/enumeration prior to virtual screening experiment.

3.3.3 National Cancer Institute NCI-Open3dMol

The National Cancer Institute, part of the United States's National Institute of Health, routinely screens thousands of compounds for their carcinogenic activity. Part of these compounds was collected and made available to the general public through a web-based interface[188], with the ability of querying the database to retrieve only molecules with certain characteristics, *e.g.* only those *drug-like* (according to Lipinski's rules). The molecules are downloadable in a variety of formats, but there is a caveat. Because of their origins, many of these molecules do not belong to chemical classes used in

pharmaceutical chemistry, either because they are reactive, or toxic to mammals. Even 'drug-like' molecules are often part of paints, herbicides, or industrial solvents. This is all stuff you would not want to enter a human body, no matter how high its drug-likeness index. To avoid this, a series of filters must be devised to remove molecules containing groups likely to interfere with the human (mammalian) metabolism. Also, to our knowledge, structures have been re-generated using SMILES as starting points. They do not seem to have protonation states assigned in a meaningful way. All available valences are filled, so that carboxylic acids, which would be de-protonated at physiological pH, are instead neutral. As a general rule, a pre-treatment within molecular modelling software is needed before using these structures in any SVS/Docking experiment.

3.4 Ad hoc databases

3.4.1 CCDC/Astex protein dataset

Since the very beginning of docking and scoring development, it has been customary to publish the PDB entries of protein-ligand complexes used for training and testing the scoring modules. First to do so was Böhm[90], with his pioneering work on empirical structure-based scoring functions. Then, the ChemScore[51, 52] authors provided their training and test set, divided by class with the explicit purpose of establishing it as a common ground for testing. In fact, upon release of version 2 of GOLD, a new test-set for scoring functions and docking programs containing both GOLD's and ChemScore's test sets, has been recently published by Astex [140], where Murray, one of ChemScore's original authors, now works. The so called CCDC/Astex dataset is not, however a simple recollection of previously compiled test sets. It is in fact a much more complete work, since the authors have gone back to the original literature. For every protein included, they retrieved from the original authors of the publications the electron density data obtained by the diffraction experiments, re-generating the structure in a standardised way, and carefully assigning charges and protonation states, sometimes correcting wrong assumptions made in the first instance. The result is a set of high quality, high resolution structures, further subdivided into subsets. There is the whole set, containing 305 proteins, then the hi-res set, containing all proteins with resolution of at least 2.5Å, and finally the 'clean' set, where all hi-res duplicate complexes have been eliminated, to end up with only 185 complexes. Waters thought to be important for

binding have been saved in a file of their own, together with mol2 files for the ligand (two files in case of binding sugars, to account for the α and β form). The CCDC/Astex set was in fact used during this study, particularly its 'clean' subset, as explained in the next chapters.

3.4.2 PDBbind

The latest addition to the plethora of protein-ligand data collections is the so called **PDBbind**[133, 142, 189], whose purpose is to gather all information available on "*Protein-Ligand complEXes with both known Binding Affinity and three-dimensional Structure*" (***PLEXBAS***, in the words of the authors). As shown and stressed by the authors, this database does not offer, as others did before, a simple recollection/recompilation of old training and test sets. It is, in fact a screening of the PDB (started with release n° 103, Jan 2003) in search of all such complexes. The authors patiently catalogued all structures and put them in relation with sparsely available affinity data in the literature, correcting for typographical errors, and translating form the various way of expressing affinity/inhibition. They also isolated the ligand in a mol2 file, with the protein (+water and inorganic components such as counter-ions) on their own in a novel PDB file. Only small organic ligand, or oligopeptides, due to the importance of peptido-mimetics for certain classes of drugs, were included. Resolution was set to the now standard level of 2.5Å or better, and finally the authors were left with 800 complexes. The entire database is accessible through a web-based interface, allowing SQL queries and even submission by third party researchers.

3.5 Conclusions

As shown, many databases are available to the computational chemist – and many more did not find space in this brief chapter. Each has peculiarities which makes it more suited to some tasks, or can be tweaked to suit the chemist's particular needs. For our work, we found the NCI Open database to obtain drug-like ligands, the CCDC/Astex set as a target collection to be used as a training/test set, and finally the SCOP database to analyse the diversity of the aforementioned CCDC/Astex set.

4 Data preparation

4.1 Introduction

The aim of this project is the creation of a scoring function able to accurately detect ligands that are active against the target of interest. As previously done by Smith et al.[49], the strategy chosen to obtain this function is to use a set of docked drug-like molecules as decoys, and train the scoring function to recognise the active ligand within this set. This is equivalent to parameterising our scoring function to maximise the enrichment. Hopefully, this will tweak the function to improve active detection within structure-based virtual screening.

To achieve this, we need some docking data to simulate a virtual screening experiment. The training procedure will then require our fitting protocol (a Genetic Algorithm, see section 5.6 or Goldberg's book [190]) to modify the coefficients for the combination of our scoring terms. This will in turn optimise the ranking position of those structures that we have flagged as actives.

In this chapter the focus is on explaining how we selected and prepared the starting data from which to generate the docking data then used for the parameterisation. Special attention is given to the ligand and protein choice, as this is of the utmost importance for the quality of the function obtained.

In the second part of this chapter the procedure behind the generation of docking data needed to parameterise our functions will be explained. Starting with the program used, passing then to the reasons for using distributed computing, and then on to the actual docking and the post-processing of poses, reasons for our choices, and for not having followed alternative routes, will be given.

4.2 Ligand dataset(s)

It is difficult to underestimate the importance of having the correct set of ligands for a structure-based virtual screening experiment. The final result of the experiment should be the selection of the most promising lead, often likened to finding a needle in a haystack [131]: but if the needle is not in there, then you will never find it. Therefore, the molecules you screen must have certain characteristics. The ensemble of these features is often referred to as *drug-likeness*, whose definition can be traced back to Lipinski and his Rule of 5 [138, 139, 191], but has now expanded to embrace other considerations, some general, some more specific to the protein(s) being targeted [191-193]. Some reviews on the subject have recently been presented by Muegge [194, 195].

In our case, there is no particular focus on a single protein class: Ideally the function will be largely applicable to both already known and yet to be discovered targets. During the training procedure, which mimics a SVS experiment, the actives would be already known from the crystal structure of the protein, purposefully inserted in the decoys' database in order to be retrieved. Our decoys, then, do not require any particular feature, except the absence of chemical groups incompatible with human physiology.

A good source for these molecules would be any of the ligand databases that we introduced in the previous chapter. So we could, in principle, assemble a library of some hundred thousands molecules and run it through our SVS protocol, docking it and looking for the active ones.

There was, however, limited time and computational power to screen our molecules, so the emphasis was on avoiding the screening of two or more very similar molecules, and then being forced to leave out others because of time constraints. The ideal solution is to select only molecules different enough to make the time spent on testing them worthwhile. The time saved can then be used to screen radically different molecules, therefore maximising the chemical space covered. This is the reason why clustering is frequently performed when selecting a ligand set to be used in screening: by highlighting similarities between the molecules a *representative* subset of the whole set can be chosen. Figure 4-1 illustrates an example of clustering procedure.

There is a natural trade-off between the ratio subset/set and the accuracy of the *representation* obtained, and of course the particular balance is determined by the specific conditions. In our case, there were no particular constraints to be imposed. Ideally, the known active ligand was to be hidden within a set as diverse and numerous as possible. The more chemotypes that were present, the better.

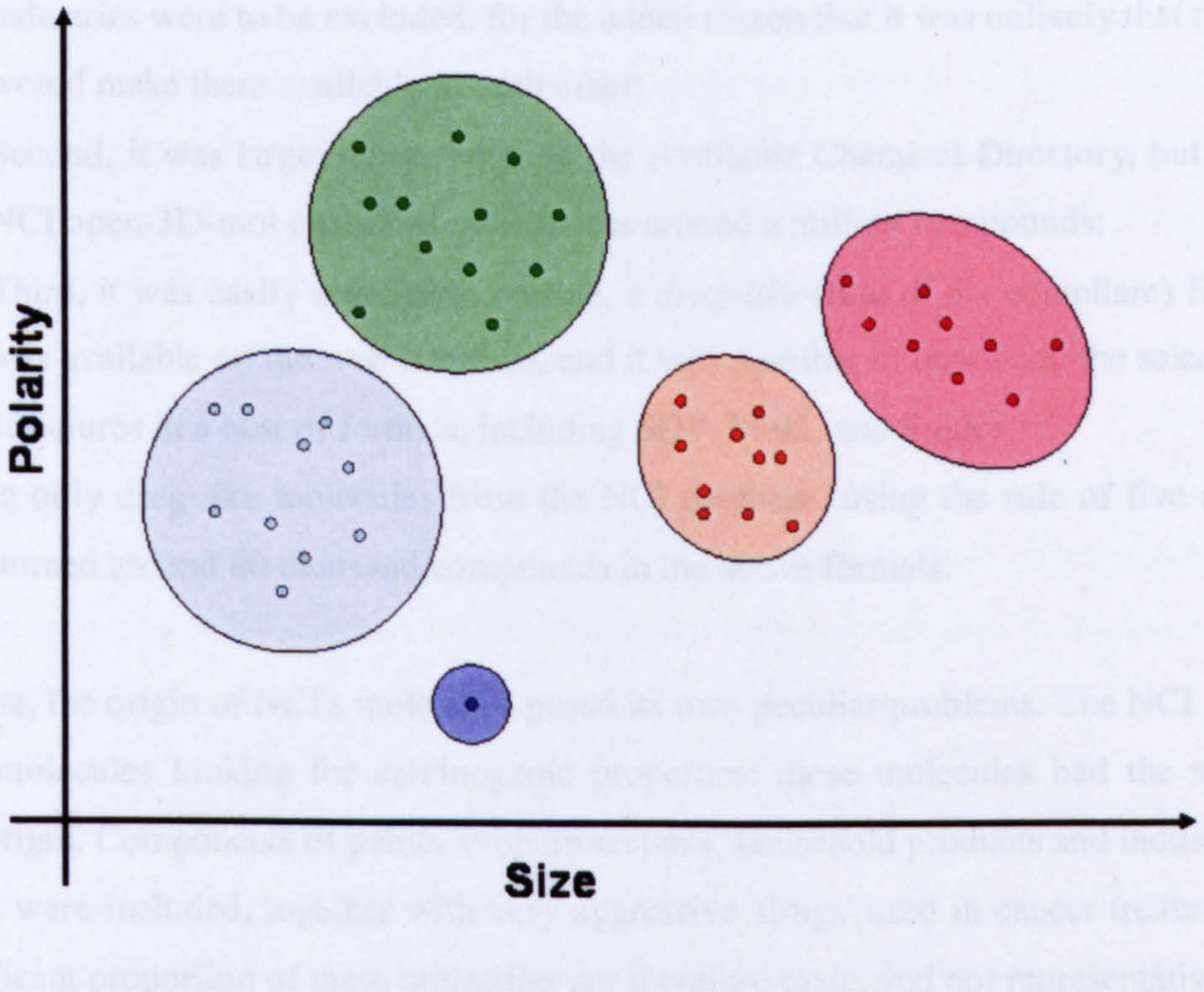


Figure 4-1: An example figure of clustering; a series of data-points (*e.g.*, molecules represented as descriptors) is partitioned in four diverse clusters (plus a singleton, in blue). A single or reduced set of representatives can be then chosen for each cluster.

Many diverse clustering methods are[JW8] used in Chemoinformatics [17, 196], hierarchical and otherwise, working from top to bottom or vice-versa, recursive or in a single pass[197]. Each has its particular strengths and weaknesses, and the choice of the best is sometimes quite subjective, or constrained by the availability of a proper implementation.

Before clustering, however, a number of molecules to be actually clustered are of course needed. Their selection will be the subject of next section.

4.2.1 NCI dataset

The NCI dataset was chosen as the primary source of structures, rather than WDI, ACD or any other set, for a very simple set of reasons:

- First, it was freely accessible. Commercial databases would forbid or otherwise restrict future availability in various modes, and the ability to publish it was a definite plus. In the same way, proprietary databases coming from our financing industries were to be excluded, for the added reason that it was unlikely that they would make them available to each other;
- Second, it was large: not as large as the Available Chemical Directory, but the NCI open-3D-mol contained nonetheless around a million compounds;
- Third, it was easily searchable on-line, a *drug-like* (rule of 5 - compliant) filter was available on the web interface, and it was possible to download the selected structures in a host of formats, including SDF, Mol2, and Smiles.

Selecting only drug-like molecules from the NCI database, using the rule of five as a filter, returned around 80 thousand compounds in the above formats.

Of course, the origin of NCI's molecules posed its own peculiar problems. The NCI was testing molecules looking for carcinogenic properties: these molecules had the most varied origin. Components of paints, crop protectants, household products and industrial solvents were included, together with very aggressive 'drugs' used in cancer treatment. A significant proportion of these molecules are therefore toxic, and not representative as drugs.

Since the problem of undesirable structures is commonplace in pharmaceutical industries, tools exist there to tackle it. One of the industrial project supervisors, Richard Lewis (then at Eli Lilly, UK, and now at Novartis, Basel, CH), used *in-house* developed filters to remove all reactive groups, substructures unsuitable for drugs or interfering with human metabolism. Additionally the filters, based on canonical smiles strings, removed around a thousand structure duplicates. Another 3 thousands entries were dropped because of errors in the structure, and finally the search for undesirable substructures only left slightly more than forty thousand (40229, to be precise).
The target, set by our availability of computing power, was to screen ten thousand at the most, and here is where the clustering mentioned in section 4.2 came into play. Again,

Richard Lewis and Ian Watson at Eli Lilly helped out with this. Initial attempts with commercial clustering software did run into memory problems, so in the end a simple sphere-exclusion algorithm was performed on the 2D descriptor space. The centroids of each exclusion sphere were selected as representatives. This returned around 10 thousand molecules, guaranteed not to be too similar, at least in the 2D sense.

A preliminary docking run on all these molecules highlighted some trouble with certain structures, which a visual inspection showed to be constituted by more than one molecule, some duplicates in crystallographic position, others a combination of an organic molecule plus a counter ion. Since, when working on them, GOLD[40] would try to respect geometrical constraints between atoms completely unconnected, this caused failures and a subsequent incorrect exit status for the program.

All such molecules were removed by a PERL filter which used a program written by us (**qtconnect**) which checked whether or not the mol2 file under analysis contained atoms which were not connected to each other, a clear indication that two or more disjointed *chains* were present within the structure.

Unfortunately, no check was performed on the protonation state of the molecules left, as it was erroneously assumed that the NCI stored them in a proper way, or that Lilly's filter had taken care of this. Ultimately, this caused such a carefully selected dataset to be dropped for the study.

4.2.2 Rognan's dataset

Fortunately, the NCI dataset was not the only dataset at our disposal. At the same time that the NCI selection was performed, steps were taken to create an independent ligand test set. Dr Didier Rognan, at Strasbourg's CNRS, France, provided us with a randomised sample of one thousand ligands out of its set of ten thousand drug-like molecules, whose selection was documented in 1999 within one of the first benchmark tests of different docking programs and scoring functions[57].

These ligands had been obtained by filtering a commercial database, the WDI; their protonation state and all other features important for docking had also been checked by the authors. Therefore, the confidence level on these molecules was higher than on the

previously described NCI selection. Being ten times less numerous, the computing costs would also be proportionally lower. It was then decided to dock these ligands with two GOLD settings, the most exhaustive default and the *library* default. Our intention was to use this docking data as a test-bed for the robustness of our function(s) when the sampling extent during the poses' generation was modified[JW9].

4.3 Protein dataset

At first, an initial attempt to select our own protein dataset was made. However, soon after completing the first easy selection on the basis of resolution, experimental technique, and such, the procedure was halted by the complications arising from the ligand selection. Therefore, the attempt was abandoned, and its results are not presented in this thesis. There is also another reason for abandoning our selection procedure. The publication of a collection of protein-ligand complexes specifically tailored for the training and testing of scoring functions, led us to embrace one of these, particularly, the CCDC/Astex dataset[140]. The reason for this particular choice was that the selection procedure followed was exceptionally thorough, even going back to the original diffraction data to re-generate the structures, and included treatment of the ligands to assess their protonation state and that of side chains close to the active site.

For the details of their selection procedure please refer to the original paper. Here is only a brief summary. Astex did not include any theoretical structure in the set, as they have limited value as a test-bed for docking programs and scoring functions. Astex did not impose any resolution filter, as this did allow the inclusion of old entries from the GOLD test set. However, a subset of the selection with a resolution threshold of 2.5 Å was created.

Because scoring functions for DNA/protein interactions tend to have their own characteristics, with different balances of the interaction terms compared to ligand/protein functions, entries for RNA- or DNA-binding proteins were discarded too. For peptidic ligands, though, a concession was made, as many important class of drugs act as peptido-mimetics to inhibit proteins whose substrate are peptides or parts of other proteins (*e.g.* proteases).

Scientists at Astex took also care of including only structures with known binding affinities, as this is (one of) the yard sticks by which scoring function performance is measured. Their final step was the thorough and painstaking job of checking the structures directly against the original diffraction data, correcting errors and re-assigning protonation states for ligands and side chains, taking into account their environment.

The final dataset is composed of 305 complexes. However, as said previously when talking about databases and scoring functions, the diversity of a set is of paramount importance. Astex did recognise this, so they compiled a *clean* 225 complex subset, where all duplicate complexes were removed. In here, *duplicate* meant complexes where the protein, the ligand, or their binding modes, were too similar. Within this subset, Astex selected a further subset, with only high resolution structures (same threshold as we used, namely 2.5Å or better).

This final clean and hi-resolution set was selected by us, and the PDB entries can be found in Table 4-1, third tier; the first and second tiers are instead devoted to the PDB codes of proteins excluded on the basis of similarity or resolution.

4.3.1 Further reduction of the set

By glancing at the next chapters, where the scoring functions are parameterised on the training set, the reader will be able to see that only 108 protein/ligand complexes were in fact used. This is a much lower number, and the missing proteins were discarded for a number of reasons: for some of them, the automated docking procedure of the decoys failed. For others, it was the charge assignment procedure of experimentally observed ligands that failed. Finally, as happened for five entries, their COLOs (Crystallographically Observed Ligand Orientations) had very bad clashes which were too strongly (and wrongly?) influencing our parameterisation. Further information on this can be found in the next chapters.

The CCDC/Astex dataset
Entries excluded because ``too similar'' to other complexes
1a07 1a0q 1a1b 1a1e 1a4k 1aha 1aj7 1ake 1b6n 1c2t 1cf8 1cin 1ctr 1ctt 1dbm 1die 1ela 1elb 1elc 1eld 1ele 1etz 1fbl 1fig 1ghb 1gpy 1hdy 1hef 1hti 1icn 1igj 1ivc 1ivd 1ive 1jao 1kno 1lkk 1lmo 1mcr 1ml 1mmb 1mnc 1mtw 1nsd 1pgp 1pha 1phf 1ppl 1qbt 1qh7 1ql7 1rbp 1sln 1srf 1srg 1srh 1stp 1tka 1tph 1vrh 1xkb 1yds 2cgr 2er7 2mip 2plv 2r04 2sim 3gch 3mth 3nos 3pgh 3ptb 4er2 4fab 4tpi 5p2p 6abp 6cpa 7cpa 8gch
Non-Duplicate entries excluded because of the low resolution (poorer than 2.5Å)
1acl 1cx2 1uvt 1acm 1bmq 1mcq 4cox 1fax 1hak 2lgs 2ypi 4cts 1dwb 1rt2 2mcp 1c12 1dbb 1dwc 1fgi 1ibg 1a9u 1baf 1dbj 1dwd 2phh 4fbp 1aaq 1eap 1tdb 2dbl 1bl7 1cqp 1ett 1hri 1ivq 1lyb 1ulb 2r07 3hvt 1acj 1ase 1did 1lyl 1uvs
Entries included in he hi-res, ``clean''(non-duplicate) subset
1a28 1a42 1a4g 1a4q 1a6w 1abe 1abf 1aco 1aec 1ai5 1aoe 1apt 1apu 1aqw 1atl 1azm 1b58 1b59 1b9v 1bbp 1bgo 1blh 1bma 1byb 1byg 1c1e 1c5c 1c5x 1c83 1cbs 1cbx 1cdg 1cil 1ckp 1cle 1com 1coy 1cps 1cvu 1d0l 1d3h 1d4p 1dd7 1dg5 1dhf 1dmp 1dog 1dr1 1dy9 1ebg 1eed 1ei 1ejn 1eoc 1epb 1epo 1eta 1etr 1ets 1f0r 1f0s 1f3d 1fen 1fkg 1fki 1fl3 1flr 1frp 1glp 1glq 1hdc 1hfc 1hiv 1hos 1hpv 1hsb 1hsl 1htf 1hvr 1hyt 1ida 1imb 1ivb 1jap 1kel 1lah 1lcp 1ldm 1lic 1lna 1lpm 1lst 1mbi 1mdr 1mld 1mmq 1mrg 1mrk 1mts 1mup 1nco 1ngp 1nis 1okl 1okm 1pbd 1pdz 1phd 1phg 1poc 1ppc 1pph 1ppi 1pso 1ptv 1qbr 1qbu 1qcf 1qpe 1qpq 1rds 1rne 1rnt 1rob 1slt 1snc 1srj 1tlp 1tmn 1tng 1tnh 1tni 1tnl 1tpp 1trk 1tyl 1ukz 1vgc 1wap 1xid 1xie 1ydr 1ydt 1yee 25c8 2aad 2ack 2ada 2ak3 2cht 2cmd 2cpp 2ctc 2fox 2gbp 2h4n 2ifb 2pcp 2pk4 2qwk 2tmn 2tsc 2yhx 3cla 3cpa 3erd 3ert 3gpb 3tpi 4aah 4dfr 4est 4lbd 4phv 5abp 5cpp 5er 6rnt 6rsa 7tim

Table 4-1: the PDB entries corresponding to the complexes in the CCDC/Astex dataset, divided in the subsets defined by its authors.

4.3.2 Classification by family

Once our 108 protein training set was selected, we decided to explore how the set was composed. The list of PDB codes corresponding to our set was run through the Structural Classification Of Protein (SCOP) database[175, 177], obtaining the list of domains inside each PDB entry, and their Family/Superfamily classification. For multi-domain proteins, the domain containing the active site was manually selected. The proteins were then grouped by superfamilies. This was done with two purposes: first, to make sure no family was overly represented in our set, to avoid bias in the scoring functions. Second, the intention was to select some subsets containing only member of a certain (super) family, to perform a tailored parameterisation. As can be seen in Table 4-2, a wealth of families is represented in the set, althoug[w10]h there probably is a greater than normal number of proteases: owing to the origins of CCDC/Astex dataset as a selected snapshot of the PDB, it carries with it the same typical bias toward proteases.

Only superfamilies with at least 4 complexes were selected as single-family sets. The reason for this was that parameterising on any lower number of proteins was not likely to be meaningful. However, owing to time restraints, there would be no time to perform the parameterisation on all of them. Proteases were grouped in a single 21-strong set, with a second set composed only of 12 serine proteases. Exopeptidases, immunoglobulin, kinase-like, dihydrofolate reductase, nuclear receptors, and neuraminidases all had their own set. An obvious inclusion, metal-proteases, would have been useful to compare against Zn-dependent exopeptidases (and other proteases too), but was left out because at the time we performed the partition there were no plans yet to use GOLD's version of CHEMSCORE, and the metal interaction term in it. We show in Table 4-3 the entries corresponding to each single-family[JW12][JW13].

Entries	Proteins
12	Trypsin-like serine proteases
10	Immunoglobulins
6	Protein kinase-like (PK-like) Lipocalins
5	Zn-dependent exopeptidases Dihydrofolate reductases Acid proteases
4	Sialidases (neuraminidases) Periplasmic binding protein-like II Nuclear receptor ligand-binding domain Metalloproteases ("zincins"), catalytic domain
3	Microbial ribonucleases LDH C-terminal domain-like
2	(Phosphotyrosine protein) phosphatases II FKBP-like FAD/NAD(P)-binding domain Carbonic anhydrase Xylose isomerase-like
1	Triosephosphate isomerase (TIM) TRAP-like (Trans)glycosidases Staphylococcal nuclease Ribosome inactivating proteins (RIP) Quinolinic acid phosphoribosyltransferase Proteins of incorrect, partial and unknown sequence Phospholipase A2, PLA2 N-terminal nucleophile aminohydrolases (Ntn hydrolases) Nitric oxide (NO) synthase oxygenase domain NAD(P)-binding Rossmann-fold domains Methanol dehydrogenase subunit Metallo-dependent hydrolases Lysozyme-like Kringle-like Insulin-like Glutathione S-transferase (GST) FMN-linked oxidoreductases Flavoproteins Enolase N-terminal domain-like EGF/Laminin Cytochrome P450 Concanavalin A-like lectins/glucanases ATPase domain of HSP90 chaperone/DNA topoisomerase II/histidine kinase Aromatic compound dioxygenase Alpha/beta-hydrolases Actinoxanthin-like Aconitase iron-sulfur domain

Table 4-2: Number of complexes present in our set for each SCOP superfamily represented.

Superfamily	Entries
Dihydrofolates	1aoe 1dg5 1dhf 1dr 4dfr
Immunoglobulins	1a6w 1c1e 1f3d 1fl3 1flr 1hsb 1kel 1ngp 1yee 25c8
Lipocalins	1bbp 1cbs 1epb 1fen 1lic 1mup
Nuclear Rec.	1a28 3erd 3ert 4lbd
Exopeptidases	1cbx 1cps 1lcp 2ctc 3cpa
Kinases	1byg 1ckp 1ei 1qcf 1qpe 1ydr 1ydt
Neuraminidases	1a4q 1b9v 1ivb 2qwk
Protease	1apt 1atl 1bma 1c5x 1d4p 1ejn 1ets 1f0r 1f0s 1hiv 1hos 1hyt 1ida 1mmq 1mts 1pso 1tmn 1tng 1tni 1tnl 3tpi
Serine Prot.	1bma 1c5x 1d4p 1ejn 1ets 1f0r 1f0s 1mts 1tng 1tni 1tnl 3tpi

Table 4-3: Protein superfamilies subject of single-family scoring function parameterisation (shown in Section Error! Reference source not found.): for each family considered, the related PDB entries are shown.

4.3.3 Subset generation

Another way of creating subsets of our training set is to divide them without taking into explicit consideration their structural relationship – in fact, the PDB codes for every subset are randomly selected. Generic subsets built this way are useful to investigate the influence of the size of the training set on the robustness of the scoring function. By chance, the number of proteins, 108, can be easily split in half, giving 54 proteins in each subset. Quartering is also possible, giving training/test subsets of 27 proteins each, or 81, corresponding to three quarters of the whole set. This way, a step-wise sequence of parameterisations is obtained, more and more comprehensive, and it becomes possible to test how robust the performance is depending on the size of the training set. Studies on the terms' preference by different subsets, overlapping with each other only in part or not at all, and on the variation of the associated terms' coefficients, could also be performed.

Our training set is composed of 108 complexes, sufficiently *diverse* from each other since they are all part of the *clean* and *hi-res* CCDC/Astex subset. Our SCOP family subdivision seems to confirm this (see previous section, 4.3.2). Therefore, there is good reason to believe that the scoring function we will generate based on the full set will not be biased. This is, of course, going to change when we decrease the number of proteins against which the function is trained. The smaller the set, the more likely that it will contain more than its share of a certain class/family, and none of some others. Of

course, accepting an automated random sampling of the full set to generate the subsets leaves us prone to some bias, but by repeating the procedure four times there was some possibility of detecting any such problem.

For example, a random selection of 54 out of 108 complexes gives us our first set: the second set is simply constituted by the complexes left unselected. These two sets were of course, because of their origin, not overlapping. A second random selection gave us a third and fourth set, also not overlapping with each other, but partially overlapping with each of the first two. Precise figures about the entries and overlaps[2] are given in the tables Table 4-4 and Table 4-5.

Set 54-A

1a28 1a4q 1a6w 1apt 1azm 1bbp 1c5x 1c83 1cbx 1cil 1cps 1dd7 1dg5 1dr 1ei 1eoc 1epb 1f0r 1fen 1fki 1flr 1glq 1hdc 1hiv 1hos 1hsb 1hyt 1ivb 1lah 1mld 1mrg 1nco 1nis 1pbd 1phg 1pso 1ptv 1qcf 1qpq 1slt 1tng 1tni 1tnl 1wap 1xid 1ydr 1ydt 25c8 2cmd 2fox 2yhx 3cpa 3ert 4dfr

Set 54-B

1ai5 1aoe 1atl 1b58 1b9v 1bma 1byb 1byg 1c1e 1cbs 1ckp 1coy 1cvu 1d0l 1d3h 1d4p 1dhf 1ejn 1ets 1f0s 1f3d 1fkg 1fl3 1hsl 1ida 1kel 1lcp 1ldm 1lic 1lst 1mdr 1mmq 1mts 1mup 1ngp 1poc 1qpe 1rnt 1snc 1tmn 1tyl 1yee 2aad 2ack 2ada 2ctc 2pk4 2qwk 3erd 3tpi 4aah 4lbd 6rnt 7tim

Set 54-C

1a28 1a4q 1a6w 1aoe 1azm 1b58 1bbp 1byg 1c1e 1cbx 1cps 1d0l 1dg5 1dhf 1dr 1ei 1eoc 1f0s 1fl3 1hos 1hsb 1ivb 1lah 1ldm 1mld 1mup 1nco 1ngp 1nis 1pbd 1phg 1poc 1pso 1ptv 1qcf 1qpq 1snc 1tmn 1tnl 1tyl 1wap 1ydr 1ydt 2ack 2ada 2fox 2pk4 3cpa 3erd 3ert 3tpi 4aah 4dfr 6rnt

Set 54-D

1ai5 1apt 1atl 1b9v 1bma 1byb 1c5x 1c83 1cbs 1cil 1ckp 1coy 1cvu 1d3h 1d4p 1dd7 1ejn 1epb 1ets 1f0r 1f3d 1fen 1fkg 1fki 1flr 1glq 1hdc 1hiv 1hsl 1hyt 1ida 1kel 1lcp 1lic 1lst 1mdr 1mmq 1mrg 1mts 1qpe 1rnt 1slt 1tng 1tni 1xid 1yee 25c8 2aad 2cmd 2ctc 2qwk 2yhx 4lbd 7tim

Table 4-4: The four 54-protein sets by PDB entries.

[2] Note that, *by construction*, set 1 overlaps with set 4 the same amount that set 2 overlaps with set 3. The same holds true for the pairs A - C and B - D.

Overlaps	Set 54-A	Set 54-B	Set 54-C	Set 54-D
Set 54-A	100 %	0 %	32(59.3 %)	22(41.7 %)
Set 54-A	-	100 %	22(41.7 %)	32(59.3 %)
Set 54-B	-	-	100 %	0 %
Set 54-D	-	-	-	100 %

Table 4-5: The four 54-protein sets: percentile overlaps between them. Sets A and C (and B - D) overlap 32 out of 54 entries. Sets A and D (and B - C), 22 out of 54.

Set 81-A

1a28 1a4q 1ai5 1aoe 1apt 1azm 1b58 1b9v 1bbp 1bma 1byb 1byg 1c1e 1c5x 1cbx 1cil 1ckp 1coy 1cvu 1d0l 1d3h 1d4p 1dd7 1dg5 1ei 1ejn 1eoc 1ets 1f0r 1f3d 1fen 1fki 1fl3 1flr 1hdc 1hos 1hsb 1hsl 1hyt 1ida 1kel 1lah 1lcp 1ldm 1lic 1lst 1mdr 1mld 1mmq 1mts 1ngp 1pbd 1phg 1poc 1pso 1ptv 1qcf 1qpq 1rnt 1slt 1snc 1tmn 1tni 1tnl 1tyl 1wap 1xid 1ydt 25c8 2ack 2ada 2cmd 2ctc 2fox 2pk4 2yhx 3cpa 3ert 3tpi 4aah 4lbd

Set 81-B

1a28 1a4q 1a6w 1aoe 1apt 1azm 1b58 1b9v 1bma 1byb 1byg 1c1e 1c83 1cbs 1cbx 1cil 1coy 1cps 1d0l 1d3h 1dhf 1dr 1ejn 1eoc 1epb 1ets 1f0r 1f3d 1fen 1fkg 1fki 1fl3 1flr 1glq 1hdc 1hos 1hsb 1hsl 1ida 1kel 1ldm 1lic 1lst 1mdr 1mld 1mmq 1mrg 1ngp 1nis 1pbd 1phg 1poc 1pso 1qcf 1qpe 1rnt 1snc 1tmn 1tng 1tni 1tnl 1tyl 1xid 1ydr 1ydt 1yee 25c8 2aad 2ack 2ada 2ctc 2qwk 2yhx 3cpa 3erd 3ert 3tpi 4aah 4dfr 4lbd 6rnt

Set 81-C

1a28 1a4q 1a6w 1ai5 1apt 1atl 1b58 1bbp 1bma 1byg 1coy 1cps 1cvu 1d0l 1d4p 1dd7 1dhf 1dr1 1ei 1ejn 1eoc 1epb 1f0r 1f0s 1f3d 1fen 1fkg 1fl3 1flr 1glq 1hiv 1hos 1hsb 1hsl 1hyt 1ida 1ivb 1kel 1lah 1lcp 1lic 1mdr 1mld 1mmq 1mrg 1mts 1mup 1nco 1ngp 1nis 1pbd 1phg 1pso 1qcf 1qpe 1qpq 1rnt 1slt 1snc 1tmn 1tng 1tni 1tnl 1wap 1xid 1ydt 1yee 2aad 2cmd 2ctc 2fox 2pk4 2qwk 2yhx 3cpa 3erd 3ert 3tpi 4aah 4dfr 6rnt

Set 81-D

1a28 1a6w 1ai5 1aoe 1apt 1atl 1azm 1b58 1bma 1byb 1byg 1c1e 1cbs 1cbx 1cil 1ckp 1coy 1cps 1cvu 1d0l 1d4p 1dg5 1dhf 1ei 1epb 1ets 1f0r 1f0s 1fkg 1fki 1fl3 1glq 1hdc 1hiv 1hos 1hsb 1hsl 1ida 1ivb 1kel 1lah 1lcp 1ldm 1lic 1lst 1mdr 1mld 1mmq 1mrg 1mts 1mup 1ngp 1nis 1pbd 1phg 1poc 1pso 1qcf 1qpe 1rnt 1slt 1snc 1tng 1tni 1tnl 1wap 1ydr 1ydt 1yee 25c8 2ack 2ada 2cmd 2ctc 2fox 2pk4 2qwk 2yhx 3cpa 4dfr 4lbd

Table 4-6: The four 81-protein sets: PDB entries.

The first two sets were further split in two to generate four 27 proteins training sets. These sets will not overlap at all with each other. However, four other training sets were generated with 81 complexes each (see Table 4-6 for the entry lists and Table 4-7 for the overlap data). The entries left from these subsets were the basis for four more 27-protein sets, partially overlapping (by various measures) with the first four. It must be noted that, in contrast to the 54 protein sets, all 81 protein sets overlap with each other, for obvious reasons. Of course, they overlap partially with three out of four of the new 27 protein sets, the exception being their complementary selection.

Overlaps	Set 81-A	Set 81-B	Set 81-C	Set 81-D
Set 81-A	100%	61(75.3 %)	58(71.6 %)	61(75.3 %)
Set 81-B	-	100%	59(72.8 %)	62(76.5 %)
Set 81-C	-	-	100%	61(75.3 %)
Set 81-D	-	-	-	100%

Table 4-7: The four 81-protein sets: percentile overlaps between them.

4.4 Docking Experiments

4.4.1 Why Use GOLD?

GOLD was, at the time this work was performed, one of the best possible options for ligand/protein docking[57]. We have already met this program in section 1.10. It was consistently ranked in the first position during independent tests measuring the ability of docking programs to reproduce known modes of binding, or simulating SVS experiments[60, 61, 198]. For this reason it was chosen to replace DOCK, used in the previous work by Smith et al.[49]. Accurate as it may be, GOLD is not however the fastest docking program available[199]. It is true that by selecting the *library screening* setting, docking times of one minute per ligand can be achieved. This, though, comes at the price of limiting the configurational sampling. Both standard and library settings were used during this work: the poses generated with the library settings, however, did not seem to pose a hard enough challenge to the GA algorithm. It was in fact very easy to generate a function able to discriminate quickly-generated decoy poses from experimental crystal structures (data not shown) – the sparser sampling of the solution space was the most likely culprit for that, as recently shown in literature [200]; all decoy poses generated with the library settings were dropped from the study and not considered any further.

For GOLD to successfully dock a ligand within an active site it is necessary to allow an adequate sampling of the configurational space for the ligand/protein complex. Sampling need not be exhaustive, as this would nullify the efficiency advantage of using a GA approach[JW16]. GAs are not 'as good as' the exhaustive search in terms of completeness of sampling, but they are almost as complete, and far more efficient. They are able to provide a suboptimal solution very close to the actual optimum with only a fraction of the sampling required for the exhaustive method – usually this is good

enough for our purposes. Times required are of the order of 10 to 30 minutes per ligand, depending on the ligand and receptor size and complexity. In the light of the study just cited [200] and the warning subsequently issued by CCDC [201], a manual modification of the library settings to allow for roughly 7 times more GA operations per ligand (1 instead of 1500) might be able to provide sufficient sampling, with substantial speed-up compared to the full settings used here. This is as of today an avenue open to future investigation and may be explored in the future.

We were forced to use quite large docking sites (up to 24 Å of radius) since the length of the ligands was quite variable and we wanted to allow enough space for large ligands to be accommodated. Since our initial idea was to dock about 10 thousands ligands, against 150 odd proteins, this would have required roughly 10,000 ligands × 10 mins/ligand × 10 poses/ligand × 150 proteins = 150 millions minutes = 2.5 million hours. That is the best part of 285 years on a single hi-specification personal workstation.

The obvious choice is to use many more computers to perform the docking. This does not pose any particular problem, as docking is a trivially parallelisable process. In contrast to other simulation techniques such as molecular dynamics, where efficient parallelisation is not trivial due to the partitioning of the simulation box and nonbonded energy calculation, the only thing required during a virtual screening experiment is to submit different ligands to different machines and then harvest the data together. Some post-processing is obviously required to rebuild a general ranking list.

4.4.2 Why Use Distributed Computing / Why Condor?

Personal computers are often used by scientists for computationally intensive tasks, and modern gaming also requires substantial resources; most of the time, though, computers are in the hands of people that only do internet-browsing, or word-processing, or other low computational intensity tasks. Even scientists' personal workstations are often idle, with few bursts of activity during which they are used 100%. On average, a modern personal computer is idle for more than 90% of its time, and excluding energy saving consideration, this is an awful lot of computational resources being wasted without any justification. Distributed computing is a parallel computing paradigm that harnesses these previously untapped resources, that has been brought to the attention of the

general public by successful internet-based computations such as the SETI@HOME project and the GIMPS (Great Internet Mersenne Prime Search) projects. Projects like the Folding@home (http://www.stanford.edu/group/pandegroup/folding/) and Anthrax Research Project (http://www.grid.org/projects/anthrax/index.htm) demonstrated the possibility of successfully applying this protocol to computer-aided drug design. Many of these projects have now decided to pool together their resources and are moving from a proprietary operating environment to a shared one, open-sourced and with robust security implemented from the very beginning. This communal project is called BOINC, which stands for "the Berkeley Open Infrastructure for Network Computing" (http://boinc.ssl.berkeley.edu/).

4.4.3 How does distributed computing work?

Distributed computing is usually based on a server-client architecture: the personal workstation runs a client program which connects to the server, downloads chunks of relevant data, and then starts the calculation. When this is complete, the client returns the results back to the server, which incorporates them in its data/knowledge base. A scheduling system is installed on the server side, to keep track of units already submitted, the resubmission of *expired*[3] work-units and so on. A system of credits awarded to participants can greatly increase the participation of the general public, although it raises the problem of 'cheats', *i.e.* having your client hacked to report a finished work-unit while truly returning unfinished, or worse, *junk* data. In this case, careful server-based procedures to validate the results must be implemented, together with hardened client architecture.

There is also the fact that participating machines range greatly between hi-spec, latest generation machines, and old wrecks forgotten in teaching labs or on some secretary's desk. Such different machines will have great difficulty in advancing the run at the same speed, and on top of this there is the added problem of bandwidth usage.

Distributed computing is very good for coarsely parallel problems, such as docking, but not really suited to molecular dynamics or other 'holistic' simulation techniques, due in first instance to bandwidth limitation for data exchange. As an example, parallel

[3] That is, a work-unit that has passed its expire date. It is often necessary to impose this, especially in the case when you need all your intermediate data before continuing.

molecular dynamics can be achieved by partitioning the simulation box into smaller parts (*e.g.* splitting a cube into 8 sub-cubes with half the dimensions): but running the simulation this way requires a great deal of information exchange every few step of the simulation. This great utilisation of network resources requires specialised hi-bandwidth links between the machines: in cases where the cpus are not equally powerful, or have unequal workloads, the need for information exchange slows down quick nodes: they end up waiting for data they need, which other slower nodes have not finished generating. An obvious solution to this is to only use machines of very similar performance, whenever possible. Scheduling servers should then allow people submitting jobs to select appropriate nodes. However, the load imposed by the users on the different computers can vary and it is not predictable *a priori*, and therefore imbalance will always be present in distributed computing.

Excessive network usage could also slow down the responsiveness perceived by the computer's owner, who may decide to opt out of the project. A distributed computing client should therefore be *invisible*, and only kick in when the computer is not being otherwise used. The program should release the machine as soon as requested by the user. Finally, spending some time generating nice, flashy graphics is recommended, as this increases the possibility of the user selecting the client as the default screensaver.

For all these reasons, writing a distributed computing architecture is not something that can be achieved by amateur programmers overnight. It is more common to see them released from within Computer Science research labs, or other specialised entities (such as hi-tech internet Interest Groups, for open-sourced, collaborative projects).

Notwithstanding all these difficulties, distributed computing can be a useful and cheap alternative to dedicated Linux Clusters, although with the limitations ensuing from the limited bandwidth. As with all instruments, you should only use it if it is suited to your particular problem and software. A number of commercial platforms are available, offering different advantages and features, such as multiple architecture environments[4], encrypted data transfer and storage[5], web-based control interfaces[6] and storage on latest

---

[4] The client runs in Linux, Windows and Mac Os.
[5] This is particularly useful in industry, when screening 'in-house' databases on secretaries' computers, who should not be able to access confidential data.

generation relational databases for easy data retrieval. A free, open-source solution is offered by the University of Wisconsin with Condor[202, 203], of which an experimental setup of around 1500 (400 at the time the work was performed) machines is now in place at University of Southampton. The machines running the Condor client were mostly devoted to internet use by undergraduate students in halls, laboratories and libraries. The typical configuration was Pentium3@1 GHz with 256 MB of RAM. The only other alternative to Condor within the University was the United Devices' Platform™, but previous experiences were frustrated by data-retrieval problems. This, and the fact that BOINC had not yet been released at the time of our decision[7], made for a very easy decision. GOLD was run on the University of Southampton Condor cluster using a six-month, 10 thousand processes licence kindly provided by CCDC in exchange for a detailed resume of the setup used[JW17].

The modifications made to fit GOLD in the special CONDOR environment were indeed minimal, and are described shortly later on (Section 4.6.1).

4.5 Docking Setup

The complex selection procedure has been exhaustively presented in this chapter[JW18], so we describe it only briefly here. All proteins selected for the docking runs were part of the CCDC/Astex dataset and, in particular, of the so-called *clean* subset (see Table 4-1 for a list of the PDB entries). This word identified a subset with both high-resolution structure (better than 2.5 Å) and where no duplicate (or too similar) complexes were present. The subset was therefore as diverse as possible. However, a certain bias may still be possible, due to the different abundance of various protein families within the set itself. This is unavoidable, and arises from the difficulty in obtaining appropriate crystals for different families of proteins. Configuration files for GOLD were prepared as follows: the original **gold.conf** [8] provided accompanying the CCDC/Astex dataset was modified to perform an exhaustive sampling rather than the default three-times speed up. The radius of the active site was increased to 24 Å, to make sure it would accommodate the largest ligands in our set. Some proteins, which showed errors at runtime with this setup, had their active site radii decreased to 16 Å, on

---

[6] So that even non-geeks can use them.
[7] Not that we were aware of any plans to use BOINC within the University. Indeed, the University of Southampton still only supports Condor.
[8] This is the configuration file which contains instructions for GOLD.

the basis of our suspicion that the errors were generated by GOLD not being able to cope with incomplete PDB structures, but rather *'cuts'* of the proteins around the active site, with dangling valences for the atom whose bonds were cut at the interface. For some proteins in the CCDC/Astex dataset, only active site coordinates are available. Another modification to the original configuration files was to flip back to 0 the flag which signalled GOLD to stop when the first three poses generated were all within a given RMSD cut-off (1.5Å). This was done to ensure that an equal number of poses (10) was generated for each ligand.

4.5.1.1 Library run settings

The **gold.conf** files were rewritten so as to correspond to the gold default 'library' setting, with a much less exhaustive configurational sampling. This was done to allow us to explore the influence of different sampling levels on the scoring function parameterisation. These results were initially used for tests of the parameterisation routines in parallel with the exhaustive sampling: preliminary results made the case for proceeding with the exhaustive settings only; the GA's task was much harder in this case, than when it had to discriminate between crystal poses and rushed docked poses. Owing to time restraint, the library runs were followed up no further, and no results are presented for scoring functions parameterised on them.

4.6 Docking Runs

During the actual docking runs, my personal workstation (herein Scarlet) acted as a server for the submission of jobs, *via* a PERL script. The recovery of the output data was instead handled by a Condor file-server. The Condor server, running on a dedicated University machine, received Scarlet's requests, temporarily stored the input data, and then dispatched the data and executable to the selected node of the grid. After completion of the work-unit, data would be stored temporarily on the server, before being retrieved by scarlet and stored in the correct place within a hierarchical system of folders. The submission script was self-balancing, only allowing a certain number of jobs to be queued on the server, to avoid overload and then unnecessary human intervention. Also, the impact of several hundred file transfer requests to and from scarlet had to be considered.

4.6.1 Specifics of Gold on Condor

The setup of Gold on Condor turned out to be relatively easy, only requiring a very minor modification to GOLD's configuration files[9], and the inclusion in the work-unit of the dynamically loaded libraries (.dll files) from the windows' version of GOLD. The number of ligands was chosen so that the work-unit elaboration time was not too short, requiring continuous data transfer, nor too long, risking interruption by some user rebooting the machine. This last event was very likely as it was the default behaviour for a student's public workstation. The work-unit for a single gold job therefore consisted of:

- **gold_win32.exe** - GOLD's executable for Windows machines;
- **gold.dll** - GOLD's basic libraries;
- **goldscore.dll** - GOLD's scoring function library. Substitute **chemscore.dll** in case you want to use the alternate scoring function in GOLD 2.1;
- **gold_licence** - The licence file for gold, without which GOLD would not run. It tells GOLD which server need to be contacted in order to receive authorisation;
- **gold.conf** - The file read by GOLD to acquire information on the docking that it must perform; it contains protein and ligand file names, active site position, size and shape, GA and scoring settings;
- **protein.mol2** - a mol2-formatted (Sybyl[JW19]) file, containing atomic coordinates for the protein target;
- **ligands.mol2** - a mol2 file, containing atomic coordinates for 10 ligands to be docked in the protein's active site;

All these files were created or copied/linked to a purpose built directory, and then bundled together by the submission script. After the work-unit was elaborated, the results were returned in the same directory whence the request was sent.

Possible alternatives to minimize further the network transfer would be to keep the protein file on the node, and always submit to that machine the ligands to be docked to that particular protein. This would avoid having to transfer the protein back and forth

---

[9] That is, just remember to leave two (instead of one) white-spaces between the ligand filename and the number of poses requested.

every time a work-unit starts. However, this would require modifying the Condor settings, since by default everything contained in the folder where the work-unit runs is deleted after completion of the same, to avoid filling up hard-drive space with dead (and possibly confidential) data. But we did not notice any network overhead except perhaps for scarlet, which handled all the file-serving and submission tasks, so we decided to keep things as simple as possible.

4.6.1.1 Performances and Timings

In the six months that GOLD was available to us (February to July 2004) we were able to perform the re-docking of the true ligands, the docking over the decoy Rognan dataset of 1000 ligands with both accurate and library settings, and the docking of 7000 out of 1 NCI ligands with accurate settings, producing 10 poses per protein-ligand complex

4.7 Conclusions

In this chapter, the data sources for our experiments, and the way they were prepared and treated, were presented. Also, the rationale behind the choices for docking programs and protocols were detailed. It was shown how these protein and ligand sets were used to actually generate our parameterisation data.

The next chapter will discuss the details of the scoring function development, such as programming language, general framework structure, and terms included.

5 Scoring Function Development

5.1 Introduction

Once the docking data were generated as described in the previous chapter, they can be used to parameterise a scoring function in such a way that the Crystallographically Observed Ligand Orientations (COLOs), or docked structures of the known actives, are ranked the topmost when compared to docking-generated structures of otherwise inactive compounds (the decoys). However, a necessary step before this is to assemble a panel of scoring terms, and to compose them in a meaningful way. These scoring terms are applied to the three-dimensional poses generated by the docking program. The outcomes are then summed using different multiplicative coefficients, and these coefficients are then optimised to attain the optimal ranking of our known active poses. The scoring function is, as a matter of fact, embedded within a program able to read the ligand and protein molecules from a file, evaluate all the necessary terms, and return each of the scores as the final step.

This chapter is devoted to the explanation of how the scoring function was designed and, subsequently, implemented[JW20]. The actual program which computes and runs the scoring function, and how to use it, is also introduced[JW21]. Input and output files for both the actual scoring and the Genetic Algorithm (herein GA) parameterisation program are presented, in a level of detail which should be sufficient to act as a short user manual. Finally, the Genetic Algorithm techniques will be presented, and their practical implementation in this project explained.

5.2 Framework Structure

To perform chemically-related tasks, a program must be able to *understand* chemical entities, such as atoms, bonds, molecules and so on. Although there were a few tools already available for the task, it was decided to code a new one from scratch in the intention of turning the coding effort into an exercise in software planning and programming. By having to tackle some problems related to the proper planning of a

medium-sized software program, some useful lessons were in fact learned. The most important is probably that, for the project to succeed, one must fully understand and not underestimate the extent of the effort needed. With hindsight, it may have been better to have chosen an alternative route, such as taking advantage of an already existing framework. It is however true that this would have entailed a steep learning curve at first, to understand the framework structure not of our own design. Also modification to suit our needs may have been more or less difficult depending on the framework chosen.

5.2.1 Why object oriented? Why C++?

To build the framework, one has first to understand properly what it has to do: in this case, what was needed was a code able to read 3D molecular files, most likely .pdb or .mol2, of both protein and ligands, and perform single-point energy evaluations, without the need for more complex things such as energy minimisations, time-integration or other features necessary for MC and MD codes.

Python[10] makes clean and tidy code, easy to read and maintain, its main philosophy: it has a modern object implementation, and other desirable features one could possibly want; in recent years, it has been used to develop molecular modelling software such as PyMol[204, 205]. Moreover, it was being used by some in the group with satisfaction. As an interpreted language however, this language does have strong limitations when performance is considered, unless proper interfaces to C/FORTRAN optimised routines/libraries are provided. The main, and most computationally intensive, part of a scoring program is the energy evaluation itself, and this part is therefore best coded in C or FORTRAN. Rather than using Python for file-reading and having to interface it with C or Fortran for the computationally demanding energy evaluations, it was decided to code everything in one of the two high-performance programming language.

One of the two choices considered was FORTRAN, well established in the scientific/academic environment, with a wealth of scientific libraries optimised over its 50 years of existence. Another options considered was C, possibly the most widely used language for professional applications, from operating systems to videogames. Its

---

[10] © 2001-2005 Python Software Foundation. Copyright 1991-1995 Stichting Mathematisch Centrum, Amsterdam.

younger brother, C++, shares C's syntax and general features, but has been extended to embrace the Object-Oriented Programming (herein OOP) paradigm.

Objects are particularly useful when coding simulations of physical phenomena, because they offer a further abstraction level. Humans naturally think in terms of objects, a chair, a car, and so on, whereas *procedural languages* (such as C and FORTRAN) think in terms of functions and variables. For example, when coding a chemical simulation in FORTRAN 77, the natural choice would be to allocate all position of N atoms to a 3×N array of real numbers, and all their formal charges and atomic numbers to two 1×N arrays, composed of integers - partial charges and atomic weights would be stored in real numbers' arrays. Then, the system's properties would be computed with a function (C), or subroutine (FORTRAN), which takes in these numbers and returns a value.

In OOP, instead, an *atom* is an entity by itself, with its own *internal* variables, accessed solely through an interface it shows to the outside world. This interface is composed of "methods": these are similar to C's functions or FORTRAN's subroutines, but they belong to the object itself, and not to the main program. This allows a method to be used in different programs without having to explicitly re-define and re-implement it every single time.

In contrast to subroutines, methods with the same name can perform different things for different objects. A method called type() belonging to an *instance*[11] of the class atom would check bonds connected to the atom, and compute and return its atom-type from this information. A method with the same name but belonging to an instance of the 'bond' class would perform different operations, to return a bond type, rather than an atom type.

In general, an object's method, when called, returns information on its internal state, modifies the same, or evaluates the result of some *interaction* between the object it belongs to and others, *i.e.* a coulombic interaction with another atom, whose reference is passed as an argument to the method.

---

[11] an *instance* is a particular case of the object, just like the chair on which you are sitting is just an instance of the class *chair*.

It is easy to see the advantages of OOP; first, the higher abstraction level compared to FORTRAN and C allows for more elegant code, where the relation between entities and variables is clear, and how the different parts interact with each other. Second, this also makes for easier maintenance of the code, not just because the code is more readable, but because internal changes to an object do not affect others, as long as the object's interface has not changed.

This greater modularity is paid for with a modest performance hit, compared to FORTRAN or plain C, but nowadays compilers can efficiently optimise OO code, and careful construction of time-intensive algorithms can further improve this – whole books have been devoted to such tricks.

Keeping in mind all this, the decision was made to write our code in C++. At the time of coding, we were not aware of the existence of another chemistry-related C++ framework, developed by the McCammon group at University of California - San Diego. However, later attempts to obtain it from the authors, for documentation purposes, were unsuccessful.

5.2.2 Basic Data Structure

The code was written in C++ using QT[12] libraries for file and string handling. QT also provided structures and interfaces for all storage related classes, such as linked lists and dictionaries, often extended to meet our needs. The application (the program) was built using **KDevelop**, a free and open-source Integrated Desktop Environment. KDevelop took care of tracking all dependencies between the source files, generated makefiles through the **qmake** application, and interfaced with compilers/debuggers. Compilation was performed using the C++ compiler **g++**, included in the GNU Compiler Collection (known as **gcc**). A number of chemical classes were created, plus some other classes which served ancillary purposes (*i.e.* vectors), and C functions to perform simple linear algebra operations (*i.e.* cross and dot product). The rest of this chapter is devoted to a more detailed explanation of all these classes.

---

[12] More documentation and the source code can be found at http://www.trolltech.net.

5.2.3 Universe

Universe is a container class which can only be instantiated once within the program. It is essentially a data-structure that keeps track of every atom, molecule, bond that is ever created, destroyed or moved from one molecule to another. It does not have other methods except those answering to queries about whether or not an atom/bond/molecule is present and/or defined within itself. However, there is nothing to stop new methods from being implemented. An example could be adding a time frame, in the unlikely case someone wanted to expand the framework to be able to provide basic molecular dynamics capabilities.

5.2.4 Atom

This is one of the most fundamental classes defined in the framework. The data structure contains variables corresponding to physical quantities relevant for an atom, such as van der Waals radius, charge, position, a series of flags recording whether or not it is bonded, to how many other atoms, *via* how many bonds; all atoms to which it is bonded and all bonds through which this occurs are also referenced through lists of pointers[13], to allow for a quick scan during interaction methods. The object Atom contains three loosely separated series of methods: a series of initialisation/reading methods to set/return all the variables; basic methods to compute distances from other atoms or points, and more elaborate methods able to compute pairwise interactions such as hydrogen bonds, van der Waals and coulombic interactions. Methods at each level use those belonging to the lower levels to optimise readability and maximise modularity.

5.2.5 Bond

In contrast to atoms, which in our framework are the cornerstone of chemical interactions, the Bond class is defined in a very simple way: as we are only concerned with non-bonded interactions, they never change during a run. Therefore, a Bond object only contains a pointer to each of the atoms which it connects, its own length, its type (as supported in mol2 format: single, double, triple, aromatic, amide), and methods to

---

[13] A *pointer* is a way of accessing a variable's value based on its actual position in the computer's memory, rather than *via* the pre-determined variable's name. It is particularly efficient when accessing arrays. Pointers are characterised by the use of the referencing(*) and dereferencing operators (→ - points to). *E.g.*, *float pippo* identifies a real number stored in a variable called pippo; *float* pippo_p* is instead a pointer to a memory location where you store a float (the pippo float, for example).

compute/return all these variables. Most computation involving bonds is in fact done in an *atom-centred* way, or through higher level classes which will be introduced later.

5.2.6 AtomList

AtomLists are the work-horses of the framework. They are a specialised class derived from a standard QT linked list to pointers, *QPtrList<atom>*. They inherit all characteristics (methods and variables) of their parents; a few more variables have been added, and quite a lot of chemically related methods. To this class fall most of the computing routines, such as Generalized Born Surface Area [206, 207] (GBSA therein), hydrogen bond detection, surface calculation, and coulombic interactions.

These methods, when needed, call the corresponding atom methods, so that the structure of the routines is clear to understand, and optimisation is straightforward, when profiling data is available. An example of this *nested* calling is given below in Table 5-1.

```
// the method's definition
double AtomList::ElectrStat (atom* OTHER) {
      double EsNrgVal = 0; // setting to 0 the value to be returned
      // setting the atom counter to the first in the list
      atom* ATOM = this→first(); // here's a pointer to an atom being defined
      while (ATOM != 0) { // a cycle over all atoms
            // compute the energy, and sum it up with the main value
            EsNrgVal += ATOM→ElectrStat(OTHER);
            // and note how you access a pointer's methods with the → arrow
            ATOM= this→next(); // go to the next atom in the list
      }
      delete ATOM; // delete the temporary reference to the n-th atom in the list
   return EsNrgVal; // and return the final value to the main() program
\}
```

Table 5-1: How the Atomlist's ElectrStat method calls the corresponding atom's method in a cascade.

5.2.7 Residues

Sometimes, molecules are composed of different units, such as amino acids in proteins, or sugars, bases and phosphate groups in nucleotides. To account for this, a class called 'residue' was created. Its internal variables are an AtomList, the residue's own name (*e.g.* **GLY-421**), and pointers to the residues immediately preceding and following it in the chain, if any. Also, the centres of mass and geometry can be stored as vectors to be used during some computations. An example of a commonly used method is

isStandard( ), which returns TRUE if the residue is in the list of standard amino acid residues commonly present in proteins, and FALSE otherwise.

5.2.8 BondList

BondLists are also derived from standard QT linked lists to pointers, but like Bonds, and unlike AtomLists, do not have many computing methods assigned. They mainly act as reference storage for Atoms, AtomLists and Molecules to operate on. As an example, whenever an oxygen atom needs to be assigned hydrogen bond donor properties, its BondList is scanned to make sure that at least one hydrogen atom is present.

5.2.9 Molecules

Molecules are, as might be expected, one of the central classes in the framework. They consist of an AtomList and a BondList, to store pointers to related objects, and a dictionary of residues (**QDict<residue>**[14]), to store pointers to the residues of which the molecule is composed. This is of course valuable in the case where the molecule is a protein, or a complex molecule such as ATP, are given – each constituent (adenosin, phosphate groups) will have the same molecule identifiers, but a different residue name. Interaction methods or those performing the selection of particular subsets of atoms, *e.g.* constrained to distance from a point/atom/residue, usually call the corresponding AtomList methods. Other methods allow for direct reading from and printing to text files in SYBYL mol2 format.

5.2.10 Ancillary classes: i.e. Grid, Dictionary, etc.

A set of different ancillary classes have also been defined. These are not implicitly *chemical* objects, but help with computation or information storage. Examples are 3D-grids to store energy values which may then be interpolated.

These grids can, given a centre, maximum spans along the three axes, and grid spacing, be computed using their creation method, and then written/read to/from a file. The latter option is quicker than re-computing them and saves, when possible, plenty of time. Once fully defined within the program, grids are usually passed as references to

---

[14] In C/C++, the notation <> usually declares a template class, *i.e.* a generic class which can then be specialised: **QDict<*float*>** is a dictionary of real numbers, **QDict<*char**>** will be a dictionary of strings (strings which are referred to as pointers to a character's array).

computation methods such as electrostatics, which uses them for tri-linear interpolation of scoring values.

Other ancillary classes are dictionaries of various kinds to store default values for van der Waals radii or charges, points (vectors) to perform linear algebra operations such as distances between atoms, angles between bonds, and so on.

5.3 Scoring Function Implementation

Every interaction is computed by one or more AtomList methods, depending on its algorithm and the constraints applied to it. This section explains in detail what most of the scoring routines do. As there are many similar routines, they are grouped by class and common elements are explained together. The main classes are hydrogen bond detection, surface calculation, electrostatic/GBSA/solvation, van der Waals forces. Van der Waals forces were used for both protein/ligand interactions, taking over the role of a missing 'clash' or 'contact' term, and ligand/ligand interactions, to act as substitute for a proper bond rotation term.

The scoring routines, as said previously, are implemented as AtomLists' methods. They call the corresponding Atom methods in cases of pairwise interaction: *e.g.*, **AtomList::HBond(*AtomList*)** calls **Atom::HBond(*AtomList*)** which in turns delegates to **Atom::HBond(*Atom*)** the actual computation, essentially a distance and angle check. In the case of properties which can not be broken down into single-atom contributions, the AtomList method takes care of all computations. In those cases, when required, atoms' variables are updated to reflect the new environment. For example, after computation of surfaces, the exposed surface area of every atom is updated. In the scoring function code, very often the interaction methods are called via molecules' methods, which simply act as a pass-through of variables and structure references to the corresponding AtomLists' methods.

5.3.1 Hydrogen Bonds

5.3.1.1 Simplified Binary Model - GoodHBond

This hydrogen bond model is very simple, based on geometrical constraints, namely angle and distance checks. The atom under consideration is passed a pointer to another

atom, the (putative) acceptor. If the first atom is an H, it retrieves the candidate donor (the only atom to which it is bonded), and after checking for correspondence of allowed donor and acceptor atom-types it computes some distances and angles between itself and the other two atoms. It then returns 1 if the donor-hydrogen-acceptor angle is between 160° and 180° and the distance hydrogen-acceptor, once their VdW radii are subtracted, is 1.5 Å or less; -1 can be returned in case a donor-donor or acceptor-acceptor interaction is detected, as explained in Section 5.3.2.1.

5.3.1.2 Simplified Gold Model

This hydrogen bond model takes a simplified version of the Gold hydrogen bond model as detailed in literature [40, 43]. The method will return a real number depending on the deviation from ideal values of the angle between acceptor, donor, and hydrogen atoms, and the overlap between the hydrogen position and ideal interaction points around the acceptor obtained with reference to its lone pairs ($\overrightarrow{L_{P1}A}, \overrightarrow{L_{P2}A}$ are the vectors joining them to the acceptor). The mathematics upon which the method is based is detailed in the following set of equations:

$$
\begin{aligned}
&HBondNRG = w(r) \bullet w(\vartheta) \bullet w(\varphi) \\
&w(r) = \begin{cases} 1 & if \ \ r < .25 \\ (r - 0.25)/1.25 & if \ \ .25 < r < 1.25 \\ 0 & if \ \ r > 1.25 \end{cases} \\
&w(\vartheta) = \begin{cases} 1 & if \ \ \vartheta < 20^\circ \\ (\vartheta - 20)/40 & if \ \ 20^\circ < \vartheta < 60^\circ \\ 0 & if \ \ \vartheta > 60^\circ \end{cases} \\
&w(\varphi) = \begin{cases} 1 & if \ \ \varphi > 160^\circ \\ (\varphi - 60)/100 & if \ \ 60^\circ < \varphi < 160^\circ \\ 0 & if \ \ \varphi < 60^\circ \end{cases}
\end{aligned}
\tag{5.1}
$$

In the equation:

- r is the distance between acceptor and the Virtual point where it would maximally interact with the Donor;
- θ is the angle between the DH vector and the lone pairs' plane ($\overrightarrow{L_{P1}A \times L_{P2}A}$), only defined when two lone pairs are present (Oxygen and Sulphur);

- φ is the angle between the vectors DH and $L_{PX}A$[JW23] ($\overrightarrow{DH - L_{PX}A}$; $X = 1,2$).

In practical terms, the position of the donor is used to create a 'virtual' interaction point 5.0 Å [JW24]along the direction of the donor-hydrogen axis, starting from the donor position. The tentative hydrogen bond is then given a distance weight which is equal to 1 for distances between the virtual and the acceptor of less than 0.25Å, and which linearly decreases up to 1.5 Å.

Then, depending on the number of lone pairs on the acceptor, the angular multiplicative factor is computed. In the case of two lone pairs, the contribution is largest when the donor-hydrogen vector lies within 20 degrees of the plane defined by the two acceptors/lone pair vectors – as measured by the angle called θ. Otherwise, it decreases linearly until it reaches zero for 60 degrees out of the plane. The final contribution is assigned based on the angle φ between the acceptor-lone pair vectors, and the donor-hydrogen vectors. When the two vectors are almost collinear (φ between 180 and 160 degrees), this gives the full contribution. Otherwise, the contribution decreases linearly to zero when φ equals 60 degrees. As there are two possible φ-contributions, only the larger (i.e. that for the more aligned vectors) is considered. The final H-bond contribution is then computed by multiplying the three terms together. If instead, the acceptor atom can only share one lone pair, the θ contribution is neglected and only the cylindrically symmetrical contribution φ is accounted for. As only the geometrical terms are taken from the GOLD implementation, but not the proper parameters, every H-Bond will have equal weight. This way, the method will serve as an ideal middle ground between the binary model and the proper GOLD Hydrogen Bond model as recovered from GOLD's output files[JW25].

5.3.2 Hydrogen Bonds: Penalties

As mentioned in section 2.2, something which has been missing from the scoring function field is a term to take into account unsatisfied hydrogen bonds. The general assumption is that in polar active sites, not forming a hydrogen bond where one would be formed by the isolated protein and ligand in solution, is a sufficiently strong penalty to discourage the molecule from binding. But this may not be enough. What happens when not only donor and acceptor are unsatisfied but they are also clashing with each

other? Does this ever happen? To investigate this, we implemented three penalty methods.

5.3.2.1 Simplified Binary Model - BadHBond

Using the same algorithm detailed in section 5.3.1.1 an attempt was made at estimating the number of *anti*-hydrogen bonds, defined as when the two atoms selected for the interactions where both acceptors, or both donors, and especially when their geometry would not allow the formation of a proper hydrogen bond between them - caution is necessary though, since some donors can also act as acceptors. If the first atom is not a hydrogen, but an acceptor or donor, and the second atom belongs to the same class, a penalty of -1 may be returned, rather than the standard +1.

This is meant to highlight an acceptor/acceptor or donor/donor clash: these clashes will be of course subject to different geometrical constraints: the acceptor-acceptor clash is assigned after a simple distance check (1.5 Å), as the lone pair clouds are assumed to be quite diffuse around the atoms, especially in the case of oxygen. Conversely, the attribution of a donor-donor penalty requires not only that the distance between the two, once their VdW radii are subtracted, is 1.5 Å [JW26]or less, but also that the angle defined by the two donor-hydrogen bonds ($\overline{D_1H_1 - H_2D_2}$) is between 160° and 200°, so that the two hydrogen atoms clash with each other.

5.3.2.2 Unsatisfied Donor and Acceptor count

It is a reasonable assumption that certain ligands, although they fit snugly in the pocket, will not bind because they simply do not have acceptors or donors in the right position to interact with those around them. To account for this, two counts were implemented, one over all hydrogen bond donors, and the other over the acceptors. Every time the 'GoodHBond' algorithm detects a hydrogen bond in which an atom is involved, it updates the internal counter of the atom by one, until it reaches a maximum value, determined by its atom-type. During the unsatisfied count, the difference between the maximum and the actual value, performed on all possible acceptors (donors) is evaluated. These values are then returned, and used for scoring. As implemented in the scoring function, the count is not performed on the whole protein, but on a selection of protein atoms up to 8 Å around any ligand atom, to save on computer time. Detection of

hydrogen bonds is however performed on the whole protein, to avoid interface artefacts. It must be said, however, that since the selection process yields different numbers of atoms on which the unsatisfied hydrogen bond count is performed, the values of this scoring term are not perfectly comparable between systems. Nevertheless, these terms have been used, the rationale being that their actual usefulness would be ascertained by the Genetic Algorithm during the parameterisation process.

5.3.3 Surface calculations

As has been shown in previous studies (see Creighton for a more extensive treatment[208]), the hydrophobic buried surface of a molecule is directly related to a phenomenon called 'hydrophobic effect', which expresses the tendency of a molecules to escape water phases and bury itself within a hydrophobic medium. From literature [209, 210], the hydrophobic effect [JW27]is usually correlated with measures of the molecules' hydrophobic surfaces buried upon binding. It was therefore decided to include some surface terms within our scoring function, as others have done before [52, 103].

Surface calculations are performed with Richmond's analytical method [211, 212]. The routine used was taken from Tinker, by a member of our group, Julien Michel, who ported it from FORTRAN to C, and then adapted by myself as an AtomList method to work within my own framework.

The method was modified to separately return the hydrophilic, hydrophobic and apolar portion of the surfaces, as well as for the use of different AtomLists at once. The first contains atoms that form a molecule (*e.g.* the ligand), the second atoms from another (*e.g.* the protein). For example, the method can be invoked in many different ways, each returning a different result:

$$\text{Ligand} \rightarrow \text{Surface}() \tag{5.2}$$

The command above will, by default, compute the exposed surface of the ligand with a probe of radius 1.4 Å (the same as water's) – that is the Solvent Accessible Surface Area (SASA) of the free ligand in the given conformation.

$$\text{Ligand} \rightarrow \text{Surface}(\text{Protein, double R}) \tag{5.3}$$

This will, instead, compute the surface of the ligand when adsorbed onto the protein's surface, with a probe of radius R. To obtain the buried hydrophobic surface area, we will have to subtract the two terms just shown, taking care to use the same radius for both computations.

$$\text{BuriedSurfaceOfTheLigand} = \text{Ligand} \rightarrow \text{Surface}() - \text{Ligand} \rightarrow \text{Surface}(\text{Protein}) \quad (5.4)$$

That is, the buried surface of the ligand is the total surface exposed by the ligand when free (e.g. in water), minus the surface which is still solvent-exposed when adsorbed in the protein.

The *Surface* methods actually return both hydrophilic and hydrophobic surface areas in one go, saving computational time. Also, as a general rule, every time a surface computation is performed, the atoms involved have their internal surface area value updated, so that a simple sum over a few can return a partial surface value without the need to re-compute it if conditions have not changed.

During the scoring, six different surface terms were considered, divided into two equal subsets of three, for hydrophilic and hydrophobic surfaces:

1) ***Ligand Hydrophobic(Hydrophilic) Surface Area*** **(LigHPhobS)**: This is a very simple surface area measure of the molecules in water, that should correlate directly with the size of the molecules. It was not originally thought to be in the set of terms, but came in as a programming mistake, and since it did perform quite well to optimise ranking efficiency, was kept in;
2) ***Ligand Buried Hydrophobic(Hydrophilic) Surface Area*** **(LigHPhobB)**: This term is obtained as follows: after the ligand is 'docked'[15], given the ligand SASA[JW28] when free in water, and its SASA when in the protein, the difference between the two is the buried SASA upon binding;
3) ***Total (Ligand + Protein) Hydrophobic(Hydrophilic) Surface Area*** **(TotHPhobS)**: this final term is the sum of the buried SASA upon binding for both ligand and

[15] NB: Although we may expect the ligand conformation when bound to be different from that when free in water, no attempts here are made to account for this. We are well aware of the approximation, but due to the application of this scoring function as a quick filter during the first phases of SVS we cannot afford proper conformational sampling to address this.

protein – it is computed by summing the amount of ligand and protein hydrophobic surface buried upon binding [16].

As already said, the three equivalent terms for hydrophilic surfaces were also computed and assigned a class of their own. To speed up all computations involving protein atoms, only atoms closer than 8 Å to any atom of the ligand were used rather than the whole protein. This did not affect the final results, as the algorithm was actually taking differences of surfaces, and 8 Å should be sufficient to include every possible surface interaction, even mediated by one or two water molecules. A probe radius of 1.4 Å was used in all cases.

5.3.4 Electrostatics and Solvation

Approximate electrostatics interactions are implemented in a few different ways within the program: the easiest to compute is the energy associated with the coulombic interaction between two atoms belonging one to the protein (a) and the other (b) to the ligand:

$$Coul = \frac{1}{4\pi\varepsilon_0\varepsilon_r}\frac{Q_a Q_b}{R} \tag{5.5}$$

This is the simple coulombic pairwise interaction, where Q_a and Q_b are charges associated with atom a (on the protein) and b (on the ligand) – ε_0 is the relative permittivity of free space; ε_r is a corrective factor called the dielectric constant, larger than one and depending on the environment where the interaction occurs. The $1/4\pi$ factor comes from the spherical symmetry of the electric field. This interaction is computed in the code by the **ES** method:

Protein->ES(molecule* ligand)

As usual, the method is associated with one molecule's structure and requires the specification of another molecule for a correct execution. The default when the input molecule is not given is not to compute the self interaction, but to return a zero value. The difficulty is that the dielectric constant ε_r varies from point to point in space, and it

[16] NB: again, no provision is made here for the ***induced fit*** phenomenon of protein side chains.

can be different inside a protein as compared to the value of 80 in water. More importantly, charges on atoms are not point charges residing on the nucleus. They are, instead, diffused through the electrons' cloud, and they react to other charges nearby by modifying their own distribution. What is more, the ε_r value depends.on the ability of this charge distribution to respond to external influences. So, how can we know the equilibrium distribution of a set of charges interacting with each other, which is what we need for an accurate evaluation of the ε_r within the protein's binding site, and of the electrostatic interaction soon after? This is commonly known as the fundamental problem of electrostatics, and is properly treated by resolution of the Poisson Boltzmann [JW29]system of differential equation. However, even an approximate numerical solution of this is computationally excessive for the sake of virtual screening. Polarisable force fields are being developed for use in molecular dynamics, but they still are too expensive for virtual screening applications.

Therefore, the basic coulombic force-field expressed above is often tweaked to return different, hopefully more accurate, results. For example, a 4R distance-dependent dielectric constant can be used, as this is said to reproduce more accurately the ε_r variation within a protein environment. Since this 4R term increases with the distance, it equates to a diminished importance of remote interactions in condensed matter as opposed to in vacuum. That is, the potential essentially falls off as $1/R^2$ rather than as $1/R$, and this simulates the screening due to the re-orientation of charges which Debye-Huckel theory describes.

All the computations just detailed are in fact performed twice but the second time, rather than using the whole protein, a cut-off of 8 Å from every atom's ligand is imposed: no protein atom farther than that contributes.

Charges are taken from the atom's structure files, which usually means user assigned charges for ligands and other generic molecules, and Amber94 charges for proteins, as assigned by QTMPP when generating the protein structure by referring to a look-up table loaded at start-up that can be modified (prior the execution) to suit the user needs. Ligand charges were in this case obtained with the AM1-BCC method [29, 30, 213], using the ***antechamber*** software, part of the ***Amber 8*** suite.

Electrostatic interactions do not only consist, however, of charge-charge interactions: because the distribution of charges around an atom or molecule can be deformed, dipoles and higher level multipoles can also play a role. However, their contributions are generally of a lesser order compared to charges – also, they require much more computation time, and were therefore not implemented.

As previously mentioned, as an attempt to minimise computation time the Coulomb interaction was also computed by limiting the interacting atomic pairs to a distance of 8 Å[JW30]. This is, in principle, incorrect, since Coulomb interactions are notoriously very long ranged: the hope was, however, that if for some reason the term were to perform well during the Genetic Algorithm parameterisation, it would have substituted the full-blown computation with little or no difference in the ranking power, and a much quicker execution time.

(De-)Solvation contributions to binding are estimated using the Generalised Born Surface Area (GBSA [206]) term, whose basic equation is given as:

$$G_{Sol} = G_{cav} + G_{vdW} + G_{Pol} \tag{5.6}$$

Where the solvent-cavity term G_{Cav} and the solute-solvent interaction term G_{VdW} are expressed in terms of a surface contribution over all atoms: here SA_k is the total solvent-accessible surface area of the *k*-th atom, and σ_k is an empirical atomic solvation parameter.

$$\begin{aligned} G_{cav} + G_{vdW} &= \sum_k \sigma_k SA_k \\ G_{Pol} &= -166 \cdot \left(1 - \frac{1}{\varepsilon}\right) \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{q_i q_j}{f_{GB}} \end{aligned} \tag{5.7}$$

As for G_{Pol}, it is expressed as a modified Coulomb-like equation known as the Born equation, where q_i are the charges of the interacting atoms and f_{GB} is a function of the atoms' Born radii α_i and their distance r_{ij}. This last term, representing the polar contribution to the solvation energy, is computed with a pairwise approximation adapted from work by Julien Michel[214, 215]. The reader can refer to his publications for further details on the implementation.

The function, as implemented in the code, computes the polar contribution directly by means of other AtomList methods, namely ***AtomList→elecsolv( )***, and uses the surface method results to estimate the non-polar contribution. Actually, methods exist for the separate computation of both polar and non-polar parts, and they are combined together to form the final result.

$$\text{AtomList :: GBSA}\begin{pmatrix}\text{point * ActiveSiteSurface}, & \text{point * LigandSurface},\\ \text{AtomList * Environment}, & \text{AtomList * WholeProt}\end{pmatrix}$$

The GBSA function shown above takes, as input, pointers to the surfaces of the active site (ActiveSiteSurface) and ligand (LigandSurface), and computes them if necessary. It also takes in the protein's AtomList (WholeProt), only used when the active site is a subset of atoms around the ligand to avoid interface errors. That is because Born radii depend on all atoms around a given atom, and not only the one selected to compute the GBSA. As for the 'Environment' list, it usually consists of the sum of active site and ligand atoms, and it is used to compute the polar contribution.

In addition to this, a simpler attempt to take into account desolvation was provided by a Coulomb interaction with distance-dependent dielectric constant, using a 4R reduced formula.

$$Coul = \frac{1}{4\pi\varepsilon_0 4R}\frac{Q_a Q_b}{R} \tag{5.8}$$

This term was computed in the case that the whole protein was used, or only an 8 Å radius cut around the ligand.

The Coulomb and GBSA terms are, in fact, strongly anti-correlated, and given that the actual solvation free energy is given by the sum of the two terms, the final step has been to combine the GBSA term with the plain coulombic in both the full host and 8 Å-limited selection, to create two new solvation terms.

So, in conclusion, there were 2 purely electrostatic terms:

1) **Coulombic;**
2) **Coul 8 Å;**

And 5 'solvation' terms, which are better defined as electrostatic with solvation treatment terms:

1) **Coul 4R;**
2) **Coul 8 Å 4R;**
3) **GBSA;**
4) **GBSA+Coul;**
5) **GBSA+Coul 8 Å;**

5.3.5 Van der Waals

Dispersion and London forces are [208] taken into account by a van der Waals potential method, with user-adjustable powers to allow for softer or harder penalisation of clashes between atoms. The basic equation is:

$$E_{vdW} = 4\varepsilon_{ij}\left(\left(\frac{\sigma_{ij}}{r_{ij}}\right)^{rep} - \left(\frac{\sigma_{ij}}{r_{ij}}\right)^{att}\right); \quad \sigma_{ij} = \sigma_i + \sigma_j \; ; \quad \varepsilon_{ij} = \sqrt{\varepsilon_i \varepsilon_j} \; ; \qquad (5.9)$$

Where ε_{ij} is the well depth, σ_{ij} is the equilibrium separation distance, *rep* and *att* the repulsive and attractive powers. Most common choices for the repulsive/attractive exponents are 6/12, 6/9 and 4/8, and all are available hardcoded within the software: also, a method which computes them all at once, greatly saving on distance computations, and a more generic method with user-defined attractive and repulsive exponent powers were implemented. Parameters for the potential function are taken from a standard van der Waals radii table from Amber 8, and future users are able to specify their own or use the hardcoded values.

5.3.5.1 Clashes

The protein/ligand van der Waals terms (4/8, 6/9, 6/12) do not only play the role of a generic clash term, with the severity depending on the repulsive indices. The presence of an attractive exponent makes the term more oriented toward a generic shape-complementarity term.

5.3.5.2 Internal Torsion

A proper internal torsional potential implementation is lacking in the scoring function, for the simple reason that there was not time to implement one. An internal ligand van der Waals term was therefore used as a substitut[JW31]e, on the intuitive basis that

repulsion between atoms of the same molecule would be minimised at the equilibrium conformation – differently from a torsional potential, the interactions were computed not only between neighbouring atoms, but did include all other atoms in the molecule, regardless of how many bond were separating them. As in the case of the clashes (5.3.5.1), three possible terms were computed, with attractive/repulsive indices of 4/8, 6/9 and 6/12: the future user can choose their own exponents and modify the parameters, if so inclined. These internal torsion terms were not computed for the protein, since although Gold allows a limited degree of flexibility for the active site side chains, we only considered the ligand structure as output, with interaction between that and the reference protein structure.

5.4 Scoring Function Use

QTMPP[17] is the name of the program which actually runs the scoring function. It is launched from a terminal, and controlled via an init(ialisation) file passed on the command-line. So, the user types in his UNIX™ shell:

> qtmpp init_file

and the program is executed. The init file contains non-default values for variables used by the program, such as the name of the protein file, a file name containing the list of ligand files (and therefore their location), the name of the output file, and whether or not the initialisation should use a GOLD[40] input file which may or may not be present in the directory to set up variables such as the active site position in the protein and the protein's filename. All the scoring routines output to a common file, whose name is specified within the init file. Later on, we will describe in some detail the said output file, since it is also the input file for our parameterising GA.

As already mentioned, the program reads a single protein and many ligands, processed one after the other. The initialisation of the protein, *i.e.* the creation of the data structure, is the first step, followed by the computation of its solvent-accessible surface, free of any ligand. Protein charges and radii are assigned via a look-up table accordingly to Amber94-ForceField parameters[216]. Ligand charges are instead read from the mol2 file when specified – that is, if the corresponding record says anything other than "No

[17] QT-Molecules++, since it uses QT to describe molecules and it is written in C++.

Charges". In this particular study, AM1-BCC ligand charges were assigned using Antechamber, a preparation program from the Amber 8 suite.

After initialisation of the protein has been finalised, the ligand database file, containing the filenames of each ligand's mol2 file, is read: the ligands inside it are counted and their number written to the output file.

lig_num 1 (1)

The line above means that we have 10 000 ligands: additionally, we can add the number of ligands whose ranks we want to optimise. This is not for QTMPP use, but rather for the next step, the parameterisation of the function with a GA. That is, writing "**lig_num 1 5**" command to the program to optimise the ranking positions of the first five ligand/lines. If no value is given, and that is the default behaviour in QTMPP, the GA program will default to 1. This is sensible because, as a general rule, we only have one active ligand per protein, and we take care in putting it first in the ligand list.
Then, every ligand is read from file, initialised, charges are assigned, the scoring terms evaluated and the results written to the output file, in a line which looks like:

protein_name.mol2 ligand_name.mol2 score1 score2 score3 ... scoreN

Finally, QTMPP wraps things up and exits gracefully. The final result is an output file composed of as many lines as the number of ligands.

This output, meant for use in the parameterisation of the scoring function for the proteins contained within the set just analysed, can be fed almost immediately into the GA program. Care must be taken to set the 'true' ligand(s) in the first line(s) of the scored list, as this is(are) the one(s) whose ranking the GA will attempt to optimise. For a single protein file, the only thing needed is to add a line on the top which specifies the number of proteins contained in the file (1 in this case): **prot_num 1**
If the user wants to create a multi-protein parameterisation file, he/she only has to pass the individual file to a purpose-built PERL script to stitch them together in the correct order. The final result will look something like Table 5-2[JW32].

prot_num	**3**	***(total number of proteins in the file)***			
lig_num 3	**1**	***(number of ligands for the first protein – number of ligands to rank up)***			
protein_1.mol2 ligand_1.mol2	score1-1/1	score2-1/1	score3-1/1	...	scoreN-1/1
protein_1.mol2 ligand_2.mol2	score1-1/2	score2-1/2	score3-1/2	...	scoreN-1/2
protein_1.mol2 ligand_3.mol2	score1-1/3	score2-1/3	score3-1/3	...	scoreN-1/3
lig_num	**5**	***(number of ligands for the second protein)***			
protein_2.mol2 ligand_1.mol2	score1-2/1	score2-2/1	score3-2/1	...	scoreN-2/1
protein_2.mol2 ligand_2.mol2	score1-2/2	score2-2/2	score3-2/2	...	scoreN-2/2
protein_2.mol2 ligand_3.mol2	score1-2/3	score2-2/3	score3-2/3	...	scoreN-2/3
protein_2.mol2 ligand_4.mol2	score1-2/4	score2-2/4	score3-2/4	...	scoreN-2/4
protein_2.mol2 ligand_5.mol2	score1-2/5	score2-2/5	score3-2/5	...	scoreN-2/5
lig_num 4	***(number of ligands for the third protein)***				
protein_3.mol2 ligand_1.mol2	score1-3/1	score2-3/1	score3-3/1	...	scoreN-3/1
protein_3.mol2 ligand_2.mol2	score1-3/2	score2-3/2	score3-3/2	...	scoreN-3/2
protein_3.mol2 ligand_3.mol2	score1-3/3	score2-3/3	score3-3/3	...	scoreN-3/3
protein_3.mol2 ligand_4.mol2	score1-3/4	score2-3/4	score3-3/4	...	scoreN-3/4

Table 5-2: a general example of input file for the scoring function parameterisation.

Here, ***score1-3/2*** means the interaction (scoring) term #1, computed for a pose of ligand #2 interacting with protein #3. When properly filled up by numbers, the final file looks something like Table 5-3.

prot_num	**3**				
lig_num 3	**1**				
protein_1.mol2 ligand_1.mol2	**3**	**25**	**6.78**	**...**	**8.97**
protein_1.mol2 ligand_2.mol2	0	12	-3.56	...	12.5
protein_1.mol2 ligand_3.mol2	1	6	3.14	...	5.62
lig_num 5	**2**				
protein_2.mol2 ligand_1.mol2	**3**	**21**	**6.78**	**...**	**0.97**
protein_2.mol2 ligand_2.mol2	**0**	**34**	**3.96**	**...**	**2.35**
protein_2.mol2 ligand_3.mol2	7	6	3.34	...	-5.2
protein_2.mol2 ligand_4.mol2	3	15	3.98	...	-8.9
protein_2.mol2 ligand_5.mol2	2	19	1.78	...	8.72
lig_num 4	**1**				
protein_3.mol2 ligand_1.mol2	**0**	**17**	**3.56**	**...**	**1.15**
protein_3.mol2 ligand_2.mol2	1	7	3.14	...	5.26
protein_3.mol2 ligand_3.mol2	3	21	3.56	...	12.9
protein_3.mol2 ligand_4.mol2	1	6	3.41	...	-5.6

Table 5-3: a more specific example of input file for the scoring parameterisation.

Other PERL scripts are provided to build histograms of the scores' distributions, select and discard outliers, scale the values and validate GA input files before handing them to the next program. The GA can then be run.

5.5 Parameterisation program

Empirical scoring functions, as explained in the previous chapters, usually consist of a series of terms which are added to give a 'score' - a numerical value which hopefully correlates to the free energy of binding. The terms are summed not with equal weights, but with each weighted by a coefficient, usually obtained through a regression approach to known binding affinities. A number of regression/fitting techniques have been applied in literature, ranging from linear least squares regression, which attempts to minimise the quadratic error between two series of experimental and computed data, to higher order regressions, neural networks, which offer an implicit non-linear weighting, and finally Genetic Algorithms[49]: one example including all of them is the work of Giordanetto et al.[68].

Genetic Algorithms (**GA**(s) herein) [190] were chosen for this work, due to the particular nature of the problem at hand. We had to code a regression program which would simultaneously optimize the ranking of N different ligands each in its own ranked list. The solution space for this problem is rather sparse and not amenable to analytic approaches, which require the knowledge of first and second order derivatives. Therefore, a method which allows some sort of guided sampling of the solution space is needed. GAs are well known for their ability to sample efficiently a large space without the need for exhaustive enumeration of all states. They have been shown to efficiently locate sub-optimal solution quickly and in a robust way, as long as that the fitness function is not *pathological*[190]. Like neural networks (NN), GAs are stochastic in nature, meaning that the result is always different. However, being a deterministic algorithm, we can repeat a parameterisation using the same random-number generator seed and get exactly the same result. An advantage of GAs is that they return a simple genome which encodes the weights to be applied to our scoring terms, or the operation between them if we are also changing the functional form (Genetic Function Approximation, or GFA)[217]. Whereas a NN interpretation is not straightforward, especially when we want to express our scoring function in words understandable by a

chemist, rather than in mathematical terms, GAs are much more amenable to interpretation. For example, a GA parameterised scoring function will look something like:

$$\Delta G = 0.2 \bullet HBond(Lig, Prot) + 0.8 \bullet HPhobSurf(Lig, Prot) \qquad (5.10)$$

This equation can be easily interpreted in terms of the relevant intermolecular forces. An NN scoring function would instead be much less easy to interpret, containing contributions from more than one term mixed in (highly) non-linear ways; the situation gets quickly worse with the increase of 'hidden' layers of neurons.

It must be said that we could have similarly complex results if instead of using a GA, we were to use GFA to create our function. This too may bring up terms with unclear physical explanation, such as products or divisions of a hydrogen bond count and a buried surface term. How could we make sense of them, and be sure they were not artefacts of excessive fitting over the training set? Furthermore, in recent literature, it was shown [68] that allowing for non linear relations between scoring terms did improve the efficacy of them only marginally, so we decided to go for the easiest method: GAs.

In practical terms, we rely only on massive amounts of data generated by docking to be used by the GA to find a (sub-optimal) solution to our (general) problem. The data on which the GA works are lists of scores, for 1001 ligands, each docked (ten times) to more than one hundred proteins. While 1000 of the ligands are drug-like molecules[18], the 1001st (the topmost in the file, in fact) is the score reported for the known ligand in the observed crystallographic structure (therein COLO, Crystallographically Observed Ligand Orientation) – in alternative parameterisation, re-docked poses of the known ligand were used, namely the closest to the crystal structure, and the one scored higher by GOLD. For every protein, then, we have a total of up to 10,001 score lines, each corresponding to a ligand orientation (pose) within that protein. The format is shown in section 5.8.

The GA's task is to try and recognise these 'original' binding modes, by modifying the parameters for a sum of the scoring terms in such a way that these 'true' poses would show up in the first few places of the ranked list. At the same time, the GA must learn

[18] The Rognan set discussed in section [Dataset].

not to be fooled by the other poses, which represent 'likely' binding modes for ligands which, however, are presumed not to be active. Learning this will hopefully allow the scoring function to minimise the number of false positives in real use. Of course, the more proteins we generate docking data for, the more of these ranked lists of COLO/Decoys there will be. This will make it harder for the GA to select a balanced combination of terms able to rank all COLOs first: hopefully, though, the resulting scoring function will be more general. The coming section is going to focus first on some Genetic Algorithm fundamentals, then on how the GA code was built, and how it performed.

5.6 Introduction to Genetic Algorithms

A Genetic Algorithm (GA) is a computer program that mimics the process of evolution by manipulating a collection of data structures called chromosomes. In our case, each of these structures encodes a possible solution (i.e. a possible ligand orientation within the protein binding site) to the docking problem and may be assigned a fitness score based on the relative merit of that solution. In a GA, a population of solutions to the problem is maintained which, over a number of generations, undergoes structured and yet randomised information exchange in order to generate new, fitter, solutions to the problem.

5.6.1 Terminology

The terminology of GAs borrows from the field of natural genetics. Each solution in the population can be considered the analogue to a chromosome in biological systems, which in turn is composed of a number of genes representing the parameters of the optimisation problem being solved (the search for the optimum ligand/receptor conformation, in our case). Each gene may take on a number of different values called *alleles*. The values for the alleles of each gene determine the chromosome fitness, calculated from a given fitness function. This fitness value is a more or less direct measure of reproductive success for said cromosome, and so directly proportional to his population share in the next generation[JW33]. Suppose we start with two different chromosomes (single-gene, for simplicity), which share equally the population, so it is 50/50 composed. The fitness value for the gene A is 1, the fitness value for the gene B is 2. Then, the chromosome containing A will reproduce themselves with half the probability than the B. Then the next generation will be composed of 2/3 B, and only

1/3 A, and they will be in proportion 2/1 in favour of B. This is the so called *Roulette-wheel*-based selection strategy, which simply corresponds to assigning to each chromosome a number of slots proportional to its fitness score from a hypothetical roulette wheel. Then you throw inside the spinning wheel as many marbles as offspring you want. You will end with a resulting population with frequence of individuals proportional to their fitness values (allowing for random variations due to the finite sample size, also known as 'luck').

5.7 Crossover and mutation

As has been described, the process loses some chromosomes (the least fit) and maximizes the percentage of the fittest in the gene pool. It does nothing to explore new regions of search space, which is to say search for different, fitter solutions. This ability is given by the ***crossover*** and the ***mutation*** operators. The first involves selecting two chromosomes at random from the population, and then choosing a random position (or more than one) at which the alleles are exchanged. This result in two new chromosomes (called ***children***). e.g., we can take two 10-genes chromosomes A (1110101000) and B (0010010111) and then select the point of exchange (let us say, from position 4 onward).

$$\begin{aligned}(C,D) &= CrossOver_{Pos=4}(A,B) = \\ &= (1110.101000\times 0010.010111) = \\ &= (1110.010111,0010.101000)\end{aligned} \tag{5.11}$$

The two chromosomes will be cut at the position 4 (the dots in the equation are the cut and paste points) and then recombined, obtaining the two children chromosomes C (1110010111) and D (0010101000). By performing this crossover we are attempting to generate new solutions to the problem. The aim is to merge attributes important for the fitness of the parent chromosomes into the offspring, much as in natural genetics. If our fitness function (F(x)) is simply the sum of all the digit in the chromosome, then the value for the various chromosomes will be: F(A) =5 ; F(B)=5 ; F(C)=7 ; F(D) =3. As can be seen, the fitness function for one of the offspring is higher than both the parents, while the second child inherited a lower fitness value. Whereas the A and B would have converged to a ratio 1/1 in the gene pool, the C and D would converge to a 7/3 relative proportion and, if we take as a measure of 'overall fitness' the sum of all fitness values of all the chromosomes times their frequency, the gene's pool will be fitter overall.

Mutation is the second principle operator used in GAs. With this operator a random chromosome is selected and a random allele within it changed. The example case deals with binary strings so the mutation operator involves inverting the bit value of an allele from either 0 to 1 or from 1 to 0. In this case, then:

$$E = Mutation_{Pos=5}(A) = MX5(1110101000) = (1110001000) \quad (5.12)$$

Using the same fitness function as before, the mutation in position 5 give an outcome with a smaller value of fitness function, since a '1' becomes a zero, therefore diminishing the fitness score. But a mutation on a position occupied by a '0' would instead increase the fitness value. In both cases, we are performing a local search of our solution space, generating chromosomes that could not be obtained simply by cross-over (which is a form of linear combination) of A with B (in fact, a double cross over at position 4, then 6 could have produced E from A and B, but it is possible to envisage cases where mutation is the only way to get a solution – even so, with mutation the transformation A→E is accomplished in one step, rather than two).

During the course of a GA, crossover may result in the loss of potentially useful information such as values of 0 or 1 at a particular position. The GA search can therefore become trapped in a particular region of search space from which it is unable to recover. Mutation attempts to fix this problem by inverting bits at random, and therefore introducing (or re-introducing) possibly important information. However, in the majority of cases, mutation results in a chromosome with reduced fitness (that is the rule in Nature too) and the mutation operator is therefore applied with a low probability (typically the order of 1 in a thousand). In our example, a mutation at position 4 would have the effect of reintroducing a '1' value at that bit, otherwise not recoverable in any way by simply crossing the existing chromosomes, because both of them have it set to 0, and no possible crossover operation would never be able to reintroduce it (if you have just 0s, you can crossover as many time as you want, you will only get 0s, and never 1). In a typical GA, chromosomes are selected for crossover and mutation using a roulette wheel based selection strategy. The next generation may therefore be composed entirely of the offspring of the previous generation. As a result, parent chromosomes that are

actually fitter solutions to the problem may be lost. A typical strategy employed, known as the 'elitist' model, prevents this from happening by ensuring that the best scoring chromosomes from the previous generation are always retained in the following one. This strategy is better than the alternative of substituting only a part of the generation (say 50%) while retaining the rest, due to the fact that the elimination of poor chromosomes is faster, and so is the convergence of population attained once the ratio between chromosomes has reached a constant value (and so, consequently, has the overall fitness). Usually, for GAs, if the fitness value is somewhat dependent on the presence of other chromosomes, the fittest solution may be an equilibrium between two or more *alleles* of the same gene. This is quite common in nature, where all the chromosomes interact each other, but unlikely in the typical application of GA to other fields, such as the docking problem. This is because the solutions do not interact with each other, and so the fitness function does not include cross-terms related to the rest of population or the percentage of the chromosome inside it. Then, the strict convergence criterion is usually satisfied when only the fittest chromosome survives in the pool, with a share of 100 % of the population.

5.8 Coding of the GA

The aim of our Genetic Algorithm is to maximise the ranking position of truly observed ligand orientation for the experimental ligand, competing with a list of decoy molecules docked on the same target protein. To attain this, the weighting coefficients of each term of the scoring function are varied until the correct balance is found. The operation is performed on multiple proteins at once, to obtain generally applicable parameters.

5.8.1 GA Implementation Details

QTGA is the GA program written and used during this work: it is based upon the M.I.T. GALib, coded and maintained by Matthew Wall[218], for the Genetic Algorithm implementation, and the **QT** libraries[19] for the part relating to file-handling and data access/storage[219]. QTGA is a console program that reads its input from a formatted

---

[19] This set of libraries is the basis for the well known K Desktop Environment, free for academic and personal use, and can be obtained at http://www.trolltech.edu

file[20] containing protein and ligand names associated with respective scoring values for up to 28 pre-defined terms. Raising this maximum genome-length will require intervention in the source code and recompilation. This limit, albeit not strictly necessary, made memory management easier within the program. There is however some space left for changes. The default terms' names are hard-coded and correspond to the original 28 terms I used (24 I implemented, plus GOLDSCORE and its 3 terms, obtained from the GOLD[40] output files); the names of the terms can be changed by pointing QTGA to a correctly formatted file. The number of proteins, of genomes, generations, and behaviour in scaling the mutation and crossover probability along the generations can all be modified from default with specific flags in the command line. As usual in console programs, running it with the '-h' flag will bring up a complete list of the accepted command-line options and an example line.

5.8.2 Getting the right sign

The Genetic Algorithm inside QTGA is defined as a population of individuals, each constituted by up to 28 genes, which can take any real value between -100 and 100. An example of a 'genome' follows:

Gene 1	Gene 2	Gene 3	Gene 4	Gene 5	Gene 6	Gene 7
-1.25	**3.46**	**34.21**	**-96.72**	**0.03**	**3.14**	**2.78**

In our model, a positive sign will cause the score for the pose to rise; we want non-binders or weak binders to have a higher score than tight binders, in parallel with their values of binding free energy. The observed ligand has a negative value for the free energy of binding, whereas the decoys structures will presumably be more positive. At the same time, many of our scoring terms output positive numbers, which will be multiplied by the coefficients chosen by the GA. Therefore, a negative sign for the coefficient will mean that the larger the corresponding term, the more likely to occur the binding.

We did not however restrain any of the parameters to be positive or negative according to preconceptions about their roles in binding, but instead allowed the GA to freely sample all space of possible solutions. Obtaining meaningful signs for the various terms

---

[20] The format is that shown in section 5.4.

was, we could say, our first checkpoint. After initial teething problems, our GAs showed remarkable consistency in this respect.

5.8.3 GA internal parameters

For the GA parameters, **Mutation** is performed on individuals with a default probability **(pMut)** of 1, as is **crossover (pXover)**. These two probabilities can be modified between the generations, and currently the code adapts the rate of change to the number of generations pre-declared at start time. In our case, **pXover** was decreased linearly[JW34] from 1 at generation 0, to reach 0 at the last generation; **pMut**, on the other hand, is nil to begin with, rising linearly up to 1 at half the run, and then decreasing to zero again in the remaining half. Of the various possibilities explored, this seemed to give the best results in terms of efficiency of sampling and convergence, initially with the random generation of population and the crossover exploring all space available, and then not interfering with the fine tuning performed by the mutation, picking the best solution around the global optimum.[21]

The GA runs by default in *elitist* mode, which means that the best n (n=1 for us) members of each generation are retained in the next one. At the moment, this can be turned off only using the proper GALib command-line keyword, to which QTGA still responds. It would be however unwise to do so, as we would risk losing a newly found very good solution just because by chance it was never selected during the roulette, no matter how big its share of the roulette wheel.

5.8.4 Fitness Function

The fitness function, the very core of the GA, is computed in a very similar way as in the previous study by Smith[49]. For every individual in the genome, its genes are multiplied by the related score term in each line of the input file. Values obtained are then summed (by line) to give an *Energy Value*[22] for each (protein/ligand) pose, following which every list is ordered. The resulting rank for the Crystallographically Observed Ligand Orientation (COLO) is the contribution to the fitness function given

---

[22] I called this value Energy since we would like this to correlate with the free energy change associated with the ligand and the protein binding, although this was not our objective.

by this protein. The rank is defined as the number of decoys with better ranking than the COLO. The rank obtained is normalised by the number of ligands in the list, to make for a direct comparison between them. In the case of many proteins, all normalised ranks are summed and then divided by the number of proteins. This number, (multiplied by 100) equals the (percentage) average rank of COLO inside the chosen parameterisation set. This percentage average rank is our final fitness measure. Every genome (individual) is then accorded a reproduction probability proportional to its fitness in a roulette-wheel based tournament. Because the number we obtain from the GA is a percentage average ranking, the global optimum will correspond to its lowest possible value; for a ligand competing with a thousand decoys, the minimum (best) and maximum (worst) possible ranking will be:

$$\begin{aligned} Rk_{min} &= 1/1001 = 0.0009 = 0.09\% \\ Rk_{MAX} &= 1001/1001 = 1.00 = 100.0\% \end{aligned} \tag{5.13}$$

Our GA will therefore attempt to find the combination of coefficients which minimizes this, so that the COLOs will, in general, have lower score than the decoys. This fits with our conception, based on thermodynamics, that active molecules should have a free-energy of binding more negative than inactive compounds.

5.8.5 Fitness Scaling

At the beginning of the run the task of the GA is quite clear, with very few good solutions within a population of generally bad ones[23]. Having a larger share of the roulette wheel, the good solutions are selected in greater proportion for the next generation, so they see their frequency increase as generations go by, and the refinement process continues. Soon enough, however, the task becomes more complicated, as good solutions start to dominate the population. Shuffled around the global optimum by the mutation, their fitness will be similar, and so will be their proportions in the next generation. In order to avoid a stall situation where there is no definite improvement over time, the difference between the fitness can be artificially increased with a standard scaling procedure. The simplest technique of scaling is the so called linear scaling, where the new fitness ***Fit'*** bears a linear relationship with the original.

$$Fit' = A + B \cdot Fit \tag{5.14}$$

---

[23] In practical terms, a random classifier will have Percentage Average Ranking of 50%. This is the average of the population at the start.

In equation (5.14), A and B are parameters usually related to the distribution of fitness values within the population, chiosen so that the relative 'spread' of the scaled fitness *Fit'* is larger than the equivalent for the unscaled fitness *Fit*. Many types of scaling are available in the GALib through QTGA, such as power law and σ truncation[190], with some serving peculiar fitness functions: our preliminary tests did not show any improvement with respect to the most basic linear scaling, so they were not considered further, and linear scaling was used.

In the next chapter, we will start to analyse the results of our GA runs, in the perspective of the chemical problem at hand.

5.9 Conclusions

In this chapter the implementation details of the scoring terms has been presented, trying to weave together science and computational details. Also, the Genetic Algorithm technique and its implementation within the project have been introduced. An *in-depth* look at the programs' input and output files was given, and how these programs are used to actually parameterise the function. In the next chapters, the results of the said parameterisation will be presented and critically discussed[JW35].

6 Scoring Function Parameterisation

6.1 Parameterisation procedure

As previously stated, our aim was to train the scoring function(s) to recognize true actives. This would ideally be achieved by mixing some experimentally determined ligand poses (references) to decoy docked poses, generated by a docking program to mimic interactions between inactive compounds and targets of pharmaceutical interest. For each of these decoys, and COLO, a series of scoring terms was evaluated, such as the 24 scoring terms whose nature and implementation have been extensively described in the previous chapter. These 24 values were generated for both decoy poses and COLOs during a re-scoring procedure. GOLDSCORE values and its three components, as given by GOLD's output files, were added to these, giving a total of 28 scoring terms, detailed in Table 6-1.

In the paragraph above, we generically referred to 'reference ligand poses': in particular, these can be represented by true, Crystallographically Observed Ligand Orientations (COLOs); the same orientations, but energy-minimised making use of a particular scoring function. Finally, they could be non experimental, but still realistic, poses generated by re-docking the known ligand in the active site. In this particular study, two sets of such poses were considered. Upon GOLD[40] re-docking, either the pose closest to the COLO, or the one with the highest GOLDSCORE, were chosen. In the rest of the study, these will be referred to as *Closest*, and *Goldest*, respectively. GA input files were therefore assembled for all four reference sets: COLOs, COLOs simplex minimised within GOLDSCORE, *Closest*, and *Goldest*.

6.2 Correlation between terms

As shown in 5.3 and Table 6-1, most of the terms computed by the scoring shared common functional forms, so a certain degree of correlation is to be expected. This can in fact be a source of error in the parameterisation of the scoring function: since two correlated terms tends to describe the same interactions, when both are present at the

same time in a given combination of scoring terms it is their overall contribution that is important.

N°	NAME	Description
1	GoodHBond	Number of donor-acceptor pairs in the correct interaction geometry[24];
2	BadHBond	Number of donor-donor and acceptor-acceptor close contacts[25];
3	UnsDonor	Number of unsatisfied donor atoms;
4	UnsAccept	Number of unsatisfied acceptor atoms;
5	GoldHBond	Gold-like hydrogen bond model;
6	Coul	Protein-ligand coulombic interaction;
7	4R Coul	Protein-ligand coulombic with 4R distance-dependent dielectric;
8	GBSA	Generalised-Born surface area solvation term. Pairwise descreening approximation;
9	CouLnGBSA	Sum of coulombic and GBSA terms;
10	8A Coul	Protein-ligand coulombic with 8 Å cut-off;
11	4R-8ACoul	Protein-ligand coulombic with 4R distance-dependent dielectric and 8 Å cutoff;
12	CL8AnGBSA	Sum of 8 Å coulombic (N° 10) and GBSA(N° 8);
13	LigHPhilB	Ligand hydrophilic buried solvent accessible surface area;
14	LigHPhobB	Ligand hydrophobic buried solvent accessible surface area;
15	TotHPhilS	Total hydrophilic buried solvent accessible surface area;
16	TotHPhobS	Total hydrophobic buried solvent accessible surface area;
17	LigHPhilS	Ligand hydrophilic solvent accessible surface area;
18	LigHPhobS	Ligand hydrophobic solvent accessible surface area;
19	4/8ExtVdW	Protein-ligand 4/8 van der Waals interaction (also as F_EExtVdW[JW36]);
20	6/9ExtVdW	Protein-ligand 6/9 van der Waals interaction (also as S_NExtVdW);
21	6/12ExVdW	Protein-ligand 6/12 van der Waals interaction (also as S_TExtVdW);
22	4/8IntVdW	Ligand-ligand 4/8 van der Waals interaction (also as F_EIntVdW);
23	6/9IntVdW	Ligand-ligand 6/9 van der Waals interaction (also as S_NIntVdW);
24	6/12InVdW	Ligand-ligand 6/12 van der Waals interaction (also as S_TIntVdW[JW37]);
25	GOLDSCORE	GOLD's own fitness score...as taken from .rnk files;
26	GOLDHBOND	GOLD's own hydrogen bond score;
27	GOLDEXVdW	GOLD's own van der Waals score;
28	GOLDITORS	GOLD's own ligand torsion score;

Table 6-1: The scoring terms' numerical IDs, names and brief descriptions (the aliases in the VdW terms from 19 to 24 are the actual name in the code).

They can therefore assume a set of different values which will describe equally well the training set, but may fail to adequately predict results for the test set. They would in this case be overlapping[JW39], playing the same role in our rough description of binding energy. This would allow them to alter their relative importance while keeping an overall good fitness. To ascertain whether this is the case, all 28 terms were plotted one against each other, looking for correlations both visually and by statistical means. The software used to perform this was **R** [220], a free and open-source statistical suite of tools, built to be a replacement for the commercial package **S/S+**. The statistical analysis protocol followed is detailed below.

As mentioned previously, every ligand/protein pose is represented by a row in the GA input file. The structure of the file and the rows within has been explained in detail in the previous chapter (5.4). For the sake of computational efficiency, and for clarity in the resulting plots, the statistical analysis was performed on a limited sample of the input files. Therefore only a certain number of poses (namely, 1000 at random, or only the references) were selected, and correlations between their scores sought.

Initially, the references subset was selected to investigate the occurrence of extreme scoring values (especially on van der Waals terms) for some COLOs. The proteins whose references turned out to be affected by this were subsequently eliminated from the training set to avoid bias. By having anomalous values of a certain scoring term, if included in the parameterisation they would have caused a greater than normal contribution from that term. The same set was suitable for the statistical analysis, since every ligand is different from every other, and the same can be said for every protein, so we are less likely to sample any specific weird interaction, as could happen if we were to use the same ligand on all proteins.

In order to have an extensive collection of poses, a randomly sampled 1000 set of proteins/decoy was selected. We aimed this way to escape possible bias due to the fact that the COLOs are by their own nature *special*: first, they are all *active* compounds; second, they are not generated by GOLD but obtained through crystallographic refinement[JW40]. From these reduced sets, all possible pairs of scoring terms were generated, and **R** computed for them all the linear correlation coefficients, r, and the determination coefficient r^2. The script that automatically did this also flagged as

suspect any r^2 value greater than 0.5, checking whether the two terms were from the same interaction class (therefore correlation was to be expected) or not. At the same time, **R** produced *x/y* plots of the two scoring terms, to complement the statistical analysis with a visual inspection. All this was done for every single input file, the original with COLOs, the *Closest* and *Goldest*, and the simplexed COLOs (provided within the CCDC/Astex set release). Since no significant differences were observed for the various cases, either when the references are chosen or when the random set of ligands was taken, reported here as an example is the statistical analysis for the *COLO* set, using both the reference ligands and a randomly selected 1% sample of the decoys – the correlation matrix can be seen in Table 6-2.

First, one can notice[JW41] how references and decoys have very similar correlation patterns, although they may vary in the details; overall, COLOs terms have a higher degree of correlation between themselves – more cells are coloured yellow. Most importantly, though, is the obvious correlation between terms of the same class, which create the orange blocks along the diagonal. Also of interest is the high correlation of GOLDEXVDW with hydrophobic surface measurement terms (and UnsAccept) for the COLOs, a situation not seen for the decoys. Conversely, the decoys' GOLDSCORE and GOLDITORS are highly correlated, whereas this is not observed for the reference poses.

R	GoodHBond	BadHBond	UnsDonor	UnsAccept	GoldHBond	Coul	4R Coul	GBSA	CouLnGBSA	8A Coul	4R-8ACoul	CL8AnGBSA	LigHPhilB	LigHPhobB	TotHPhilS	TotHPhobS	LigHPhilS	LigHPhobS	4/8ExtVdW	6/9ExtVdW	6/12ExVdW	4/8IntVdW	6/9IntVdW	6/12InVdW	GOLDSCORE	GOLDHBOND	GOLDEXVdW	GOLDITORS
GoodHBond		0.09	0.21	0.22	0.27	-0.39	-0.37	0.40	0.40	-0.38	-0.29	0.40	0.60	-0.27	0.58	-0.23	0.56	-0.22	0.35	0.60	0.51	0.23	0.27	0.32	0.69	0.45	0.13	0.08
BadHBond	0.20		-0.05	-0.22	-0.14	-0.12	-0.15	0.12	0.12	-0.19	-0.18	0.12	0.13	-0.27	0.06	-0.27	0.10	-0.25	0.17	0.00	-0.06	-0.04	-0.04	-0.06	0.07	-0.17	-0.19	-0.06
UnsDonor	0.25	0.10		0.64	-0.02	-0.07	-0.02	0.08	0.08	0.02	0.09	0.08	0.27	0.28	0.45	0.28	0.29	0.28	-0.25	0.10	0.12	0.04	0.05	0.04	0.12	0.13	0.46	0.26
UnsAccept	0.56	0.08	0.18		0.18	0.01	0.10	0.00	0.00	0.16	0.14	0.00	0.22	0.55	0.41	0.56	0.23	0.54	-0.43	0.15	0.21	0.10	0.12	0.15	-0.05	0.15	0.76	0.38
GoldHBond	0.40	0.09	0.15	0.05		0.09	0.15	0.17	0.17	-0.08	-0.08	0.15	0.22	-0.09	0.22	-0.06	0.18	-0.06	0.05	0.14	0.11	0.38	0.39	0.39	0.23	0.17	0.02	0.15
Coul	-0.40	-0.09	0.01	0.04	0.05		0.96	-0.95	-1.00	0.91	0.72	-1.00	-0.43	0.24	-0.36	0.21	-0.43	0.25	-0.29	-0.43	-0.40	0.07	0.06	0.04	-0.21	-0.34	-0.05	-0.04
4R Coul	-0.36	-0.09	0.02	0.03	0.05	0.96		-0.95	-0.95	0.97	0.80	-0.95	-0.40	0.26	-0.33	0.22	-0.42	0.26	-0.29	-0.38	-0.36	0.07	0.06	0.05	-0.17	-0.36	0.02	-0.04
GBSA	0.40	0.09	0.01	-0.03	-0.05	-1.00	-0.95		1.00	-0.91	-0.72	1.00	0.43	-0.23	0.37	-0.21	0.44	-0.24	0.29	0.44	0.40	-0.06	-0.05	-0.04	0.22	0.34	0.05	0.04
CouLnGBSA	0.40	0.09	0.01	-0.03	-0.05	-1.00	-0.95	1.00		-0.91	-0.72	1.00	0.43	-0.23	0.37	-0.21	0.44	-0.24	0.29	0.44	0.40	-0.06	-0.05	-0.04	0.22	0.34	0.05	0.04
8A Coul	-0.38	-0.19	0.02	0.16	0.17	0.91	0.97	-0.91	-0.91		0.85	-0.91	-0.33	0.28	-0.23	0.30	-0.40	0.33	-0.35	-0.27	-0.29	0.08	0.07	0.08	-0.11	-0.40	-0.09	0.06
4R-8ACoul	-0.29	-0.18	0.09	0.14	0.17	0.72	0.80	-0.72	-0.72	0.85		-0.72	0.23	-0.23	0.28	-0.21	0.23	-0.26	0.25	0.26	0.29	-0.07	-0.06	-0.05	0.26	0.36	-0.08	-0.01
CL8AnGBSA	0.40	0.12	0.08	0.00	0.08	1.00	0.95	1.00	1.00	0.91	0.72		0.43	0.23	0.37	0.21	0.44	0.24	0.29	0.44	0.40	0.06	0.05	0.04	0.22	0.34	0.05	0.04
LigHPhilB	0.60	0.13	0.27	0.27	0.22	0.43	0.40	0.43	0.43	0.33	0.33	0.43		0.27	0.23	0.30	0.40	0.33	0.35	0.40	0.51	0.23	0.27	0.32	0.69	0.45	0.13	0.08
LigHPhobB	-0.27	-0.27	0.28	0.55	-0.09	0.24	0.26	-0.23	-0.23	0.33	0.28	0.23	0.27		0.41	0.56	0.23	0.54	-0.43	0.15	0.21	0.10	0.12	0.15	-0.05	0.15	0.76	0.38
TotHPhilS	0.58	0.06	0.45	0.41	0.22	0.37	0.33	0.37	0.37	0.29	0.23	0.37	0.23	0.41		0.28	0.29	0.28	-0.25	0.10	0.12	0.04	0.05	0.04	0.12	0.13	0.46	0.26
TotHPhobS	-0.23	-0.27	0.28	0.56	-0.06	0.21	0.22	-0.21	-0.21	0.30	0.23	0.21	0.23	0.56	0.28		0.27	0.97	-0.87	-0.87	-0.84	-0.47	-0.41	-0.41	-0.31	-0.29	-0.27	-0.08
LigHPhilS	0.56	0.10	0.29	0.23	0.18	-0.43	-0.42	0.44	0.44	-0.40	-0.34	0.44	0.40	0.23	0.29	0.27		0.97	-0.87	-0.87	-0.84	-0.47	-0.41	-0.41	-0.31	-0.29	-0.27	-0.08
LigHPhobS	-0.22	-0.25	0.27	0.54	-0.06	0.18	0.26	-0.24	-0.24	0.33	0.26	0.24	0.22	0.54	0.23	0.26	0.27		-0.87	-0.87	-0.84	-0.47	-0.41	-0.41	-0.31	-0.29	-0.27	-0.08
4/8ExtVdW	0.35	0.17	-0.25	-0.43	0.05	-0.29	-0.29	0.29	0.29	-0.35	-0.27	0.29	0.31	-0.87	-0.87	-0.87	-0.87			1.00	1.00	0.03	0.03	0.03	0.02	0.02	0.01	0.01
6/9ExtVdW	0.60	0.00	0.10	0.15	0.23	-0.43	-0.38	0.44	0.44	-0.37	-0.28	0.44	0.53	-0.38	0.37	-0.25	0.44	-0.24	0.29		1.00	0.03	0.03	0.03	0.02	0.02	0.01	0.01
6/12ExVdW	0.51	-0.06	0.12	0.21	0.11	-0.40	-0.36	0.40	0.40	-0.33	-0.25	0.40	0.53	-0.25	0.37	-0.22	0.44	-0.24	0.29	0.44		0.03	0.03	0.03	0.02	0.02	0.01	0.01
4/8IntVdW	0.23	-0.04	0.04	0.10	0.07	-0.07	-0.06	0.08	0.08	-0.06	-0.05	0.08	0.18	-0.17	0.12	-0.17	0.15	-0.16	0.19	0.18	0.19		1.00	1.00	0.63	0.47	0.12	0.05
6/9IntVdW	0.27	-0.04	0.05	0.12	0.06	-0.07	-0.06	0.08	0.08	-0.06	-0.05	0.08	0.19	-0.18	0.13	-0.18	0.16	-0.17	0.19	0.19	0.18	1.00		1.00	0.63	0.47	0.12	0.05
6/12InVdW	0.32	-0.04	0.05	0.12	0.07	-0.07	-0.06	0.08	0.08	-0.06	-0.05	0.08	0.18	-0.17	0.12	-0.17	0.15	-0.16	0.19	0.18	0.19	1.00	1.00		0.63	0.47	0.12	0.05
GOLDSCORE	0.45	-0.17	0.13	0.15	0.23	-0.21	-0.17	0.22	0.22	-0.16	-0.11	0.22	0.33	-0.15	0.22	-0.15	0.27	-0.15	0.14	0.23	0.16	0.10	0.14	0.20		0.33	0.15	0.03
GOLDHBOND	0.69	0.07	0.12	-0.05	0.17	-0.34	-0.36	0.34	0.34	-0.40	-0.38	0.34	0.47	-0.48	0.34	-0.47	0.40	-0.45	0.52	0.50	0.40	0.31	0.32	0.32	0.52		0.05	0.05
GOLDEXVdW	0.13	-0.19	0.46	0.76	0.02	-0.05	0.02	0.05	0.05	0.09	0.08	0.05	0.16	0.75	0.38	0.79	0.18	0.74	-0.67	-0.02	0.07	-0.09	-0.08	-0.05	0.16	0.00		0.12
GOLDITORS	0.08	-0.06	-0.26	-0.38	0.15	-0.04	-0.04	0.04	0.04	-0.06	-0.01	0.04	0.02	-0.50	-0.11	-0.54	-0.03	-0.51	0.42	0.04	-0.07	0.03	0.06	0.10	0.71	0.25	-0.48	

Table 6-2: Correlation matrix for the 28 terms shown in Table 6-1: above the diagonal the values of the correlation coefficient r relative to a random 1% selection of decoy poses, and below it for the COLO references: highlighted in light yellow are values larger than 0.3 (or smaller than -0.3); in orange are values higher than 0.7, or lower than -0.7.

R^2	GoodHBond	BadHBond	UnsDonor	UnsAccept	GoldHBond	Coul	4R Coul	GBSA	CouLnGBSA	8A Coul	4R-8A Coul	CL8AnGBSA	LigHPhilB	LigHPhobB	TotHPhilS	TotHPhobS	LigHPhilS	LigHPhobS	4/8ExtVdW	6/9ExtVdW	6/12ExVdW	4/8IntVdW	6/9IntVdW	6/12InVdW	GOLDSCORE	GOLDHBOND	GOLDEXVdW	GOLDITORS
GoodHBond		0.04	0.06	0.01	0.16	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.07	0.01	0.13	0.03	0.07	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.24	0.00	0.00
BadHBond	0.01		0.01	0.01	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.06	0.02	0.11	0.03	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.01	0.04	0.00
UnsDonor	0.04	0.00		0.32	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.07	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00
UnsAccept	0.07	0.02	0.40		0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.05	0.00	0.00	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00
GoldHBond	0.21	0.08	0.07	0.01		0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.11	0.06	0.16	0.06	0.13	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Coul	0.14	0.00	0.03	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4R Coul	0.02	0.00	0.03	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
GBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
CouLnGBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
8A Coul	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4R-8A Coul	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
CL8AnGBSA	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LigHPhilB	0.07	0.03	0.01	0.00	0.03	0.01	0.00	0.00	0.00	0.00	0.00	0.00		0.28	0.06	0.00	0.03	0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LigHPhobB	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.28		0.03	0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
TotHPhilS	0.13	0.07	0.02	0.00	0.07	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.06	0.03		0.00	0.00	0.05	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
TotHPhobS	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LigHPhilS	0.07	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
LigHPhobS	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4/8ExtVdW	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6/9ExtVdW	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
6/12ExVdW	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00	0.00
4/8IntVdW	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00	0.00
6/9IntVdW	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00
6/12InVdW	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00
GOLDSCORE	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00
GOLDHBOND	0.01	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00
GOLDEXVdW	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00
GOLDITORS	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	

Table 6-3: Determination matrix for the 28 terms shown in Table 6-1: above the diagonal the values of the determination coefficient r^2 relative to a random 1% selection of decoy poses, and below it for the COLO references: highlighted in light yellow are values larger than 0.3 (or smaller than -0.3); in orange are values higher than 0.7, or lower than -0.7.

All these differences probably arise from the different origins of the two sets, but a detailed explanation of the reasons behind this is beyond our scope: the important conclusion to draw from this is that similar terms do in fact show a somewhat high degree of correlation, and it would be wise to exclude one when the other is already present in the parameterisation.

6.2.1 COLO: references and decoys

As we already observed, the general trend is for low or negligible correlation, as long as the terms belong to different classes. Within the same class, instead, correlations are observed, and frankly expected; for example, between different kind of protein/ligand van der Waals interactions (X/YExtVdW, where X and Y are the attractive and repulsive power): 6/9, 4/8 and 6/12 are all correlated with each other: $r > 0.99$ for the decoys, although correlation for the COLOs decreases with difference in the attractive/repulsive powers; $r = 0.96$ for the pair 6/9 – 6/12, but 4/8 – 6/12 do not seem to be very correlated at $r = 0.51$; correlation [JW43]is slightly higher between 4/8 and 6/9 ($r = 0.66$).

Correlation between these three terms and the GOLDEXTVDW also turns out to be lower than expected, with all ***r*** around ***0*** except in the case of the 4/8ExtVdW: in this case *r= -0.67* for COLOs, but only *.01* for the decoys (see Figure 6-1 for the correlation between GOLDEXTVDW and 6/9ExtVdW, and Figure 6-2 for 4/8ExtVdW).

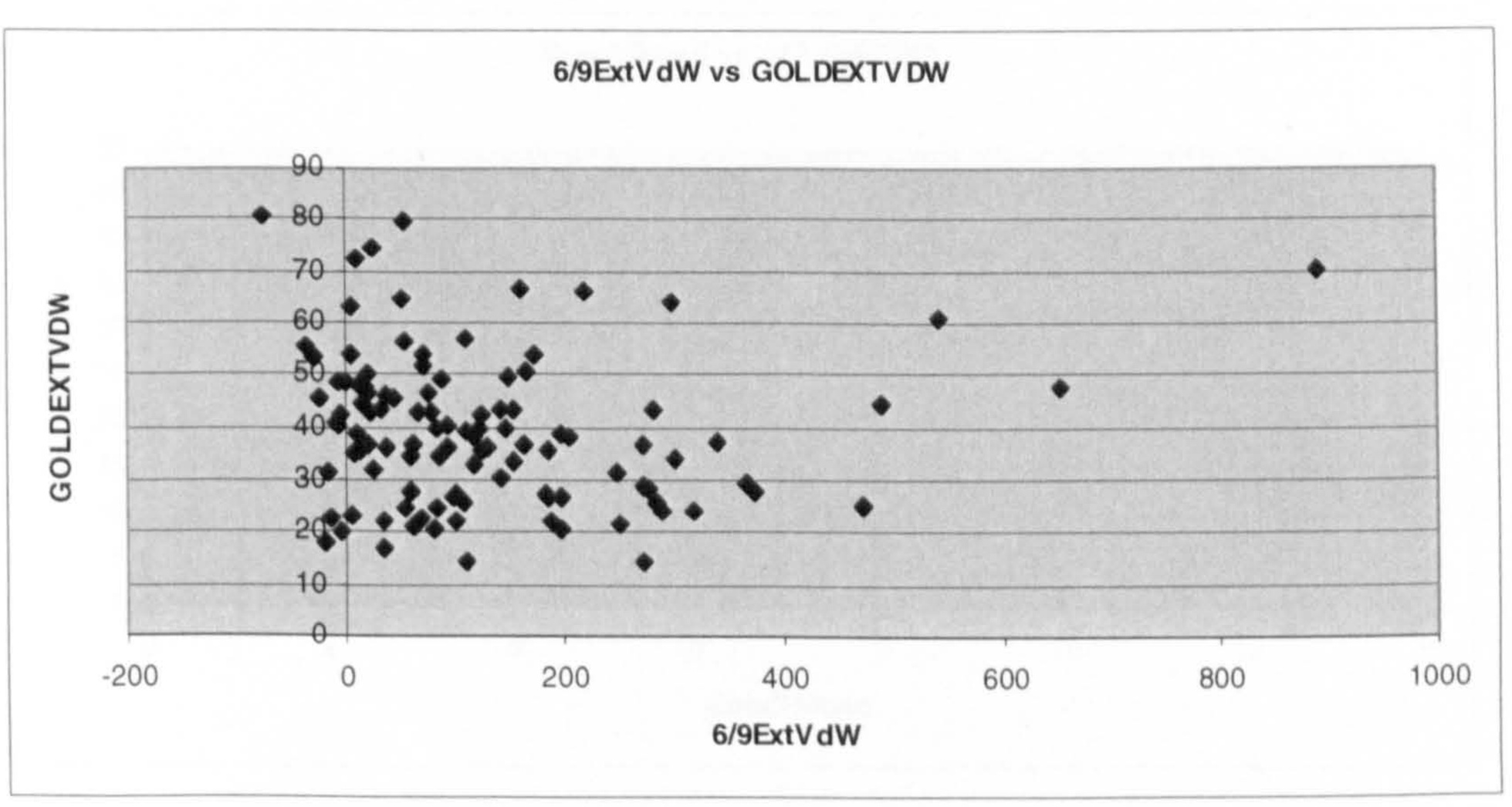


Figure 6-1: Correlation plot for the two terms GOLDEXVDV and 6/9ExtVdW, for the 108 Crystallographic reference poses (COLOs) - *r=-0.02*.

Noticeably, GOLDEXVDW uses a 4/8 model, which may account for the similarity with our own 4/8 terms, although the differences in implementation and parameters give an overall negative correlation.

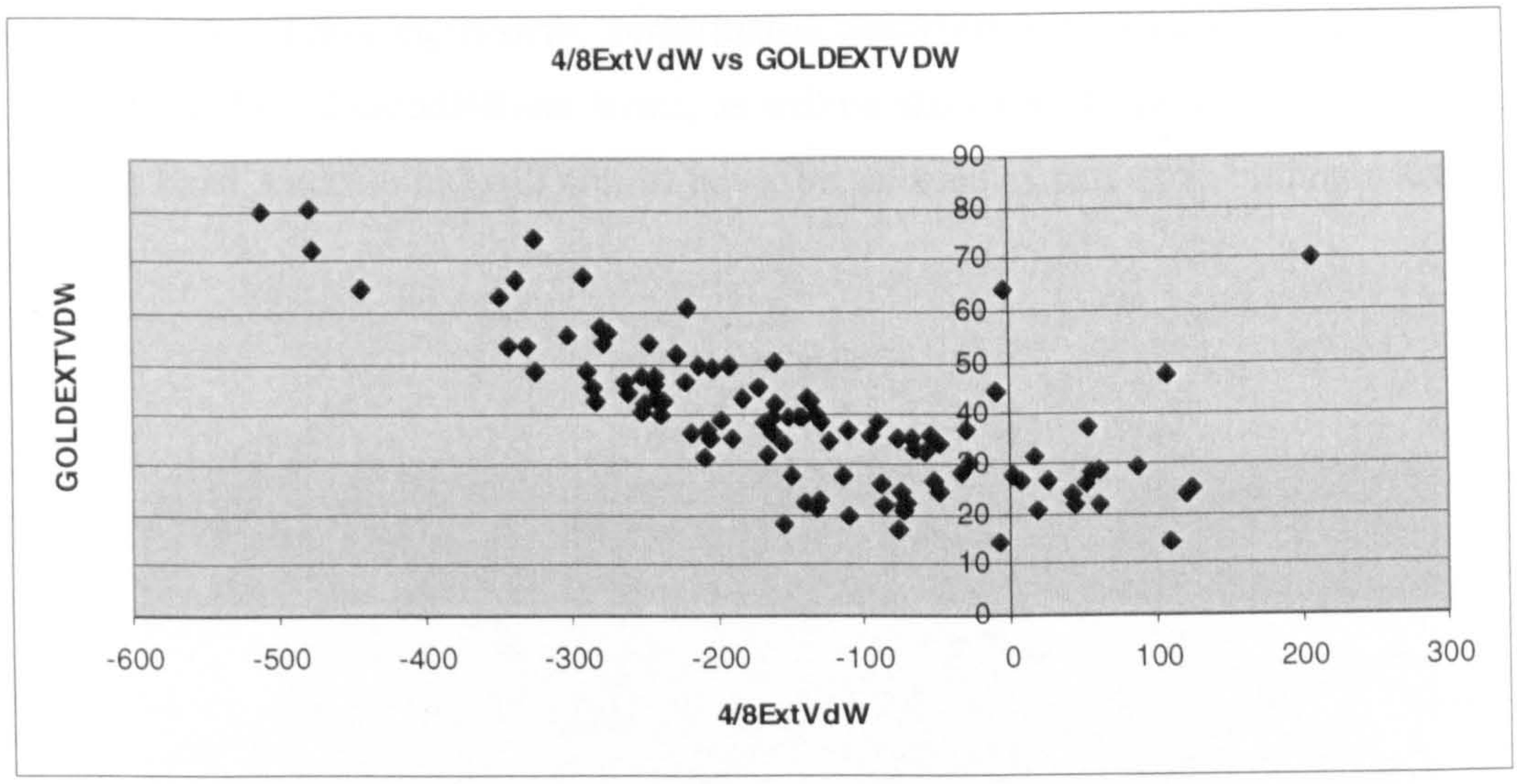


Figure 6-2: Correlation plot for the two terms GOLDEXVDV and 4/8ExtVdW, for the 108 Crystallographic reference poses (COLOs) - *r=-0.67*.

While the negative value is not a cause for concern, the lower than expected correlation is. However, this seems to anticipate the fact that GOLDEXTVDW and 4/8ExtVdW will have similar performances, as will be shown in the next sections.

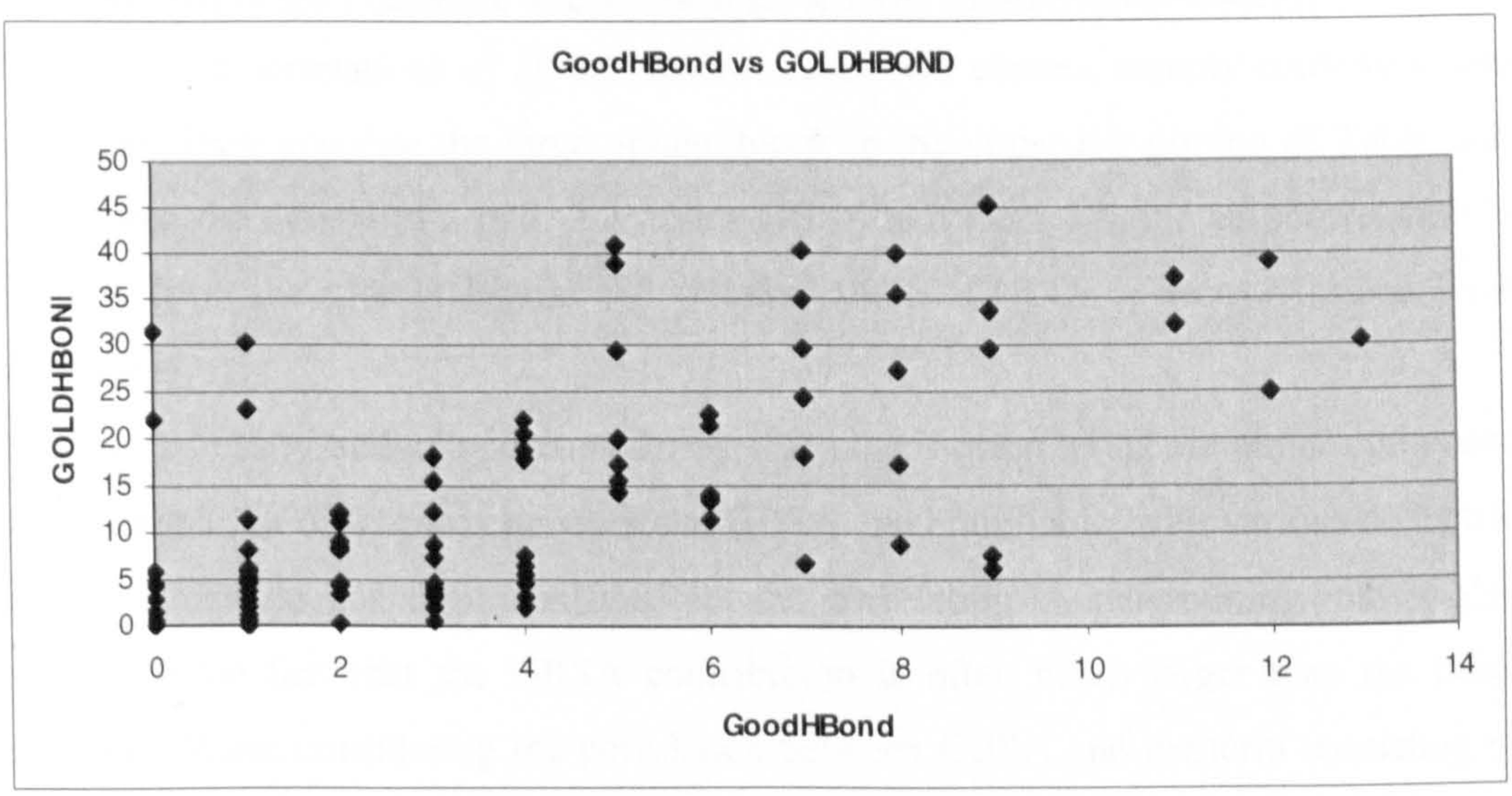


Figure 6-3: Correlation plot for the two terms GOLDHBOND and GoodHBond, for the 108 Crystallographic reference poses (COLOs) - *r=-0.69*.

A good correlation is shown in Figure 6-3 between our simple, discrete GoodHBond and GOLDHBOND ($r = 0.69$, for the references, but only $r = 0.49$ for the decoys); whereas our own simplified implementation of the latter (aptly named GoldHbond) shows lower correlation with the GOLDHBOND, probably due to an oversimplification – r is only 0.32 for the references, and 0.17 for the decoys. Once again, this rationalises

the very good, almost equivalent, performance observed in ranking references for the GOLDHBOND and GoodHBond terms, as will be shown in the following sections: on the other hand, GoldHbond will almost never be selected as part of an optimum scoring function.

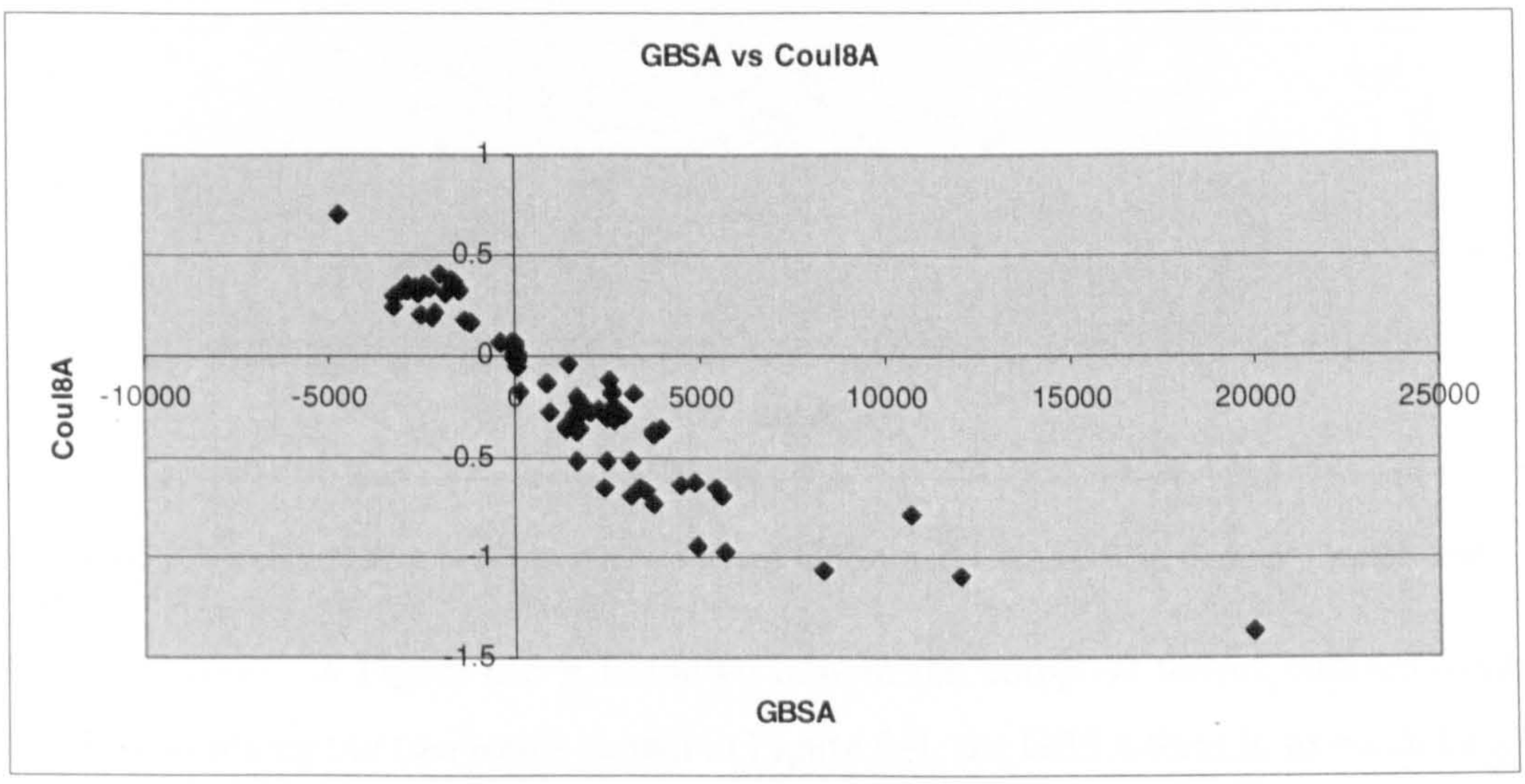


Figure 6-4: GBSA and Coulombic with 8Å cutoff are strongly anti-correlated - *r=-0.91*.

The strongest correlations of all are shown across two classes, namely coulombic and solvation. They generate the large orange block in the upper left portion of Table 6-2. Going into the details of a pair, *e.g.* terms GBSA and Coul 8Å, the anti-correlation is evident, from the plot in Figure 6-4 (relative to the COLOs – decoys instead have $r = -0.93$).

This can be easily understood considering that all solvation terms are either composed of coulombic, or differences between the GBSA and coulombic, with various dielectric models which do not significantly affect the correlation. A determining role is also played by the fact that the GBSA contribution is often much larger than the other members. When considering the correlation between GBSA and the term consisting of GBSA and coulombic with 8 Å cut-off, the contribution from GBSA totally dominates the term. These results also raise doubt as to whether it is reasonable to use such a computationally demanding method (GBSA) to obtain little extra information compared to what the cheaper coulombic would provide. At least at the level of a preliminary screening of a database, which is the aim of our function, GBSA may be unnecessary. Such doubts are deepened by the fact that the performance of GBSA will be shown to be not very decisive during the selection of terms by the GA.

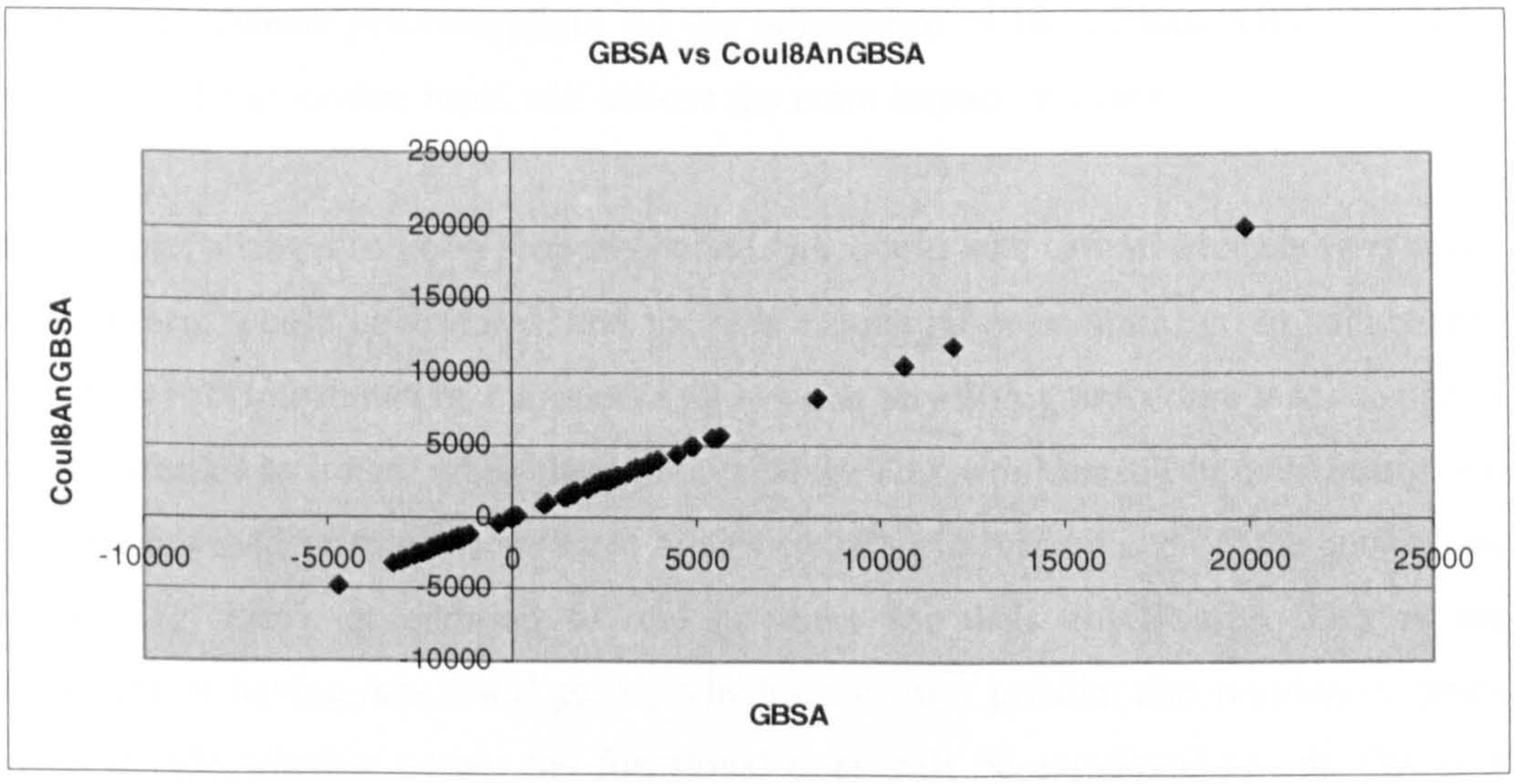


Figure 6-5: Correlation Plot between the two terms GBSA and Cl8AnGBSA (sum of Coul8A and GBSA) – *r = 1*.

As a final proof, in Figure 6-5 it is shown how in the composit term Coul8AnGBSA, made up summing the two terms shown in Figure 6-4, the GBSA term is so much larger than the Coul8A that it completely dominates the value, flattening the correlation plot to a straight line (correlation coefficient $r = 1$).

Following these analyses, it is clear that correlated terms should not be present in the function. Constraining on the basis of computed ***r*** values would have been an ideal option, but it had an unwanted consequence: depending on the detailed value of ***r***, some term pairs would have been prohibited with some input files, but not with others. This would have rendered the comparison of results much harder. Therefore, the decision was made to constrain only terms belonging to the same class from being simultaneously present. Any other fine tuning will be left to the GA and the competition between terms, which even unsupervised should get rid of maladaptive (low-fitness collections of highly correlated terms) combinations.

6.3 Selecting the right terms

The GAs task was not only to select the correct coefficient values to yield the best possible scoring function, but it would be ideal to have an automated routine that decides which terms were the best based solely on the enrichment performances[26]. In

this way, no *human* preconceptions on the importance of the various terms would be involved[27]: the algorithm itself will choose the more important terms.

In principle, a single run of a properly coded GA, could take care of everything at once: the best term would be selected, and the less influential ones would go to values very close to zero. Unfortunately, experience shows that any fitting procedure tends to use as many variables as it can, when they are available. This would result in over-fitting. An alternative would be to use a series of binary switches to turn on and off the genes (one per scoring term), in addition to real numbers for their coefficients. This is the equivalent of having *functional* genes, which code for a protein, and *regulatory* genes which decide whether or not the functional ones will be expressed or not: this is a common sight in real life genetics, where however the switches are modulatable and not simply on-off. However, this did not seem to be the ideal solution: the already difficult task of optimising the coefficients to *enrich* not one but many proteins' ranked lists could have been disrupted at any moment, when a mutation would turn on/off a scoring term, thereby moving the solution to a completely different region of the solution space. A considerable amount of computational power was available for the task in the coarsely parallel form of a newly built Linux cluster, and quite easily a single processor could be assigned to the optimisation of a single combination of terms. The results were then analysed to select the best combinations. This was the route chosen.

A recent paper by Giordanetto et al.[68] shows that, in a similar case, a scoring function composed of up to seven/eight terms is the optimum, with four-to-six term combinations missing some descriptive power, and those above eight terms incurring in over-fitting, and therefore decreasing their predictive power on test cases. Even with only seven/eight members, the amount of computation needed to select the best performer out of all possible combinations of terms (and coefficients) was too high. Given our 28 scoring terms presented in Table 6-1, if, to select the best 7-term scoring function, one were to exhaustively optimise every single possible combination, this would require selecting the first out of 28 possibilities: the second, out of the remaining 27 terms left; the third, and so on. To find the best seven-term combination one should evaluate all the possible 7-strong permutations of a 28-strong set. That is mathematically expressed as:

$$\frac{28!}{(28-7)!} = 28\times27\times26\times25\times24\times23\times22 = 1.38\times10^{9}\text{combinations.} \qquad (6.1)$$

Preliminary runs of the parameterising Genetic Algorithm suggested that as a rule-of-thumb, an additional 100 generations had to be run for each term added[28], and each 100-generations run on 1000 individuals requires one hour on single-cpu, hi-specification machines (AMD Opterons, 3.6GHz). This brings us to a grand total of 1.4 billion computer hours. Such an exhaustive, and mindless, optimisation was therefore out of question.

Steps were taken to overcome this *impasse*. First, our scoring terms were already grouped by classes, corresponding to generic interaction modes such as 'hydrogen bond', 'coulombic', 'solvation', and so on (see section 5.3 – previous chapter). Care was taken in the next steps never to include two terms from the same class in the same scoring function. There is an obvious rationale behind this. Having two terms describing the same interaction is not redundant, but can actually be harmful. Indeed, the predictive power of the model may decline, as terms strongly correlated, such as those belonging to the same class, can 'exchange' influence, when present in the same scoring function. This may not alter performance on the training set, but this may (and will) change in the real applications. This is an example of the risk of *over-fitting*. Not selecting terms from the same class will decrease the 1.4 billion figure by a great degree, but even avoiding these overlaps, the sheer number of possible combinations is still too high with respect to our computational resources.

But do we really need to get the best scoring function for all terms at once? Is it not possible to grow the scoring function? Since we are going to use Genetic Algorithms to select this function, we can draw upon the evolutionary analogies. Like the human eye, our scoring function needs a careful and balanced assembly of pieces to work seamlessly. But having a function limited in number of terms, like having a rudimentary eye, is better than having none at all. So, we can quote Dawkins when he says that '***Vision that is 5 per cent as good as yours or mine is very much worth having in comparison with no vision at all. So is 1 per cent vision better than total blindness. And 6 per cent is better than 5, 7 per cent better than 6, and so on up the gradual, continuous series***'[221]. Therefore, a bottom-up approach was devised, where the optimal scoring function would be built by assembling new terms on top of the best

performing scoring function already selected. Rather than a continuous series, there is a discrete series, here, since at every step we add a little piece to our function. It is like building our rudimentary eye by mixing together lots of anatomical pieces which may or may not compose the real human eye. The highly parallel process of selection and recombination that is the GA will allow us to select the best two-pieces assembly: this may be the pair retina + ball, for the eye; hydrogen-bond + hydrophilic surface for our function. And by building on top of that, the best three-term assembly will be identified, and so on.

As a first step, all the possible two-term combinations were considered: with our particular choice of terms and classes, these combinations numbered a total of 343, manageable in a few hours computer time at the most. Alternatively, a start from a single-term function was evaluated. Lacking the interplay with even one other term, the only useful information from such runs is the sign with which each term should be taken, on its own. Drawing again on our analogy, if you only consider the ball cavity, the only thing you get is darkness: if you consider only the retina, it turns out to be blind outside the eyeball, but that is going to change dramatically if the two are put together. The same is true for the signs by which a single term is taken: on its own a particular term will better rank the COLOs with a minus sign, once it interacts with another term it may well turn its role into a penalty, therefore picking up a plus sign.

The ranking power of single term functions is especially unreliable in the case of terms which return integer numbers, as the ranking routine may fail: that is, integer scores applied by themselves yield identical rankings for compounds with the same score. To make things clearer, consider two terms, the first returns a floating point value for each pose, such as GOLDHBOND. So, it is unlikely that the COLO will have *exactly* the same score as any of the decoys[JW46]. If the GA assigns a negative coefficient, the rank may be:

$$\text{Rank}^{\text{GOLDHBOND}}(\text{COLO},-) = 256/1001 = 0.2557 \qquad (6.2)$$

If we were to take the plus sign, we would instead have

$$\text{Rank}^{\text{GOLDHBOND}}(\text{COLO},+) = 745/1001 = 0.7443 \qquad (6.3)$$

The expected sum of the two ranks would be 1. So it is. However, in the case of a term returning integers, such as GoodHBOND, things change: many decoys will have the same results as the COLO, and therefore will share the same rank. If the COLO is in 56th position, which it shares with other 35, for example, we will have:

$$\text{Rank}^{\text{GoodHBOND}}(\text{COLO},-) = 56/1001 = 0.0559 \quad (6.4)$$

If we took the plus sign, we would instead have

$$\text{Rank}^{\text{GoodHBOND}}(\text{COLO},+) = (945+35)/1001 = 0.979 \quad (6.5)$$

Now the sum of the two ranks is not 1 anymore, but 1.035. *Ouch!* This cannot be avoided, as it is due to the fact that molecules with the same score must share the same rank.

The 343 two-term combinations were sufficiently few to be performed in several hours on our 600 CPU (AMD Opteron™) cluster: every GA was executed three times (runs A, B, C), for 600 generations[29], to make sure it had attained the global minima – the three runs were performed with different seeds initialising the random number generator: this made for different initial populations, and different mutations and crossover history. Other GA parameters were: a linearly decreasing crossover probability (from 1, at 0 generations, to reach 0, at 600 generations); mutation probability linearly increasing from 0 to its maximum during the first half, and then decreasing to 0. Varying the GA parameters in such a way was an attempt to maximise the search of adequate solutions in the first phase, leaving then the GA to settle down to the best region during the last phase, when the '*annealing*' of the population in the optimum region is not disrupted by crossovers, and a little mutation still helps to do a moderate local search.

Once all the two-term combinations were completed, the best thirty runs were selected: since every combination was run three times, this could in fact include between ten and thirty different combinations. Usually the three runs of a particular combination yielded very similar average rankings: from Table 6-4, where the results of the very first selection step, the two-term combination scanning, is shown, one sees that the first three

runs do all return an average ranking between 8.72 and 8.81%. That's a very narrow window, indicating that the three final populations are very similar – the best ranking individual in all scores the same, 8.68%. However, the details of the solution were somewhat different.

COMBINATION	Best Rank	Ave Rank
2_terms/14-19/A	8.68%	8.72%
2_terms/14-19/B	8.68%	8.75%
2_terms/14-19/C	8.68%	8.81%
2_terms/18-19/B	8.90%	8.93%
2_terms/18-19/A	8.90%	8.96%
2_terms/18-19/C	8.90%	8.96%
2_terms/16-19/B	11.76%	11.81%
2_terms/16-19/A	11.76%	11.88%
2_terms/16-19/C	11.76%	11.99%

Table 6-4: Best performers among the two-term combinations: An example outcome from the first ten combinations selected from two-term runs, parameterised against COLOs.

Let us spend some time explaining what exactly is shown in the aforementioned table. In the first column (COMBINATION) is the codename for the run, detailing the terms from which each combination is composed. *e.g.*, first line reads: 2 terms, terms #14 and 19, run A. The second column (Best Rank) shows the performance of the best ranking solution found during the GA – indicating that the active compound is found after scanning 8.7% of the ranked lists, on average[JW47] – in some proteins the reference will show up first, in some others at 16% of the list. The third column (Ave Rank) shows the average ranking of the COLOs in the whole population at the end of the GA – this is the percentage average ranking that the average solution is likely to assign to the COLO of the average protein. The difference between Best Rank and Ave Rank can be interpreted as a rough measure of convergence of the GA. The closer the two values, the more similar the average member of a population will be to the best member. As for the terms, the corresponding number-to-term is given in Table 6-1, page 95. For example, 13 means ligand hydrophilic buried surface. Surface measures and van der Waals terms are prominent in describing the physical interaction between protein and ligand.

To each of the combinations selected, that is to say, the best thirty GA runs, we add all possible terms from classes not already included. For example, for the first result in

Table 6-5, we cannot add any term belonging to class 'HydroPhobic Surfaces' (to which LigHphobS belongs) or 'Protein-Ligand van der Waals' (term 19 is the 4/8ExtVdWaals). But the addition of a 'hydrogen bond' term, such as term number 1, or 26, is welcome. The GA is then run once again on these newly generated combinations. The best three-term combinations are then identified.

Step-by-step, the procedure is repeated until there is no gain in ranking power, or we reach a pre-determined number of terms.

2_terms/14-19/A				
Term	LigHPhobB	4/8ExtVdW	SCORERANK	1st/2ndRATIO
Coefficient	52.9	68.6	8.68%	0.77
2_terms/14-19/B				
Term	LigHPhobB	4/8ExtVdW	SCORERANK	1st/2ndRATIO
Coefficient	44.1	57.1	8.68%	0.77
2_terms/14-19/C				
Term	LigHPhobB	4/8ExtVdW	SCORERANK	1st/2ndRATIO
Coefficient	25.5	33.0	8.68%	0.77

Table 6-5: Best performers for the three best runs of the two-term combinations.

As can be seen from Table 6-5, in general the three runs on the same combination will have very similar results in terms of performance. The GA task for two-term combinations is sufficiently easy that all three attempts will have almost identical performance. However, a detailed look to the results shows something unexpected.

Apparently, for one problem – since the three runs use the same term pair of terms - there are three very different solutions. Although the performance (SCORERANK) is the same, the actual values for the coefficients are different. How can this be possible? This comes from the fact that, when getting a ranking, what counts is not the actual value of the coefficients, but their ratio: *i.e.*, different sets of coefficients will give the same ranking as long as they are in the same ratio. The ratio between the first coefficient selected (LigHPhobB from Table 6-5) and the second (4/8ExtVdW) is 0.77 in all three runs. This means that the three solutions are perfectly equivalent, and they have just been scaled. In fact, if they are treated as vectors in the solutions space, and normalised, they correspond to the same values.

$$Norm_Sol = 100 * \frac{Sol}{\|Sol\|} \tag{6.6}$$

In the equation above, Sol is the original solution, $\|Sol\|$ is its length, so that the division of the two will return a unit vector parallel to the original solution. After normalisation the coefficients are multiplied by 100, giving a percent-like contribution. That is different from a proper percent contribution, since the different components are relative to different axes[JW48], so that as a general rule their sum totals more than 100%. The SCORERANK entry is also multiplied by 100, so that is now the proper measure of Percentage Average Ranking, *i.e.* the average location of the active in the ranked list. The equivalent normalised solution is shown in Table 6-6 and graphically in Figure 6-6.

2_terms/14-19/(A-B-C)/qtga.out (normalised)				
Term	LigHPhobB	4/8ExtVdW	SCORERANK	1st/2ndRATIO
Coefficient	61.11	79.15	8.68%	0.77

Table 6-6: All three solutions in Table 6-5 correspond to this scaled solution.

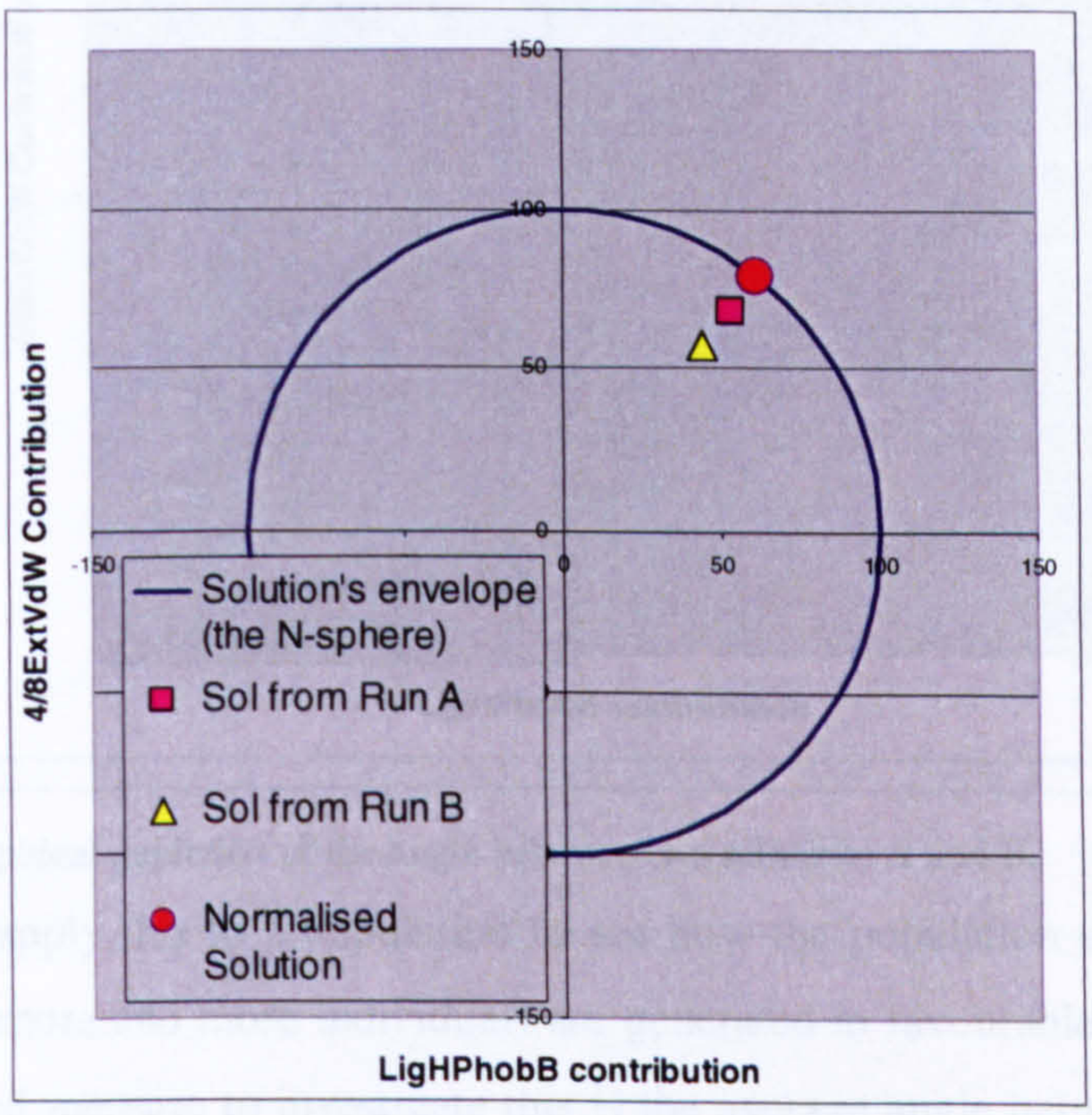


Figure 6-6: Graphs showing the solutions from Run A (purple square) and B (yellow triangle) projected within the 2-sphere (circle) of all possible solutions. Their normalised solution (red circle) sits on the blue circle line.

6.3.1 Angular distance between solutions

Since now all solutions can be thought of as different orientations on the surface of an N-sphere, where N is the number of terms, a good measure of expressing how different they are is taking the angle between them. That is most easily done by taking the dot product of the two normalised vectors.

$$\vartheta = \arccos\left(sol_1 \bullet sol_2\right) \tag{6.7}$$

The geographical meaning of ϑ is shown in Figure 6-7. In contrast to a *euclidean* distance measurement, this is not affected by the actual size of the vectors: the angle does not vary when normalising the solutions.

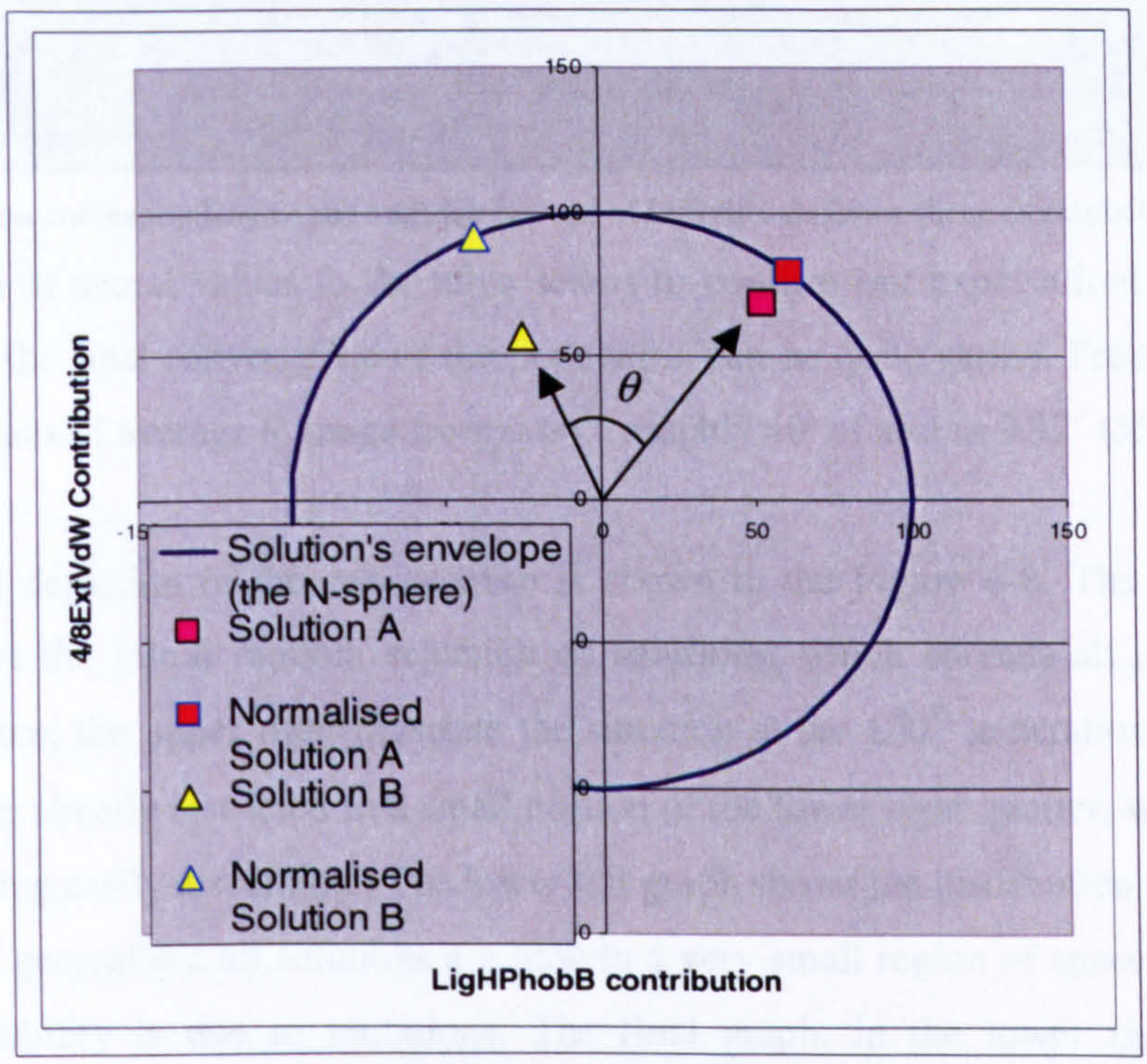


Figure 6-7: Graphical depiction of the angle between two solutions A and B.

We can then apply this to a population to see how the population diversity decreases over time, as more and more individuals are generated in favourable regions of the N-sphere. A good measure to investigate this is the average angle between solutions in a population. At the beginning of the GA calculation, angles between individuals/solutions will vary between 0° and 180°, as they will be spread all around the (N-)sphere's surface. We can at this time expect an average value of 90°. As soon as the optimisation begins, though, the solutions will start to converge. Not only the average, but also the maximum value observed will diminish, since exploration of and

permanence in regions far from already found minima will be discouraged. An example of this behaviour is shown in Table 6-7.

Run	Generations	Average θ	Maximum θ
Run A	*Start (1st gen)*	90.54°	177.80°
	Half (300th gen)	3.43°	9.42°
	End (600th gen)	0.90°	2.10°
Run B	*Start (1st gen)*	90.43°	178.63°
	Half (300th gen)	2.69°	9.81°
	End (600th gen)	0.66°	1.66°
Run C	*Start (1st gen)*	89.69°	179.65°
	Half (300th gen)	3.10°	9.88°
	End (600th gen)	0.92°	3.37°

Table 6-7: Data corresponding to the angular spread of individuals from three example GA runs[30].

Observation of actual values in the table seems to confirm our expectations, although we see that the final convergence of the population can be quite varied. From the table the final values of average θ, range from 0.66° (roughly 40' of arc) to 0.92° (55').

A graphical depiction of the convergence is shown in the Figure 6-8. The upper left graph shows the initial random selection of solutions, which spreads all around the solution space; the upper right illustrate the situation at the 150th generation, when all solutions are already restricted to a small portion of the lower right quarter, with a four-way clustering easily discernible. The lower left graph shows the distribution of solution at the 300th generation: all solutions are now in a very small region of space, and most of the variability is due to mutations. The final graph, in the lower right corner, illustrates the final solution. Very few independent and high-performing solutions have survived, concentrated in a tiny region of the solution space – less than 100 points are seen here, due to the fact that most individuals do overlap perfectly – they are in fact each others' clones.

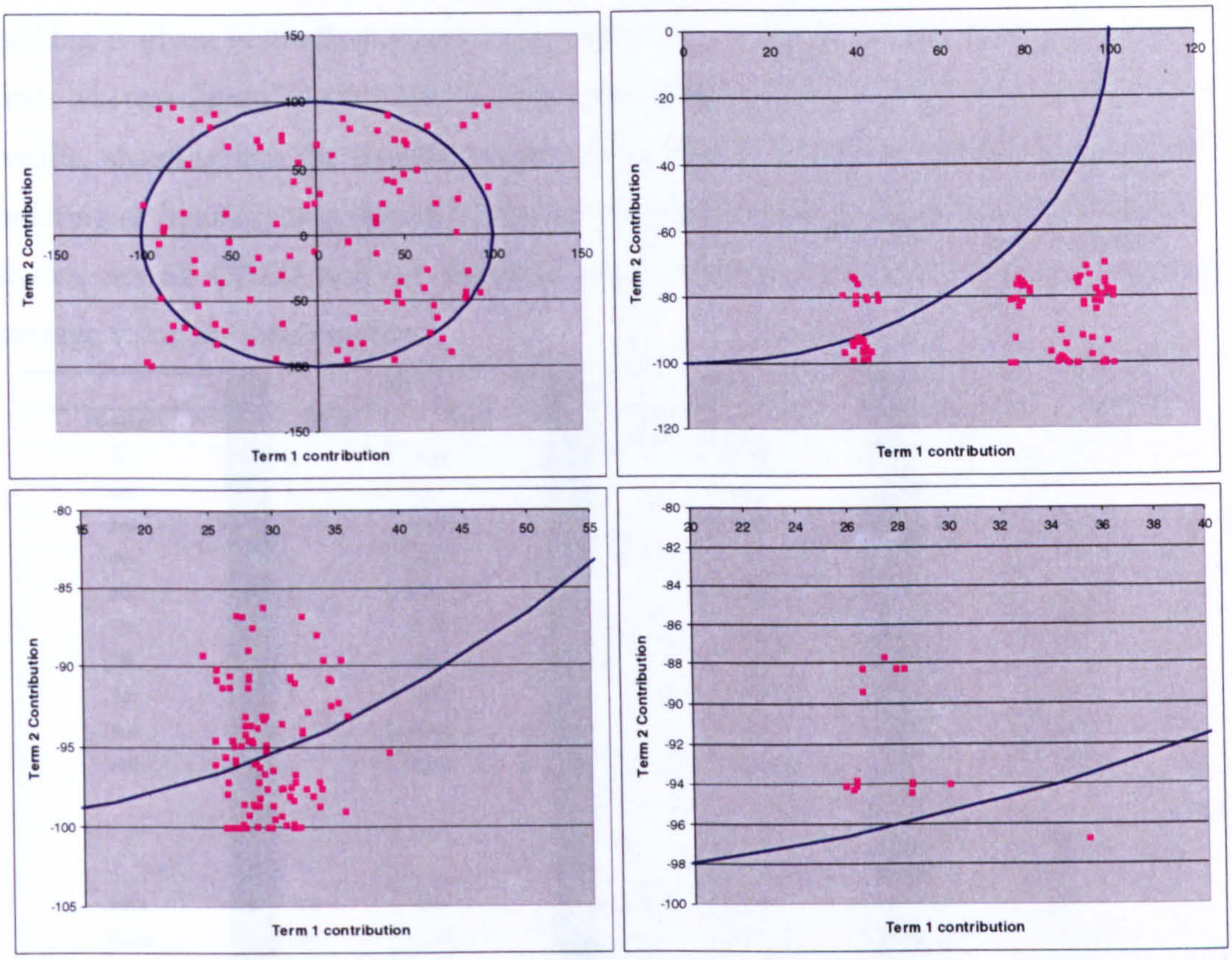


Figure 6-8: The graphs shown illustrate the convergence of a GA towards its final solution. The population's distribution is shown at the 0th (upper left), 150th (upper right), 300th (lower left) and 600th generation (lower right).

Of course, the angular distance can be used to estimate the difference between final solutions belonging to different runs, such as A, B, and C in Table 6-8. When one computes those angles, it is clear that the difference between the best solution of run A and that of run B is negligible (0.02°, or 1' of arc), whereas that between the B and C best individuals is of barely half a degree, or 30' of arc.

	GOLDHBOND	GOLDEXTVDW	GOLDITORS
Run A	-99.96	-2.21	-1.76
Run B	-99.96	-2.23	-1.73
Run C	-99.94	-3.30	-1.28

Table 6-8: Best solution (normalised) from the three GA runs whose average angle variation along time is shown in Table 6-7; the final solutions are very, very close.

Let us now return to our solutions. Since the solutions are equivalent, they also share the same ranking for the COLOs. Table 6-9 reads as follows: for every PDB entry, the number to its left shows the ranking position of the COLO after the optimisation. The

ranking is given in absolute terms, not a percentage – that is, 1azm's true ligand ranks first, whereas 2qvm's ranks 344th out of 1001 ligands. All three runs present the same results, showing that the absolute value of the terms' coefficients does not affect the ordering of ligands; only the ratio between the coefficients is significant. As Table 6-9 shows, not all COLO will be placed at 8.7% of their ranking list – that is just the average value for their rankings.

	Run A		Run B		Run C	
Position	Rank	PDB	Rank	PDB	Rank	PDB
1st	1	1azm	1	1azm	1	1azm
2nd	1	1cil	1	1cil	1	1cil
3rd	1	1dr1	1	1dr1	1	1dr1
4th	1	1eoc	1	1eoc	1	1eoc
5th	1	1lah	1	1lah	1	1lah
6th	1	1ldm	1	1ldm	1	1ldm
7th	1	1lst	1	1lst	1	1lst
8th	1	1nis	1	1nis	1	1nis
9th	1	1pbd	1	1pbd	1	1pbd
10th	1	1qpq	1	1qpq	1	1qpq
...	...	...	...	...	...	...
99th	308	1fen	308	1fen	308	1fen
100th	344	2qwk	344	2qwk	344	2qwk
101st	346	1epb	346	1epb	346	1epb
102nd	400	1cbx	400	1cbx	400	1cbx
103rd	411	1pso	411	1pso	411	1pso
104th	429	1apt	429	1apt	429	1apt
105th	442	4dfr	442	4dfr	442	4dfr
106th	484	1tni	484	1tni	484	1tni
107th	486	1bma	486	1bma	486	1bma
108th	543	1lic	543	1lic	543	1lic

Table 6-9: First and last ten rankings for the LigHPhobB + 4/8ExtVdW combination.

In Table 6-10, every COLO's rank within its own list is given. As can be seen, almost half of all COLOs are retrieved within the very first ten positions, corresponding to 1% ranking or better. More than 70% are in fact within the first 10% of their own ranking list, and the vast majority of the contribution to that final value of 8.7% comes from the worst performing ten compounds which sit in positions lower than halfway down their own list. These protein/ligand complexes' interactions are not well explained by our model, or at least can not be described at the same time as the other high rankings. Because this combination of terms and coefficients is the best one on all proteins overall, the only hope to bring their ranks up is to add new terms: and by doing so improve the description by complementing those already present.

Rank	PDB	Rank	PDB	Rank	PDB	Rank	PDB	Rank	PDB	Rank	PDB	Rank	PDB
1	1azm	3	1glq	3	1glq	6	2cmd	25	2aad	68	2ack	173	1coy
1	1cil	3	1hsl	3	1hsl	7	1tyl	27	1mup	69	1qpe	173	1ydt
1	1dr1	3	1kel	3	1kel	8	1dhf	27	4aah	78	1a4q	177	1ckp
1	1eoc	3	1lcp	3	1lcp	10	1ets	29	25c8	81	1c1e	180	1hsb
1	1lah	3	1mld	3	1mld	11	4lbd	32	1ydr	81	1tnl	232	1cvu
1	1ldm	3	1wap	3	1wap	12	1phg	34	1d4p	87	1ida	240	2pk4
1	1lst	3	1yee	3	1yee	14	1d3h	36	3tpi	88	1f0r	292	1ei1
1	1nis	4	1b58	4	1b58	16	1bbp	38	1cps	93	1hiv	292	1hdc
1	1pbd	4	1rnt	4	1rnt	17	1fkg	40	1ptv	94	1hyt	308	1fen
1	1qpq	4	2yhx	4	2yhx	20	1aoe	42	1a6w	97	1dg5	344	2qwk
1	1xid	5	1mts	5	1mts	20	2fox	42	1nco	97	1hos	346	1epb
1	2ada	6	1c5x	6	1c5x	21	1a28	46	1byg	113	3ert	400	1cbx
1	6rnt	6	1c83	6	1c83	21	1tng	49	1qcf	125	1fl3	411	1pso
2	1ai5	6	1d0l	6	1d0l	21	2ctc	50	3erd	126	1ngp	429	1apt
2	1byb	6	1dd7	6	1dd7	22	1f0s	51	1f3d	126	1poc	442	4dfr
2	1ivb	6	1ejn	6	1ejn	23	1b9v	52	1atl	134	3cpa	484	1tni
2	1snc	6	1flr	6	1flr	24	1mrg	57	7tim	139	1mmq	486	1bma

Table 6-10 Full COLO's rankings for the combination of terms LigHPhobB - 4/8ExtVdW.

6.3.2 Good performers and bad performers

It is now clear that, as long as the ratio between the coefficients is respected, the ranking power of the solution is maintained. This holds true whatever the number of terms included in our scoring function.

COMBINATION	Best Rank	Ave Rank
4_terms/18-19-26-28/B	3.83%	3.88%
4_terms/18-19-26-28/C	3.83%	3.91%
4_terms/18-19-26-28/A	3.83%	3.94%
4_terms/18-19-25-26/C	4.16%	4.44%
4_terms/18-19-25-26/A	4.17%	4.27%
4_terms/18-19-25-26/B	4.17%	4.27%
4_terms/18-19-25-28/A	4.20%	4.25%
4_terms/18-19-25-28/B	4.20%	4.25%
4_terms/18-19-25-28/C	4.21%	4.27%
4_terms/14-19-26-28/C	4.40%	4.47%

Table 6-11 Best performers between the four-term combinations: refer to Table 6-1 for translation between numerical IDs and terms' names.

The more terms, the more rewarding in terms of ranking power the final solution is likely to be, but the larger the solution space, the more difficult the optimisation task becomes. For example, for the four-term combinations, the best ten performers are

shown in Table 6-11. The three independent runs of our best performing four-term combination (Table 6-12) now have slightly different performances, a sign that the three GA runs have converged to three slightly different (and sub-optimal) solutions.

4_terms/18-19-26-28/A				
LigHPhobS	4/8ExtVdW	GOLDHBOND	GOLDITORS	*SCORERANK*
8.53	14.71	-98.01	20.60	***3.8328%***
4_terms/18-19-26-28/B				
LigHPhobS	4/8ExtVdW	GOLDHBOND	GOLDITORS	*SCORERANK*
7.95	13.14	-97.23	21.98	***3.8290%***
4_terms/18-19-26-28/C				
LigHPhobS	4/8ExtVdW	GOLDHBOND	GOLDITORS	*SCORERANK*
9.49	15.15	-100.00	24.12	***3.8306%***

Table 6-12: Solution for the three runs of the best four-term combination selected by our protocol. Ranking performances are near-identical.

However, the differences in performance are minimal: the ranking powers differ only at the third decimal point of a percentage – that is less than a one in ten-thousand difference, and the ratio between the terms' coefficients is still quite similar. Run's A solution stands 1°23' from B, and 1°42' from C. The three solutions are therefore very close to each other and most likely very close to the optimum value.

When we explore in detail the rankings of the single COLOs (Table 6-13), we see something similar: the three different solutions manage to bring up in first position slightly different sets of PDB entries. 1bpp and 1c83, for example, characterise run A. 4aah's COLO ranks first in run B (and run A too, not shown since it is below the cutoff). Run C only has 9 COLOs at the first place of their respective ranking list, whereas run A has 14 COLOs in first position (not all of them are shown), and run B has only the ten shown in the table.

Analysis of the bottom lines is also useful: even here, the ordering is quite similiar. However, COLOs at this end change ranks to a greater extent: for example, 1tni is at 263/4 in runs B and C, but is 315 in run A. The same is true for 2pk4's COLO, 168-9 for runs A and B but up to 135 in run C. However, most COLOs stay in the same region, as the solutions are so similar. At the bottom, however, the variability due to small

differences in the coefficient's ratio is larger. As Tolstoy once argued, there may be one way of getting things right, but there are millions of ways they can go wrong [222].

	Run A		Run B		Run C	
Position	Rank	PDB	Rank	PDB	Rank	PDB
1st	1	1bbp	1	1eoc	1	1eoc
2nd	1	1c83	1	1lah	1	1lah
3rd	1	1eoc	1	1ldm	1	1ldm
4th	1	1lah	1	1lst	1	1lst
5th	1	1ldm	1	1mld	1	1mld
6th	1	1lst	1	1nis	1	1nis
7th	1	1mdr	1	1pbd	1	1pbd
8th	1	1mld	1	2cmd	1	2cmd
9th	1	1nis	1	2ctc	1	2ctc
10th	1	1pbd	1	4aah	2	1azm
...	...	...	...	...	...	...
99th	96	1hsb	93	1f0r	115	1mmq
100th	107	1hos	95	1mmq	124	1pso
101st	107	1mmq	123	1hos	128	1slt
102nd	137	1c1e	143	1slt	135	2pk4
103rd	150	1slt	148	1c1e	136	1c1e
104th	167	1coy	162	1coy	143	1hos
105th	168	2pk4	169	2pk4	159	1coy
106th	188	1ckp	179	1ckp	181	1ckp
107th	315	1tni	263	1tni	264	1tni
108th	642	1lic	648	1lic	649	1lic

Table 6-13: First and last ten rankings for the LigHPhobS + 4/8ExtVdW + GOLDHBOND + GOLDITORS combination.

Things are not always so clean cut, however. Until now, only the best performers have been taken into consideration. They are combinations whose terms interact well with each other, forming a smooth and funnel-shaped surface where the GA can easily find the minimum. What happens if we consider a *second-choice* scoring function, whose terms do not interplay as well with each other? Let us take, as an example, the combination shown in Table 6-14: they are made up of 8Å Coul, LigHPhobS, 4/8ExtVdW, GOLDHBOND[31]: its three runs come in the list of the 504 four-term combinations tested at positions 91, 93, 105, according to their ranking power. They are definitely worse than the best performing combinations, but not too bad, with ranking powers of around 5.4%, rather than 3.8% - roughly, a 25% worse performance compared to our best choices.

COMBINATION	Best Rank	Ave Rank
4_terms/10-18-19-26/A	5.39%	5.46%
4_terms/10-18-19-26/B	5.42%	5.46%
4_terms/10-18-19-26/C	5.45%	5.55%

Table 6-14: a mediocre performer in the four-term combinations.

In this case, too, the performances between the three runs are the same, although slightly wider in range than in the case of the best performers. What is striking, though, is the diversity of the coefficient's values related to such similar results (see Table 6-15). The first two terms to be chosen, LigHPhobS and 4/8ExtVdW, keep themselves more or less at the same ratio seen before (just less than 0.5), whereas the third one to appear, GOLDHBOND, varies a little more. What is more surprising, though, is the huge variation of the "8A Coul" term, which goes from negative to positive without, in fact, influencing very much the ranking.

4_terms/10-18-19-26/A				
8A Coul	LigHPhobS	4/8ExtVdW	GOLDHBOND	*SCORERANK*
100.00	6.13	13.18	-91.27	***5.39%***
4_terms/10-18-19-26/B				
8A Coul	LigHPhobS	4/8ExtVdW	GOLDHBOND	*SCORERANK*
36.79	8.34	16.81	-99.21	***5.42%***
4_terms/10-18-19-26/C				
8A Coul	LigHPhobS	4/8ExtVdW	GOLDHBOND	*SCORERANK*
-17.56	7.63	15.27	-90.58	***5.45%***

Table 6-15: Best solutions for each run presented in Table 6-14.

This does essentially mean that adding this term does not give any more information to the function on how to better rank the COLOs: its relative importance must be very low. In fact, one would be better off without it, as we save the computational cost.

As a general rule, good performers will not have a poorly performing component, since they will have been selected exactly to avoid that. If, given a specific combination of terms, every new one added does nothing to improve the ranking power, it may either mean that our function already optimally describes the protein/ligand interactions, or equally likely that more diverse terms are needed to improve our description.

6.3.3 Are bad functions really that bad?

Before moving to consider other problems, let us look at one of the worst performers, in fact the worst four-term combination (Table 6-16):

COMBINATION	Best Rank	Ave Rank
4_terms/1-12-16-19/C	9.08%	10.08%
4_terms/1-12-16-19/B	10.02%	10.33%
4_terms/1-12-16-19/A	10.64%	11.20%

Table 6-16: The worst performer amongst the four-term combinations.

This is the combination of GoodHBond, CL8AnGBSA, TotHPhobS, 4/8ExtVdW[32]. Even after GA-optimisation, it still ranks the COLOs barely below 10% of the ranked list, a result worse than that posted by the best two-term scoring function (not shown here). It is not, by any means, an awful combination, as it is still able to rank the COLOs on average in the first 10% of the list. It is, in fact, able to rank around twenty (from 19 in run A to 23 in run B) COLOs in first position, as compared to only 9 to 14 for the best performer (Table 6-12). Why, then is the overall performance so (relatively) bad? The answer must lie in the distribution of all rankings, not only the particularly good ones, which represent the head of the distribution. Since presenting all 108 proteins in text form would just be confusing, we will graph their distribution.

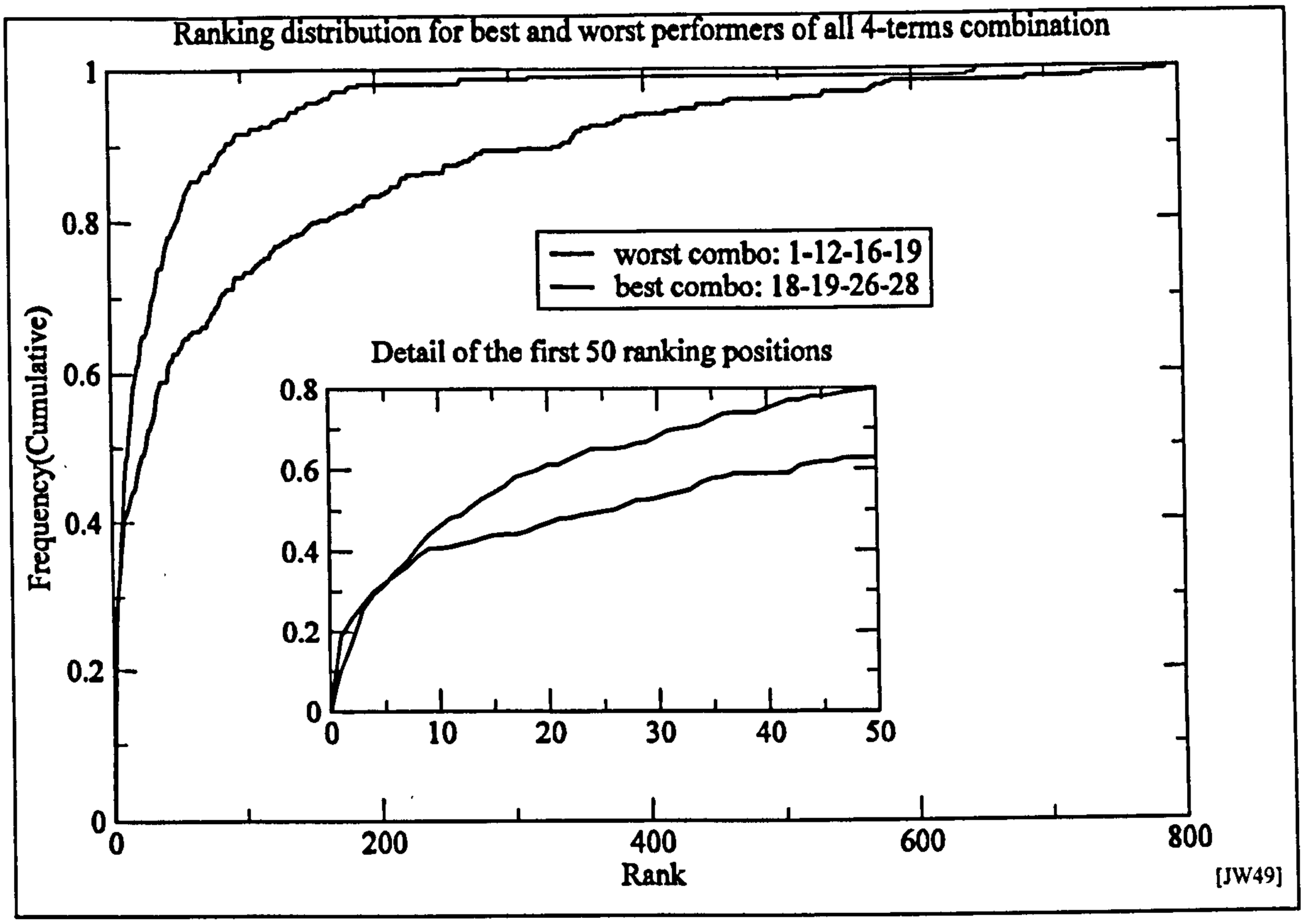


Figure 6-9: Plot showing a measure of 'enrichment', namely how many COLOs have been found as a function of the number of decoys selected out of a ranked list.

In Figure 6-9 we represent the (cumulative) distribution of rankings for the two cases under consideration, the best (Table 6-12) and worst (Table 6-16) of all four-term combinations. This graph is absolutely equivalent to an Enrichment Factor plot, with the only difference that here actives are the COLOs corresponding to various proteins, not only one. The plot is normalised, so that the value of the blue line at rank position 200 means that 98% of COLOs have been found by then. The detail box shows how the two combinations of terms manage in the very first part of the graph. At the very beginning the worst (black line) is at a definite advantage: the best (blue line) ranks first in their lists only 10% of COLOs, versus 20% ranked by the worst. The worst combination is actually better until we hit rank 5. After that, best's performance gets much better, and the blue line does reach 100% before the black line. The best is better in the long run, or if you want, is a better compromise than the worst, which is very good on some proteins, but much worse on the rest. Please note that both functions are anyway (much) better than a random classifier, whose graph would be represented by a straight line from the origin to the top-right hand corner.

As shown in Figure 6-9, although the worst combination is very good in ranking (and therefore describing the interactions of) twenty-odd COLOs, it fails to prioritise the rest, therefore obtaining a lower result in the end. This does not, however, means that the function is useless. It may, in fact, indicate that this selection of terms is particularly indicated when working with these proteins. Therefore, we might suppose that the combination of GoodHBond, CL8AnGBSA, TotHPhobS, and 4/8ExtVdW might be a good starting point to create a scoring function tailored on a particular class of proteins[JW50].

6.3.3.1 Checking classes and ligands

So, do these twenty-odd complexes, whose interactions our 'worst' scoring function describes particularly well, share something peculiar? Do they, for example, belong to a particular protein family? Their PDB entries are known to us, so it is possible to check. Results are, to say the least, puzzling. Only six belong to classes for which we have parameterised 'tailor-made' scoring functions (see section 10.3 for these results), but with the exception of two immunoglobulins, they all belong to different families: 1bbp is a lipocalin, 1dr1 belongs to the dihydrofolate superfamily (from SCOP), 1tmn is a protease, and finally 1yee and 1kel are the immunoglobulins. The remaining entries are interspersed between poorly represented (in our set) super-families.

1bbp	**Lipocalins**	**1tmn**	**Protease**
1dr1	**Dihydrofolate**	**1yee**	**Immunoglobulin**
1kel	**Immunoglobulin**	**4lbd**	**Nuclear Receptor**
1azm	Carbonic anhydrase	1ldm	LDH C-terminal domain-like
1c83	(Phosphotyrosine protein) phosphatases II	1lst	Periplasmic binding protein-like II
1cil	Carbonic anhydrase	1nis	Aconitase iron-sulfur domain
1d0l	Lysozyme-like	1pbd	FAD/NAD(P)-binding domain
1dd7	Nitric oxide (NO) synthase oxygenase domain	1ptv	(Phosphotyrosine protein) phosphatases II
1fki	FKBP-like	1qpq	Quinolinic acid phosphoribosyltransferase
1glq	Glutathione S-transferase (GST)	1xid	Xylose isomerase-like
1lah	Periplasmic binding protein-like II	2ada	Metallo-dependent hydrolases

Table 6-17: Proteins for which the scoring function detailed in Table 6-16 ranks the COLOs first: those in bold and above the horizontal line are parts of subsets for which we have parameterised single-family scoring functions; the rest are not.

As can be seen from Table 6-17 above, there is no obvious prominence of a family (although there may be at the level of the fold, which we did not consider). If the reason is not in the proteins, might it be in the ligands (Figure 6-10)?

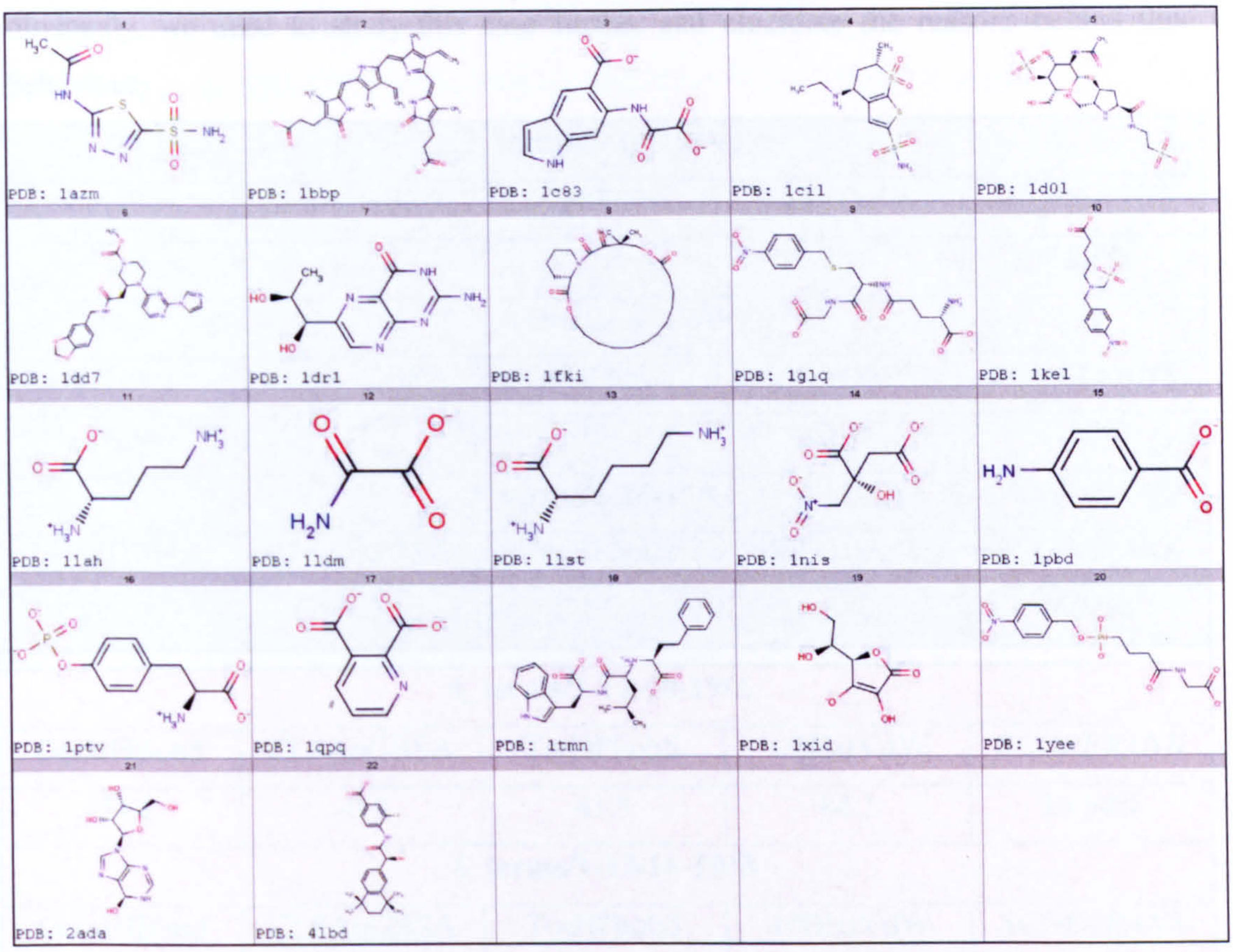


Figure 6-10: Reference ligands of the 22 entries ranked first by the scoring function detailed in Table 6-16.

They are all ligands with polar (often charged) groups in them, so one may expect the CL8AnGBSA term to play an important role. This term is the sum of the coulombic interaction, with an 8 Å cut-off, and the GBSA solvation method contribution. These two terms are strongly anti-correlated and their sum should express the desolvation contribution to the free energy of binding. This seems a very neat idea; polar ligands require a lot of energy to desolvate, so a solvation term should be included to accurately describe their interactions. Unfortunately, the idea is contradicted by the results observed. The parent of this combination, 1-16-19 (GoodHBond - TotHPhobS - 4/8 ExtVdW), lacking a term for solvation, or even a simple coulombic interaction, has a better ranking power than the child function (see Table 6-18). 1-16-19 ranks the COLOs on average at around 7% of the ranked list, rather than 10% or so for 1-12-16-19. In fact, the parent combination is quite good, occupying places 25-27(out of 720) of the 3 term list ordered by ranking power. All three runs did enter the list of thirty three-term GA runs selected for improvement by the addition of one more term, but, by adding the solvation term, we have in fact worsened the overall performance. Quite

obviously, we need to study this case further and elucidate the reasons behind this behaviour.

3_terms/1-16-19/A				
GoodHBond	CL8AnGBSA	TotHPhobS	4/8ExtVdW	*SCORERANK*
-99.9	0.0	2.1	4.1	***7.04%***
3_terms/1-16-19/B				
GoodHBond	CL8AnGBSA	TotHPhobS	4/8ExtVdW	*SCORERANK*
-99.9	0.0	1.9	3.7	***7.02%***
3_terms/1-16-19/C				
GoodHBond	CL8AnGBSA	TotHPhobS	4/8ExtVdW	*SCORERANK*
-99.9	0.0	1.9	3.7	***7.02%***
4_terms/1-12-16-19/A				
GoodHBond	CL8AnGBSA	TotHPhobS	4/8ExtVdW	*SCORERANK*
-34.8	-0.9	41.5	84.1	***10.64%***
4_terms/1-12-16-19/B				
GoodHBond	CL8AnGBSA	TotHPhobS	4/8ExtVdW	*SCORERANK*
-89.0	-0.4	19.6	41.2	***10.02%***
4_terms/1-12-16-19/C				
GoodHBond	CL8AnGBSA	TotHPhobS	4/8ExtVdW	*SCORERANK*
-97.9	-0.2	9.0	18.6	***9.08%***

Table 6-18: Results for combination 1-16-19 (grey background) and its *daughter* combination 1-12-16-19 (pink background). The *mother*'s performances (*scorerank*) are better.

What we have just shown is, however, not uncommon. After all, our procedure exploits exactly this sort of drop-out to select the best scoring combinations. But, one would expect any new term would add a little more ranking power to a given combination, and then we would select the term which has added the most. Instead, some terms seem to actually have a detrimental effect, even at their best tuning.

Can we envisage a reason why this decrease in performance could happen? One tentative reason may be that some of our terms are correlated, so that their addition damages the equilibrium attained by the terms already presents. However, this explanation does not hold. If every combination of the four terms were to give a worse result than a three-term combination contained within, then our GA should simply select

a 'zero' value for the fourth coefficient added (CL8AnGBSA in this case). We should, in fact, revert to the old, three terms only, combination. Why is this not happening? Is the GA performing its task as it should?

We had at our hand two series of data: first, the *mother* three-term combination; second, the *daughter* four-term combination (again, see Table 6-18). Considering each separately, we kept the coefficients for the three terms already determined constant, and varied the newly introduced term's coefficient (CL8AnGBSA). In this way, two one-dimensional sections (slices) of the 4-dimensional solution space were selected: one for the mother, one for the daughter solution. As expected, if we start from the coefficients of the parent three-term combination, the fourth term's best contribution is nil (see Figure 6-11, black line). The minimum of the black curve is right on zero. This curve corresponds to the solution highlighted by a grey background in Table 6-18. The red curve represents instead solutions generated starting from the coefficients shown in the pink-background portion of the same table.

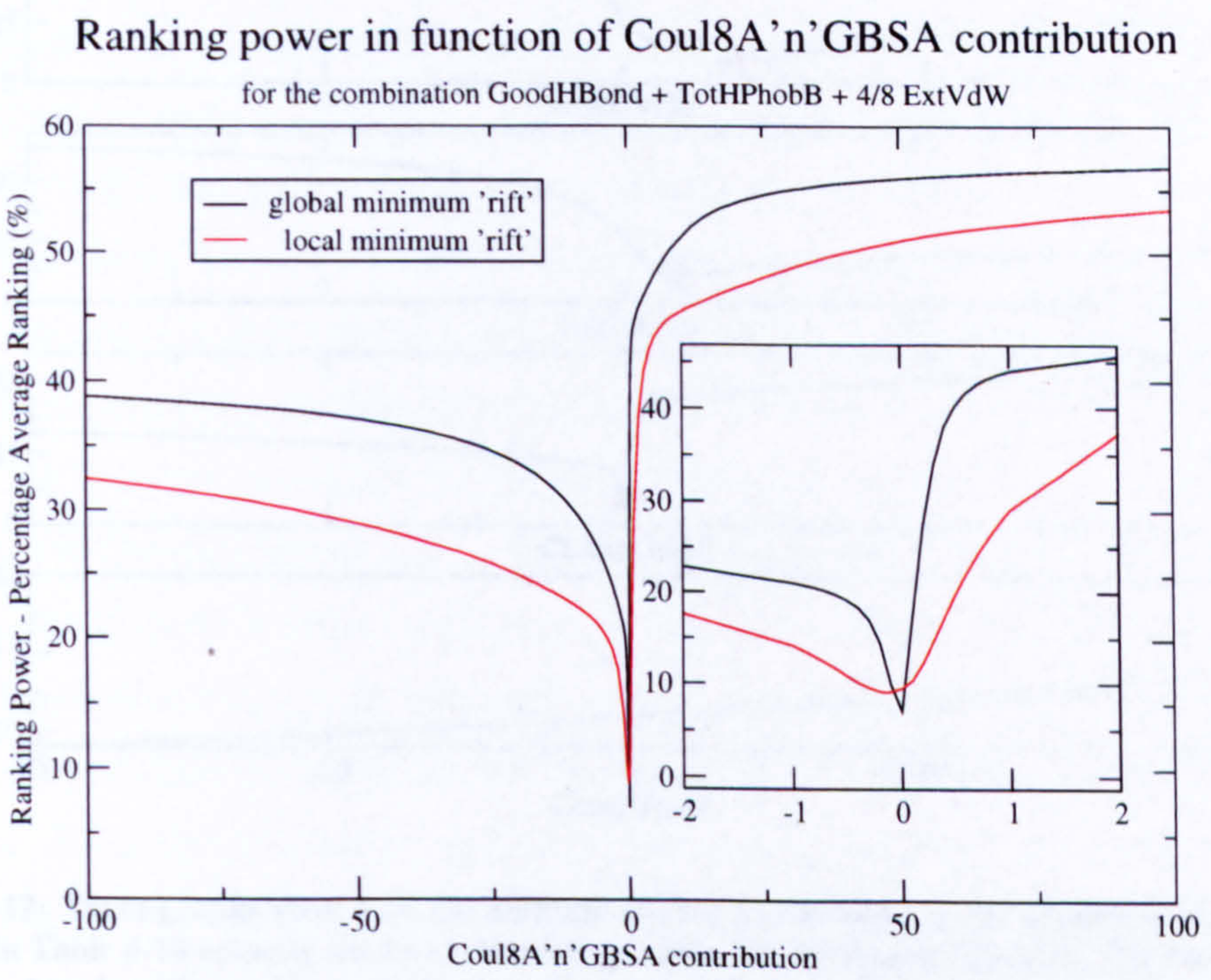


Figure 6-11: The ranking power variation with the varying contribution of Coul8A'n'GBSA. The small square represents, in greater detail, the region close to the zero contribution.

If the scanning along the Coul8Å'n'GBSA axis is performed starting from the said solutions, the minimum lies exactly where the GA has converged in the four-term scenario. As shown in Figure 6-11, the red line has a minimum at a negative value of the Coul8Å'n'GBSA conribution, close but not equal to zero.

That this actually is a minimum, and not a saddle point, is shown in Figure 6-12, where variation along all axes is shown. The black cross represents the optimum value, whereas the red 'X' is the converged value of the GA. They overlap almost exactly in all four directions. It can be noted that the GoodHBond contribution is monotonic so that a value even lower than -100 would be welcome: but that is the limit we artificially imposed to our contributions. As we already explained, what is important is the ratio between them, and not their absolute value.

Still, it is obvious from the graph in Figure 6-11 that it only is a local minimum, and that the global optimum is in fact on the black line. Why, then, the GA did not select the same three coefficients chosen in the previous run, and just find the better minimum along the new degree of freedom?

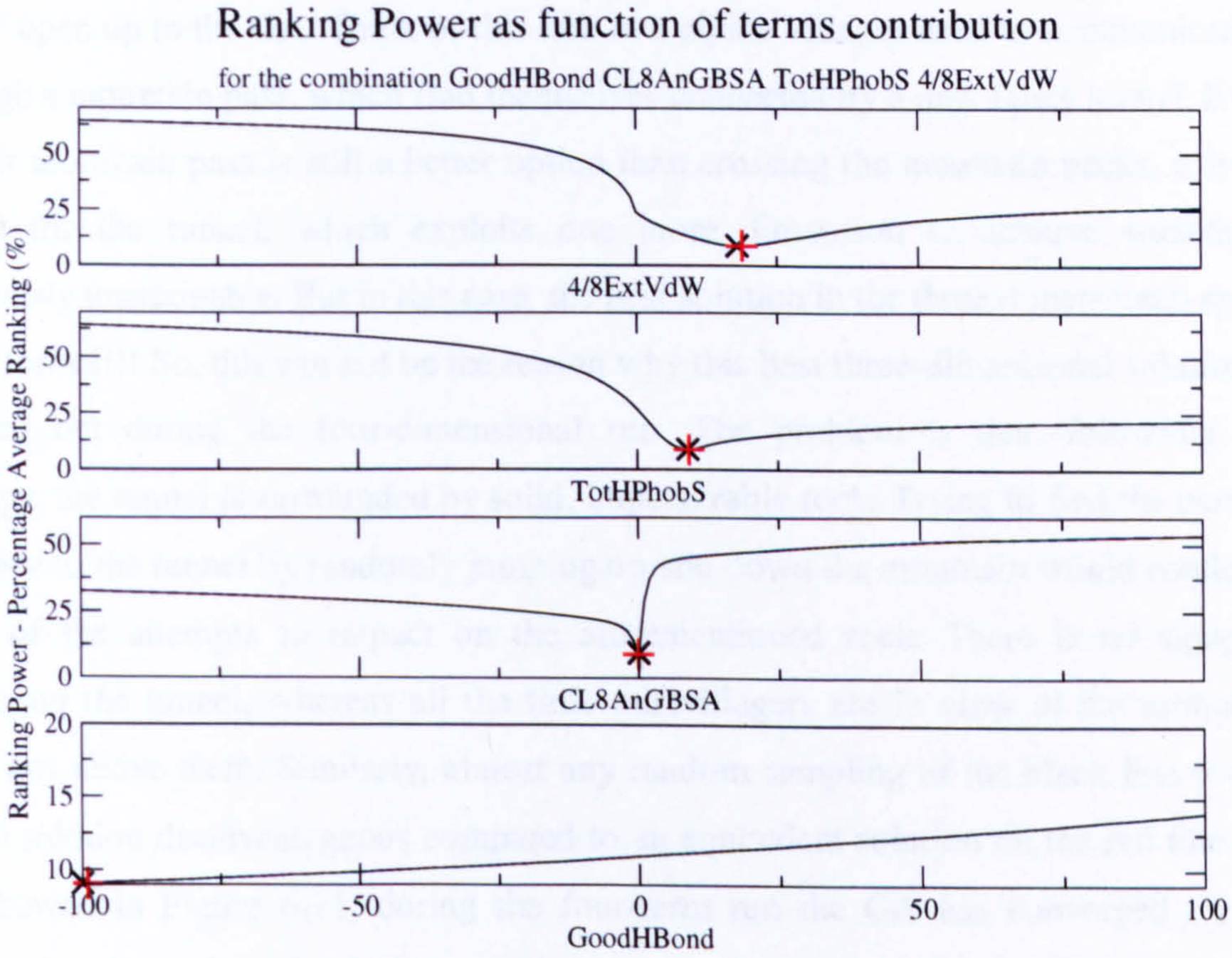


Figure 6-12: These graphs show how the solution chosen by the GA for the combination of terms detailed in Table 6-18 actually sits in a minimum position for the fitness function. The curves show the fitness trends with varying contributions along the given axis. The black crosses in the various graphs correspond to the computed optimum values, the red crosshairs are instead the values selected by the GA. All are almost overlapping.

6.3.4 How Genetic Algorithms really work

I must here remind the reader that this is not how the GA works. Every time a new term is introduced, we do not perform a minimisation with the old values constrained. A

completely new GA run is started, and the coefficients' values previously found are not retained. The new population is generated randomly, so there is no room for 'memory effects'. Our GA cannot remember good solution from run to run, and include derivatives of them in the new sampling - that would be some sort of Lamarckian GA, where children can exploit adaptations won by their parents during their life. In a Darwinian GA like ours, the optimisation is performed from scratch in the new environment, that is, the newly enlarged space.

Even if the GA could include such 'advantageous' solutions, it would not do them any good because, as I am now going to show, they would be, most likely, weeded out. The fundamental thing to keep in mind here is that every time the dimensionality of the space is enlarged, the 'fitness' landscape changes drastically, and a whole lot of new 'paths' open up to the GA. Think of this like two alpine villages used to communicating through a mountain pass, which find themselves connected by a new fancy tunnel. Even if their mountain pass is still a better option than crossing the mountain peaks, it is no match for the tunnel, which exploits one more dimension to achieve something previously unattainable. But in this case, the best solution in the three dimensional space is the tunnel!!! So, this can not be the reason why this best three-dimensional solution is weeded out during the four-dimensional run. The problem is that, following the analogy, the tunnel is surrounded by solid, impenetrable rock. Trying to find the narrow entrance to the tunnel by randomly jumping up and down the mountain would condemn most of the attempts to impact on the aforementioned rock. There is no signpost indicating the tunnel, whereas all the time our villagers are in view of the mountain pass, just above them. Similarly, almost any random sampling of the black line would find a solution disadvantageous compared to an equivalent solution on the red line. As we showed in Figure 6-11, during the four-term run the GA has converged on the minimum of a series of solutions good on average (red line), but not containing the global optimum. The global optimum is, as we now know, identical for both three- and four-term run, and sits on the black line. However, because of the change in the fitness landscape when passing from 3 to 4 dimensions, this best solution is part of a solution set (black line) which stays mostly above the red line, and only dives below it in a very narrow field of values around the minimum itself. At the start of our run, the GA randomly selects individuals all around the solution space. Then it applies a roulette selection to improve on the best ones. The GA is trying to minimise the fitness. So,

lower fitness gives to a solution an advantage over the others. As most of the red curve is below the black one, in almost every possible confrontation between two randomly selected solutions belonging to the two families, the red ones are likely to win over the blacks, and will get a larger slice of the roulette. A larger share means more likelihood of bearing children, either clones (identical) or mutated (very close). The roulette selection causes the red solutions to quickly dominate the population. The GA does not know, and does not care, that by choosing a momentarily bad performing solution on the black line it may later on arrive to the true optimum. It blindly chooses the best solutions at hand, which quickly out-compete all others, and keeps improving them. Once the black solutions have been weeded out, it is very unlikely that they will be re-generated, especially if the two solution space slices are farther apart than the standard deviation of the population along that axis.

6.3.4.1 Similarities between GA and evolution in Nature

A parallel can be made here with evolution in nature [221], upon which GAs are built. Suppose you have a tree-dwelling monkey, and you are trying to devise a set of limbs for it to run on plain ground. The optimum solution would be to go back to the drawing board, and devise wheels for the monkey to be able to run faster than any other animal on plain ground. Choosing a sub-optimal solution is instead equivalent to slowly adapting the lower limbs it already has. Those limbs, adapted to climb trees, are not exactly suited for running, so even at their best they will never be as good as a pair of wheels. However, for the monkeys to be able to grow wheels, they would have to undergo more dramatic changes, which would forbid, in their first stages even limited mobility. Sitting, limbless monkeys waiting to grow wheels would be strongly disadvantaged against the others, so they would become extinct quite rapidly. This is, in short, the reason why baboon, ostrich or even cheetahs do not have wheels. The jury is still out on the cart-wheeling spider, which, thanks to its different symmetry (it has eight limbs), has been able to 'become' a wheel without incurring severe disadvantages. Instead of running on the legs, it just folds them around the body like spikes around a hub. Rolling itself on the sand, it moves much quicker than by running[33]. Both in nature, and GAs, mutations are allowed to survive only if they do not cause an immediate disadvantage compared to other individuals present in the environment, such as the parents themselves. Mutations are of course actively encouraged by selection whenever they offer an immediate advantage compared to other alleles. In short, evolution goes

forward, and sometimes sidewise (genetic drift, the slow accumulation of silent mutations), but cannot backtrack. It is like climbing a mountain range: each individual picks its own route based on its immediate surrounding, and quickly finds itself at the top of a high enough mountain. It may be high enough, but not the tallest, though, since they cannot backtrack, even if they were to realize they have gone the 'wrong' way.

6.3.5 Checking the good combinations

Now, let us go back to our scoring function, and the problem of generally disadvantaged solution sets nonetheless containing the optimum solution. Is this a problem which manifests itself only when *bad* terms are added, or even for good ones? Are we really getting the best coefficient value for, say, the best 4 term combination selected (see subsection 6.3.2)? This can be easily checked: taking one step back, we see that the best 4 term combination was generated by the second best 3 term combination:

LigHphobS (6.00), 4/8ExtVdW (12.49), GOLDHBOND (-90.60)

To these three, GOLDITORS was added, going from a percentage average ranking of 5.4% for the COLOs to 3.8%. Of course, since the optimisation was performed from scratch, the coefficients for all terms were now:

LigHphobS (7.95), 4/8ExtVdW (13.14), GOLDHBOND (-97.23), GOLDITORS (21.98)

However, the ratio between them stays similar, as shown in Figure 6-13.

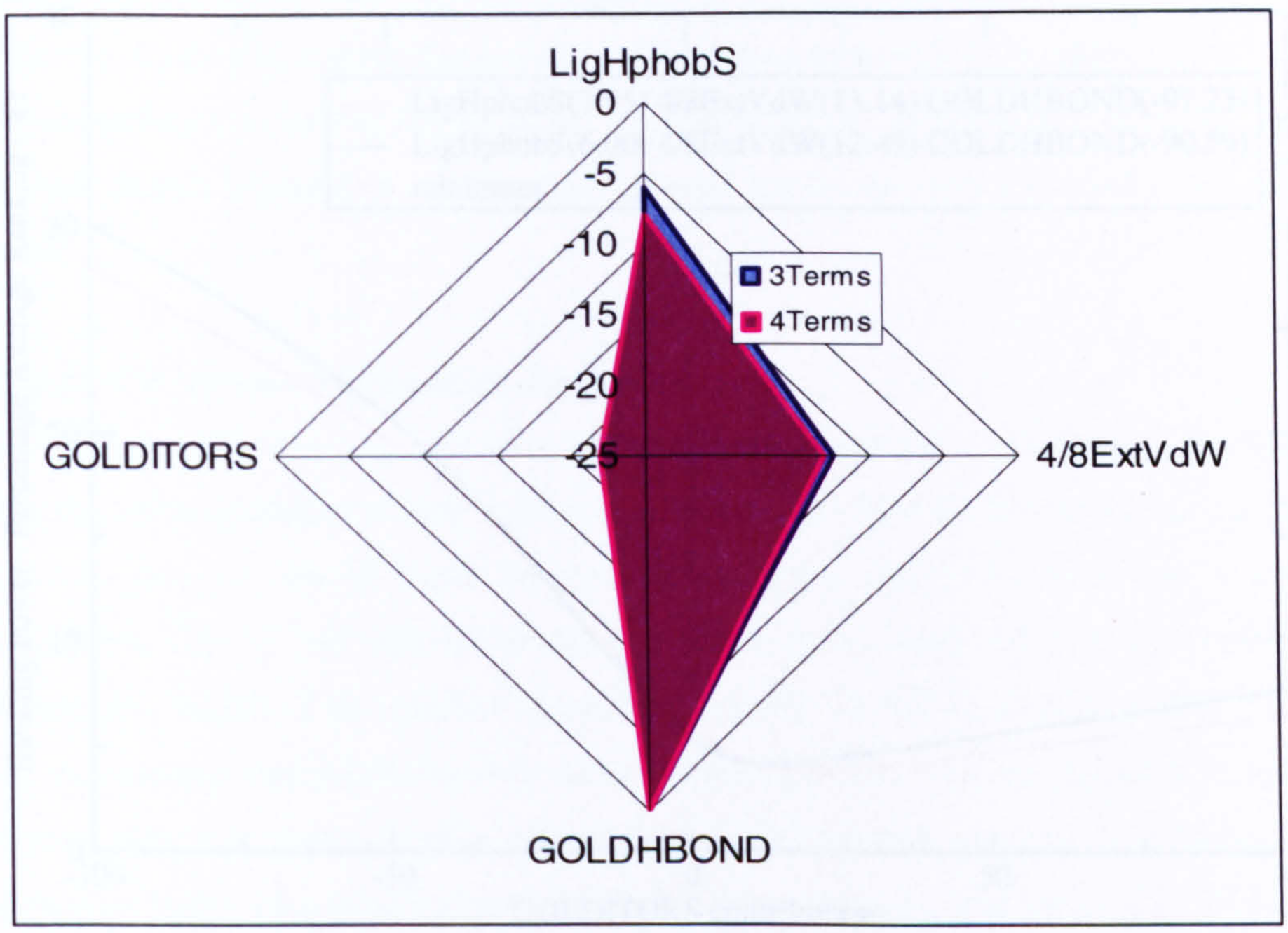


Figure 6-13: Representation of the three and four terms solutions in the terms space. Similar functions generate similar shapes.

Was the very best coefficient picked for GOLDITORS? Could there have been a better choice?

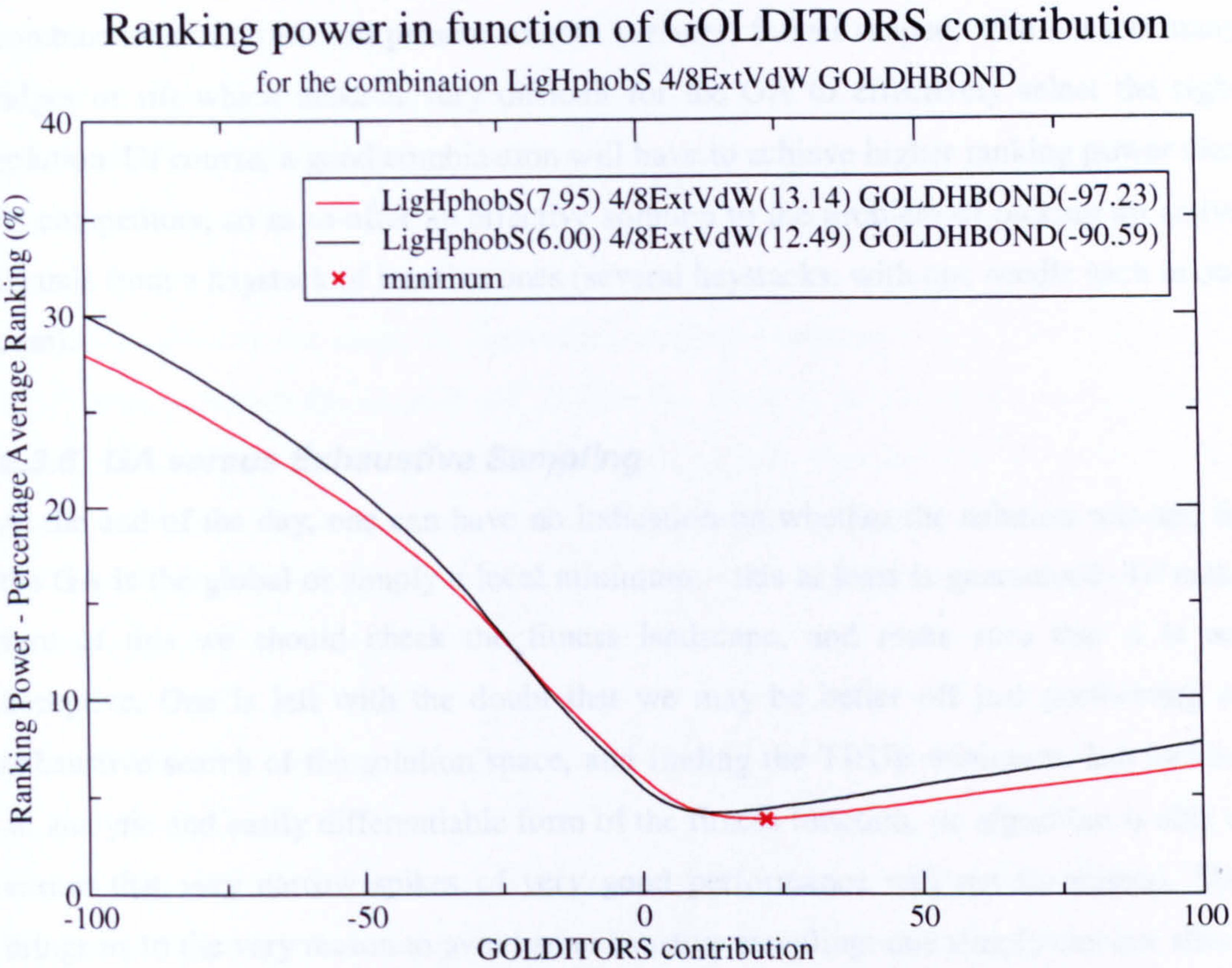


Figure 6-14: The ranking power variation with the varying contribution of GOLDITORS. Black is the optimal three-term solution. Coefficients for the red line are taken from the optimal four-term solution. The cross is where the GA has converged – that is, the absolute minimum.

From what is shown in Figure 6-14, our conclusions can be drawn. In this case, the black line represents the family of solutions generated starting from the best 3 term solution, and scanning the GOLDITORS contribution from its minimum (-100) to its maximum (100) value. The red line is instead the family of solutions, generated the same way, starting from the coefficients values of the best four-term solution found. The red line has a minimum lower than the black, but most importantly it is also lower in general. In this case, the GA sampling and selection routines can easily explore both families of solutions, but will generally prefer points on the red line. Therefore, when given the possibility to explore the GOLDITORS axis together with the other three, it will correctly converge on its minimum (red cross).

The conclusion drawn from the two examples shown is that good combinations of terms are characterised not only by the presence of solutions with high ranking power, but also by the fact that their solution landscape is not deceptive[34] for the GA: *i.e.*, it does not trick the GA into choosing a local minimum rather than the truly global one. Good

combinations of terms will present smooth surfaces, funnel shaped, with not too many ridges or rift which make it very difficult for the GA to effectively select the right solution. Of course, a good combination will have to achieve higher ranking power than its competitors, so as to offer an effective solution to the problem of picking up active ligands from a haystack of inactive ones (several haystacks, with one needle each in our case).

6.3.6 GA versus Exhaustive Sampling

At the end of the day, one can have no indication on whether the solution selected by the GA is the global or simply a local minimum – this at least is guaranteed. To make sure of this we should check the fitness landscape, and make sure that it is not deceptive. One is left with the doubt that we may be better off just performing an exhaustive search of the solution space, and finding the TRUE minimum. But lacking an analytic and easily differentiable form of the fitness function, no algorithm is able to ensure that very narrow spikes of very good performance will not be missed. This brings us to the very reason to avoid an exhaustive sampling: one simply can not afford it.

On the average machine used during this part of the project (AMD Athlon64^{tm}, 3 GHz), with an input file of 108 proteins and one thousand ligands per protein, every fitness evaluation requires around one second: the time is spent ranking all 108 reference ligands versus the other thousand decoys.

If one were to perform an exhaustive scanning of all possible solutions for every one of the terms, the number of points to evaluate grows with the N^{th} power of the sampling point density, where N is the number of axes to be considered. Something similar has been performed in section 6.3.3, but limiting the scanning to one axis only: this required the evaluation of every solutions between -100 and 100, with a step 1; that is 201 evaluations, or slightly more than three minutes in real terms. To perform an exhaustive scan of a three-term combination with a step of 1 of would require 201^3 fitness function evaluations. The third power of 201 is somewhat over 8 million, so, 2200 hours. This is for a single combination of 3 terms (out of 28), and quite a large sampling step. As shown by Figure 6-11, some terms can vary their contribution steeply in a very narrow range of values. Therefore, one should allow for iterative sampling around sub-optimal

region with smaller steps in order to select the 'true' minimum: and this is only going to increase computational demands for an exhaustive scan. As a result, requirements for the evaluation of all possible combination, or even only those selected in a step-wise process like ours, are prohibitive.

Of course, the correct mode to approach complete sampling would be to perform a constrained selection by classes and values, but even at this level the computational requests would be several orders of magnitude greater than the one we used for this whole work. So, like mother nature, we have to make good of what we have, get the best scoring function found by the GA, and leave the truly perfect scoring function on its shelf, just on the side of our wheeled monkey.

6.4 Different Functions for Different References

The scoring function(s) being created here will be used in a structure-based virtual screening environment, where it/they will have to select active compounds from a database of docked structures. This is certainly a task different from the one we just detailed, where the reference structure to be highlighted is the COLO. A more proper training would be for the function to pick up reference ligands actually docked within the target, the same way the decoys are. This has been pursued following two different approaches: a common step to both is that, using GOLD as the docking engine, we have re-docked the true ligands within their own active site. The same GOLD settings as for the decoys were used, modifying the active site radius in the cases where a close enough conformation to the COLO could not be obtained. In those cases, active site radii were first scaled down to 8 Å, then the sampling parameters modified to force the production of the wanted pose (however unphysical this could be). We did this since we wanted to acquire at least a ligand pose generated by GOLD with an RMSD less than 2.5 Å from the COLO. RMSD was measured for all atoms (not only heavy atoms) using a PERL script developed in-house.

Then, two new references for each protein were selected: the ligand pose closest to the COLO (herein *Closest*), and the one with the best (highest) GOLDSCORE value (herein *Goldest*). For proteins which required restricted radii, or modified GOLD search parameters to select a *Closest* structure, poses generated during the "failed" attempts

were used when choosing the *Goldest*. These two sets of references were then used to assemble two new GA input files, which then underwent the same step-wise parameterisation already described. A shell script selected the ligand with the lowest RMSD (or the highest GOLDSCORE value) and assembled the corresponding GA input files.

As a general rule, we would expect the functions created from these data to be less reliant on clash terms when differentiating inactive decoys and active ligands: now both references and decoys have been minimised within the same force field (GOLDSCORE). Previously, the COLOs had been minimised within the force field of the program that solved/refined the crystal structure; because of the different functional form, atom types and parameters, we would expect some clashes when applying GOLDSCORE to these orientations.

We will treat the two new parameterisations (*Goldest* and *Closest*) together, because the results for at least the first few terms selected are similar, and because a direct confrontation is useful, both in terms of the interactions included, and the difference in their values, to understand the common differences with the COLO-trained functions.

6.4.1 Closest and Goldest

We want to show in this section how the new parameterisation set affected the resulting scoring functions: for the *Closest* set, the first two terms to be selected were 4/8ExtVdW with a coefficient of 0.62, and GOLDHBOND, with -85.9. The terms' ratio is -138, once again reflecting their different roles, when you account for the difference in typical magnitude for the two scoring terms. One term (GOLDHBOND) is favourable for binding, the other (4/8ExtVdW) adverse. At the same time, 4/8ExtVdW scores in the region of thousands, and GOLDHBOND typically between one and ten. The ranking power of this combination is such that the true ligands are found, on average, at 8.6% of the ranked list of one thousands decoys.

For the *Goldest* set, the terms selected are the same (4/8ExtVdW: 1.6; GOLDHBOND: -87.9), but with a different ratio between the two: -54.9. Ranking power is higher than its sibling *Closest* function, at 7.1%. This seems to suggest that *Goldest* is better at selecting actives. This is welcome, since during a SVS we would not have any

information on which generated pose is closest to the experimental structure for a new, uncharacterised active ligand, whereas selecting the best scoring pose is straightforward. We do not know at the present time if this is peculiar to GOLDSCORE, or applicable to every scoring function used to generate/minimise our poses.

	4/8ExtVdW	**GOLDHBOND**	***SCORERANK***	*Ratio HBnd/VdW*
COLO	2.1	-89.3	***12.4%***	*-43.1*
Closest	0.6	-85.9	***8.6%***	*-138.0*
Goldest	1.6	-87.9	***7.1%***	*-54.9*

Table 6-19: Coefficient values for the best two terms selected by the GA parameterisation routine when using three different reference structures: COLO is the experimentally observed ligand orientation; Closest is the re-docked pose most similar to COLO, Goldest is the pose with the highest GOLDSCORE value.

Table 6-19 shows the two solutions against each other. We also include the solution for the same combination, but obtained using the COLO as references. We note that in this case, the combination was definitely well performing. It did rank as the sixth best out of the 343 two-term combinations. However, its performances (over three runs) were around 40% worse than the best performers (see Table 6-4).

Let us consider more in detail the terms and coefficients selected. The preference of both *Goldest* and *Closest* for a hydrogen bond term rather than a surface term means that now the main factor discriminating binding and non-binding molecules is the specificity of interaction, since hydrogen bonds are very directional and cannot be formed by any molecule that just happens to fit in the pocket. True ligands, once minimised the same way as the decoys, are no longer seen as if they were 'hiding' from water, a nonspecific interaction, but are actively seeking favourable interactions, such as lone pairs and polar hydrogen atoms with which to establish (hydrogen) bonds. This still happens, of course, under the strong constraint of shape-complementarity, as shown by the presence of the 4/8ExtVdW term, which acts as a clash term because of its positive coefficient.

Let us now look at the different ratios between GOLDHBOND and 4/8ExtVdW. Comparing *Goldest* and *COLO* we see that the contribution of the van der Waals term is decreasing, as expected. This may mean that with *Goldest*, a lower van der Waals contribution is able to discriminate between decoys and actives. However performances for the COLO reference structures are poorer than for *Goldest*, as we saw earlier. Once

again, this can be ascribed to the references being optimised with a different force field to the decoys. So, it turns out that with less van der Waals and a tad more GOLDHBOND, *Goldest* poses are more easily selected than COLO.

We can now compare *Goldest* and *Closest* best solutions against each other, especially in the case where both work well. Apparently, for the *Goldest* there is a contribution twice as large from the van der Waals term (4/8ExtVdW), compared to the *Closest* (half the ratio is equal to twice the importance). This sounds counter-intuitive. If *Closest* is the nearest pose to the COLO, for which the clash term was very strong, should it not be more sensitive to the clash contribution? How can this be explained? Well, this is in fact what the scoring function is telling us. *Goldest* reference structures require a higher van der Waals coefficient than *Closest*. This is because *Goldest* poses should be at the very minimum of the pocket, being the optimum GOLDSCORE potential for that molecule. *Closest* are instead some sort of compromise between the position we know to be the best in the GOLDSCORE sense, and the truly observed configuration. *Closest* poses will still suffer from minor clashes with the protein, compared to *Goldest*. Therefore, 4/8ExtVdW still has a good discriminating power between *Goldest* and decoys, but much less between those and *Closest*. Consequently, the resulting coefficient derived for Goldest, will need to be larger to offer discrimination.

108 proteins:	***Tested On***		
Trained On↓	**COLO**	*Goldest*	*Closest*
COLO	***12.4%***	***7.3%***	***9.2%***
Goldest	***12.6%***	***7.1%***	***9.0%***
Closest	***14.3%***	***7.6%***	***8.6%***

Table 6-20: performances of functions trained on different kind of references against each other training set – the functions used are presented in Table 6-19.

We have tried to see how effective our COLO-parameterised solution is to retrieve the *Goldest* orientations from the SVS results on the basis of their score (Table 6-20). All combinations of training and test set were considered, although ideally one would like a function trained on COLO to recover *Goldest* hits from SVS – with training on *Goldest* as second best option.

As can be expected from the similarity of the scores, the function trained on COLO turned out to be only slightly less efficient than the *Goldest*-parameterised solution

itself, ranking the reference on average at 7.3% rather than at 7.1%. When the opposite is done, the *Goldest*-parameterised solution is able to rank (on average) the COLOs within the first 12.6% of the docked molecules' list. When the COLO-parameterised solution was used against the *Closest* poses, retrieval strength was at 9.2% - strikingly, it was better at retrieving *Closest* structures than the structures it was parameterised against. The opposite procedure, Closest retrieving COLOs from the lists, gave a result of 14.3%. There is nothing unexpected in this. Solutions not-too-far from optimum still have a good result; however, it is not by any means a rigorous test of the goodness of our scoring function, as the protein and ligands are essentially the same. More rigorous tests will come later.

6.4.1.1 Higher order combinations

Continuing our usual procedure of adding one term after another, the best third term to add is the same in both cases: 4/8IntVdW, our substitute for a rotational term. This is hardly surprising: since the poses are minimised within GOLDSCORE, its internal torsional term, GOLDITORS, cannot offer any discriminating power between decoys and references. Also, with 4/8ExtVdW offering basic shape-complementarity, and GOLDHBOND giving directionality, there is not much else to use to get rid of decoys, other than checking those molecules that adopt strained internal conformations[JW52]. In fact, the term turns out to have the same sign in both cases, with more or less the same absolute value, and also the increase in performance is similar, a gain in absolute ranking power of 0.9% for *Goldest*, and of 0.7% for *Closest* – that is roughly equivalent to a 10% improvement in both cases.

The 4th term added is the same (BadHBond) too, but the improvement in performance is minimal, and the further addition of terms is superfluous.

Concluding, it can be said that the most important feature when using this alternate references is that the hydrophobic surface term, expressing the tendency of molecules to hide from water, is taken out of the equation. The van der Waals contribution is instead more significant. The reason for this is as yet unclear: certainly it appears that GOLD's van der Waals term does a very good job at matching shapes of ligand and protein.

6.4.2 Crystallographically Observed Ligand Orientations simplexed in GOLDSCORE

Selecting the *Closest* docked pose is not the only way to get a pose as close as possible to the COLO. There is another way of getting rid of (or at least lessen) the clashes introduced by the use of a different force field in refining the structure, and/or during docking. It is possible to perform a simplex minimisation directly within GOLD, and then rescore the COLOs to obtain new scoring terms. Of course, the new simplexed position (COLO-spx) will only be minimised with respect to GOLDSCORE, not our own scoring terms, but this is meaningful, since GOLDSCORE is built to reproduce known binding structures; moreover, the terms that will be used have not been chosen yet, so for example one would not know whether to use one hydrogen bond model or the other, or both at once.

As for the decoys, their structures have been produced within GOLD using GOLDSCORE to drive the sampling, so they are already at minimum positions for the GOLDSCORE potential – since GOLD by default performs a simplex minimisation on every output pose. They do not need further minimisation, although this would not take long, if another program/function had been used – e.g. in the case where they have been docked using CHEMSCORE. Of course, minimising poses obtained by other functions/programs does not ensure that the global minimum will be found, only the nearest local minimum.

However, parameterising against these new simplexed positions gives a few surprises: as in *Goldest* and *Closest*, LigHphob terms disappear from the preferred two-term combination, substituted by GOLSCORE's own hydrogen bond model, GOLDHBOND. The contribution is -14.9, much smaller than for *Goldest* and *Closest* but still decisively in favour of binding, showing that true ligands form more and stronger hydrogen bonds than decoys. The surprise lies in the fact that 4/8ExtVdw is substituted by a rotational/internal-clash term, 4/8IntVdW (coefficient 0.2, a slight penalty). This situation is most likely due to the fact that, having minimised protein/ligand clash interactions within GOLDSCORE, the corresponding terms (GOLDEXTVDW, 4/8ExtVdW and similar) are not sufficiently discriminating to distinguish between true ligands and decoys[JW54]: we can only guess as why this was not observed in the equally minimised *Goldest* and *Closest* – may be the poses being generated from scratch rather

than just minimised resets the clock and still allows the Protein/Ligand 4/8ExtVdW to play a role – albeit reduced, as shown in Table 6-19. Note also how in *Closest*, whose reference poses are arguably more similar to the minimised COLO, 4/8ExtVdW has the least importance of all (again, see Table 6-19).

GOLDHBOND, instead, retains its very important role, because most decoys will not form a favourable network of hydrogen bonds with the protein, so that the function's ability to select the active ligands will still be strong. However, the ranking performance is worse: the reference is found, on average, at 12% of the ranked list, so the approach of directly minimising the reference pose is probably not the best option, and when possible, a proper re-docking should be performed. This has limited consequences in the real world application of the scoring functions, however, since during SVS all ligands are docked and we do not usually know which our actives are.

6.5 Conclusions

In this chapter, by means of presenting some results of our variable selection and parameterisation routine, *caveats* have been raised concerning the correlation of terms to be included in the scoring, and the shape of the energy surfaces they generate; also, general tools to analyse the goodness and diversity of results have been introduced.

The general conclusions that can be drawn from this are various. Great care must be taken during implementation to make sure that scoring terms do not describe the same interaction. Such correlations are discarded by the parameterisation routines on the basis of their limited efficiencies compared to more diverse variable sets – they represent, however, a waste of computational resources better employed elsewhere. Second, the parameterisation routines consistently choose the best combinations, and these seem to exhibit a remarkable simplicity of the energy surface, with a smooth funnel-shaped form leading to the best solution, and without ridges, crevices or other structures which can inhibit a proper sampling of the minimum. Finally, using non-crystallographic reference structures can still provide good performances in the ranking. However, docked structures are to be preferred to poses generated by simple minimisation in a force field, even when the poses to be minimised are of experimental origin.

The Table 6-21 summarises some of the results shown until now. For every different kind of reference pose, the best performing function with a given number of terms is presented, together with its performance on the set itself.

Terms→ Set↓	2	3	4	5
COLO on COLO	LigHPhobB 61.11 4/8ExtVdW 79.15 8.68 %	LigHPhobS 2.48 4/8ExtVdW 4.16 GoodHBond -99.88 5.14 %	LigHPhobS 7.89 4/8ExtVdW 13.03 GOLDHBOND -96.40 GOLDITORS 21.79 3.83 %	LigHPhobS 8.07 4/8ExtVdW 10.04 GOLDSCORE -54.18 GOLDITORS 70.78 UnsAccept 43.46 3.59 %
Goldest on Goldest	4/8ExtVdW 1.600 GOLDHBOND -87.93 7.10 %	4/8ExtVdW 1.58 GOLDHBOND -79.09 4/8IntVdW 0.17 7.98 %	4/8ExtVdW 1.77 GOLDHBOND -87.29 4/8IntVdW 0.17 BadHBond 29.07 5.96 %	4/8ExtVdW 1.70 GOLDHBOND -83.52 4/8IntVdW 0.18 BadHBond 35.21 TotHPhilB -0.69 5.84 %
Closest on Closest	4/8ExtVdW 0.62 GOLDHBOND -85.89 8.62 %	4/8ExtVdW 0.77 4/8IntVdW 0.20 GOLDHBOND -100.00 7.95 %	BadHBond 32.68 4/8ExtVdW 0.54 4/8IntVdW 0.19 GOLDHBOND -75.73 7.60 %	BadHBond 42.25 Coul -18.13 4/8ExtVdW 0.72 4/8IntVdW 0.18 GOLDHBOND -92.29 7.49 %

Table 6-21: Performances of the best 2-,3-,4-,5-terms functions trained on the COLO, *Goldest* and *Closest* training set.

In the next chapter, we will present further results of the parameterisation, and performance of the solutions selected, with particular emphasis on the robustness of said functions when the number of parameterisation complexes is reduced and varied – cross-validated performances on test sets will also be provided.

7 Scoring Function Testing

7.1 Testing the robustness

A common pitfall of empirical scoring functions is the strong dependence of the terms' coefficients from the set of complexes used for the parameterisation. This is of course expected when the function is being optimised to reproduce known experimental data related exactly to that set. However, this also means that the good performances of a scoring function will be limited to complexes and interaction modes already represented within its own training set. In short, the functions are not necessarily transferable, with values of the characterising parameters fine-tuned to describe the peculiarities of their training set.

In this chapter, the protocol followed to test the robustness of our function is presented. Rather than obtaining a set of coefficients from all 108 protein complexes in our training set as shown in the previous chapter, the set was split, a procedure extensively detailed in section 4.3.3. Parameterisation was then performed on each subset, and the resulting functions tested on the others. With this setup, it has been possible to test the function(s) on a subset sometimes non-overlapping, sometimes partially overlapping (but to different extents) with the training set. This also provided an indication on how the size of a training set (27, 54, 81, 108 complexes) can influence the performance and the robustness of the functions.

7.1.1 Nomenclature

Because so many subsets have been used during this part of the study, a clear and careful nomenclature is necessary to avoid confusion: the method devised is to call each training/test set by the number of protein/ligand complexes it contains, followed by a letter: so, 27-A will label the first 27 complexes set. 54-B, instead, is the second of those with 54 proteins inside them. And so on. As previously explained in section 4.3.3, three series of four sets have been created, numbering 27, 54 and 81 complexes. What is left of the main set once the four 81-complex sets are taken, is four other 27-complex

sets. These are called 27-E to 27-H. Let us now see some results from the various parameterisations and testing of these sets.

7.1.2 27-complex sets

In this section, the results relative to functions trained on the four 27 [JW55]complex sets will be presented. Please remember that the four sets were obtained by recursively splitting in half the full training set, so that none overlaps with any of the others. All underwent the GA-based parameterisation procedure which is by now familiar, starting from 2-term combinations and building step by step up to an n-terms scoring function. Intuitively, it would be easier for the GA to find coefficients able to describe a mere quarter of the proteins previously used: the expectation was then to see better enrichment in these cases compared with training on the full 108 complexes set (results shown in chapter 8). However, it was unknown whether this would affect the reliability of our functions when used outside their own training sets, and if so, by how much.

27-A			27-B		
COMBO	Best Rank	Ave Rank	COMBO	Best Rank	Ave Rank
2_terms/14-19/C	6.13%	6.15%	2_terms/14-19/B	7.83%	7.85%
2_terms/18-19/C	6.44%	6.47%	2_terms/18-19/A	8.28%	8.30%
2_terms/13-19/A	8.35%	8.36%	2_terms/18-28/B	8.85%	8.85%
2_terms/16-19/B	8.83%	8.84%	2_terms/14-28/C	9.08%	9.10%
27-C			27-D		
COMBO	Best Rank	Ave Rank	COMBO	Best Rank	Ave Rank
2_terms/14-19/B	8.06%	8.07%	2_terms/18-19/C	5.72%	5.74%
2_terms/18-19/B	8.81%	8.82%	2_terms/14-19/A	6.17%	6.18%
2_terms/16-19/A	9.30%	9.31%	2_terms/16-19/C	9.80%	9.82%
2_terms/20-26/B	10.38%	10.46%	2_terms/13-20/B	10.01%	10.02%

Table 7-1: Best performers between the 2-term combinations trained on four different 27 strong complex sets: the performance shown for each combination is on its own training set. Only one result per combination is shown (i.e. 14-19/C but not A or B).

A first test for this was to perform a cross scoring using parameters determined using one set (*e.g.* set 27-A) on another set (*e.g.* set 27-B). If the ranking power is conserved from one to the other, then the same performance level for every complex the function may encounter in its use may be confidently assumed. Exceptions must be allowed for atypical complexes, where interactions may be peculiar or for which, simply, terms missing in our model could be required.

So, which pair of terms did our functions find that best describe protein-ligand interactions? Results are shown in the Table 7-1.

A comparison with the equivalent results for the full training set, from Table 6-4 shows that the terms preferred are pretty much the same, *i.e.* a van der Waals method (n° 19 or 28, mostly) plus a surface evaluation (n° 14 and 18 preponderant). As expected, the fitness obtained at the end of the GA is improved compared to the one reported when training on the full set (see Table 7-2). This is mainly due, as explained before, to the fact that the GA's task is easier now, with only 27 complexes to work with. The performances of the best functions are spread along the 6 to 8% of Average Ranking for the reference. The observed differences can be attributed to the detailed composition of the four sets. However, set 27-B shows a peculiar difference in the first terms selected, since GOLDITORS (term n° 28) appears there. Arguably, this means that for one or more complexes present in 27-B (but not in the others) this term is particularly important to discriminate between reference and decoys. For example, some reference ligand may be inserted in a narrow pocket which causes strained poses for the decoys – or if the sign of the contribution is reversed, the reference ligand itself may be strained.

SET	*LigHPhobB*	*4/8ExtVdW*	SCORERANK	1st/2ndRATIO
108 p	61.11	79.15	***8.68 %***	0.77
27-A	47.21	88.15	***6.13 %***	0.54
27-B	83.66	54.78	***7.83 %***	1.53
27-C	51.25	85.87	***8.06 %***	0.60
27-D	62.99	77.67	***6.17 %***	0.81

Table 7-2: Optimal solutions for the combination *LigHPhobB – 4/8ExtVdW* when trained on the full training set and its four subsets.

Of course, if differences in the composition can affect the fitness, they will surely affect the equilibrium between the terms themselves. This is shown by a more detailed look at the selected solutions for the first few terms: for the sake of simplicity and clarity, only the combination LigHPhobB-4/8ExtVdW was chosen, since it ranks first in three out of four sets, and it is the best performing combination for the full training set too (Table 8.4[JW56]). All solutions have been reduced to their normalised percentage form, so that comparison between them is easier[JW57].

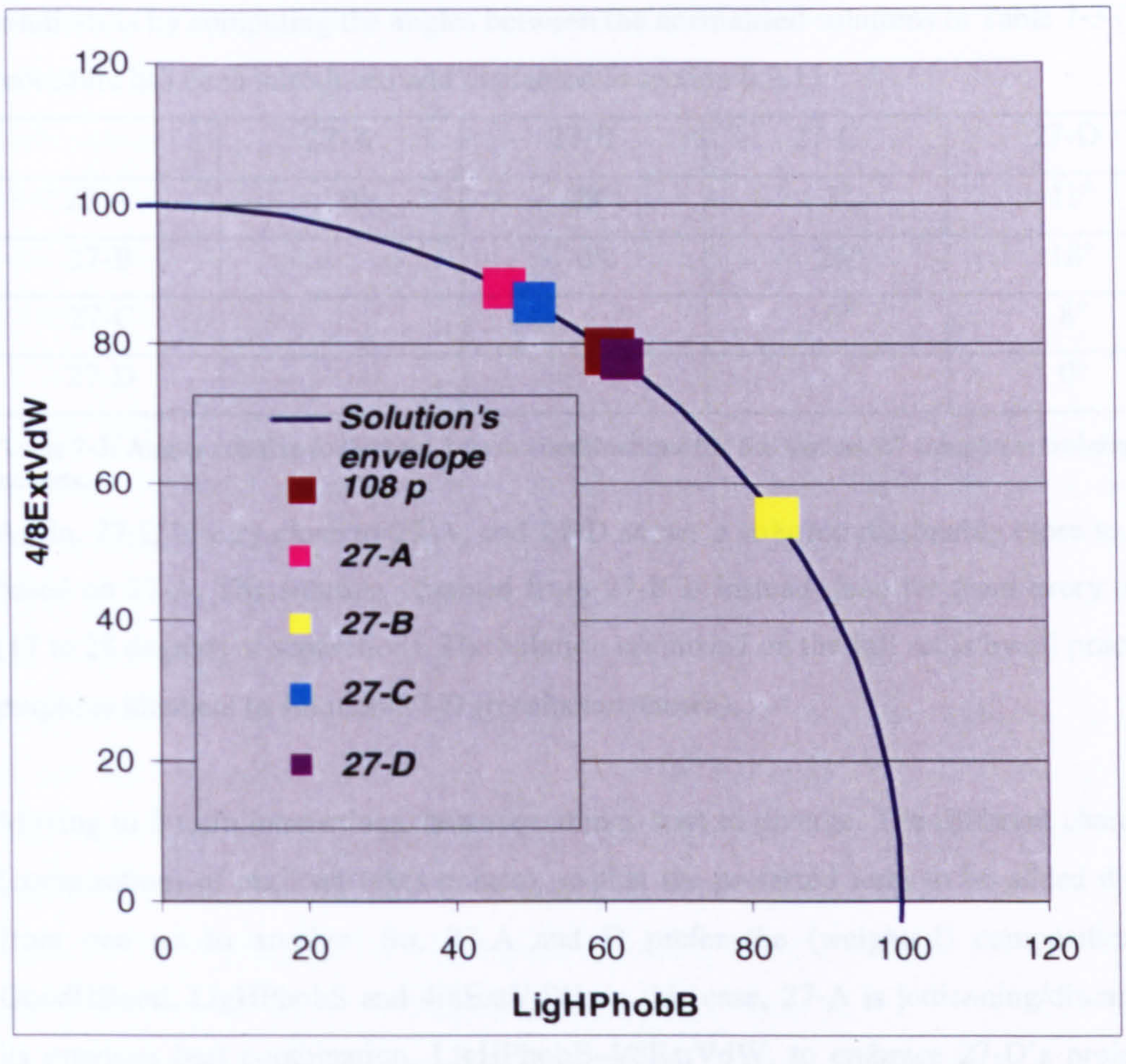


Figure 7-1: Best solutions[JW58] for the combination LigHPhobB – 4/8 ExtVdW when parameterised against COLO reference poses.

Table 7-2 compares the results of the combination LigHPhobB – 4/8 ExtVdW on the full training set of 108 complexes and the four mutually exclusive subsets of the same (27-A to D). Figure 7-1 shows the same results graphically. The "Scorerank" column in the table gives the ranking performance, whereas the last column attempts to establish similarities between the different results by providing the ratio between the first and the second coefficient. As can be seen from the figure, we mainly observe a fairly similar balance of the two terms, with the exception of set 27-B.

This goes a long way to show that even if the preferred terms are the same for every set their optimal ratio varies to reflect differences in the sets' compositions. The ratio between first and second term varies from roughly 1/2 for 27-A to 3/2 for 27-B, with 27-C and 27-D somewhere in between. This is, of course, an indication that the complexes inside each and every set have a different balance between the various kinds of interaction. The ratio's average value is .87, not far from the .77 reported for the solution trained on the full set. Another way of looking at the difference between

solutions is by computing the angles between the normalised solutions in Table 7-3 (this procedure has been introduced and explained in section 6.3.1):

	27-A	27-B	27-C	27-D
27-A	0°	29°	3°	11°
27-B		0°	26°	18°
27-C			0°	8°
27-D				0°

Table 7-3: Angles relative to the best 2-term combinations for the various 27 complexes training subsets.

Again, 27-C is very close to 27-A, and 27-D shows a solution reasonably close to that based on 27-A. The solution obtained from 27-B is instead quite far from every other (17 to 28 degrees of separation). The solution optimised on the full set is by all practical purposes identical to solution 27-D (results not shown).

Moving to 3-term interactions, however, things start to diverge. The different character (composition) of each set takes control, so that the preferred term to be added differs from one set to another. So, 27-A and D prefer the (weighted) composition of GoodHBond, LigHPhobS and 4/8ExtVdW: in this case, 27-A is jettisoning/discarding its previous best combination, LigHPhobB-4/8ExtVdW, to embrace 27-D's preferred solution. It must be noted though that term 14(LigHPhobB) and 18(LigHPhobS) are strongly correlated, so this is not the huge change that it appears to be at first. Set 27-B instead keeps its best two-term combination, and adds the 6/12IntVdW as third term; that is the intra-molecular Lennard-Jones potential which in our model is supposed to express the internal strain of the ligand. Finally, set 27-C keeps the LigHPhobB-4/8ExtVdW, though it adds GOLDHBOND as third term. In all these cases, the three runs performed for every combination still manage to give the same exact solution, meaning that the GA is still (reasonably) converging on what is the global optimum. Performances on their own training sets look good, with best combinations able to rank the Crystallographic Observed Ligand Orientations (herein COLOs) at 3-5% of the ranked list of 1000, on average.

The different preferences of terms can immediately be translated into a tentative explanation: set 27-A, C and D added a hydrogen bond term, therefore indicating that this interaction is important for the correct description of their complexes. It can be

noted that for set A, the best second combination is given by LigHPhobB, 4/8ExtVdW and GOLDITORS, adding to the original best 2-terms a ligand internal strain term. The performance of this triad is only slightly worse than the best chosen option. However, whereas the hydrogen bond term's contribution is usually good for binding, the GOLDITORS term acts as a penalty, which implies that COLOs *suffer* from it less than decoy molecules.

At the level of the fourth terms selected, the divergence originating from the sets increases. These terms are less critical, and therefore the larger variability is expected. A few combinations which are also best performers when trained against the full training set are still present here. Namely, 18-19-25-26 at position 4 in 27-C was second best in the 4-term combinations for the full set (see Table 6-11[JWS9]). Mostly, the new combinations are derived from the best performers for each set in the previous tier. This is also the last level at which there is a considerable increase in ranking performance. The percentages at which the COLOs are found on average are halved compared to the best three terms functions. At five terms, instead, the *relative* improvement is barely some 10%. This suggests, once again, that more than four (of our) terms do not improve the description of the protein/ligand interaction. Ranking powers for COLOs of the respective training sets are once again slightly better than for the full 108 complexes set, because of the ease for the GA to treat only a quarter of the proteins. The function just presented will be tested later on, in the second part of this same chapter: the testing sets for functions trained on 27-A will be sets 27-B to D, and so on.

We now consider functions trained against sets containing more complexes.

7.1.3 54 protein sets

The four 54-protein sets used here are, as previously detailed in section 4.3.3, randomly picked from the full training set of 108 complexes. With reference to the 27 protein sets previously introduced, the first 54 protein set, 54-A, is composed by the union of the 27-protein sets 27-A and 27-B. The set 54-B, likewise, is composed by the union of 27-C and 27-D. Another independent partitioning of the full 108 complexes' training set has given us 54-C and 54-D, again with no elements in common between them, but each partially overlapping with both 54-A and 54-B. Please refer to section 4.3.3 for further details.

The parameterisation results from these sets do not offer any particular insight or surprise. The best two-terms selected are once again the same for all sets, LigHPhobB or LigHPhobS (n° 14 or 18 in Table 6-1[rw62]) and 4/8ExtVdW (n°19). Other selected terms such as LigHphilB (n°13) or TotHhobB (n°16) or again 6/9ExtVdW (n°20) suffer from a *relative* performance hit of around 20% with respect to the best performers. In general, performance on their own training set is worse than for the 27-complex cases: their larger size causes a greater difficulty for the GA in finding a good compromise. Rankings of COLOs range depending on the sets: from 10% of the ranked list for the best performer on 54-A, to slightly more than 7% for 54-B and 54-C, with 54-D's best performer at 9%.

Evolution to 3- and 4-term combinations for the 54 complex sets is more regular than for the 27-proteins' sets seen before. Every set now has twice the number of proteins, so it is a more representative sample of the full set, and hopefully for the whole protein world too. There is less likelihood now for sets with irregular distributions, where some complexes may skew the results towards uncommon, otherwise neglected interactions.

7.1.4 81 protein sets

This is our last partitioning of the full 108 protein set. 81 complexes were randomly selected four different times. Each of the 81-protein sets overlaps with any other for 60-odd members. The overlaps are however different for any given pair. As far as parameterisation is concerned, passing from 54 to 81 training complexes one would expect to obtain functions with greater generality: even if this translates initially into less focused functions, unable to rank all of the training complexes at the top of the list, robustness and transferability should improve. This is, more or less, what the results seem to show. As seen before, LigHPhobB (term n°14) and LigHPhobS (n°18) compete to pair with 4/8ExtVdW (n°19) as best 2-term combination; GOLDHBOND (26) or GoodHBond (1) enter as typical third terms. Usually, the fourth term to be selected is GOLDITORS, except for some combinations where it is substituted by the full GOLDSCORE, strongly correlated with it; notably the GOLDSCORE/GOLDITORS contribution is of opposite sign to that of GOLDHBOND, often already present in the combination. A very interesting annotation can be made on the results, shown in Table 7-4:

81 proteins	*LigHPhobB*	*4/8ExtVdW*	SCORERANK
Set 81-A	*45.29*	*89.16*	9.20%
Set 81-B	*59.00*	*80.74*	9.21%
Set 81-C	*59.01*	*80.73*	9.56 %
Set 81-D	*59.61*	*80.29*	10.27 %

Table 7-4: Best 2-term functions trained on the four different 81-protein sets.

As can be seen from Table 7-4, the best performing two-term functions trained on the four 81 complexes share the same terms. The similarities do not end here: three of them, trained on set 81-B, C and D, are almost exactly the same function. 81-A's function is reasonably different, separated by less than 10°: this difference may be ascribed to proteins present only in this set that differ. However, it must be noted here that such over-rationalisation can be dangerous, and should be avoided: the difference may lie in the proteins present only in set A, true, but could also be ascribed to proteins missing from 81-A and present in the others instead. Finally, since the difference is not that large, it could just be a perfectly acceptable fluctuation of the values around a *generic* function, perfectly justified by small changes in the importance of interactions describing each set of complexes. Certainly, though, even if these differences cannot be fully explained, this case shows how training on different sets can provide a fundamentally equivalent function – and conversely, how the function may drift due to a peculiar training set.

7.2 Testing the performances

Until now, the discussion has been centred on ranking performances against the training set. This is, however, not really indicative of their performances on true cases, and is mostly an indication of how efficient the fitting process has been in extracting information from the data[35]. In fact, a fitting procedure that is too efficient may be harmful. It would *over-fit* our function(s) to our training data, making it useless on other cases, even slightly different. Fortunately, the only functional form our scoring functions can take are linear combination of terms, and this makes it almost impossible to find a single combination of coefficients able to rank all COLOs included in the training set at the first place in their own list. In essence, the idea is to extract as much

information as possible regarding the balance of the different interactions, but with the minimum quantity of noise.



Figure 7-2:The ability to first-rank a training set's truly active compound increases with the number of terms used, but after four terms, the performance on the test set lags behind. Results shown are for functions trained using COLOs as references.

It is now time to properly test our functions, and (lacking a proper independent test-set) it was decided to do this by cross-scoring complexes from the various training subsets with functions parameterised on the others. Having both subsets of equal size but differently composed, and training subsets of different dimensions, the investigation can be conducted along two different axes: first, whether or not the different size of the training set affects its reliability, and how. Second, how, with the size of the training set equal to that of the test set, a function performs on test sets different in composition from the training set.

For reason of space, we could not possibly present and discuss testing of functions composed of two to six terms for all sets: so, if one were to pick a number of terms on which to perform the analysis, which one should he go for? From what we have shown previously, we believe that for our setup the optimal number of terms is four (see Figure 7-2); further terms seems to add very little to the binding description, and we do not see any improvements in the ability to rank active compounds other than those contained in the training set. Additional information to this regard is presented in the next section, Figure 7-3.

Therefore, for the next section we did consider investigating the 2, 3 and 4 terms combinations and their performances on independent test sets. The 2-terms show the starting point, and how a very simple function can already capture the basics of interaction, and offer reasonable performances. The 4-terms are included, because they are the most likely to be used in real applications. The three term combinations generally represent a transitional phase, and as such do not offer (as a general rule) enough additional information to justify lengthy discussions on them, although this will be done in some particular cases.

7.2.1 Performances of the 27P parameterised functions

We do not consider 2 term combinations for the 27-protein sets, because of what was shown when presenting the parameterisation results. The different sets have different functions as their best performer, and the sensitivity to little biases in the training set seemed quite high, We can expect these functions not to be very transferable from one set to the other, and this is confirmed by the analysis, which were ran, but will not be shown in detail. Let us start by selecting some 4-term functions parameterised against one of the 27 complex subsets. These combinations will be fed to our program QSCORE, which will in turn use them to evaluate the ranks for references contained in other GA input files whose structures are exactly the same as that previously shown in section 5.4 and 5.5.

Best performers from the four-term combinations for every 27-complex set are shown in Table 7-5: in every line the set number and the corresponding best combination, with terms and coefficients, are detailed. Commonly present terms are surface (LigHPhobB/S) and protein/ligand van der Waals (4/8ExtVdW). Then, hydrogen bond (GoodHBond/GOLDHBOND), internal clash terms (such as 6/12InVdW or GOLDITORS), and some coulombic/solvation (Coul 4R) are incorporated.

Direct comparison of the terms' values and proportion is meaningless here, since they are composed of different terms. For example, comparing functions trained on set 27-A and B shows that the two functions share LigHPhobS and 4/8ExtVdW, but with different values; the hydrogen bond term is different, though, even if GoodHBond and GOLDHBOND have similar strongly negative coefficients; finally, the fourth term is different for the two functions (GOLDITORS versus the 6/12InVdW). Indeed these two

functions are quite different, albeit they express the same kind of interactions. However, strong similarities can be traced between the functions trained on set 27-B and C: most terms are shared, with very similar coefficients, and the highly correlated[36] LigHphobB and LigHPhobS play similar roles in each function – coefficients for these two terms are similar too. On the basis of this, we would expect functions parameterised on set 2 to perform well on set 3 too, and *vice versa* (hint: look at Table 7-6).

Set	Term 1	Term 2	Term 3	Term 4
27-A	GoodHBond *-99.65*	LigHPhobS *2.81*	4/8ExtVdW *4.91*	GOLDITORS *6.20*
27-B	GOLDHBOND *-96.51*	LigHPhobS *15.78*	4/8ExtVdW *20.88*	6/12InVdW *-0.03*
27-C	GOLDHBOND *-97.14*	LigHPhobB *11.13*	4/8ExtVdW *20.99*	6/12InVdW *-0.01*
27-D	GoodHBond *-77.67*	4R Coul *62.73*	LigHPhobS *2.84*	4/8ExtVdW *4.92*

Table 7-5: The best 4-term functions for each set – using COLOs as references. Different colours indicate different interaction classes[JW63]

Let us now properly consider the ranking performance of the best combination resulting from parameterisation on set 27-A. When running on its own parameterising set, it obtains the remarkable result of ranking the COLOs on average at 1.58% of the ranked list. When ran on other sets, we can expect this performance to decrease. But how much degradation do we actually observe? When ran on set 27-B, it scores 5.96%, worsening the performance almost by a factor of 4. Runs on set 27-C and D fare slightly better, ranking the COLOs at 4.99% and 3.72%. Results for all possible pairs training/test set are presented in short in Table 7-6. The value in ***bold+italic*** is the results for the function parameterised on set 27-C, when tested on set 27-B. Along the diagonal, in *italic*, the performance of the function against its own parameterisation set.

27 proteins: Trained On → Tested On ↓	Set 27-A	Set 27-B	Set 27-C	Set 27-D
Set 27-A	*1.58%*	5.96%	4.99%	3.72%
Set 27-B	7.49%	*2.24%*	*6.55%*	7.86%
Set 27-C	4.63%	6.11%	*4.01%*	6.54%
Set 27-D	3.67%	7.25%	7.08%	*2.91%*

Table 7-6: Results for the functions detailed in Table 7-5 on the various sets – COLOs as references.

It has been noted in the previous paragraph that the best functions for sets 27-B and C look quite similar, so that one would expect them to keep their good performance when passing from one set to the other. Unfortunately, this does not seem to happen. Probably, the two sets are different enough to mess up the balance of the terms, but most of all, we must remember that the hydrophobic surface matching terms did differ, albeit with similar coefficients. Still, the worsening of performances is less severe than the one observed when the function trained on set 27-A is applied on the same sets: the function trained on set 27-B ranks COLOs at 2.24% of the list on this same test, but is able to rank set 27-C's complexes only to an average of 6.11% in the list. Conversely, 27-C's function ranks its own complexes at 4%, and set 27-B's complexes at 6.55%.

The best performers trained on set 27-D tend instead to keep their good ranking powers on set A too. This may be due to the fact that 27-D's function, like 27-C's, did not perform particularly well, to start with, ranking its COLOs only at 2.91% on average. The performance of the first fifteen combinations trained on set 27-D (data not shown - only the best is shown in the Table 7-6), ranged in average ranking from 2.9 to 3%, only decrease slightly when they are applied on set 27-A, averaging at 3.5%. The same combination of terms which compose the best performer trained on set 27-A is present as 13th and 15th best performing function trained on set 27-D, albeit with different coefficients (the combination 1-18-19-28 parameterised on set 27-D lies at 2° of angular distance from that same function, but trained on set 27-A). Set D is peculiar in other ways too. Best performing functions are quite diverse from each other, showing more variability in the choice of terms. Particularly, Coulombic interactions, which usually do not perform very well, are present.

In Figure 7-3 the performances of the functions parameterised on 27-A (COLO) with various number of terms (from 2 to 5) against all 27 proteins sets are presented, using the COLO as references. Again, it is possible to see here the same phenomenon anticipated in Figure 7-2. Even if the ranking performance on the training set (blue line) gets better (lower) with increasing number of terms, the actual performances on independent sets is mostly unchanged after the fourth term is added. For some cases (set 27-B) performances of the five terms function actually gets worse, a clear sign of overfitting of the function to its own training set. On the basis of this being a common phenomenon, we will limit discussion to two terms combinations, which offer a quick way to detect the main players of binding, and four terms functions, which ideally represent the optimum in terms of performances.

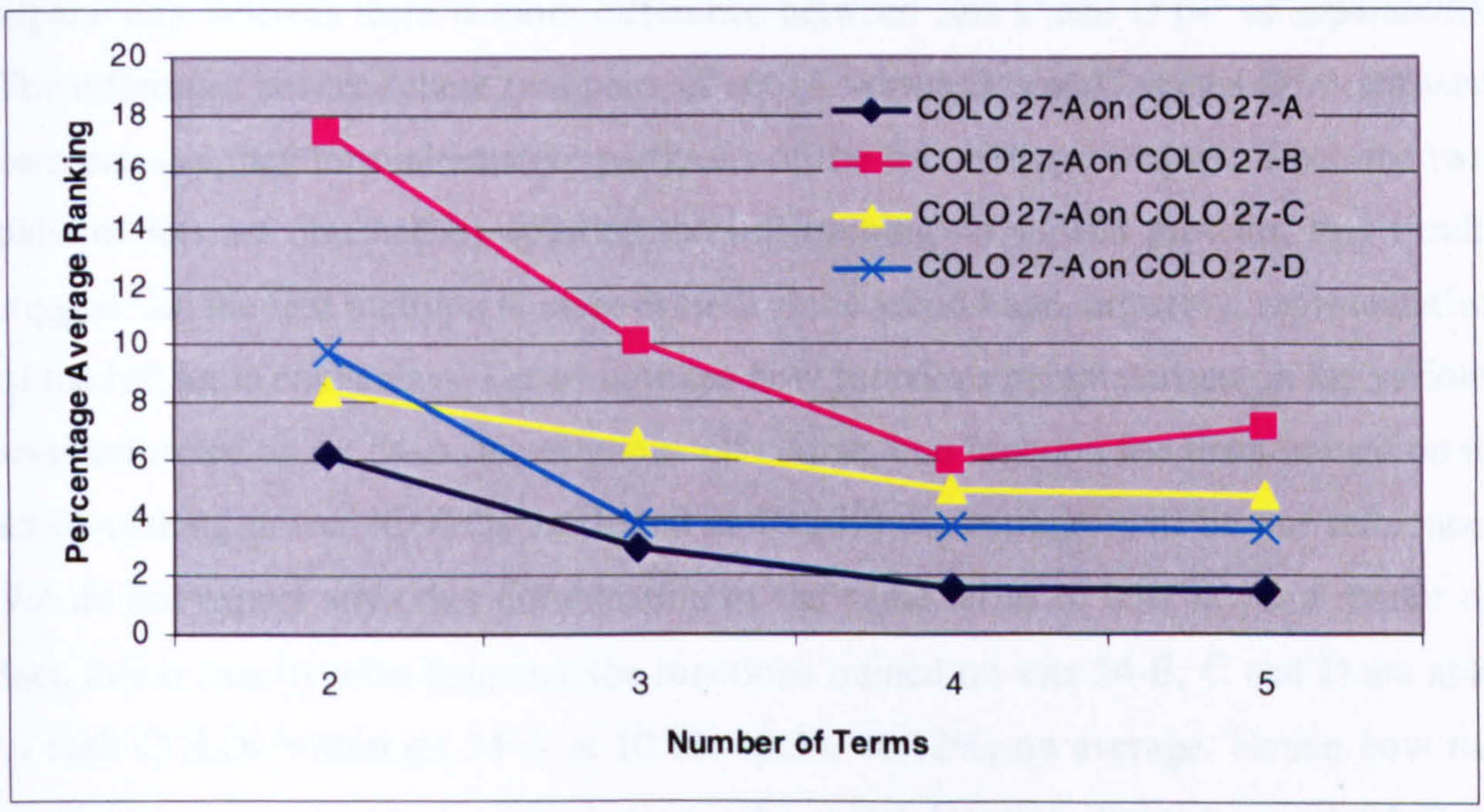


Figure 7-3: Performance on the four 27 proteins COLO sets of the best function parameterised on set 27-A (COLO) with various number of terms.

However, it is possible to conclude that, even with functions parameterised only on 27 complexes, it is reasonable and realistic to expect an ability to rank unknown complexes within the 6% of a decoy list, and this certainly is a valuable result.

7.2.2 Testing the 54p

Let us now consider how the functions trained on the four different 54-protein sets performed when used to score the others. It has been said in the previous section (7.1.3) that these functions were much more similar to each other than those trained from the 27p sets. This leads us to anticipate that they will be more able to keep their performance on sets against which they have not been explicitly trained. By looking

first at 2-term functions it can be seen that this is, in fact, the case. The best 2-term functions parameterised against the four 54 protein sets are shown in Table 7-7:

54 proteins	*LigHPhobB*	*4/8ExtVdW*	SCORERANK
Set 54-A	*61.00*	*79.42*	10.10%
Set 54-B	*60.07*	*79.95*	10.13%
Set 54-C	*64.74*	*76.22*	10.33%
Set 54-D	*58.91*	*80.81*	10.12%

Table 7-7: Results for functions parameterised against 54 complexes – COLOs as references.

It is interesting to notice that sets 54-A and B show very similar functions (40' of arc of separation), whereas there is more difference between sets C and D (4° of separation). The difference between these two pairs of set (A versus B, and C versus D) is stressed here because they form alternative partitions of the full 108 proteins set. Since the two pairs of sets are obtained by splitting the full training set of 108 proteins, this would suggest that the first partition is more even than the second and, arguably, representative of the full set in equal ways. Let us now see how functions parameterised on the various sets performed on set 54-A, for example. Of course, one function has been trained on it, so its ranking power (COLOs are found at 10.10% on average) will be our reference: We do not expect any other combination of the same terms to beat it. As a matter of fact, this is exactly what happens: the functions trained on sets 54-B, C and D are able to rank COLOs within set 54-A at 10.13, 10.33, 10.12%, on average. Notice how the performances are however very similar – the worst function, trained on set 54-C, is barely a quarter of percent worse, even if the variation of its coefficients compared to the reference is around 5%. This suggests the presence of a minimum in a shallow valley in the parameter space, with slowly raising slopes. Moving on to four-term combinations, to allow a direct comparison with the 27-protein sets discussed before, the results presented in Table 7-8 and Table 7-9 are obtained.

Set	GoodHBond	LigHPhobS	4/8ExtVdW	GOLDITORS
Set 54-A	*-99.69*	*2.74*	*3.59*	*6.51*
Set 54-B	*-96.24*	*9.84*	*17.83*	*17.95*
Set 54-C	*-99.82*	*2.78*	*3.76*	*3.69*
Set 54-D	*-96.24*	*8.66*	*15.08*	*20.86*

Table 7-8: Best 4-term [JW64]functions for each 54-protein set – using COLOs as references. As before, different colours flag different classes.

The first thing to note when comparing Table 7-8 with results for the 27-protein subsets, in Table 7-6, is that all four sets returned a combination of the same four terms as best performer: as highlighted before, this is due to the fact that this compounds are more homogeneous compared to the 27-complex ones. Second, although the 'indigenous' (*i.e.*, functions which have been trained against that set) functions are less effective than before, due to the fact that more complexes have to be optimised, the functions turn out to be more transferable on other sets (Table 7-9 gives the ranking performances).

Whereas 4-term functions parameterised on 27 proteins were scoring at 1.6% on their own set, and at 6% on others, equivalent functions parameterised on sets twice as big now they fare only 3% their own, but 4.5% on others.

54 proteins: ***Trained On →*** ***Tested On ↓***	**Set 54-A**	**Set 54-B**	**Set 54-C**	**Set 54-D**
Set 54-A	***3.09%***	4.60%	2.49%	5.20%
Set 54-B	3.79%	***4.15%***	3.06%	4.89%
Set 54-C	3.36%	4.64%	***2.29%***	5.71%
Set 54-D	3.44%	4.24%	2.86%	***4.82%***

Table 7-9: Results for the functions detailed in Table 7-8 on the various sets – using COLOs as references. In *bold+italic*, the performance on the test against which each function has been parameterised.

Compared to the functions trained on the four 27-protein sets, performances on their own training set have worsened, since there are more complexes between which a compromise must be struck. At the same time, performances on the test set have improved because the functions themselves are of more general applicability.

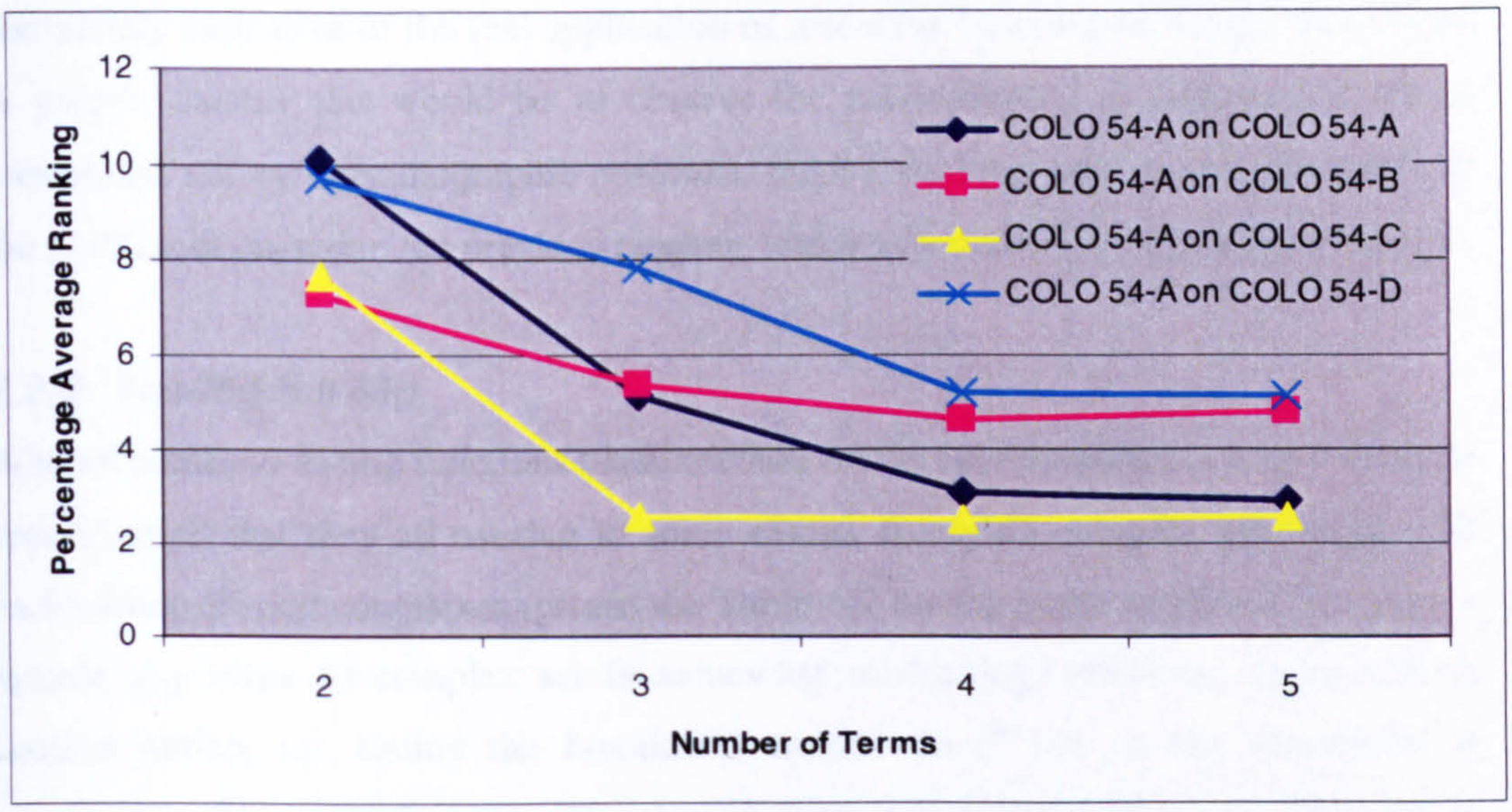


Figure 7-4: Performances on function parameterised on COLO 54-A on its own set and other 54-protein sets.

This is exemplified by the fact that in Figure 7-4, showing performances of best 2 to 5-term functions trained against set 54-A, performances are very similar when these functions are applied to 54-B, with which 54-A does not share any complexes. To use an analogy, since the little garden they are grown in is now larger, it is more representative of the world outside, so they can cope with the latter when released in the wild. These functions can even manage to have better results on sets other than their own, although always lesser than functions explicitly parameterised on that set. For example, the best function trained on set 54-A ranks COLOs on the same set at 3.09%. But it can rank set C's COLOs at 2.49%. That is to say, it performs better on set C than on its own training set. However, functions explicitly trained on set C can achieve a ranking as good as 2.29%, therefore better still (and possibly optimal). However, we must remember that whereas set A and B are mutually exclusive, set C (and D) partially overlap with each of them. It is not a surprise, then, that as shown in Table 4-5 set C has a larger amount of overlap with set A (32 complexes in common), compared to set D (only 22 complexes shared).

All this is a good indication that our functions will be able to perform their task on protein/ligand poses never seen before, as long as they are parameterised on sets large and representative enough. For these functions the expected level of performance on retrieving new, previously unknown actives is around 5%. This is however not

completely indicative of the real application of a scoring function in Virtual Screening. A proper, harsher test would be to observe the performances in ranking references constituted not by crystallographic positions, but by docking-generated poses, such as the *Goldest* discussed in the previous chapter, which will make a re-appearance later on.

7.2.3 Testing the 81p

When it comes to testing functions parameterised on the four 81-proteins sets, it must be kept in mind that they all overlap to some extent. Every 81-complex set shares with each sibling 60-odd complexes (please see Table 4-7 for the exact numbers). So, testing against any other 81-complex set is somewhat misleading. However, there also is another option, *i.e.* testing the function(s) trained on i^{th} set on the remainder of complexes, those which were not selected from the full training set. This set is composed of 108-81 = 27 complexes, labelled 27-E to H. Both kinds of tests are briefly discussed in this section.

These four 81-complex sets overlap each other for around sixty entries out of 81 (roughly 75% as one would expect). Parameterisation results have been presented previously in this same chapter, section 7.1.4. The best two-term function trained on the first 81-complex set (81-A, following the usual convention) is able to rank its own COLOs at 9.20% on average (Table 7-10). Performance turns out to be very similar for the other three sets, where the very same function ranks COLOs at 9.67% (Set B and C) and 10.96% (Set D). When the same function is put on trial on the 27-complex complementary set, it shows a remarkable level of performance, ranking COLOs at 8%. So, performance seems to be actually better on the test set than it is on the training set. This may be happening by chance, because of a fortunate distribution of complexes in the pair training/test sets: still, it is an indication that our functions are able to perform the task for which they have been built.

Full results for 2- and 4-term functions (only the best performers for every training set) are shown in the next tables (Table 7-11 and Table 7-13).[JW66]

81 proteins	*LigHPhobB*	*4/8ExtVdW*	SCORERANK
Set 81-A	*45.29*	*89.16*	9.20%
Set 81-B	*59.00*	*80.74*	9.21%
Set 81-C	*59.01*	*80.73*	9.56%
Set 81-D	*59.61*	*80.29*	10.27%

Table 7-10: Best 2-terms functions for each 81-protein set – using COLOs as references.

81 proteins: Trained On → Tested On ↓	Set 81-A	Set 81-B	Set 81-C	Set 81-D	Complement
Set 81-A	***9.20%***	9.67%	9.67%	10.96%	27-E 8.00%
Set 81-B	9.21%	***9.21%***	9.56%	10.28%	27-F 7.17%
Set 81-C	9.21%	9.21%	***9.56%***	10.28%	27-G 6.14%
Set 81-D	9.22%	9.23%	9.56%	***10.27%***	27-H 3.95%

Table 7-11:Two term functions for 81-p sets. The last column shows the performance of the function trained against set *i* on its own complement, *i.e.* the remainder of the full 108 complex training set after selection.

Passing to the performances for 4-term functions (see Table 7-12 and Table 7-13), one sees much the same trend observed in the previous section: functions trained on set 81-A attain rankings of 4.14% within it, and this is conserved on average along the other partially overlapping 81p sets. On its 27p complement, again, performance improves to 2.89%. This confirms that the phenomenon is due to the set composition, and not a quirk in the functions considered previously.

Term	GoodHBond	LigHPhobS	4/8ExtVdW	GOLDITORS
Set 1	*-99.77*	*2.64*	*3.86*	*4.88*
Set 2	*-96.99*	*7.44*	*13.61*	*18.79*
Set 3	*-99.77*	*2.64*	*4.01*	*4.84*
Set 4	*-99.68*	*2.76*	*3.65*	*6.61*

Table 7-12: Best 4-terms functions for each 81-protein set. Again, each set independently chooses the same 4 terms.

Fo[JW67]r functions trained on any other set the same holds true, performance can be easily transferred to other sets. Also, all functions get a performance boost when tested

against the complement of their training set. Particularly impressive is the performance improvement for functions trained on set 81-D: when they are applied to set D's complement (27-H), both the 2- and 4-term best combinations of coefficients manage to double their ranking power.

81 proteins: ***Trained On →*** ***Tested On ↓***	**Set** 81-**A**	**Set** 81-**B**	**Set** 81-C	**Set** 81-D	**Complement**
Set 81-**A**	***4.14%***	4.04%	3.82%	4.51%	27-E 2.89%
Set 81-**B**	4.30%	***3.87%***	4.25%	4.76%	27-F 3.91%
Set 81-**C**	4.15%	4.04%	***3.80%***	4.53%	27-G 3.89%
Set 81-**D**	4.18%	4.00%	3.87%	***4.47%***	27-H 1.97%

Table 7-13: Ranking performances for 4-term combinations trained on the 81-protein sets. The last column (Complement) shows their performances against the complement sets.

7.3 Trained versus COLO, tested versus Goldest

When performing a scoring function parameterisation, the objective is to improve our ability to select, during structure-based virtual screening (SVS), near-crystal poses for molecules which are likely to be active in the given receptor. In that case, one will have no information whatsoever on their actual crystal structure, unless, that is, the answer is already known, and the (simulated) SVS is performed just to confirm that our method works. In section 6.4, a procedure was presented which generated structures for the reference ligand by re-docking them inside the active site. Out of all poses generated for every complex, only two were selected: the first was the closest (in RMSD sense) to the COLO; substituting the scoring terms of these poses to the corresponding COLOs in the GA input file generated the new training set of actives subsequently called *Closest*. The second pose to be selected was the best pose suggested by GOLD according to its own scoring function, GOLDSCORE, known for being able to correctly reproduce binding modes (but not so good on affinities); these reference poses were assembled with the decoys in a new GA input set called *Goldest*.

These two new *Goldest and Closest* sets are more suited to test the kind of performances we want, having the exact form which our functions will have to tackle

during their day-to-day use – that is, all poses within are generated by docking programs. They have already been used as training set, so the parameterisation detailed in the previous sections of this chapter could be repeated, after splitting the *Goldest* set in 27, 54 and 81 protein sets. But they can also be used directly as test set for the functions already trained using COLOs as references.

In this section, however, only results relative to the *Goldest* set will be presented, for reason of space, but also because they have been generated in a more rigorous way compared to the *Closest* poses. The latter were sometimes generated by modifying *ad hoc* GOLD's parameters until they gave an output within 2.5 Å (all atoms) RMSD of the COLO.

Let us see, then, how a function trained on the 81 protein set number 81-A performs on the remainder of 27 proteins left out of the selection, labelled "***complement***". The whole progression of terms is not presented here, since no new parameterisation is being performed: these are the function introduced in section 7.2, just applied to a different set: a full analysis was performed, but only some example cases will be shown, whenever they present phenomena worth of note.

Once again, the first thing to be investigated will be how well the functions fare when selecting COLOs, and how efficient are they at selecting the *Goldest* reference poses.

Trained on	Tested On	Reference	
		COLO	Goldest
Set 81-A	Set 27-E	*2.89%*	*16.22%*
Set 81-B	Set 27-F	*3.91%*	*11.87%*
Set 81-C	Set 27-G	*3.89%*	*5.59%*
Set 81-D	Set 27-H	*1.97%*	*9.36%*

Table 7-14: 4-term [JW68]functions trained against the various 81-protein sets: performances against complement set using COLOs and *Goldest* poses.

From what is shown in the Table 7-14, the performance hit taken when using a function trained on 81-A, B and D's COLOs to recover actually docked poses is severe, and the

ability to prioritise the rank of the references is sometimes worsened by a factor 6. Only the function trained on 81-C's complexes seems to perform adequately. Why this decrease in performances? First, it is to be expected. *Goldest* poses can be very different from the COLOs, and they can require quite a nudge to the coefficients for the best terms to optimise their ranking: from what was shown in the previous chapter, it may even be necessary to change the preferred terms to describe the ligand-protein interactions. It may just be that set number 81-C contains ligands whose COLO was closer to GOLDSCORE's optimum.

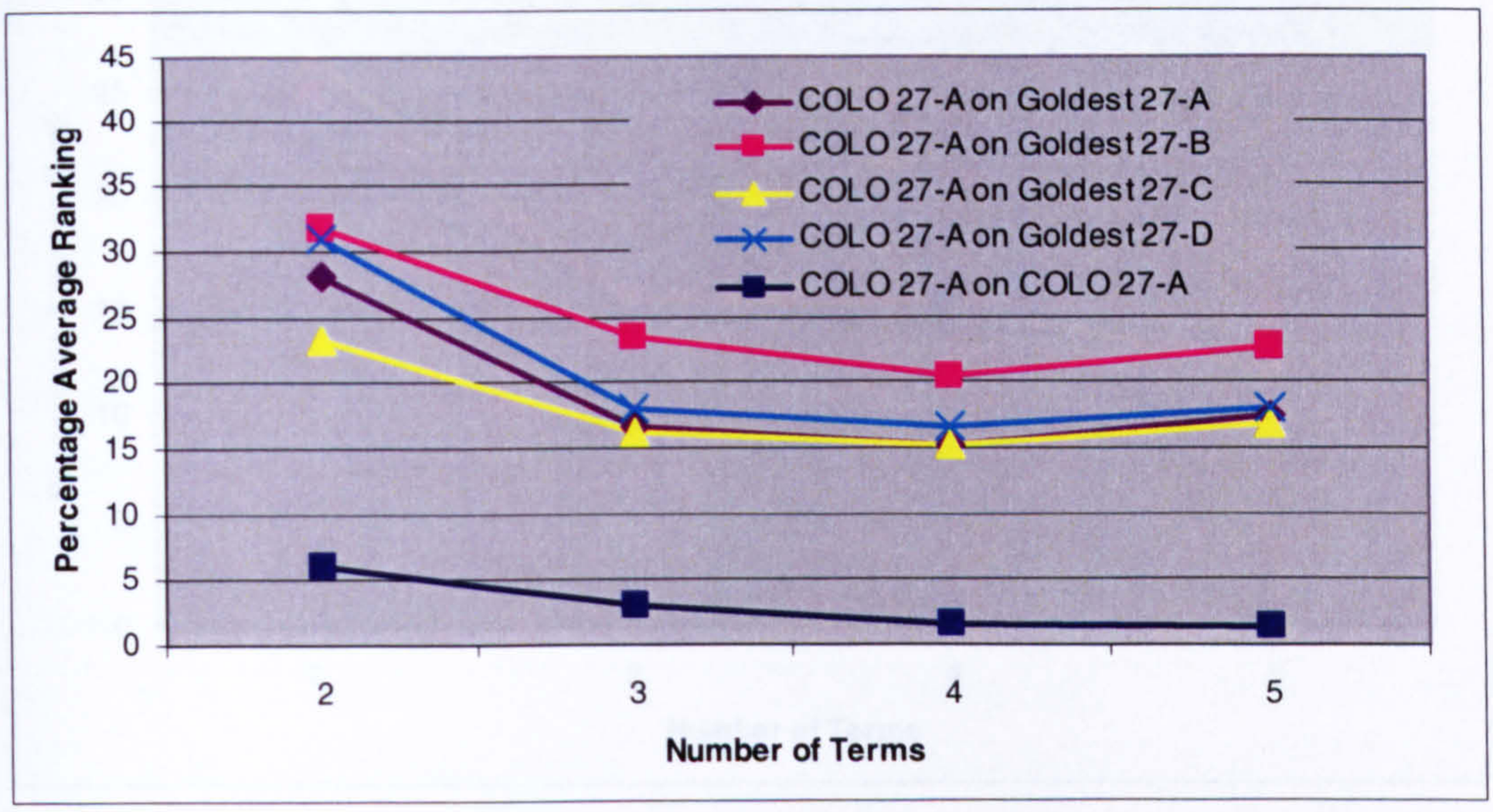


Figure 7-5: Performance on the four 27 protein Goldest sets of the best function parameterised on set 27-A (COLO) with various number of terms.

As a further example, Figure 7-5 shows the performance on Goldest sets 27-A to D of the best 2 to 5 term functions parameterised on COLO 27-A: the self-performance on this set is also shown for comparison. Again, the performance hit taken is severe, with docked references recovered with much lower efficiency by the functions. And again, we see that going over four terms is actually detrimental to the ranking power, further strengthening our idea that four is a magic number for terms, at least with the setup of terms chosen for this work. At the level of four terms, our function is extracting all possible information from the structure, and what's left is mainly noise, as shown by the fact that keeping the reference ligand but changing the pose (from COLO to *Goldest*) performance of five-terms functions is worst than their predecessors – the fifth term is describing the noise associated with the COLO poses, not present in the *Goldest*'s.

Finally, Figure 7-6 highlights the same decay in performances due to the strong difference between crystallographic and re-docked poses. However, it also shows how the performance obtained is roughly the same regardless of the set to which the functions are applied, be it the training or the test set. Also, compared with performances shown in Figure 7-5, those in Figure 7-6 are slightly better, once again showing how (predictably) a function trained on a larger, more diverse set will have greater generality, and suffer less from bias.

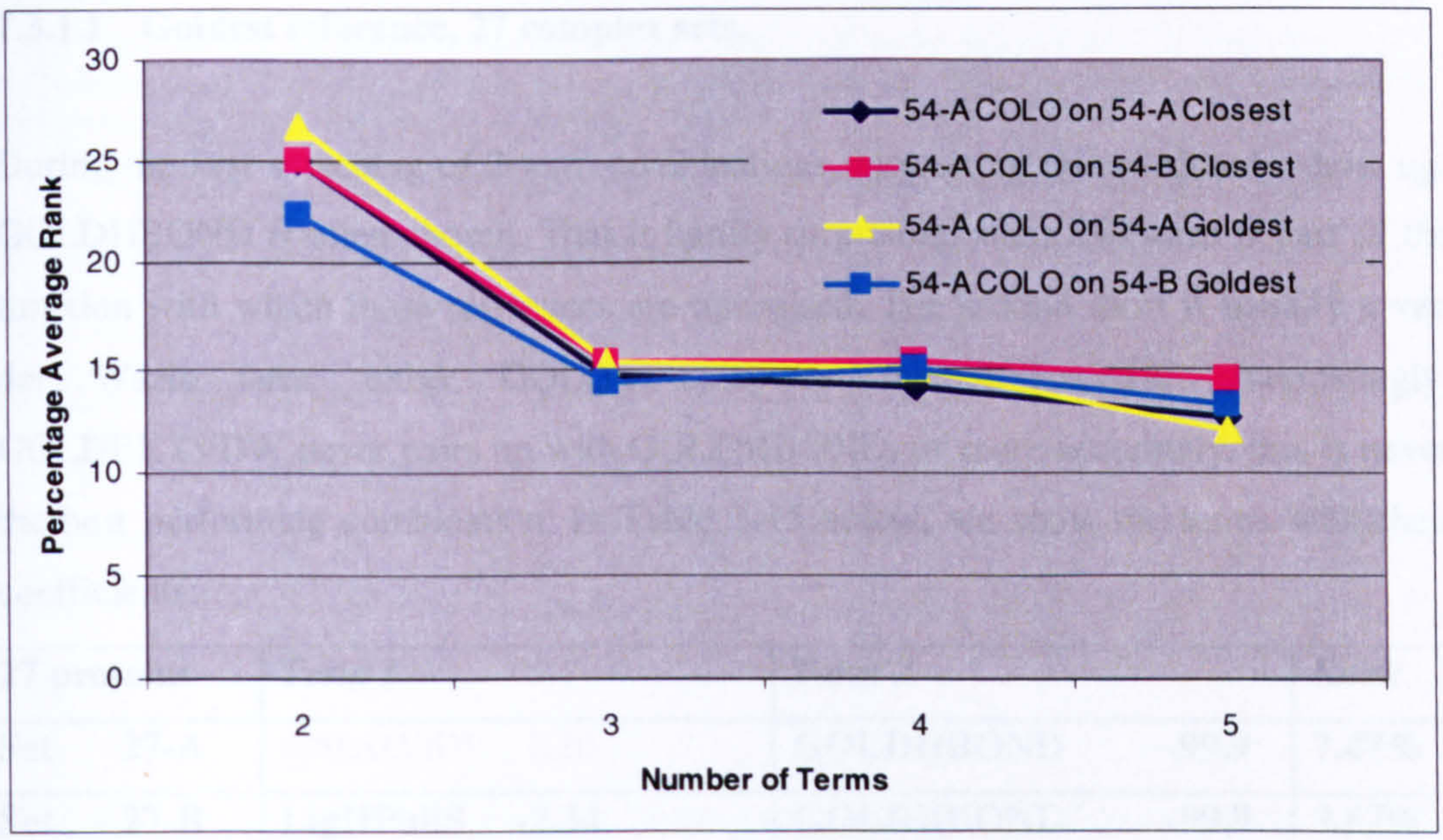


Figure 7-6: Ranking performances of functions parameterised on set 54-A using COLO, when applied to Closest and Goldest sets 54-A and B.

But let us now move the subject onto functions properly trained on and tested against the *Goldest* reference poses. They need to be trained against one set, and tested against the complement, non overlapping sets, as was shown for the COLO-trained function in the previous sections. If the Goldest-trained functions were to perform better than the COLO-parameterised functions, when pitted against similar training/test cases, it would be good news. This would in fact imply that one can efficiently parameterise scoring functions without the need for knowing bioactive conformations – as long as one is interested only in recovering actives, and not in getting reliable estimates of binding free energy.

7.3.1 Training and testing against Goldest

In this section, the results relative to parameterisation on reduced *Goldest* training subsets will be presented and discussed. They should, from GOLD's point of view, be the best possible pose(s). The subset division is exactly the same as detailed in the previous sections: We will, for the sake of brevity, present results only for the 27- and 81-protein sets, glossing over the half-sized 54 complex sets.

7.3.1.1 Goldest reference, 27 complex sets.

During the first screening of 2-term combinations, interesting things already show up: GOLDHBOND is often chosen. That is hardly surprising, since this term is part of the function with which these references are optimised. The second term is usually a van der Waals term, either GOLD's own, or the 4/8ExtVdW. Interestingly, GOLDEXTVDW never pairs up with GOLDHBOND, or more accurately, this is never the best performing combination. In Table 7-15 below, we show the terms with their coefficients:

27 proteins	*Term 1*		*Term 2*		*Rank*
Set 27-A	4/8ExtVdW	2.20	GOLDHBOND	-99.9	7.47%
Set 27-B	LigHPhilS	-2.34	GOLDHBOND	-99.9	3.67%
Set 27-C	4/8ExtVdW	2.47	GOLDHBOND	-99.9	5.57%
Set 27-D	4/8IntVdW	0.32	GOLDHBOND	-99.9	8.21%

Table 7-15: Best 2-term functions for each 27-protein set, trained using *Goldest* as references.

In contrast to the functions parameterised on the COLOs, now sets B and D have chosen a different pair of terms. More specifically, LigHPhilS in set 27-B and 4/8IntVdW in 27-D; both these terms are paired with GOLDHBOND.

Usual considerations apply for the terms' signs: GOLDHBOND, with its strong negative coefficient, gives the most important favourable contribution to binding, whereas 4/8ExtVdW and 4/8IntVdW with their positive signs acts as an internal and inter-molecular clash term. LigHPhilS turns out to provide a favourable contribution – the physical explanation for this is unclear.

27 proteins: Trained On → Tested On ↓	Set 27-A	Set 27-B	Set 27-C	Set 27-D
Set 27-A	***7.47%***	4.38%	5.67%	11.34%
Set 27-B	11.64%	***3.67%***	14.03%	14.32%
Set 27-C	7.54%	4.76%	***5.57%***	11.52%
Set 27-D	9.26%	5.71%	12.47%	***8.21%***

Table 7-16: Results for the functions detailed in Table 7-15 on the various sets. In *Bold+Italic*, the performance on the test against which every function has been parameterised (intrinsic performances of the functions shown in Table 7-15.

It is, however, difficult to underestimate GOLDHBOND's potent contribution to binding in all cases, being present in the best combination for each set, always with a -99.9 value. In fact, what is strange is that the 4/8ExtVdW term is preferred to GOLDEXTVDW, and that similarly the 4/8IntVdW manages to outperform GOLDITORS. That is most unexpected since the references are optimised with exactly these terms, so they should be advantaged in the selection. As far as ranking powers are concerned, Table 7-16 presents both training and testing results for the functions in Table 7-15.

Compared to similar results obtained training on the 'COLO' references, performance on the same set (see values on the diagonal versus the third column of Table 7-2) can be better (27-B), equivalent (27-C) or worse (27-A and D). That they may be worse is expected, as the true actives' poses generated by GOLD using GOLDSCORE would be expected to be more difficult to select from a host of similarly generated poses, more difficult, that is, compared to selecting COLOs from the same set of supposedly inactive decoys. That the *Goldest*-parameterised functions may be better is, following the same reasoning, harder to explain. One reason may be that the greater similarity between different reference poses produced by the same program may play to our advantage, making it easier for a well balanced function to bring them all up at once.

The greater similarity between the reference and the decoys in the *Goldest* sets is also reflected in the terms selected. The importance of surface matching measures is rather diminished, and the focus has moved to the hydrogen-bond network that the molecules

will be able to create within the pocket. Still, performance decreases[JW71], in both the training and the test sets.

Stepping up to the 4-term combinations, the same trend shows up (Table 7-17):

Set 27-A	**GoodHBond** *-87.4672*	**LigHPhobS** *1.8624*	**GOLDEXVdW** *-48.4340*	**4/8IntVdW** *0.4078*
Set 27-B	**GOLDHBOND** *-66.5326*	**BadHBond** *74.6330*	**4/8ExtVdW** *0.5711*	**LigHPhilS** *-1.7347*
Set 27-C	**GOLDHBOND** *-95.7263*	**UnsDonor** *28.7658*	**4/8ExtVdW** *2.7688*	**LigHPhilS** *1.1585*
Set 27-D	**GOLDHBOND** *-93.2674*	**BadHBond** *36.0404*	**4/8ExtVdW** *1.4820*	**4/8IntVdW** *0.3090*

Table 7-17: Best 4-term functions parameterised on the four 27-protein sets, using *Goldest* as references.

On set 27-A for example, compared to the COLO's results, shown in Table 7-5, GoodHBond is substituted by GOLDHBOND, and the GOLDEXTVDW takes on 4/8ExtVdW's role; In both cases a term from GOLDSCORE substitutes the equivalent *in house* term. Surprisingly enough, however, 4/8IntVdW substitutes GOLDITORS. As already said considering the 2-term combinations, this is unexpected, since GOLDEXTVDW and GOLDITORS are part of GOLDSCORE. Rationalising this different choice is difficult: the other term in the function is LigHPhobS, but it is not a driving force; 4/8IntVdW is in fact the third term to be added (results for 3-terms are not shown here); after that, LigHPhobS is added. Apparently, 4/8IntVdW really is better than GOLDITORS at describing the interactions occurring between the protein and the ligand in set 27-A, or, at the very least, it is a harsher judge when discriminating the reference (the *Goldest* pose) from the decoys. It stands to be confirmed whether this choice will bring good results when testing on other sets. The panorama of best functions for the other sets is quite varied, and includes for the first time penalty hydrogen-bond methods, such as unsatisfied donor count (UnsDonor) and the donor-donor/acceptor-acceptor clash count (BadHBond). After this brief highlight of the terms in each function, let us have a look at their performances.

Once again, as shown in Table 7-18, transferability of performances is less than ideal, and even on their own training set the functions do not perform as well as the COLO-parameterised ones. Even increasing the number of terms, performances do not improve significantly. Unfortunately, there does not seem to be a way out of this *impasse*, most likely due to the greater difficulty in distinguishing the *Goldest* poses from the decoys, compared to the COLOs.

27 proteins: ***Trained On →*** ***Tested On ↓***	**Set 27-A**	**Set 27-B**	**Set 27-C**	**Set 27-D**
Set 27-A	***5.24%***	8.84%	14.50%	8.90%
Set 27-B	9.78%	***2.48%***	10.71%	11.29%
Set 27-C	9.59%	7.11%	***4.31%***	11.72%
Set 27-D	7.04%	4.71%	7.47%	***6.80%***

Table 7-18: Results for the functions detailed in Table 7-17 on the various sets. In *Bold+Italic*, the performance on the test against which every function has been parameterised.

Coming back to the discussion regarding the preference towards terms able to play the same role as GOLDSCORE's original terms, such as the 4/8ExtVdW instead of GOLDEXTVDW, or 4/8IntVdW rather than GOLDITORS, we notice that test results are consistent with the training. Other things being equal, functions containing GOLDSCORE's terms other than GOLDHBOND do fare worse than functions containing their *in house* analogues. As an example, we show how the best 2-term scoring function trained on set 27-A performs against its own and the other sets in Table 7-19. In all cases, the combination containing the 4/8ExtVdW term outperforms (has lower percentage average ranking for the reference) the one containing GOLDEXTVDW, even if in some cases the difference is marginal.

Combo		Performances on			
1st Term	2nd Term	Set 27-A	Set 27-B	Set 27-C	Set 27-D
GOLDHBOND ***-99.9***	**4/8ExtVdW** ***2.2***	***7.47%***	4.38%	11.34%	3.67%
GOLDHBOND ***-96.7***	**GOLDEXVdW** ***-25.4***	***7.67%***	5.35%	14.53%	3.89%

Table 7-19: Percentage average ranking on all *Goldest* 27-protein sets of two 2-term functions, respectively best and second best from training set 27-A.

In Table 7-19, values highlighted in ***Bold+Italic*** correspond to the performances of the functions on their own training set. 7.67% means that the function ranks the *Goldest* reference poses at that point on the list, on average. Or alternatively, in a VS experiment, it would be able to recover half the actives after scanning 7.67% of the ranked list.

However, the actual ranks vary depending on the particular complex. Although performances on the training set 27-A are quite similar, the two functions can behave quite differently on the test sets, particularly sets B and C. However, it can be noted that the function containing 4/8ExtVdW always out-performs the one with GOLDEXTVDW, confirming our previous discussion. Again, the fact that GOLDSCORE is taken apart into its component, and only some are selected by th GA to assemble a better function seems to indicate that the choice of a function's terms is not trivial, and is highly dependent on the task at hand.

7.4 Conclusions

In this chapter, the same parameterising procedure has been applied to various training sets with changing size, composition, or different reference poses. The underlying aim was to explore the robustness of the parameterisation to changes in all these parameters.

From this experience, some useful conclusions can be drawn: first, changing the size of the training set does affect the goodness of fitting and predictivity, but good functions do not need a huge number of complexes to be trained. Good results can be obtained training your functions on a set composed of as little as thirty complexes. However, since the sensitivity to its composition is significant, the training set should be as diverse as possible. Also, one should avoid complexes with peculiar interactions which may skew the balance of coefficients, unless this is wanted as in the case of *local* scoring functions, to be applied only to a single family of proteins. Second, the terms chosen by the GA as optimal tends to be consistent across quite a wide range of training set sizes and compositions, with shape complementarity (various van der Waals), hydrophobicity (via hydrophobic buried surfaces) and hydrogen bonds being the most important. On functions trained on *Goldest* poses, instead, the hydrophobicity term seem to lose its usefulness, and is replaced by measures of internal strain of the ligand,

but most often by hydrogen bond penalties. This is consistent with what was seen in the previous chapter on the full training set, and we can therefore be sure that it is not a fluke due to the training set's size.

Functions trained on COLOs, albeit highly efficient at recovering the COLOs themselves, do not perform particularly well on reference poses obtained from docking programs (*e.g. Goldest*). Unfortunately, this is exactly the kind of application for which the functions are being developed. An explanation for this may be that the GA ends up over-describing the details of interactions, picking up, along with the information contained in the crystal poses, noise, namely structural tweaks imposed by different refinement methods. Therefore, functions parameterized on such references may not be an ideal choice to maximize enrichment during SVS.

On the other hand, *Goldest* parameterized functions seem to be more robust to small movement of the reference structures. Therefore, it may not be strictly necessary to have crystal structures to parameterize a function: a docking program able to reproduce the bioactive conformations may be all that you need.

Finally, there seems to be space for assembling a better function based not only on newly implemented terms, but on known and tested terms cherry-picked from the panorama of available scoring function. Such an exercise will be presented in the next chapter.

8 Further parameterisation and testing

8.1 Introduction

In this chapter, some different parameterisation and testing results will be presented. Until now, the functions presented were composed of a mix and match of newly devised terms. However, any pharmaceutical company has its own programs, either commercial or developed *in house*. The aim of this chapter is to show that the parameterisation method presented is applicable to all functions, as long as one can recover detailed information for it, and is prepared to generate enough docking data to re-parameterise it.

This chapter is composed of two different parts, different in make-up but equally important. First, a simple parameterisation exercise of a commercially available set of scoring functions: the new version of GOLD [40, 53] used for the generation of docking poses also included a CHEMSCORE [51, 52] implementation. It was possible to use CHEMSCORE to either drive the docking, or to rescore already generated poses. Unfortunately, owing to time restrictions it was not possible to perform a full re-docking using this second function; the rescoring of docked poses, however, would not take more than a few hours. Therefore, all docked poses for the full training set were rescored using CHEMSCORE. The resulting score and terms composing it were collated with the corresponding GOLDSCORE terms. This created a new input file for the genetic algorithm, subsequently used to parameterise new functions.

The second, shorter part of the chapter is devoted to presenting functions trained on those single-family training sets whose creation was described in section 4.3. In this case, the small number of proteins in each family did not allow for a proper test on independent complexes to be performed. Nonetheless, results are discussed since insight can be gained on how different complexes prefer different balances of terms.

8.2 GOLDSCORE+CHEMSCORE functions

As previously stated, the ligand poses were generated using GOLD, with GOLDSCORE as the driving function for docking. The CHEMSCORE rescoring was performed on all poses, plus the crystallographic reference ligands, and the *Goldest* poses. Once the new GA input files were assembled, they contained 11 terms: GOLDSCORE, its own three terms, CHEMSCORE, and its five (six with the ΔG_0 constant) terms. By doing so, a newly parameterized function could, as in the previous exercises, pick up a full scoring function (*i.e.* GOLDSCORE) and complement it with terms taken from others, or just create its own mix.

A mix of two full functions was not allowed in this exercise, since some preliminary tests showed there would be no advantage in doing so. Indeed, as shown in Table 8-1, a three-fold improvement is obtained when creating a weighted consensus between the two, but still the performance is below that obtained if the two functions are re-parameterised term-by-term (three- to six-fold improvement).

Score	Before Retraining	After Retraining
GOLDSCORE (GS)	48.6%	15.5%
CHEMSCORE (CS)	10.9%	1.4%
GS+CS	29.61%	10.3%

Table 8-1: Performances upon re-parameterisation of scoring functions term-by-term: the value "Before Retraining" for GS+CS assumes an equal weight for both – performances are expressed as percentage average rankings of reference ligand poses (COLOs).

Table 8-1 shows the performances (expressed as Percentage Average Ranking) of GOLSCORE and CHEMSCORE (as implemented in GOLD 2), plus their sum with a 50/50 weighting. Where the raw GOLDSCORE function only manages to put the crystallographic orientations (COLOs) at around 50% of their ranked list of decoys, reparameterising the coefficients of its three terms brings the ranking of COLOs down to 15.5%, roughly a three-fold improvement. CHEMSCORE instead, already much better with a raw ranking performance of 10.9%, after retraining of its six terms reaches an impressive 1.4%.

However, the mix of the two full scores is mediocre when the sum is taken with equal weights, but even after retraining does not improve significantly over the performance

of the best component (CHEMSCORE, in this case). This convinced us to only allow mixing between terms, or a function and some additional terms, but not between the two full functions.

The experimental/computational setup for these runs was the same previously described, with different parameterisations executed on different nodes of a Linux cluster. Results were then transferred to a local Linux workstation for analysis.

8.2.1 Results

Let us now see some of the results that came out of these parameterisations: having performed both COLO and *Goldest* training, and having tested the two outomes against the other's training set, there are four different ways of looking at things. Let us start by observing the *best* function trained on *COLO*s, and how it performed on *Goldest* poses. The first two terms selected were the hydrogen bond and the clash term of CHEMSCORE: ranking performances on the training set were quite high, with average ranking of *COLO*s at 1.6% of the sorted decoy list.

Term 1	Term 2	Tested on	AveRank
CS Hbond	CS Clash	*COLO*	1.6%
-53.56	28.62	*GOLDEST*	8.3%

Table 8-2: Performances of the best 2-term function trained on the full *COLO* set, when tested on the same and *Goldest* references.

However, the function loses lots of its predictive power when applied to retrieve the poses preferred by the GOLDSCORE scoring function, which was the objective function for the docking. Still, the percentage average rank of these poses is below 10%, with only two terms that are quickly computed. Interestingly, the second best function after the one shown above is composed of the full CHEMSCORE, minus one of its components:

Term 1	Term 2	Tested on	AveRank
CS	CS_lipo	*COLO*	2.5%
-84.48	8.31	*GOLDEST*	10.8%

Table 8-3: Performance of the second best 2-term function trained on the full *COLO* set, when tested on the same and *Goldest* references.

In fact, CHEMSCORE is taken with a 'minus' sign, whereas its component is 'added' to it, albeit with a different importance than before – this means that its contribution to CHEMSCORE is being diminished.

$$\Delta G_{binding} = \Delta G_0 + \Delta G_{hbond} \sum_{il} g_1(\Delta r) g_2(\Delta \alpha) + \\ + \Delta G_{metal} \sum_{aM} f(r_{aM}) + \Delta G_{lipo} \sum_{iL} f(r_{iL}) + \Delta G_{rot} H_{rot} \quad (8.1)$$

However, consider the CHEMSCORE equation, as given in the original paper[52]. In the following[51], the explicit values for the ΔG_x regression coefficients are given:

ΔG_0	ΔG_{hbond}	ΔG_{metal}	ΔG_{lipo}	ΔG_{rot}
-5.48	-3.34	-6.03	-0.117	2.56

The clash term was missing at the time, since it was added during the GOLD re-implementation[53]. In Murray et al.[51], the terms were modified in a Bayesian regression exercise, but the signs remained the same. So, what does a positive sign during our parameterisation tell us? The original lipo coefficient is negative, but the full function is taken with a negative coefficient. Minus-by-minus gives a plus sign, and our term turns out to be increasing the 'lipo' term's importance in our modified version of CHEMSCORE. Taking as reference the ranking performance of CHEMSCORE from Table 8-1, we can see how this tweak greatly increases its efficiency.

How about complementing CHEMSCORE with some of GOLDSCORE's terms? Against our intuition, never does a function like this show in first place. Although this did not happen when training on *COLO*s, it did when the *Goldest* poses were used as reference. Results are shown in Table 8-4:

Term 1	Term 2	Tested on	AveRank
GS HB Ext	CS	*COLO*	9.4%
-98.24	-40.34	***GOLDEST***	6.2%

Table 8-4: Performances of the best 2-term function trained on the full *Goldest* set, when tested on the *COLO*s and *Goldest* references.

The second best function shares instead the same terms of the *COLO*-trained best function (Table 8-5).

Term 1	Term 2	Tested on	AveRank
CS Hbond	CS Clash	*COLO*	2.5%
-57.29	7.22	***GOLDEST***	6.3%

Table 8-5: Performances of the[JW72] second best 2-term function trained on *Goldest* set, when tested on the *COLO*s and *Goldest* references.

The balance of terms is quite different, with the two solutions making an angle of 20° 56' in the CS Hbond / CS Clash space. In Figure 8-1, this is shown graphically - the original solutions can be anywhere within the blue circle – normalized solutions are projections of the original on the blue circle itself; the angle is computed with reference to the origin of axes. Again, performances are reasonably similar, and actually this function seems to represent a better compromise between accuracy in selecting actives both from truly determined crystallographic poses and from those which our docking programs are able to generate.

When 3-term functions are considered, one can notice the inclusion of CHEMSCORE's Metal term:

Term 1	Term 2	Term 3	Tested on	AveRank
CS Hbond	CS Metal	CS Clash	***COLO***	1.04%
-24.70	-99.22	15.69	*GOLDEST*	6.9%

Table 8-6: Best 3-term function trained on the full *COLO* set – its performances in ranking of both *COLO*s and *Goldest* reference poses are shown.

The weight of the term is very high but, once again, it must be pointed out that its range of variation is somewhat lower than other terms: for example, CS Lipo varies between 22 and 470 in subset 27-A, whereas CS Metal only goes from 0 to 4.

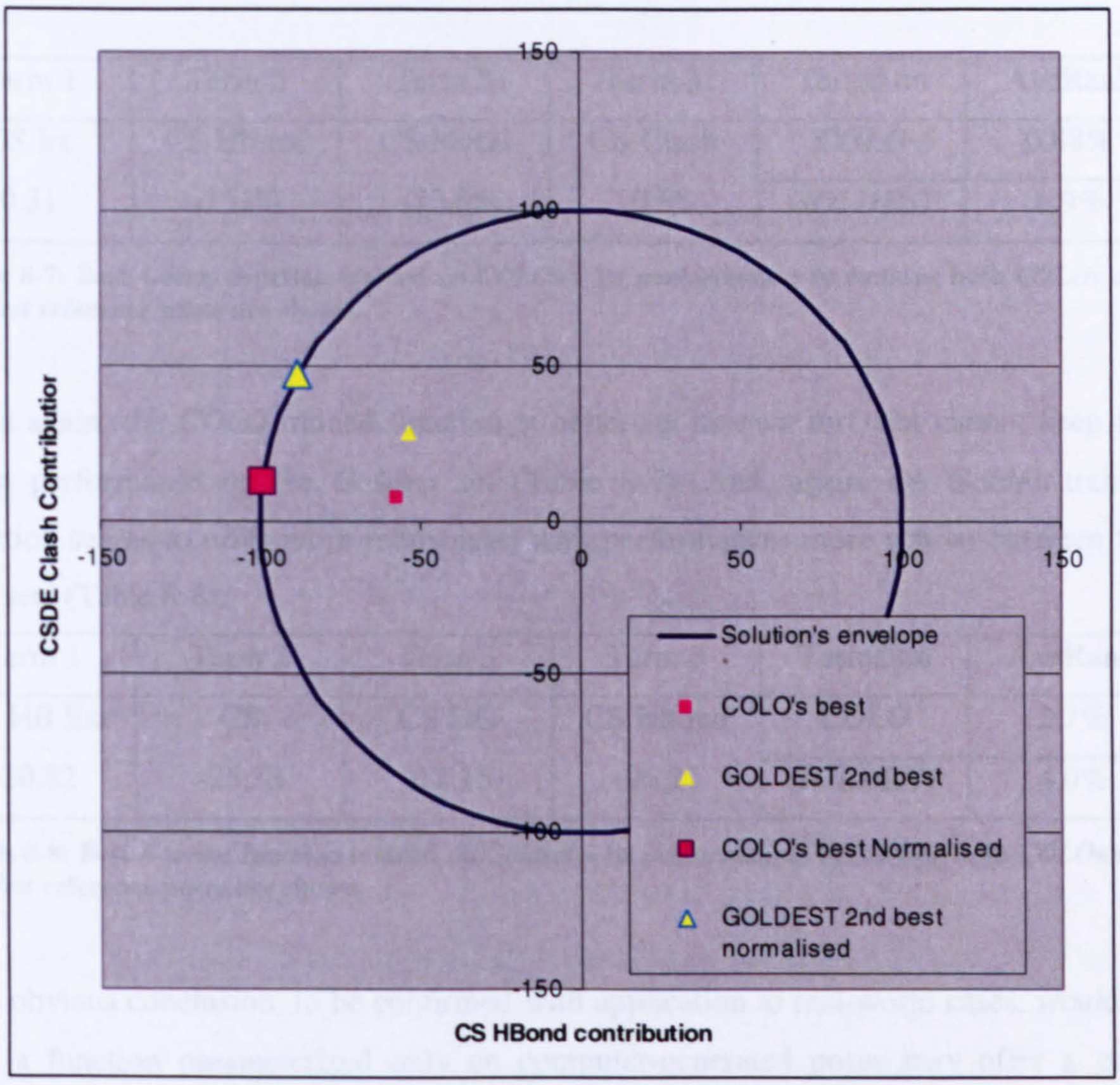


Figure 8-1: Plot showing the projection on the CS Hbond / CS Clash space of the original and normalised solutions for *COLO*'s best function, and *GOLDEST* second best.

Also, its importance is limited by its low rate of occurrence: referring again to subset 27-A as an example, only a few complexes (1azm, 1cil, 1dd7, 1xid, 3cpa, all of them metal-containing enzymes – Zn^{2+}, Mn^{2+}, Fe^{2+} - in Heme) will have poses with a non-zero value for this term: in those complexes, it may however make a great difference in the ranking of the reference compounds, and this is reflected in the large multiplicative coefficient. However, on adding CS Metal a general performance improvement is observed, which indicates that metal-containing proteins were not adequately modelled by the 2-term function.

Functions composed of four terms are the last to be discussed in this section:

Term 1	Term 2	Term 3	Term 3	Tested on	AveRank
GS Int	CS Hbond	CS Metal	CS Clash	*COLO*	0.88%
0.31	-17.00	-73.85	10.88	*GOLDEST*	6.9%

Table 8-7: Best 4-term function trained on *COLO*s – its performances in ranking both *COLO*s and *Goldest* reference poses are shown.

Once again, the *COLO*-trained function is better on its own turf, but cannot keep the same performance on the *Goldest* set (Table 8-7). And, again, the *Goldest*-trained function seems to offer more robustness, with performances more similar between the two sets (Table 8-8).

Term 1	Term 2	Term 3	Term 3	Tested on	AveRank
GS HB Ext	CS	CS DG	CS Hbond	*COLO*	2.7%
-20.82	-25.32	-17.15	-96.21	*GOLDEST*	4.0%

Table 8-8: Best 4-terms function trained on *Goldest* – its performances in ranking both *COLO*s and *Goldest* reference poses are shown.

The obvious conclusion, to be confirmed with application to real-world cases, would be that a function parameterized only on computer-generated poses may offer a good discriminatory power in both experimental and virtual trials, even better than a function solely trained against[JW73].

Unfortunately, because of an error in the GA initialization files, the two hydrogen bond terms present in the input were allowed to mix (one was labelled as HBond, the other as HBOND – the check routine was case-sensitive). This resulted in the best function containing both CHEMSCORE's and GOLDSCORE's hydrogen bond terms[JW74], interesting in its own regards, since it implies that the two implementations are complementary.

However, this does not alter the meaningfulness of other GA runs, with a proper selection of terms. If the performing list is scanned, the first combination unaffected by this issue is the following (Table 8-9).

Term 1	Term 2	Term 3	Term 3	Tested on	AveRank
CS	CS DG	CS_HBond	CS Metal	*COLO*	1.6%
-4.87	3.83	-30.62	-99.07	***GOLDEST***	4.3%

Table 8-9: *Correct* best 4-term function trained on *Goldest* – its performances in ranking both *COLO*s and *Goldest* reference poses are shown.

Performances on *Goldest* are only marginally worse compared to the best (but wrong) function containing two HBond terms (Table 8-8): 4.3% of average ranking for the references, instead of 4.0%. However, this second-best function outperforms the best in ranking crystallographic references (COLO: 1.6% vs 2.7%). Our conclusions are not changed in their substance by this substitution. *Goldest*-trained functions seem to be more robust, and of greater general applicability, than *COLO*-trained ones.

8.2.2 Training on subsets

The GOLDSCORE+CHEMSCORE training set was also split along the same lines presented before, and here are shown some example results for the 27-complex sets. For the sake of brevity, only combinations parameterized on the first of the four 27-complex sets are shown. However, performances on all sets will be presented, so as to offer a measure of how well the functions may perform on previously unseen cases.

The best 2-term combination trained on set 27-A was (Table 8-10):

Term 1	Term 2	Tested on	AveRank
CS	CS Lipo	***Set 27-A***	1.14%
-97.90	9.88	***Set 27-B***	2.02%
		Set 27-C	2.22%
		Set 27-D	4.51%

Table 8-10: Best 2-term scoring function trained against set 27-A, and its performance on sets 27-A to D.

Once again, CHEMSCORE (CS), taken with a minus sign, is the best performer, once its lipophilic interaction term is enhanced. Performance on its own training set (27-A) is very good, and this is well conserved across the three test sets.

Results for the 4-terms combinations follow (Table 8-11):

Term 1	Term 2	Term 3	Term 4	Tested On	AveRank
GS Int 5.72	CS -84.59	CS Lipo 9.98	CS Clash 71.13	**Set 27-A**	0.64%
				Set 27-B	1.02%
				Set 27-C	1.10%
				Set 27-D	3.95%

Table 8-11: Best 4-term scoring function trained against set 27-A, and its performance on sets 27-A to D.

The ratio between the first two terms is well conserved, as shown inFigure 8-2: On the left plot, you can see the square (2-terms solution) and the triangle (4-terms solution) quite apart from each other, owing to the fact that the latter receives contribution from other axes not shown; However, when the contribution from these other terms is not taken into account and the remaining two terms are re-normalised, the triangle overlaps with the square – this is shown in the right[JW75]-hand side of the figure, and implies that the ratio between the CHEMSCORE and CS Lipo terms is the same in both solutions.

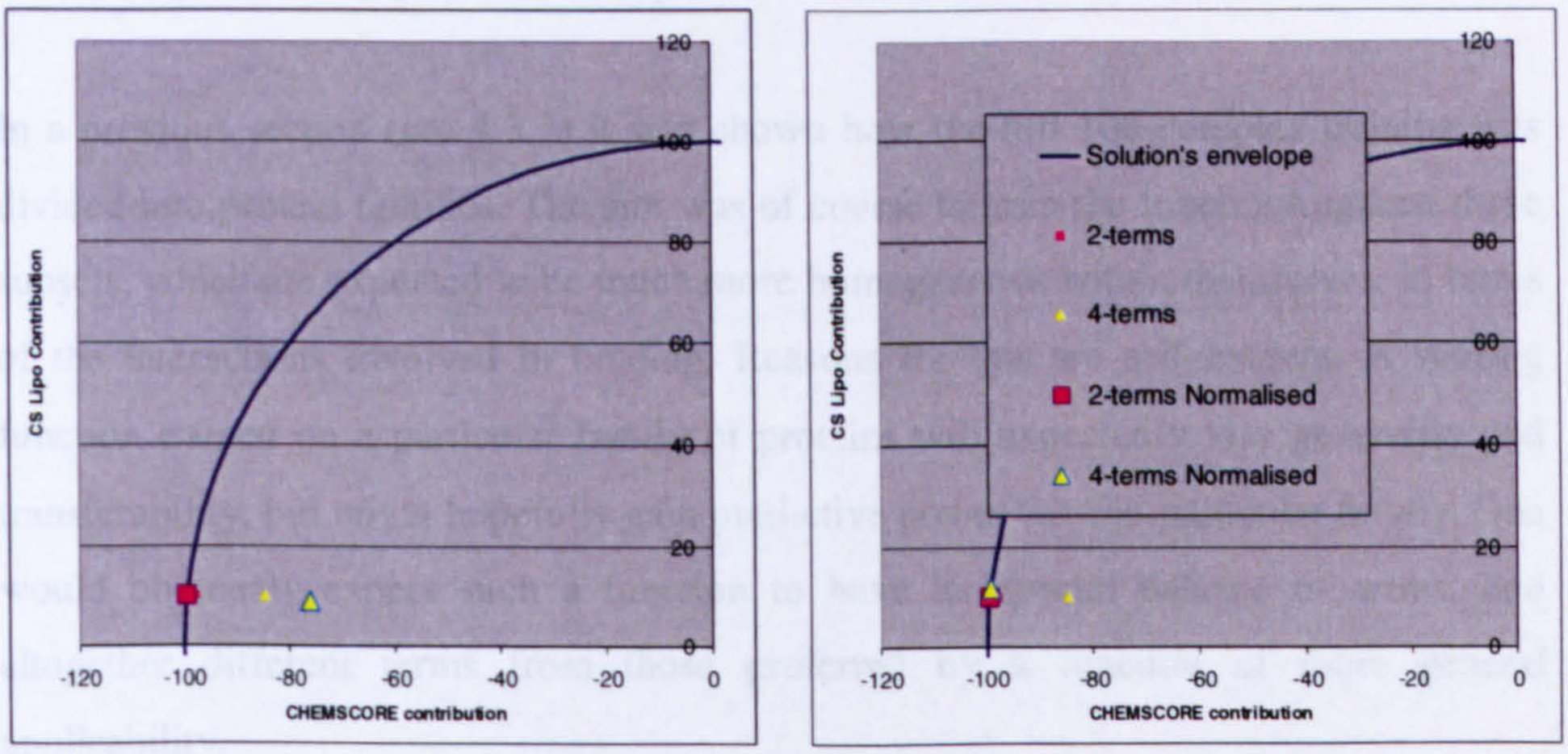


Figure 8-2: Plots showing the best 2-term and 4-term solutions in the CHEMSCORE / CS Lipo space: left, when all components for the 4-terms are accounted; right, neglecting contributions from components other than those represented[JW77].

Ranking performances, too, vary very much in the same way seen in the full set; the function trained on 27-A doubles its performance when passing from 2-term to 4-term descriptions, at least on test sets B (from 2% to 1%) and C (from 2.2% to 1.1%): set D instead lags behind at 4% or so, with little improvement when doubling the number of terms in the function (from 4.51% to 3.95%).

8.2.3 Conclusions

It is time to draw some conclusions regarding this first section of the chapter. The results presented show, once more, how it is possible to modify existing scoring functions to improve their ranking efficiency. It is not necessary to implement new interaction terms from scratch: good and transferable results can be obtained by simply altering the relative contribution of terms within a function already available. Another opportunity is to cherry-pick terms from different functions to build a superior one. Also, it appears that parameterising on COLOs can inadvertently decrease performance on 'real world' scenarios – a working alternative seems to be to train against molecules which are known to be active against a target, but using poses generated by the same docking program which will be used later on to perform the virtual screening. The following section will instead explore how training on different families can affect both the mix of terms selected, and their balance when a combination is forcefully imposed.

8.3 Single Family parameterisation

In a previous section (see 4.3.2) it was shown how the full 108-complex training was divided into protein families. The aim was of course to train the functions against these subsets, which are expected to be much more homogeneous within themselves, in terms of the interactions involved in binding. Reasons for this are self evident. A scoring function trained on a particular family of proteins will expectedly lose generality and transferability, but might hopefully gain predictive power for that particular family. One would obviously expect such a function to have its special balance of terms, and altogether different terms from those preferred by a function of more general applicability.

8.4 Families and subsets

A detailed explanation of how the splitting was performed was already given in section 4.3.2. Suffice it to say that we had nine different subsets, with numbers of complexes ranging from 4 (DHFRs) to 21 (Proteases). They were selected at the "superfamily" level using SCOP [175, 180]. The largest of them, namely the protease set, is actually an assembly of all different protease super-families present in the full set. Serine proteases

were also given their own subset. The sets with their entries are given in Table 4-2 and Table 4-3.

8.5 Training

The training procedure is exactly the same as detailed previously. Scoring functions are incrementally assembled, and then compared between different families to show how they prefer different kinds of interactions, or different balances between the same interactions. Unfortunately, there was no opportunity to test these functions on independent test sets. For the moment, we focus on two example cases: a training experiment using our 24 terms plus the three terms from GOLDSCORE, to show how different families prefer different terms. Another function where CHEMSCORE terms are re-weighted was parameterised, to show that when forced to make do with what is available, the different families will give different weights to the various terms, depending on how important those interactions are for the accurate description of their active sites.

8.5.1 Choose your own terms

In the first tier of the parameterisation, the training was performed using GA input files containing 28 terms per pose, 24 of our own implementation, plus GOLDSCORE, and its three constituent terms. For the sake of clarity, the discussion will be limited to the 2 and 3-term combinations, although parameterisation was pushed up to six terms[JW78]. Furthermore, in light of what has been seen in the previous experiments, using more than four terms did not improve the predictivity of the functions; and when training on a handful of proteins, like the single family sets used here, using more than three is unlikely to be meaningful, and will actually induce the over-fitting of the functions

Let us then, have a look at the first results in Table 8-12. First, there is something to say about the terms chosen by each family. While the full set chose LigHPhobB and 4/8ExtVdW (both as penalties, one may notice), single-family subsets show different behaviour. Immunoglobulins keep the same terms (LigHPhobB is substituted by the strongly correlated LigHPhobS), more or less with the same ratio, with a few tweaks to

accommodate the diversity of the surface term, and to account for the difference in training. Other families prefer a hydrogen bond (GoodHBond for nuclear receptors, GOLDHBOND for proteases) to surface terms, therefore indicating strong polar interactions within the site.

Set	1st Term	2nd Term	SCORERANK
Full 108 proteins	LigHPhobB 61.11	4/8ExtVdW 79.15	**8.68 %**
Dihydrofolate Reductases	GoodHBond -99.93	LigHPhilB -3.67	**0.22 %**
Immunoglobulins	LigHPhobS 65.26	4/8ExtVdW 75.77	**3.14 %**
Lipocalins	Coul 100. 00	6/9ExtVdW 0.99	**0.98 %**
Nuclear Receptors	GoodHBond -99.91	4/8ExtVdW 4.35	**0.10 %**
Exopeptidases	CL8 Å n GBSA -33.86	LigHPhobS 94.09	**0.18 %**
Kinases	BadHBond 100.00	4/8IntVdW 0.07	**1.95 %**
Neuraminidases	4/8IntVdW 2.81	GOLDHBOND -99.96	**1.24 %**
Proteases	4/8ExtVdW 2.25	GOLDHBOND -99.97	**5.86 %**
Serine Proteases	Coul 0.43	4R-8Å Coul -100.0	**2.54 %**

Table 8-12: Best two-term functions selected by the GA. Reference poses are COLOs. Term names and coefficient values are shown for each protein's superfamily. The first line refers to the equivalent result for the full 108 protein set.

Other proteins, such as exo-peptidases, swap the van der Waals term for the sum of coulombic and GBSA, which was classified as a "solvation" term. Lipocalins choose a coulombic term associated with a steeper than usual 6/9ExtVdW; the multiplicative coefficient applied to this term is somewhat smaller, though, possibly to keep values in check.

Surprisingly, serine proteases go for a strange pair, pure coulombic plus a coulombic with 4R dielectric shielding and cutoff at 8Å. The two terms were allowed to be selected together even if they are very similar and strongly correlated since they ideally belong to two different classes: Coulomb is a purely electrostatic term whereas Coul-8Å-4R is defined as 'solvation' method (see section 6.2.4). However, a scan of the best 2-term solutions for serine proteases shows that they all are electrostatic/solvation combinations. To find the first 'sensible' choice of terms, Coul-4R-8Å - 4/8ExtVdW, we must get to the 11th best solution, although the performance is not much worse than the absolute best (2.8% instead of 2.5%).

Set	1st Term	2nd Term	3rd Term	Rank
Full 108 proteins	GoodHBond -99.88	LigHPhobS 2.48	4/8ExtVdW 4.16	**5.14 %**
Dihydrofolate Reductases	GoodHBond -79.53	BadHBond 60.55	LigHPhilB -3.10	**0.12 %**
Immunoglobulins	LigHPhobS 68.50	4/8ExtVdW 72.74	4/8IntVdW 4.00	**2.40 %**
Lipocalins	Coul 99.99	LigHPhilS 1.07	6/9ExtVdW 0.44	**0.42 %**
Nuclear Receptors	GoodHBond -52.17	8Å Coul 85.30	4/8ExtVdW 1.73	**0.10 %**
Exopeptidases	CL8 Å nGBSA -4.39	LigHPhobS 90.48	4/8ExtVdW 42.35	**0.14 %**
Kinases	GoodHBond -11.83	BadHBond 99.30	4/8IntVdW 0.14	**1.17 %**
Neuraminidases	LigHPhobS 4.60	GOLDHBOND -48.24	GOLDEXVdW -87.47	**0.28 %**
Proteases	LigHPhilB -90.14	4/8ExtVdW 43.30	6/12InVdW -0.15	**3.79 %**
Serine Proteases	4R Coul 28.60	8Å Coul -95.82	4/8ExtVdW 0.05	**1.93 %**

Table 8-13: Best three-term functions selected by the GA. Reference poses are COLOs. Terms names and coefficient values are shown for each protein's superfamily considered. First line refers to the equivalent result for the full 108 protein set.

It is worth noticing that 4/8ExtVdW would be the next term to appear, if we were to check the best three-term combination for serine proteases (Table 8-13). This preference for electrostatic-based terms may be due to the highly polar nature of their active site, where a highly conserved group of charged or polar amino acids (His57, Ser195, Asp102) perform the protein cleavage.

It certainly looks like, in general, our expected result of having different terms assembled to tailor the function to the protein family yields improved rankings, although it may be worth performing further analysis on the unusual cases. Again, serine proteases show a definitely bizarre behaviour, selecting 4R Coul (purely coulombic), 8Å Coul (solvation) and a more normal 4/8ExtVdW; this last term's coefficient is, however, very small as compared to the others, and its contribution to enrichment is definitely *minor* (from 2.5% to 1.9% to recover half the true ligands).

8.5.2 Choose your coefficients

So, it has been shown that when given the opportunity, different families will be better described by functions with different interaction types, reflecting the importance of those interactions in their binding modes. An influence is to be expected also depending on the ligands used as references when training – multiple binding modes are possible, and the presence or absence of a particular chemical group can greatly affect the binding mode.

What happens if, instead, families are not allowed to select their own terms, but forced to use a given selection and just asked to parameterise them? This may be a better solution in view of practical application: this way, one will not have to change scoring terms every time, but rather just modify their weights to better suit that family. Such a modification to the program is easily set-up.

For this experiment, it was decided to only consider terms derived from commercially available scoring functions: and since CHEMSCORE terms performed so well in section 8.2, its terms were chosen.

Family	#P	DG	HB	Metal	Lipo	Clash	Int	*AveRank*
DHFRs	5	36.0	-98.1	13.2	9.1	58.3	94.6	***0.12%***
ExoPeptidases	5	2.1	-93.3	17.8	2.8	24.6	79.4	***0.09%***
ImmunoGlobulins	10	17.8	-80.5	10.2	0.9	45.7	89.3	***1.16%***
Kinases	7	55.3	-29.8	39.4	7.0	98.4	-6.1	***1.21%***
LipoCalins	6	27.4	16.5	-36.4	2.2	56.2	70.3	***0.55%***
NeurAminidases	4	24.6	-98.8	-45.1	0.9	28.6	47.5	***0.28%***
Nuclear Receptors	4	51.0	-83.2	-18.4	3.1	97.9	87.3	***0.10%***
Proteases	21	9.9	-87.2	-97.4	0.1	41.4	44.4	***0.14%***
Serine Proteases	12	-8.4	-95.3	91.0	-1.9	39.2	52.9	***0.13%***

Table 8-14: Table showing the weights chosen (and performances obtained) for the CHEMSCORE terms trained on nine different families of proteins.

In the Table 8-14, the two left-most columns show the name of the family, and the number of complexes included in the set. The rightmost is the performance attained by CHEMSCORE's terms when they are weighted by the coefficients shown. Negative coefficients are highlighted in red, so that it is easier to spot patterns: particularly, it is interesting to note how the hydrogen bond term (CS_HB) is favourable to binding for all but lipocalin complexes. This is not surprising since these proteins do principally bind hydrophobic ligands as their name suggest (***lipo***, as in fat, and ***calix***, as in cup)[223, 224]. If the CS_HB score for the reference poses is compared between lipocalins and kinases, which bind more polar ligands, the average score is higher for kinases (see Table 8-15).

Let us spend some moments to explain the information conveyed by this table: for four protein families, average CS HB scores and their standard deviations for decoys are reported; this gives us a measure of the significance of this term for the decoys, and how much variability one can expect. Also, the same values are reported for the reference ligands. The same rule of sign applies as before: negative values mean that the term is favouring binding, whereas positive ones oppose it. If a term is an important force for binding, one would expect the average score for the reference to be significantly different from the average for decoys: significantly higher in the case of penalties, lower

for terms which favour binding, but remember that a negative coefficient would invert this ordering.

Family	Reference		Decoys		Distance[JW79] decoys/references (in stdev)
	Average[JW80]	± STDEV[JW81]	Average	± STDEV	
Lipocalins	1.06 17.49	0.98 16.16	0.8407 25.05	0.73 11.95	0.60
Kinases	3.071429 -91.53	3.31441 98.77	0.809967 -24.14	0.8407 25.05	2.69
Imm.Globulins	3.61 -290.4	2.01 -161.6	0.66 -53.01	0.67 -53.71	4.42
Ser Proteases	2.87 -273.75	1.24 -118.42	0.73 -70.00	0.66 -62.74	3.25

Table 8-15: Average CS HB score for both decoys and reference poses in lipocalins, kinases, immunoglobulins and serine protease protein families – in black, absolute values – red or blue, values scaled by the coefficient obtained from Table 8-14 (*e.g.* 1.06*16.5=17.49).

The standard deviation is a good measure of intrinsic variability of the decoys scores: assuming that our decoys have normally distributed values implies that 66% will be within one standard deviation of the average (and 95% within 2 STDEV); our CS HB values are not normally distributed (some are only positive). CS HB resembles more a Poisson distribution. Still, the rationale applies – qualitatively, if not in purely numerical terms. References with scores far apart from the average will be more readily discriminated.

For kinases, a higher CS HB score is observed in the reference (-91.53) than in the decoys (-24.14 ± 25.05). Unfortunately, the standard deviation for the references is high too (98.77, mostly due to pdb entry 1ei1, having an anomalously high CS HB score), so the separation is not perfect; still, it is clear that the term is an important driver for binding. Lipocalins' references, on the other hand, have a lower score (17.49), much closer to the average of decoys and within the decoys' variability (10.30±11.95): this means that the functions cannot use the hydrogen bond to discriminate between binders and non-binders. It is not that decoys would not form hydrogen bonds, it is just that the lipocalins pockets are (relatively) devoid of anchor points: Figure 8-3 shows a

comparison between three randomly selected lipocalins and three other randomly selected proteins, an exopeptidase, a protease, and an oxido-reductase: all of which show at least 4 hydrogen bond anchor point per ligand; the lipocalins ligands, by comparison, only have 1 or 2 anchor points. Also, most of the residues surrounding lipocalins' ligands are hydrophobic; the opposite is true for the other three proteins. Therefore, the average scaled CS HB scores for both references and decoys are similar, and even with a decidedly positive coefficient the term does not play any role in discriminating between the two.

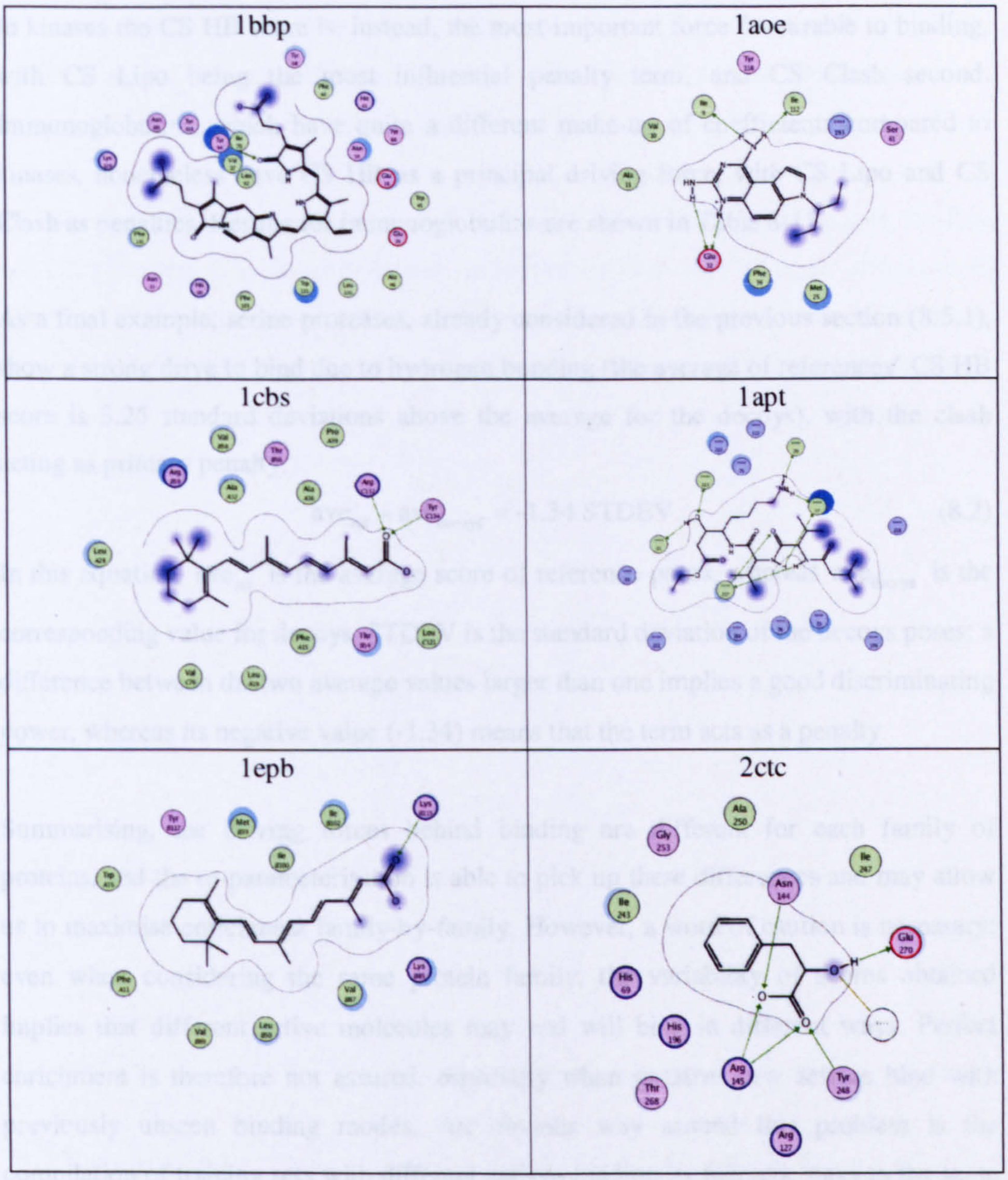


Figure 8-3: on the left side, three of the lipocalins used in the work, showing only one or two Hydrogen bonds per ligand; on the right side, an oxido-reductase (1aoe), a protease (1apt), and an exopeptidase (2ctc) - interaction diagrams obtained with MOE (Chemical Computing Group).

Considering the significance of the lipocalins' data for each term, the most important term turns out to be CS Clash: rather unsurprisingly, since once directional interactions are taken out, the main contributor to binding is shape complementarity, with sharing of hydrophobic patches. The second most significant term is CS DE, the internal energy term, although the average for references within a standard deviation of the decoys' average indicates a low discriminating power for this term.

In kinases the CS HB score is, instead, the most important force favourable to binding, with CS Lipo being the most influential penalty term, and CS Clash second. Immunoglobulins, which have quite a different make-up of coefficients compared to kinases, nonetheless have CS HB as a principal driving force, with CS Lipo and CS Clash as penalties. Results for immunoglobulins are shown in Table 8-15.

As a final example, serine proteases, already considered in the previous section (8.5.1), show a strong drive to bind due to hydrogen bonding (the average of references' CS HB score is 3.25 standard deviations above the average for the decoys), with the clash acting as primary penalty:

$$ave_{ref} - ave_{decoys} = -1.34\ STDEV \qquad (8.2)$$

In this equation, ave_{ref} is the average score of reference poses, whereas ave_{decoys} is the corresponding value for decoys. STDEV is the standard deviation of the decoys poses: a difference between the two average values larger than one implies a good discriminating power, whereas its negative value (-1.34) means that the term acts as a penalty

Summarising, the driving forces behind binding are different for each family of proteins, and the re-parameterisation is able to pick up these differences and may allow us to maximise enrichment family-by-family. However, a word of caution is necessary: even when considering the same protein family, the variability of scores obtained implies that different active molecules may and will bind in different ways. Perfect enrichment is therefore not assured, especially when putative new actives bind with previously unseen binding modes. An obvious way around this problem is the compilation of training sets with different actives binding in different ways to the same protein.

8.6 Conclusions

In this chapter, further scoring functions parameterisation have been presented, focusing more on a simple to setup approach to enable commercial scoring functions to be re-trained to quickly and efficiently discriminate between active and inactive molecules. Hybrid scoring function can be created by mixing terms from different scoring functions, providing a superior enrichment as compared to a simple weighted consensus of the scoring functions themselves. The approach has been exemplified using

GOLDSCORE and CHEMSCORE, showing how the outcome of the parameterisation is often non-trivial but still responds to common sense intuition. Performances on independent test set have been presented.

Finally, the re-parameterisation of CHEMSCORE terms on different protein families has been presented, with an extensive discussion of how and why the difference in distribution of scores between reference poses and decoys, and its interplay with the GA-selected coefficients, influences the discriminatory power of the resulting function.

9 Conclusions and Future Work

9.1 Summary of conclusions

We started with a brief explanation of how structure-based virtual screening can help during the Hit Identification phase of drug development: we then explored more in detail the need for more accurate scoring, covering the important issues to date. We set on clarifying some of these issues in the following chapters. After presenting our data setup and procedures, our parameterisation routine particularly aimed at overcoming the lack of experimental structures for weakly and non binding small molecules.

In detail, within the context of the particular parameterisation setup chosen, *i.e.* linear combination of scoring terms, a known active ligand per protein whose ranking we optimise, and Genetic Algorithms (GA) to achieve this, problems were tackled such as: the accuracy and robustness of empirically parameterised force-field-like scoring functions, and the dependency of their performances on the choice of the training set. The influence of the kind of reference poses chosen to act as actives was investigated, comparing crystallographical orientations (COLO), with both the best docked structures (*Goldest*) and those closest to the COLOs. Surprisingly, it was found that using Goldest as references can yield functions with good predictivity and in some cases more robustness than functions trained on crystallographical orientations of the same active ligands.

The performances of the said functions is dependent on the size and quality of the training set, and this has been shown with examples at various level of detail. What is more important, though, is that even functions parameterised on about thirty complexes are capable of offering good results in unknown cases, and there is space for performing *ad hoc* parameterisation aimed at single protein families with even higher ranking power.

The possibility of optimising in such a way not only newly implemented scoring functions, but also of re-optimising commercially available functions, where terms are available, or even pick-and-mix terms coming from different scoring functions, was explored with good success, especially in the case of functions explicitly created to reproduce known binding affinities, such as CHEMSCORE.

9.2 Future work

The project is far from being complete, and many avenues of investigation deserve further attention: unfortunately, owing to time constraints, and sometimes opportunity, there was no time to perform some important tests, such as an assessment of the performances of the newly built (or re-trained, in the case of CHEMSCORE) scoring function on real virtual screening experiments coming from one of the funding companies: a thorough test is particularly needed in the case of single family scoring functions, to assess their actual performances, and their robustness to variation of the training set. It is our hope that this will be performed in the near future, to properly complete the work and provide one more (hopefully valuable) tool and technique to the field. On a personal level, the hope is to be able to follow the project in its continuation, possibly as an official collaborator and perhaps supervisor.

9.3 Final remarks

Once again, I want to thank everybody who helped me put together this work, those who supported me during the whole length or only part of it, and who provided entertainment in and/or out of the work hours. This thesis is in part yours, too. The good that it's in it, I mean. I take full responsibility for the bad parts. ☺

References

1. Newman, D. J.; Cragg, G. M.; Snader, K. M., Natural Products as Sources of New Drugs over the Period 1981-2002. *J. Nat. Prod.* **2003,** 66, 1022-1037.
2. Sharp, K., Entropy-enthalpy compensation: fact or artifact? *Protein Sci* **2001,** 10, (3), 661-7.
3. Cornish-Bowden, A., Enthalpy-entropy compensation: a phantom phenomenon. *J Biosci* **2002,** 27, (2), 121-6.
4. *Chemical and Structural Approaches to Rational Drug Design.* CRC: 1994; p 288.
5. *Biological Approaches to Rational Drug Design.* CRC: 1994; p 208.
6. Lin, J.; Sahakian, D. C.; de Morais, S. M. F.; Xu, J. H.; Polzer, R. J.; Winter, S. M., The role of absorption, distribution, metabolism, excretion and toxicity in drug discovery. *Current Topics in Medicinal Chemistry* **2003,** 3, (10), 1125-1154.
7. Kerns, E. H.; Di, L., Pharmaceutical profiling in drug discovery. *Drug Discovery Today* **2003,** 8, (7), 316-323.
8. Roberts, S. A., Drug metabolism and pharmacokinetics in drug discovery. *Current Opinion in Drug Discovery & Development* **2003,** 6, (1), 66-80.
9. Li, A. P., Screening for human ADME/Tox drug properties in drug discovery. *Drug Discovery Today* **2001,** 6, (7), 357-366.
10. Eddershaw, P. J.; Beresford, A. P.; Bayliss, M. K., ADME/PK as part of a rational approach to drug discovery. *Drug Discovery Today* **2000,** 5, (9), 409-414.
11. van de Waterbeemd, H.; Lennernäs, H.; Artursson, P.; Mannhold, R.; Kubinyi, H.; Folkers, G., *Drug Bioavailability: Estimation of Solubility, Permeability, Absorption and Bioavailability.* Wiley-VCH: 2003; p 602.
12. Alighieri, D., La Divina Comedia. In ca 1300.
13. Rapaport, D. C., *The Art of Molecular Dynamics Simulation.* Cambridge University Press: 2004; p 564.
14. Allen, M. P.; Tildesley, D. J., *Computer Simulation of Liquids.* Clarendon Press: 1989; p 404.
15. McQuarrie, D. A., *Statistical Mechanics* University Science Books,U.S.: 2000; p 640.
16. Atkins, P. W.; Friedman, R. S., *Molecular Quantum Mechanics* Oxford University Press: 2004; p 608.
17. Leach, A. R.; Gillet, V. J., *An Introduction to Chemoinformatics.* Kluwer Academic Press: 2003; p 274.
18. Leach, A. R.; Jothi, H.; Hann, M. M.; Murray, C.; Murcko, M. A.; Walters, W. P.; Stevens, R. C.; Essex, J. W.; Verdonk, M., *Structure-Based Drug Discovery.* Kluwer Academic Press: 2006; p 300.
19. Nielsen, S. O.; Lopez, C. F.; Srinivas, G.; Klein, M. L., Coarse grain models and the computer simulation of soft materials. *Journal of Physics-Condensed Matter* **2004,** 16, (15), R481-R512.
20. Tozzini, V., Coarse-grained models for proteins. *Current Opinion in Structural Biology* **2005,** 15, (2), 144-150.

21. Sotriffer, C.; Klebe, G.; Stahl, M.; Bohm, H.-J., Docking and Scoring Functions / Virtual Screening. In *Burger's Medicinal Chemistry and Drug Discovery*, 6th ed.; 2004; Vol. 1, pp 281-333.
22. Fischer, E., Einfluss der configuration auf die wirkung derenzyme. *Ber. Dt. Chem. Ges.* **1894,** 27, 2985-2993.
23. Murray, C. W.; Baxter, C. A.; Frenkel, A. D., The sensitivity of the results of molecular docking to induced fit effects: Application to thrombin, thermolysin and neuraminidase. *Journal of Computer-Aided Molecular Design* **1999,** 13, (6), 547-562.
24. Ragno, R.; Frasca, S.; Manetti, F.; Brizzi, A.; Massa, S., HIV-reverse transcriptase inhibition: Inclusion of ligand-induced fit by cross-docking studies. *Journal of Medicinal Chemistry* **2005,** 48, (1), 200-212.
25. Koshland, D. E., The Key-Lock Theory and the Induced Fit Theory. *Angewandte Chemie International Edition in English* **2003,** 33, (23-24), 2375-2378.
26. Noble, M.; Barrett, P.; Endicott, J.; Johnson, L.; McDonnell, J.; Robertson, G.; Zawaira, A., Exploiting structural principles to design cyclin-dependent kinase inhibitors. *Biochim Biophys Acta* **2005,** 1754, (1-2), 58-64.
27. Kroeze, W. K.; Sheffler, D. J.; Roth, B. L., G-protein-coupled receptors at a glance. *Journal of Cell Science* **2003,** 116, (24), 4867-4869.
28. De Ontellano, P. R. O., *Cytochrome P450: structure, mechanism, and biochemistry.* 2nd ed.; Kluwer Academic: 2004; p 709.
29. Jakalian, A.; Jack, D. B.; Bayly, C. I., Fast, efficient generation of high-quality atomic charges. AM1-BCC model: II. Parameterization and validation. *Journal of Computational Chemistry* **2002,** 23, (16), 1623-1641.
30. Jakalian, A.; Bush, B. L.; Jack, D. B.; Bayly, C. I., Fast, efficient generation of high-quality atomic Charges. AM1-BCC model: I. Method. *Journal of Computational Chemistry* **2000,** 21, (2), 132-146.
31. Wang, J. M.; Cieplak, P.; Kollman, P. A., How well does a restrained electrostatic potential (RESP) model perform in calculating conformational energies of organic and biological molecules? *Journal of Computational Chemistry* **2000,** 21, (12), 1049-1074.
32. Fox, T.; Kollman, P. A., Application of the RESP methodology in the parametrization of organic solvents. *Journal of Physical Chemistry B* **1998,** 102, (41), 8070-8079.
33. Bowie, J. U.; Luthy, R.; Eisenberg, D., A Method to Identify Protein Sequences That Fold into a Known 3-Dimensional Structure. *Science* **1991,** 253, (5016), 164-170.
34. Nayeem, A.; Sitkoff, D.; Krystek, S., A comparative study of available software for high-accuracy homology modeling: From sequence alignments to structural models. *Protein Science* **2006,** 15, (4), 808-824.
35. Sussman, J. L.; Abola, E. E.; Lin, D.; Jiang, J.; Manning, N. O.; Prilusky, J., The protein data bank - Bridging the gap between the sequence and 3D structure world. *Genetica* **1999,** 106, (1-2), 149-158.
36. Bernstein, F. C.; Koetzle, T. F.; Williams, G. J.; Meyer, E. E. J.; Brice, M. D.; Rodgers, J. R.; Kennard, O.; Shimanouchi, T.; Tasumi, M., The protein data bank: a computer-based archival file for macromolecular structures. *Journal of Molecular Biology* **1977,** 112, 535-542.
37. Taylor, R. D.; Jewsbury, P. J.; Essex, J. W., A review of protein-small molecule docking methods. *Journal of Computer-Aided Molecular Design* **2002,** 16, (3), 151-166.

38. Wang, R. X.; Lu, Y. P.; Wang, S. M., Comparative evaluation of 11 scoring functions for molecular docking. *Journal of Medicinal Chemistry* **2003,** 46, (12), 2287-2303.
39. Kuntz, I. D.; Blaney, J. M.; Oatley, S. J.; Langridge, R.; Ferrin, T. E., A Geometric Approach to Macromolecule-Ligand Interactions. *Journal of Molecular Biology* **1982,** 161, (2), 269-288.
40. Jones, G.; Willett, P.; Glen, R. C.; Leach, A. R.; Taylor, R., Development and validation of a genetic algorithm for flexible docking. *Journal of Molecular Biology* **1997,** 267, (3), 727-748.
41. Morris, G. M.; Goodsell, D. S.; Halliday, R. S.; Huey, R.; Hart, W. E.; Belew, R. K.; Olson, A. J., Automated docking using a Lamarckian genetic algorithm and an empirical binding free energy function. *Journal of Computational Chemistry* **1998,** 19, (14), 1639-1662.
42. Jones, G.; Willett, P.; Glen, R. C., Molecular Recognition of Receptor-Sites Using a Genetic Algorithm with a Description of Desolvation. *Journal of Molecular Biology* **1995,** 245, (1), 43-53.
43. Jones, G.; Willett, P.; Glen, R. C., A genetic algorithm for flexible molecular overlay and pharmacophore elucidation. *Journal of Computer-Aided Molecular Design* **1995,** 9, (6), 532-549.
44. Friesner, R. A.; Banks, J. L.; Murphy, R. B.; Halgren, T. A.; Klicic, J. J.; Mainz, D. T.; Repasky, M. P.; Knoll, E. H.; Shelley, M.; Perry, J. K.; Shaw, D. E.; Francis, P.; Shenkin, P. S., Glide: A new approach for rapid, accurate docking and scoring. 1. Method and assessment of docking accuracy. *Journal of Medicinal Chemistry* **2004,** 47, (7), 1739-1749.
45. Halgren, T. A.; Murphy, R. B.; Friesner, R. A.; Beard, H. S.; Frye, L. L.; Pollard, W. T.; Banks, J. L., Glide: A new approach for rapid, accurate docking and scoring. 2. Enrichment factors in database screening. *Journal of Medicinal Chemistry* **2004,** 47, (7), 1750-1759.
46. McMartin, C.; Bohacek, R. S., QXP: Powerful, rapid computer algorithms for structure-based drug design. *Journal of Computer-Aided Molecular Design* **1997,** 11, (4), 333-344.
47. Skillman, G.; Stahl, M.; McGann, M., Docking and virtual screening with FRED. *Abstracts of Papers of the American Chemical Society* **2001,** 222, U384-U384.
48. Stahl, M.; Skillman, G., Omega: Conformer generation redux. *Abstracts of Papers of the American Chemical Society* **2005,** 229, U776-U776.
49. Smith, R.; Hubbard, R. E.; Gschwend, D. A.; Leach, A. R.; Good, A. C., Analysis and optimization of structure-based virtual screening protocols (3). New methods and old problems in scoring function design. *Journal of Molecular Graphics & Modelling* **2003,** 22, (1), 41-53.
50. Liu, H. Y.; Kuntz, I. D.; Zou, X. Q., Pairwise GB/SA scoring function for structure-based drug design. *Journal of Physical Chemistry B* **2004,** 108, (17), 5453-5462.
51. Murray, C. W.; Auton, T. R.; Eldridge, M. D., Empirical scoring functions. II. The testing of an empirical scoring function for the prediction of ligand-receptor binding affinities and the use of Bayesian regression to improve the quality of the model. *Journal of Computer-Aided Molecular Design* **1998,** 12, (5), 503-519.
52. Eldridge, M. D.; Murray, C. W.; Auton, T. R.; Paolini, G. V.; Mee, R. P., Empirical scoring functions .1. The development of a fast empirical scoring function to estimate the binding affinity of ligands in receptor complexes. *Journal of Computer-Aided Molecular Design* **1997,** 11, (5), 425-445.

53. Verdonk, M. L.; Cole, J. C.; Hartshorn, M. J.; Murray, C. W.; Taylor, R. D., Improved protein-ligand docking using GOLD. *Proteins-Structure Function and Genetics* **2003,** 52, (4), 609-623.
54. Damm, W.; Halgren, T. A.; Murphy, R. B.; Smondyrev, A. M.; Friesner, R. A.; Jorgensen, W. L., OPLS_2002: A new version of the OPLS-AA force field. *Abstracts of Papers of the American Chemical Society* **2002,** 224, U471-U471.
55. Jorgensen, W. L.; Tirado-Rives, J., Development of the OPLS-AA force field for organic and biomolecular systems. *Abstracts of Papers of the American Chemical Society* **1998,** 216, U696-U696.
56. Halperin, I.; Ma, B. Y.; Wolfson, H.; Nussinov, R., Principles of docking: An overview of search algorithms and a guide to scoring functions. *Proteins-Structure Function and Genetics* **2002,** 47, (4), 409-443.
57. Bissantz, C.; Folkers, G.; Rognan, D., Protein-based virtual screening of chemical databases. 1. Evaluation of different docking/scoring combinations. *Journal of Medicinal Chemistry* **2000,** 43, (25), 4759-4767.
58. Ha, S.; Andreani, R.; Robbins, A.; Muegge, I., Evaluation of docking/scoring approaches: A comparative study based on MMP3 inhibitors. *Journal of Computer-Aided Molecular Design* **2000,** 14, (5), 435-448.
59. Kellenberger, E.; Rodrigo, J.; Muller, P.; Rognan, D., Comparative evaluation of eight docking tools for docking and virtual screening accuracy. *Proteins-Structure Function and Bioinformatics* **2004,** 57, (2), 225-242.
60. Kontoyianni, M.; McClellan, L. M.; Sokol, G. S., Evaluation of docking performance: Comparative data on docking algorithms. *Journal of Medicinal Chemistry* **2004,** 47, (3), 558-565.
61. Perola, E.; Walters, W. P.; Charifson, P. S., A detailed comparison of current docking and scoring methods on systems of pharmaceutical relevance. *Proteins-Structure Function and Bioinformatics* **2004,** 56, (2), 235-249.
62. Atkins, P. W., *Physical Chemistry*. 6th ed.; Oxford University Press: 1998.
63. Leach, A. R., *Molecular Modelling: Principles and Applications*. 2nd ed.; Prentice Hall: 2001; p 720.
64. Ajay; Murcko, M. A., Computational methods to predict binding free energy in ligand-receptor complexes. *Journal of Medicinal Chemistry* **1995,** 38, (26), 4953-4967.
65. Bohm, H. J.; Klebe, G., What can we learn from molecular recognition in protein-ligand complexes for the design of new drugs? *Angewandte Chemie International Edition in English* **1996,** 35, (22), 2589-2614.
66. Mardis, K. L.; Luo, R.; Gilson, M. K., Interpreting trends in the binding of cyclic ureas to HIV-1 protease. *Journal of Molecular Biology* **2001,** 309, (2), 507-517.
67. Luo, R.; Gilson, M. K., Synthetic adenine receptors: Direct calculation of binding affinity and entropy. *Journal of the American Chemical Society* **2000,** 122, (12), 2934-2937.
68. Giordanetto, F.; Cotesta, S.; Catana, C.; Trosset, J. Y.; Vulpetti, A.; Stouten, P. F. W.; Kroemer, R. T., Novel scoring functions comprising QXP, SASA, and protein side-chain entropy terms. *Journal of Chemical Information and Computer Sciences* **2004,** 44, (3), 882-893.
69. Wang, R. X.; Lu, Y. P.; Fang, X. L.; Wang, S. M., An extensive test of 14 scoring functions using the PDBbind refined set of 800 protein-ligand complexes. *Journal of Chemical Information and Computer Sciences* **2004,** 44, (6), 2114-2125.
70. Cornell, W. D.; Cieplak, P.; Bayly, C. I.; Gould, I. R.; Merz, K. M.; Ferguson, D. M.; Spellmeyer, D. C.; Fox, T.; Caldwell, J. W.; Kollman, P. A., A 2nd

Generation Force-Field for the Simulation of Proteins, Nucleic-Acids, and Organic-Molecules. *Journal of the American Chemical Society* **1995,** 117, (19), 5179-5197.

71. Brooks, B. R.; Bruccoleri, R. E.; Olafson, B. D.; States, D. J.; Swaminathan, S.; Karplus, M., Charmm - a Program for Macromolecular Energy, Minimization, and Dynamics Calculations. *Journal of Computational Chemistry* **1983,** 4, (2), 187-217.
72. Meng, E. C.; Shoichet, B. K.; Kuntz, I. D., Automated Docking with Grid-Based Energy Evaluation. *Journal of Computational Chemistry* **1992,** 13, (4), 505-524.
73. Ewing, T. J. A.; Makino, S.; Skillman, A. G.; Kuntz, I. D., DOCK 4.0: Search strategies for automated molecular docking of flexible molecule databases. *Journal of Computer-Aided Molecular Design* **2001,** 15, (5), 411-428.
74. Shoichet, B. K.; Leach, A. R.; Kuntz, I. D., Ligand solvation in molecular docking. *Proteins-Structure Function and Genetics* **1999,** 34, (1), 4-16.
75. Vieth, M.; Hirst, J. D.; Kolinski, A.; Brooks, C. L., Assessing energy functions for flexible docking. *Journal of Computational Chemistry* **1998,** 19, (14), 1612-1622.
76. Jeffrey, G. A. a. S., W., *Hydrogen bonding in biological structures.* Springer-Verlag: 1991.
77. Majeux, N.; Scarsi, M.; Apostolakis, J.; Ehrhardt, C.; Caflisch, A., Exhaustive docking of molecular fragments with electrostatic solvation. *Proteins-Structure Function and Genetics* **1999,** 37, (1), 88-105.
78. Zou, X. Q.; Sun, Y. X.; Kuntz, I. D., Inclusion of solvation in ligand binding free energy calculations using the generalized-born model. *Journal of the American Chemical Society* **1999,** 121, (35), 8033-8043.
79. Greer, J.; Erickson, J. W.; Baldwin, J. J.; Varney, M. D., Application of the 3-Dimensional Structures of Protein Target Molecules in Structure-Based Drug Design. *Journal of Medicinal Chemistry* **1994,** 37, (8), 1035-1054.
80. Bostrom, J.; Norrby, P. O.; Liljefors, T., Conformational energy penalties of protein-bound ligands. *Journal of Computer-Aided Molecular Design* **1998,** 12, (4), 383-396.
81. Pearlman, D. A., Free energy grids: A practical qualitative application of free energy perturbation to ligand design using the OWFEG method. *Journal of Medicinal Chemistry* **1999,** 42, (21), 4313-4324.
82. Pearlman, D. A.; Charifson, P. S., Improved scoring of ligand-protein interactions using OWFEG free energy grids. *Journal of Medicinal Chemistry* **2001,** 44, (4), 502-511.
83. Raha, K.; van der Vaart, A. J.; Riley, K. E.; Peters, M. B.; Westerhofft, L. M.; Kim, H.; Merz, K. M., Pairwise decomposition of residue interaction energies using semiempirical quantum mechanical methods in studies of protein-ligand interaction. *Journal of the American Chemical Society* **2005,** 127, (18), 6583-6594.
84. Raha, K.; Merz, K. M., Large-scale validation of a quantum mechanics based scoring function: Predicting the binding affinity and the binding mode of a diverse set of protein-ligand complexes. *Journal of Medicinal Chemistry* **2005,** 48, (14), 4558-4575.
85. Raha, K.; Merz, K. M., A quantum mechanics-based scoring function: Study of zinc ion-mediated ligand binding. *Journal of the American Chemical Society* **2004,** 126, (4), 1020-1021.

86. Raha, K.; Merz, K. M., Zinc mediated ligand binding: A quantum mechanics based approach. *Abstracts of Papers of the American Chemical Society* **2004,** 227, U1016-U1017.
87. Mark, A. E.; Vangunsteren, W. F., Decomposition of the Free-Energy of a System in Terms of Specific Interactions - Implications for Theoretical and Experimental Studies. *Journal of Molecular Biology* **1994,** 240, (2), 167-176.
88. Andrews, P. R.; Craik, D. J.; Martin, J. L., Functional-Group Contributions to Drug Receptor Interactions. *Journal of Medicinal Chemistry* **1984,** 27, (12), 1648-1657.
89. Kuntz, I. D.; Chen, K.; Sharp, K. A.; Kollman, P. A., The maximal affinity of ligands. *Proceedings of the National Academy of Sciences of the United States of America* **1999,** 96, (18), 9997-10002.
90. Bohm, H. J., The Development of a Simple Empirical Scoring Function to Estimate the Binding Constant for a Protein Ligand Complex of Known 3-Dimensional Structure. *Journal of Computer-Aided Molecular Design* **1994,** 8, (3), 243-256.
91. Head, R. D.; Smythe, M. L.; Oprea, T. I.; Waller, C. L.; Green, S. M.; Marshall, G. R., VALIDATE: A new method for the receptor-based prediction of binding affinities of novel ligands. *Journal of the American Chemical Society* **1996,** 118, (16), 3959-3969.
92. Pan, Y. P.; Huang, N.; Cho, S.; MacKerell, A. D., Consideration of molecular weight during compound selection in virtual target-based database screening. *Journal of Chemical Information and Computer Sciences* **2003,** 43, (1), 267-272.
93. Bohm, H. J.; Stahl, M., Rapid empirical scoring functions in virtual screening applications. *Medicinal Chemistry Research* **1999,** 9, (7-8), 445-462.
94. Laederach, A.; Reilly, P. J., Specific empirical free energy function for automated docking of carbohydrates to proteins. *Journal of Computational Chemistry* **2003,** 24, (14), 1748-1757.
95. Verdonk, M. L.; Chessari, G.; Cole, J. C.; Hartshorn, M. J.; Murray, C. W.; Nissink, J. W. M.; Taylor, R. D.; Taylor, a. R., Modeling Water Molecules in Protein-Ligand Docking Using GOLD. *Journal of Medicinal Chemistry* **2005,** 48, 6504-6515.
96. Todorov, N. P.; Mancera, R. L.; Monthoux, P. H., A new quantum stochastic tunnelling optimisation method for protein-ligand docking. *Chemical Physics Letters* **2003,** 369, (3-4), 257-263.
97. Mancera, R. L.; Kallblad, P.; Todorov, N. P., Ligand-protein docking using a quantum stochastic tunneling optimization method. *Journal of Computational Chemistry* **2004,** 25, (6), 858-864.
98. Todorov, N. P.; Mancera, R. L.; Kallblad, P.; Monthoux, P., easyDock: A new docking program for high-throughput screening and binding-mode search. *Abstracts of Papers of the American Chemical Society* **2002,** 224, U345-U345.
99. Gehlhaar, D. K.; Verkhivker, G. M.; Rejto, P. A.; Sherman, C. J.; Fogel, D. B.; Fogel, L. J.; Freer, S. T., Molecular Recognition of the Inhibitor Ag-1343 by Hiv-1 Protease - Conformationally Flexible Docking by Evolutionary Programming. *Chemistry & Biology* **1995,** 2, (5), 317-324.
100. Stahl, M.; Rarey, M., Detailed analysis of scoring functions for virtual screening. *Journal of Medicinal Chemistry* **2001,** 44, (7), 1035-1042.
101. Yang, J. M.; Chen, C. C., GEMDOCK: A generic evolutionary method for molecular docking. *Proteins-Structure Function and Bioinformatics* **2004,** 55, (2), 288-304.

102. Gohlke, H.; Hendlich, M.; Klebe, G., Predicting binding modes, binding affinities and 'hot spots' for protein-ligand complexes using a knowledge-based scoring function. *Perspectives in Drug Discovery and Design* **2000,** 20, (1), 115-144.
103. Gohlke, H.; Hendlich, M.; Klebe, G., Knowledge-based scoring function to predict protein-ligand interactions. *Journal of Molecular Biology* **2000,** 295, (2), 337-356.
104. Muegge, I.; Martin, Y. C., A general and fast scoring function for protein-ligand interactions: A simplified potential approach. *Journal of Medicinal Chemistry* **1999,** 42, (5), 791-804.
105. DeWitte, R. S.; Shakhnovich, E. I., SMoG: de Novo design method based on simple, fast, and accurate free energy estimates .1. Methodology and supporting evidence. *Journal of the American Chemiçal Society* **1996,** 118, (47), 11733-11744.
106. Muegge, I., A knowledge-based scoring function for protein-ligand interactions: Probing the reference state. *Perspectives in Drug Discovery and Design* **2000,** 20, (1), 99-114.
107. Hendlich, M.; Bergner, A.; Gunther, J.; Klebe, G., Relibase: Design and development of a database for comprehensive analysis of protein-ligand interactions. *Journal of Molecular Biology* **2003,** 326, (2), 607-620.
108. Gunther, J.; Bergner, A.; Hendlich, M.; Klebe, G., Utilising structural knowledge in drug design strategies: Applications using relibase. *Journal of Molecular Biology* **2003,** 326, (2), 621-636.
109. Mizutani, M. Y.; Tomioka, N.; Itai, A., Rational Automatic Search Method for Stable Docking Models of Protein and Ligand. *Journal of Molecular Biology* **1994,** 243, (2), 310-326.
110. Verkhivker, G.; Appelt, K.; Freer, S. T.; Villafranca, J. E., Empirical Free-Energy Calculations of Ligand-Protein Crystallographic Complexes .1. Knowledge-Based Ligand-Protein Interaction Potentials Applied to the Prediction of Human-Immunodeficiency-Virus-1 Protease Binding-Affinity. *Protein Engineering* **1995,** 8, (7), 677-691.
111. DeWitte, R. S.; Ishchenko, A. V.; Shakhnovich, E. I., SMoG: De novo design method based on simple, fast, and accurate free energy estimates .2. Case studies in molecular design. *Journal of the American Chemical Society* **1997,** 119, (20), 4608-4617.
112. Mitchell, J. B. O.; Laskowski, R. A.; Alex, A.; Forster, M. J.; Thornton, J. M., BLEEP - Potential of mean force describing protein-ligand interactions: II. Calculation of binding energies and comparison with experimental data. *Journal of Computational Chemistry* **1999,** 20, (11), 1177-1185.
113. Mitchell, J. B. O.; Laskowski, R. A.; Alex, A.; Thornton, J. M., BLEEP - Potential of mean force describing protein-ligand interactions: I. Generating potential. *Journal of Computational Chemistry* **1999,** 20, (11), 1165-1176.
114. Muegge, I., Effect of ligand volume correction on PMF scoring. *Journal of Computational Chemistry* **2001,** 22, (4), 418-425.
115. Muegge, I.; Martin, Y. C.; Hajduk, P. J.; Fesik, S. W., Evaluation of PMF scoring in docking weak ligands to the FK506 binding protein. *Journal of Medicinal Chemistry* **1999,** 42, (14), 2498-2503.
116. Muegge, I., Protein-ligand docking with a knowledge-based scoring function. *Abstracts of Papers of the American Chemical Society* **1999,** 218, U493-U493.
117. Muegge, I. A., PMF scoring revisited. *Abstracts of Papers of the American Chemical Society* **2004,** 228, U515-U515.

118. Ishchenko, A. V.; Shakhnovich, E. I., SMall molecule growth 2001 (SMoG2001): An improved knowledge-based scoring function for protein-ligand interactions. *Journal of Medicinal Chemistry* **2002,** 45, (13), 2770-2780.
119. Deng, Z.; Chuaqui, C.; Singh, J., Structural interaction fingerprint (SIFt): A novel method for analyzing three-dimensional protein-ligand binding interactions. *Journal of Medicinal Chemistry* **2004,** 47, (2), 337-344.
120. Chuaqui, C.; Deng, Z.; Singh, J., Interaction profiles of protein kinase-inhibitor complexes and their application to virtual screening. *Journal of Medicinal Chemistry* **2005,** 48, (1), 121-133.
121. Verdonk, M. L.; Berdini, V.; Hartshorn, M. J.; Mooij, W. T. M.; Murray, C. W.; Taylor, R. D.; Watson, P., Virtual screening using protein-ligand docking: Avoiding artificial enrichment. *Journal of Chemical Information and Computer Sciences* **2004,** 44, (3), 793-806.
122. Velec, H. F. G.; Gohlke, H.; Klebe, G., DrugScore(CSD)-knowledge-based scoring function derived from small molecule crystal data with superior recognition rate of near-native ligand poses and better affinity prediction. *Journal of Medicinal Chemistry* **2005,** 48, (20), 6296-6303.
123. Feher, M.; Deretey, E.; Roy, S., BHB: A simple knowledge-based scoring function to improve the efficiency of database screening. *Journal of Chemical Information and Computer Sciences* **2003,** 43, (4), 1316-1327.
124. Muryshev, A. E.; Tarasov, D. N.; Butygin, A. V.; Butygina, O. Y.; Aleksandrov, A. B.; Nikitin, S. M., A novel scoring function for molecular docking. *Journal of Computer-Aided Molecular Design* **2003,** 17, 597-605.
125. Charifson, P. S.; Corkery, J. J.; Murcko, M. A.; Walters, W. P., Consensus scoring: A method for obtaining improved hit rates from docking databases of three-dimensional structures into proteins. *Journal of Medicinal Chemistry* **1999,** 42, (25), 5100-5109.
126. Wang, R. X.; Wang, S. M., How does consensus scoring work for virtual library screening? An idealized computer experiment. *Journal of Chemical Information and Computer Sciences* **2001,** 41, (5), 1422-1426.
127. Wang, R. X.; Lai, L. H.; Wang, S. M., Further development and validation of empirical scoring functions for structure-based binding affinity prediction. *Journal of Computer-Aided Molecular Design* **2002,** 16, (1), 11-26.
128. Wang, R. X.; Liu, L.; Lai, L. H.; Tang, Y. Q., SCORE: A new empirical method for estimating the binding affinity of a protein-ligand complex. *Journal of Molecular Modeling* **1998,** 4, (12), 379-394.
129. Guo, J. X.; Hurley, M. M.; Wright, J. B.; Lushington, G. H., A docking score function for estimating ligand-protein interactions: Application to acetylcholinesterase inhibition. *Journal of Medicinal Chemistry* **2004,** 47, (22), 5492-5500.
130. Jacobsson, M.; Liden, P.; Stjernschantz, E.; Bostrom, H.; Norinder, U., Improving structure-based virtual screening by multivariate analysis of scoring data. *Journal of Medicinal Chemistry* **2003,** 46, (26), 5781-5789.
131. Klon, A. E.; Glick, M.; Thoma, M.; Acklin, P.; Davies, J. W., Finding more needles in the haystack: A simple and efficient method for improving high-throughput docking results. *Journal of Medicinal Chemistry* **2004,** 47, (11), 2743-2749.
132. Bissantz, C.; Bernard, P.; Hibert, M.; Rognan, D., Protein-based virtual screening of chemical databases. II. Are homology models of G-protein coupled receptors suitable targets? *Proteins-Structure Function and Genetics* **2003,** 50, (1), 5-25.

133. Wang, R. X.; Fang, X. L.; Lu, Y. P.; Yang, C. Y.; Wang, S. M., The PDBbind database: Methodologies and updates. *Journal of Medicinal Chemistry* **2005,** 48, (12), 4111-4119.
134. Ferrara, P.; Gohlke, H.; Price, D. J.; Klebe, G.; Brooks, C. L., Assessing scoring functions for protein-ligand interactions. *Journal of Medicinal Chemistry* **2004,** 47, (12), 3032-3047.
135. Roche, O.; Kiyama, R.; Brooks, C. L., Ligand-Protein DataBase: Linking protein-ligand complex structures to binding data. *Journal of Medicinal Chemistry* **2001,** 44, (22), 3592-3598.
136. Krovat, E. M.; Langer, T., Impact of scoring functions on enrichment in docking-based virtual screening: An application study on renin inhibitors. *Journal of Chemical Information and Computer Sciences* **2004,** 44, (3), 1123-1129.
137. Vigers, G. P. A.; Rizzi, J. P., Multiple active site corrections for docking and virtual screening. *Journal of Medicinal Chemistry* **2004,** 47, (1), 80-89.
138. Lipinski, C. A.; Lombardo, F.; Dominy, B. W.; Feeney, P. J., Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Advanced Drug Delivery Reviews* **2001,** 46, (1-3), 3-26.
139. Lipinski, C. A.; Lombardo, F.; Dominy, B. W.; Feeney, P. J., Experimental and computational approaches to estimate solubility and permeability in drug discovery and development settings. *Advanced Drug Delivery Reviews* **1997,** 23, (1-3), 3-25.
140. Nissink, J. W. M.; Murray, C.; Hartshorn, M.; Verdonk, M. L.; Cole, J. C.; Taylor, R., A new test set for validating predictions of protein-ligand interaction. *Proteins-Structure Function and Genetics* **2002,** 49, (4), 457-471.
141. Bindewald, E.; Skolnick, J., A scoring function for docking ligands to low-resolution protein structures. *Journal of Computational Chemistry* **2005,** 26, (4), 374-383.
142. Wang, R. X.; Fang, X. L.; Lu, Y. P.; Wang, S. M., The PDBbind database: Collection of binding affinities for protein-ligand complexes with known three-dimensional structures. *Journal of Medicinal Chemistry* **2004,** 47, (12), 2977-2980.
143. Taylor, R., Life-science applications of the Cambridge Structural Database. *Acta Crystallographica Section D-Biological Crystallography* **2002,** 58, 879-888.
144. Available Chemical Directory. http://www.mdli.com/products/experiment/available_chem_dir/index.jsp
145. Bohm, H. J., Towards the automatic design of synthetically accessible protein ligands: Peptides, amides and peptidomimetics. *Journal of Computer-Aided Molecular Design* **1996,** 10, (4), 265-272.
146. World Drug Index http://scientific.thomson.com/products/wdi/
147. Laskowski, R. A.; Chistyakov, V. V.; Thornton, J. M., PDBsum more: new summaries and analyses of the known 3D structures of proteins and nucleic acids. *Nucleic Acids Research* **2005,** 33, D266-D268.
148. Laskowski, R. A.; Hutchinson, E. G.; Michie, A. D.; Wallace, A. C.; Jones, M. L.; Thornton, J. M., PDBsum: a Web-based database of summaries and analyses of all PDB structures. *Trends in Biochemical Sciences* **1997,** 22, (12), 488-490.
149. Laskowski, R. A., PDBsum: summaries and analyses of PDB structures. *Nucleic Acids Research* **2001,** 29, (1), 221-222.

150. Bruno, I. J.; Cole, J. C.; Lommerse, J. P. M.; Rowland, R. S.; Taylor, R.; Verdonk, M. L., IsoStar: A library of information about nonbonded interactions. *Journal of Computer-Aided Molecular Design* **1997,** 11, (6), 525-537.
151. Verdonk, M. L.; Cole, J. C.; Taylor, R., SuperStar: A knowledge-based approach for identifying interaction sites in proteins. *Journal of Molecular Biology* **1999,** 289, (4), 1093-1108.
152. Boer, D. R.; Kroon, J.; Cole, J. C.; Smith, B.; Verdonk, M. L., SuperStar: Comparison of CSD and PDB-based interaction fields as a basis for the prediction of protein-ligand interactions. *Journal of Molecular Biology* **2001,** 312, (1), 275-287.
153. Verdonk, M. L.; Cole, J. C.; Watson, P.; Gillet, V.; Willett, P., SuperStar: Improved knowledge-based interaction fields for protein binding sites. *Journal of Molecular Biology* **2001,** 307, (3), 841-859.
154. Deshpande, N.; Addess, K. J.; Bluhm, W. F.; Merino-Ott, J. C.; Townsend-Merino, W.; Zhang, Q.; Knezevich, C.; Xie, L.; Chen, L.; Feng, Z. K.; Green, R. K.; Flippen-Anderson, J. L.; Westbrook, J.; Berman, H. M.; Bourne, P. E., The RCSB Protein Data Bank: a redesigned query system and relational database based on the mmCIF schema. *Nucleic Acids Research* **2005,** 33, D233-D237.
155. Bourne, P. E.; Westbrook, J.; Berman, H. M., The Protein Data Bank and lessons in data management. *Briefings in Bioinformatics* **2004,** 5, (1), 23-30.
156. Westbrook, J.; Feng, Z. K.; Jain, S.; Bhat, T. N.; Thanki, N.; Ravichandran, V.; Gilliland, G. L.; Bluhm, W.; Weissig, H.; Greer, D. S.; Bourne, P. E.; Berman, H. M., The Protein Data Bank: Unifying the archive. *Nucleic Acids Research* **2002,** 30, (1), 245-248.
157. Bernstein, H. J.; Bernstein, F. C.; Bourne, P. E., CIF applications. VIII. pdb2cif: translating PDB entries into mmCIF format. *Journal of Applied Crystallography* **1998,** 31, 282-295.
158. Wallace, A. C.; Laskowski, R. A.; Thornton, J. M., Ligplot - a Program to Generate Schematic Diagrams of Protein Ligand Interactions. *Protein Engineering* **1995,** 8, (2), 127-134.
159. Hutchinson, E. G.; Thornton, J. M., PROMOTIF - A program to identify and analyze structural motifs in proteins. *Protein Science* **1996,** 5, (2), 212-220.
160. 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* **1993,** 26, 283-291.
161. Pearl, F. M. G.; Lee, D.; Bray, J. E.; Buchan, D. W. A.; Shepherd, A. J.; Orengo, C. A., The CATH extended protein-family database: Providing structural annotations for genome sequences. *Protein Science* **2002,** 11, (2), 233-244.
162. Pearl, F. M. G.; Bennett, C. F.; Bray, J. E.; Harrison, A. P.; Martin, N.; Shepherd, A.; Sillitoe, I.; Thornton, J.; Orengo, C. A., The CATH database: an extended protein family resource for structural and functional genomics. *Nucleic Acids Research* **2003,** 31, (1), 452-455.
163. Pearl, F.; Todd, A.; Sillitoe, I.; Dibley, M.; Redfern, O.; Lewis, T.; Bennett, C.; Marsden, R.; Grant, A.; Lee, D.; Akpor, A.; Maibaum, M.; Harrison, A.; Dallman, T.; Reeves, G.; Diboun, I.; Addou, S.; Lise, S.; Johnston, C.; Sillero, A.; Thornton, J.; Orengo, C., The CATH Domain Structure Database and related resources Gene3D and DHS provide comprehensive domain family information for genome analysis. *Nucleic Acids Research* **2005,** 33, D247-D251.

164. Orengo, C. A.; Michie, A. D.; Jones, S.; Jones, D. T.; Swindells, M. B.; Thornton, J. M., CATH - a hierarchic classification of protein domain structures. *Structure* **1997,** 5, (8), 1093-1108.
165. Orengo, C. A.; Todd, A. E.; Thornton, J. M., From protein structure to function. *Current Opinion in Structural Biology* **1999,** 9, (3), 374-382.
166. Orengo, C. A.; Martin, A. M.; Hutchinson, G.; Jones, S.; Jones, D. T.; Michie, A. D.; Swindells, M. B.; Thornton, J. M., Classifying a protein in the CATH database of domain structures. *Acta Crystallographica Section D-Biological Crystallography* **1998,** 54, 1155-1167.
167. Orengo, C. A.; Pearl, F. M. G.; Bray, J. E.; Todd, A. E.; Martin, A. C.; Lo Conte, L.; Thornton, J. M., The CATH database provides insights into protein structure/function relationships. *Nucleic Acids Research* **1999,** 27, (1), 275-279.
168. Orengo, C. A.; Pearl, F.; Lee, D.; Bray, J.; Todd, I.; Thornton, J. M., Insights into protein evolution and function from the CATH structural database. *Biophysical Journal* **2001,** 80, (1), 32a-32a.
169. Orengo, C. A.; Bray, J. E.; Buchan, D. W. A.; Harrison, A.; Lee, D.; Pearl, F. M. G.; Sillitoe, I.; Todd, A. E.; Thornton, J. M., The CATH protein family database: A resource for structural and functional annotation of genomes. *Proteomics* **2002,** 2, (1), 11-21.
170. Pearl, F.; Todd, A. E.; Bray, J. E.; Martin, A. C. R.; Salamov, A. A.; Suwa, M.; Swindells, M. B.; Thornton, J. M.; Orengo, C. A., Using the CATH domain database to assign structures and functions to the genome sequences. *Biochemical Society Transactions* **2000,** 28, 269-275.
171. Pearl, F. M. G.; Lee, D.; Bray, J. E.; Sillitoe, I.; Todd, A. E.; Harrison, A. P.; Thornton, J. M.; Orengo, C. A., Assigning genomic sequences to CATH. *Nucleic Acids Research* **2000,** 28, (1), 277-282.
172. Pearl, F. M. G.; Martin, N.; Bray, J. E.; Buchan, D. W. A.; Harrison, A. P.; Lee, D.; Reeves, G. A.; Shepherd, A. J.; Sillitoe, I.; Todd, A. E.; Thornton, J. M.; Orengo, C. A., A rapid classification protocol for the CATH Domain Database to support structural genomics. *Nucleic Acids Research* **2001,** 29, (1), 223-227.
173. Murzin, A. G.; Brenner, S. E.; Hubbard, T.; Chothia, C., Scop - a Structural Classification of Proteins Database for the Investigation of Sequences and Structures. *Journal of Molecular Biology* **1995,** 247, (4), 536-540.
174. Brenner, S. E.; Chothia, C.; Hubbard, T. J. P.; Murzin, A. G., Understanding protein structure: Using scop for fold interpretation. *Computer Methods for Macromolecular Sequence Analysis* **1996,** 266, 635-643.
175. Hubbard, T. J. P.; Murzin, A. G.; Brenner, S. E.; Chothia, C., SCOP: A structural classification of proteins database. *Nucleic Acids Research* **1997,** 25, (1), 236-239.
176. Hubbard, T. J. P.; Ailey, B.; Brenner, S. E.; Murzin, A. G.; Chothia, C., SCOP, structural classification of proteins database: Applications to evaluation of the effectiveness of sequence alignment methods and statistics of protein structural data. *Acta Crystallographica Section D-Biological Crystallography* **1998,** 54, 1147-1154.
177. Hubbard, T. J. P.; Ailey, B.; Brenner, S. E.; Murzin, A. G.; Chothia, C., SCOP: a structural classification of proteins database. *Nucleic Acids Research* **1999,** 27, (1), 254-256.
178. Lo Conte, L.; Ailey, B.; Hubbard, T. J. P.; Brenner, S. E.; Murzin, A. G.; Chothia, C., SCOP: a Structural Classification of Proteins database. *Nucleic Acids Research* **2000,** 28, (1), 257-259.

179. Lo Conte, L.; Brenner, S. E.; Hubbard, T. J. P.; Chothia, C.; Murzin, A. G., SCOP database in 2002: refinements accommodate structural genomics. *Nucleic Acids Research* **2002,** 30, (1), 264-267.
180. Andreeva, A.; Howorth, D.; Brenner, S. E.; Hubbard, T. J. P.; Chothia, C.; Murzin, A. G., SCOP database in 2004: refinements integrate structure and sequence family data. *Nucleic Acids Research* **2004,** 32, D226-D229.
181. Apweiler, R.; Attwood, T. K.; Bairoch, A.; Bateman, A.; Birney, E.; Biswas, M.; Bucher, P.; Cerutti, L.; Corpet, F.; Croning, M. D. R.; Durbin, R.; Falquet, L.; Fleischmann, W.; Gouzy, J.; Hermjakob, H.; Hulo, N.; Jonassen, I.; Kahn, D.; Kanapin, A.; Karavidopoulou, Y.; Lopez, R.; Marx, B.; Mulder, N. J.; Oinn, T. M.; Pagni, M.; Servant, F.; Sigrist, C. J. A.; Zdobnov, E. M.; Consortium, I.; InterPro - an integrated documentation resource for protein families, domains and functional sites. *Bioinformatics* **2000,** 16, (12), 1145-1150.
182. Bateman, A.; Coin, L.; Durbin, R.; Finn, R. D.; Hollich, V.; Griffiths-Jones, S.; Khanna, A.; Marshall, M.; Moxon, S.; Sonnhammer, E. L. L.; Studholme, D. J.; Yeats, C.; Eddy, S. R., The Pfam protein families database. *Nucleic Acids Research* **2004,** 32, D138-D141.
183. Bateman, A.; Birney, E.; Cerruti, L.; Durbin, R.; Etwiller, L.; Eddy, S. R.; Griffiths-Jones, S.; Howe, K. L.; Marshall, M.; Sonnhammer, E. L. L., The Pfam Protein Families Database. *Nucleic Acids Research* **2002,** 30, (1), 276-280.
184. Bateman, A.; Birney, E.; Durbin, R.; Eddy, S. R.; Howe, K. L.; Sonnhammer, E. L. L., The Pfam protein families database. *Nucleic Acids Research* **2000,** 28, (1), 263-266.
185. Bateman, A.; Birney, E.; Durbin, R.; Eddy, S. R.; Finn, R. D.; Sonnhammer, E. L. L., Pfam 3.1: 1313 multiple alignments and profile HMMs match the majority of proteins. *Nucleic Acids Research* **1999,** 27, (1), 260-262.
186. Sonnhammer, E. L. L.; Eddy, S. R.; Birney, E.; Bateman, A.; Durbin, R., Pfam: multiple sequence alignments and HMM-profiles of protein domains. *Nucleic Acids Research* **1998,** 26, (1), 320-322.
187. Sonnhammer, E. L. L.; Eddy, S. R.; Durbin, R., Pfam: A comprehensive database of protein domain families based on seed alignments. *Proteins-Structure Function and Genetics* **1997,** 28, (3), 405-420.
188. National Cancer Institute Open3D-mol. http://129.43.27.140/ncidb2/
189. Wang, R. X.; Fang, X. L.; Lu, Y. P.; Yang, C. Y.; Wang, S. M., Collection of binding affinities for protein-ligand complexes with known 3-D structures: The PDBbind database. *Abstracts of Papers of the American Chemical Society* **2004,** 228, U527-U527.
190. Goldberg, D. E., *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional: 1989; p 432.
191. Lipinski, C. A., Drug-like properties and the causes of poor solubility and poor permeability. *Journal of Pharmacological and Toxicological Methods* **2000,** 44, (1), 235-249.
192. Ajay; Walters, W. P.; Murcko, M. A., Can we learn to distinguish between "drug-like" and "nondrug-like" molecules? *Journal of Medicinal Chemistry* **1998,** 41, (18), 3314-3324.
193. Oprea, T. I., Current trends in lead discovery: Are we looking for the appropriate properties? *Journal of Computer-Aided Molecular Design* **2002,** 16, (5-6), 325-334.
194. Muegge, I., Pharmacophore features of potential drugs. *Chemistry-a European Journal* **2002,** 8, (9), 1976-1981.

195. Muegge, I., Selection criteria for drug-like compounds. *Medicinal Research Reviews* **2003,** 23, (3), 302-321.
196. Brown, F., Chemoinformatics - a ten year update - Editorial opinion. *Current Opinion in Drug Discovery & Development* **2005,** 8, (3), 298-302.
197. Jain, A. K.; Murty, M. N.; Flynn, P. J., Data clustering: A review. *Acm Computing Surveys* **1999,** 31, (3), 264-323.
198. Kontoyianni, M.; Sokol, G. S.; McClellan, L. M., Evaluation of library ranking efficacy in virtual screening. *Journal of Computational Chemistry* **2005,** 26, (1), 11-22.
199. Warren, G. L.; Andrews, W.; Capelli, A. M.; Clarke, B. P.; LaLonde, J. M.; Lambert, M. H.; Lindvall, M.; Nevins, N.; Peishoff, C. E.; Semus, S. F.; Senger, S.; Tedesco, G.; Wall, I. D.; Woolven, J. M.; Head, M. S., Critical assessment of docking programs and scoring functions. *Abstracts of Papers of the American Chemical Society* **2004,** 228, U513-U514.
200. Chen, H.; Lyne, P. D.; Giordanetto, F.; Lovell, T.; Li, J., On evaluating molecular-docking methods for pose prediction and enrichment factors. *J Chem Inf Model* **2006,** 46, (1), 401-15.
201. CCDC Using GOLD for Structure-based Virtual Screening. http://www.ccdc.cam.ac.uk/products/life_sciences/validate/gold_validation_virtual_screening/
202. Tannenbaum, T.; Litzkow, M., The Condor Distributed-Processing System. *Dr Dobbs Journal* **1995,** 20, (2), 40-&.
203. Thain, D.; Tannenbaum, T.; Livny, M., Distributed computing in practice: the Condor experience. *Concurrency and Computation-Practice & Experience* **2005,** 17, (2-4), 323-356.
204. Hinsen, K., The Molecular Modeling Toolkit: a case study of a large scientific application in Python. In *Proceedings of the 6th International Python Conference*, 1997.
205. DeLano, W. L., The PyMOL Molecular Graphics System. In 2002.
206. Still, W. C.; Tempczyk, A.; Hawley, R. C.; Hendrickson, T., Semianalytical Treatment of Solvation for Molecular Mechanics and Dynamics. *Journal of the American Chemical Society* **1990,** 112, (16), 6127-6129.
207. Massova, I.; Kollman, P. A., Combined molecular mechanical and continuum solvent approach (MM-PBSA/GBSA) to predict ligand binding. *Perspectives in Drug Discovery and Design* **2000,** 18, 113-135.
208. Creighton, T. E., *Proteins: Structures and Molecular Properties.* 2 ed.; Freeman: 1993; p 505.
209. Muller, N., Search for a realistic view of the Hydrophobic Effect. *Accounts of Chemical Research* **1990,** 23, 23-28.
210. Privalov, P. L. a. G., S. J., The Hydrophobic Effect: a Reappraisal. *Pure and Applied Chemistry* **1989,** 61, 1097-1104.
211. Lee, B.; Richards, F. M., Interpretation of Protein Structures - Estimation of Static Accessibility. *Journal of Molecular Biology* **1971,** 55, (3), 379-&.
212. Richmond, T. J.; Richards, F. M., Packing of Alpha-Helices - Geometrical Constraints and Contact Areas. *Journal of Molecular Biology* **1978,** 119, (4), 537-555.
213. Gilson, M. K.; Gilson, H. S. R.; Potter, M. J., Fast assignment of accurate partial atomic charges: An electronegativity equalization method that accounts for alternate resonance forms. *Journal of Chemical Information and Computer Sciences* **2003,** 43, (6), 1982-1997.

214. Michel, J.; Taylor, R. D.; Essex, J. W., The parameterization and validation of generalized born models using the pairwise descreening approximation. *Journal of Computational Chemistry* **2004**, 25, (14), 1760-1770.
215. Michel, J.; Taylor, R. D.; Essex, J. W., Efficient generalized Born models for Monte Carlo simulations. *Journal of Chemical Theory and Computation* **2006**, 2, (3), 732-739.
216. Pearlman, D. A.; Case, D. A.; Caldwell, J. W.; Ross, W. S.; Cheatham, T. E.; Debolt, S.; Ferguson, D.; Seibel, G.; Kollman, P., Amber, a Package of Computer-Programs for Applying Molecular Mechanics, Normal-Mode Analysis, Molecular-Dynamics and Free-Energy Calculations to Simulate the Structural and Energetic Properties of Molecules. *Computer Physics Communications* **1995**, 91, (1-3), 1-41.
217. Rogers, D.; Hopfinger, A. J., Application of Genetic Function Approximation to Quantitative Structure-Activity-Relationships and Quantitative Structure-Property Relationships. *Journal of Chemical Information and Computer Sciences* **1994**, 34, (4), 854-866.
218. Wall, M., GALib v2.4.6. In 2005.
219. TrollTech Qt - CODE LESS. CREATE MORE. http://www.trolltech.com/products/qt
220. The R Project for Statistical Computing. http://www.r-project.org/
221. Dawkins, R., *The Blind Watchmaker: Why the Evidence of Evolution Reveals a Universe Without Design*. Penguin Books Ltd: 1990; p 368.
222. Tolstoy, L., *Anna Karenina*. 1875-77.
223. D. R. FLOWER; ACT. NORTH; ATTWOOD, T. K., Structure and sequence relationships in the lipocalins and related proteins. *Protein Science* **1993**, 2, (5), 753-761.
224. Flower DR; North AC; CE., S., The lipocalin protein family: structural and sequence overview. *Biochim Biophys Acta* **2000**, 1482, (1-2), 9-24.