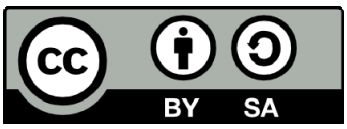
UNIVERSITAT DE BARCELONA

New computational methods for structural modeling protein-protein and protein-nucleic acid interactions

Luis Ángel Rodríguez Lumbreras



UNIVERSITAT DE BARCELONA

FACULTAT DE BIOLOGIA
(Supervisor: Juan Fernández Recio)

New computational methods for structural modeling of protein-protein and protein-nucleic acid interactions

LUIS ÁNGEL RODRÍGUEZ LUMBRERAS

UNIVERSITAT DE BARCELONA

FACULTAT DE BIOLOGIA

DOCTORAT EN BIOMEDICINA

Código HDK05

New computational methods for structural modeling of protein-protein and protein-nucleic acid interactions

Memoria presentada por Luis Ángel Rodríguez Lumbreras para optar al título de Doctor por la Universidad de Barcelona

Tesis doctoral realizada bajo la dirección del Doctor Juan Fernández Recio, en el Barcelona Supercomputing Center (BSC) y los dos últimos años en el Instituto de Ciencias de la Vid y del Vino (ICVV-CSIC).

Tesis adscrita al Departamento de Bioquímica y Biología Molecular de la Facultad de Biología (Universitat de Barcelona).

Director	Tutor	Ph.D. candidate
Dr. Juan Fernández Recio	Dr. Josep Lluís Gelpí Buchaca	Luis Ángel Rodríguez Lumbreras

Barcelona, 2022

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

CSIC
Consejo Superior de Investigaciones Científicas

ICVV

Universitat de Barcelona

DECLARACIÓN DE ORIGINALIDAD

Yo, LUIS ANGEL RODRIGUEZ LUMBRERAS, matriculado en el programa de doctorado de BIOMEDICINA de la Universidad de Barcelona declaro que la tesis titulada “New computational methods for structural modeling of protein-protein and protein-nucleic acid interactions” es original, que la investigación que voy a realizar cumple con los códigos éticos y las buenas prácticas y que la tesis no incluye plagio. Soy consciente y acepto por escrito que mi tesis se someterá al proceso adecuado para probar la originalidad de mis resultados.

31 de Octubre de 2022

A mis padres

"Ninguna ciencia, en cuanto a ciencia, engaña; el engaño está en quien no la sabe."

Miguel de Cervantes

Acknowledgments

Primeramente, estoy muy agradecido a Juan por todos los conocimientos transmitidos todos estos años, todos sus consejos y sugerencias, toda la ayuda y apoyo, así como su paciencia y su gran capacidad investigadora. De nuevo, muchas gracias por hacer posible este proceso duro pero enriquecedor.

Seguidamente, muchas gracias a todos los compañeros del BSC y del ICVV. Siempre es un orgullo trabajar con ustedes. Especial agradecimiento a Brian y Miguel por su apoyo cuando empecé en esto de escribir código. Una especial mención a todos los compañeros del grupo que tengo especiales recuerdos y aprecio, porque cada uno me ha aportado algo, Didier, Dàmaris, Jorge, Ha, Bruno, Sergio, Laura, Lucia, Chiara, Silvia y Augustina. ¡Un agradecimiento especial a Carmen Castell por hacer posible una colaboración con investigadores de mi alma mater! ¡Otro agradecimiento especial va para Mireia, una gran compañera, se te echa de menos por el laboratorio! ¡Y el último agradecimiento especial va para Sergio, con el aprendí mucho durante mi tesina de Máster y primer año de doctorado!

Por último, dar las gracias también a todos los compañeros de Life en BSC. Gracias por las comidas, cafés, y momentos de desconexión Pau, Dani G, Rti, Enrico, Dani L y Pierlauro. Paralelamente esta tesis no hubiera sido posible sin el apoyo incondicional de toda la gente que me quiere y me aprecia, a todos vosotros, muchísimas gracias por compartir risas, momentos y fuerzas.

Por último, gracias infinitas a mi familia, a cada uno de vosotros, gracias.

Abstract

The study of the 3D structural details of protein-protein and protein-DNA interactions is essential to understand biomolecular functions at the molecular level. Given the difficulty of the structural determination of these complexes by experimental techniques, computational tools are becoming a powerful to increase the actual structural coverage of protein-protein and protein-DNA interactions. pyDock is one of these tools, which uses its scoring function to determine the quality of models generated by other tools. pyDock is usually combined with the model sampling methods FTDOCK or ZDOCK. This combination has shown a consistently good prediction performance in community-wide assessment experiments like CAPRI or CASP and has provided biological insights and insightful interpretation of experiments by modeling many biomolecular interactions of biomedical and biotechnological interest. This software combination has demonstrated good predictive performance in the blinded evaluation experiments CAPRI and CASP. It has provided biological insights by modeling many biomolecular interactions of biomedical and biotechnological interest.

Here, we describe a pyDock software update, which includes its adaptation to the newest python code, the capability of including cofactor and other small molecules, and an internal parallelization to use the computational resources more efficiently.

A strategy was designed to integrate the template-based docking and *ab initio* docking approaches by creating a new scoring function based on the pyDock scoring energy basis function and the TM-score measure of structural similarity of protein structures. This strategy was partially used for our participation in the 7th CAPRI, the 3rd CASP-CAPRI and the 4th CASP-CAPRI joint experiments. These experiments were challenging, as we needed to model protein-protein complexes, multimeric oligomerization proteins, protein-peptide, and protein-oligosaccharide interactions. Many proposed targets required the efficient integration of rigid-body docking, template-based modeling, flexible optimization, multi-parametric scoring, and experimental restraints. This was especially relevant for the multi-molecular assemblies proposed in the 3er and 4th CASP-CAPRI joint experiments.

In addition, a case study, in which electron transfer protein complexes were modelled to test the software new capabilities. Good results were achieved as the structural models obtained help explaining the differences in photosynthetic efficiency between red and green algae.

Contents

1. INTRODUCTION

1.1. Biomolecules

1.1.1. From genes to proteins

Living organisms can be seen as data storage machines that fight against entropy to conserve and transmit information. As a consequence, the classic functions that define a living organism, i.e. nutrition, relationship, and reproduction, emerge from this struggle.

Therefore, the molecule responsible for storing the information must be stable and have a large storage capacity. Chemical and biological evolution selected deoxyribonucleic acid (DNA) as such molecule [1]. It is a polymer formed by deoxyribonucleotides. The deoxyribonucleotides are composed of a nitrogenous base, a deoxyribose (sugar), and a phosphate group. The nitrogenous base can be derived from purine, such as adenine (A) and guanine (G), or from pyrimidine, such as cytosine (C) and thymine (T). Phosphate groups form the backbone of DNA, binding the nucleotides together and giving directionality to the polymeric chain. The oxygen in the phosphate is covalently attached to the C5' position of the sugar. The phosphate group then binds to the next nucleotide at the sugar's C3' position, giving a 5' to 3' directionality and forming a single-stranded polymer.

However, this polymeric molecule alone would not be stable enough for long time information storing. This stability is achieved by non-covalent binding of two single-stranded DNA polymers, which can adopt a stable double helix conformation, according to a highly specific base complementarity. The bases A and T must face each other by forming two hydrogen bonds, while G and C bases face each other by forming three hydrogen bonds. The repetition of these base pairs builds a double-stranded DNA. The information is encoded in triplets (three consecutive nucleotides of the sequence) or codons, which code up to 20 types of amino acids. Amino acids form a different kind of polymer, proteins

1.1.2. DNA structure

The combination of Chargaff's rules (A+G=T+C) and X-ray study of the sodium salt of DNA of Rosalind Franklin, lead James Watson and Francis Crick to discover the DNA double helix structure [2]. As can be seen in Figure 1.1, the double helix is formed by two chains in an antiparallel arrangement. In addition, it is possible to observe the structure of the major

and minor grooves. The major groove is essential in proteins-DNA interactions (see Chapter 1.3.2).

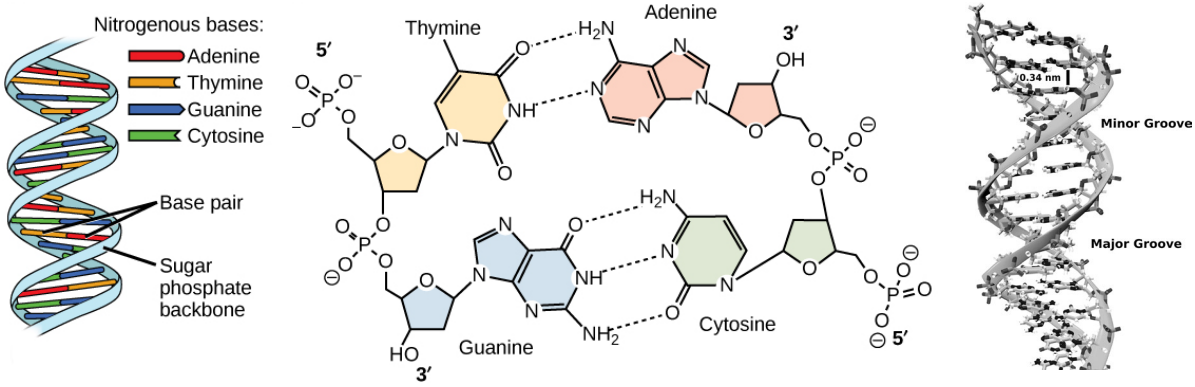


Figure 1.1. Left panel shows the double helix structure of DNA. Center panel shows the hydrogen bonds between paired bases. Right panel shows the major and minor grooves, which are binding sites for DNA binding proteins during transcription (copying RNA from DNA) and replication. Image modified from "DNA structure and sequencing:" by OpenStax College, Biology (CC BY 3.0).

In addition to this canonical helical structure (B-DNA), the DNA, can present other configurations so called A-DNA and Z-DNA.

The helix sense of the A-DNA is right-handed like the B-DNA. The A-DNA is generally wider and shorter with a narrow and deep major groove, as opposed to the minor groove, and like the B-DNA, the N-glycosidic linkage is anti. On the other hand, the Z-DNA has an entirely different structure, starting with a left-handed helix sense. This molecule is generally the narrowest and longest of the three mentioned conformations, where the major groove is shallow, and the minor groove is narrow and deep (see Figure 1.2).

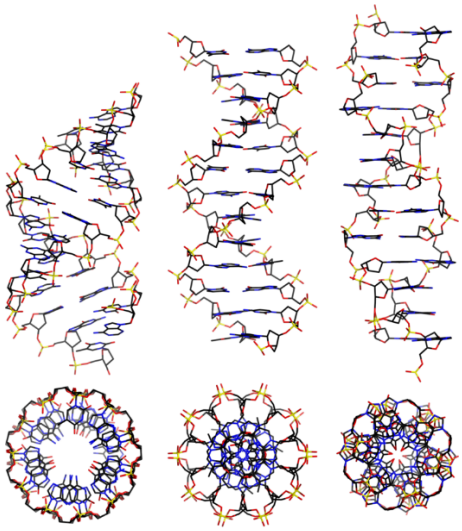


Figure 1.2. Side and top view of A-, B-, and Z-DNA conformations. By Mauroesguerroto - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=35919357

DNA can have alternative structures, such as DNA Bubble, slipped loop, cruciforms, H-DNA, and G-quadruplex/i-motif in double-stranded DNA. All of these structures are important, as the DNA conformation defines the type of protein interactions in which it is involved.

1.1.3. Protein structure

There are four levels of structural complexity in proteins.

Primary structure refers to the sequence of amino acids that form a polypeptidic chain, defined from the N-terminal to the C-terminal. This is the order in which the ribosomes form the peptide bonds. The amino acid sequence depends on the DNA sequence codons that form the specific gen.

The secondary structure of proteins is formed by local interactions between amino acid residues that are nearly located in the polypeptide chain. Hydrogen bonds between the backbone NH and CO groups stabilise those local folding events. This leads to the two most common types of secondary structure: alpha helices and beta sheets. Such periodic structures were predicted by Linus Pauling and Robert Carey in 1951, years before they were experimentally confirmed [3].

Tertiary structure is formed by the interaction between different secondary structure elements in the 3D space, and it is stabilized by hydrogen bonds, van der Waals, salt bridges, hydrophobic packing, and disulphide bonding.

The quaternary structure of proteins can be defined as the interaction between different polypeptides that have a tertiary structure and interact with each other.

In general, the sequence of amino acids of a protein determines its 3D structure. While the majority of proteins have tertiary structure, there are many proteins that are partially or even totally unstructured [4]. In many cases, disordered regions can become structured when interacting with another protein or biomolecule [5].

1.2. Protein-protein complexes

1.2.1. Importance of protein interactions

Protein-protein interactions mediate most cellular functions. In this regard, the interactome is the complex and dynamic network formed by the entire set of protein interactions in a living organism or a cell. Mapping the interactome would provide insights into how biological systems work, and how they could be modulated. Recent studies have made progress towards the description of this intricate network of interactions using proteome-wide techniques [6, 7], although it is estimated that the real number of protein-protein interactions are much larger than currently known, between 130,000 and 600,000 [8, 9]. All data from these experiments have been collected in several IPP databases such as: DIP [10], MPIDB [11], HPRD [12], BIND [13], MINT [14] , MatrixDB [15], STRINGS v10[16], BioGRID [17], Reactome[18], etc. In addition to these databases, there is Interactome 3D and MIntAct project [19], which try to collet several databases centralising the information in one place.

However, although these initiatives are systematically extending the number of known interacting protein pairs, they have difficulty providing the structural architecture of these interactions, which is essential to unveil the underlying molecular mechanisms that maintain health or contribute to disease. Such structural knowledge may ultimately lead to

the rational design of new drugs, and the interpretation and design of relevant mutations for biomedical or biotechnological purposes.

1.2.2. Structural characterization of protein-protein complexes

Biophysical methods, mainly X-ray crystallography [20], NMR spectroscopy [21] and cryo-electron microscopy (cryo-EM) [22, 23], can be used to solve protein–protein interactions at atomic resolution. These methods have determined de 3D structure of a large number of complexes, so many studies have been reported to analyze unique characteristics of protein-protein interactions.

But it has been difficult to find general structural and energetics features for all protein-protein complexes. It has been found that in stable complexes, binding are driven by hydrophobic interactions, as compared to more transient complexes [24] that are more hydrophilic. Stable complexes show more packed interfaces, with fewer hydrogen bonds between the subunits than the less stable complexes.

According to several studies, protein interfaces are dominated by aromatic (Phe, His, Try) and aliphatic (Met, Val, Ile, Leu) residues, and with the exception of arginine, they are depleted in charged residues. Interestingly, arginine is one of the residues that appear most often at the interfaces. The area values of known protein-protein interfaces follow a normal distribution that peaks in the range of 600-800 $Å^2$ [25], suggesting that complex formation requires a minimum number of contacts, in addition to the displacement of water molecules [26].

Another area of research focused on the energetic contributions to the binding affinity. Hydrophobic [27], electrostatic [28, 29], and van der Wall [30] forces play an essential role in characterising the binding affinity. Depending on the complex type, each energetic term can have a different weight in the binding affinity.

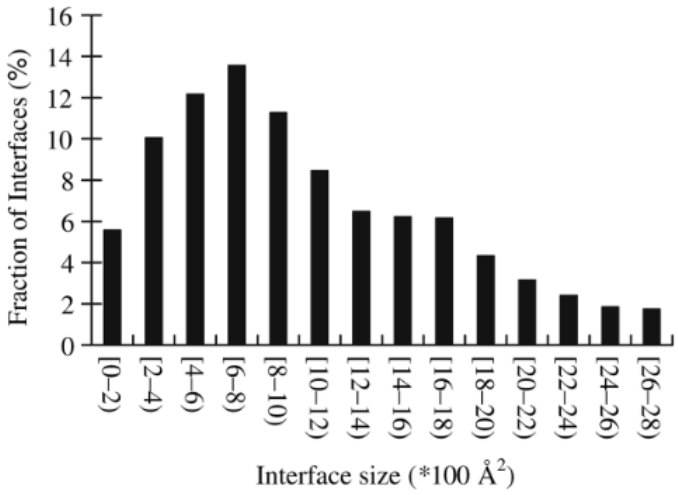


Figure 1.3. Interface size distribution. Interface size is calculated separately for each side of an interface. The distribution has a peak at 600–800 Å^2. About 25% of the interfaces have a (one-sided) size in the range of 800 (±200) Å^2 . Figure reproduced from Yan, C., et al., Characterization of protein-protein interfaces. Protein J, 2008. 27(1): p. 59-70.

1.3. Protein-DNA complexes

1.3.1. Importance of protein-DNA interactions

Protein-DNA interactions regulate many biological processes such as protein synthesis, signal transduction, DNA storage, and DNA replication and repair, among others. Learning how protein and DNA interact is fundamental to fully elucidate many central biological processes and disease mechanisms and can also support the discovery of novel therapeutic targets. There is a large variety of protein-DNA binding mechanisms. On the one side, DNA-binding proteins can be very specific to DNA sequence, such as the restriction endonucleases. But on the other side, there are DNA-binding proteins, such as histone proteins and DNA polymerases, which do not discriminate DNA sequences when binding.

To better understand these diversity of mechanisms, it would be important to obtain structural data at atomic resolution about all these protein-DNA interactions.

1.3.2. Structure characterization of protein-DNA complexes

Although 192,025 structures have been experimentally determined and deposited in the June 2022 release of Protein Data Bank (PDB)[31], only 10,480 of them correspond to protein-nucleic acid complexes. Among them, there are more than 6,732 high-resolution structures of protein-DNA complexes deposited in the Protein Data Bank (PDB) [31]. These, together with advances in new technologies for exploring binding modes, such as ChIP-seq, protein-binding microarrays (PBMs) [32] and SELEX (Systematic Evolution of Ligands by EXponential Enrichment) [33], are providing invaluable information about how the protein recognises its specific target sites. Based on available structural data, the existence of two readout modes has been proposed: base readout and shape readout [34, 35].

The base readout is the direct interaction between protein and DNA bases in the major and minor grooves. The binding site is discriminated through shape and electrostatic complementarity, including the formation of hydrogen bonds. Despite there is no exact correspondence between amino acids and DNA bases, some DNA-amino acids pairs are found with more probability, such as arginine with guanine, and asparagine and glutamine with adenine [36]. In addition, other recent studies suggest that π- π interactions between aromatic residues and DNA bases play an important role in the specific recognition of the protein-DNA interaction [37-41].

Shape reading acts both locally and globally on target DNA sequences in protein-DNA recognition [34, 42-47]. The proteins act on pre-existing DNA structural deformations in the target sequence. But they can also cause DNA deformations in the target sequence and in the flanking regions, especially the A- or T-rich stretch [42, 48, 49]. A recent metagenomic study demonstrated that the general shape changes in the DNA caused by a CpG methylation affect the bind sequence differently by altering roll and propeller twist, making it recognizable by the DNA binding protein [50].

More recently, it has been proposed that binding specificity is directly related to the number of hydrogen bonds, the major groove size, number of base contacts, and the propeller and rise angle in DNA [37].

However, the number of protein-DNA structures that are experimentally determined is clearly much smaller than the number of protein-DNA complexes that are expected to be formed in cells. This gap is partially explained by the difficulty of the experimental determination process, i.e. a very time-consuming process in the best scenarios or impossible in many cases due to limitations on the experimental techniques. For this reason, new computational approaches on modelling protein-DNA interactions are strongly needed.

1.4. Modeling protein complexes

As indicated in previous sections, structural data on protein-protein and protein-nucleic acid complexes is limited. As a consequence, computational strategies have emerged as a valuable complement to experimental techniques to improve our ability to predict 3D protein–protein complexes.

1.4.1. Template-based protein-protein docking

This strategy to generate docking models for a protein-protein target complex is based on superimposing the structures (or models) of the target subunits onto the corresponding subunits of an available template complex with sufficient homology with the target complex [51]. This makes it feasible to generate models for even multimeric complexes, by using suitable templates for each interface, which is particularly challenging for *ab initio* docking approaches (see section 1.4.2). With an increasing number of structurally solved complexes, template-based modelling is being extended to more cases. Thanks to this technique, it has been possible to build reliable models for about 5,311 human protein-protein complexes (Interactome3D, version 2020_05). It has been suggested that the PDB already contains structural templates to model most of the characterised protein interactions [52], but this is debatable as other studies claim otherwise. Actually, this approach is limited by the quality of the templates. If models are generated based only on remotely homologous

templates, the predictive success drastically decrease as the identification of the correct models is more uncertain [53]. In addition, oligomerisation between close homologues is not always conserved, which can lead to a model with an incorrect oligomeric state.

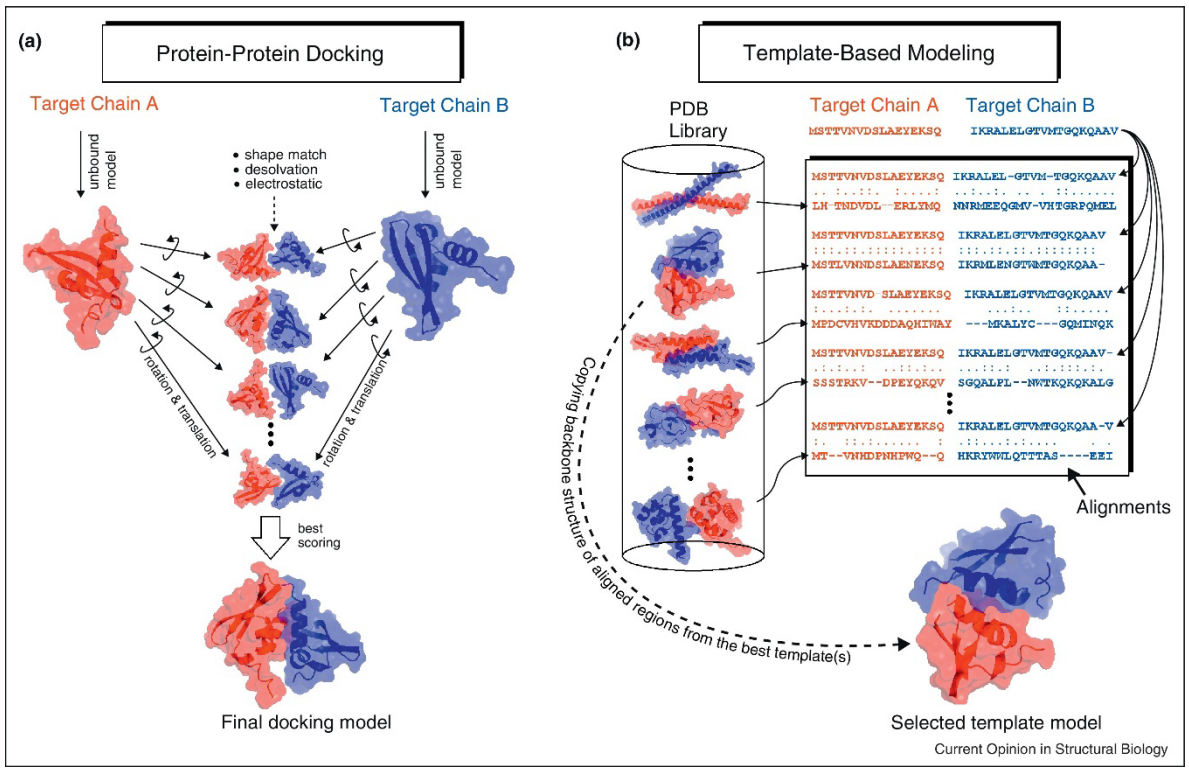


Figure 1.4. The two main methods existing in computational modeling of protein-protein interactions

Even when good templates are available, the superposition method can largely affect to the predictive success. Multiple sequence alignment (MSA) methods such as Clustal Omega [54], T-coffee [55], or MUSCLE [56] might not be able to provide a sufficiently accurate alignment. In this regards, structural alignment methods, such as DALI [57], STAMP [58], TM-Align[59], MM-Align[60], SuperPose[61], or VAST[62], can help to achieve better accuracy in the alignments.

The new generation of machine learning-based methods may make more efficient use of MSA data, revisiting the above mentioned statement that PDB already contains structural templates to model most of the characterised protein-protein interactions.

Finally, we should mention that the quality of the template-based docking models strongly depends on the quality of the structural models of the interacting subunits. If the experimental structure of the subunits is not available, there are many modeling methods

that can provide high quality models, such as RaptorX [63] or the most recent deep-learning approaches like Alphafold [64] o RoseTTAFold [65].

1.4.2. *Ab initio* protein docking-protein docking

In the majority of protein complexes, the quality of the templates is insufficient, in which case *ab initio* docking methods are the only practical alternative. The aim is to provide structural models for PPIs based on the structure (or model) of their unbound components.

For over the last three decades, a number of protein-protein *ab initio* docking approaches have been reported. From a methodological point of view, it is important to distinguish between sampling and scoring, which often define two independent phases. Sampling aims to search for different orientations, and scoring aims to evaluate the generated orientations to identify the correct binding mode.

Sampling of the translational space between the two proteins interacting proteins was accelerated by the pioneering use of Fast Fourier Transform (FFT) algorithms. FTDock [66], ZDOCK [67], and MolFit [68] are based on this strategy, with proteins represented by 3D grids, and scoring defined by mostly geometrical considerations. Other FFT-based methods using geometric grid representation of proteins are HEX [69] and FRODOCK [70], which use polar coordinates to speed up the calculations. There are also geometric methods that are not based on FFT, such as Patchdock [71], which are shape-matching methods that use a graph representation of the geometric features, such as convex, flat, and concave regions. On a completely different strategy, ICM-DISCO [72, 73], RosettaDock [74], HADDOCK [75], SwarmDock [76] or LightDock [77] aims to explore the protein-protein binding energy landscape by stochastic search methods, in search for the global energy minimum (though quite frequently trapped in local minima). Although most of them are rigid-body docking methods, they can be combined with molecular mechanic methods like molecular dynamics (MD) to address conformational flexibility, either before docking (generating conformation ensembles for the interacting monomers), during docking (which is computationally challenging), or after docking (to minimize the complex models). This facilitates their use even in proteins where large conformational changes can occur during the interaction, although the predictive success in these cases is low.

In all the above mentioned methods, scoring is essential in order to efficiently identify the correct models among all the ones that have been generated during the sampling phase. A large variety of scoring functions have been developed [78], from those based on statistical potentials to those using energetic descriptors, usually including van der Waals, desolvation or electrostatics terms. Scoring functions can also be built by combination of more simple parameters with traditional linear regression and optimization methods or with more recent machine learning and AI-based algorithms. Several scoring algorithms that can be independently applied to previously generated docking models are available, such as pyDock [79], ZRANK [80], or SIPPER [81]. In this regard, both FFT- and energy-based sampling approaches can be combined in the pyDock protocol [79, 82], with very efficient energy-based scoring. This protocol is publicly available on a web server (https://life.bsc.es/pid/pydock) [83]. Many methods, in addition to scoring, introduce flexible refinement of the docking models, such as FireDock [83], ICM-DISCO[73], HADDOCK [75] or RossetaDock [84].

Regarding the predictive capabilities, state-of-the-art docking methods provide acceptable models within the top 10 generated docking orientations in between 20% and 40% of the cases, according to the assessment on available protein-protein docking benchmarks [85-87]. In addition, most reported protein-protein docking methods have been assessed in blind conditions through the Critical Assessment of PRediction of Interactions (CAPRI) community-wide experiment, and the predictive rates of the top methods are in the same line. The accuracy of the predictions is dramatically changing with the recently developed deep-learning modeling approaches. For instance, the AF-Multimer method [88] provides impressive predictive rates on available protein-protein docking benchmarks, and its performance in blind conditions is currently being evaluated on the 5th Joint CASP-CAPRI experiment (on-going).

1.4.3. Docking of proteins and nucleic acids

Even though theoretical models of macromolecular structures are usually less accurate than direct experimental measurements, they can yield sufficient information to build a working hypothesis, complement experimental approaches in elucidating protein-

DNA interactions, and guide further experimental analyses to identify essential amino acids or nucleotide residues.

From a computational point of view, there are two main approaches to model the structure of a protein-DNA complex: template-based modelling and *ab initio* docking. Template-based modelling aims to model a complex based on the structure of a homologous complex. The popularity of template-based methods has increased in the past years, especially for modelling protein-protein complexes, thanks to the development and support of many structural databases of protein interactions that can provide the required templates, such as 3D Complex [89], Dockground [90], or Interactome3D [91]. However, the quality of template-based predictions clearly depends on the availability of suitable templates, not particularly high in the case of protein-DNA interactions, which makes template-based approaches of very limited applicability for protein-DNA modeling. On the other hand, *ab initio* docking methods aim to predict the three-dimensional structures of macromolecular complexes, starting from the atomic coordinates of their components. *Ab initio* docking methods do not depend on available structural data for homologous complexes, which makes them more useful in the actual protein-DNA context. Macromolecular docking protocols that accept protein and DNA coordinates as input include FTDock [66], GRAMM-X [92], HEX [93], PatchDock [71, 93] and NPDock [94], as well as HDock [95], ClusPro [96] and HADDOCK [97] servers. From this list of tools, only NPDock and HDock were originally developed for protein-nucleic acid docking; the rest were developed as protein-protein docking tools that can also accept nucleic acids coordinates, but they lack an intrinsic scoring function dedicated to assessing protein-DNA interactions. These protocols usually report high predictive rates in bound conditions, i.e. when the co-crystallized partners in a known complex structure are separated and then re-docked. However, despite bound docking is useful for testing and development purposes, it does not represent realistic conditions and thus it is of limited practical value for biology. Therefore, it is important to have available datasets to test protein-DNA docking tools in unbound conditions.

Interestingly, during the past editions of the CAPRI experiment [98], targets other than protein-protein complexes were proposed: protein-RNA complex [99] (T33, T34),

protein-peptide (T60-64) or protein-heparin (T57) among others. However, protein-DNA docking received limited attention from the CAPRI community and developers of computational methods. Compared to protein-protein docking, where the most recent release of the standard Protein-Protein Docking Benchmark 5.5 [87] has 257 entries, and to protein-RNA docking, where there are different reported benchmarks [100-103], for protein-DNA docking there is only one available benchmark, which contains 47 complexes [104]. Using this benchmark, protein-DNA docking protocols report moderate success rates in unbound conditions. For instance, on a subset of 23 cases from this benchmark, HDock success rate for top 10 models (i.e. at least one near-native structure within the top 10 models) is less than 10%, while success rate for top 100 is slightly over 30% [95]. NPDock reports a maximum success rate (i.e. at least one near-native conformation found in the entire prediction set) of 7/47 (15%) [94]. Protein-DNA docking with HADDOCK reported an excellent performance [105] when using restraints based on the real interface. This represents a very promising approach, but in a realistic scenario, lack of knowledge on the actual complex interface might limit its application. A more recent coarse-version of HADDOCK protein-DNA docking shows similar accuracy with ~6-fold speed increase over atomistic calculations [106]. The need of new computational tools to address unbound protein-DNA docking is clear.

2. OBJECTIVES

The objectives of this thesis can be grouped in these general aims:

i. Improve existing protein-protein docking functionalities

ii. Development of new protein–DNA docking protocols

ii. Exploration of new procedures that integrate *ab initio* and template-based protein-protein docking

iii. Implementation of the developed tools and protocols as web servers to share them with the community

iv. Evaluation of the new developments in blind conditions (CAPRI-CASP)

v. Application to systems of biological and biotechnological interest

3. METHODS

The protocols described here for the installation and execution of pyDock have been reported in this publication: Rosell, M., L.A. Rodríguez-Lumbreras, and J. Fernández-Recio, *Modeling of Protein Complexes and Molecular Assemblies with pyDock*, in *Protein Structure Prediction*, D. Kihara, Editor. 2020, Springer US: New York, NY. p. 175-198.

3.1. pyDock

The pyDock method is a set of protocols for protein-protein docking and scoring, previously developed [79, 82], validated [107] and successfully applied to many cases of biological and biotechnological interest [108]**.** The pyDock software needs the coordinates of the two interacting proteins, usually as PDB files. Hydrogen atoms are not needed in the PDB files, and if present, they will be removed and rebuilt again by pyDock. In addition, all HETATM coordinates will be removed in the docking calculations.

In this thesis, new functionalities have been implemented to be able to efficiently use AMBER coordinate and topology files. The newest version of the pyDock program optimized during this thesis can use AMBER coordinate files (with extensions such as inpcrd, .restrt, .rs7, .crd) and topology files (with extensions such as .prmtop, .parm7, .top) created by the PARM, LeAP, SANDER, or GIBBS programs from AMBER [109]. In this case, the HETATM coordinates from the cofactors and other compounds will be included in pyDock calculations.

3.1.1. pyDock installation

The pyDock 3.0 package is available at https://life.bsc.es/pid/pydock/get_pydock.html to get pyDock you need to apply for a license by filling in your data; for academic use, you will receive a link to the pyDock distribution file by e-mail; for commercial use, you will be contacted by the authors). Uncompress and untar the pyDock distribution file to extract the pyDock3 directory.

Next, we need to change permissions of the pyDock/data directory:

```
> cd pyDock3
> chmod go+rx data
> chmod u+x pyDock3
```

The pyDock3 directory can be moved to any location of your choice. For instance, let us say that it is moved to */usr/local/software/* directory; then, pyDock can be called by:

```
> /usr/local/software/pyDock3/pyDock3
```

Moreover, the PYDOCK variable can be defined in your .bashrc file, as follows:

```
export PYDOCK=/usr/local/software/pyDock3/
```

so that the executable of pyDock can be called in a more convenient way:

```
> $PYDOCK/pyDock3
```

The pyDock binary has been compiled for Linux 32-bit to increase the compatibility with older S.O.

3.1.2. pyDock external programs

3.1.2.1. SCWRL

In the case of input PDB files with incomplete side-chains, pyDock uses SCWRL 3.0 (http://dunbrack.fccc.edu/) to rebuild them. We should note that this version is outdated and cannot be directly downloaded from the above web, so you need to obtain the installation file scwrl3_lin.tar.gz from their authors. Then, unzipping this file will extract its contents to a new directory called scwrl3_lin. Inside this directory, run:

```
> ./setup
```

This will create a SCWRL 3.0 binary (scwrl3) in that directory.

3.1.2.2. FTDOCK 2.0

The program needs some external programs to generate a set of rigid-body docking poses. In this regard, pyDock is ready to process the output of FTDock 2.0, and we will show here how to install it. The first step is to install the FFTW libraries, that can be downloaded here: https://www.fftw.org/fftw-2.1.5.tar.gz

```
> sudo apt-get install mpi-default-dev

> wget https://www.fftw.org/fftw-2.1.5.tar.gz

> tar -xvf fftw-2.1.5.tar.gz
```

Uncompressing this file will extract its contents to a new directory called *fftw-2.1.5*. Within this new directory, compile the libraries by:

```
> ./configure --enable-type-prefix --enable-mpi --
prefix='/path/to/lib/fftw'

> make install
```

Now, the *ftdock-mpi-master.zip* file can be download. This is the custom FTDOCK version, with grid-optimization and ready to run in parallel [82]. It can be downloaded from the GitHub repository (https://github.com/brianjimenez/ftdock-mpi). This zipped file should be unpacked, and within the new ftdock-mpi-master directory, the Makefile file should be edited to set the FFTW_DIR variable to the full path (--prefix) used as input in the configure command above. Now, within the ftdock-mpi-master directory, type:

```
> ./make
```

This will create the program binaries, such as *ftdock*. More information can be found in the README.md file

3.1.2.3. ZDOCK

Another external tool that can be used for the generation of rigid-body docking poses is ZDOCK (https://zdock.umassmed.edu/). The pyDock pipeline is ready to process the output of ZDOCK 2.1.

3.1.3. Automatic use of external programs

For automatic use of FTDock, ZDOCK, and SCWRL programs within pyDock, after installing them locally, it is necessary to indicate the full path of the FTDock and ZDOCK directories, and that of the SCWRL binary, by modifying the corresponding lines in the $PYDOCK/pyDock3/etc/pydock.conf file, as follows:

```
(...)
ZDOCK=/<your-installation-directory>/zdock2.1_linux_64bit/
FTDOCK=/<your-installation-directory>/ftdock-mpi /
SCWRL=/<your-installation-directory>/scwrl3_lin/scwrl3
(...)
```

3.1.4. Running pyDock

pyDock has a highly modular architecture, with a series of modules performing the different functionalities of the program (Figure 3.1). The general syntax for running pyDock is:

```
> $PYDOCK/pyDock3 DOCKNAME modulename
```

Thus, the executable pyDock3 usually needs two arguments: (1) DOCKNAME, which is the name of the pyDock project and the base for all the files that will be created during the docking pipeline, and (2) modulename, which will call for the specific module. The details of the different pyDock modules are described in the running instructions below.

First, you need to create a text file called DOCKNAME.ini, in which DOCKNAME will be the name of the pyDock project. This file will contain all the needed information about the interacting proteins (PDB files, chain IDs...).

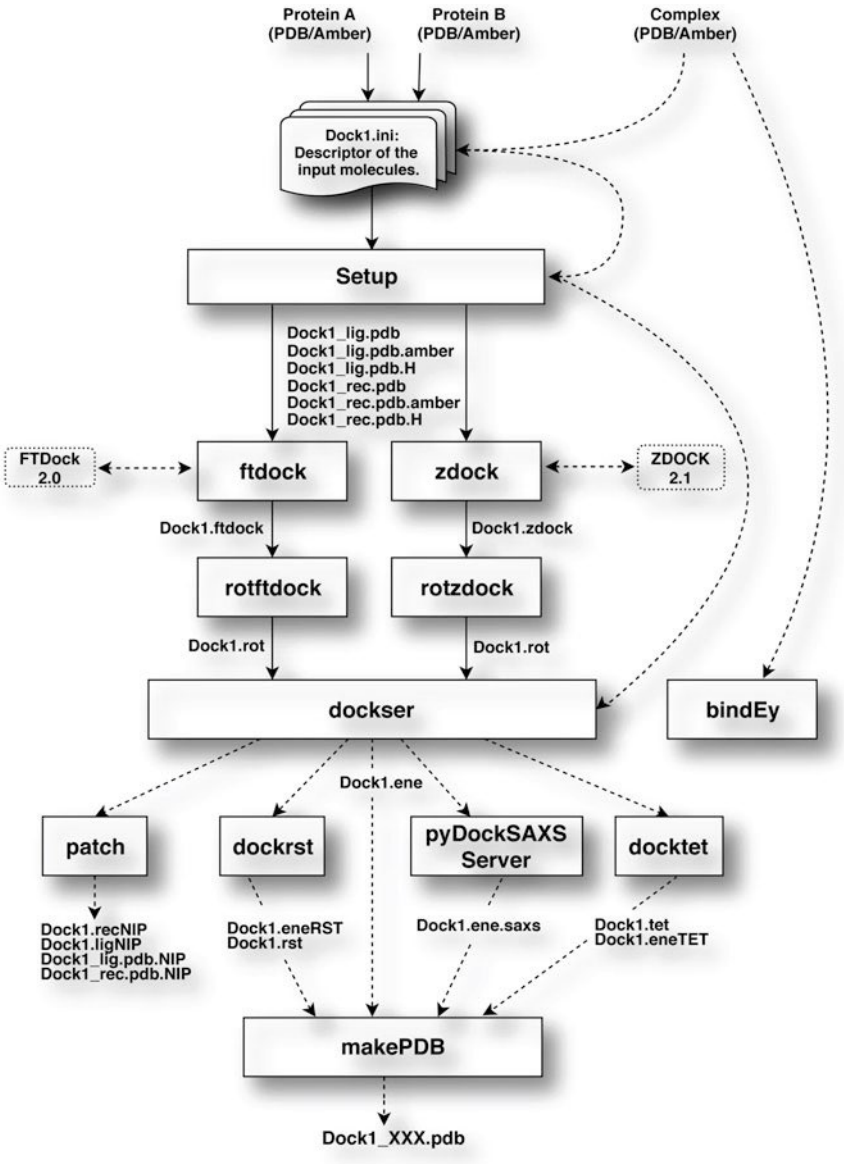


Figure 3.1 Scheme of the pyDock pipeline. The pyDock pipeline with the different pyDock modules is shown

We will illustrate this with an example, in which we will model the structure of the complex formed between the proteins TolB and Pal (PDB 2HQS) from E. coli, involved in maintaining the outer membrane stability of the bacteria [110]. For this example, we also have the structures of the unbound TolB and Pal proteins: PDB 1c5k (chain ID A) and 1oap (chain ID A), respectively. We will download the coordinate files of these proteins (1c5k.pdb

and 1oap.pdb) from www.pdb.org. In that case, we will create a text file called Dock1.ini (we will use "Dock1" as base name for the docking files) with the following information:

```
[receptor]
pdb = 1c5k.pdb
mol = A
newmol = A

[ligand]
pdb = 1oap.pdb
mol = A
newmol = B
```

The *mol* field is the original chain ID (one character) of the protein chain/s that will be used for docking, whereas the *newmol* will be the name of the chain ID of that protein in the docking output files. For convention, one of the proteins (usually the largest one) will be the receptor (static position), and the other the ligand (mobile position). pyDock can also take a protein structure in AMBER format. A real practical example can be found in Chapter 7.2.1 (but using the newer pyDock 4.0 version).

3.1.4.1. Set Up the Receptor and Ligand Coordinates

We need to generate the coordinate files correctly parsed for pyDock (similar to that in Chapter 7.2.1, but with a PDB file), from the receptor and ligand PDB files indicated in the *DOCKNAME.ini* parameter file. For that, run the pyDock setup, writing the following line in your console:

```
> $PYDOCK/pyDock3 Dock1 setup
```

This command will create the new PDB files for the receptor and the ligand *Dock1_rec.pdb* and *Dock1_lig.pdb*, respectively, which are suitable as input for pyDock.

Some PDBs may have incomplete side-chains, that is, there are missing atoms in the structures. These side-chains can be rebuilt with external programs. One of them is SCWRL,

which can be run automatically from pyDock if its location is indicated in the *pydock.conf* file (Chapter 3.1.3).

3.1.4.2. Rigid-Body Docking

pyDock can be applied to score rigid-body docking orientations generated by a variety of methods, but it is ready to automatically process the output from ZDOCK 2.1 or FTDock 2.0 docking programs. These programs can be run independently, but we will describe here how to call them automatically from pyDock, for which they should be previously installed (Chapter 3.1.2) and their location indicated in the *pydock.conf* file (Chapter 3.1.3).

In this example we will use ZDOCK 2.1, we can use the following commands to run ZDOCK (if previously installed) with pyDock:

```
> $PYDOCK/pyDock3 Dock1 zdock
```

This will produce the file *Dock1.zdock*, where all the resulting docking poses will be stored. Then the output data from ZDOCK (*Dock1.zdock* in our example) needs to be transformed to a file suitable for further pyDock scoring, containing the rotation and translation matrix for each docking solution:

```
> $PYDOCK/pyDock3 Dock1 rotzdock
```

This calculation is quite fast and will create a file (*Dock1.rot* in our example) containing the abovementioned transformation matrices for all docking poses.

3.1.4.3. pyDock3 Scoring

The next step is to use the pyDock energy function to score and rank all positions by running the *dockser* module with the following command:

```
> $PYDOCK/pyDock3 Dock1 dockser > dockser.log &
```

In the case of multicore computer architectures, this command will be executed in parallel. See Chapter 3.1.5.

The main output of the pyDock scoring step is a table (*Dock1.ene* in our example) with the detail of the different energy terms for each docking pose. See below the results obtained for this example when using ZDOCK 2.1:

Table 3.1. *Dock1.ene*

Conf(1)	Ele(2)	Desolv(3)	VDW(4)	Total(5)	RANK(6)
299	-31.439	1.680	38.489	-25.910	1
1682	-18.056	-7.028	19.563	-23.128	2
422	-27.451	0.377	52.760	-21.798	3
1969	-17.089	-11.647	79.455	-20.791	4

1. Conformation number of the docking pose (same as that in the .rot file, last column). 2. Electrostatic energy term. 3. Desolvation energy term. 4. Van der Waals energy term. 5. Total binding energy (Ele + Desolv +0.1∗VDW). 6. Rank of the docking pose according to its total binding energy.

The PDB file of a selection of resulting docking poses can be generated with the *makePDB* pyDock module, by indicating a range of models as ranked in the docking energy file. In our example, we will build the PDB files for the docking poses ranked 1–3 in the Dock1.ene file, as follows:

```
> $PYDOCK/pyDock3 Dock1 makePDB 1 3
```

This will create three files named *Dock1_299.pdb*, *Dock1_1682.pdb*, and *Dock1_422.pdb*, whose names will indicate the conformation numbers (Conf column in *Dock1.ene* file) of these top 3 ranked docking models.

3.1.5. Running pyDock in parallel

To run pyDock in parallel within a given pyDock project, you can download the file parallel-master.zip file with useful scripts from the GitHub repository (https://github.com/pyDock/parallel). Unpacking this zipped file within the $PYDOCK directory will create the parallel-master folder (alternatively, you can unzip the file from any location and copy the parallel-master folder to the $PYDOCK directory). In our example,

assuming that we are using a 4-core/8-thread computer, we can launch FTDOCK in parallel as follows (keeping one free core for the OS to prevent system instability):

```
> $PYDOCK/parallel-master/run_parallel_ftdock.sh Dock1 6
```

The *run_parallel_ftdock.sh* script has three arguments, the name of the pyDock project, the number of CPU threads (in case of multicore processor, it is advisable to define a number slightly smaller than the total number of cores), and an optional noelec parameter to deactivate the electrostatic evaluation during the model generation.

Let us suppose we have two different .rot files from FTDock and ZDOCK (e.g. *Dock1.rot_ftdock* and *Dock1. rot_zdock*). We can join both files into one and renumber the conformation numbers as follows:

```
> cat Dock1.rot_ftdock Dock1.rot_zdock > tmp.rot
```

```
> awk '{$13=NR;print $0}' tmp.rot | column -t > Dock1.rot
```

This new *Dock1.rot* file can be scored by pyDock, effectively including docking poses from FTDock and ZDOCK in a single set.

In case of using multicore computer architectures, the scoring module of pyDock can be run in parallel for faster execution times (especially with large-size proteins). For this, as described in Note 5, you will need to download the parallel-master.zip file with useful scripts from the GitHub repository (https://github.com/pyDock/parallel). Unpacking this zipped file within the $PYDOCK directory will create the parallel-master folder.

In our example, assuming that we are using a 4-core/8-thread computer, we can launch the pyDock dockser module in parallel as follows:

```
> $PYDOCK/parallel-master/run_dockser_parallel.sh Dock1 6
```

3.2. Structural Annotation Formats

There are many formats for storing three-dimensional protein information, but only the most common ones used during the thesis development will be shown here.

3.2.1. PDB

Among the variety of different formats for storing three-dimensional protein atomic coordinates, the most widely used is the one proposed by the Protein Data Bank (PDB). It was originally described in 1976 as a way for researchers to exchange data and to store them into a database [111]. Its fixed column width format has a maximum of 80 columns, which may limit its use for large macromolecular complexes. The format has been often updated since then. The most recent update was on 13 July 2011 (https://www.wwpdb.org/documentation/file-format-content/format33/v3.3.html).

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

Figure 3.2. Example of a PDB - seal myoglobin (PDB 1MBS)

The HEADER, TITLE and AUTHOR records provide information about the researchers who defined the structure. REMARK records may contain free-form annotations. REMARK 350 BIOMT records describe how to calculate the coordinates of the experimentally observed multimer from the specified monomer. The SQRES, give the sequences of the

peptide chains. And finally, the ATOM records describe the coordinates of the atoms that make up the protein. For example, the first ATOM line represents the alpha-N atom of the first residue of peptide chain A, which is a proline residue; the first three floating point numbers are its x, y and z coordinates and are in units of Angstroms. The following three columns are the occupancy, temperature factor, and atom name. The HETATM records describe the coordinates of the heteroatoms, i.e., the atoms that are not part of the protein or nucleic acid molecule. They can be cofactors or metal atoms, such as Heme, Fe, Cu, etc. (details of the format in the Figure 3.2).

3.2.2. PDBx/mmCIF

The current reference format in structural biology is PDBx/mmCIF [112]. It has practically no limitations, from small molecules to large macromolecular complexes. Actually, PDBx/mmCIF is derived from the CIF format, which was first used for small molecules. About 20 years ago, it was updated to mmCIF, aiming to be the successor of the PDB format. Data storage is done in a similar way to XML or JSON, with a dictionary or schema needed to know the internal structure and to be able to access the information.

The fact that PDB format is more user-friendly than PDBx/mmCIF, and that the PDB file can be easily retrieved from mmCIF (see Figure 3.3), makes the PDB format to be still widely used, due to its simplicity.

```
loop_ #More than one element, in this case, multiple atoms.
_atom_site.group_PDB
_atom_site.id
_atom_site.type_symbol
_atom_site.label_atom_id
_atom_site.label_alt_id
_atom_site.label_comp_id
_atom_site.label_asym_id
_atom_site.label_entity_id
_atom_site.label_seq_id
_atom_site.pdbx_PDB_ins_code
_atom_site.Cartn_x
_atom_site.Cartn_y
_atom_site.Cartn_z
_atom_site.occupancy
_atom_site.B_iso_or_equiv
_atom_site.pdbx_formal_charge
_atom_site.auth_seq_id
_atom_site.auth_comp_id
_atom_site.auth_asym_id
_atom_site.auth_atom_id
_atom_site.pdbx_PDB_model_num
ATOM   1    N  N   . VAL A 1 1   ? 6.204   16.869  4.854   1.00 49.05 ? 1   VAL A N   1
ATOM   2    C  CA  . VAL A 1 1   ? 6.913   17.759  4.607   1.00 43.14 ? 1   VAL A CA  1
ATOM   3    C  C   . VAL A 1 1   ? 8.504   17.378  4.797   1.00 24.80 ? 1   VAL A C   1
ATOM   4    O  O   . VAL A 1 1   ? 8.805   17.011  5.943   1.00 37.68 ? 1   VAL A O   1
ATOM   5    C  CB  . VAL A 1 1   ? 6.369   19.044  5.810   1.00 72.12 ? 1   VAL A CB  1
```

Figure 3.3. PDBx/mmCIF truncated example of X-ray crystallographic studies of seal myoglobin (PDB 1MBS)

In the example shown in Figure 3.3 the field labels appear first, and then immediately below the data (in PDB-like format). For example, *_atom_site.id* corresponds to the atom number, and the rest of the fields similarly describe the PDB columns.

3.2.3. Mol2

A Tripos Mol2 (.mol2) file is a complete and portable representation of a SYBYL molecule [113]. The most important characteristics of this file is that it explicitly contains atom type and bond information. In many cases, it is essential to use or convert PDBs to this file type to be able to perform the parameterization using *antechamber*. This was the case for the cofactors processed in Chapter 7 (see Figure 3.4).

```
# Name: benzene
# Creating user name: ChemicBook
# Creation time: Sat Mar 20 00:18:30 2021

@<TRIPOS>MOLECULE
benzene
 12 12 0 0 0
SMALL
NO_CHARGES
****
Any comment about the molecule goes here. No need for a pound sign here

@<TRIPOS>ATOM
      1 C          -0.7600    1.1691   -0.0005 C.ar    1  BENZENE       0.000
      2 C           0.6329    1.2447   -0.0012 C.ar    1  BENZENE       0.000
      3 C           1.3947    0.0765    0.0004 C.ar    1  BENZENE       0.000
      4 C           0.7641   -1.1677    0.0027 C.ar    1  BENZENE       0.000
      5 C          -0.6288   -1.2432    0.0001 C.ar    1  BENZENE       0.000
      6 C          -1.3907   -0.0751   -0.0015 C.ar    1  BENZENE       0.000
      7 H          -1.3536    2.0792    0.0005 H       1  BENZENE       0.000
      8 H           1.1243    2.2140   -0.0028 H       1  BENZENE       0.000
      9 H           2.4799    0.1355   -0.0000 H       1  BENZENE       0.000
     10 H           1.3576   -2.0778    0.0063 H       1  BENZENE       0.000
     11 H          -1.1202   -2.2126   -0.0005 H       1  BENZENE       0.000
     12 H          -2.4759   -0.1340   -0.0035 H       1  BENZENE       0.000
@<TRIPOS>BOND
     1     1     2   ar
     2     2     3   ar
     3     3     4   ar
     4     4     5   ar
     5     5     6   ar
     6     1     6   ar
     7     1     7    1
     8     2     8    1
     9     3     9    1
    10     4    10    1
    11     5    11    1
    12     6    12    1
```

Figure 3.4. Mol2 example Benzene**.**

3.3. Program languages (R, python, etc)

In this section, we will briefly mention some of the most relevant programming languages used in the thesis, with focus on some of their strong points: python 2.7 [114], python 3.x.x [115], perl [116], r [117] and the Linux console bash [118].

Python

Guido van Rossum developed Python in 1991. It is a high-level, interpreted, cross-platform and object-oriented programming language. It is also the most popular language (as to 2022). It contains a large number of tools for the bioinformatics community, such as Biopython [119], ProDy [120], and pyProCT [121]. Of particular note is its extensive use in Machine Learning (ML), with Scikit-learn [122], Theano [123] and TensorFlow [124]. It is also the main language used in the development of pyDock 4.0

Perl

Larry Wall initially developed Perl in 1987. Several years later, Andy Dougherty and Tom Christian, among others, joined the project (see https://perldoc.perl.org/perlhist). Perl is based on a block style like AWK and was widely adopted by genomic bioinformaticians for its text-processing prowess. But in my opinion, it is not an object-oriented language and In the Structural Biology field, it is not widely used.

R

Ross Ihaka and Robert Gentleman developed it in the 1990's. R was born as a tool strictly for statistical analysis. But over the years it has been adapted and there are also multiple packages to apply ML. To mention a few: data.table, dplyr, ggplot2, caret, e1071, xgboost, randomForest, etc...

Bash

It is the Swiss army knife for controlling Linux-based systems and although there are exceptions, it is the only way to give commands to the large computing clusters used during the development of the thesis. Bash is an interactive command interpreter and runs in a terminal, where commands are typed. You can also generate a list of commands in a file called a script and execute them.

3.4. Molecular Visualization Software

There are many and varied programs to visualise molecules, but the most commonly used by the bioinformatics community are UCSF Chimera [125], UCSF ChimeraX [126], Jmol/JSMol [127], PyMoL , VMD [128], ICM-Browser [129]. Also worth mentioning NGL [130] is embedded in the pyDockDNA server.

In the following sections, I will discuss more details of the three molecular visualisers mainly used during the thesis development.

3.4.1. ICM

ICM-Browser is a free version of the ICM program (www.molsoft.com) with many features for molecular visualization and structural analysis. It can display surfaces of ligand binding pockets, optimise hydrogens to a PDB, superimpose (structural align) protein structures,

measure distances and angles, generate and display surfaces, among others. All the functionalities can be accessed through the command line, but not all of them are easily found in the graphical user interface, e.g. through the command line one can open a collection of *SDF* files of molecules and perform different measurements such as RMSD calculation, but in the graphical user interface, this option is not visible.

3.4.2. UCSF Chimera

The program Chimera was used as an alternative to ICM, especially for the use of functions that were only available in its commercial version. The most interesting feature of Chimera is the possibility of using python scripting to do specific intensive tasks in addition to the command line (https://www.rbvi.ucsf.edu/trac/chimera/wiki/Scripts). You can do visual representations in a similar way as in ICM, as well as Molecular Dynamics (but only for teaching purposes).

3.4.3. Pymol

PyMOL is an open-source but proprietary program written in the Python programming language. It allows the creation of plug-ins that extend its functionality, among which is the Autodock plugin, which allows the setup of a docking grid (AutoDock Vina [131]) and view the docking results.

3.5. Benchmarks and evaluation sets

3.5.1. Protein-protein docking benchmark 4.0

The protein-protein docking Benchmark 4.0 [132] (BM4) was used as the target library for validating the protein docking funcionalities developed in this thesis. This protein benchmark provides 176 complexes solved by x-ray crystallography (119 dimers and 57 multimers) with at least 3.25 Å resolution, where the bound and unbound states are known. This benchmark is a non-redundant set of protein complexes that include, among others, enzyme-inhibitor, enzyme-substrate, and antigen-antibody complexes. The targets are classified as rigid body, medium and challenging in terms of the expected difficulty for *ab*

initio protein–protein docking. For further details, see https://zlab.umassmed.edu/benchmark/ web site.

3.5.2. DOCKGROUND

In this study, we used the structural templates included in the DOCKGROUND resource v1.0 [133]. The structures of protein-protein complexes included in this library were solved by x-ray crystallography with a resolution better than 3.5 Å. Only complexes with a mean accessible surface area buried by each chain greater than 250 $Å^2$ and containing at least 10 interface residues are included. Structural diversity was ensured with the MM-align program by using a TM-score cut-off of 0.9, which resulted in a dataset of 7,107 diverse protein–protein interfaces. See http://dockground.compbio.ku.edu for a full database description.

3.5.3. Protein-DNA

In order to test the new pyDockDNA docking protocol developed in this thesis, we used a previously reported protein-DNA docking benchmark (version 1.2) [104]. The benchmark contained bound and unbound x-ray crystallography and NMR structures for 47 protein-DNA complexes in which DNA is in B-DNA conformation. These were classified as "easy", "intermediate" or "difficult" cases, based on the interface RMSD values between the bound and unbound components of the complex (Table 3.2).

Table 3.2. Protein-DNA docking benchmark (version 1.2)

HTH	Zinc-coord	Other α-helix	β-sheet	B-harpin	Enzyme
2C5R	1BY4	1KSY	1HJC	1EA4	1PT3
1FOK	1R40	2FIO	1QNE	1AZP	1EMH
3CRO	1ZME	1JJ4		1CMA	1DIZ
1H9T		1QRV		1BDT	1VRR
1TRO		1B3T			1KC6
1RPE					1Z63
1MNN					1VAS
1F4K					4KTQ
1K79					1G9Z
1W0T					1A73/1A74
1Z9C					3BAM
1DDN					1RVA
2IRF					1DFM
1JT0					7MHT
1ZS4					2FL3
1O3T					1EYU
					2OAA

Classification of cases as previously described [134]. Underlined cases have only 1 DNA molecule. Green are the easy cases with interface $RMSD_{b-u}$ (between bound and unbound of the complex) ranging from 0.0 Å to 2.0 Å. Blue are the medium cases with interface $RMSD_{b-u}$ between 2.0 Å and 5.0 Å. Red ones are the hard cases with interface $RMSD_{b-u}$ above 5.0 Å

An additional set of case studies was compiled following the criteria selection used in the above-described protein-DNA docking benchmark. This test set is composed of ten protein-DNA complexes, where both bound and unbound structures are available for each reference complex, and the sequences are different from those in the first protein-DNA docking benchmark (Table 3.3). Protein-DNA complex and unbound structures were compiled from the Protein-DNA Interface Database (PDIdb) [135] and the Protein Data Bank (PDB) [31]. Only complexes that meet the following conditions were considered: i) DNA sequence length larger than eight base pairs, and ii) proteins without mutations in the core of the complex interface. To find the protein unbound structures of the selected protein-DNA complexes, all the PDB entries containing only protein structures were retrieved, including structures solved by NMR. Crystallographic structures with a resolution worse than 3.0 Å were not considered. To avoid redundancy, entries with sequence similarity ≥ 90% were discarded. PDBeFOLD [136] was used to find correspondences between bound and unbound protein structures. This tool performs structural alignments between two

(pairwise alignment) or more (multi-alignment) molecules using their 3-dimensional structures. The alignment is based on the Secondary Structure Matching algorithm [136]. Alignments with a Q-score higher than 8.0, high P-score and sequence similarity around 90-100% were accepted as the corresponding unbound. Then, the bound and unbound structures for each case, were post-processed according to the protocol followed in a previously developed protein-DNA docking benchmark, for instance by checking consistency between unbound and bound coordinates in chain IDs, residue numbers and atom names [104]. The unbound DNA models were generated by using the software 3DNA [137, 138], in canonical B-DNA conformation (fiber model 4).

This additional test set (is freely available at the "Help" section of the server (https://model3dbio.csic.es/pydockdna/info/faq_and_help#extended_bechmark).

Table 3.3. List of the case external test set.

PDB complex	Protein	PDB unbound protein	RMSD unbound-bound protein	DNA	RMSD unbound-bound DNA
5JLT	phage T4 MotA DNA-binding domain	1KAF	0.83[a]	22bp dsDNA	1.89
2X6V	TBX5	2X6V	0.55	11bp DNA	2.03
3POV	SOX	3FHD	1.46	19bp DNA	2.26
4UUV	ETV4 DNA-binding ETS domain	5ILU	1.24	10bp DNA	2.81
2NTC	sv40 large T antigen	2FUF	1.13[a]	21-nt PEN element of the SV40 DNA origin	2.96
2ITL	sv40 large T antigen	4NBP	5.37[a]	24-nt PEN element of the SV40 DNA origin	3.84
3MFK	Protein C-Ets1	1GVJ	5.61[a]	stromelysin-1 promoter DNA	4.34
2PI0	IRF-3	3QU6	0.76[a]	PRDIII-I region of human interferon-B promoter strand 1	4.46
1O3R	catabolite gene activator protein	4R8H	0.65	11bp DNA	4.77
3MLO	Ebf1	3LYR	0.71[a]	22bp DNA	5.11

[a] In cases with more than one protein-DNA interface in the x-ray structure, the average value is provided.

3.5.4. CAPRI

The Critical Assessment of Predicted Interactions (CAPRI) community-wide experiment started around 2001, when the community of developers of protein-protein docking methods aimed to evaluate the success rate of such algorithms [98, 139]. CAPRI has been crucial in pushing its community members to improve and add new functions to their docking protocols [140, 141]. Most of the targets proposed by the CAPRI experiment focused on protein-protein docking procedures. Still, recently there have been new challenges, such as protein-peptide and protein-oligosaccharide docking (see Chapter 6.5.1) [142].

The experiment consists in an open competition in which the predictive success rates of the different docking methods are compared in double-blind conditions. The organizers choose the targets, consisting of experimentally determined complex structures that are not yet publicly available. Thus, the targets are blind for the participants, and the participant names are blind for the organizers when evaluating their predictions.

For each target, there are usually two modes of participation in the experiment: predictors and scorers. In predictors, the groups are asked to submit ten models from the sequences of the target structures. Generally, the 3D structures or reasonable templates of the interacting molecules are available, which can be used as a starting point for protein-protein docking. In the scorer participation, the groups are invited to evaluate a common

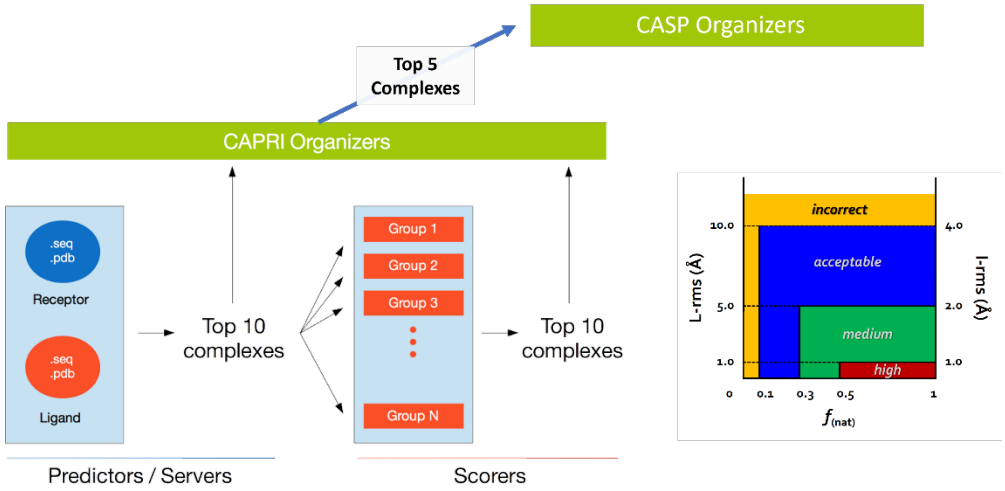


Figure 3.5. CAPRI-CASP evaluation process.

set of docking models submitted by the predictors groups. Only the top 5 or the top 10 models by each participant are considered for the assessment.

At the end of each round, which may consist of several targets, the ten models submitted by each participant (either as predictors or as scorers or both) are evaluated based on the ligand RMSD, the fraction of native contacts and the interface RMSD with respect to the actual complex structure (see Figure 3.5).

3.5.5. CASP-CAPRI

The CASP and CAPRI communities established close ties during the CASP 2014 edition, where the section of "multimeric assemblies" was organized jointly with the CAPRI community [143]. Since then, a total of five joint CASP-CAPRI rounds were held [107, 144, 145], including this year (2022) edition.

These joint CASP-CAPRI rounds have encouraged many developers to integrate their *ab-initio* docking methods with structure prediction methods. Many of these protocols are periodically collected in books, such as the seventh edition of Protein Structure Prediction 2020 [146]. Participation and evaluation in these rounds are done independently by CASP and CAPRI organizers, the latter in a similar way as explained in the previous section of this chapter, see Figure 3.5.

3.5.6. Physiological/non-physiological homodimers

The groups of R.L. Dunbrack and E.D. Levy developed a benchmark set of physiological/non-physiological dimers (version 3) in the context of the Activity II of the 3DBioInfo ELIXIR community, in which I have participated during this thesis.

The Benchmark contains a number of protein homo-dimeric x-ray structures that are classified as either "physiological" or "non-physiological". Physiological homodimers are defined as those that are likely to occur in the cell. Non-physiological are homodimeric interactions that are seen in the crystal structure but are unlikely to occur in the cell, because the known physiological state is monomeric or because the true homodimer involves a different interface.

Physiological dimers were identified using two complementary methods. The first one based on 3D structure interface conservation in the PDB defined by the ProtCID database [147] . And the second through homologues, based on the QSalign resource [148].

Non-physiological dimers were defined in two batches. The first batch was identified as homodimeric structures whose interface was not conserved between homologues in QsAlign. The second batch was identified as cases of homodimers whose interfaces were not similar to any interface across any crystalline form, or that of homologous proteins as defined in the ProtCID database (see Supplementary Methods for details). Each batch was further pruned to include sets with the same interface area distribution as the physiological dimers, to focus on those cases that are more challenging for predictions. This filter was applied to avoid distinguishing non-physiological dimers from their physiological counterparts based solely on their smaller interface area. Thus, the total number of non-physiological dimers was 841 (142 dimers from batch 1 and 700 dimers from batch 2), in good balance with the number of physiological dimers (836). More details about this benchmark will be published in an upcoming publication.

4. pyDock 4.0: ADDRESSING CURRENT PROTEIN-PROTEIN DOCKING CHALLENGES

4.1. Need of a new pyDock version 4.0

From a technical point of view, this new version was essential for the future maintenance and implementation of new code. The necessary modifications were made so that the code can be executed by a Python 3.x.x interpreter [115]. In this way, the current and future code will benefit from the updates of Python libraries. For example, the 3.11 version of Python is expected to gain between 10% and 60% speed, which is very important for computationally intensive programs such as pyDock.

From a more conceptual point of view, this new version introduced the ability to evaluate interactions between proteins and other macromolecules, like DNA, RNA, or carbohydrates, as well as the inclusion of cofactors in protein-protein docking (see Chapter 7.4).

4.2. Software reimplementation of pyDock in Python 3.

Before starting the reimplementation, it was necessary to adapt the external library, *nwalign* ver 0.3.0, used by pyDock. The previously adapted repository of this tool did not work correctly, so we decided to adapt the existing version. After some research on the Cython code, we noticed that the only problem was that string variables had to be of binary type in Python 3.x.x [115] , so we replaced the variables in the source code (*PyString* ----> *PyBytes*) (Table 4.1). Now we use an if statement that selects the variable type depending on whether we use Cython in Python 2.7 or Python 3.x.x., so the new code is compatible with both.

Table 4.1. Modified variables in the Cython code of nwaling.

Cython Python 2	Cython Python 3.x.x
PyString_FromStringAndSize	PyBytes_FromStringAndSize
PyString_AS_STRING	PyBytes_AsString
_PyString_Resize	_PyBytes_Resize
PyString_FromStringAndSize	PyBytes_FromStringAndSize

The changes implemented in pyDock are mainly related to changes in the built-in functions of Python 3.x.x compared to Python 2.7. For example, exception handling, for

which the way to express exceptions has changed. The *zip* function now returns an iterator, but in the pyDock code, a list is expected. Dictionaries and their associated functions, such as *sorted* and *iteritems*, have disappeared, and some changes are needed to generate the same behaviour. Other functions disappeared, like *reduce*, which had to be created *ad-hoc*, etc. (Table 4.2).

Table 4.2. Coding differences between Python 2 and Python 3

Comments	**Python 2.7**	**Python 3**
ConfigParser	ConfigParser.readfp	ConfigParser.read.file
Handling exceptions has slightly changed in Python 3.x.x Print statement has been replaced by the print() function, meaning that we have to wrap the object we want to print in parentheses.	try: print "ok" except Exception, e:	try: print ("ok") except Exception as e:
xrange returns an iterator and only keeps one number in memory at a time. range keeps the entire list of numbers in memory	range()	xrange()
We need a list, zip() returns a lazy iterator.	zip(*[iter(fields)] * dimensions)	list(zip(*[iter(fields)] * dimensions))
	file(data_file).readlines()	open(data_file).readlines()
Change of fortran compiler	f2py -c -m access access.f	f2py3 -c -m access access.
The way of importing modules has also changed.	from libpydock.pdb import PDBIO	from pyDock4.libpydock.pdb import PDBIO
Dictionary item iterator exit to Python 2.7 but not to Python 3.x.x (Many changes done through the code)	for atom_type, atom_list in atom_dict.iteritems():	for atom_type, atom_list in iter(atom_dict.items()):

reduce is a built-in function of Python 2.7, but not in Python 3.x.x.	reduce()	def reduce(func, seq):
		first = seq[0]
		for i in seq[1:]:
		first = func(first, i)
		return first

The code worked with all the above changes but did not reproduce the same results as in pyDock version 3. The problem was in how the atom bonds were represented and sorted. In Python 3.x.x, the output sorting of the *dictionary.items()* method (extracting items from a dictionary) is the same as when it was created. In the case of Python 2.7, the sorting is related to each element's place in the memory hash table. To emulate the behaviour of pyDock 3, we mapped the equivalence between the two versions and forced the sorting of the resulting list. An example of the changes made to the AminoAcid class can be seen in During reduplicity testing of the code, a previous bug has been identified and fixed in addition to the changes mentioned above. This bug, which affected the pyDock configuration, meant that when selecting more than one PDB, the hydrogens at the amino-terminal end of the n+1 polypeptide chains that form the ligand or receptor (selected in the ini file, see Chapter 3.1.4) were not added correctly. By correcting this error, the reduplicity of pyDock 4.0 versus pyDock 3 is lost, giving different pyDock energies, which are now correct in pyDock 4.0.

```
      def C4_three_H(self, carbon, bondStructure):
          bonded = self.get_bonded_to(carbon.atom_type, bondStructure)
          atom2 = self.get_atom_by_type(self.get_next_in_bonded_list(bonded, carbon.atom_type))
          bonded = self.get_bonded_to(atom2.atom_type, bondStructure)
+         if bonded == ['CD2', 'CD1', 'CB']: bonded=['CB', 'CD1', 'CD2']
+         if bonded == ['N', 'CB', 'C']: bonded=['CB', 'C', 'N']
          atom3 = self.get_atom_by_type(self.get_next_in_bonded_list(bonded, carbon.atom_type))
+         #print("C4_three_H", carbon,atom2,atom3,bonded)
..
          log.debug('C4_three_H:')
          log.debug(carbon)
          log.debug(atom2)
          log.debug(atom3)
```

Figure 4.1. Code extract from the AminoAcid class, pyDock version 4.0.

4.3. Software Parallelization.

4.3.1. External Programs

As explained in Chapter 3.1.2, FTDOCK 2.0 needs to be compiled to be used. The compilation differs for single or multi-threading use. Because of this, many users compiled the code for single threading and tried to run the code (using *mpirun* [149] or *srun* [150]) in parallel. But no warning is shown, and it has been the case that the same docking has been executed on several cores without the correct division of work. In prevention, an automatic pipeline has been added to pyDock 4.0, which detects whether FTDOCK and ZDOCK binaries are

compiled for parallel use. If they are not, it acts accordingly, warning the user and executing the software using a single thread by default. Also, the pyDock configuration file has been updated so the user can activate the parallel execution of the Python code and select the number of threads used by pyDock and the external programs.

4.3.2. pyDock 4.0

In earlier versions of pyDock, to make use of the computational resources of the MareNostrum (MN) at the Barcelona Supercomputing Center (BSC), the greasy software (https://github.com/BSC-Support-Team/GREASY) was used together with a helper script to perform an embarrassingly parallel calculation of the pyDock scoring function, so that the code was actually run on a single thread but executed several times. It is not a problem for a large supercomputer, but it can be a problem on modest computers. We have therefore chosen to parallelise the code internally. We use the standard Python 3 multiprocessing, which allows code to be parallelised at the node level. This makes it easy to use on computing clusters, workstations, and home PCs, and no longer relies on greasy software.

Not all supercomputers have this software installed by default. Furthermore, the current parallelisation scheme can be exported to the multi-node level by changing a few lines of code. But in this code refactoring, we tried not to call external classes to avoid increasing the size of pyDock software.

So far only the *dockser* and *pyCluster* modules have been parallelised, but there are plans to extend the parallelisation of the code to all the modules where it could be be useful.

4.4. New modules for the use of AMBER files and clustering

4.4.1. Use of AMBER files in pyDock 4.0

The *amber* module makes it possible to perform pyDock molecule *setup* (see Chapter 3.1.4.1) using the AMBER coordinate files (with extensions such as *.inpcrd,. restrt, .rs7, .crd*) and topology files (with extensions such as *.prmtop,.parm7,.top*) created by the PARM, LeAP, SANDER, or GIBBS programs from AMBER [109].

In the initial versions, this *amber* module could only be used for protein-protein docking with modified amino acids. This version only used the atom charges of the AMBER topology file. The van der Waals (VDW) parameters and the atomic solvation parameters (ASPs) were internally defined by pyDock data files. The original pyDock version used *parm94* [151], which limited the type of atoms that can be mapped. Also, the atomic solvation parameters are unique to pyDock (they are not included in the general molecular mechanics force fields) [152].

To make this module work on a wider variety of molecules, the pyDock 4.0 function that reads the topology files was modified to extract both the charge and VDW values and write them in the *.amber* file setup parameters (this file will be used to calculate the energy

scoring function by *dockser* module). To use the pyDock solvation parameters, we created

```
! equivalences between new amber atom types and old (parm94) atom types
! from parm99.dat
LP      Cl     #lone pair
HZ      HC     #H bond sp C (Howard et al.JCC,16,243,1995)

! from frcmod.ionsff99_tip3p
Li+     Li     #lithium   pol: J. Phys. Chem. 11,1541,(1978)
Na+     Na     #sodium    pol: J. Phys. Chem. 11,1541,(1978)
K+      K      #potassium
Rb+     Rb     #rubidium
Cs+     Cs     #cesium
F-      F      #fluorine
Cl-     Cl     #chlorine  (Applequist)
Br-     Br     #bromine   (Applequist)
I-      I      #iodine    (Applequist)

! from parm10.dat
CI      CT     #parmbsc0
CP      CK     #sp2 C 5 memb.ring in purines (G)
CS      CM     #sp2 C pyrimidines in pos. 5 & 6 (U)
CX      CT     #protein C-alpha (new to ff10)    # copied from parm10   #GLYCAM_06j.dat
C5      CK     #sp2 C 5 memb.ring in purines (A)
C4      CM     #sp2 C pyrimidines in pos. 5 & 6 (C)
OP      O2     #2- phosphate oxygen

! from frcmod.parmbsc0_ezOL1
C3      CT     #sp3 aliphatic C

! from frcmod.parmbsc1
CE      CT
C1      CK     #sp2 C purines in pos. 8 Ade (CK)
C2      CM     #sp2 C pyrimidines in pos. 5 & 6 Thy Ura (CM)

! from parm15ipq_10.3.dat
CO      C      #sp2 C carboxylate group
2C      CT     #sp3 aliphatic C with two (duo) heavy atoms   # copied from ff12SB   #GLYCAM_06j.dat
3C      CT     #sp3 aliphatic C with three (tres) heavy atoms  # copied from ff12SB   #GLYCAM_06j.dat
C8      CT     #sp3 aliphatic C basic AA side chain
OD      O      #ASN/GLN amide O; backbone
ND      N      #ASN/GLN amide O; NHE
OA      OH     #SER/THR/TYR hydroxyl O
NL      N3     #LYS ammonium N; backbone N-terminus
TN      NT     #PRO backbone N
TJ      CT     #PRO backbone CA
TG      CT     #GLY backbone CA
TP      CT     #ARG/HIP/LYS backbone CA
TM      CT     #ASP/CYM/GLU backbone CA
TH      H      #ARG nitrogen-attached H
TA      CT     #HID/HIE/PHE/TRP/TYR CB

```

Figure 4.2. Partial snapshot of the equivalence dictionary between *parm94* and the newest forcefield parameters find in the lasted AMBER version. (*https://github.com/pyDock/parallel/blob/master/amber_old_to_new.map*)

a dictionary of equivalences between the newest amber atom types and the old (*parm94*) atom types (see Figure 4.2)

Finally, the pyDock function that creates the output PDB files was modified. The resulting PDB can be used directly with FTDOCK, without requiring future modifications. A use case for this module upgrade can be found in Chapter 7.3.1.

4.4.2. pyCluster: Clustering in pyDock 4.0

The RMSD matrix required to apply the BSAS algorithm [153] is computed with an *ad-hoc* ICM script in the in CAPRI and CASP-CAPRI rounds. But the use of parallelisation in ICM is

not trivial. The initial option was to implement the clustering protocol that we used to test pyDockDNA, pyProCT, but this program (as standalone) is a Python 2.7 module, which is incompatible with the new pyDock version. Therefore, we decided to implement the BSAS algorithm as a new module in pyDock 4.0.

The new *pyCluster* module is able to generate the RMSD matrix very quickly thanks to the use of standard Python 3 multiprocessing, just like the *dockser* module. And it can be used both for the direct output of pyDock 4.0 and for collections of independent models, as it is the case for the CAPRI scorer experiment. See Figure 4.3A and Figure 4.3B for examples of INI configuration files that can be used.

A

```
[clustering]
modelslist  = cluster_1AVX_clustering.list
scoring_function = PyDock

[receptor]
mol_cluster = A

[ligand]
mol_cluster = B
```

B

```
[receptor]
pdb         = 1AVX_r_u.pdb
mol         = A
newmol      = A

[ligand]
pdb         = 1AVX_l_u.pdb
mol         = B
newmol      = B

[reference]
pdb         = 1AVX_b.pdb
recmol      = A
ligmol      = B
newrecmol   = A
newligmol   = B

[clustering]
RMSD_cutoff = 20
Nmodels     = 100
```

Figure 4.3. PyClust inifile. (A) The ini file have as inputs a list of PDBs. The ligand and the receptor chains must be specified. If the RMSD_cutoff and Nmodels are not specified, they are set to 4Å and 100 models, respectively. (B) Corresponds to an ini file, using the direct output of pyDock 4.0. Here, you can select the RMSD_cutoff and Nmodels

5. pyDockDNA: A NEW WEB SERVER FOR ENERGY-BASED PROTEIN-DNA DOCKING AND SCORING

The webserver described here have been reported in this publication: Rodríguez-Lumbreras, L.A., et al., *pyDockDNA: A new web server for energy-based protein-DNA docking and scoring.* Frontiers in Molecular Biosciences, 2022. 9.

5.1. Development of pyDockDNA: a new protein-DNA docking procedure.

5.1.1. Sampling

In this first step, the input files with the coordinates in PDB format for the structures (or models) of a protein and a DNA molecule (which can be B-DNA or any other conformation) are checked for potential format errors. Missing side-chains in the protein are rebuilt with SCWRL 3.0 [154], and the electrostatics Amber94 force field [151] is loaded, assigning the charges to the atoms. Then, rigid-body docking poses between the protein and the DNA, represented as 3D grids, are generated with a faster and parallelized version of the original FTDock (v2.0) software [66] in which the number of cells in the grid is optimized for maximum computing efficiency [82]. The molecule (protein or DNA) with the longest maximal distance between any pair of atoms is considered the receptor, that is, the fixed molecule, and the other one is the ligand or mobile molecule. By default, the program uses 0.7 Å grid cell size, 1.3 Å surface thickness, 12º rotation sampling, and keeps the best 3 poses for each rotation. For each target, a total of 10,000 docking poses are generated.

5.1.2. Scoring

Then, the protein-DNA docking poses are ranked using a scoring function composed of electrostatics, desolvation and van der Waals energy. This new pyDockDNA scoring function is adapted from the previously pyDock scoring function for protein-protein docking [82, 155], which now includes atom types for nucleotides from Amber94 force field [151] in order to calculate for the modelled protein-DNA complexes. The nucleotide AMBER atom types have been mapped to the previously defined atom types in pyDock within a new parameter set (nuc.dat).

5.1.3. Clustering of protein-DNA docking models in benchmarking

When testing this software (see Chapter 5.3) we have run several docking executions in parallel, using different initial random rotations for the input structures, and the best-scoring 100 resulting models for each individual run were merged into a single pool. To avoid redundancy in the final set, all docking orientations were clustered by pyProCT

analysis software [121], which implements the *GROMOS* clustering algorithm [156]. The distance matrix is built with pyRMSD with the option *"QCP OMP CALCULATOR"* to compute the ligand root-mean-square deviation (L-RMSD) values for all pairs of docking orientations after their receptors were superimposed (https://github.com/victor-gil-sepulveda/pyRMSD/). A cut-off value of 4.0 Å was used for L-RMSD to define the clusters. For each defined cluster of models, the orientation with the lowest docking score is selected as the cluster representative.

5.2. Implementation of pyDockDNA as a web server

The pyDockDNA program has been built as a module of the new pyDock 4.0 version, and it includes the same third-party programs, modules and tools of the previous pyDock versions, as well as new functionalities to handle nucleic acid structures in a proper way (see Chapter 4). The program has been implemented as a web server, in a virtual machine hosted in one of the data processing centres (CPD) of the Spanish National Research Council (CSIC) and is accessible through the following link: https://model3dbio.csic.es/pydockdna. The operating system installed is Debian10. For the correct management of the available resources, *Slurm* [150] was installed, which is a task management system for clusters. Thanks to this system, we can control the jobs received by the server and keep them in queue when resources are limited. For security reasons, the reverse proxy *Nginx* was installed in conjunction with the *uWSGI* server, as it has a lower rate of serious vulnerabilities compared to Apache (https://www.cvedetails.com/).

As far as the web application is concerned, it is defined as a backend and a frontend. The backend is essentially a daemon, i.e. a resident program running in the background. This daemon is an adaptation of the version used by pyDockWEB [82] but updated to run on the newest version of python so that it can host pyDockDNA [157] web application. Essentially, it is in charge of submitting jobs to the *Slurm* queue, monitoring their progress and handling execution errors. The job executes several pyDock 4.0 modules in a concerted way, according to the options selected by the user in the frontend. This information is reported to a database that acts as a bridge between the backend and the frontend.

The frontend is developed using web2py, a framework for designing websites using the python language. On the main page, the user can choose the name of the job and an email address to be notified at the end of the job. One can use the RCSB code to select the structures (ligand and receptor) or upload them from his computer. In the following steps the user can select the chains to be docked, the energetic scoring function, and even include external information (from available experimental data or using predictive methods such as the DBSI server [158], for instance) as residue-nucleotide distance restraints to rescore docking models as previously described for pyDockRST [159]. The output will be a set of docking models represented in different formats: i) the 3D structure of the best-scoring 10 docking models in terms of scoring can be visualized in the output screen, ii) the PDB files for the best-scoring 100 models can be directly downloaded, and iii) the

Figure 5.1. pyDockDNA server results. The top 10 predictions based on the user-selected scoring function are displayed as a table on the top left and as a 3D representation on the upper right. In the lower right area, a graph of the energy distribution of the 10,000 models are represented.

rotation/translation vectors are provided to generate up to a total of 10,000 docking poses. A summary of the docking results can be visualized as a plot with the distribution of the different energy values obtained for all docking poses (Figure 5.1).

5.3. Performance of pyDockDNA evaluated on the protein-DNA docking benchmark.

The pyDockDNA web server has been tested on the 47 cases of a previously reported protein-DNA docking benchmark (see Methods). It is known that using different randomly rotated input structures can slightly affect docking predictions of FFT-based docking protocols as in FTDOCK, because this can modify the mapping of the atom positions on the 3D grids [70, 160]. To check for convergence, we applied pyDockDNA to 10 different random rotations of the initial input structures for each benchmark case and computed the predictive success rates for the results obtained from each randomly rotated input structures. The results indicate even more differences in the predictive values than previously reported for protein-protein docking (Appendix 1: Table 10.1.1). For instance, the success rates for the top 10 models ranged from 12.8% to 21.3%. Therefore, for a more robust evaluation, we merged the results of all 10 docking executions and clustered the

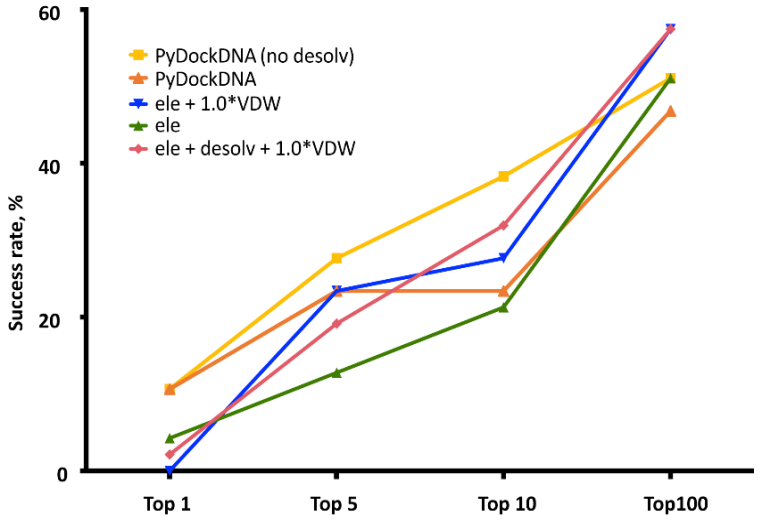


Figure 5.2. Predictive performance for the top N=1, 5, 10, 100 models of pyDockDNA and different combinations of scoring terms on the protein-DNA docking benchmark.

obtained docking models to remove similar orientations. Figure 5.2 shows the predictive success rates of the cluster representatives resulting from merging these 10 docking runs (see more details in Appendix 1: Table 10.1.2). The predictive success for the default pyDock scoring function (including parameters for nucleotide atoms, see Chapter 5.1.2) are better than those obtained for the individual docking runs, which means that increasing sampling

variability when using different random initial rotations, followed by redundancy removal with clustering, have improved the docking results.

We further analyzed whether a scoring function previously developed for protein-protein docking was really optimal for protein-DNA docking, since for the latter, electrostatics energy term is expected to have a larger contribution to binding energy due to the higher overall charge of DNA molecules. Moreover, desolvation atomic parameters were previously derived for protein-protein docking in pyDock, but they were not specifically optimized here for nucleotide atoms. To analyze the role of desolvation in protein-DNA scoring, we rescored the generated docking models with the pyDockDNA scoring function but excluding desolvation energy. This greatly improved the success rates, as the curve pyDockDNA (no desolv) shows in Figure 5.2. This indicates that desolvation is not really needed for scoring the protein-DNA docking models generated by FFT-based sampling, perhaps because the parameters have not been yet optimized for nucleotide atoms, or because electrostatics is more relevant in protein-DNA interactions than in protein-protein complexes. We tested other solvation parameters for protein-DNA reported in the literature [161] , but the docking results did not improve (further work is needed on the optimization of these parameters in search of a better desolvation for protein-DNA). In addition, we have also tried other combinations of energy terms, for instance, increasing the factor for van der Waals to 1.0 (we previously found that geometrical complementarity was very important in protein-RNA; [162], or removing desolvation and van der Waals terms from the scoring function to test the relevance of elecrostatics scoring alone, but none of these new combined scoring functions improved the prediction rates (Figure 5.3).

In a rigid-body docking approach as pyDock, it is known that protein flexibility upon binding is perhaps the most determinant factor for docking success. To further analyze whether the docking performance of pyDockDNA is affected by the flexibility of the protein or DNA input molecules during the complex formation, we have grouped the docking results on the protein-DNA docking benchmark according to the flexibility of the protein or the DNA, that is, based on the RMSD between the unbound molecules and the corresponding ones in the complex. Regarding protein flexibility, in order to make groups of similar size,

we defined these three categories: low (unbound-bound RMSD < 1 Å), medium (1 Å ≤ unbound-bound RMSD < 3 Å) and high (unbound-bound RMSD ≥ 3 Å) flexible cases. As for DNA flexibility, we defined these three categories: low (unbound-bound RMSD < 3 Å), medium (3 Å ≤ unbound-bound RMSD < 5 Å) and high (unbound-bound RMSD ≥ 5 Å) flexible cases. The results are shown in Figure 5.3. We can observe that the docking predictive performance does not worsen when protein flexibility is higher (actually, for pyDockDNA with no desolvation, success rates increase when protein flexibility is medium or high).

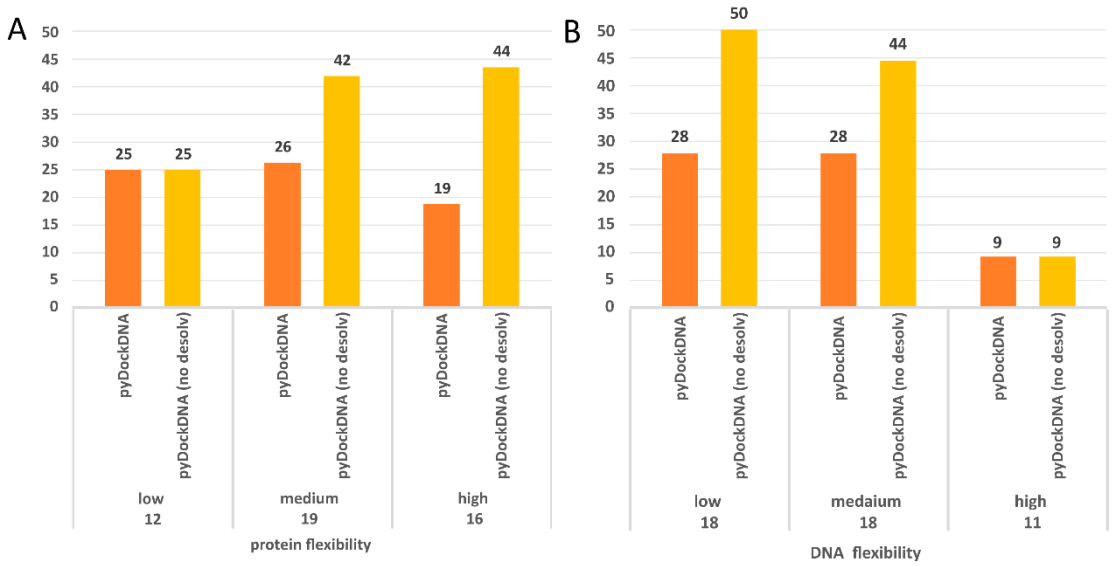


Figure 5.3. Predictive performance for the top 10 models of pyDockDNA (with and without desolvation) on the protein-DNA docking benchmark when cases are grouped according to (A) protein flexibility (low: RMSD < 1 Å; medium: 1 Å ≤ RMSD < 3 Å; high: RMSD ≥ 3 Å), and (B) DNA flexibility (low: RMSD < 3 Å; medium: 3 Å ≤ RMSD < 5 Å; high: RMSD ≥ 5 Å)

However, we can see that the docking performance for highly flexible DNA molecules is dramatically low. We should note that in this benchmark, proteins generally show smaller flexibility (average unbound-bound RMSD 2.6 Å) compared to DNA (average 4.2 Å). In addition, due to the different RMSD cut-off values used to define the flexibility groups for proteins and DNA, the unbound-bound RMSD values for the group of high flexible proteins (average 4.8 Å) are much smaller than for the group of high flexible DNA (average 7.8 Å), which could explain the much worse predictive rates in the latter.

5.4. Application to external case studies

For further testing, we have applied pyDockDNA to a set of ten additional protein-DNA cases (Table 3.3) where the structures for the complex and the unbound protein were available at PDB, and the unbound DNA was modelled in canonical B-DNA conformation (see Methods 3.6.3).

We performed a single pyDockDNA execution for each case study using the randomly rotated unbound protein and DNA structures. This represented a realistic scenario, since the pyDockDNA server only provides results for a docking execution (randomly rotated input structures should be provided to the server in independent executions for a more thorough docking study similar to the benchmark performance analysis above shown). Overall, we obtained predictive success rates of 10% (for the top 10 models) and 30% (for the top 100 models) when using pyDockDNA scoring function, and 10% and 60% (for the top 10 and 100 models, respectively), when using pyDockDNA without desolvation.

Given the small number of cases of these additional set, these values are within the expected range according to the larger docking benchmark set. The most successful case is the complex between the DNA binding domain of Early B-cell Factor 1 (Ebf1) bound to a 22bp DNA (PDB 3MLO), where a near-native docking model (L-RMSD 3.33 Å with respect to the reference) is found with rank 5 when using pyDockDNA (no desolvation) scoring function (Figure 5.4A). When using pyDockDNA (including desolvation) scoring function, this docking model is ranked 6, so it is still within top 10 models. This case has low-flexible protein but high-flexible DNA.

Another case is the complex between the catabolite gene activator protein and a 11bp DNA (PDB 1O3R), where we found an almost acceptable docking model (L-RMSD 10.76 Å with respect to the reference) with rank 5, when using pyDockDNA either including solvation or not (Figure 5.4B). This case has also low-flexible protein but medium-flexible DNA. Incidentally, if this case were considered acceptable, the success rate for the top 10

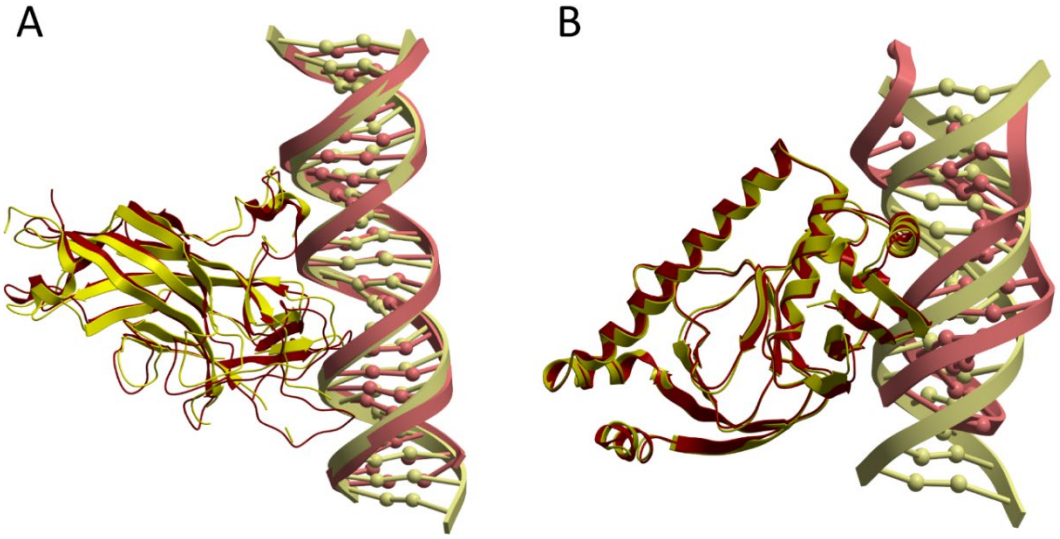


Figure 5.4. Application of pyDockDNA to case studies (A) Near-native model (in yellow) obtained by pyDockDNA docking between a modelled 22bp DNA (receptor) and Ebf1 (ligand). This model was ranked 5 with pyDockDNA (no desolvation) scoring function and has L-RMSD 3.33 Å with respect to the reference (PDB 3MLO; in red) (B) Reasonable model (in yellow) obtained by pyDockDNA docking between the catabolite gene activator protein (receptor) and a modelled 11bp DNA (ligand). This model was ranked 5 with pyDockDNA (either with desolvation or with no desolvation) scoring function and has L-RMSD 10.76 Å with respect to the reference (PDB 1O3R; in red).

would be 20%. However, these percentage values are perhaps not very meaningful considering the low number of cases in this external test set. Interestingly, when using van der Waals term with weighing factor 1.0 (instead of the default factor in pyDock and pyDockDNA, that is 0.1), we find near-native solutions in 3 more cases, in addition to 3MLO: i) 5JLT (L-RMSD 7.08 Å) with rank 1 when using desolvation; ii) 2NTC (L-RMSD 7.25 Å) with rank 3 without using desolvation, and iii) 2PI0 (L-RMSD 6.63 Å) with rank 3 and 2, with or without desolvation, respectively. Therefore, for half of these external case studies, we found near-native docking models within the top 10 models with pyDockDNA, using different variants of the scoring function.

In summary, we present here the pyDockDNA web server to model protein-DNA complexes, which implements a docking method based on pyDock, with new scoring parameters for DNA. We have evaluated the performance on unbound proteins and modelled DNA molecules in canonical B-DNA conformation, using a known protein-DNA docking benchmark. The results show near 40% success rate for the top 10 models when using the pyDockDNA (no desolvation) scoring function, after merging the results from 10 docking executions using different randomly rotated initial structures and clustering the models to remove redundant ones. The method has been applied to external case studies, with similar predictive performance.

5.5. Usage of server

Despite the web server aims to be used to model protein-DNA complexes, many of the submitted jobs (around 50%) during the first weeks of the web server usage aimed to model protein-RNA interactions. Although the server was not developed to model this type of interaction, it is technically possible to submit a job using protein and RNA molecules as input structures. Therefore, we plan to carry out a more in-depth test of our protocol in the future to evaluate the capabilities for predicting this type of interaction, using the available protein-RNA bechamarks [100-103].

The acceptance of the server by the community has been very good, considering its recent publication, and according to the number of visits it is already in second place

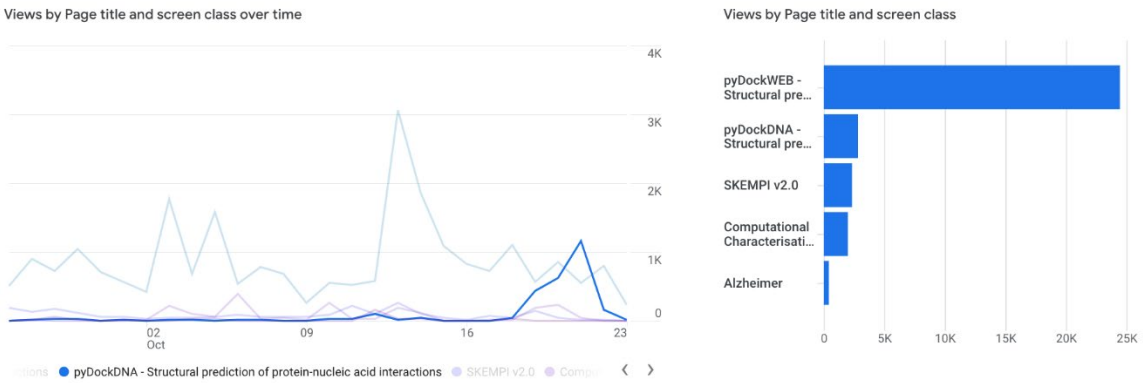


Figure 5.5. Visits received by users to the servers hosted by our research group from 26 September to 23 October.

amongst our tools, after pyDockWeb (which is ranked 1st by far). More details can be seen in Figure 5.5, extracted from Google Analytics (https://analytics.google.com).

6. NEW APPROACHES FOR INTEGRATING *AB INITIO* AND TEMPLATE-BASED DOCKING

The protocols described and the results have been reported in the following publications:

1. Lensink, M.F., et al., Modeling protein-protein, protein-peptide, and protein-oligosaccharide complexes: CAPRI 7th edition. Proteins, 2020. 88(8): p. 916-938.
2. Rosell, M., et al., Integrative modeling of protein-protein interactions with pyDock for the new docking challenges. Proteins, 2020. 88(8): p. 999-1008.
3. Lensink, M.F., et al., Prediction of protein assemblies, the next frontier: The CASP14-CAPRI experiment. Proteins, 2021. 89(12): p. 1800-1823.
4. Lensink, M.F., et al., Blind prediction of homo- and hetero-protein complexes: The CASP13-CAPRI experiment. Proteins, 2019. 87(12): p. 1200-1221

6.1. Introduction

The computational prediction of 3D protein–protein complexes typically includes two distinct strategies: template-based modeling and *ab initio* docking. Template-based modeling methods build a model of a protein-protein complex based on template complex structures, that is, formed between proteins that are homologous to those in the modelled complex, assuming that binding mode will be conserved in this situation. *Ab initio* docking approaches explore the potential binding modes between the interacting proteins through steric and physicochemical complementarity, in cases where no template complex structure is available.

6.1.1. Template-based modeling

A wide range of template-based methods have been developed, exploiting the template information differently. Homology modeling uses sequence identity (S.I.) for template identification and model building [163, 164]; threading methods' thread' sequences onto structural templates [165, 166]; template-based docking usually refers to global superimposition of the structures of unbound monomers onto the corresponding subunits in a template complex structure [52]; and structure interface alignment exploits local similarity and generates models by superimposing the interacting monomers onto the interfaces of templates [167, 168]. However, although template-based methods remain the most reliable [169, 170], they critically depend on the availability of templates.

Interestingly, a study has postulated that the Protein Data Bank, [171] www.rcsb.org; [171] already contains structural templates to model most characterized protein interactions [52]. By aligning the interacting structures onto the monomers of templates, this study shown that in most cases it is possible to find templates whose individual monomers shared TM-score$_{min}$ > 0.4 with monomers of target structures. They also suggested that alignments with TM-score$_{min}$ values greater than 0.4 have the same mode of binding. However, Negroni and colleagues [172], found that while templates indeed exist to model the majority of interactions, they mostly lead to incorrect complex structures, especially in cases of remote homology (e.g cases sharing sequence identity < 30% with templates). To identify templates, they aligned the monomers of target

complexes to the interfaces of templates and observed that the performance significantly deteriorates when templates share only moderate structural similarity with the target (TM-score ~ 0.4 – 0.6), which is at odds with [52] findings.

Another limitation, in addition to the limited availability of templates, is that the low sequence similarity of remote homologous makes template identification difficult when it is based only on the alignment of individual monomer structures. In these cases, *ab initio* docking can be used to build a putative model of the protein-protein complex from the known monomer structures (see below).

We have studied in more detail the capability of template-based modelling under different conditions as well as a new approach integrating *ab initio* docking with template-based modelling that can assist in identifying *ab initio* docking models under conditions of low sequence identity.

6.1.2. *Ab initio* docking

Current *ab initio* methods generate a vast number of conformations using efficient sampling techniques and discriminate near-native models from incorrect poses employing sophisticated scoring functions. Fast Fourier Transform (FFT) sampling algorithms discretize proteins into grids to accelerate the search space process, which are implemented in programs such as GRAMM-X [92], ZDOCK [67] and FTDock [66]. Other approaches for generating docking poses use Monte Carlo-based searching, [129] such as ICM [129], or RosettaDock [173], molecular dynamics as in HADDOCK [174], and normal modes as in ATTRACT [175] or SwarmDock [176]. Various scoring functions have been developed to select, among the thousands of generated docking poses, those ones that are most likely to resemble the native structures. These functions often include electrostatics, desolvation, and van der Waals energy terms such as in pyDock [79] and ZRANK [177] or statistical potentials as in SIPPER [178] or PIE [179]. However, although *ab initio* docking methods have proved valuable in yielding high-quality protein–protein models [180], the limited ability of sampling methods to search the conformational space, and the multiple minima that scoring functions generate lead to an extremely high rate of false positives.

In any case, docking functions can be also used to score template-based complex models from remote templates.

In addition, there is growing interest on repurposing the scoring functions to analyse energetic aspects derived from crystallographic structures and to investigate whether these structures are biologically meaningful (see Chapter 6.6 for more details).

6.2. Limitations of template-based docking

6.2.1. Template-base model generation

We explored here to what extent template-based docking depends on the quality of the available templates, in terms of sequence identity with respect to the target.

To validate the approach, we used as target structures the 176 reference complexes of the protein-protein benchmark version 4.0 (BM4) [132]. We tried to identify suitable templates for these complexes from DOCKGROUND (version 1.1), which contains 7107 non-redundant PDB structures [181]. First, to obtain sequence identity (SI) values, sequence alignments were performed between the unbound target structures of BM4 and the DOCKGROUND complexes using the *SSEARCH* program of the FASTA package version 35.4.7 [182]. The BLOSUM50 matrix was used as a scoring matrix with open and gap penalties of 10 and 0.5, respectively. The E-value equal to 10^{-5}, was considered as a threshold value to consider the alignment statistically significant. Once these alignments were obtained, we performed the relevant SI filters, by removing templates with higher SI than a given value (see Figure 6.1).

In addition, structural alignments were performed between the unbound target structures of BM4 and the interfaces of the 7107 complexes extracted at 12 Å from the DOCKGROUND database, as well as on the complete monomers, by using both TM-align, version 20130511 [59] and MM-align[60], version 20130815, respectively. In the case of TM-align approach, we selected the best combination of the four possible alignments between the templates and the targets of BM4. From the two TM-scores we calculated the averaged TM-score (TM-score_a), and the lowest TM-score (TM-score_{min}).

When MM-align is used, this software automatically generates the best combination, but interface monomers often differ in size, so their contribution to the global

TM-score may be different. To calculate such contribution, we computed individual TM-score of ligands and receptors, aligned previously with MM_align, and the two alignments generated by TM-align, using the TM-score program [183], version 20130511, and calculated the average TM-score.

$$TM\text{-}score_a = \frac{TM\text{-}score_{lig} + TM\text{-}score_{rec}}{2} \quad (6.1)$$

Where TM-score$_{rec}$ refers to the TM-score of the aligned receptors, TM-score$_{lig}$ represents the TM-score of the aligned ligands, and the TM-score$_a$ is simply the average of TM-score$_{rec}$ and TM-score$_{lig}$.

Then, the models are generated by superimposition. When the templates were selected by TM-align (full monomers), the selected template is aligned with MM-align to obtain a rotation and translation matrix, thus generating the model by superposition. When the templates were selected with MM-align (interfaces at 12Å), the rotation and translation matrix is a direct output, which can be used to generate the model in the same way. It should be noted that the results presented here have been obtained by using the BM4 monomers in the unbound conformation.

Finally, models with a solvent accessible surface area (SASA) of less than 250 Å were discarded, in line with the approach used by DOCKGROUND to create the library of non-redundant templates. The quality criterion that defines a template as a near-native structure is that the Cα-LigRMSD between the base model of the template and the real structure is less than or equal to 10 Å. Finally, we filtered by sequence identity: 100%, 95%, 70%, 30%, and TM-score: 0.4, 0.5, 0.6, 0.7 and 0.8.

6.2.2. Predictive success of template-based docking

As can be seen in Figure 6.1, template-based docking shows a success rate over 50% when SI is 100%. This relatively low success rate is mainly due to two reasons. The first reason is due to the low redundancy of DOCKGROUND, which means that there are no high SI templates for all BM4 cases. The second reason is that the unbound monomers were used to generate the final models, so the conformational changes (between unbound and bound monomers) are partly responsible for the models not being as good as could be expected. In fact, this approach is very close to how template-based modelling is done in real life, as in the blind condition of CASP or CAPRI experiments.

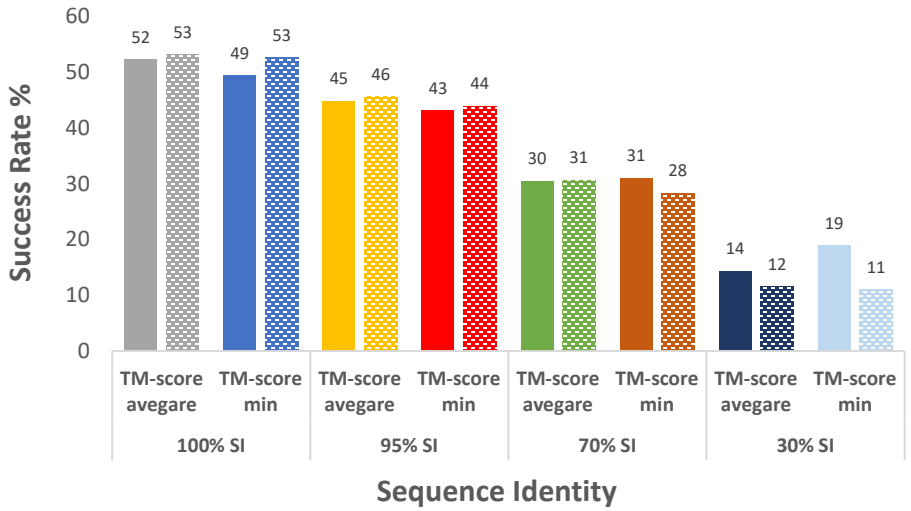


Figure 6.1. Bar-plot showing the success rates of the TM-align (filled bars) and MM-align (patterned bars) software. Templates are filtered according to their SI, so that only templates with lower or equal SI are used to generate the models.

Figure 6.1 also shows that it makes virtually no difference whether we use MM-align or TM-align to select the templates. But in the so-called "twilight zone", that is, in cases with SI below 30%, the use of MM-align provides a significant advantage.

In the case of the MM-align results, we performed a more detailed study (Table 6.1), by analysing the success rates at different threshold values of sequence identity and TM-scores. The success rate is usually calculated on the total number of the targets of BM4, but here we have done it on the number of cases for which a successful model can be made. Thus, to put the success rate values in the correct context, the coverage (%) is also displayed. Consequently, the success rates are apparently higher than in Figure 5.1. For

example, when the TM-score is set to 0.4, the TM-score$_{min}$ values are better in all SI conditions as compared to TM-score$_a$, but the coverage decreases dramatically between them. This can also be observed for all TM-score thresholds.

Table 6.1. Template-based docking success rate for the top 10 models and their coverage (percentage of cases for which a model can be generated over the 176 of the BM4) for the applied TM-scores and sequence identity thresholds.

Representative template-base scoring	Success Rate (%)				Coverage (%)			
TM-score$_a$ 0.4	55.6	47.8	33.6	17.3	93.1	91.4	87.4	76.4
TM-score$_{min}$ 0.4	84.2	84.1	74.5	61.5	54.6	47.1	31.6	14.9
TM-score$_a$ 0.5	66.9	61.0	46.9	35.6	71.3	67.8	56.3	33.9
TM-score$_{min}$ 0.5	85.9	86.1	80.0	75.0	52.9	45.4	28.7	11.5
TM-score$_a$ 0.6	83.5	83.3	73.3	64.0	55.7	48.3	34.5	14.4
TM-score$_{min}$ 0.6	86.2	86.1	80.5	76.9	50.0	41.4	23.6	7.5
TM-score$_a$ 0.7	84.6	84.4	80.0	71.4	523	44.3	25.9	8.0
TM-score$_{min}$ 0.7	87.2	87.1	81.8	85.7	49.4	40.2	19.0	4.0
TM-score$_a$ 0.8	88.2	88.4	81.8	100.0	48.9	39.7	19.0	2.9
TM-score$_{min}$ 0.8	89.0	89.1	80.8	100.0	47.1	36.8	14.9	2.3
	100	95	70	30	100	95	70	30
	Sequence Identity							

Both TM-score$_a$ or TM-score$_{min}$ have advantages and disadvantages. In the case of TM-score$_{min}$ it could be difficult to find a template with a TM-score over 0.4, but if found, it will likely lead to the generation of good models. The opposite happens with TM-score$_a$. It is easy to make models for almost all targets, but at a cost of a lower quality. Therefore, in order to obtain acceptable models, a threshold higher than 0.4 will be needed. More details on the resulting success rates can be found in Appendix 2: Table 10.2.1

6.3. *Ab initio* docking can improve template identification

6.3.1. A new protocol for combining *ab initio* and template-based docking

We have devised a modeling strategy that uses *ab initio* docking to improve template identification, in three stages: i) sampling docking orientations with *ab initio* docking, ii) searching structural templates for docking orientations using the MM-align program, and

iii) scoring docking orientations using a new function that combines pyDock docking energy and TM-score. Figure 5.3 illustrates the overall procedure, and the details are as follows:

***Ab initio* docking**

We used the pyDock scheme to generate a pool of docking models.

Sampling

The unbound subunits of the Benchmark 4.0 complexes were translated and rotated randomly to remove possible bias of initial binding conditions. The docking poses were generated using FTDock 2.0 [66], a fast Fourier transform algorithm, which is based on surface complementarity and electrostatics, using 0.7 Å grid cell size, surface thickness of 1.3 Å, a rotation angle of 12º and 3 translations for each rotation. For each benchmark case, a total of 10,000 docking poses were obtained.

Scoring

We used the pyDock scoring function [184], developed in our group, which incorporates electrostatics, solvation, and van der Waals energy contributions to rank the docking poses. For each target case, we selected the top 100 ranked docking conformations.

Re-scoring docking conformations using the structural similarity score (TM-score)

i. **Interface extraction**

   Interfaces were extracted from the top 100 selected benchmark docking poses and from the DOCKGROUND template library by selecting only those residues with Cα atoms within 12 Å distance from the interface [185]

ii. **Structural alignment of target docking interfaces with templates**

We used the MM-align program [60], version 20130815, to structurally align the whole target docking interfaces with those of the templates. MM-align employs a modified Needleman-Wunsch dynamic programming algorithm to generate an optimal alignment between the two interfaces, which is subsequently used to calculate the global TM-score, a measure of the structural similarity [183].

Because interface monomers often differ in size, their contribution to the global TM-score may be different. To calculate such contribution, we computed individual TM-score of ligands and receptors, aligned previously with MM_align, using the TM-score program [183], version 20130511, and calculated the average TM-score as follows:

$$TM\text{-}score_a = \frac{TM\text{-}score_{lig} + TM\text{-}score_{rec}}{2} \quad (6.2)$$

where $TM\text{-}score_{rec}$ refers to the TM-score of the aligned receptors, $TM\text{-}score_{lig}$ represents the TM-score of the aligned ligands, and the $TM\text{-}score_a$ is simply the average of $TM\text{-}score_{rec}$ and $TM\text{-}score_{lig}$. MM-align and TM-score programs were adjusted to normalize the TM-score by the shorter length of the aligned pair. Template selection was made using $TM\text{-}score_a$, due to the high coverage achieved by this scoring function (see Chapter 6.2.2).

To ensure that only remote homologous are considered in the analysis, templates sharing more than 30% sequence identity with the target benchmark complexes were removed. Sequence similarity alignments were performed by the *SSEARCH* program included within the FASTA suite, version 36.3.8b [182]. The program was set to use the BLOSUM50 scoring matrix, and open and gap extension penalties of 10 and 0.5, respectively, as in previous studies [172]. Only alignments with E-values lower than 10^{-5} were considered as valid. Full-length protein sequences were used for generating the alignments and extracted from experimentally determined PDB structures.

iii. Integration of the pyDock score with the structural similarity score (TM-score)

We defined a new scoring function that integrates *ab initio* docking and template-based modeling. The scoring function incorporates the normalized pyDock docking energy and the TM-score$_a$ obtained for every model, and was defined as follows:

$$TM\text{-}score_a + PyDock = Zscore_{TM\text{-}score_a} + Zscore_{pyDock}\,, \tag{6.3}$$

where $Zscore_{TM-score_a}$ and $Zscore_{pydock}$ are the normalized values of TM-score$_a$ and pyDock for target docking case. To normalize the TM-score$_a$ values, we computed the $Zscore_{TMa}$ for each model by using the mean (u) and standard deviation (σ) of TM-score$_a$ over the 100 selected models for each target case, as follows:

$$Zscore_{TMa} = \frac{TM\text{-}score_{a_i} - \mu_{TM-score_a}}{\sigma_{TM-score_a}} \tag{6.4}$$

To normalize the pyDock energy, we computed the $Zscore_{pyDock}$ for each model by using the mean (u) and standard deviation (σ) of pyDock values for each target case, as follows:

$$Zscore_{pyDock} = \frac{pyDock_i - \mu_{pyDock}}{\sigma_{pyDock}} \tag{6.5}$$

We used the pyDock scoring function (see Chapter 3.1.4.3) to select the best 100 models, then the models were reranked according to the TM-score(s) and the integrated scoring function. In addition to the scoring function defined here, the combination of pyDock with TM-score$_{min}$ was also tested but yielded inferior results. The success rate was defined as the percentage of target complexes with at least one near-native structure within a specific set of predictions (top 1, top 5, top 10).

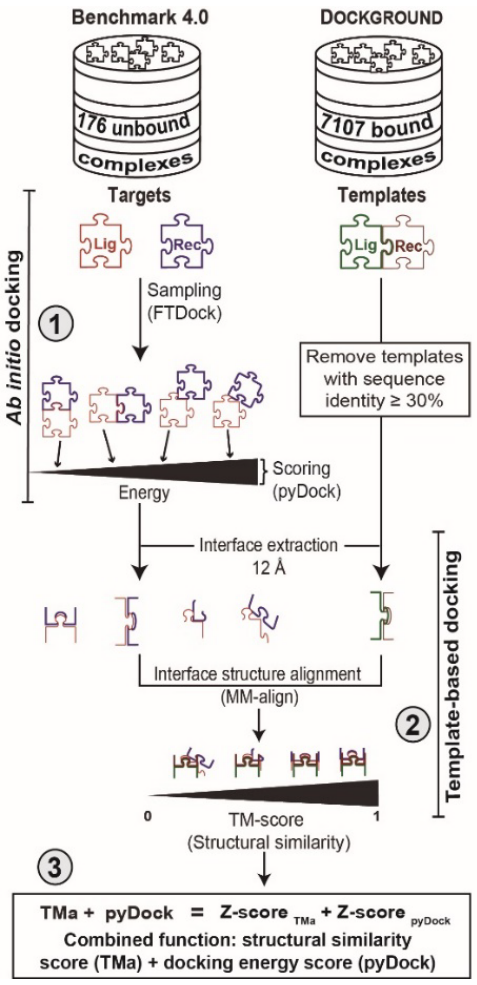


Figure 6.2. Benchmarking flowchart of a new protocol that combines *ab initio* docking and template-based modeling. The approach includes three stages: (1) *Ab initio* docking generates a pool of docking models, which are scored using the pyDock function; (2) the top 100 docking models (according to pyDock score) are selected, and their interface extracted. MM-align structurally aligns docking and template interfaces to obtain the best template (according to the TM-score) for each docking orientation; (3) the combined function (TMa + pyDock) is used to score the docking orientations to obtain the final selection of models

6.3.2. Evaluation of the combined docking protocol.

First, the above mentioned docking protocol was evaluated in a scenario where only templates within the so-called twilight zone are available, for which only alignments with a TM-score value greater than or equal to 0.4 were considered (see Figure 6.3). In this scenario, the combination of pyDock scoring with average TM-score average provided the

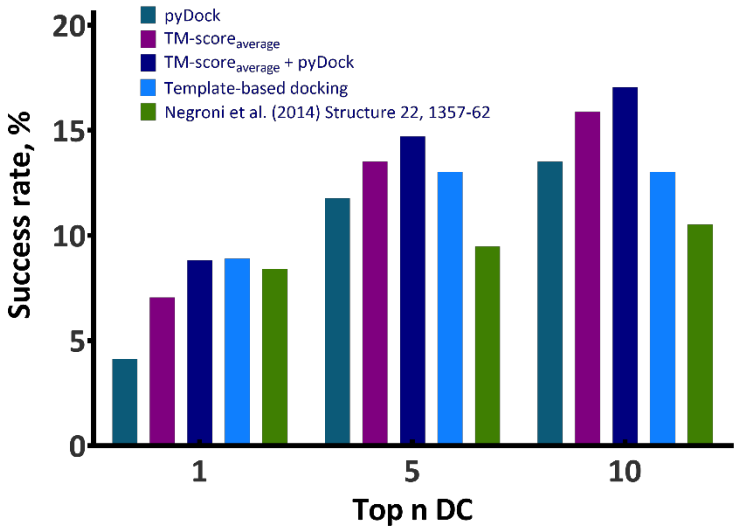


Figure 6.3. Comparison of the success rates for the top 10 models obtained for template docking, *ab initio* docking and the integrated protocol (TMa + pyDock). The score functions were applied on the 176 cases of the BM4 using MM-align to select the templates. For comparison, we also show the success rates for template-based when only the dimer of the BM4 and TM-align were used to select the templates as previously reported

best results.

The success rate of this combined scoring function (considering the top 10 models) achieved 18.8% if no TM-score threshold value is applied. The success rate for the combination with the TM-score min is 15.3%, very similar to the 15.9% success rate obtained with the pyDock scoring function alone.

Following a similar analysis as for the template-based models, here we also explored the success rates for different threshold values of sequence identity and TM-score.

Contrary to what was previously observed in the template-based models, the use of average TM-score shows both a higher success rate (up to 17.1%) and a higher coverage when using templates within the twilight zone (30% SI). It is also observed that above the TM-score threshold of 0.6, there is no benefit in using either of the two combined scoring

functions. The same is true for the use of templates with 100% and 95% SI. But in the case of templates between 70% and 30% SI, using TM-score values between 0.4 and 0.5, the combined functions are shown to be beneficial. Especially in the twilight zone in which TM-score$_a$+pyDock achieves reasonable predictive performance according to success rate and coverage. For more information on the resulting success rates, please see Appendix 2: Table 10.2.3.

In summary, when only low-quality templates are available (based on TM-score or SI), the combination of *ab initio* sampling with structure-based (using a template) and energy-based scoring functions, like TM-score and pyDock, respectively, could help to generate acceptable models.

Table 6.2. *Ab initio* docking selection models, using the template-based docking success rate for the top 10 models and their coverage (percentage of cases over the 176 of the BM4 for which a model can be generated) for the applied TM-scores and sequence identity thresholds

Representative scorings		Success Rate (%)				Coverage (%)			
		100	95	70	30	100	95	70	30
TM-score$_a$	0.4	28.0	25.7	21.3	15.9	99.4	99.4	98.9	96.6
pyDock+TMs-ave	0.4	29.1	27.4	22.4	17.1				
TM-score$_{min}$	0.4	40.2	33.7	25.0	8.2	49.4	48.9	40.9	34.7
pyDock+TM-score$_{min}$	0.4	40.2	33.7	42.9	8.2				
TM-score$_a$	0.5	29.5	26.5	21.5	16.8	88.6	88.1	84.7	67.6
pyDock+TMs-ave	0.5	30.1	27.7	22.1	16.8				
TM-score$_{min}$	0.5	61.4	55.6	52.0	20.0	25.0	20.5	14.2	5.7
pyDock+TM-score$_{min}$	0.5	61.4	55.6	52.0	20.0				
TM-score$_a$	0.6	44.7	39.5	33.9	31.3	48.3	43.2	33.5	9.1
pyDock+TMs-ave	0.6	43.5	39.5	32.2	25.0				
TM-score$_{min}$	0.6	71.9	66.7	66.7	66.7	18.2	13.6	10.2	1.7
pyDock+TM-score$_{min}$	0.6	71.9	66.7	66.7	66.7				
TM-score$_a$	0.7	75.7	72.4	72.2	66.7	21.0	16.5	10.2	1.7
pyDock+TMs-ave	0.7	75.7	72.4	72.2	66.7				
TM-score$_{min}$	0.7	74.1	77.8	76.9	100.0	15.3	10.2	7.4	0.6
pyDock+TM-score$_{min}$	0.7	74.1	77.8	76.9	100.0				
TM-score$_a$	0.8	76.9	77.8	83.3	100.0	14.8	10.2	6.8	0.6
pyDock+TMs-ave	0.8	76.9	77.8	83.3	100.0				
TM-score$_{min}$	0.8	88.9	84.6	85.7	100.0	10.2	7.4	4.0	0.6
pyDock+TM-score$_{min}$	0.8	88.9	84.6	85.7	100.0				

Sequence Identity

6.4. Docking functions can be used to score models from template-based docking

6.4.1. Automatic processing of input structures

We have explored other protocols to identify correct models from template-based docking in CASP and CAPRI community-wide experiments. In the 7thCAPRI (7th Critical Assessment of PRedicted Interactions) experiment, when one or more protein subunits were not available for docking and needed to be modelled, MODELLER 9v19 was applied with default parameters [186] using the template(s) suggested by the organizers or other homologous proteins found by the BLAST search tools [187] . The final selected model was the one with the lowest DOPE score [188]. In some cases (e.g. T131, T132, T133, T167), loops and highly flexible areas were removed before docking to avoid problems when generating the models, and these areas were reconstructed in the final models by filtering out those that might have clashes. When the monomers were structurally determined, the cofactors, water molecules and solvent ions, were removed during the preparation of the molecules for docking. In some cases (T123, T124, T136), we also performed multiple template modelling of any of the interacting subunits with I-TASSER [189]. In the case of the peptides (T134-T135), they were modelled similarly, based on the available templates. In target T130, the input structure of the receptor had an SI of 99%, except for a mutation at residue 9. This amino acid was mutated using SCWRL3 [190] from Gly to Asn. All templates and inputs PDBs used in the 7th edition of CAPRI are available in the Appendix 2: Table 10.2.4.

In contrast, in the 3rd and 4th joint CASP-CAPRI protein assembly prediction challenge, monomers structures were available a few days after the CASP server competition opened. In general, we used the monomer structures from the CASP server competition, to generate the models based on *ab initio* docking. The chosen servers were the best performers in previous editions: ZHANG, ROSETTA and QUARK. In targets T165 and T177, some interfaces were modelled with MODELLERv9.19 (template-based modeling) because there were not available models at the CASP-hosted servers. Cofactors, water molecules and solvent ions were not considered for scoring.

6.4.2. Template-base docking

Template-based docking was often used in both joint CASP-CAPRI protein assembly prediction challenges.

In the 3rd joint CASP-CAPRI challenge (CASP13-CAPRI46), complexes were modelled based on templates in almost 50 % of the cases (T137-T144, T152-T154, T158). In the 4th challenge (CASP14-CAPRI50) this percentage increased to 70 % of the cases (T164-T168, T170, T171, T175-T177, T180, T181). The templates were found thanks to the information provided by the models extracted from the CASP-hosted servers: ZHANG, ROSETTA, QUARK, MULTICOM-CONSTRUCT and RAPTORX-DeepModeller, as well as from a BLAST search. These templates were analysed and selected based on their structural similarity and biological unit of interest. Templates that did not add relevant information were also filtered out (i.e. redundant templates were removed). Monomeric models were superimposed on the corresponding subunits of each non-redundant template and subsequently scored with the pyDock scoring energy function.

6.4.3. *Ab initio* docking

In general, for the protein-protein and protein-peptide cases, we generated 10,000 docking poses with FTDock 2.0 [66] (with electrostatics and 0.7 Å grid resolution) and 2,000 poses with ZDOCK 2.1 [67], with the exception of a few targets (T131, T132, T136, T149, T159), where ZDOCK was not used due to the computational cost of very large proteins. In three cases (T131-T133), we used the stochastic docking method LightDock [77] to generate additional flexible docking poses during the sampling process.

In the scoring phase, the docking poses were scored with the pyDock scoring function. In the above mentioned T131-T133 cases, the pyDockLite [77] and DFIRE [191] functions were used. In this default protocol, cofactors, water molecules and solvent ions were not included in our docking calculations. In case of homo-oligomeric targets, we kept only the docking positions with the expected symmetry (e.g. C2 for homo-dimers, C3 for homo-tetramers, etc.).

For protein-saccharide targets (T126-130), we used rDock [192] (http://rdock.sourceforge.net/) to generate and score the models. In addition, we developed a new pyDock module specially adapted for scoring saccharide molecules.

6.4.4. Combining *ab initio* and template-based methods

In the 7th CAPRI experiment and in the 3rd and 4th joint CASP-CAPRI protein assembly prediction challenges, some targets were modelled by combining template-based and *ab initio* docking.

In the T136 target of 7th CAPRI, the homo-decamer interfaces were modelled based on the available BLAST template, by superimposing the binary *ab initio* docking models on the global template (PDB 5FKZ).

In the 3rd joint CASP-CAPRI experiment, there were at least three targets where the same strategy was used:

- T146 (A2B2): the homodimer interfaces were modelled based on all available templates from CASP-hosted servers, and the heteromeric interfaces were obtained from *ab initio* docking. We used Cryo-EM information[193] to localize the ligand-protein and filter the docking results.
- T147 (A8): available templates (PDB codes 2W1V, 2GGL and 5H8I) were used to generate homodimeric models, then *ab initio* docking was performed to build tetramers, keeping only those with 2-fold helical symmetry.
- T159 (A6B6C6): the three homo-hexameric rings were independently modelled based on the templates found in the CASP-hosted servers (PDB ID: 1Y12, 3EAA, 4HE1 y 3V4H). Then, using the 3J2M template, we built the hetero-dodecamer. From this macrostructure and the 3rd ring, rigid body docking was applied to model the final homo-octadecamer, keeping only those models where the interacting rings overlapped in a consistent way.

In the 4th joint CASP-CAPRI experiment, there were challenging targets where the combined sampling strategy was applied:

- T165 (A3H3L3): the homotrimeric glycoprotein was modelled by template-based docking (CASP models to available templates were used), while the hetero-dimeric antibody was modelled with MODELLERv9.19 because no models were available on the CASP-hosted servers. These models were docked to form the final model.
- T170 (A6B3C12D6): in this case we applied an *ad hoc* modelling procedure combining *ab initio* docking, template-based modelling and manual fitting using a cryogenic electron microscopy (cryo-EM) map. The target consists of three rings with different stoichiometry and protein composition. The first ring was a homo-hexamer, arranged as a dimer of trimers, and was modelled by structural superimposition fitting the X-ray monomer structure (PDB 5NGJ, chain A) in the available cryo-EM map of the tail of bacteriophage T5 (EMDB ID: 3689). The second ring is formed by three protein subunits of one type and twelve of a second type and was modelled by sequentially building binary interactions with *ab initio* docking and symmetry constraints. The third ring was modelled in a similar way.

  The final assembly of the modelled rings was performed by *ab initio* docking, selecting only those models in which the symmetry axes of the rings were aligned (see Figure 6.4).

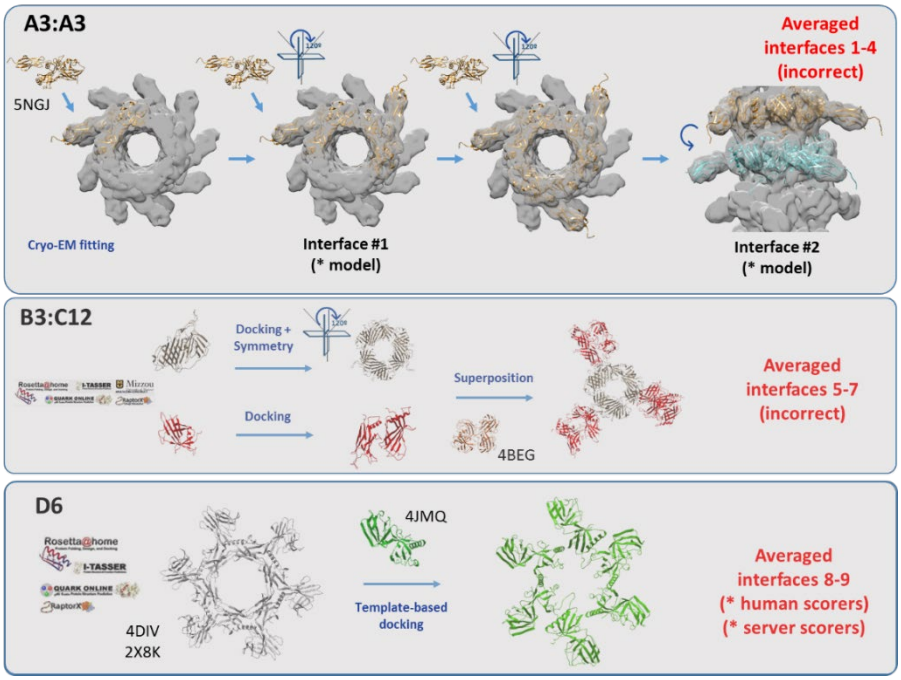


Figure 6.4. Modeling strategies for the three major rings of target T170

- T177 (A20): this complex is formed by two homodecameric rings. Each ring was modelled with MODELLERv9.19 from available templates, and the final assembly was built by applying *ab initio* docking to the two modelled decamers.

6.4.5. Inclusion of restraints from available external data

Besides the above mentioned automatic modeling procedures, we often used available data for each specific case in order to restrain the docking orientations and help selecting the correct models. Some of the most important restraints we applied were based on the oligomerization states of the targets. This information was mainly obtained from the CAPRI and CASP description of the targets, but we also extracted it from templates or oligomerization state predictors [194]. If homo-oligomers were involved, we assumed symmetric oligomerization, e.g. cyclic symmetry C2, C3..., to filter the resulting docking models. This strategy was widely used in the 7th CAPRI (T125, T124, T136) as well as 3rd (T137-T141, T143-T144, T147, T148, T152-T154, T158, T159) and 4th joint CASP-CAPRI experiments (T164, T165, T167-T171, T174-T176, T178, T179).

If experimental information was also available, we included it in the modeling procedure as distance restraints. We used a variety of restraint sources (see Figure 6.5). For example, we estimated interface residues to be used as docking and scoring restraints from homologous protein structures or conserved protein-protein interactions. More specifically, these were the restraints applied in each of the experiments:

7th CAPRI:

In the case of the T126-T130 targets (protein-saccharide), a more specific software was used to generate the rDock complexes [195] (http://rdock.sourceforge.net/). rDock can include constraints to target the binding cavity. The center of the cavity used was defined as the center of mass of known ligands bound to homologous proteins (PDB 5F7V for T126-T129; PDB 3D5Z for T130).

The pyDockRST [159] module was used in multiple targets to efficiently add distance restraints. In some cases, we found homologous templates from which we deduced the restraints to be applied: In targets T134-T135 (homologous template: PDB 1F95) a distance

restraint of 5 Å with respect to the residues of the peptide was used. Similarly, a distance restraint of 10 Å was applied in targets T153 (homologous template: PDB 3W36) and T136 (homologous template: PDB 5FKZ). In other targets, we directly applied information about the interaction that was available in the literature, such as in T122 (Trp156 and IL-23A) [196], T125 (LLT1 Lys169, and NKR-P1 Glu205) [197, 198], and T131-T132 (Tyr35 and Ile92 in the common hCEACAM1 protein) [199].

3rd joint CASP-CAPRI experiment:

The target T149 was highly challenging as it not only involved the dimerization of a 5-domain protein, but it was also necessary to describe the assembly of the 5 different domains (D) within each monomer. The pyDockTET module [141] was used together with an *ad hoc* strategy to generate the models. Each domain was modelled independently based on the rank 1 prediction of the QUARK CASP-host server (each domain was a CASP13 target). Next, the intermolecular orientation between the first domains of each monomer (D1-D1') was modelled based on a template (PDB 1DQS). The interaction between D1 and D2 of the same monomer was modelled by docking, imposing restraints derived from the inter-domain bonds with the pyDockTET module. For each D1-D2 model, a copy of it (D1'-D2') was superimposed on D1-D1' to generate D2-D2' pairs. This strategy was iteratively applied to the other domains (D2-D3 by docking, D2'-D3' by overlapping, D3-D4 by docking, etc.). A visual curation of the models was performed to avoid large clashes between domains and then the models were scored using the pyDock scoring function.

Another interesting targets were T149, T150 and T151, which were sequentially released for the same protein complex, but with increasing available experimental information. In target T150 we re-evaluated the *ab initio* docking orientations generated for target T149 with the help of SAXS data, for which we used the pyDockSAXS module. In the case of T151 we also added the newly available cross-linking information in the form of distance restraints using pyDockRST.

4th joint CASP-CAPRI experiment:

Target T168 was a trimer. We initially built docking trimers with *ab initio* docking and symmetry constraints, which were compared with an available template (PDB 6FTD), so that models with Cα-RMSD larger than 10 Å were filtered out. In the case of T170, we used a cryo-EM map to select the final models. In target T181 we also applied restraints, since the structure of the separate proteins (PDB IDs 1N3U and 6XDC, respectively) was known. We also knew that 6XDC had a transmembrane domain (residues 44-64, 68-128) to which 1n3u could not bind. This region was used as a "negative" restraint, eliminating the *ab initio* docking models that showed binding in this region.

6.4.6. Final selection of the models.

In general, the scoring of the models, both for the predictors and scorers experiments, was performed with the pyDock bindEy module, which calculates the docking energy of pyDock

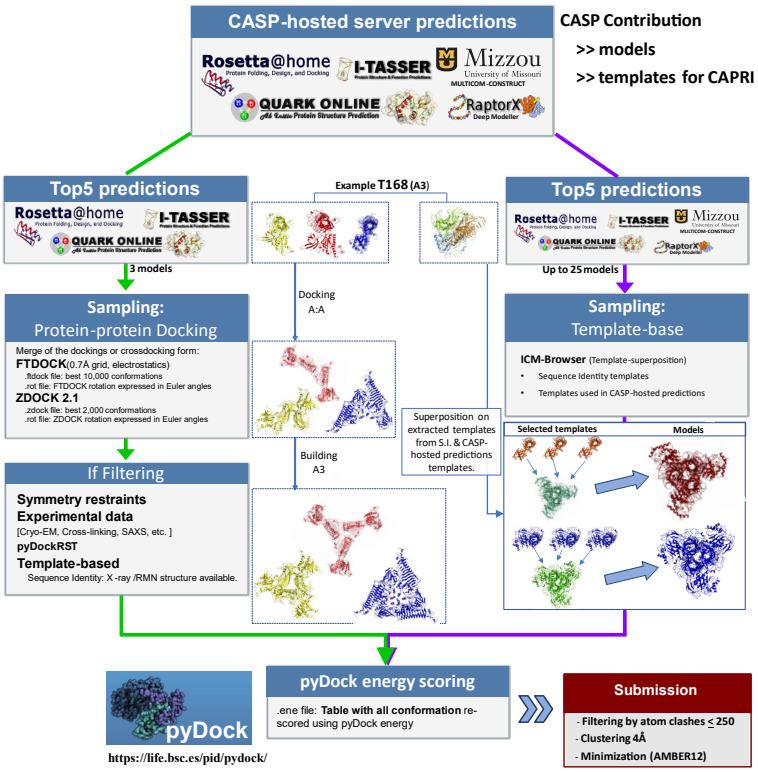


Figure 6.5. Protocol followed in the joint CAPRI-CASP14 experiments using the T168 target as an example. The combination of template-based modelling and *ab initio* docking is shown. It also shows how templates and relevant information can be used to filter between the *ab initio* models.

[80] (for details, see Chapter 3.1.4.3) for a given (experimentally determined or modelled) protein complex. For targets where possible interface residues could be defined based on available experimental information or homologous complexes, this information was usually included in the final score as distance restraints with pyDockRST [159], pyDockSAXS [140] and/or pyDockTET [141], as described in previous section.

Other scoring strategies were also used for specific targets, as in T133, an artificially designed complex based on an old target (T47). Such a redesign increased the prediction complexity with respect to T47 and allowed us to explore different strategies. Therefore, we changed the traditional pyDock scoring method successful applied to T47 by integrating new methodologies such as lightDock [77], which added more flexibility to the models, and CCharPPi [78], which provided a more significant number of scoring functions, which were integrated using the IRaPPA voting algorithm [86]. This protocol is implemented in the pyDockRescoring (https://life.bsc.es/pid/pydockrescoring/) server. For the protein-peptide targets of the 7th CAPRI experiment (T134, T135 and T121), we constrained the docking models to adopt the anti-parallel β-chain orientation, which turned out to be correct for targets T134, T135, but incorrect for target T121. After scoring, we removed redundant predictions using a BSAS algorithm [153] with a distance limit of 4.0 Å, as previously described [200]. For models based on templates or symmetry constraints, we eliminated those with strong clashes that would be difficult to solve with minimization.

The number of available templates and their reliability determined the percentage of template-based complex models included in the final 5 or 10 models that we submitted for CASP or CAPRI, respectively. Models with more than 250 collisions (i.e., intermolecular pairs of atoms closer than 4 Å) were also eliminated. Then the final ten selected docking poses were minimized using different versions of AMBER (AMBER12 [193] or AMBER17 [194]), with implicit solvent to improve the quality of the docking models and reduce the number of interatomic clashes, as previously described [196]. We always used the same atomic parameters: from AMBER ff99SB [201] force field for proteins, and gaff force field for polysaccharides [202]. The minimization protocol consisted of a 500-cycle steepest descent (SD) minimization with harmonic constraints applied at a force constant of 25 kcal/(mol-Å2) to all backbone atoms in order to optimize the side chains, followed by

another 500 cycles of unconstrained conjugate gradient (CG) minimization. In some cases, due to time constraints or earlier convergence, the minimization protocol varied (e.g., in T134 we used 200-cycle SD and 300-cycle GC; in T135 500-cycle SD and 100-cycle GC; in T136 some models were not minimized or were vacuum minimized; the larger CASP targets T149-151, T159, T165, T170, T177, and T180 were vacuum minimized). In targets T131 and T132, loops previously removed for docking were reconstructed by MODELLER before the final minimization step.

The protocol we used for the final selection of models in the scorers experiment was the same as the one we used in the predictor experiments, except for a few exceptions, as follows: Distance restraints were not used as scorers in T121 and T136; IRaPPA was not used as scorers in T133. In target T174, models with a Cα-RMSD<10 were filtered against a common template domain. In target T175, no template was used in scorers (while it was used in predictors). In T181, in addition to filtering the models using the 6XDC transmembrane region (only the A chain was used for *ab initio* docking), the interface region with the other monomer, estimated from the template (amino acids 221-288), was also used to filter the models.

6.4.7. Scoring of protein-saccharide complex models

For the protein-oligosaccharide targets (T126-T130), we used rDock [192] (http://rdock.sourceforge.net/) to generate and score the models. In addition, we had to implement new functionalities in pyDock (see Chapter 4.4), since the original version did not have atomic electrostatics, solvation, and van der Waals parameters for saccharide molecules.

After this implementation, pyDock was able to read topology and coordinate files from AMBER for all types of molecules, and thus calculate the energy-based scoring function. The data obtained from AMBER files are van der Waals energies and atomic partial charges. As for the atomic solvation parameters (ASPs), a dictionary of equivalences was created between the new AMBER atom types and the pyDock atom types originally used for proteins (https://github.com/pyDock/parallel/blob/master/amber_old_to_new.map). Basically, the ASPs for saccharide C and O atoms were considered as those for "C aliphatic"

and "O hydroxyls" of original pyDock, respectively. To obtain the AMBER files mentioned above, we used antechamber with the AM1-BCC charge model [203], setting the net charge to 0, and then parmchk2 to obtain the charges, energy angle parameters, and a mol2 file. We then used LEaP to load the general AMBER force field (GAFF) and followed the procedure to generate a library with the information obtained from the antechamber. As a final step, we used LEaP to load each docking pose and obtain its coordinate (.incrd) and topology (.prmtop) files. These two files are the ones that can be directly used by pyDock to calculate the energy with the bindEy module. In the scoring experiment, we used another charge model due to time constraints, the empirical atomic partial charges of Gasteiger-Marsili [203], and the final score was based solely on this new version of pyDock adapted to glycosidic protein interactions (rDock was not used).

6.5. Evaluation of the predictive results in CAPRI and CASP

The models submitted to CAPRI and CASP with the developed methodology described in the previous section were officially evaluated by the organization of CAPRI and CASP, which is a useful exercise that allows us to have a more objective knowledge about the applicability and limitations of our methodological approaches, and a fair comparison between methods from other groups.

In the 7th edition of CAPRI, we participated in all targets, as predictor, scorers and servers, the latter with the exception of the protein-saccharide cases since our pyDockWeb [82] server was not ready for automatic processing of this type of interactions. The predictive performance of our group as well as that of other participants is described in full detail in a previous publication [204].

In the case of the 3rd and 4th joint CASP-CAPRI experiments, we participated as predictors and scorers. The predictive performance was described in full detail in previous publications [142, 144, 145]. Below we summarize our results for the 51 targets proposed in the 7th CAPRI, 3rd and 4th joint CASP-CAPRI experiments (considering the hetero-meric and homo-meric interfaces at the T125 target as two separate targets). The performance for 7th CAPRI is summarized in Table 6.3, and in Appendix 2: Tables 10.2.6, 10.2.7, 10.2.8,

for the 3rd joint CASP-CAPRI in Table 6.4 and in Appendix 2: Table 10.2.10, and for the 4th CASP-CAPRI in Table 6.5 and in Appendix 2: Table 10.2.11.

6.5.1. 7th CAPRI

The results are consistent with previous participations, as we submmited acceptable models for 10 targets as predictors, four as servers and 13 as scorers (Table 6.3). The total number of evaluated interfaces was 19, as there were multiple interfaces that were considered independent targets. These results, considering our 10 best models, represent a success rate of 53% as predictors, 21% as servers and 68% as scorers, the latter being the best of all participants. Using the CASP criteria in which only the top 5 submitted models are evaluated, the results as predictors and servers did not change, but the performance as scorers was significantly reduced. Based on the results from the evaluation for all participants, the organization considered that there were nine difficult targets, three of medium difficulty, and five easy targets. The performance summary is represented in Table 6.3, which shows the quality of the 10 best predictions submitted for each interface and/or target in which we participated (high***, medium**, and acceptable* [204]).

6.5.1.1. Successful predictions

In most of the targets, if the submitted models for predictors were successful, they were also successful for scorers. In servers, we submitted an acceptable model for T122, while we were not successful as predictors in the same target. For the scorers, in addition to the targets in which we were successful as predictors, we were successful in four additional

Table 6.3 Quality of submitted predictions by our group for the 7th CAPRI experiment predictions.

Target	Level	Stoich.	Submission quality for: Predictors	Servers[1]	Successful groups[2]	Submission quality for Scorers[1]	Successful groups
Protein-protein complexes							
T122	Difficult	A1B1C1	-	**M08***	11/35	**M08** (M05*)**	9/20
T123	Difficult	A1B1	-	-	0/32	-	0/19
T124	Difficult	A2B1	-	-	0/29	-	0/18
T125h[5]	Difficult	A2B4	-/-	-/-	15/1 of 30	**M02**** / -	18/0 of 19
T125	Difficult	A2B4	**M01***/ -	**M01***/ -	26/0 of 30	**M02*** (M01**)** -	17/0 of 19
T131	Difficult	A1B1	-	-	1/30	**M06****	4/19
T132	Medium	A1B1	-	-	3/30	**M08****	6/19
T133	Easy	A1B1	**M01****	**M05***	30/35	**M02****	18/20
T136	Easy	A10	**M01**/M02*/ M07***	- [M01**]/- [M01*]/-	26/29/26 of 30	**M02** (M01*)/M02**/ M08** (M02*)**	16/16/16 of 16
Protein-peptide complexes							
T121	Difficult	A1B1	-	-	5/33	-	8/17
T134	Easy	A2B1	**M03***	-	15/31	**M09****	15/18
T135	Easy	A2B1	**M01***	**M02***	23/30	**M01*****	17/17
Protein-oligosaccharide complexes							
T126	Difficult	A1B1	**M08** (M03*)**	-	19/29	**M08***	16/17
T127	Difficult	A1B1	**M01***	-	26/28	-	13/14
T128	Medium	A1B1	**M02***	-	28/29	**M10** (M01*)**	17/17
T129	Medium	A1B1	**M02***	-	27/30	**M09** (M02*)**	16/16
T130	Easy	A1B1	**M02* (M01*)**	-	26/29	**M06****	17/17

[1]In general, we indicated the best quality models within the top 10 submitted models (in parenthesis, we indicated if there were another successful model within the top 5 or top 1). T136: in square brackets there are quality models that were disqualified due to clashes. ***=high-quality models; **=medium-quality models; *=acceptable-quality models for the results in the performance [176].

[2] In general, docking servers are included in predictors groups. [3]Target T125, corresponding to heteromeric interfaces.

targets (T122, T125, T131, T132). Target T132 was classified as medium, but the other ones were considered difficult.

The availability of templates and experimental information made a difference in the predictive results. Target T122, despite an available template (PDB 1I1R, 25% SI), was a

difficult case for which only 9 groups obtained models within the top 5 submitted models. Target T125 contained 4 different interfaces (LLT1:NKR-P1 / LLT1:LLT1 / LLT1:NKR-P1 / NKR-P1:NKR-P1), where the heteromeric and homomeric interfaces were evaluated independently, as two different targets. In this target we obtained high quality models in all categories for the LLT1:LLT1 interface, but in reality this model was built using standard template-based modeling (PDB 4QKH, 100% SI). As scorers, we could also identify a medium quality model for the LLT1:NKR-P1 interface. The target T136 is a clear example of a successful application of combining *ab initio* docking and template-based docking (see 5.5.4). We obtained acceptable results for the protein-peptide targets T134-T135; restraints were also used for these targets (see 5.5.5 and 5.5.6). In modeling protein-oligosaccharide complexes targets T126-T130, we obtained acceptable results for all targets, and a medium-quality model for target T126 defining the center of the homologous protein cavity for docking was crucial to obtain successful predictions (see 5.5.5). For the T133 protein complex, although PDB 3U43 (CAPRI target T47) was a good template, we did not use it to model the protein-protein complex. Interestingly, we found acceptable (as servers) and medium (as predictors and scorers) models, all of which was due to the use of the new strategy using IRaPPA (see 5.5.6).

Overall, as a scoring tool, we can say that the pyDock energy function has been able to obtain successful predictions for 13 of the 19 displayed targets (most of them of medium quality). Moreover, in the protein-saccharide scoring experiments, models were all selected according to the newly adapted version of pyDock. For targets T122, T125, T131, and T132, despite no successful result was obtained as predictors, the scoring function was able to find medium-quality models in the scoring experiment. The references (experimentally determined models) for almost all targets are listed in Appendix 2: Table 10.2.5

6.5.1.2. Unsuccessful predictions

The targets T122-T124, T131, T132, T125 (hetero-complex) and T121 were really challenging, and indeed for some of them (T123 and T124) there were no successful predictions from any participating group, and very few groups were successful for the other ones.

We believe that the difficulties in targets T122-T124 are not related to the fact that they are interactions with nanobodies, but to the problems in modeling IL-23R (T122) and PorM (T123, T124) structures, respectively. On the other side, the strategy we followed in target T125 failed to capture the two different LLT1:NKR-P1 interfaces existing in the complex, since we assumed that there was only one interface. For targets T131 and T132, the distance restraints, based on the hCEACAM1 interface (Tyr35 and Ile92) [199], were incorrectly assumed. Furthermore, in the case of target T121 (protein-peptide), models were selected by docking with the β-strands of the targeting peptide and protein in an antiparallel fashion, which turned out to be a wrong decision.

6.5.2. 3rd joint CASP-CAPRI experiment

The 3rd joint CASP-CAPRI experiment (CASP13-CAPRI46) proposed a total of 20 targets, including 14 homocomplexes and 6 heterocomplexes, classified as 14 homodimers, 5 multimeric complexes, and one heterodimer. In terms of difficulty, there were nine easy and 13 difficult targets (T149-T151 related to the same oligomer) (Appendix 2: Table 10.2.9). Considering the top 10 models submitted by our group, we submitted acceptable models for 13 targets as predictors and 12 for scorers (Table 6.4). This represents a success rate of 65% (predictors) and 63% (scorers) with respect to all targets in which we participated (Table 6.4). This performance was the second best of all participants in predictors and the best one in scorers [142, 144].

6.5.2.1. Successful predictions

The participation of our group in the 3rd joint CASP-CAPRI experiment was very satisfactory, as we obtained the highest number of successfully predicted targets [144]. All the easy targets were successfully predicted, except T147, where we could not predict all interfaces. In the difficult targets, we were able to predict some of the interfaces. For example, we were the only group that successfully predicted the T154 heterodimer and one of the few that submitted acceptable templates for the T157 case in predictors (Table 6.4).

We used templates in 70 % of the targets, which was generally successful, as the pyDock scoring function identified the correct models in most of them (T139, T140, T142,

T143, T144, T152, T153, T158) (Table 5.6.2), while it was unsuccessful in only three cases (T137, T138 and T154).

Our *ab initio* docking protocol pyDock, was successful as predictors in CAPRI (considering 10 submitted models, instead of the 5 submitted models in CASP) in two difficult cases: T154, where we obtained two acceptable models (no other group was successful), and T157 (Table 5.6.2). Furthermore, it was also successful in T153 where both strategies (template-based docking and *ab initio* docking) showed good results.

Table 6.4 Quality of submitted models for the joint CASP13-CAPRI46 experiment

Easy Targets	Stoich.	Submission quality for Predictors	Successful groups[2]	Submission quality for Scorers	Successful groups
T140	A2	**	22/30	**	17/18
T143	A2	***	25/29	**	16/18
T144	A2	**	27/29	**	17/18
T152	A2	***	26/31	***	18/18
T153	A2	***	28/33	***	18/18
T147	A2/A4/A8	**/-/-	19/17/16 of 26	**/**/***	15/14/12 of 16
T158	A3	**	18/25	**	16/16
T139	A4	***/***	27/26 of 28	***/***	17/16 of 17
T142	A1B1	**	12/30	**	12/18
Medium-Diff. Targets	**Stoich.**	**Submission quality for Predictors**	**Successful groups**	**Submission quality for Scorers**	**Successful groups**
T137	A2	-/-	1/0 of 28	-	cancelled
T138	A2	-/-	0/1 of 28	-/-	0/17
T141	A2	-	7/29	*	9/18
T148	A2	-	0/32	-	0/17
T149	A2	*/-/-/-/-	12/3/4/0/2 of 21		
T150_SAXS	A2	**/-/**/-/-	9/3/3/0/0 of 21		
T151_XL	A2	***/-/-/-/-	10/1/2/0/0 of 21	***/-/-/-/-	16/1/2/0/0 of 16
T154	A2	*	1/30	-	1/18
T155	A1B1	-	1/30	-	0/17
T156	A1B1	-	3/30	-	1/17
T157	A1B1	*	5/29	-	1/17
T146	A2B2	-/-/-	7/0/2 of 29	-/-/-	7/0/4 of 18
T159	A6B6C6	**/-/-/-/-/-	18/0/0/0/0/9/8 of 22	**/-/-/-/-/**/-	15/0/0/0/0/14/12 of 15

[1]***=high-quality models; **=medium-quality models; *=acceptable-quality models for the results in the performance [170]. [2]In general, docking servers are included in predictors groups.

For cases involving more than three protein-protein interfaces, including inter-domain interaction (T149-T151), we used the combined sampling method (Chapter 6.4.4), selection of models based on experimental data (Chapter 6.4.5), and final scoring by our pyDock energy function. For targets T149-151, which actually refer to the same protein complex but with different experimental information (SAXS spectrum in T150; cross-linking data in T149), we obtained models for two of the five proposed interfaces, using the strategy explained in Chapter 6.4.5. For target T147 (A8), the homo-dimeric interfaces were successfully predicted by the template-based models but not by *ab initio* docking. In T159, only the template-based docking-based strategy was successful (see Chapter 6.4.4), but only for one specific interface.

Our performance in scorers was the best amongst all participants. We predicted medium or better-quality models for all the easy targets. In the difficult complexes, we predicted acceptable models for T141, and high-quality models for the homodimeric interface of T149-T151. A special case was the T159 target, where we submitted medium quality models for two interfaces.

6.5.2.2. Unsuccessful predictions

Most unsuccessful predictions corresponded with targets considered difficult by the organization. However, in T147, an easy target, not all interfaces could be predicted (Table 6.4). The problem was that we did not correctly identify the helical rise value for octahedral fiber formation. In targets T137, T138, T146, T155 and T156, we did not obtain successful predictions. Indeed, only a few groups were able to obtain successful predictions for these targets (Table 6.4).

For targets involving more than three protein-protein interfaces, i.e. T149-T151, T159, and T147 (above mentioned), not all interfaces could be modeled. On the other hand, for the target T146 (A2:B2), using template-based restraints and SAXS spectroscopy was unsuccessful, most likely due to significant conformational rearrangements. Finally, for T141 we just could not send the models to the predictor experiment due to agenda problems.

6.5.3. 4th joint CASP-CAPRI experiment

We have participated as human predictors, human scorers, and server scorers, in all the 12 evaluated targets, comprising a total of 23 assessed interfaces (T177 have 3, T170 have 9 and T180 have 2 interfaces) (Appendix 2: Table 10.2.9). We applied a similar strategy to that in the CASP13-CAPRI experiment, combining *ab initio* docking, template-based modeling, and energy-based scoring [144]. For the easy targets T166, T168 and T177, we achieved models with at least acceptable quality for predictors and scorers. We failed in target T164 as predictors but were successful as scorers. In the case of the difficult targets, we were successful in targets T178, T170 and T180 as predictors, and in targets T179, T170 and T180 as scorers. In summary, we obtained a success rate of 30%, 61% and 57% as predictors, scorers (humans) and scorers (servers), respectively, considering the total number of interfaces, and 50%, 58% and 58%, respectively, taking into account the number of cases.

6.5.3.1. Successful predictions

The results in the cases classified as easy, T166 and T168, were favoured by the existence of good templates, obtaining models of medium quality as predictors, scores (human) and scores (server). In the case of T164 only acceptable models were obtained in scores (human and server), since the *ab initio* docking strategy followed in this target as predictors was unsuccessful. In target T177, which was an eicosameric protein, consisting of two decamer rings that we modelled by using MODELLERv9.19 [205] based on available templates, followed by pyDock scoring, and building the final assembly of the two decamers by *ab initio* docking. We obtained medium quality predictions (averaged over the 3 interfaces) for this target as predictors, human and server scores.

Regarding the difficult cases, we obtained acceptable results for target T178, but only as predictors, and for target T179, for both human and server scorers. For target T180, a homo-240-mer, we modelled the minimum number of subunits needed to define unique interfaces by *ab initio* docking and obtained acceptable results (averaged over the two interfaces evaluated) as predictors, human scorers and server scorers. As for target T170 we

Table 6.5 Quality of submitted predictions for the CASP14-CAPRI50 experiment predictions.

Easy Targets	Stoich.	Submission quality for Predictors	Successful groups[2]	Submission quality for Scorers (human)	Submission quality for Scorers (servers)	Successful groups for Scorers
T164	A2	-	19/28	*	*	17/23
T166	A1B1	**	17/24	**	**	14/19
T168	A3	**	18/24	**	**	17/20
T177	A20	***/***/-	21/22/14 of 24	**/***/**	***/***/-	17/18/14 of 18
Medium-Diff. Targets	**Stoich.**	**Submission quality for Predictors**	**Successful groups**	**Submission quality for Scorers (human)**	**Submission quality for Scorers (servers)**	**Successful groups for Scorers**
T169	A2	-	0/27	-	-	0/20
T176	A2	-	3/27	-	-	9/21
T178	A2	*	13/26	-	-	10/19
T179	A2	-	10/25	*	*	17/19
T165	A3H3L3	-	0/27	-	-	0/22
T174	A3	-	0/24	-	-	0/19
T170	A6/B3/C12/D6	*/*/-/-/-/-/-/-/-	13/1/5/4/12/1/1/7/6 of 23	**/-/-/*/**/-/-/*/*	**/-/-/*/**/-/-/*/*	17/0/10/9/16/2/2/13/13 of 17
T180	A240	-/**	1/19 of 25	*/**	*/*	4/17 of 18

[1]***=high-quality models; **=medium-quality models; *=acceptable-quality models for the results in the performance [170]. [2]In general, docking servers and CASP14-CAPRI predictors are included. For T167, T175 and T181, there are no evaluated data provided by CAPRI. The 3 targets of T171-173 are cancelled by CASP.

applied an *ad-hoc* modeling procedure (see Chapter 6.4.4), also combining *ab initio* docking and template-based modeling. We obtained two acceptable models for interfaces #1 (A:B) and #2 (A:E), where we were the only group to submit an acceptable model. However, in the averaged evaluation for interfaces #8 (D) and #9 (C_iD), we obtained acceptable results as human and server scorers (along with Venclovas group, these were the only acceptable rank one submissions from all participants).

6.5.3.2. Unsuccessful predictions

This 4th joint CASP-CAPRI protein assembly prediction challenge has proven to be more complex than the previous one, with higher proportion of difficult cases. For targets T169, T165, and T174, no group managed to generate models of acceptable quality. For targets T164, T169, T176 and T174, the preferred strategy in predictors and scorers was *ab initio*

docking, which proved to be unsatisfactory. This was also the case for target T164, where we used a combination of template-based and *ab initio* docking. Interestingly, we were successful as scorers in both T164 and T179 targets, where the lack of templates was a determinant factor for predictors but not for scorers.

6.6. Protein docking functions for the identification of physiological homodimers

In this section, we have evaluated the application of docking and scoring functions for the prediction of homo-dimer assemblies in other challenging scenarios. As mentioned in the introduction to this chapter, structural data on protein-protein interactions are crucial to elucidate their mechanism of action at the molecular level. The formation of these macromolecular assemblies depends on protein concentrations and physicochemical conditions such as pH and ionic strength [206]. But both the protein concentrations and the experimental conditions used during the use of such methods often differ from the physiological conditions under which the interactions and/or formation of macromolecular complexes occur. Therefore, in some cases, when experimental parameters differ sufficiently, the 3D structures determined may result in assemblies that do not represent physiological reality. Non-physiological assemblies can also be formed when the structure under study represents a part of a larger complex and has been reconstituted in the absence of additional components.

This problem is severe in the case of X-ray diffraction crystallography, by which 86% of the protein structures in PDB are determined (data from October 2022). When proteins form a crystal, spurious contacts between proteins can occur only to stabilize the crystal lattice, but might not be relevant in solution. In some proteins, especially in apparently homo-dimeric crystals, identifying which of these contacts are physiologically relevant is problematic, which makes it difficult to assign the true oligomeric state, i.e., whether it is a monomer, a homodimer, or a higher order assembly. The authors generally provide these data during the PDB deposition process but remain prone to errors, as they require independent biophysical/biochemical characterisation, which can give ambiguous results.

Moreover, there is a significant fraction (18%) of protein assemblies resolved by X-ray crystallography on the PDB for which no associated publication exists.

Several computational methods have been developed to infer the oligomeric state of proteins directly from the contacts made by proteins in the resolved crystal lattice structure. These methods make use of the results of a large number of previous studies, in which the properties of the interfaces of native protein complexes were systematically evaluated and compared with those of the crystal contacts, considered to represent weak non-specific interactions [207]. The most used methods are PISA [194], EPPIC [208] and PRODIGY-crystal [209] .

PISA evaluates the chemical and structural properties of the interfaces, while EPPIC uses geometric measurements and sequence entropy of homolog sequences, which is often associated with regions involved in biological function [210].

To address the problem, we have participated in an activity proposed by the ELIXIR 3D-BioInfo Community, which provided a benchmark composed of 1677 homo-dimeric complex structures, of which 841 are non-physiological and 863 are physiological (so-called "dimer benchmark version 3"; for more details see Chapter 3.5.6). We have evaluated the capabilities of our pyDock scoring functions, as well as other descriptors in CCharPPI (Computational Characterisation of Protein-Protein Interactions) server, regarding the identification of physiological homo-dimers in this benchmark.

6.6.1. Applicability of pyDock and CCharPPI scoring functions to the identification of physiological homodimers.

We evaluated pyDock scoring [79], as well as each of its individual energetics terms, electrostatic, desolvation, and van der Waals, together with 88 descriptors in CCharPPI [78] (https://life.bsc.es/pid/ccharppi/), by applying them to the proposed cases in the dimer benchmark version 3 developed in collaboration with the ELIXIR 3D-BioInfo Community.

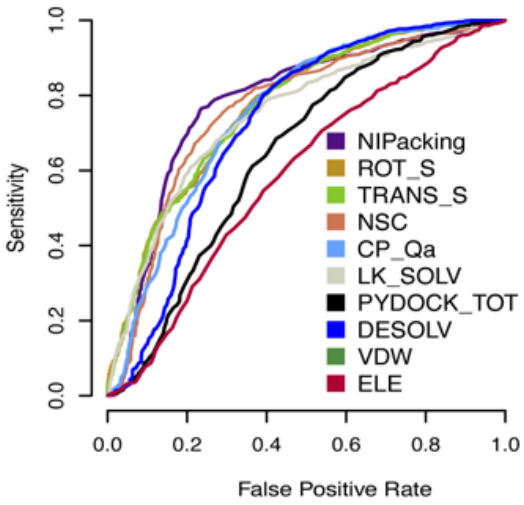


Figure 6.6. ROC curves of the best descriptors from CCharPII

For each of these descriptors, we evalueted whether they were able to identify the correct physiological dimers, and thus calculated different predictive success metrics, such as sensitivity or coverage (TPR), precision or positive predictive value (PPV), accuracy (ACC) and Matthews correlation coefficient (MCC).

$$Precision\ or\ Positive\ Predicted\ Value\ (PPV) = \frac{TP}{TP + FP} \tag{6.6}$$

$$Negative\ Predicted\ Value\ (NPV) = \frac{TN}{TN + FN} \tag{6.7}$$

$$Sensitivity\ or\ True\ Positive\ Rate\ (TPR) = \frac{TP}{TP + FN} \tag{6.8}$$

$$Accurancy\ (ACC) = \frac{TP + TN}{TP + FN + TN + FP} \tag{6.9}$$

$$\text{Matthews}\ (MCC) = \frac{TP * TN + FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{6.10}$$

In Figure 6.6 a ROC curves comparainsion between the selected descriptors (acording the AUC (Area under the ROC curve)) of the four best CCharPII descriptors and the pyDock energetic terms,) are shown. In addition, an analysis comparing sensitivity and safety is shown in Figure 6.7. As can be seen, the best descriptors extracted from CCharPII are NIPacking described in [211], change in rotational entropy upon complexation (ROT_S), change in translational entropy upon complexation (TRANS_S) both calculated as in

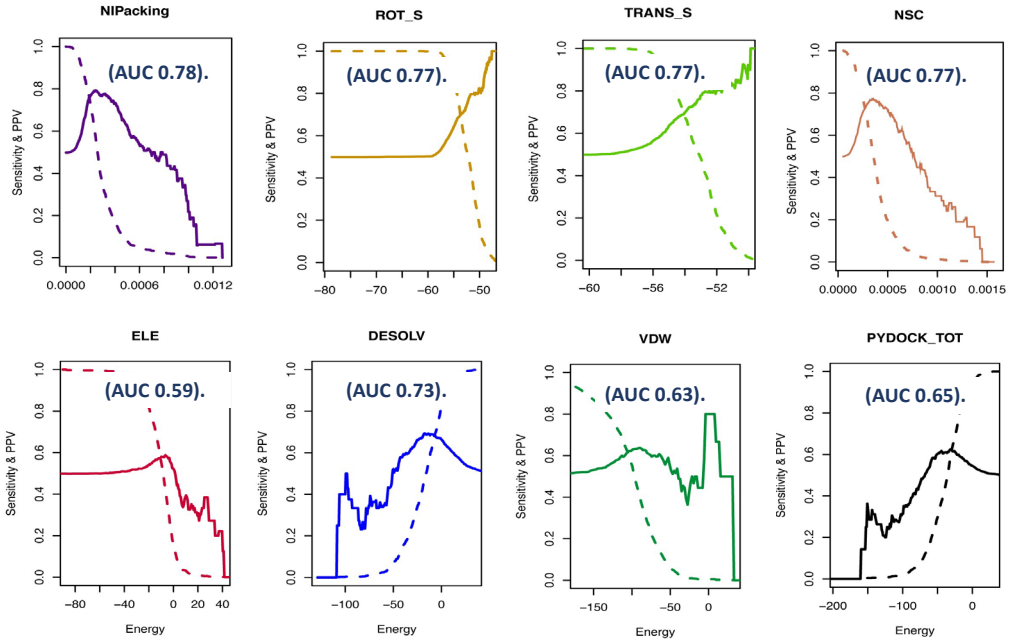


Figure 6.7. Shows the sensitivity (dashed line(s)) and accuracy (solid line(s)) of the four best CCharPII descriptors, the pyDock energetic terms and pyDock scoring function.

CHARMM [212] and the surface complementarity score (NSC) described in [211]. In the case of the pyDock energetic terms, the best descriptor was desolvation energy term.

These descriptors are complementary with other descriptors from different groups also participating in this activity of the 3D-BioInfo community. When all descriptors are aggregated using machine learning methods, such as random forests, they yield a consensus scoring function that obtains a higher discriminatory power: 0.94 AUC (Area Under the Curve).

7. APPLICATION TO A CASE STUDIO: MODELING ELECTRON TRANSFER PROTEIN COMPLEXES.

The results described here have been reported in this publication: Castell, C., et al., *New Insights into the Evolution of the Electron Transfer from Cytochrome f to Photosystem I in the Green and Red Branches of Photosynthetic Eukaryotes.* Plant Cell Physiol, 2021. **62**(7): p. 1082-1093

7.1. Introduction

In this chapter of the thesis, we have applied some of the developed docking tools to model protein complexes involved in electron transfer in photosynthesis. Photosynthesis is essential for capturing and storing solar energy in the biosphere. Thanks to this process, plants absorb millions of tonnes of CO2 per year on a net basis. At the molecular level, the efficiency of this process lies in a series of highly optimized multi-protein complexes for electron transfer, such as Photosystem I, one of the most efficient photoelectric systems in nature, which can convert solar energy into chemical energy at almost 100% efficiency. The basic mechanisms and components of photosynthesis have been conserved throughout evolution from cyanobacteria to higher plants, although there are interesting differences.

In photosynthetic organisms, two proteins act as electron transporters from cytochrome f to photosystem I: plastocyanin (Pc) (containing copper) and cytochrome c_6 (Cc_6) (containing iron). The two proteins are very different in composition but have equivalent structural and functional aspects. By analysing the available three-dimensional structures and applying some of the computational modelling methods developed and/or optimize in this thesis, it has been possible to identify important details of the molecular mechanisms of photosynthesis in different organisms.

In cyanobacteria and many green algae, the two proteins can be used alternatively, depending on the environmental conditions. However, higher plants (green lineage) have only Plastocyanin, which forms strong and efficient complexes for electron transfer. On the other hand, species of the red lineage, such as red algae and diatoms, have only cytochrome c_6, which forms weaker and less efficient complexes for electron transfer. Interestingly, plastocyanin genes have been found in oceanic diatoms. In fact, in the case of *Thalassiosira oceanica*, it possesses such genes and weakly expressing CC_6, reducing its dependence on iron. But at the expense of generating less efficient electron transfer complexes.

7.2. Molecular structures and modelling

Modelling of the proteins in this study was carried out with MODELLER version 9v23 [205]; https://salilab.org/modeller/, using the default settings [213]. Template searching and sequence alignments were carried out with BLAST [187] and ClustalW [214].

The truncated Cf forms of *Phaeodactylum tricornutum* (UniProtKB: A0T0C9 CYF_PHATC) and *Thalassiosira oceanica* (UniProtKB: E7BWE1_THAOC), were modelled using as template a truncated Cf form that was found in *Chlamydomonas reinhardtii* (PDB ID 1CFM) with which they share 58.2% and 44.6% sequence identity, respectively. The coordinates of the cofactors (heme and Cu ion) were taken from the template structures. The structure of the small domain deletion variant of *P. tricornutum* Cf was generated by removing residues 171-229 from the intact protein structure with ICM Browser software (http://www.molsoft.com) [215]. The *Thalassiosira oceanica* plastocyanin model was obtained using *Chlamydomonas reinhardtii* plastocyanin as a template (PDB code 2plt). For each modelled protein, 100 models were built and the models with the best DOPE (Discrete Optimized Protein Energy) score were selected [188], as previously described [216]. The 3D structure of *P. tricornutum* Cc_6 was obtained from the Protein Data Bank, PDB code: 3DMI [217]. The representation of the electrostatic surface potentials, shown in Figures 7.1 and 7.2, was generated with the UCSF Chimera program. The Cu atom in Pc was assigned a charge of +2, while the heme atomic charges for both Cf and Cc_6 (-2) were distributed as: Fe (+2), two nitrogen atoms of the ring (-1 each), and the two propionic acid side chains (-1 each) [218].

7.3. Protein-protein docking simulations

7.3.1. Docking sampling and scoring

The protein-protein docking simulations were performed by pyDock scheme [79, 82], adapted here for the inclusion of cofactors during docking calculations. The setup step (see Chapter 3.1.4) needs the coordinates of the two interaction proteins, usually as PDB files, but it can also take AMBER coordinates and topology files. In the original implementation of this functionality it was possible to include modified amino acids as part of the interacting molecules, but now we needed to modify the automatic protocol to incorporate Cu and the heme group as separated molecules but still part of the receptor or ligand.

Below is a detailed description of the steps needed to perform *ab into* docking on the models of Cytochrome f and Plastocyanin of *Thalassiosira oceanica,* using pyDock:

1) Before going through the steps to generate the docking models, it is necessary to obtain the coordinate and topology files from AMBER. We will first generate the necessary additional libraries and parameters to do so.

   a) The heme atomic coordinates (mol2 file) were downloaded from the supplementary data of [219] and saved as a PDB file with the UCSF Chimera program. The PDB (Hemo_cit_f_H.pdb) was edited by removing the iron atom and the associated connectivity, and the residue name field was changed from HEM to HEC. And finally, the following commands were run to generate the heme library *antechamber* and *tleap* tools:

```
> conda activate #Anaconda was installed to facilitate the use of AMBER programs.
> pdb4amber -i Hemo_cit_f_H.pdb -o Hemo_cit_f_H_renumb.pdb #Renumber and check connectivity.
> antechamber -fi pdb -fo mol2 -i Hemo_cit_f_H_renumb.pdb -o HEC.mol2 -c bcc -pf y -nc -4 #Calculation of molecule charges.
> parmchk2 -i HEC.mol2 -o HEC.frcmod -f mol2
#Generation of the amber parameters
> tleap -f Heme_lib.leap #Generation of the heme unit library.
> tleap -f Fe_lib.leap #Generation of the Fe unit library.
```

```
#Tleap script to create the
heme group library.
tleap
source leaprc.gaff
HEC = loadmol2 HEC.mol2
loadamberparams HEC.frcmod
check HEC
saveoff HEC HEC.lib
```

```
#Tleap script to create the Fe
group library.
source leaprc.gaff
FE = loadmol2 FE.mol2
loadamberparams frcmod.hemall
check FE
saveoff FE FE FE.lib
```

Figure 7.1. Library scripts. The right panel shows the script Heme_lib.leap, used to obtain the library from the mol2 and frcmod files, obtained from running antechamber and parmchk2. The left panel shows a similar script loading the iron data, obtained from the advanced amber tutorial.

   b) The mass and iron parameters (*frcmod.hemall* file) were extracted from the AMBER parameter database (https://personalpages.manchester.ac.uk/staff/Richard.Bryce/amber/cof/frcmod.hemall), and the mol2 file was obtained from an amber tutorial

(https://ambermd.org/tutorials/advanced/tutorial20/files/mcpbpy_heme/FE.mol2). The iron charge was set to 2.0 in the FE.mol2 file.

c) In the case of Plastocyanin, it was not necessary to create the amber library and parameter files as they were extracted from the advanced amber tutorials: *plc.frcmod* from https://ambermd.org/tutorials/advanced/tutorial1_orig/files/plc.frcmod and *cua.lib* from https://ambermd.org/tutorials/advanced/tutorial1_orig/files/cua.lib.

d) With this last command we generate the AMBER coordinates and topology files.

```
> tleap -I $HOME/anaconda*/dat/leap/cmd/oldff -f Docking.leap
```

```
#Generate topology and coordinates files: Docking.leap
source leaprc.gaff
source leaprc.ff99SB
loadamberparams HEM.frcmod
loadoff HEM.lib
loadamberparams frcmod.hemall
loadoff FE.lib
loadamberparams plc.frcmod
loadoff cua.lib
target = loadpdb ModeloCfThalassiosira.pdb
saveamberparm target ModeloCfThalassiosira.prmtop
ModeloCfThalassiosira.inpcrd
target2 = loadpdb ModelPcThalassiosira.pdb
saveamberparm target2 ModeloPcThalassiosira.prmtop
ModeloPcThalassiosira.inpcrd
```

Figure 7.2. Docking.leap script**.** The parameters and libraries of the heme group, the iron of cytochrome f, as well as the copper of plastocyanin are loaded and the coordinate and parameters files are obtained.

e) In addition, the solvation reference areas of the heme group atoms were calculated using ICM browser (http://www.molsoft.com) (ICM script in Figure 7.3), and the

```
#HEC_Res_area.icm
read pdb "Hemo_cit_f_H_renumb.pdb "
show surface area mute
atoms =a_1.//
n = Nof(atoms)
add column t Sarray(n,
Name(Obj(atoms))[1]),Trim(Label(atoms),all),Area(atoms)
write t separator="," "HEC_Res_area.csv"
```

Figure 7.3. HEC_Res_area.icm script.

areas of the atoms were mapped to corresponding Amber types (see Appendix 3: Table 10.3.1). This information was added to the pyDock *res.dat* file, where the information about the amino acid areas is also included.

2) Set up the receptor and ligand for docking.

Before any docking calculation, we need to generate the coordinate files correctly

```
# Thalassiosira.ini
[receptor]
amber = ModeloCfThalassiosira.inpcrd,ModeloCfThalassiosira.prmtop
mol =
newmol = A

[ligand]
amber = ModeloPcThalassiosira.inpcrd,ModeloPcThalassiosira.prmtop
mol =
newmol = B
```

Figure 7.4. PyDock INI configuration file, Thalassiosira.ini.

parsed for pyDock. We created an INI configuration file (https://en.wikipedia.org/wiki/INI_file), including the coordinates and parameter files created earlier (see Figure 7.4) Then we can run the pyDock setup, executing the following command in the console:

```
> pydock4 Thalassiosira setup
```

This command will create the new files for the receptor: *Thalassiosira_rec.pdb* *Thalassiosira_rec.pdb.H* and *Thalassiosira_rec.pdb.amber*. And new files for the ligand *Thalassiosira_lig.pdb*, *Thalassiosira_lig.pdb.H* and *Thalassiosira_lig.pdb.amber*, which are the files pyDock needs for execution.

3) Generating rigid-body docking

pyDock is ready to run and automatically process the FTDock 2.0 docking program output. For which it must be previously installed, and its location indicated in the pydock.conf (see Chapter 3.1.3).

```
> pydock4 Thalassiosira ftdock
```

The FTDOCK parameters can be modified by changing the pydock.conf configuration file, including the possibility of enabling parallel execution.

4) Converting the rigid-body docking poses to pyDock Format

Now, we need to transform the output data from FTDock (*Thalassiosira.ftdock* in our example, in which each solution is represented by the position of the ligand in Cartesian coordinates, and its rotation based on Euler angles) to the rotation and translation matrix that transforms the original ligand coordinates into the different orientations generated by FTDock, using the following command:

```
> pydock4 Thalassiosira rotftdock
```

5) Scoring the Rigid-Body Docking Poses with pyDock

The next step is to use the pyDock energy function to score and rank all positions by running the *dockser* module with the following command:

```
> pydock4 Thalassiosira dockser
```

Thanks to the files generated in the pyDock setup and the transformation of the FTDOCK 2.0 data mentioned above, Table 7.1 of energies is generated: *Thalassiosira.ene*.

Table 7.1 *Thalassiosira.ene*

Conf[1]	Ele[2]	Desolv[3]	VDW[4]	Total[5]	RANK[6]
1205	-30.221	-3.238	-4.739	-33.933	1
5759	-36.059	2.491	-0.006	-33.568	2
5023	-30.522	-3.588	8.68	-33.242	3
1508	-38.239	6.805	-6.838	-32.117	4
5060	-28.692	-3.03	-2.369	-31.958	5
2994	-29.711	-3.741	16.667	-31.785	6
2452	-31.816	-0.352	8.15	-31.353	7
511	-33.246	3.636	-13.647	-30.975	8
4480	-44.473	11.928	18.707	-30.675	9
6622	-34.556	2.178	19.062	-30.472	10

[1]Conformation number of the docking pose. [2]Electrostatic energy term. [3]Desolvation energy term.[4]Van der Waals energy term. [5]Total binding energy (Ele + Desolv +0.1∗VDW).[6]Rank of the docking pose according to its total binding energy.

6) Distance calculation

In addition to the use of pyDock energies, the distances between the Fe atoms of the two cofactors were used to select the best models. For this purpose, the *Distance* command of ICM Browser software was used. The distance calculated was the minimum distance from the Fe^{+2} or Cu^{+2} atoms to all the atoms of the pyrrole ring of the corresponding heme groups.

7.3.2. Minimization of the protein-protein docking poses

In order to improve the quality of the docking models and reduce their interatomic clashes, they were minimized with the Sander program from AMBER [109], using AMBER ff99SB and GAFF force field parameters [202], with implicit solvent. The protocol consisted of a 300-cycle steepest descent (SD) minimization, with harmonic restrains applying a force constant of 1000 kcal (mol $Å^2$)$^{-1}$ to the heme group and the coordinated atoms (Fe, Cu) and residues (Tyr, His, and Cys) to optimize the coordination states and the side chain conformations, followed by a 500-cycle conjugate gradient (CG) minimization with the same harmonic restraints.

7.4. Docking structures of [Cf:Cc$_6$] and [Cf:Pc] in red Algae lineage

For the photosynthetic organisms of red lineage, no structural data are available for the interaction between cytochrome f and the soluble electron carriers plastocyanin [Cf:Pc] and cytochrome c_6 [Cf:Cc$_6$]. In order to analyse the differences between the lineages, we performed docking simulations to study the [Cf:Cc$_6$] complex in *P. tricornutum* and *T. oceanica*. And in the case of *T. oceanica*, docking models were also performed with the acquired Plastocyanin that forms the [Cf:Pc] complex, so being able to compare the complexes with each other. The modelling and docking protocol was similar to that explained in Chapter 7.2 and Chapter 7.3 respectively. As a control, we also generated docking simulations for the [Cf:Cc$_6$] complex of the green alga *C. reinhardtii* that was previously studied by Brownian dynamics (DB). The models obtained in this work are in agreement with the results previously described [218, 220], presenting an orientation that converges towards an orientation with the smallest distance between cofactors (Figure 7.5A).

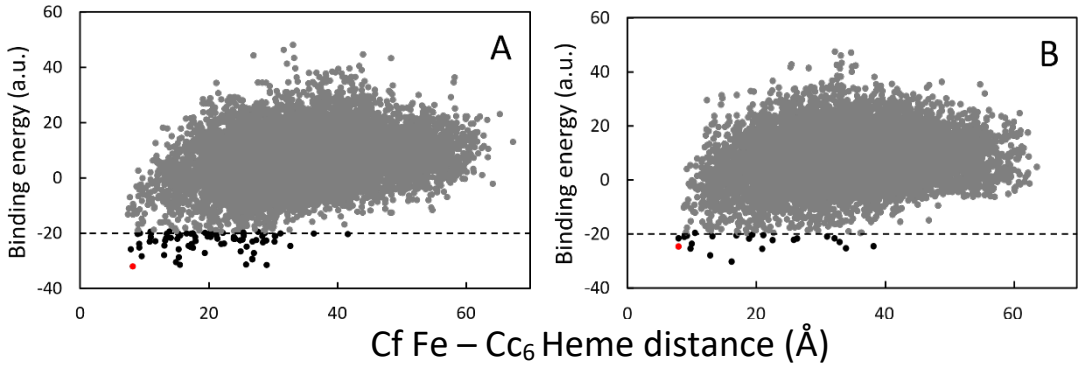


Figure 7.5. Landscapes for the computational docking results of the [Cf:Cc6] complex of (A) C. reinhardtii and (B) P. tricornutum. The distances between the iron atom of Cf and the heme group of Cc6 were considered. The lowest-energy and best distance docking orientations, showed in the next Figure, are highlighted in red.

The same energy-distance landscape can also be observed for the docking models of P. tricornutum [Cf:Cc_6] (Figure 7.5B). However, the orientation of cytochrome c_6 is different from that obtained in *C. reinhardtii* (Figure 7.6B, 7.6C). These orientations have smaller interfaces and weaker electrostatic interactions. The hydrophobic interactions gain more weight, being similar to what has been previously described in cyanobacteria [221, 222]. Comparing the best energies-distance docking models for the [Cf:Cc_6] complex, we can claim that the model for *C. reinhardtii* has better affinity (-32 a.u. versus -24.7 a.u.) than the *P. tricornutum*

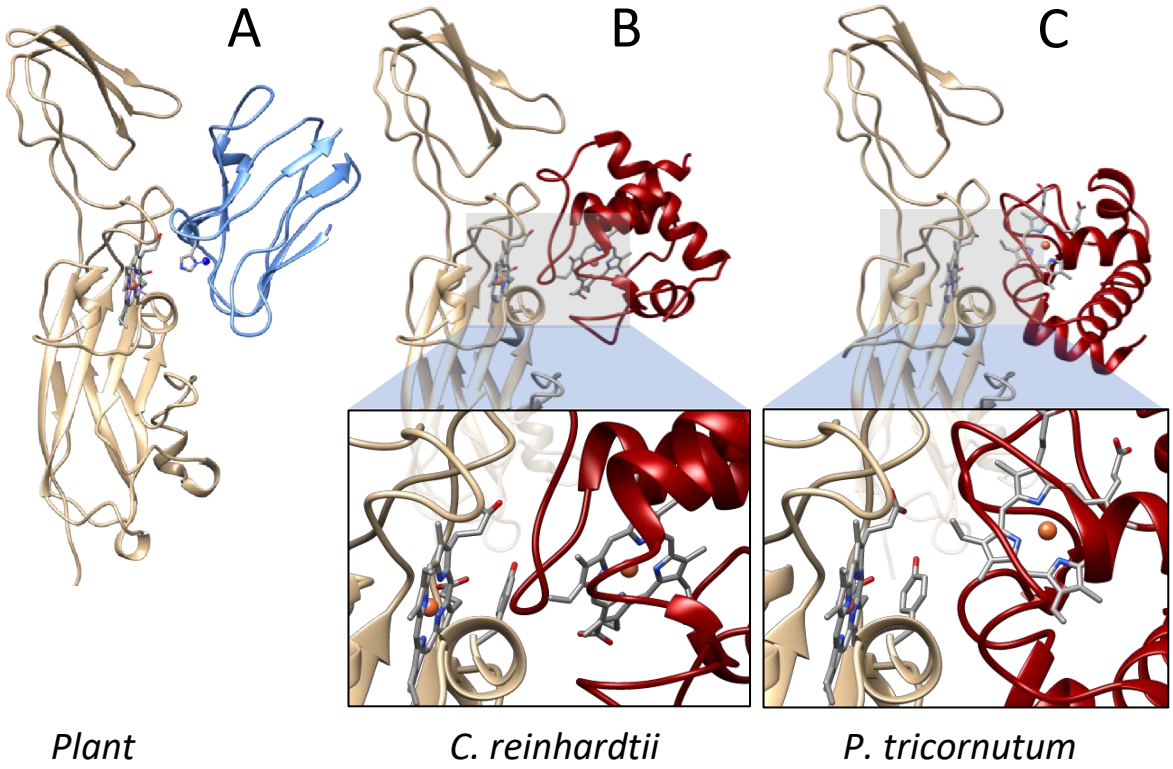


Figure 7.6. Representative structures for the [Cf:Pc] complex of plants and best-energy docking models for the [Cf:Cc6] complexes of *C. reinhardtii* and *P. tricornutum*. (A) Representative structure for the plant [Cf:Pc] complex (turnip Cf and spinach Pc; PDB code, 2pcf) (Ubbink et al., 1998). Pc is coloured in blue and the copper-bound His87 is shown. (B, C) Best-energy docking models for efficient ET between Cf (in light brown) and Cc6 (in red) of the green alga C. reinhardtii (rank 1, docking energy –32.0 a.u., distance between Fe in Cf to heme in Cc6 of 8.2 Å) and P. tricornutum (rank 6, docking energy –24.7 a.u., distance between Fe in Cf to heme in Cc6 of 8.0 Å, the shortest distance model).

As in previous studies, a version of Cf with the small domain deletion was used for docking. Surprisingly, for P. tricornutum, the small domain appears to play only a minor role in the interaction with Cc_6 (Figure 7.7). This is in contrast with docking simulation in *C.*

reinhardtii, where deletion of the small domain of Cf affected the Cf binding positions of both Pc and Cc6 (Haddadian and Gross, 2006).

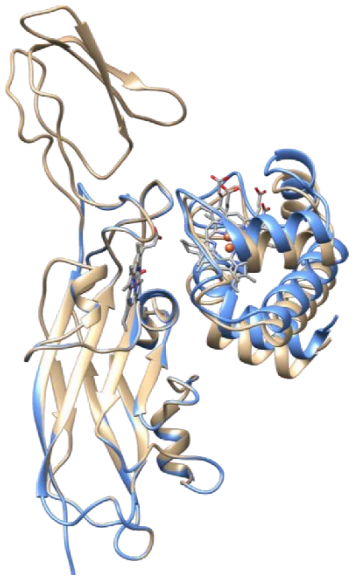


Figure 7.7. Superimposed docking models of *P. tricornutum*. Superimposed docking models of the *P. tricornutum* [Cf:Cc6] complex and the model (in blue) corresponding to a truncated Cf without the small domain (rank 2, docking energy –31.0 a.u., distance between Fe in Cf to heme in Cc6 of 8.4 Å).

Finally, we compared models of the [Cf:Pc] complex from *T. oceanica* with the previously described [Cf:Cc6] complex from *P. tricorn7.5.utum*, as well as with the equivalent [Cf:Pc] complex from *C. reinhardtii* in the green lineage. The docking between Cf and Pc from *T. oceanica* generated a large number of low-energy orientations compared to the [Cf:Cc6] complex from *P. tricornutum*. This seems to indicate a higher binding affinity, possibly due to the higher electrostatics of the acquired "green-type" Plastocyanin. But the energy-distance landscape does not converge towards a single orientation, resulting in different orientations in a similar range of distances and energies. They can coexist and be biologically functional. These possible orientations are

(i) "Head-on" Pc orientation, which is relatively similar to that of some complexes observed in cyanobacteria. Hydrophobic interactions are shown to predominate, and there is no interaction between the electrostatic zones and the small Cf domain (energy of -24.2 a.u. and the shortest distance between Fe and Cu of 11.1 Å) (Figure 7.9A);

(ii) "Side-on" Pc orientation, similar to that of the green lineage complexes (figures 7.6A and 7.9A), which includes the electrostatic and hydrophobic patches of both proteins and the small Cf domain (energy of -31.1 a.u.; Fe-Cu distance of 12.7 Å)

(iii) "Intermediate" Pc arrangement, which includes the hydrophobic patches and some residues of the electrostatic patches (energy of -30.1 a.u.; Fe-Cu distance of 12.2 Å) (Figures 7.9C).

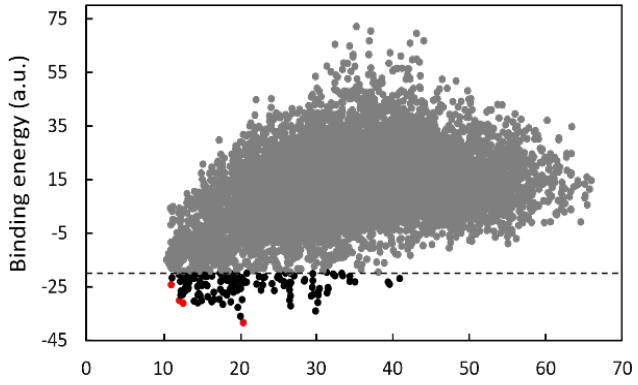


Figure 7.8. Landscape for the computational docking results of the [Cf:Pc] complex of T. oceanica. The docking orientations showed in the next Figure are highlighted in red. The distances between the iron in Cf and the copper in Pc were considered

But let's look for models with the lowest energy, regardless of distances. We find a population of models with even more favourable energies (-38 to -36 a.u.), in which strong electrostatic interactions are established, also involving additional positive groups outside the usual electron transfer region in Cf. However, the Fe-Cu distances are about 20 Å (Figures 7.9D). Nevertheless, in this orientation, the highly conserved Y84 residue in Pc (typically named Y83 in cyanobacterial and eukaryotic Pc) points directly towards the heme-binding Y1 of Cf (Y1-Y84 distance of 5.1 Å; Fe-Y84 distance of 9.9 Å) (Figures 7.9D). One could speculate that this is an alternative binding mode and that electron transfer may be possible.

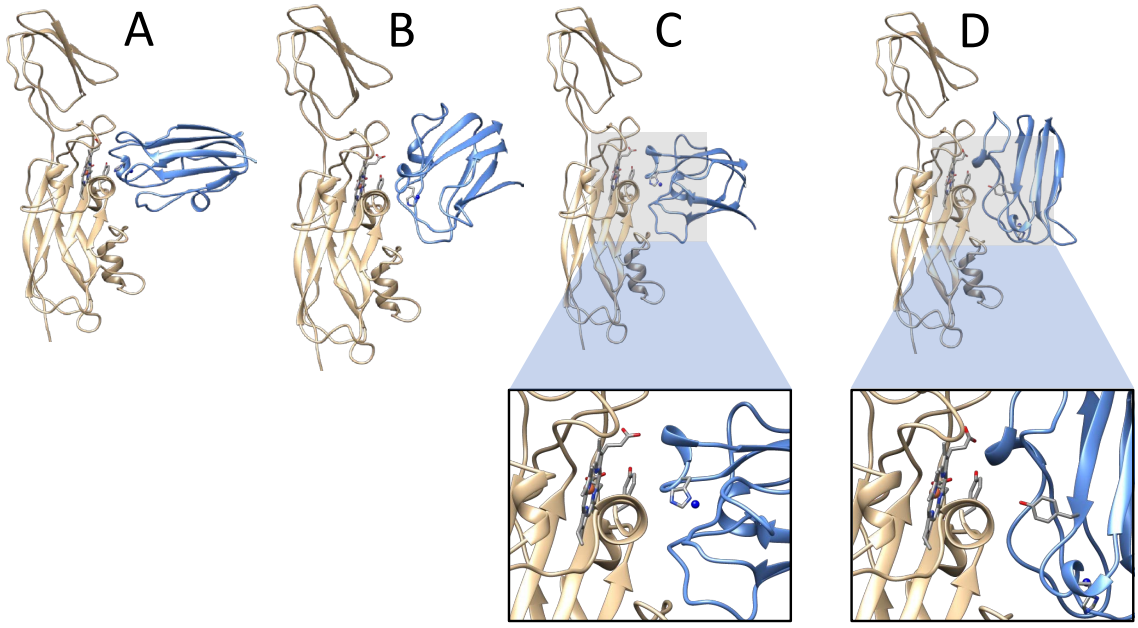


Figure 7.9 Representative structures for the [Cf:Pc] complex of plants and best-energy docking models for the [Cf:Cc6] complexes of *C. reinhardtii* and *P. tricornutum*. (A) Representative structure for the plant [Cf:Pc] complex (turnip Cf and spinach Pc; PDB code, 2PCF) (Ubbink et al., 1998). Pc is coloured in blue and the copper-bound His87 is shown. (B, C) Best-energy docking models for efficient ET between Cf [1](in light brown) and Cc6 (in red) of the green alga *C. reinhardtii* (rank 1, docking energy –32.0 a.u., distance between Fe in Cf to heme in Cc6 of 8.2 Å) and *P. tricornutum* (rank 6, docking energy –24.7 a.u., distance between Fe in Cf to heme in Cc6 of 8.0 Å, the shortest distance model).

7.6. Conclusions

In this chapter, we have illustrated the applicability of computational docking using pyDock on a case study relevant to understanding photosynthesis at the molecular level. The structural models provide an explanation for the differences in photosynthetic efficiency between red and green algae. But the lower docking energy model obtained for the [Cf:Pc] complex in *T. oceanica* (Figure 7.8), despite some evidence, is still highly speculative. Other approaches, such as molecular dynamics (MD) followed by experimental confirmation, will be necessary to be able to make a definite statement

8. General discussion

New developments for pyDock and pyDockDNA to address current challenges

This thesis describes the development of technical upgrade and new functionalities in the protein-protein docking software pyDock, as well as the implementation of a new web server for protein-DNA docking.

The program pyDock has been substantially updated in order to extend its applicability from a technical point of view and be ready for the new challenges in the field, like the use of molecules different from proteins, including cofactors. For the immediate future, it will be extending by integrating pyProCT into this new code in order to increase the reproducibility of the clustering protocol shown in Chapter 5.1.3 (facilitating a broader applicability). Currently pyDock 4.0 is in development alpha phase, and will be soon updated to the Beta-phase to make it publicly available for the scientific community, as local distribution and also as a web server.

The new server pyDockDNA shows reasonable predictive success rates on the available benchmarks. But the number of cases that are available for benchmarking is still too low for optimal testing of new developments. The number of cases in which the structure of the unbound DNA is available is not likely to increase, so we will need to rely on modeling methodologies. Fortunately, there is a variety of modeling strategies, especially those based on deep learning, which might provide accurate models for unbound protein and DNA in a much larger number of cases. These models might include ensembles of conformers, which can also provide better predictions when used in docking. In addition, the pyDockDNA server will be extended with new functionalities. For instance, there are plans to apply more efficient distance restraints between residues and nucleotides, to increase the quality of the generated models.

Integration of *ab initio* and template-based docking

In this thesis, we have explored the combination of *ab initio* docking with template-based docking. This strategy is particularly adequate for cases in which only low-quality templates are available, according to either their SI values (using sequence alignments) or their TM-scores (using structural alignments). For that, a set of models were generated by *ab initio*

sampling to identify suitable templates with structural alignment methods. Then, the models were scored by a combined function based on the template similarity (TM-score) and the binding energy (pyDock). This strategy improved the predictions with respect to using *ab initio* or template-based docking alone. The advantages of combining *ab initio* and template-based docking were confirmed through our participation in the CAPRI and CASP-CAPRI rounds. In particular, in the 3rd Joint CASP-CAPRI experiment our group obtained excellent results, where the energy-based scoring function helped to identify correct models among the different alternatives generated by template-based docking approaches. In a similar way, the application of energy-based scoring as well as other functions as those implemented in CCharPPI server [78, 79] can be extended to the discrimination of biologically meaningful crystallographic homo-dimeric complexes from the artefactual dimeric interactions observed in crystal packing.

Assessment of protein-protein docking

In our participation in CASP-CAPRI rounds, our protein-protein docking approaches, either *ab initio* or template-based, needed the structure or reasonable models of the interacting subunits. These were in general obtained from the CASP-hosted servers, which previously had automatically generated models for these subunits, as they were also targets for CASP in other categories. During the 4th Joint CASP-CAPRI round, the developers of AlphaFold (AF) participated in CASP15 targets, obtaining unprecedented results in the *ab initio* prediction of the protein structures [64]. But since this program did not participate as servers, the CAPRI community did not use these models for the multimeric assembly round. Many of the targets proposed in that round (Chapter 6.5.3) did not have sufficiently good templates for quality modeling, which affected to the success of the docking predictions. This has changed in the last round (5th Joint CASP-CAPRI), where the AlphaFold models for the individual subunits and the complexes were available for the participants in the multimeric assembly section. In addition, the AF-Multimer version can now model protein complexes, with reported success rates of 51% with a false positive rate (FRP) of 1% (paper in preprint) [88]. This Joint CASP-CAPRI round will be a fire test to AF and will provide the

predictive capabilities of this approach in blind conditions. These results will be discussed at the next meeting, to be held in Antalya (Turkey) from 10 to 13 December 2022.

Application to cases of biological interest

We have illustrated the applicability of computational docking using pyDock on a case study relevant to understanding photosynthesis at the molecular level. The structural models provide an explanation for the differences in photosynthetic efficiency between red and green algae. But the lower docking energy model obtained for the [Cf:Pc] complex in *T. oceanica* (Figure 7.7), despite some evidence, is still highly speculative. Other approaches, such as molecular dynamics (MD) followed by experimental confirmation, will be necessary to be able to make a definite statement.

9. Conclusions

- Refactoring of pyDock has successfully provided a more efficient and accurate version of the software to address the new technical and scientific challenges of docking.

- The new functionalities implemented in pyDock version 4.0 have extended the applicability of docking to more relevant types of molecules and provides a more efficient clustering of docking models.

- The implementation of the new pyDockDNA web server facilitates open access to state-of-the-art modeling of protein-DNA complexes.

- The new protein-DNA docking method showed around 40% success rate for the top 10 models on a standard benchmark set of unbound proteins and modelled DNA molecules in canonical B-DNA conformation.

- The integration of *ab initio* docking scoring and template-based docking improves the predictive success rates for modeling protein-protein complexes when using templates with remote homology.

- The pyDock scoring function was particularly successful on the multimeric targets of the CASP13 edition and on the varied protein complexes of the last CAPRI rounds.

- *Ab initio* docking was found to be adequate for predicting difficult targets in which template-based docking was not helpful.

- Docking models of complexes involving redox proteins provide a mechanistic explanation for the differences in photosynthetic efficiency between red and green algae.

10. Appendices

10.1. Appendix 1. Supplementary material for Chapter 5

Table 10.1.1. pyDockDNA docking performance for protein-DNA docking benchmark using ten random rotations of input structures for each case**.**

	% success rates			
rotation	**top 1**	**top 5**	**top 10**	**top 100**
#1	8.5	12.8	17.0	44.7
#2	4.3	14.9	21.3	44.7
#3	8.5	12.8	17.0	44.7
#4	4.3	8.5	12.8	36.2
#5	6.4	14.9	19.1	53.2
#6	6.4	12.8	17.0	42.6
#7	8.5	17.0	19.1	53.2
#8	4.3	8.5	12.8	42.6
#9	4.3	12.8	14.9	46.8
#10	4.3	8.5	12.8	40.4
average	***6.0***	***12.4***	***16.4***	***44.9***

Table 10.1.2. Predictive performance for the top N=1, 5, 10, 100 models of pyDockDNA and different combinations of scoring terms on the protein-DNA docking benchmark in detail.

ELECTROSTATICS + VAN DER WAALS							
	AVERAGE	SIMP COMBINATION	GEO REP/GEO REP ENERGY	GEO REP/CLUSTER MIN ENERGY	GEO REP/CLUSTER AVE ENERGY	LOW ENERGY REP/CLUSTER MIN ENERGY	LOW ENERGY REP/CLUSTER AVE ENERGY
1	4.89	0.00	6.38	0.00	6.38	0.00	6.38
5	13.40	19.15	17.02	23.40	21.28	23.40	21.28
10	**22.13**	**25.53**	**23.40**	**27.66**	**23.40**	**27.66**	**23.40**
100	46.81	48.94	51.06	53.19	48.94	57.45	51.06
pyDock							
	AVERAGE	SIMP COMBINATION	GEO REP/GEO REP ENERGY	GEO REP/CLUSTER MIN ENERGY	GEO REP/CLUSTER AVE ENERGY	LOW ENERGY REP/CLUSTER MIN ENERGY	LOW ENERGY REP/CLUSTER AVE ENERGY
1	5.96	10.64	10.64	8.51	8.51	10.64	8.51
5	12.34	19.15	19.15	23.40	14.89	23.40	14.89
10	**16.38**	**23.40**	**23.40**	**23.40**	**21.28**	**23.40**	**21.28**
100	44.89	36.17	44.68	46.81	42.55	46.81	42.55
ELECTROSTATICS + 0.1 VAN DER WAALS							
	AVERAGE	SIMP COMBINATION	GEO REP/GEO REP ENERGY	GEO REP/CLUSTER MIN ENERGY	GEO REP/CLUSTER AVE ENERGY	LOW ENERGY REP/CLUSTER MIN ENERGY	LOW ENERGY REP/CLUSTER AVE ENERGY
1	6.38	10.64	12.77	8.51	14.89	10.6383	14.89
5	17.66	21.28	31.91	25.53	25.53	27.6596	25.53
10	**21.28**	**27.66**	**31.91**	**36.17**	**29.79**	**38.2979**	**27.66**
100	49.36	46.81	53.19	48.94	51.06	51.0638	53.19
ELECTROSTATICS + VAN DER WALLS + SOLVATION							
	AVERAGE	SIMP COMBINATION	GEO REP/GEO REP ENERGY	GEO REP/CLUSTER MIN ENERGY	GEO REP/CLUSTER AVE ENERGY	LOW ENERGY REP/CLUSTER MIN ENERGY	LOW ENERGY REP/CLUSTER AVE ENERGY
1	3.83	2.13	6.38	2.13	2.13	2.13	2.13
5	12.77	14.89	14.89	19.15	14.89	19.15	14.89
10	**19.36**	**23.40**	**21.28**	**27.66**	**19.15**	**31.91**	**19.15**
100	46.17	53.19	53.19	57.45	53.19	57.45	57.45

10.2. Appendix 2. Supplementary material for Chapter 6

Table 10.2.1. Success rate of template base modelling by using MM-align.

Success Rate MM-align 100% SI and 0.4 of TM-score (MM-align)					Success Rate MM-align 95% SI and 0.4 of TM-score (MM-align)				Success Rate MM-align 70% SI and 0.4 of TM-score (MM-align)				Success Rate MM-align 30% SI and 0.4 of TM-score (MM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	83	51.23	78	82.11	71	44.65	67	81.71	46	30.26	39	70.91	18	13.53	14	53.85
5	88	54.32	80	84.21	75	47.17	69	84.15	51	33.55	41	74.55	23	17.29	16	61.54
10	90	55.56	80	84.21	76	47.80	69	84.15	51	33.55	41	74.55	23	17.29	16	61.54
100	91	56.17	80	84.21	77	48.43	69	84.15	54	35.53	41	74.55	27	20.30	16	61.54
Coverage	162	93.10	95	54.60	159	91.38	82	47.13	152	87.36	55	31.61	133	76.44	26	14.94

Success Rate MM-align 100% SI and 0.5 of TM-score (MM-align)					Success Rate MM-align 95% SI and 0.5 of TM-score (MM-align)				Success Rate MM-align 30% SI and 0.5 of TM-score (MM-align)				Success Rate MM-align 30% SI and 0.5 of TM-score (MM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	81	65.32	78	84.78	69	58.47	67	84.81	43	43.88	39	78.00	16	27.12	14	70.00
5	83	66.94	79	85.87	72	61.02	68	86.08	46	46.94	40	80.00	21	35.59	15	75.00
10	83	66.94	79	85.87	72	61.02	68	86.08	46	46.94	40	80.00	21	35.59	15	75.00
100	83	66.94	79	85.87	72	61.02	68	86.08	47	47.96	40	80.00	21	35.59	15	75.00
Coverage	124	71.26	92	52.87	118	67.82	79	45.40	98	56.32	50	28.74	59	33.91	20	11.49

Success Rate MM-align 100% SI and 0.6 of TM-score (MM-align)					Success Rate MM-align 95% SI and 0.6 of TM-score (MM-align)				Success Rate MM-align 70% SI and 0.6 of TM-score (MM-align)				Success Rate MM-align 30% SI and 0.6 of TM-score (MM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	80	82.47	75	86.21	69	82.14	62	86.11	43	71.67	33	80.49	15	60.00	10	76.92
5	81	83.51	75	86.21	70	83.33	62	86.11	44	73.33	33	80.49	16	64.00	10	76.92
10	81	83.51	75	86.21	70	83.33	62	86.11	44	73.33	33	80.49	16	64.00	10	76.92
100	81	83.51	75	86.21	70	83.33	62	86.11	44	73.33	33	80.49	16	64.00	10	76.92
Coverage	97	55.75	87	50.00	84	48.28	72	41.38	60	34.48	41	23.56	25	14.37	13	7.47

Success Rate MM-align 100% SI and 0.7 of TM-score (MM-align)					Success Rate MM-align 95% SI and 0.7 of TM-score (MM-align)				Success Rate MM-align 70% SI and 0.7 of TM-score (MM-align)				Success Rate MM-align 30% SI and 0.7 of TM-score (MM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	77	84.62	75	87.21	65	84.42	61	87.14	36	80.00	27	81.82	10	71.43	6	85.71
5	77	84.62	75	87.21	65	84.42	61	87.14	36	80.00	27	81.82	10	71.43	6	85.71
10	77	84.62	75	87.21	65	84.42	61	87.14	36	80.00	27	81.82	10	71.43	6	85.71
100	77	84.62	75	87.21	65	84.42	61	87.14	36	80.00	27	81.82	10	71.43	6	85.71
Coverage	91	52.30	86	49.43	77	44.25	70	40.23	45	25.86	33	18.97	14	8.05	7	4.02

Success Rate MM-align 100% SI and 0.8 of TM-score (MM-align)					Success Rate MM-align 95% SI and 0.8 of TM-score (MM-align)				Success Rate MM-align 70% SI and 0.8 of TM-score (MM-align)				Success Rate MM-align 30% SI and 0.8 of TM-score (MM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	75	88.24	73	89.02	61	88.41	57	89.06	27	81.82	21	80.77	5	100	4	100
5	75	88.24	73	89.02	61	88.41	57	89.06	27	81.82	21	80.77	5	100	4	100
10	75	88.24	73	89.02	61	88.41	57	89.06	27	81.82	21	80.77	5	100	4	100
100	75	88.24	73	89.02	61	88.41	57	89.06	27	81.82	21	80.77	5	100	4	100
Coverage	85	48.85	82	47.13	69	39.66	64	36.78	33	18.97	26	14.94	5	2.87	4	2.30

Table 10.2.2. Success rate of template base modelling by using TM-align.

Success Rate TM-align 100% SI and 0.4 of TM-score (TM-align)					Success Rate TM-align 95% SI and 0.4 of TM-score (TM-align)				Success Rate TM-align 70% SI and 0.4 of TM-score (TM-align)				Success Rate TM-align 30% SI and 0.4 of TM-score (TM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	90	52.02	87	53.05	75	43.35	71	44.10	48	27.91	44	28.03	17	9.94	13	8.90
5	91	52.60	90	54.88	77	44.51	75	46.58	51	29.65	48	30.57	18	10.53	19	13.01
10	92	53.18	90	54.88	79	45.66	75	46.58	53	30.81	48	30.57	20	11.70	19	13.01
100	92	53.18	90	54.88	79	45.66	75	46.58	53	30.81	48	30.57	21	12.28	20	13.70
Coverage	173	100.00	164	94.80	173	100.00	161	93.06	172	99.42	157	90.75	171	98.84	146	84.39
Success Rate TM-align 100% SI and 0.5 of TM-score (TM-align)					Success Rate TM-align 95% SI and 0.5 of TM-score (TM-align)				Success Rate TM-align 70% SI and 0.5 of TM-score (TM-align)				Success Rate TM-align 30% SI and 0.5 of TM-score (TM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	90	55.21	87	65.41	75	47.47	71	58.20	48	33.10	43	43.00	17	13.82	13	18.84
5	91	55.83	88	66.17	77	48.73	73	59.84	51	35.17	45	45.00	18	14.63	16	23.19
10	91	55.83	88	66.17	78	49.37	73	59.84	52	35.86	45	45.00	19	15.45	16	23.19
100	92	56.44	88	66.17	78	49.37	73	59.84	52	35.86	45	45.00	20	16.26	16	23.19
Coverage	163	94.22	133	76.88	158	91.33	122	70.52	145	83.82	100	57.80	123	71.10	69	39.88
Success Rate TM-align 100% SI and 0.6 of TM-score (TM-align)					Success Rate TM-align 95% SI and 0.6 of TM-score (TM-align)				Success Rate TM-align 70% SI and 0.6 of TM-score (TM-align)				Success Rate TM-align 30% SI and 0.6 of TM-score (TM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	90	66.67	83	76.85	75	59.52	67	72.04	48	46.15	39	60.00	17	24.64	10	29.41
5	91	67.41	84	77.78	77	61.11	69	74.19	51	49.04	41	63.08	18	26.09	13	38.24
10	91	67.41	84	77.78	77	61.11	69	74.19	51	49.04	41	63.08	18	26.09	13	38.24
100	91	67.41	84	77.78	77	61.11	69	74.19	51	49.04	41	63.08	19	27.54	13	38.24
Coverage	135	78.03	108	62.43	126	72.83	93	53.76	104	60.12	65	37.57	69	39.88	34	19.65
Success Rate TM-align 100% SI and 0.7 of TM-score (TM-align)					Success Rate TM-align 95% SI and 0.7 of TM-score (TM-align)				Success Rate TM-align 70% SI and 0.7 of TM-score (TM-align)				Success Rate TM-align 30% SI and 0.7 of TM-score (TM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	85	78.70	82	81.19	69	75.00	65	78.31	41	64.06	33	68.75	11	40.74	7	46.67
5	85	78.70	82	81.19	69	75.00	65	78.31	41	64.06	33	68.75	11	40.74	7	46.67
10	85	78.70	82	81.19	69	75.00	65	78.31	41	64.06	33	68.75	11	40.74	7	46.67
100	85	78.70	82	81.19	69	75.00	65	78.31	41	64.06	33	68.75	11	40.74	7	46.67
Coverage	108	62.43	101	58.38	92	53.18	83	47.98	64	36.99	48	27.75	27	15.61	15	8.67
Success Rate TM-align 100% SI and 0.8 of TM-score (TM-align)					Success Rate TM-align 95% SI and 0.8 of TM-score (TM-align)				Success Rate TM-align 70% SI and 0.8 of TM-score (TM-align)				Success Rate TM-align 30% SI and 0.8 of TM-score (TM-align)			
Top	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	TM-score average	NSC	TM-score min
1	82	82.83	81	84.38	66	81.48	62	83.78	34	72.34	27	75.00	5	50.00	5	62.50
5	82	82.83	81	84.38	66	81.48	62	83.78	34	72.34	27	75.00	5	50.00	5	62.50
10	82	82.83	81	84.38	66	81.48	62	83.78	34	72.34	27	75.00	5	50.00	5	62.50
100	82	82.83	81	84.38	66	81.48	62	83.78	34	72.34	27	75.00	5	50.00	5	62.50
Coverage	99	57.23	96	55.49	81	46.82	74	42.77	47	27.17	36	20.81	10	5.78	8	4.62

Table 10.2.3. Success rate of *ab initio* modelling combining the TM-score(s) and pyDock scoring functions.

Success Rate MM-align 100% SI and 0.4 of TM-score (MM-align)									Success Rate MM-align 70% SI and 0.4 of TM-score (MM-align)							
Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	37	24.34	34	31.19	35	23.03	29	26.61	20	13.89	16	17.20	23	15.97	15	16.13
5	41	26.97	37	33.94	41	26.97	35	32.11	25	17.36	20	21.51	25	17.36	20	21.51
10	42	27.63	37	33.94	42	27.63	38	34.86	26	18.06	20	21.51	26	18.06	21	22.58
100	45	29.61	38	34.86	45	29.61	38	34.86	29	20.14	21	22.58	29	20.14	21	22.58
Coverage	152	86.36	109	61.93	152	86.36	109	61.93	144	81.82	93	52.84	144	81.82	93	52.84
Success Rate MM-align 100% SI and 0.5 of TM-score (MM-align)									Success Rate MM-align 70% SI and 0.5 of TM-score (MM-align)							
Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	37	34.91	26	54.17	35	33.02	26	54.17	20	24.39	12	40.00	21	25.61	13	43.33
5	39	36.79	28	58.33	39	36.79	28	58.33	22	26.83	14	46.67	22	26.83	14	46.67
10	40	37.74	28	58.33	40	37.74	28	58.33	23	28.05	14	46.67	23	28.05	14	46.67
100	40	37.74	28	58.33	40	37.74	28	58.33	23	28.05	14	46.67	23	28.05	14	46.67
Coverage	106	60.23	48	27.27	106	60.23	48	27.27	82	46.59	30	17.05	82	46.59	30	17.05
Success Rate MM-align 100% SI and 0.6 of TM-score (MM-align)									Success Rate MM-align 70% SI and 0.6 of TM-score (MM-align)							
Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	35	53.03	22	68.75	33	50.00	23	71.88	18	45.00	11	61.11	19	47.50	12	66.67
5	36	54.55	23	71.88	36	54.55	23	71.88	19	47.50	12	66.67	19	47.50	12	66.67
10	36	54.55	23	71.88	36	54.55	23	71.88	19	47.50	12	66.67	19	47.50	12	66.67
100	36	54.55	23	71.88	36	54.55	23	71.88	19	47.50	12	66.67	19	47.50	12	66.67
Coverage	66	37.50	32	18.18	66	37.50	32	18.18	40	22.73	18	10.23	40	22.73	18	10.23
Success Rate MM-align 100% SI and 0.7 of TM-score (MM-align)									Success Rate MM-align 70% SI and 0.7 of TM-score (MM-align)							
Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	27	72.97	19	70.37	26	70.27	20	74.07	13	72.22	9	69.23	13	72.22	9	69.23
5	28	75.68	20	74.07	28	75.68	20	74.07	13	72.22	10	76.92	13	72.22	10	76.92
10	28	75.68	20	74.07	28	75.68	20	74.07	13	72.22	10	76.92	13	72.22	10	76.92
100	28	75.68	20	74.07	28	75.68	20	74.07	13	72.22	10	76.92	13	72.22	10	76.92
Coverage	37	21.02	27	15.34	37	21.02	27	15.34	18	10.23	13	7.39	18	10.23	13	7.39
Success Rate MM-align 100% SI									Success Rate MM-align 70% SI and 0.8 of TM-score (MM-align)							
Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	20	76.92	15	83.33	20	76.92	16	88.89	10	83.33	5	71.43	10	83.33	6	85.71
5	20	76.92	16	88.89	20	76.92	16	88.89	10	83.33	6	85.71	10	83.33	6	85.71
10	20	76.92	16	88.89	20	76.92	16	88.89	10	83.33	6	85.71	10	83.33	6	85.71
100	20	76.92	16	88.89	20	76.92	16	88.89	10	83.33	6	85.71	10	83.33	6	85.71
Coverage	26	14.77	18	10.23	26	14.77	18	10.23	12	6.82	7	3.98	12	6.82	7	3.98

Success Rate MM-align 95% SI and 0.4 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	30	19.87	27	25.23	30	19.87	22	20.56
5	35	23.18	31	28.97	35	23.18	29	27.10
10	36	23.84	31	28.97	36	23.84	32	29.91
100	39	25.83	32	29.91	39	25.83	32	29.91
Coverage	151	85.80	107	60.80	151	85.80	107	60.80

Success Rate MM-align 30% SI and 0.4 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	7	4.43	2	33.33	8	5.06	2	33.33
5	21	13.29	2	33.33	17	10.76	2	33.33
10	21	13.29	2	33.33	25	15.82	2	33.33
100	35	22.15	2	33.33	35	22.15	2	33.33
Coverage	158	89.77	6	3.41	158	89.77	6	3.41

Success Rate MM-align 95% SI and 0.5 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	30	30.30	19	46.34	29	29.29	19	46.34
5	32	32.32	21	51.22	32	32.32	21	51.22
10	33	33.33	21	51.22	33	33.33	21	51.22
100	33	33.33	21	51.22	33	33.33	21	51.22
Coverage	99	56.25	41	23.30	99	56.25	41	23.30

Success Rate MM-align 30% SI and 0.5 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	4	5.88	2	50	5	7.35	2	50
5	12	17.65	2	50	11	16.18	2	50
10	12	17.65	2	50	12	17.65	2	50
100	15	22.06	2	50	15	22.06	2	50
Coverage	68	38.64	4	2.272727	68	38.64	4	2.272727273

Success Rate MM-align 95% SI and 0.6 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	28	48.28	15	62.50	27	46.55	16	66.67
5	29	50.00	16	66.67	29	50.00	16	66.67
10	29	50.00	16	66.67	29	50.00	16	66.67
100	29	50.00	16	66.67	29	50.00	16	66.67
Coverage	58	32.95	24	13.64	58	32.95	24	13.64

Success Rate MM-align 30% SI and 0.6 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	2	66.67	2	100	2	66.67	2	100.00
5	2	66.67	2	100	2	66.67	2	100.00
10	2	66.67	2	100	2	66.67	2	100.00
100	2	66.67	2	100	2	66.67	2	100.00
Coverage	3	1.70	2	1.136364	3	1.70	2	1.14

Success Rate MM-align 95% SI and 0.7 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	21	72.41	13	72.22	20	68.97	14	77.78
5	21	72.41	14	77.78	21	72.41	14	77.78
10	21	72.41	14	77.78	21	72.41	14	77.78
100	21	72.41	14	77.78	21	72.41	14	77.78
Coverage	29	16.48	18	10.23	29	16.48	18	10.23

Success Rate MM-align 30% SI and 0.7 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	2	100.00	1	100.00	2	100	1	100.00
5	2	100.00	1	100.00	2	100	1	100.00
10	2	100.00	1	100.00	2	100	1	100.00
100	2	100.00	1	100.00	2	100	1	100.00
Coverage	2	1.14	1	0.57	2	1.136363636	1	0.57

Success Rate MM-align 95% SI and 0.8 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	14	77.78	10	76.92	14	77.78	11	84.62
5	14	77.78	11	84.62	14	77.78	11	84.62
10	14	77.78	11	84.62	14	77.78	11	84.62
100	14	77.78	11	84.62	14	77.78	11	84.62
Coverage	18	10.23	13	7.39	18	10.23	13	7.39

Success Rate MM-align 30% SI and 0.8 of TM-score (MM-align)

Top	NSC	TM-score average	NSC	TM-score min	NSC	pyDock+TM-score average	NSC	pyDock+TM-score min
1	1	100.00	0	0	1	100.00	0	0
5	1	100.00	0	0	1	100.00	0	0
10	1	100.00	0	0	1	100.00	0	0
100	1	100.00	0	0	1	100.00	0	0
Coverage	1	0.57	0	0	1	0.57	0	0

Table 10.2.4. Structural availability of the interacting molecules and additional information for the preparation of the submitted models as servers and predictors.

Target[a]	RECEPTOR		LIGAND		COMPLEX		
	Unbound Structure (int RMSD)[b]	Template (SI, RMSD)[c]	Unbound Structure (int RMSD)[b]	Template (SI, RMSD)[c]	Template (SI, RMSD)[c]	Docking restraints[d]	Target structure[e]
Protein-protein							
T122	5MXA (3.4 A)	-	-	111R (25%, 10.4 A)	-	Literature	5MZV
T123	5LMW (N/A)	5BOZ (76%, N/A)	-	multi-template I-TASSER (N/A, N/A)	-	-	6EY0
T124	5FWO (2.2 A)	4GRW:H (78%, 3.0 A)	-	multi-template I-TASSER (N/A, N/A)	Dimer: 3MTR (17%, 21.9 A)	-	6EY6
T125 ***hetero***	4QKH (1.4 A)	-	-	3T3A (47%, 10.6 A)	-	Literature	5MGT
T125 ***homo***	4QKH (1.4 A)	-	4QKH (1.3 A)	-	4QKH (100%, 0.7 Å)	-	5MGT
T131	4WHD (1.4 Å)	-	-	5LP2 (96%,0.9 Å)	-	Literature	6GBG
132	4WHD (1.0 Å)	-	-	5LP2 (57%,2.9 Å)	-	Literature	6GBH
T133	-	3U43:B (87%, 0.8 Å)	-	3U43:A (83%,1.5 Å)	-	-	6ERE
T136	-	5FKZ:E (45%, N/A), multi-template I-TASSER (N/A)	-	5FKZ:E (45%, N/A), multi-template I-TASSER (N/A, N/A)	2VYC (40%, N/A), 5FKZ (45%, N/A)	5FKZ interface	6Q6I
Protein-peptide							
T121	1LR0 (N/A)	-	-	multi-template I-TASSER (N/A,N/A), 2HQS (42%, N/A), 2W8B (38%, N/A)	-	1TOL interface	6S3W
T134	1F3C (2.0 Å)	-	-	1F95 (33%,1.2 Å), 1F96 (9%, 2.4 Å), 3P8M (36%, 2.3 Å)	-	1F95 interface	6GZL
T135	1F3C (2.0 Å)	-	-	1F95 (33%, 1.2 Å)	-	1F95 interface	6GZL
Protein-saccharide							
T126	-	5F7V (30%, N/A)	2C7F.2X8S, 3QEF, CID 53356682 (N/A)	-	-	-	6RKH
T127	-	5F7V (30%, N/A)	CID 74539968, (N/A)	-	-	-	6RKX
T128	-	5F7V (30%, N/A)	5HOF, CID 74539969 (N/A)	-	-	-	6RL2
T129	-	5F7V (30%, N/A)	1GYE, CID 74539970 (N/A)	-	-	-	6RL1
T130	3D5Z (0.7 Å)	-	5HOF (2.6 Å), CID 74539969 (2.3 Å)	-	-	-	6F1G

[a] Underscored: target of special difficulty, with only 3 or fewer groups that submitted correct models within their top 5 submitted ones.
[b] PDB code (and chain ID if needed) of the unbound structure (or that bound to a different partner in the case of the saccharides) used for docking. In rackets, interface RMSD with respect to the bound structure, calculated on all the atoms from interface residues. Interface residues are those with at least one atom within 8 Å distance from any atom of the partner molecule, after superimposing the unbound structure onto the corresponding bound structure in the complex. "N/A": if reference complex structure is not yet available. PubChem CID code given in case of non-peptidic ligands. [c]PDB code (and chain ID if needed) of the template used for homology-based modeling of the receptor or ligand, or template-based docking for the complex (or part of it), as indicated. In some targets, I-TASSER was applied, using multiple templates, as indicated. In brackets, sequence identity, and global RMSD of the Cα atoms of the template with respect to the bound structure ("N/A": if reference complex structure is not yet available). d Use of distance restraints for docking based on interface residues, either from available template for the complex, or from literature, as indicated. e PDB code of the complex reference, if it is now available

Table 10.2.5. Information on the targets of the 7th CAPRI

CAPRI 7th experiment						
Target	**Level**	**Stoich.**	**#Int**	**#Res**	**PDB**	**Description**
Protein-protein complexes						
T122	Difficult	A1B1C1	1	198/328/330	5MZV	Human cytokine hetero-dimer/receptor complex IL23/IL23R
T123	Difficult	A1B1	1	174/121	6EY0	PorM-Nt/nb(02)
T124	Difficult	A2B1	1	202/141	6EY6	PorM-Ct/nb(130)
T125	Difficult	A2B4	5	135/146	5MGT	Hetero-hexamer of LLT1/NKR-P1 (extra-cellular domains)
T131	Difficult	A1B1	1	108/404	6BGB	Human CEACAM1/HopQ-Type-I *H. pylori*
T132	Medium	A1B1	1	108/418	6BGH	Human CEACAM1/hopQ-Type-II *H. pylori*
T133	Easy	A1B1	1	69/95	6ERE	Redesigned Colicin E2 DNase/Im2 complex
T136	Easy	A10	3	751	6Q6I	LdcA *P.aeroginosa;* EM
Protein-peptide complexes						
T121	Difficult	A1B1	1	115/13	6S3W	*P.aeroginosa* TolAIII domain/N-terminus *P.aeruginosa* TolB
T134	Easy	A2B1	1	88/50	6GZJ	DLC8 dimer/MAG 50-residue fragment
T135	Easy	A2B1	1	88/12	6GZL	DLC8 dimer (Rat)/MAG 12-residue fragment
Protein-oligosaccharide complexes						
T126	Difficult	A1B1	1	415/6	6RKH	Arabino-oligosaccharide binding protein, *G.stearothermophilus*, with AbnE/A6
T127	Difficult	A1B1	1	415/5	6RKX	Idem with AbnE/A5
T128	Medium	A1B1	1	415/4	6RL2	Idem with AbnE/A4
T129	Medium	A1B1	1	415/3	6RL1	Idem with AbnE/A3
T130	Easy	A1B1	1	315/5	6F1G	Arabino-oligosaccharide binding protein, *G.stearothermophilus*, with AbnB/A5

Note: The target list is subdivided into categories: protein-protein (R39:T122-T124; R40:T125; R42:T131, T132; R43:T133; R45:T136), protein-peptide (R38:T121; T44:T134, T135), and protein-polysaccharide (R41:T126-T130). Columns 1 to 3 list the CAPRI target ID, its difficulty, and the number of assessed interfaces; columns 4 to 6 list the number of residues, and the PDB ID; column 7 contains a textual description of the target.

Table 10.2.6. Overall 7th CAPRI performance ranking for the protein-protein targets. Extracted from [204]

			top1		top5		top 10	
Rank	Group name	#targets	rank	performance[1]	rank	performance[1]	rank	performance[1]
1	Kozakov/Vajda	11	2	5/1***/3**	1	6/1***/5**	1	6/1***/5**
2	Venclovas	8	1	5/2***/3**	2	5/2***/3**	2	5/2***/3**
3	Seok	11	6	4/3**	3	5/1***/4**	3	5/1***/4**
3	Pierce	9	3	4/2***/1**	3	5/2***/2**	3	5/2***/2**
5	Andreani/Guerois	11	6	4/3**	5	5/1***/3**	5	5/1***/3**
6	Zou	11	6	3/1***/2**	6	4/1***/3**	6	4/1***/3**
6	Zacharias	11	14	4/1***	6	5/1***/2**	6	5/1***/2**
6	Kihara	11	6	4/1***/1**	6	5/1***/2**	6	5/1***/2**
6	Gray	11	3	5/1***/2**	6	5/1***/2**	6	5/1***/2**
10	Shen	11	14	3/1***/1**	10	4/1***/2**	6	5/1***/2**
10	Moal	8	24	2**	10	4/1***/2**	11	4/1***/2**
10	MDOCKPP	11	5	4/1***/2**	10	4/1***/2**	11	4/1***/2**
10	HADDOCK	11	6	3/1***/2**	10	4/1***/2**	11	4/1***/2**
14	Grudinin	11	6	3/1***/2**	14	3/1***/2**	16	3/1***/2**
14	Fernandez-Recio	11	6	3/1***/2**	14	3/1***/2**	16	3/1***/2**
14	CLUSPRO	11	20	3/2**	14	4/3**	16	4/3**
14	Bonvin	11	6	3/1***/2**	14	3/1***/2**	16	3/1***/2**
18	Weng	11	14	3/1***/1**	18	3/1***/1**	20	3/1***/1**
18	SWARMDOCK	11	14	3/1***/1**	18	3/1***/1**	20	3/1***/1**
18	LZERD	11	14	3**	18	3**	20	3**
18	Chang	11	20	2/1***/1**	18	3/1***/1**	11	4/1***/2**
18	Bates	11	14	3/1***/1**	18	3/1***/1**	11	4/1***/2**
23	Vakser	11	24	2**	23	2/1***/1**	24	2/1***/1**
23	Takeda-Shitaka	8	20	2/1***/1**	23	2/1***/1**	24	2/1***/1**
23	Huang	9	20	2/1***/1**	23	2/1***/1**	20	3/1***/1**
26	Wolfson	7	24	2/1***	26	2/1***	27	2/1***
26	PYDOCKWEB	11	28	1***	26	2/1***	24	3/1***
26	HDOCK	8	30	1**	26	2**	27	2**
26	GALAXYPPDOCK	8	24	2**	26	2**	27	2**
30	Laine	3	28	1***	30	1***	30	1***
31	Ritchie	5	30	1**	31	1**	32	1**
31	Iwadate	1	30	1**	31	1**	32	1**
31	INTERPRED	1	30	1**	31	1**	32	1**
31	Del Carpio	11	30	1**	31	1**	32	1**
31	Carbone	6	36	0	31	1**	30	2/1**
31	Brini	1	30	1**	31	1**	32	1**
37	ZDOCK	1	36	0	37	0	37	0
37	Wang	0	36	0	37	0	37	0
37	Wallner	4	36	0	37	0	37	0
37	UUcourse	0	36	0	37	0	37	0
37	Tuffery	0	36	0	37	0	37	0
37	Schueler-Furman	0	36	0	37	0	37	0
37	Schneidman	1	36	0	37	0	37	0
37	Sanner	0	36	0	37	0	37	0
37	Niv	0	36	0	37	0	37	0
37	Negi	4	36	0	37	0	37	0
37	GRAMM-X	3	36	0	37	0	37	0
37	Gong	0	36	0	37	0	37	0
37	Czaplewski	1	36	0	37	0	37	0
37	Carazo	1	36	0	37	0	37	0

Columns 1 to 3 list the rank, name group and the number of assessed interfaces; columns 4 to 6 list the rank, the quality of the models for the top 1, 5 and 10 uploaded models submitted. [1]This column shows the success of the groups, the first number corresponds to the number of successful targets, then separated by a slash, if necessary, the number and quality of the models is displayed. For example, 5/1***/3**, means 5 successful targets, for which there is one target with high-quality models (***), 3 of medium-quality models (**) and the remaining one is of acceptable quality.

Table 10.2.7. Overall 7th CAPRI performance ranking for the protein-peptide targets. Extracted from [204]

			top1		top 5		top 10	
Rank	Group name	#targets	rank	performance	rank	performance	rank	performance
1	Zacharias	3	1	2/1***/1**	1	2***	2	2***
1	Schueler-Furman	3	1	2/1***/1**	1	2***	1	3/2***/1**
1	Andreani/Guerois	3	1	2/1***/1**	1	2***	2	2***
4	Venclovas	3	11	1	4	2/1***/1**	2	2***
4	Seok	3	7	2/1**	4	3/2**	5	3/2**
4	Moal	3	1	2/1***/1**	4	2/1***/1**	5	2/1***/1**
4	Huang	3	1	2/1***/1**	4	2/1***/1**	5	2/1***/1**
8	HDOCK	2	6	2/1***	8	2/1***	8	2/1***
9	Zou	2	11	1	9	2	12	2
9	Shen	3	8	1**	9	1**	12	1**
9	Kozakov/Vajda	2	11	1	9	1**	8	2**
9	GALAXYPPDOCK	3	21	0	9	1**	12	1**
9	Fernandez-Recio	3	11	1	9	2	12	2
9	CLUSPRO	2	11	1	9	1**	12	1**
9	Brini	2	8	1**	9	1**	12	1**
9	Bonvin	3	8	1**	9	1**	8	2**
17	UUcourse	1	11	1	17	1	12	1**
17	SWARMDOCK	3	21	0	17	1	12	2
17	PYDOCKWEB	2	21	0	17	1	23	1
17	Kihara	3	11	1	17	1	12	1**
17	HADDOCK	3	21	0	17	1	12	2
17	Del Carpio	2	11	1	17	1	23	1
17	Czaplewski	2	11	1	17	1	12	2
17	Bates	3	11	1	17	1	11	3
25	Wang	1	21	0	25	0	25	0
25	Wallner	1	21	0	25	0	25	0
25	Vakser	2	21	0	25	0	25	0
25	Tuffery	1	21	0	25	0	25	0
25	Takeda-Shitaka	1	21	0	25	0	25	0
25	Sanner	1	21	0	25	0	25	0
25	Ritchie	1	21	0	25	0	25	0
25	Niv	1	21	0	25	0	25	0
25	Negi	1	21	0	25	0	25	0
25	MDOCKPP	2	21	0	25	0	25	0
25	LZERD	3	21	0	25	0	25	0
25	Grudinin	3	21	0	25	0	25	0
25	Gong	1	21	0	25	0	25	0
25	Chang	3	21	0	25	0	25	0

Columns 1 to 3 list the rank, name group and the number of assessed interfaces; columns 4 to 6 list the rank, the quality of the models for the top 1, 5 and 10 uploaded models submitted. 1This column shows the success of the groups, the first number corresponds to the number of successful targets, then separated by a slash, if necessary, the number and quality of the models is displayed. For example, 5/1***/3**, means 5 successful targets, for which there is one target with high-quality models (***), 3 of medium-quality models (**) and the remaining one is of acceptable quality**.**

Table 10.2.8. Overall 7th CAPRI performance ranking for the protein-oligosaccharide targets. Extracted from [204]

			top1		top 5		top 10	
Rank	Group name	#targets	rank	performance	rank	performance	rank	performance
1	Andreani/Guerois	5	4	4/1**	1	5/1***/2**	1	5/1***/3**
2	Seok	5	1	5/1***/1**	2	5/1***/1**	2	5/1***/2**
2	LZERD	5	4	5	2	5/1***/1**	2	5/1***/2**
2	Chang	5	2	4/1***	2	5/1***/1**	4	5/1***/1**
5	Kozakov/Vajda	5	9	3/1**	5	5/2**	8	5/2**
5	Huang	5	9	4	5	5/2**	8	5/2**
5	HDOCK	5	24	1	5	4/3**	4	5/3**
5	CLUSPRO	5	24	1	5	5/2**	8	5/2**
9	Zou	5	9	4	9	5/1**	14	5/1**
9	Zacharias	5	4	5	9	5/1**	14	5/1**
9	Venclovas	5	4	3/1***	9	4/1***	14	4/1***
9	Takeda-Shitaka	5	9	4	9	5/1**	14	5/1**
9	MDOCKPP	5	14	3	9	5/1**	14	5/1**
9	Kihara	5	9	4	9	4/2**	4	5/3**
9	Grudinin	5	2	4/1***	9	4/1***	8	5/1***
9	Gray	5	24	1	9	4/2**	14	4/2**
9	Fernandez-Recio	5	19	2	9	5/1**	8	5/2**
9	Bonvin	5	14	2/1**	9	3/1***/1**	8	4/1***/1**
19	Shen	5	19	2	19	5	14	5/1**
19	HADDOCK	5	14	2/1**	19	3/1***	4	5/1***/1**
19	Carbone	4	4	3/1***	19	3/1***	22	3/1***
22	Vakser	4	19	2	22	4	24	4
22	Moal	5	30	0	22	4	22	5
22	GALAXYPPDOCK	5	14	3	22	3/1**	14	4/1***
25	Pierce	2	14	2/1**	25	2/1**	25	2/1**
25	Bates	5	19	2	25	3	25	3
27	SWARMDOCK	5	24	1	27	2	28	2
27	Negi	4	19	2	27	2	28	2
29	Ritchie	1	24	1	29	1	28	1**
29	Del Carpio	5	24	1	29	1	25	3

Columns 1 to 3 list the rank, name group and the number of assessed interfaces; columns 4 to 6 list the rank, the quality of the models for the top 1, 5 and 10 uploaded models submitted. 1This column shows the success of the groups, the first number corresponds to the number of successful targets, then separated by a slash, if necessary, the number and quality of the models is displayed. For example, 5/1***/3**, means 5 successful targets, for which there is one target with high-quality models (***), 3 of medium-quality models (**) and the remaining one is of acceptable quality.

Table 10.2.9. Information on the targets of the third and fourth CAPRI-CASP experiments

3rd Joint CAPRI-CASP experiment

Easy targets	CASP ID	Stoich.	#Int[1]	#Res[2]	PDB[3]	Description
T140	T0973	A2	1	146	6YFN	Bacteriophage ESE058 coat protein
T143	T0983	A2	1	245	6UK5	Cals10 protein
T144	**T0984**	A2	1	752	6NQ1	Two-pore calcium channel protein; EM
T152	T1003	A2	1	474	6HRH	ALAS2, 50-Aminolevulinate synthase 2
T153	T1006	A2	1	79	6QEK	Putative membrane transporter (C. desulfamplus)
T147	T0995	A2/A4/A8	3	330	-	Cyanide dihydratase (B. pumilus); EM
T158	T1020	A3	1	577	7WNQ	SLAC1 protein
T139	**T0961**	A4	2	505	6SD8	Acyl-CoA dehydrogenase from Bdellovibrio bacteriovorus
T142	H0974	A1B1	1	70/80	6TRI	Repressor-antirepressor complex (lysogeny switch)
Difficult targets	**CASP ID**	**Stoich.**	**#Int[1]**	**#Res[2]**	**PDB[3]**	**Description**
T137	T0965	A2	2	326	6D2V	NADP-dependent reductase
T138	T0966	A2	2	494	5W6L	RasRap1 site-specific endopeptidase
T141	T0976	A2	1	252	6MXV	Rhodanese-like family protein, bacteria
T148	T0997	A2	1	228	-	LD-transpeptidase
T149	T0999	A2	5	1589	6HQV	Pentafunctional AROM polypeptide: five main enzymes of the shikimate pathway
T150	T0999	A2	5	1589	6HQV	Idem; with SAXS data
T151	T0999	A2	5	1589	6HQV	Idem; with crosslinking data
T154	T1009	A2	1	718	6DRU	Alpha-xylosidase
T155	H1015	A1B1	1	89/129	-	CDI_213 protein, bacteria
T156	H1017	A1B1	1	111/129	-	201_INDD4 protein, E. coli
T157	H1019	A1B1	1	58/88	-	CDI207t protein, E. coli
T146	H0993	A2B2	3	275/112	-	Lipid-transport, bacterial outer membrane
T159	H1021	A6B6C6	7	148/351/295	6RAP	18-mer heterocomplex; EM

4th Joint CAPRI-CASP experiment

Easy targets	CASP ID	Stoich.	#Int[1]	#Res[2]	PDB[3]	Description
T164	T1032	A2	1	284	6n64	SMCHD1 (human) residues 1616–1899
T166	H1045	A1B1	1	157/173	6xod	PEX4/PEX22 complex from *Arabidopsis thaliana*
T168	T1052	A3	1	832	-	Tail fiber of the Salmonella virus epsilon15
T177	H1081	A20	3	758	7pk6 2vyc	Arginine decarboxylase/bacteria
Difficult targets	**CASP ID**	**Stoich.**	**#Int[1]**	**#Res[2]**	**PDB[3]**	**Description**
T169	T1054	A2	1	190	6v4v	Outer-membrane lipoprotein from *Acinetobacter baumannii*
T176	T1078	A2	1	138	7cwp	Tsp1 from Trichoderma virens, small secreted cysteine rich protein (SSCRP)
T178	T1083	A2	1	98	6nq1	Helical segment from *Nitrosococcus oceani*
T179	T1087	A2	1	93	-	Helical segment from *Methylobacter tundripaludum*
T165	H1036	A3H3L3	1	931/128/106	6vn1	MC Ab 93 k bound to varicella-zoster virus glycoprotein gB
T174	T1070	A3	1	335	7rej	Protein of attachment region to phage tail
T170	H1060	A6/B3/C12/D6	9	464/298/140/142	2vyc	Component of the T5 phage tail distal complex
T180	T1099	A240	8 (4)	262	6yhg	Capsid of duck hepatitis B virus

Table 10.2.10. Overall 3rd joint CASP-CAPRI performance ranking [144]

Rank	Predictor group Human	#[a]	Rank	Top-1	Ran	Top-5	Rank	Top-10
1	Venclovas	20	3	9/4***/4**	1	13/6***/6**	1	13/6***/6**
2	Fernandez-Recio	19	1	11/3***/7**	2	12/5***/6**	2	13/5***/6**
3	Seok	20	4	10/2***/6**	3	11/4***/7**	3	11/4***/7**
4	Kihara	20	2	10/3***/6**	4	11/4***/6**	4	11/4***/6**
5	Weng	20	4	9/4***/3**	5	10/5***/4**	4	11/5***/4**
	Kozakov	19	8	10/9**	5	13/11**	6	13/11**
7	Vakser	20	4	11/1***/7**	7	11/2***/8**	6	11/3***/7**
	Huang	19	8	9/3***/4**	7	11/4***/4**	6	12/4***/4**
	Zou	20	8	11/8**	7	13/2***/6**	6	13/3***/5**
	Bates	18	4	9/3***/5**	7	9/5***/4**	10	9/5***/4**
11	Chang	20	12	10/8**	11	10/3***/6**	11	10/3***/6**
12	Eisenstein	12	8	9/3***/4**	12	9/4***/3**	11	10/4***/4**
13	Pierce	20	14	6/2***/3**	13	7/4***/3**	13	8/4***/3**
	Shen	20	12	11/3***/5**	13	11/3***/5**	14	11/3***/5**
15	Elofsson	20	16	6/3**	15	8/2***/3**	15	8/2***/3**
16	Czaplewski	17	17	5/3**	16	7/1***/4**	16	7/2***/3**
	Grudinin	20	18	3/1***/2**	16	7/1***/4**	16	7/1***/5**
18	Moal	17	15	6/1***/2**	18	6/2***/2**	18	6/2***/3**
19	Carbone	20	20	2/1***/1**	19	5/2***/1**	19	5/2***/2**
20	Schneidman	12	18	3/2***	20	3/2***	20	4/2***/1**
21	Hou	11	21	2**	21	2/1***/1**	21	3/2***
22	Ritchie	4	22	1**	22	1**	22	1**
23	Liwo	11	23	0	23	0	23	1
	Crivelli	12	23	0	23	0	23	1
	EMBO 2017 course	1	23	0	23	0	23	0
	Del Carpio	13	23	0	23	0	23	0
Rank	Server	#[a]	Rank	Top-1	Rank	Top-5	Rank	Top-10
1	HDOCK	20	3	7/4***/3**	1	10/5***/5**	1	12/5***/6**
2	SWARMDOCK	20	1	9/3***/5**	2	9/5***/4**	2	9/5***/4**
3	CLUSPRO	20	3	10/8**	3	12/10**	3	12/10**
4	LZERD	20	2	8/3***/5**	4	9/3***/6**	4	9/3***/6**
5	MDOCKPP	20	5	10/1***/4**	5	11/1***/5**	5	11/2***/4**
	HADDOCK	19	5	8/3***/2**	5	9/3***/3**	6	9/3***/3**
7	GALAXYPPDOCK	17	7	6/1***/4**	7	7/3***/2**	7	8/3***/2**
8	HAWKDOCK	7	8	1**	8	2/1**	8	2/1***
Rank	Scorers[b]	#[a]	Rank	Top-1	Rank	Top-5	Rank	Top-10
1	Fernandez-Recio	19	10	7/2***/5**	1	12/5***/6**	1	12/5***/6**
2	Oliva	19	1	11/4***/7**	2	12/4***/7**	1	12/5***/6**
3	Zou	19	4	10/8**	3	12/2***/9**	3	13/3***/8**
	MDOCKPP	19	10	8/1***/6**	3	13/2***/8**	3	13/3***/8**
5	Chang	19	2	10/1***/8**	5	11/2***/9**	7	12/3***/8**
	HDOCK	19	4	9**	5	12/1***/10**	7	12/3***/8**
7	Venclovas	19	2	10/1***/8**	7	11/2***/8**	3	13/2***/10**
	Kihara	19	4	9/1***/7**	7	11/3***/6**	3	12/5***/5**
Rank	Scorers[b]	#[a]	Rank	Top-1	Rank	Top-5	Rank	Top-10
10	LZERD	19	4	10/8**	10	10/3***/6**	10	11/5***/4**
	Bates	19	12	8/6**	10	11/2***/7**	12	11/2***/8**
12	Bonvin	18	13	8/5**	12	12/1***/7**	11	12/3***/6**
13	Carbone	19	8	8/1***/7**	13	9/3***/5**	12	11/3***/6**

14	Weng	19	14	6/5**	14	9/1***/6**	12	12/1***/9**
15	Seok	17	16	4/1***/3**	15	8/1***/5**	15	10/2***/5**
16	Grudinin	18	15	5**	16	6/1***/5**	16	8/2***/5**
17	HAWKDOCK	13	16	4/1***/3**	17	5/2***/3**	17	6/2***/4**
18	QASDOM	13	18	3**	18	5**	17	7**

Table 10.2.11. Overall 4th CASP-CAPRI performance ranking[145]

Rank	Predictors	Participation	Top-1	Top-5	Score
1	Seok	14	8/2**	9/4**	13
	Venclovas	14	7/2**	8/1***/3**	13
3	Chang	14	7/2**	8/3**	11
	Zou	14	5/3**	8/3**	11
5	Kihara	14	5/3**	7/3**	10
	Pierce	13	6/3**	7/3**	10
7	Huang	14	5/3**	5/3**	8
	Bates, Kozakov/Vajda	14	4/3**	5/3**	8
	Fernandez-Recio	14	3/2**	5/3**	8
11	Shen	14	3/1**	6/1**	7
	Vakser	14	3**	4/3**	7
13	Nakamura	11	2/1**	3/2**	5
14	Liwo	12	2	3	3
	Czaplewski	13	2/1**	2/1**	3
	Grudinin	13	1	1	1
	CASP-only predictors	**Participation**	**Top-1**	**Top-5**	**Score**
	Baker	14	7/4**	8/1***/3**	13
	CoDock	10	5/1**	6/2**	8
	Takeda-Shitaka	14	2/1**	4/1***/1**	7
	Seok-assembly	14	5/1**	5/1**	6
	Kiharalab-assembly	13	3/1**	5/1**	6
	Lamoureux	11	3**	3**	6
	UNRES	13	2	3	3
	DATE	11	1	2/1**	3
	Risoluto	14	2	2	2
	Elofsson	13	1	2	2
	VoroCNN-select	13	1	1**	2
	Ornate-select	10	1	1**	2
	SBROD	11	0	1	1
Rank	**Servers**	**Participation**	**Top-1**	**Top-5**	**Score**
1	MDOCKPP	14	4/2**	7/1***/2**	11
2	LZERD	14	4/2**	6/2**	8
3	GALAXYPPDOCK	14	5/1**	5/1**	6
	SWARMDOCK	14	3/2**	4/2**	6
5	HDOCK, CLUSPRO	14	2/1**	3/1**	4
Rank	**Scorers and scoring servers**	**Participation**	**Top-1**	**Top-5**	**Score**
1	Zou	14	5/3**	10/3**	13
	Chang	14	6/3**	9/4**	13
	MDOCKPP	14	5/3**	9/4**	13
	Takeda-Shitaka	14	5/1***/2**	8/1***/3**	13
5	Shen	14	5/3**	9/3**	12
	LZERD	14	7/1***/2**	8/1***/2**	12
7	Huang	14	5/4**	7/4**	11
8	Oliva	14	6/3**	7/3**	10
	Fernandez-Recio	14	5/2**	7/3**	10
	PYDOCKWEB	14	5/1**	7/3**	10

	Kihara	14	5/1***/1**	7/1***/1**	10
	Bates, SWARMDOCK	14	4/3**	6/1***/2**	10
	HAWKDOCK	10	3/2**	6/1***/2**	10
15	Venclovas	13	6/2**	7/2**	9
	HDOCK	14	5/3**	5/4**	9
17	Grudinin	14	1	5/1**	6
	Bonvin	14	3/2**	4/2**	6

10.3. Appendix 3. Supplementary material for Chapter 7

Table 10.3.1. Res.dat

	PDB ATOM TYPE	AMBER ATOM TYPE	Accessible surface area (ASA)		PDB ATOM TYPE	AMBER ATOM TYPE	Accessible surface area (ASA)
RES HEC				RES HEM			
ATOM	NA	5	6.350062	ATOM	NA	5	6.350062
ATOM	CAA	1	2.073181	ATOM	CAA	1	2.073181
ATOM	CAB	1	9.847612	ATOM	CAB	1	9.847612
ATOM	CAC	1	8.811021	ATOM	CAC	1	8.811021
ATOM	CAD	1	2.073181	ATOM	CAD	1	2.073181
ATOM	NB	5	5.896486	ATOM	NB	5	5.896486
ATOM	CBA	1	0	ATOM	CBA	1	0
ATOM	CBB	2	21.25011	ATOM	CBB	2	21.25011
ATOM	CBC	2	21.76841	ATOM	CBC	2	21.76841
ATOM	CBD	1	0	ATOM	CBD	1	0
ATOM	NC	5	5.896486	ATOM	NC	5	5.896486
ATOM	CGA	1	12.43909	ATOM	CGA	1	12.43909
ATOM	CGD	1	12.95738	ATOM	CGD	1	12.95738
ATOM	ND	5	4.989335	ATOM	ND	5	4.989335
ATOM	CHA	1	6.73784	ATOM	CHA	1	6.73784
ATOM	CHB	1	10.36591	ATOM	CHB	1	10.36591
ATOM	CHC	1	10.36591	ATOM	CHC	1	10.36591
ATOM	CHD	1	10.36591	ATOM	CHD	1	10.36591
ATOM	CMA	1	7.256135	ATOM	CMA	1	7.256135
ATOM	CMB	1	5.182954	ATOM	CMB	1	5.182954
ATOM	CMC	1	5.182954	ATOM	CMC	1	5.182954
ATOM	CMD	1	6.73784	ATOM	CMD	1	6.73784
ATOM	C1A	1	6.219544	ATOM	C1A	1	6.219544
ATOM	O1A	8	42.70625	ATOM	O1A	8	42.70625
ATOM	C1B	1	6.219544	ATOM	C1B	1	6.219544
ATOM	C1C	1	6.73784	ATOM	C1C	1	6.73784
ATOM	C1D	1	6.73784	ATOM	C1D	1	6.73784
ATOM	O1D	8	43.55191	ATOM	O1D	8	43.55191
ATOM	C2A	1	3.628067	ATOM	C2A	1	3.628067
ATOM	O2A	8	36.36373	ATOM	O2A	8	36.36373
ATOM	C2B	1	4.146363	ATOM	C2B	1	4.146363
ATOM	C2C	1	4.664659	ATOM	C2C	1	4.664659
ATOM	O2D	8	36.36373	ATOM	O2D	8	36.36373
ATOM	C2D	1	4.664659	ATOM	C2D	1	4.664659
ATOM	C3A	1	3.628067	ATOM	C3A	1	3.628067
ATOM	C3B	1	5.701249	ATOM	C3B	1	5.701249
ATOM	C3C	1	5.701249	ATOM	C3C	1	5.701249
ATOM	C3D	1	3.628067	ATOM	C3D	1	3.628067
ATOM	C4A	1	6.73784	ATOM	C4A	1	6.73784
ATOM	C4B	1	7.256135	ATOM	C4B	1	7.256135
ATOM	C4C	1	6.73784	ATOM	C4C	1	6.73784
ATOM	C4D	1	6.219544	ATOM	C4D	1	6.219544
RES FE				RES CUA			
ATOM	FE	0	5.919677	ATOM	CU	0	1

Bibliography

1. Rios, A.C. and Y. Tor, *On the Origin of the Canonical Nucleobases: An Assessment of Selection Pressures across Chemical and Early Biological Evolution.* Isr J Chem, 2013. **53**(6-7): p. 469-483.
2. Watson, J.D. and F.H. Crick, *The structure of DNA.* Cold Spring Harb Symp Quant Biol, 1953. **18**: p. 123-31.
3. Berg, J.M., L. Stryer, and J.L. Tymoczko, *Bioquímica*. 2007: Reverté.
4. Dyson, H.J. and P.E. Wright, *Intrinsically unstructured proteins and their functions.* Nat Rev Mol Cell Biol, 2005. **6**(3): p. 197-208.
5. Hadzi, S., et al., *The Thermodynamic Basis of the Fuzzy Interaction of an Intrinsically Disordered Protein.* Angew Chem Int Ed Engl, 2017. **56**(46): p. 14494-14497.
6. Ewing, R.M., et al., *Large-scale mapping of human protein-protein interactions by mass spectrometry.* Mol. Syst. Biol., 2007. **3**: p. 89.
7. Rual, J.-F., et al., *Towards a proteome-scale map of the human protein–protein interaction network.* Nature, 2005. **437**(7062): p. 1173-1178.
8. Venkatesan, K., et al., *An empirical framework for binary interactome mapping.* Nat Methods, 2009. **6**(1): p. 83-90.
9. Bonetta, L., *Interactome under construction.* Nature, 2010. **468**(7325): p. 851-852.
10. Salwinski, L., et al., *The Database of Interacting Proteins: 2004 update.* Nucleic Acids Res, 2004. **32**(Database issue): p. D449-51.
11. Goll, J., et al., *MPIDB: the microbial protein interaction database.* Bioinformatics, 2008. **24**(15): p. 1743-4.
12. Keshava Prasad, T.S., et al., *Human Protein Reference Database--2009 update.* Nucleic Acids Res, 2009. **37**(Database issue): p. D767-72.
13. Isserlin, R., R.A. El-Badrawi, and G.D. Bader, *The Biomolecular Interaction Network Database in PSI-MI 2.5.* Database (Oxford), 2011. **2011**: p. baq037.
14. Licata, L., et al., *MINT, the molecular interaction database: 2012 update.* Nucleic Acids Res, 2012. **40**(Database issue): p. D857-61.
15. Launay, G., et al., *MatrixDB, the extracellular matrix interaction database: updated content, a new navigator and expanded functionalities.* Nucleic Acids Res, 2015. **43**(Database issue): p. D321-7.
16. Szklarczyk, D., et al., *STRING v10: protein-protein interaction networks, integrated over the tree of life.* Nucleic Acids Res, 2015. **43**(Database issue): p. D447-52.
17. Oughtred, R., et al., *The BioGRID interaction database: 2019 update.* Nucleic Acids Res, 2019. **47**(D1): p. D529-D541.
18. Jassal, B., et al., *The reactome pathway knowledgebase.* Nucleic Acids Res, 2020. **48**(D1): p. D498-D503.
19. Orchard, S., et al., *The MIntAct project--IntAct as a common curation platform for 11 molecular interaction databases.* Nucleic Acids Res, 2014. **42**(Database issue): p. D358-63.
20. Smyth, M.S. and J.H. Martin, *x ray crystallography.* Mol Pathol, 2000. **53**(1): p. 8-14.
21. Wüthrich, K., *Protein structure determination in solution by NMR spectroscopy.* J Biol Chem, 1990. **265**(36): p. 22059-62.
22. Kuhlbrandt, W., *Cryo-EM enters a new era.* Elife, 2014. **3**: p. e03678.
23. Callaway, E., *The revolution will not be crystallized: a new method sweeps through structural biology.* Nature, 2015. **525**(7568): p. 172-174.

24. Jones, S. and J.M. Thornton, *Analysis of protein-protein interaction sites using surface patches11Edited by G.Von Heijne.* Journal of Molecular Biology, 1997. **272**(1): p. 121-132.
25. Yan, C., et al., *Characterization of protein-protein interfaces.* Protein J, 2008. **27**(1): p. 59-70.
26. Bahadur, R.P. and M. Zacharias, *The interface of protein-protein complexes: analysis of contacts and prediction of interactions.* Cell Mol Life Sci, 2008. **65**(7-8): p. 1059-72.
27. Tsai, C.-J., et al., *Protein-Protein Interfaces: Architectures and Interactions in Protein-Protein Interfaces and in Protein Cores. Their Similarities and Differences.* Critical Reviews in Biochemistry and Molecular Biology, 1996. **31**(2): p. 127-152.
28. Xu, D., S.L. Lin, and R. Nussinov, *Protein binding versus protein folding: the role of hydrophilic bridges in protein associations.* J Mol Biol, 1997. **265**(1): p. 68-84.
29. Sheinerman, F.B., R. Norel, and B. Honig, *Electrostatic aspects of protein-protein interactions.* Curr Opin Struct Biol, 2000. **10**(2): p. 153-9.
30. Jones, S. and J.M. Thornton, *Principles of protein-protein interactions.* Proc Natl Acad Sci U S A, 1996. **93**(1): p. 13-20.
31. Berman, H.M., et al., *The Protein Data Bank.* Nucleic Acids Research, 2000. **28**(1): p. 235-242.
32. Berger, M.F., et al., *Compact, universal DNA microarrays to comprehensively determine transcription-factor binding site specificities.* Nat Biotechnol, 2006. **24**(11): p. 1429-35.
33. Chai, C., Z. Xie, and E. Grotewold, *SELEX (Systematic Evolution of Ligands by EXponential Enrichment), as a Powerful Tool for Deciphering the Protein–DNA Interaction Space*, in *Plant Transcription Factors: Methods and Protocols*, L. Yuan and S.E. Perry, Editors. 2011, Humana Press: Totowa, NJ. p. 249-258.
34. Rohs, R., et al., *Origins of specificity in protein-DNA recognition.* Annu Rev Biochem, 2010. **79**: p. 233-69.
35. Slattery, M., et al., *Absence of a simple code: how transcription factors read the genome.* Trends Biochem Sci, 2014. **39**(9): p. 381-99.
36. Luscombe, N.M., R.A. Laskowski, and J.M. Thornton, *Amino acid-base interactions: a three-dimensional analysis of protein-DNA interactions at an atomic level.* Nucleic Acids Res, 2001. **29**(13): p. 2860-74.
37. Corona, R.I. and J.T. Guo, *Statistical analysis of structural determinants for protein-DNA-binding specificity.* Proteins, 2016. **84**(8): p. 1147-61.
38. Baker, C.M. and G.H. Grant, *Role of aromatic amino acids in protein-nucleic acid recognition.* Biopolymers, 2007. **85**(5-6): p. 456-70.
39. Farrel, A., J. Murphy, and J.T. Guo, *Structure-based prediction of transcription factor binding specificity using an integrative energy function.* Bioinformatics, 2016. **32**(12): p. i306-i313.
40. Wilson, K.A., J.L. Kellie, and S.D. Wetmore, *DNA-protein pi-interactions in nature: abundance, structure, composition and strength of contacts between aromatic amino acids and DNA nucleobases or deoxyribose sugar.* Nucleic Acids Res, 2014. **42**(10): p. 6726-41.
41. Wintjens, R., et al., *Contribution of cation-pi interactions to the stability of protein-DNA complexes.* J Mol Biol, 2000. **302**(2): p. 395-410.
42. Azad, R.N., et al., *Experimental maps of DNA structure at nucleotide resolution distinguish intrinsic from protein-induced DNA deformations.* Nucleic Acids Res, 2018. **46**(5): p. 2636-2647.
43. Mathelier, A., et al., *DNA Shape Features Improve Transcription Factor Binding Site Predictions In Vivo.* Cell Syst, 2016. **3**(3): p. 278-286 e4.

44. Otwinowski, Z., et al., *Crystal structure of trp represser/operator complex at atomic resolution.* Nature, 1988. **335**(6188): p. 321-329.
45. Rohs, R., et al., *The role of DNA shape in protein-DNA recognition.* Nature, 2009. **461**(7268): p. 1248-53.
46. Shakked, Z., et al., *Determinants of repressor/operator recognition from the structure of the trp operator binding site.* Nature, 1994. **368**(6470): p. 469-473.
47. Travers, A.A., *DNA conformation and protein binding.* Annu Rev Biochem, 1989. **58**: p. 427-52.
48. Gordan, R., et al., *Genomic regions flanking E-box binding sites influence DNA binding specificity of bHLH transcription factors through DNA shape.* Cell Rep, 2013. **3**(4): p. 1093-104.
49. Jolma, A., et al., *DNA-binding specificities of human transcription factors.* Cell, 2013. **152**(1-2): p. 327-39.
50. Rao, S., et al., *Systematic prediction of DNA shape changes due to CpG methylation explains epigenetic effects on protein-DNA binding.* Epigenetics Chromatin, 2018. **11**(1): p. 6.
51. Szilagyi, A. and Y. Zhang, *Template-based structure modeling of protein-protein interactions.* Current opinion in structural biology, 2014. **24**: p. 10-23.
52. Kundrotas, P.J., et al., *Templates are available to model nearly all complexes of structurally characterized proteins.* Proceedings of the National Academy of Sciences, 2012. **109**(24): p. 9438-9441.
53. Negroni, J., R. Mosca, and P. Aloy, *Assessing the applicability of template-based protein docking in the twilight zone.* Structure (London, England : 1993), 2014. **22**(9): p. 1356-62.
54. Sievers, F. and D.G. Higgins, *Clustal Omega, Accurate Alignment of Very Large Numbers of Sequences*, in *Multiple Sequence Alignment Methods*, D.J. Russell, Editor. 2014, Humana Press: Totowa, NJ. p. 105-116.
55. Notredame, C., D.G. Higgins, and J. Heringa, *T-Coffee: A novel method for fast and accurate multiple sequence alignment.* J Mol Biol, 2000. **302**(1): p. 205-17.
56. Edgar, R.C., *MUSCLE: multiple sequence alignment with high accuracy and high throughput.* Nucleic Acids Res, 2004. **32**(5): p. 1792-7.
57. Holm, L., *DALI and the persistence of protein shape.* Protein Sci, 2020. **29**(1): p. 128-140.
58. Russell, R.B. and G.J. Barton, *Multiple protein sequence alignment from tertiary structure comparison: assignment of global and residue confidence levels.* Proteins, 1992. **14**(2): p. 309-23.
59. Zhang, Y. and J. Skolnick, *TM-align: a protein structure alignment algorithm based on the TM-score.* Nucleic Acids Res, 2005. **33**(7): p. 2302-9.
60. Mukherjee, S. and Y. Zhang, *MM-align: a quick algorithm for aligning multiple-chain protein complex structures using iterative dynamic programming.* Nucleic acids research, 2009. **37**(11): p. e83-e83.
61. Maiti, R., et al., *SuperPose: a simple server for sophisticated structural superposition.* Nucleic Acids Res, 2004. **32**(Web Server issue): p. W590-4.
62. Gibrat, J.-F., T. Madej, and S.H. Bryant, *Surprising similarities in structure comparison.* Current Opinion in Structural Biology, 1996. **6**(3): p. 377-385.
63. Wang, S., et al., *RaptorX-Property: a web server for protein structure property prediction.* Nucleic Acids Res, 2016. **44**(W1): p. W430-5.
64. Jumper, J., et al., *Applying and improving AlphaFold at CASP14.* Proteins, 2021. **89**(12): p. 1711-1721.

65. Baek, M., et al., *Accurate prediction of protein structures and interactions using a three-track neural network.* Science, 2021. **373**(6557): p. 871-876.
66. Gabb, H.a., R.M. Jackson, and M.J. Sternberg, *Modelling protein docking using shape complementarity, electrostatics and biochemical information.* Journal of molecular biology, 1997. **272**(1): p. 106-120.
67. Chen, R., L. Li, and Z. Weng, *ZDOCK: an initial-stage protein-docking algorithm.* Proteins, 2003. **52**(1): p. 80-7.
68. Redington, P.K., *MOLFIT: A computer program for molecular superposition.* Computers & Chemistry, 1992. **16**(3): p. 217-222.
69. Ritchie, D.W. and G.J.L. Kemp, *Protein docking using spherical polar Fourier correlations.* Proteins: Structure, Function and Genetics, 2000. **39**(2): p. 178-194.
70. Garzon, J.I., et al., *FRODOCK: a new approach for fast rotational protein-protein docking.* Bioinformatics, 2009. **25**(19): p. 2544-51.
71. Schneidman-Duhovny, D., et al., *PatchDock and SymmDock: Servers for rigid and symmetric docking.* Nucleic Acids Research, 2005. **33**(SUPPL. 2): p. 363-367.
72. Fernández-Recio, J., R. Abagyan, and M. Totrov, *Improving CAPRI predictions: Optimized desolvation for rigid-body docking.* Proteins: Structure, Function and Genetics, 2005. **60**(2): p. 308-313.
73. Fernandez-Recio, J., M. Totrov, and R. Abagyan, *ICM-DISCO docking by global energy optimization with fully flexible side-chains.* Proteins, 2003. **52**(1): p. 113-7.
74. Lyskov, S. and J.J. Gray, *The RosettaDock server for local protein-protein docking.* Nucleic Acids Res, 2008. **36**(Web Server issue): p. W233-8.
75. Dominguez, C., R. Boelens, and A. Bonvin, *HADDOCK: a protein− protein docking approach based on biochemical or biophysical Information.* Journal of the American Chemical Society, 2003(2): p. 1731-1737.
76. Moal, I.H. and P.A. Bates, *SwarmDock and the use of normal modes in protein-protein docking.* Int J Mol Sci, 2010. **11**(10): p. 3623-48.
77. Jimenez-Garcia, B., et al., *LightDock: a new multi-scale approach to protein-protein docking.* Bioinformatics, 2018. **34**(1): p. 49-55.
78. Moaly, I.H., et al., *CCharPPI web server: computational characterization of protein-protein interactions from structure.* Bioinformatics, 2014. **31**(1): p. 123-125.
79. Cheng, T.M.-K., T.L. Blundell, and J. Fernandez-Recio, *pyDock: Electrostatics and desolvation for effective scoring of rigid-body protein–protein docking.* Proteins: Structure, Function, and Bioinformatics, 2007. **68**(2): p. 503-515.
80. Pierce, B. and Z. Weng, *ZRANK: reranking protein docking predictions with an optimized energy function.* Proteins, 2007. **67**(4): p. 1078-86.
81. Pons, C., et al., *Scoring by intermolecular pairwise propensities of exposed residues (SIPPER): a new efficient potential for protein-protein docking.* J Chem Inf Model, 2011. **51**(2): p. 370-7.
82. Jiménez-García, B., C. Pons, and J. Fernández-Recio, *pyDockWEB: A web server for rigid-body protein-protein docking using electrostatics and desolvation scoring.* Bioinformatics, 2013. **29**(13): p. 1698-1699.
83. Andrusier, N., R. Nussinov, and H.J. Wolfson, *FireDock: fast interaction refinement in molecular docking.* Proteins, 2007. **69**(1): p. 139-59.
84. Gray, J.J., et al., *Protein–Protein Docking with Simultaneous Optimization of Rigid-body Displacement and Side-chain Conformations.* Journal of Molecular Biology, 2003. **331**(1): p. 281-299.

85. Porter, K.A., et al., *What method to use for protein-protein docking?* Curr Opin Struct Biol, 2019. **55**: p. 1-7.
86. Moal, I.H., et al., *IRaPPA: information retrieval based integration of biophysical models for protein assembly selection.* Bioinformatics, 2017. **33**(12): p. 1806-1813.
87. Vreven, T., et al., *Updates to the Integrated Protein-Protein Interaction Benchmarks: Docking Benchmark Version 5 and Affinity Benchmark Version 2.* Journal of Molecular Biology, 2015. **427**(19): p. 3031-3041.
88. Evans, R., et al., *Protein complex prediction with AlphaFold-Multimer*. 2021.
89. Levy, E.D., et al., *3D complex: a structural classification of protein complexes.* PLoS Comput Biol, 2006. **2**(11): p. e155.
90. Kundrotas, P.J., et al., *Dockground: A comprehensive data resource for modeling of protein complexes.* Protein Sci, 2018. **27**(1): p. 172-181.
91. Mosca, R., A. Céol, and P. Aloy, *Interactome3D: adding structural details to protein networks.* Nature Methods, 2013. **10**(1): p. 47-53.
92. Tovchigrechko, A. and I.A. Vakser, *GRAMM-X public web server for protein-protein docking.* Nucleic Acids Research, 2006. **34**(WEB. SERV. ISS.): p. 310-314.
93. Macindoe, G., et al., *HexServer: An FFT-based protein docking server powered by graphics processors.* Nucleic Acids Research, 2010. **38**(SUPPL. 2): p. 445-449.
94. Tuszynska, I., et al., *NPDock: A web server for protein-nucleic acid docking.* Nucleic Acids Research, 2015. **43**(W1): p. W425-W430.
95. Yan, Y., et al., *HDOCK: A web server for protein-protein and protein-DNA/RNA docking based on a hybrid strategy.* Nucleic Acids Research, 2017. **45**(W1): p. W365-W373.
96. Comeau, S.R., et al., *ClusPro: A fully automated algorithm for protein-protein docking.* Nucleic Acids Research, 2004. **32**(WEB SERVER ISS.): p. 96-99.
97. Van Zundert, G.C.P., et al., *The HADDOCK2.2 Web Server: User-Friendly Integrative Modeling of Biomolecular Complexes.* Journal of Molecular Biology, 2016. **428**(4): p. 720-725.
98. Janin, J., et al., *CAPRI: A critical assessment of PRedicted interactions.* Proteins: Structure, Function and Genetics, 2003. **52**(1): p. 2-9.
99. Lensink, M.F. and S.J. Wodak, *Docking and scoring protein interactions: CAPRI 2009.* Proteins: Structure, Function and Bioinformatics, 2010. **78**(15): p. 3073-3084.
100. Barik, A., et al., *A protein-RNA docking benchmark (I): Nonredundant cases.* Proteins: Structure, Function and Bioinformatics, 2012. **80**(7): p. 1866-1871.
101. Pérez-Cano, L., B. Jiménez-García, and J. Fernández-Recio, *A protein-RNA docking benchmark (II): Extended set from experimental and homology modeling data.* Proteins: Structure, Function and Bioinformatics, 2012. **80**(7): p. 1872-1882.
102. Huang, S.Y. and X. Zou, *A nonredundant structure dataset for benchmarking protein-RNA computational docking.* Journal of Computational Chemistry, 2013. **34**(4): p. 311-318.
103. Nithin, C., S. Mukherjee, and R.P. Bahadur, *A non-redundant protein-RNA docking benchmark version 2.0.* Proteins, 2017. **85**(2): p. 256-267.
104. van Dijk, M. and A.M.J.J. Bonvin, *A protein-DNA docking benchmark.* Nucleic Acids Research, 2008. **36**(14): p. e88-e88.
105. van Dijk, M. and A.M.J.J. Bonvin, *Pushing the limits of what is achievable in protein-DNA docking: Benchmarking HADDOCK's performance.* Nucleic Acids Research, 2010. **38**(17): p. 5634-5647.
106. Honorato, R.V., J. Roel-Touris, and A. Bonvin, *MARTINI-Based Protein-DNA Coarse-Grained HADDOCKing.* Front Mol Biosci, 2019. **6**: p. 102.

107. Lensink, M.F., et al., *The challenge of modeling protein assemblies: the CASP12-CAPRI experiment.* Proteins, 2018. **86 Suppl 1**: p. 257-273.
108. Grosdidier, S. and J. Fernandez-Recio, *Identification of hot-spot residues in protein-protein interactions by computational docking.* BMC Bioinformatics, 2008. **9**: p. 447.
109. Case, D.A., et al., *AMBER 2017.* 2017.
110. Bonsor, D.A., et al., *Molecular Mimicry Enables Competitive Recruitment by a Natively Disordered Protein.* Journal of the American Chemical Society, 2007. **129**(15): p. 4800-4807.
111. Berman, H., *The Protein Data Bank: a historical perspective.* Acta Crystallographica Section A, 2008. **64**(1): p. 88-95.
112. Westbrook, J.D., et al., *PDBx/mmCIF Ecosystem: Foundational Semantic Tools for Structural Biology.* J Mol Biol, 2022. **434**(11): p. 167599.
113. Homer, R.W., et al., *SYBYL Line Notation (SLN): A Single Notation To Represent Chemical Structures, Queries, Reactions, and Virtual Libraries.* Journal of Chemical Information and Modeling, 2008. **48**(12): p. 2294-2307.
114. van Rossum, G., *Python reference manual*. 1995, CWI.
115. Van Rossum, G. and F.L. Drake, *Python 3 Reference Manual*. 2009: CreateSpace.
116. Wall, L., T. Christiansen, and J. Orwant, *Programming Perl*. 2000: O'Reilly Media.
117. R Development Core Team, *R: A language and environment for statistical computing*. 2010, R Foundation for Statistical Computing.
118. GNU, P., *Free Software Foundation. Bash (3.2. 48)[Unix shell program]*. 2007.
119. Cock, P.J., et al., *Biopython: freely available Python tools for computational molecular biology and bioinformatics.* Bioinformatics, 2009. **25**(11): p. 1422-3.
120. Bakan, A., L.M. Meireles, and I. Bahar, *ProDy: protein dynamics inferred from theory and experiments.* Bioinformatics, 2011. **27**(11): p. 1575-7.
121. Gil, V.A. and V. Guallar, *PyProCT: Automated cluster analysis for structural bioinformatics.* Journal of Chemical Theory and Computation, 2014. **10**(8): p. 3236-3243.
122. Pedregosa, F., et al., *Scikit-learn: Machine Learning in Python.* Journal of Machine Learning Research, 2012. **12**.
123. Team, T., et al., *Theano: A Python framework for fast computation of mathematical expressions.* 2016.
124. Abadi, M., et al., *TensorFlow: A System for Large-Scale Machine Learning*, in *OSDI*. 2016. p. 265--283.
125. Pettersen, E.F., et al., *UCSF Chimera--a visualization system for exploratory research and analysis.* J Comput Chem, 2004. **25**(13): p. 1605-12.
126. Goddard, T.D., et al., *UCSF ChimeraX: Meeting modern challenges in visualization and analysis.* Protein Sci, 2018. **27**(1): p. 14-25.
127. Hanson, R., *Jmol - a paradigm shift in crystallographic visualization.* Journal of Applied Crystallography, 2010. **43**(5 Part 2): p. 1250-1260.
128. Humphrey, W., A. Dalke, and K. Schulten, *VMD: Visual molecular dynamics.* Journal of Molecular Graphics, 1996. **14**(1): p. 33-38.
129. Abagyan, R., et al., *ICM?A new method for protein modeling and design: Applications to docking and structure prediction from the distorted native conformation.* J. Comput. Chem., 1994. **15**(5): p. 488-506.
130. Rose, A.S., et al., *NGL viewer: web-based molecular graphics for large complexes.* Bioinformatics, 2018. **34**(21): p. 3755-3758.
131. Eberhardt, J., et al., *AutoDock Vina 1.2.0: New Docking Methods, Expanded Force Field, and Python Bindings.* J Chem Inf Model, 2021. **61**(8): p. 3891-3898.

132. Hwang, H., et al., *Protein-protein docking benchmark version 4.0.* Proteins, 2010. **78**(15): p. 3111-4.
133. Douguet, D., et al., *DOCKGROUND resource for studying protein-protein interfaces.* Bioinformatics, 2006. **22**(21): p. 2612-2618.
134. Luscombe, N.M., et al., *An overview of the structures of protein-DNA complexes.* Genome biology, 2000. **1**(1): p. REVIEWS001-REVIEWS001.
135. Norambuena, T. and F. Melo, *The Protein-DNA Interface database.* BMC bioinformatics, 2010. **11**: p. 262-262.
136. Krissinel, E. and K. Henrick, *Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions.* Acta Crystallographica Section D: Biological Crystallography, 2004. **60**(12 I): p. 2256-2268.
137. Lu, X.J. and W.K. Olson, *3DNA: A software package for the analysis, rebuilding and visualization of three-dimensional nucleic acid structures.* Nucleic Acids Research, 2003. **31**(17): p. 5108-5121.
138. Lu, X.J. and W.K. Olson, *3DNA: A versatile, integrated software system for the analysis, rebuilding and visualization of three-dimensional nucleic-acid structures.* Nature Protocols, 2008. **3**(7): p. 1213-1227.
139. Janin, J., *Welcome to CAPRI: A Critical Assessment of PRedicted Interactions.* Proteins: Structure, Function, and Bioinformatics, 2002. **47**(3): p. 257-257.
140. Jimenez-Garcia, B., et al., *pyDockSAXS: protein-protein complex structure by SAXS and computational docking.* Nucleic Acids Res, 2015. **43**(W1): p. W356-61.
141. Cheng, T.M., T.L. Blundell, and J. Fernandez-Recio, *Structural assembly of two-domain proteins by rigid-body docking.* BMC Bioinformatics, 2008. **9**: p. 441.
142. Rosell, M., et al., *Integrative modeling of protein-protein interactions with pyDock for the new docking challenges.* Proteins, 2020. **88**(8): p. 999-1008.
143. Lensink, M.F., et al., *Prediction of homoprotein and heteroprotein complexes by protein docking and template-based modeling: A CASP-CAPRI experiment.* Proteins: Structure, Function and Bioinformatics, 2016. **84**(May 2015): p. 323-348.
144. Lensink, M.F., et al., *Blind prediction of homo- and hetero-protein complexes: The CASP13-CAPRI experiment.* Proteins, 2019. **87**(12): p. 1200-1221.
145. Lensink, M.F., et al., *Prediction of protein assemblies, the next frontier: The CASP14-CAPRI experiment.* Proteins, 2021. **89**(12): p. 1800-1823.
146. *Protein Structure Prediction*. 4 ed. Methods in Molecular Biology. 2020: Humana New York, NY. XII, 358.
147. Xu, Q. and R.L. Dunbrack, Jr., *ProtCID: a data resource for structural information on protein interactions.* Nat Commun, 2020. **11**(1): p. 711.
148. Dey, S., J. Prilusky, and E.D. Levy, *QSalignWeb: A Server to Predict and Analyze Protein Quaternary Structure.* Front Mol Biosci, 2021. **8**: p. 787510.
149. Jammer, T., C. Iwainsky, and C. Bischof. *Automatic Detection of MPI Assertions*. 2020. Cham: Springer International Publishing.
150. Yoo, A.B., M.A. Jette, and M. Grondona. *SLURM: Simple Linux Utility for Resource Management*. 2003. Berlin, Heidelberg: Springer Berlin Heidelberg.
151. Cornell, W.D., et al., *A Second Generation Force Field for the Simulation of Proteins, Nucleic Acids, and Organic Molecules.* Journal of the American Chemical Society, 1995. **117**(19): p. 5179-5197.
152. Fernández-Recio, J., M. Totrov, and R. Abagyan, *Identification of protein-protein interaction sites from docking energy landscapes.* Journal of Molecular Biology, 2004. **335**(3): p. 843-865.

153. Theodoridis, S. and K. Koutroumbas, *Chapter 11 - Clustering: Basic Concepts*, in *Pattern Recognition (Fourth Edition)*, S. Theodoridis and K. Koutroumbas, Editors. 2009, Academic Press: Boston. p. 595-625.
154. Bower, M.J., F.E. Cohen, and R.L. Dunbrack, *Prediction of protein side-chain rotamers from a backbone-dependent rotamer library: A new homology modeling tool.* Journal of Molecular Biology, 1997. **267**(5): p. 1268-1282.
155. Grosdidier, S., et al., *Prediction and scoring of docking poses with pyDock.* Proteins, 2007. **69**(4): p. 852-8.
156. Daura, X., et al., *Peptide Folding: When Simulation Meets Experiment.* Angewandte Chemie International Edition, 1999. **38**(1/2): p. 236-240.
157. Rodríguez-Lumbreras, L.A., et al., *pyDockDNA: A new web server for energy-based protein-DNA docking and scoring.* Frontiers in Molecular Biosciences, 2022. **9**.
158. Sukumar, S., et al., *DBSI server: DNA binding site identifier.* Bioinformatics, 2016. **32**(18): p. 2853-5.
159. Chelliah, V., T.L. Blundell, and J. Fernandez-Recio, *Efficient restraints for protein-protein docking by comparison of observed amino acid substitution patterns with those predicted from local environment.* J Mol Biol, 2006. **357**(5): p. 1669-82.
160. Pallara, C., et al., *Conformational heterogeneity of unbound proteins enhances recognition in protein-protein encounters.* Journal of Chemical Theory and Computation, 2016. **12**(7): p. 3236-3249.
161. Kagawa, T.F., et al., *Quantitative Analysis of DNA Secondary Structure from Solvent-Accessible Surfaces: The B- to Z-DNA Transition as a Model.* Biochemistry, 1989. **28**(16): p. 6642-6651.
162. Pérez-Cano, L., M. Romero-Durana, and J. Fernández-Recio, *Structural and energy determinants in protein-RNA docking.* Methods, 2016: p. 1-8.
163. Launay, G. and T. Simonson, *Homology modelling of protein-protein complexes: a simple method and its possibilities and limitations.* BMC Bioinformatics, 2008. **9**: p. 427.
164. Kundrotas, P.J., M.F. Lensink, and A. Emil, *Homology-based modeling of 3D structures of protein–protein complexes using alignments of modified sequence profiles.* Int. J. Biol. Macromol., 2008. **43**(2): p. 198-208.
165. Lu, L., H. Lu, and J. Skolnick, *MULTIPROSPECTOR: an algorithm for the prediction of protein-protein interactions by multimeric threading.* Proteins, 2002. **49**(3): p. 350-364.
166. Aloy, P., et al., *Structure-based assembly of protein complexes in yeast.* Science, 2004. **303**(5666): p. 2026-9.
167. Sinha, R., et al., *Docking by structural similarity at protein-protein interfaces.* Proteins: Struct. Funct. Bioinf., 2010. **78**(15): p. 3235-3241.
168. Tuncbag, N., et al., *Predicting protein-protein interactions on a proteome scale by matching evolutionary and structural similarities at interfaces using PRISM.* Nat. Protoc., 2011. **6**(9): p. 1341-1354.
169. Aloy, P. and R.B. Russell, *Structural systems biology: modelling protein interactions.* Nat. Rev. Mol. Cell Biol., 2006. **7**(3): p. 188-197.
170. Mosca, R., A. Céol, and P. Aloy, *Interactome3D: adding structural details to protein networks.* Nat. Methods, 2013. **10**(1): p. 47-53.
171. Rose, P.W., et al., *The RCSB Protein Data Bank: new resources for research and education.* Nucleic Acids Res., 2012. **41**(D1): p. D475-D482.
172. Negroni, J., et al., *Assessing the Applicability of Template-Based Protein Docking in the Twilight Zone.* Structure, 2014. **22**(9): p. 1356-1362.

173. Gray, J.J., et al., *Protein-protein docking with simultaneous optimization of rigid-body displacement and side-chain conformations.* J. Mol. Biol., 2003. **331**(1): p. 281-299.
174. Dominguez, C., et al., *HADDOCK: A Protein−Protein Docking Approach Based on Biochemical or Biophysical Information.* J. Am. Chem. Soc., 2003. **125**(7): p. 1731-1737.
175. de Vries, S. and Z. Martin, *Flexible docking and refinement with a coarse-grained protein model using ATTRACT.* Proteins: Struct. Funct. Bioinf., 2013. **81**(12): p. 2167-2174.
176. Moal, I.H. and P.A. Bates, *SwarmDock and the use of normal modes in protein-protein docking.* Int. J. Mol. Sci., 2010. **11**(10): p. 3623-3648.
177. Pierce, B., P. Brian, and W. Zhiping, *ZRANK: Reranking protein docking predictions with an optimized energy function.* Proteins: Struct. Funct. Bioinf., 2007. **67**(4): p. 1078-1086.
178. Pons, C., et al., *Scoring by intermolecular pairwise propensities of exposed residues (SIPPER): a new efficient potential for protein-protein docking.* J. Chem. Inf. Model., 2011. **51**(2): p. 370-377.
179. Ravikant, D.V.S. and R. Elber, *PIE-efficient filters and coarse grained potentials for unbound protein-protein docking.* Proteins, 2010. **78**(2): p. 400-419.
180. Lensink, M.F., et al., *Prediction of homo- and hetero-protein complexes by protein docking and template-based modeling: a CASP-CAPRI experiment.* Proteins: Struct. Funct. Bioinf., 2016. **84 Suppl 1**: p. 323-48.
181. Douguet, D., et al., *Dockground resource for studying protein-protein interfaces.* Bioinformatics, 2006. **22**(21): p. 2612-2618.
182. Pearson, W.R. and D.J. Lipman, *Improved tools for biological sequence comparison.* Proc. Natl. Acad. Sci. U. S. A., 1988. **85**(8): p. 2444-2448.
183. Zhang, Y., Z. Yang, and S. Jeffrey, *Scoring function for automated assessment of protein structure template quality.* Proteins: Struct. Funct. Bioinf., 2004. **57**(4): p. 702-710.
184. Cheng, T.M.-K., T.L. Blundell, and J. Fernandez-Recio, *pyDock: electrostatics and desolvation for effective scoring of rigid-body protein-protein docking.* Proteins, 2007. **68**(2): p. 503-515.
185. Sinha, R., P.J. Kundrotas, and I.A. Vakser, *Protein docking by the interface structure similarity: How much structure is needed?* PLoS ONE, 2012. **7**(2): p. e31349-e31349.
186. Šali, A. and T.L. Blundell, *Comparative Protein Modelling by Satisfaction of Spatial Restraints.* Journal of Molecular Biology, 1993. **234**(3): p. 779-815.
187. Altschul, S.F., et al., *Basic local alignment search tool.* Journal of Molecular Biology, 1990. **215**(3): p. 403-410.
188. Shen, M.Y. and A. Sali, *Statistical potential for assessment and prediction of protein structures.* Protein Sci, 2006. **15**(11): p. 2507-24.
189. Roy, A., A. Kucukural, and Y. Zhang, *I-TASSER: a unified platform for automated protein structure and function prediction.* Nat Protoc, 2010. **5**(4): p. 725-38.
190. Canutescu, A.a., A.a. Shelenkov, and R.L. Dunbrack, *A graph-theory algorithm for rapid protein side-chain prediction.* Protein science : a publication of the Protein Society, 2003. **12**(9): p. 2001-2014.
191. Liu, S., et al., *A physical reference state unifies the structure-derived potential of mean force for protein folding and binding.* Proteins, 2004. **56**(1): p. 93-101.
192. Ruiz-Carmona, S., et al., *rDock: A Fast, Versatile and Open Source Program for Docking Ligands to Proteins and Nucleic Acids.* PLoS Computational Biology, 2014. **10**(4): p. 1-7.
193. Ekiert, D.C., et al., *Architectures of Lipid Transport Systems for the Bacterial Outer Membrane.* Cell, 2017. **169**(2): p. 273-285 e17.
194. Bliven, S., et al., *Automated evaluation of quaternary structures from protein crystals.* PLoS Comput Biol, 2018. **14**(4): p. e1006104.

195. Daniel Soler, Y.W.R.S., *Extensive benchmark of rDock as a peptide-protein docking tool.* Journal of Computer-Aided Molecular Design, 2019(January).
196. Desmet, J., et al., *Structural basis of IL-23 antagonism by an Alphabody protein scaffold.* Nat Commun, 2014. **5**: p. 5237.
197. Kamishikiryo, J., et al., *Molecular basis for LLT1 protein recognition by human CD161 protein (NKRP1A/KLRB1).* J Biol Chem, 2011. **286**(27): p. 23823-30.
198. Kita, S., et al., *Crystal structure of extracellular domain of human lectin-like transcript 1 (LLT1), the ligand for natural killer receptor-P1A.* Eur J Immunol, 2015. **45**(6): p. 1605-13.
199. Javaheri, A., et al., *Helicobacter pylori adhesin HopQ engages in a virulence-enhancing interaction with human CEACAMs.* Nat Microbiol, 2016. **2**: p. 16189.
200. Pons, C., et al., *Optimization of pyDock for the new CAPRI challenges: Docking of homology-based models, domain-domain assembly and protein-RNA binding.* Proteins, 2010. **78**(15): p. 3182-8.
201. Tian, C., et al., *ff19SB: Amino-Acid-Specific Protein Backbone Parameters Trained against Quantum Mechanics Energy Surfaces in Solution.* J Chem Theory Comput, 2020. **16**(1): p. 528-552.
202. Cheatham, T.E., 3rd, P. Cieplak, and P.A. Kollman, *A modified version of the Cornell et al. force field with improved sugar pucker phases and helical repeat.* J Biomol Struct Dyn, 1999. **16**(4): p. 845-62.
203. Jakalian, A., D.B. Jack, and C.I. Bayly, *Fast, efficient generation of high-quality atomic charges. AM1-BCC model: II. Parameterization and validation.* Journal of Computational Chemistry, 2002. **23**(16): p. 1623-1641.
204. Lensink, M.F., et al., *Modeling protein-protein, protein-peptide, and protein-oligosaccharide complexes: CAPRI 7th edition.* Proteins, 2020. **88**(8): p. 916-938.
205. Fiser, A. and A.B.T.M.i.E. Šali, *Modeller: Generation and Refinement of Homology-Based Protein Structure Models*. 2003, Academic Press. p. 461-491.
206. McPherson, A., *Protein Crystallization*, in *Protein Crystallography: Methods and Protocols*, A. Wlodawer, Z. Dauter, and M. Jaskolski, Editors. 2017, Springer New York: New York, NY. p. 17-50.
207. Elez, K., A.M.J.J. Bonvin, and A. Vangone, *Biological vs. Crystallographic Protein Interfaces: An Overview of Computational Approaches for Their Classification.* Crystals, 2020. **10**(2).
208. Mitra, P. and D. Pal, *Combining Bayes classification and point group symmetry under Boolean framework for enhanced protein quaternary structure inference.* Structure, 2011. **19**(3): p. 304-12.
209. Jimenez-Garcia, B., et al., *PRODIGY-crystal: a web-tool for classification of biological interfaces in protein complexes.* Bioinformatics, 2019. **35**(22): p. 4821-4823.
210. Ashkenazy, H., et al., *ConSurf 2016: an improved methodology to estimate and visualize evolutionary conservation in macromolecules.* Nucleic Acids Res, 2016. **44**(W1): p. W344-50.
211. Mitra, P. and D. Pal, *New measures for estimating surface complementarity and packing at protein-protein interfaces.* FEBS Lett, 2010. **584**(6): p. 1163-8.
212. Brooks, B.R., et al., *CHARMM: the biomolecular simulation program.* J Comput Chem, 2009. **30**(10): p. 1545-614.
213. Eswar, N., et al., *Comparative Protein Structure Modeling Using MODELLER.* Current Protocols in Protein Science, 2007. **50**(1): p. 2.9.1-2.9.31.
214. Sievers, F., et al., *Fast, scalable generation of high-quality protein multiple sequence alignments using Clustal Omega.* Mol Syst Biol, 2011. **7**: p. 539.

215. Abagyan, R., et al., *Disseminating structural genomics data to the public: from a data dump to an animated story.* Trends Biochem Sci, 2006. **31**(2): p. 76-8.
216. Bernal-Bayard, P., et al., *Interaction of photosystem I from Phaeodactylum tricornutum with plastocyanins as compared with its native cytochrome c6: Reunion with a lost donor.* Biochim Biophys Acta, 2015. **1847**(12): p. 1549-59.
217. Akazaki, H., et al., *Crystallization and structural analysis of cytochrome c(6) from the diatom Phaeodactylum tricornutum at 1.5 A resolution.* Biosci Biotechnol Biochem, 2009. **73**(1): p. 189-91.
218. Haddadian, E.J. and E.L. Gross, *Brownian dynamics study of cytochrome f interactions with cytochrome c6 and plastocyanin in Chlamydomonas reinhardtii plastocyanin, and cytochrome c6 mutants.* Biophys J, 2005. **88**(3): p. 2323-39.
219. Shahrokh, K., et al., *Quantum mechanically derived AMBER-compatible heme parameters for various states of the cytochrome P450 catalytic cycle.* J Comput Chem, 2012. **33**(2): p. 119-33.
220. Haddadian, E.J. and E.L. Gross, *A Brownian dynamics study of the effects of cytochrome f structure and deletion of its small domain in interactions with cytochrome c6 and plastocyanin in Chlamydomonas reinhardtii.* Biophys J, 2006. **90**(2): p. 566-77.
221. Crowley, P.B., et al., *Hydrophobic Interactions in a Cyanobacterial Plastocyanin−Cytochrome f Complex.* Journal of the American Chemical Society, 2001. **123**(43): p. 10444-10453.
222. Cruz-Gallardo, I., et al., *The cytochrome f-plastocyanin complex as a model to study transient interactions between redox proteins.* FEBS Lett, 2012. **586**(5): p. 646-52.
223. Castell, C., et al., *New Insights into the Evolution of the Electron Transfer from Cytochrome f to Photosystem I in the Green and Red Branches of Photosynthetic Eukaryotes.* Plant Cell Physiol, 2021. **62**(7): p. 1082-1093.