Alma Mater Studiorum – Università di Bologna

DOTTORATO DI RICERCA IN

NANOSCIENZE PER LA MEDICINA E PER L'AMBIENTE

Ciclo XXXIII

Settore Concorsuale: 03/C1

Settore Scientifico Disciplinare: CHIM/06

DEVELOPMENT OF REVERSE DOCKING PROTOCOLS FOR VIRTUAL SCREENING IN NANOMEDICINE

Presentata da: Dr. Fabio Bologna

Coordinatore Dottorato Prof. Dario Braga

Supervisore Prof. Matteo Calvaresi

Esame finale anno 2021

Abstract

Computational chemistry within the pharmaceutical industry plays a role in many aspects of drug design, from target selection to lead identification and optimization[1,2] which resulted in the birth of a new branch of pharmaceutical research: *in silico drug design*. One of the first *in silico drug design* tools has been *molecular docking* which infers the interaction between two molecules, usually a small active compound and a receptor, on the basis of their 3D structures.

During my three years of PhD studies I worked mainly on repurposing molecular docking tools to investigate the interactions between a single molecule of interest and a collection of proteins with potential therapeutical applications. Since this process is essentially the inverse of what is usually done in pharmaceutical sciences, the technique is called *reverse screening*. Reverse screening can help in the identification of new therapeutical targets, predicting toxicological effects and unwanted interactions or the design of new therapeutic platforms based on the conjugation between a synthetic compound and a protein carrier.

In this work, reverse screening has been applied to porphine and phthalocyanine, two chemically related photosensitizers employed in photodynamic therapy, and Gd@C82, the most promising endohedral gadofullerene for theranostic applications. Photodynamic therapy (PDT) is a non-invasive treatment for various types of tumors that revolves around photosensitizers (PS), molecules that in their excited states are can produce cytotoxic agents through photochemical reactions while being harmless in their ground state[3]. Even though porphine and phthalocyanine have been used with great success as photosensitizers[4,5], poor solubility, non-specific cellular and tissue uptake and aggregation phenomena, that quench photodynamic properties, hamper their application. A solution to these shortcomings has been the conjugation of PS with carrier systems such as proteins, which have already been explored as drug-carriers[6]. A reverse docking protocol based entirely on the rigid-body docking software PatchDock[7] has been applied to screen a large structural database of proteins with potential therapeutical application[8] against porphine and phthalocyanine. PatchDock generates docking solutions based on shape complementarity alone, which is also strongly taken into consideration in the scoring

phase. As result, it is very accurate in predicting the binding of highly hydrophobic molecules since their driving force to the binding are van der Waals interactions, which scales with area of contact between the interacting molecules. The application of the protocol resulted in the identifications of possible targets for photodynamic therapy and carrier systems.

Gadofullerenes, i.e., fullerenes that traps a Gd atom inside their carbon cage, are characterized by substantial electron transfer from the encaged metal atom to the carbon cage, a phenomenon known as 'intrafullerene electron transfer'[9,10]. As result, they can act as efficient contrast agents for MRI, inducing up to 20 times stronger proton relaxivity than commercially available contrast agents while avoiding completely metal leakage in vivo[11]; in addition, they share photodynamic and photothermal/photoacoustic properties with their pristine counterparts[12,13]. Their application in clinical setting as theranostic agents is severely held back by poor water solubility, low biocompatibility, aggregation and non-specific cellular uptake. These shortcomings can be overcome by using protein carriers. Although a reverse screening protocol based on PatchDock was successful in identifying the correct binding site of C_{60} fullerene on lysozyme[14,15], study on the thermodynamics of binding of Gd@C_{60} on lysozyme discovered that the partial charges of the carbon cage, which derive from the intrafullerene electron transfer, play a key role in the determination of the binding site[16]. To properly take into consideration electrostatic contributions, a new protocol was designed which combines PatchDock, for docking candidates' generation, with binding site structure optimization, with Molecular Mechanics energy minimization using the Free and Open Source Software (FOSS) AmberTools16[17], and scoring with the MM-GBSA method[18]. The protocol was used to identify possible carrier systems, imaging and therapeutic targets as well as novel solutions for cancer immunoassays.

Finally, the application of reverse screening protocols for the investigation of the interactions between nanoparticles and biomolecules, a current hot-topic in research, was explored. Although theoretically possible, the sheer number of atoms that must be considered and the nanoparticle size result in a plethora of problems since docking tools have been designed with small active molecules in mind. The same applies in the case of force-field based tools as well since they have been designed to handle large molecular entities only in the case of biomolecules, such as proteins and nucleic

acids. Nevertheless, by repurposing protein-protein docking tools and combining them with force-field bases scoring functions it has been possible to investigate the interactions between gold, silver and silica nanoparticles and proteins. Although this initial version of the protocol carried many shortcomings, a new protocol has been devised to overcome these limitations and investigate 2D materials-biomolecule interactions. The new protocol uses ZDOCK 2.1, a shape-complementarity based protein-protein docking tool, which reward large continuous areas of contact between the two interacting partners, to generate the docking candidates. Using a combination of Bash tools and utilities from GROMACS 5.1[19] clustering analysis is performed to identify only the most significant docking solution and ease the computational cost. Finally the scoring phase is performed either at the MM-GBSA level using the MM-PBSA.py[20] Python script from AmberTools16[17] or at the more accurate, yet computationally expensive, MM-PBSA level using *g_mmpbsa*, a FOSS tool that combines GROMACS[19] subroutines and the FOSS solver of Poisson Boltzmann equations *apbs*[21]. This protocol will be applied in the next future to investigate the interactions between graphene and proteins.

References

[1] A. Hillisch, N. Heinrich, H. Wild, *ChemMedChem* **2015**, *10*, 1958–1962.

[2] C. J. Manly, S. Louise-May, J. D. Hammer, *Drug Discov Today* **2001**, *6*, 1101–1110.

[3] D. E. J. G. J. Dolmans, D. Fukumura, R. K. Jain, *Nature Reviews Cancer* **2003**, *3*, 380–387.

[4] L. M. Moreira, F. V. dos Santos, J. P. Lyon, M. Maftoum-Costa, C. Pacheco-Soares, N. S. da Silva, *Aust. J. Chem.* **2008**, *61*, 741–754.

[5] L. B. Josefsen, R. W. Boyle, *Theranostics* **2012**, *2*, 916–966.

[6] F. A. de Wolf, G. M. Brett, *Pharmacol Rev* **2000**, *52*, 207–236.

[7] D. Duhovny, R. Nussinov, H. J. Wolfson, in *Algorithms in Bioinformatics* (Eds.: R. Guigó, D. Gusfield), Springer, Berlin, Heidelberg, **2002**, pp. 185–200.

[8] Z. Gao, H. Li, H. Zhang, X. Liu, L. Kang, X. Luo, W. Zhu, K. Chen, X. Wang, H. Jiang, *BMC Bioinformatics* **2008**, *9*, 104.

[9] H. Shinohara, *Reports on Progress in Physics* **2000**, *63*, 843–892.

[10] R. D. Johnson, M. S. de Vries, J. Salem, D. S. Bethune, C. S. Yannoni, *Nature* **1992**, *355*, 239–240.

[11] H. Kato, Y. Kanazawa, M. Okumura, A. Taninaka, T. Yokawa, H. Shinohara, *J. Am. Chem. Soc.* **2003**, *125*, 4391–4397.

[12] Z. Chen, L. Ma, Y. Liu, C. Chen, *Theranostics* **2012**, *2*, 238–250.

[13]R. Bakry, R. M. Vallant, M. Najam-ul-Haq, M. Rainer, Z. Szabo, C. W. Huck, G. K. Bonn, *Int J Nanomedicine* **2007**, *2*, 639–649.

[14]M. Calvaresi, F. Zerbetto, *ACS Nano* **2010**, *4*, 2283–2299.

[15]M. Calvaresi, F. Arnesano, S. Bonacchi, A. Bottoni, V. Calò, S. Conte, G. Falini, S. Fermani, M. Losacco, M. Montalti, G. Natile, L. Prodi, F. Sparla, F. Zerbetto, *ACS Nano* **2014**, *8*, 1871–1877.

[16]F. Bologna, E. J. Mattioli, A. Bottoni, F. Zerbetto, M. Calvaresi, *ACS Omega* **2018**, *3*, 13782–13789.

[17]D. A. Case, R. M. Betz, D. S. Cerutti, T. E. C. III, T. A. Darden, R. E. Duke, T. J. Giese, H. Gohlke, A. W. Goetz, N. Homeyer, S. Izadi, P. Janowski, J. Kaus, A. Kovalenko, T. S. Lee, S. LeGrand, P. Li, C. Lin, T. Luchko, R. Luo, B. Madej, D. Mermelstein, K. M. Merz, G. Monard, H. Nguyen, H. T. Nguyen, I. Omelyan, A. Onufriev, D. R. Roe, A. Roitberg, C. Sagui, C. L. Simmerling, W. M. Botello-Smith, J. Swails, R. C. Walker, J. Wang, R. M. Wolf, X. Wu, L. Xiao, P. A. Kollman, *AMBER 16 and AmberTools16*, University Of California, San Francisco, **2016**.

[18]J. Srinivasan, T. E. Cheatham, P. Cieplak, P. A. Kollman, D. A. Case, *Journal of the American Chemical Society* **1998**, *120*, 9401–9409.

[19]M. J. Abraham, D. van der Spoel, E. Lindahl, B. Hess, the GROMACS development team, *GROMACS 5.1.5*, **2017**.

[20]B. R. Miller, T. D. McGee, J. M. Swails, N. Homeyer, H. Gohlke, A. E. Roitberg, *J. Chem. Theory Comput.* **2012**, *8*, 3314–3321.

[21]E. Jurrus, D. Engel, K. Star, K. Monson, J. Brandi, L. E. Felberg, D. H. Brookes, L. Wilson, J. Chen, K. Liles, M. Chun, P. Li, D. W. Gohara, T. Dolinsky, R. Konecny, D. R. Koes, J. E. Nielsen, T. Head-Gordon, W. Geng, R. Krasny, G.-W. Wei, M. J. Holst, J. A. McCammon, N. A. Baker, *Protein Science* **2018**, *27*, 112–128.

1. Introduction

Every attempt to employ mathematical methods in the study of chemical questions must be considered profoundly irrational and contrary to the spirit of chemistry. If mathematical analysis should ever hold a prominent place in chemistry - an aberration which is happily almost impossible - it would occaision a rapid and widespread degeneration of that science. - A. Compte, Philosophie Positive, 1830

Yet, here we are! Computational chemistry, one of the most recent branches of this science, has been developing at a staggering rate in the last decades thanks to the parallel development of computational hardware and today stands beside theory and experiment as one of the pillars of research in chemistry. Nowadays, computational chemistry contributes to many fields of science, from designing novel materials to exploring the possible chemistry reactions happening in the vacuum of space. In particular, computational chemistry within the pharmaceutical industry plays a role in many aspects of drug design, from target selection to lead identification and optimization[1,2] which resulted in the birth of a new branch of pharmaceutical research: *in silico drug design*, i.e., designing novel therapeutic agents by describing molecular entities from a mathematical point of view and computing their behavior on the basis of the laws of physics. One of the first *in silico drug design* tools has been *molecular docking*, Molecular docking infers the interaction between two molecules on the basis of their 3D structures. It is used in the pharmaceutical industry to investigate the interactions between a collection of small active molecules and a biomolecule involved in biological processes of interest from a therapeutical point of view, such a membrane receptor involved in a particular disease.

During my three years of PhD studies I worked mainly on repurposing molecular docking tools to investigate the interactions between a single molecule of interest and a collection of proteins with potential therapeutical applications. Since this process is essentially the inverse of what is usually done in pharmaceutical sciences, the technique is called *reverse screening*.

Docking itself has several shortcomings which derive from the strong approximations that it is necessary to employ. However, the choice of molecules that

are investigated plays a significant role in the impact of these approximations on the accuracy of the results and by choosing rigid, hydrophobic molecules it is possible to predict interactions that are later confirmed by experimental evidence[3–5].

The thesis starts with a brief chapter on the theory behind the computational methods that are employed in this work. First, the theory behind Molecular Mechanics (MM) techniques will be analyzed; as the reader will see, MM has been extensively employed in this work as an accurate and flexible way to gauge the interactions between biomolecules and molecules of interest. Follows a quick introduction on quantum mechanics techniques, since they have been used to obtain certain properties of the systems that will be presented. Finally, a general introduction on molecular docking is presented, followed by a more in depth analysis of the docking tools that have been applied in this work.

Each chapter that follows is organized as a small self-contained scientific paper in itself and represents a significant step in my journey of applying this technique, from the most basic form to new and innovative ways to employ this tool in research. In each chapter, a small introduction on the molecule that is investigated, and how the application of reverse screening protocols can aid the research effort, is followed by the type of reverse screening protocol that has been applied and by an analysis of the results. The latter are scoreboards of the interaction between a collection of biomolecules and the molecule of interest and can offer a glimpse of possible future applications of the latter. The most interesting potential applications are discussed with a brief analysis of the literature regarding the biomolecule involved. Each chapter is followed by its own references.

In Chapter 3 , it will be analyzed the most simple application of the protocol, that is using a docking tool without any combination with other computational chemistry tools, as it was in the case of two drug-like small molecules, porphine and phthalocyanine. Both molecules are widely used as photosensitizers agents in photodynamic therapy, but suffer from poor pharmacokinetics properties and non-specific tissue and cellular uptake, which is especially essential in photodynamic therapy. Given their high hydrophobic character and structural rigidity, a simple rigid-body docking tool in the form of PatchDock[6] is capable of accurately investigate their interactions with biomolecules.

Chapter 4 will introduce the reader to a more 'exotic' molecule of interest, Gd@C82, the most promising member of the gadofullerene family from a therapeutical point of view. We demonstrate that in this case the sole application of docking tools is not sufficient to gauge correctly the interaction of such unique molecule[7] and it is necessary to combine it with Molecular Mechanics (MM) tools in the form of structure optimization of the gadofullerene-protein complex and scoring with the MM-PBSA method[8].

Finally, chapter 5 will analyze the most advanced and innovative application, i.e., using reverse screening protocols to investigate the interactions between ultrasmall nanoparticles (< 5nm) and proteins. Since the docking and Molecular Mechanics tools have been designed to treat large molecules only if they are biological, such proteins, nucleic acids and lipids, this results in a series of obstacles that must be cleared to successfully apply this kind of analysis. Nevertheless, promising results have been reached by combining the protein-protein docking tool ZDOCK 2.1[9] with advanced versions of the MM-PBSA method as scoring function, alongside structure manipulation and clustering tools from a series of packages. The resulting Free (and almost) Open Source code written in Bash will be analyzed and will be fully available in Appendix B.

1.1. References

[1] A. Hillisch, N. Heinrich, H. Wild, *ChemMedChem* **2015**, *10*, 1958–1962.

[2] C. J. Manly, S. Louise-May, J. D. Hammer, *Drug Discov Today* **2001**, *6*, 1101–1110.

[3] S. H. Friedman, D. L. DeCamp, R. P. Sijbesma, G. Srdanov, F. Wudl, G. L. Kenyon, *J. Am. Chem. Soc.* **1993**, *115*, 6506–6509.

[4] M. Calvaresi, F. Zerbetto, *ACS Nano* **2010**, *4*, 2283–2299.

[5] M. Calvaresi, F. Arnesano, S. Bonacchi, A. Bottoni, V. Calò, S. Conte, G. Falini, S. Fermani, M. Losacco, M. Montalti, G. Natile, L. Prodi, F. Sparla, F. Zerbetto, *ACS Nano* **2014**, *8*, 1871–1877.

[6] D. Duhovny, R. Nussinov, H. J. Wolfson, in *Algorithms in Bioinformatics* (Eds.: R. Guigó, D. Gusfield), Springer, Berlin, Heidelberg, **2002**, pp. 185–200.

[7] F. Bologna, E. J. Mattioli, A. Bottoni, F. Zerbetto, M. Calvaresi, *ACS Omega* **2018**, *3*, 13782–13789.

[8] J. Srinivasan, T. E. Cheatham, P. Cieplak, P. A. Kollman, D. A. Case, *Journal of the American Chemical Society* **1998**, *120*, 9401–9409.

[9] R. Chen, Z. Weng, *Proteins* **2003**, *51*, 397–408.

2. Theory of Computational Methods

Organic computational chemistry studies organic and bio-molecular systems using a theoretical approach: the molecules are translated into numerical model systems and results on properties and chemical-physical are obtained by solving physical equations.

The accuracy of these methods depends on the number of variables considered during calculation; even if it is impossible to consider every variable which describes a system with the currently available computing power and some approximation needs to be applied, thanks to the great advances in computer science computational chemistry simulations can deliver results that are in line with experimental evidence.

A vast array of computational approaches is available currently and depending on the type of information researched some methods are applicable, while others cannot produce reliable results. In general, computational methods in chemistry fall into two great groups depending if they are based on Quantum Mechanics (QM) or on Molecular Mechanics (MM) methods. Both methods can perform similar operations and yield the same type of information: the energy of a particular structure, optimizing the geometry of a system, calculating various observables ecc. QM methods are based on the concept of wave function and consider explicitly both nuclear and electronic motions. MM methods are based on Newtonian mechanics and rely on a heavy approximation of the studied system in which only nuclear motions are explicitly described while electronic motion is described only indirectly. As result, QM methods allows for an accurate description of chemical systems and can be used to describe chemical bond formation and breaking, since they are phenomena that depends on the electronic structure of a molecule, and to calculate variables, such as partial atomic charges, with high precision. However, they are extremely computationally expensive and their use is limited to relatively small systems. MM methods trades accuracy for performance thanks to the afore mentioned approximation and can be used on large systems, such as biological macromolecules; however, they cannot be used to obtain information which depends on the electronic structure.

A paramount example of MM calculations is Molecular Dynamics (MD), which allows to study the evolution of a system during time; nowadays MD simulations are widely used to study biological macromolecules and processes based on conformational changes of the molecule involved, such as the motion of an ion through ion channels.

In addition, post-processing of the atomic trajectories obtained from MD simulation can be performed to quantify the binding affinity between a determinate ligand and a receptor, a method known as Molecular Mechanics-Poisson-Boltzmann Surface Area (MM-PBSA).

Finally, molecular docking is a method used to predict the structure of the complex between two or more molecules. This method generates various possible spacial and conformational combinations (docking candidates) of the molecules taking part in the formation of the complex and select the best among them using a scoring function, which approximate the binding free energy of the complex. Many methods can be used for the generation of the candidates, such as shape complementarity, Monte Carlo simulations, Molecular Dynamic and genetic algorithms. Therefore, Molecular docking cannot be easily allocated in one of the two categories above and belongs to the more general field of molecular modelling, where QM and MM methods belong as well.

2.1. Molecular Mechanics

Molecular Dynamics simulates the evolution of a molecular system during time, modelling atoms as rigid spheres and bonds as oscillators defined by harmonic potentials. The method solves the Newton's equation of motion and quantifies the trajectory of the atoms, generating a displacement during time. In the following paragraphs the key concepts of the technique will be briefly analyzed.

2.1.1. The Force-field

The simulation is based on Newton's second law:

$$\boldsymbol{F} = m\boldsymbol{a} \qquad \textbf{Eq.1}$$

Where $\boldsymbol{F}$ is the force exerted on a body, which generates an acceleration $\boldsymbol{a}$ inversely proportional to the mass of said body m. MD simulations are N-body simulations, where each body is an atom of known mass; to obtain acceleration produced on it, it is necessary to know the force applied.
The force applied on each atom is derivable from the potential as it is the negative of the derivate of potential energy (V) with respect to the position of the body and to calculate it the initial coordinates and velocities of the system must be known:

$$\boldsymbol{F}_i = -\frac{\partial V}{\partial r_i} \qquad \textbf{Eq.2}$$

Then, solving Newton's second law for each atom, allows to calculate the position r_i of the i-th atom as function of time t:

$$\boldsymbol{F_i} = m_i\boldsymbol{a}_i = m_i\frac{d^2r_i}{dt^2} \qquad \textbf{Eq.3}$$

Because the trajectories of the atoms depend on the potential energy, it is evident that an accurate description of the potential energy is needed.

The potential energy function is separable into a sum of functions that represent intra- and inter-molecular forces within the system, due to the additivity principle:

$$V = V_{bond} + V_{angle} + V_{dihedral} + V_{improper} + V_{Coulomb} + V_{LJ} \quad \textbf{Eq.4}$$

V_{bond}, V_{angle}, $V_{dihedral}$ and $V_{improper}$ represent covalent interactions defined by a determinate number of bonds, angles and atoms. $V_{Coulomb}$ and V_{LJ} represent interactions between atoms separated by more than three covalent bonds (non-covalent interactions) and do not depend on a defined number of bonds.

V_{bond} , V_{angle} and $V_{improper}$ are modelled by harmonic potentials for stretching, bending and variation of improper angle ξ_i. Commonly, V_{bond} and V_{angle} are considered frozen at r.t., due to their high-energy constants. $V_{dihedral}$ is the only potential not represented with an harmonic potential and a more complex form of the potential is utilized; in addition, this is the only covalent interaction with a significantly low constant.

$$V_{bond} = \Sigma_{i=1}^{N_b} \tfrac{1}{2} k_i^b \left(r_i - r_{0,i}\right)^2 \quad \textbf{Eq.5}$$

$$V_{angle} = \Sigma_{i=1}^{N_\vartheta} \tfrac{1}{2} k_i^\vartheta \left(\vartheta_i - \vartheta_{0,i}\right)^2 \quad \textbf{Eq.6}$$

$$V_{improprer} = \Sigma_{i=1}^{N_\xi} \tfrac{1}{2} k_i^\xi \left(\xi_i - \xi_{0,i}\right)^2 \quad \textbf{Eq.7}$$

$$V_{dihedral} = \Sigma_{i=1}^{N_\psi} \tfrac{1}{2} k_i^\phi \cos\left[\left(\phi_i - \phi_{0,i}\right)\right] \quad \textbf{Eq.8}$$

Non-covalent interactions are represented by Coulomb and van der Waals potentials; while the first is described by the homonym theoretical law, Van der Waals potential is described empirically, usually as a Lennard-Jones potential:

$$V_{Coulomb} = \Sigma_{i<j} \frac{1}{4\pi\varepsilon_0 \varepsilon_r} \frac{q_i q_j}{r_{ij}} \quad \textbf{Eq.9}$$

$$V_{LJ} = \Sigma_{i<j} \left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}}\right) \quad \textbf{Eq.10}$$

The sum of these terms, which is the **Eq.4**, is the functional form of the force field, which describes the dynamics of a system. The parameters that describe the course

of the potentials usually derive from empirical data or theoretical calculations. To correctly model a system, a library of parameters for each constant at different conditions of state variables is needed.

'Parametrizing' a molecule consists in assigning the correct parameters to each atom of it structure on the basis of its atomic number, connectivity, spatial orientation and bond lengths. The combination of the parameter that represent an atom in a Molecular Mechanics representation is called *atom-type*: in other words, parametrizing a molecule consist in assigning the correct *atom-types* to the atoms that make up its structure. In the case of biomolecules such as proteins, lipids and nucleic acids, many tools are available for the automated assignment of atom-types since they are comprised of a series of basic building blocks that is constant among different chemical entities. Parametrization of different molecules, usually called *non-standard molecules*, is much more difficult and relies on a series of tools, among which Quantum Mechanical tools.

2.1.2. Temporal evolution of the system: the Verlet integrator

As already stated, MD simulations consist in the repeated solving of the Newton's equations of motion during time; one of the most accurate method available is the Verlet integrator, which is supported by the simulation engines of the majority of Molecular Mechanics suites. This method starts with the Taylor expansion of position:

$$\boldsymbol{r}(t+\delta t) = \boldsymbol{r}(t) + \boldsymbol{v}(t)\delta t + \frac{\boldsymbol{a}(t)}{2}\delta t^2 + \frac{\boldsymbol{b}(t)}{6}\delta t^3 + .. \qquad \textbf{Eq.11}$$

$$\boldsymbol{r}(t-\delta t) = \boldsymbol{r}(t) - \boldsymbol{v}(t)\delta t + \frac{\boldsymbol{a}(t)}{2}\delta t^2 - \frac{\boldsymbol{b}(t)}{6}\delta t^3 + .. \qquad \textbf{Eq.12}$$

Combining the two expansion gives:

$$\boldsymbol{r}(t+\delta t) = 2\boldsymbol{r}(t) - \boldsymbol{r}(t-\delta t) + \boldsymbol{a}(t)\delta t^2 + .. \qquad \textbf{Eq.13}$$

Substituting **Eq.1** in **Eq.13** gives an expression of the future position which depends on current and previous coordinates and on the force applied according to Newton's

second law. The value of velocity is estimated using the position terms and the mean value theorem.

$$\boldsymbol{r}(t+\delta t) = 2\boldsymbol{r}(t) - \boldsymbol{r}(t-\delta t) + \frac{\boldsymbol{F}(t)}{\boldsymbol{m}}\delta t^2 \qquad \textbf{Eq.14}$$

$$\boldsymbol{v}(t+\delta t) = \frac{\boldsymbol{r}(t+\delta t)-\boldsymbol{r}(t-\delta t)}{\boldsymbol{2\delta t}} \qquad \textbf{Eq.15}$$

Eq.14 and **Eq.15** yield the atomic positions and velocities during time from the initial coordinates: this information constitutes the trajectory of the system.

2.1.3. Approximations in Molecular Mechanics

Besides the afore mentioned approximation of MM methods, which is that they do not explicitly consider electronic motions, additional approximations are adopted in MD simulations to reach a good balance between computational cost and accuracy of the result.

Describing the solvent environment in an accurate manner is extremely challenging as an overwhelming number of molecules should be described; however, it is computational unrealistic and a vastly inferior number of solvent molecules can be considered. To represent a solvated system effectively, without using an infinite number of solvent molecules, ***periodic boundary conditions*** are employed: only a finite box solvent of a certain geometry is generated and it is replicated in all three dimensions to generate a periodic array.

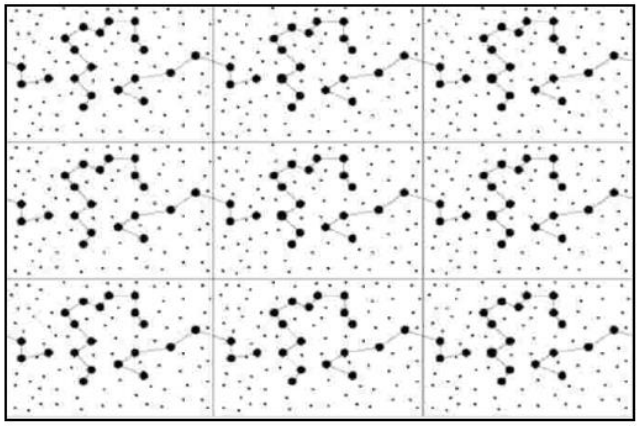


Figure 1 - Simple representation of periodic boundary conditions. Only atoms belonging to the central box are 'real' and independent, while atoms in the neighboring periodic boxes are just mirror images.

Each particle in a box, moves in the same fashion of the corresponding particle in the original box, but only the latter is represented; however, the effects are reproduced in all mirror images of the periodic system and each particle interacts not only with the other represented particles but also with their mirror images in the neighboring boxes. When a particle leaves the box, it is replaced by its mirror image which enters from the opposite side, as if it is coming out from the neighboring box. Another computationally demanding operation is the calculation of ***long-range non-bonded interactions*** because it considers the non-bonded potential for every atom pair in the system. However, Van der Waals interactions are usually modelled using Lennard-Jones potentials (**Eq.10**) which are proportional to r^{-6}; because of this, the Lennard-Jones potential between two distant atoms is negligible, allowing the application of a cut-off distance beyond which potential is ignored. On the contrary, electrostatic interactions are described by Coulomb's law (**Eq.9**) and decrease proportionally to r^{-1}, making long range contributions significant and preventing the safe usage of a cut-off distance. To overcome this limitation, ***Ewald summation method*** is employed. This method allows rapid calculation of electrostatic interactions by replacing the direct summation of the interaction energies between pairs of particles with the sum of a short-range potential and a long-range potential, which are differently defined. While the short-range potential is calculated in real space, the long-range potential is calculated in the Fourier space; as result, the calculation of these interaction is computationally affordable.

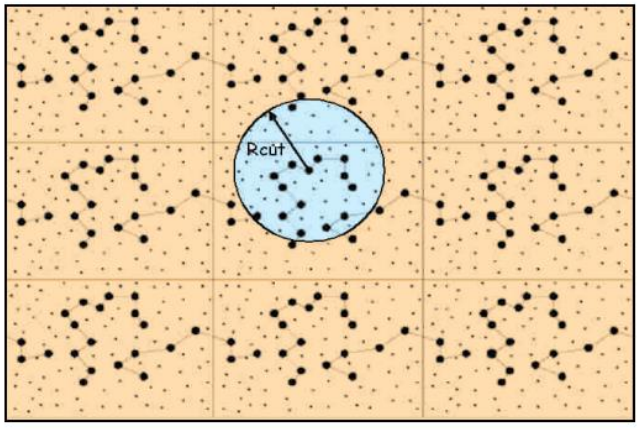


Figure 2 - Graphical representation of Particle Mesh Ewald method. The blue area in the picture contains the atoms within the short-range cut-off of the electrostatic interactions from the atom at the center of the area: coulomb interactions with these atoms in calculated in real space. Interactions with 'real' atoms and the periodic images in neighboring periodic boxes are calculated in Fourier space.

Even with these stratagems, the current computational cost for MD simulations sets the limit for the system size at 30k to 50k atoms and for the simulation lenght at few of microseconds at most; in both cases the use of accellerators in the form of GPGPU is necessary to reach such performance. However, for observing certain phenomena or obtaining certain information, such as the binding energy of a complex, this is an excellent tool if the force field is correctly parametrized for system.

2.1.4. Molecular Mechanics – Poisson Boltzmann Surface Area

The binding affinity between a ligand and a receptor can be quantified in terms of Gibbs free energy G.

$$G = U + pV - TS = H - TS \qquad \textbf{Eq.16}$$

Where U is the internal energy, p the pressure, V the volume, T the temperature, S the entropy and H the enthalpy of the system. According to the second law of thermodynamic, an isolated system evolves naturally to a state of lower Gibbs free energy; in other words, a process occurs spontaneously only if the variation of the Gibbs free energy between the final and initial state is negative. In the case of a binding process, the initial state are the isolated and non-interacting ligand and receptor while the final state is their complex and the Gibbs free energy variation (ΔG) of this process can be calculated as:

$$\Delta G = G_C - (G_L + G_R) \qquad \textbf{Eq.17}$$

Where G_C is the Gibbs free energy of the ligand-receptor complex, G_L is the Gibbs free energy of the ligand and G_R is the Gibbs free energy of the receptor. ΔG allows a quantification of the binding process and the more negative is value is, the more favorite is the process. Computational methods which allow to calculate the binding energy today are widely used; among them, Molecular Mechanics-Poisson-Boltzmann Surface Area (MM-PBSA) and Molecular Mechanics-Generalized Born Surface Area (MM-GBSA) methods compute binding free energies using molecular mechanics and continuum solvent.

The MM-PBSA/GBSA method was introduced by Srinivasan et al. in 1998 [1] as post-processing of an ensemble of frames generated with a Molecular Dynamics simulation. The binding energy of can be calculated from the average Gibbs free energies $\langle G_i \rangle$ of the complex, the ligand and the receptor as:

$$\Delta G = \langle G_C \rangle - (\langle G_L \rangle + \langle G_R \rangle) \qquad \textbf{Eq.18}$$

While in MD simulations, the system is solvated in a periodic box on explicit solvent molecules, in the MM-PBSA/GBSA approach the solvent is replaced by a continuum medium to save computational resources.

Theoretically, the variation of Gibbs free energies could be computed as represented in Figure 3, with the ligand, receptor and resulting complex in the solvated state; however, fluctuations in solvent-solvent interaction are of magnitude larger than the binding energy itself, effectively preventing its calculation.

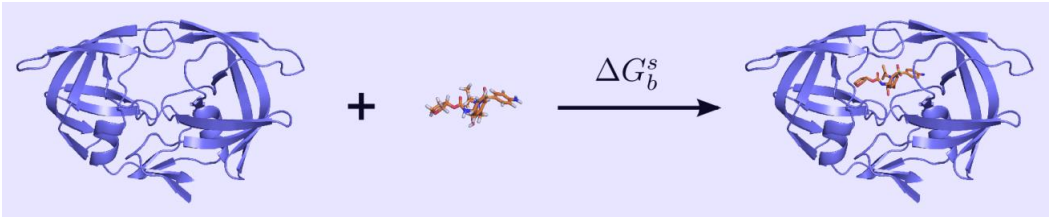


Figure 3 - Theoretically possible binding free energy calculations in the solvated state

Thanks to Gibbs free energy being a state function, $\Delta G_{bind,solv}$ can be calculated by building a thermodynamic cycle in which the initial and final states are a sum of states in the cycle, as represented in Figure 4. In this method binding energy and the solvation energy are separated; as result, $\Delta G_{bind,solv}$ is computed as:

$$\Delta G_{bind,solv} = \Delta G_{bind,gas} + \Delta G_{solv} \qquad \textbf{Eq.19}$$

$$\Delta G_{bind,gas} = G_{gas,C} - \left(G_{gas,L} + G_{gas,R}\right) \qquad \textbf{Eq.20}$$

$$\Delta G_{solv} = \Delta G^0_{solv,C} - \left(\Delta G^0_{solv,L} + \Delta G^0_{solv,R}\right) \qquad \textbf{Eq.21}$$

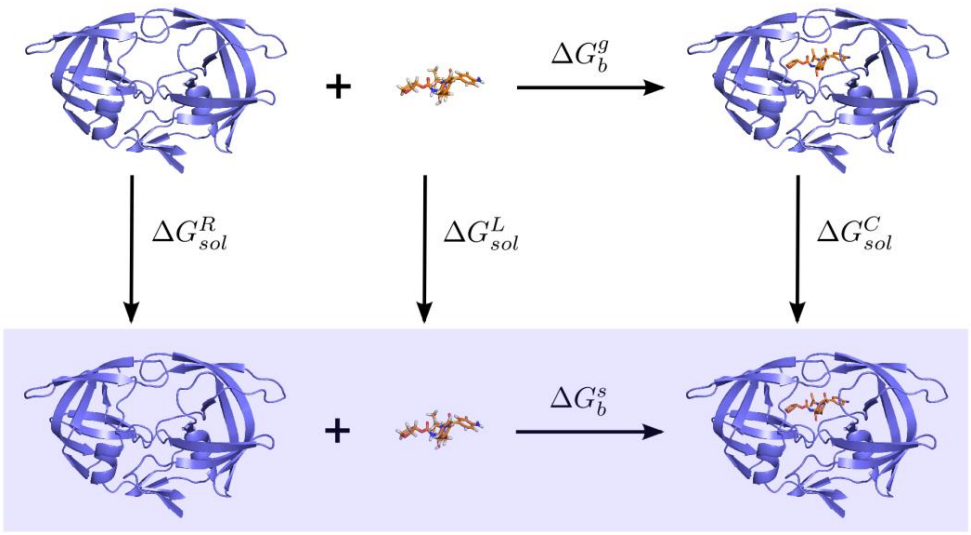


Figure 4 - Binding free energy thermodynamic cycle. Light blue background represents solvent environment, white background represent vacuum conditions.

Where $G_{gas,C}$ is the Gibbs free energy of complex formation in vacuum, $G_{gas,L}$ and $G_{gas,R}$ are respectively the Gibbs free energy in vacuum of the receptor and the ligand. $\Delta G_{bind,gas}$ is calculated via the force field used to model the system during the previous MD simulation and can be divided into a sum of internal, electrostatic and van der Waals contributions:

$$\Delta G_{bind,gas} = \Delta G_{int} + \Delta G_{vdW} + \Delta G_{elec} \qquad \textbf{Eq.22}$$

The solvation of each of the three entities in the thermodynamic cycle, namely complex, receptor and ligand, has a corresponding variation Gibbs free energy ($\Delta G_{solv,X}$) and it is more challenging to calculate. This free energy variation is separated into two components: a polar solvation term and non-polar solvation term.

$$\Delta G_{solv,X} = \Delta G_{polar,X}^{solv} + \Delta G_{non-polar,X}^{solv} \qquad \textbf{Eq.23}$$

Where X refers to one of the three afore mentioned entities. The polar solvation contribution is calculated by solving either the PB or GB equations; both methods consider the solvent as a high dielectric constant medium and the solute as an ensemble of fixed point charges embedded in a lower dielectric continuum. Only the

solvent environment as a whole and its effect on the behavior of the molecules that interact in the complex is considered, while the molecular nature of water is ignored. Therefore, chemical interactions such as hydrogen bonds are ignored as well, which is an important approximation of MM-PBSA/GBSA calculations.

The Poisson's equation describes the relationship between an electrostatic potential $\phi(\boldsymbol{r})$ at a point $\boldsymbol{r}$ generated by a charge distribution $\rho(\boldsymbol{r})$ in an environment of dielectric coefficient ε_r (relative to permittivity in vacuum ε_0).

$$\nabla[\varepsilon_0 \varepsilon_r \nabla\phi(\boldsymbol{r})] = -4\pi\rho(\boldsymbol{r}) \qquad \textbf{Eq.24}$$

Poisson-Boltzmann equation is derived from the latter equation. First, the charge distribution $\rho(\boldsymbol{r})$ is divided into two components: the solute charge density $\rho_f(\boldsymbol{r})$ and a contribution from the ions in the solvent environment $c(\boldsymbol{r})$. For N ion species with charge and bulk concentration, the ion charge distribution is given by:

$$c(\boldsymbol{r}) = 4\pi\Sigma_{i=1}^{N} q_j c_j^{\infty} e^{-\beta q_j \psi(r)};\ \ \beta = \tfrac{1}{k_B T} \qquad \textbf{Eq.25}$$

Where N is the number of ions species, q_j is the charge of said species and c_j^{∞} is their bulk concentration. Substituting $\rho(\boldsymbol{r})$ into **Eq.24** with **Eq.25** and by following steps considering electrostatic neutrality, expanding the expression function as a Taylor series, we obtain the linearized Poisson-Boltzmann equation:

$$\nabla\varepsilon_r\nabla\phi(\boldsymbol{r}) - 8\pi q^2 c^{\infty}\beta\psi(\boldsymbol{r}) = 4\pi\rho(\boldsymbol{r}) \qquad \textbf{Eq.26}$$

. The Poisson-Boltzmann equation needs to be solved both in the solvent and in vacuum, using the respective dielectric constants. At the end of this iterative computation, the polar term of solvation Gibbs free energy is:

$$\Delta G_{polar,X}^{solv} = \tfrac{1}{2}\Sigma_i q_i(\phi_i^{wat} - \phi_i^{vac}) \qquad \textbf{Eq.27}$$

Where q_i is the charge assigned to a grid point and ϕ_i^{wat} and ϕ_i^{vac} are the potentials in the same grid points, respectively, in water and vacuum.

An alternative implicit solvent model is the GB model, which is an approximation of the PB equation. In the GB model, the polar term for the solvation Gibbs free energy is described as the summation of the atomic charges multiplied for g_{ij}^{GB}.

$$\Delta G_{polar,X}^{solv} = \frac{1}{2}\Sigma_{i,j\in X} q_i q_j g_{ij}^{GB} \qquad \textbf{Eq.28}$$

g_{ij}^{GB} is determined by solving the Generalized-Born, whose functional form is:

$$g_{ij}^{GB} = \left(\frac{1}{\varepsilon} - 1\right)\left[r_{ij}^{n} + B_{ij} exp\left(-\frac{r_{ij}^{n}}{AB_{ij}}\right)\right]^{-\frac{1}{n}} \qquad \textbf{Eq.29}$$

Where B_{ij} is a parameter which depends by the distance from the solute-solvent dielectric boundary of atom *i-th* and *j-th* and the shape of the entire molecule considered. *A* and *n* have a preset value.

Finally, the non-polar term is constituted by two components: a van der Waals interaction term and the energy necessary for breaking the solvent structure and generating a cavity for the solute, which is taken as proportional to Solvent Accessible Surface Area (SASA) of the molecule X:

$$\Delta G_{non-polar,X}^{solv} = \Delta G_{non-polar}^{vdW} + \Delta G_{non-polar}^{cav} = \gamma SASA + \beta \qquad \textbf{Eq.30}$$

Where β and γ are parameters dependent of the method employed.

2.2. Quantum Mechanics

Quantum mechanics describe subatomic particles and electrons as both corpuscles and waves. The dual character of the electrons prevents the localization of such particles and as result renders impossible to distinguish one electron from the other. QM techniques are computationally demanding, therefore can only be used to study small sized chemical systems, 300-400 atoms at most, and for very short simulation lengths, from femtoseconds to few picoseconds. However, are the only methods that are capable to describe properties that arise from the electronic distribution of a

molecule, such as bond breaking and formation and dipoles. Going back to the theory behind these techniques, an electron has only a probability to be in a determinate point of the space, according to Heisenberg's indetermination principle, and this expressed by the wave function ψ, which is the solution of the Schrödinger equation:

$$\left[-\frac{h^2}{8m\pi^2}\nabla^2 + V(x,y,z)\right]\psi(x,y,z) = E\psi(x,y,z) \qquad \textbf{Eq.31}$$

The wave function of a mono-electronic system is perfectly solvable and is expressed as the product of a radial part $R_{n,l}(r)$ and an angular part $Y_l^m(\theta,\varphi)$it depends (on ()).

$$\psi_{n.l.m}(r,\theta,\varphi) = R_{n,l}(r)Y_l^m(\theta,\varphi) \qquad \textbf{Eq.32}$$

Where *r, θ, φ* are spherical coordinates. The functional form of a generic atomic orbital (AO) is **Eq.32**. The orbital assumes dimensions and shapes usually seen according to the values of the quantum numbers *n, l* and *m*.

2.2.1. Basis set

Mono-electronic wave functions that express the probability to find an electron in a certain point of space are referred to as Atomic Orbitals (AO) in the atomic case or molecular orbitals (MO) in the molecular case. MO are expressed as linear combination of atomic orbitals (LCAO).

$$\psi_i = \Sigma_i c_{ij}\phi_j \qquad \textbf{Eq.33}$$

Where ψ_i is the MO *i-th*, ϕ_j the atomic orbital and c_{ij} the associated coefficient of the linear expansion. The *j-th* wave function is part of a set of AO called *basis set*, which are linearly combined to generate MOs. This representation of a MO lowers the computational cost compared to MO calculated as numerical solutions of the Schrödinger equation. The quality of the MO representation depends on the number of AOs in the basis set, with the bigger basis sets yielding better representations of the MO. However, increasing the size of this ensemble of AO, increases the computational resources necessary to their usage; as result, a good balance between accuracy and computational efficiency is sought.

The AOs which constitute the basis set exist as solutions of the Schrödinger equation only for the hydrogen atom, therefore different representations of the atomic orbital

were devised. Slater Type Orbitals (STOs) mimic hydrogen wave functions and they are built with empirical parameters:

$$\chi_j^{STO} = Nr^{n-1}e^{\zeta r} \qquad \textbf{Eq.34}$$

Where N is a normalization constant, r is the distance of the electron from the nucleus and ζ is the empirical parameter mentioned above; the latter is related to the effective charge of the nucleus which is shielded by the electronic environment. However, these orbitals do not allow for precise calculation of polycentric integrals. A new generation of AO known as Gaussian Type Orbitals (GTOs). was developed. These atomic orbitals are widely employed in computational methods and described by:

$$\chi_j^{GTO} = Nx^l y^m z^n e^{-a(x^2+y^2+z^2)} \qquad \textbf{Eq.35}$$

Where N is a normalization constant and a is the orbital exponent, which is a constant and determines the radial expansion of the wavefunction. Defining the azimutal quantum number: $L = l + m + n$, for $L = 1$ the GTO describes a *s* orbital, $L = 2$ a p orbital and so on. The main difference between GTO and STO is the exponential dependence of the radial part, which makes GTOs worse in describing the electron density near and far from the nucleus than STOs. To overcome this limitation, a linear combination of Gaussian functions χ_j^{GTO}, called primitive Gaussian functions, is utilized:

$$\phi_i = \Sigma_{ij} b_{ij} \chi_j^{GTO} \qquad \textbf{Eq.36}$$

The resulting function ϕ_i is called contracted Gaussian and is the b_{ij} contraction coefficient which is held constant during the calculation.

Independently from the nature of the AO, the number of functions in the basis set is paramount for the accuracy of the calculation. The *minimal basis set* gives to each atom a number of basis function sufficient to place each electron. The basis functions

can be STOs expressed as linear combination of GTOs; with this method, we can obtain an accurate orbital with lower computing-demanding costs.

$$\phi_i^{STO} = \Sigma_{ij} b_{ij} \chi_j^{GTO} \qquad \textbf{Eq.37}$$

The summation runs from $i = 1$ to N and the resulting orbital is called STO-NG, which means that the orbital is obtained as a sum of N GTOs. STO-NG is the most popular minimal basis set as good geometry description for molecule in the ground state and composed by elements of the first period, which are the most recurrent; however, calculation of observables like energy are poorly accurate with this basis sets. The main problems are:

- During the course of a reaction, MO physiognomy change while the coefficients and the GTO functions remain fixed, giving an unnatural rigidity to the orbitals of the system;
- For the same reason anisotropy of the MOs is neglected;
- Elements of the same period are described with the same number of basis function, even if the number of electron increases from left to right. As results elements on the left side are described more accurately.

Some limitations of the minimal basis set can be avoided with the use of an *extended basis set*, which contains a greater number of contracted Gaussian function for the description of an orbital. *Double-ζ* (DZ) and *triple-ζ* (TZ) basis set contains two times and three times the number of functions of a minimal basis. When only the valence functions are doubled or tripled the basis set is called *split valence* (SV). Another improvement is represented by the use of *polarization functions*.

2.2.2. Correlation energy

The analytic solution of the Schrödinger equation system with more than one electron does not exist. As result, many approximate methods were developed to calculate wave functions that could describe the electronic environment of molecules and atoms.

The most famous *ab initio* method is the Hartree-Fock (HF) method, which revolves around the solution of a pseudo- Schrödinger equation, the Fock equation; its solutions are wave functions that are also necessary to express the Fock operator itself. This paradox is solved through the iterative approach using an input wave function obtained with more approximate methods.

Improvements of the HF method are the Roothan-Hall equations, which expand each MO as a LCAO (**Eq.33**) and the unrestricted Hartree-Fock (UHF) method. However, these methods are all based on the *independent-particle model*, that does not consider explicitly the electron-electron interactions but a mean potential felt by each electron and generated by the electronic surroundings.

This resulting error is known as *correlation energy* and is defined as:

$$E_{corr} = E - E_{HF} \qquad \textbf{Eq.38}$$

Where E_{HF} is the limit energy calculated by the Hartree-Fock method and E is the exact eigenvalue of the Schrödinger equation. The magnitude of the error is about the 1% of the total energy of a generic molecular system, which might seem not a big problem; however the energy involved in a chemical reaction and other observable of chemical interest are of the same magnitude as the correlation energy.

Considering a *N*-electron system, the Hamiltonian operator associated is espressed as:

$$\hat{H} = \Sigma_i \hat{h}_i + \tfrac{1}{2}\Sigma_{ij}\hat{h}_{ij} \qquad \textbf{Eq.39}$$

Where $\hat{h}_i$ is the monoelectronic Hamiltonian operator defined as the hydrogen-like Hamiltonian:

$$\hat{h}_i = -\tfrac{1}{2}\nabla_i^2 - \Sigma_j \tfrac{Z_j}{r_{ij}} \qquad \textbf{Eq.40}$$

$\hat{h}_{ij}$ in **Eq.39** is the Hamiltonian operator which represent the interaction between the *i-th* and *j-th* electron. This interaction is null if r_{ij} tends to infinite, an effect is known as "Coulomb's hole". The independent particle model ignores this phenomenon because the mean potential is constant and independent from the distance between electrons. A result, the probability of founding to found two electrons with opposite spin in the same point of the space is not null. The Pauli's principle on the other hand forbids the analogue phenomenon for electrons which possess the same spin; this means that the electrons are characterized by a "Fermi's

hole". HF method takes into account only the latter. The correlation energy can be divided in two components:

- *Internal* or *structure dependent correlation energy,* that refers to electrons represented by different spatial orbitals;
- *External* or *dynamic correlation energy* associated to the motion of the antiparallel electrons characterized by Coulomb's hole.

The calculations that take into account the dynamic correlation energy are refered to as *post Hartree-Fock methods.* The most promising member of this family of techniques is the Density Functional Theory (DFT), which has seen a widespread use in the last decades.

2.2.3. DFT: Density Functional Theory

The DFT method is broadly used because it is less computational demanding than other method, even for large molecular systems, but yields results in line empirical data, thanks to its ability to describe the correlation energy with high accuracy.
This approach is based on the Hohenberg-Kohn theorem [2] which states that all the fundamental state proprieties of the system are determined univocally by the electronic density $\rho(\boldsymbol{r})$ and that any other electronic density $\rho(\boldsymbol{r}')$ conducts to higher energy states than the real one; the electronic energy is expressed as functional of electronic density.

$$E = F[\rho(\boldsymbol{r})] \qquad \textbf{Eq.41}$$

Where E is the electronic energy, $\rho(\boldsymbol{r})$ is the electronic density and F is the functional which relates E to $\rho(\boldsymbol{r})$. The exact form of this functional is unknown however and various approximations were devised; the one used today is that proposed by Kohn and Sham[3]. Kohn-Sham equations reduce the problem of a structure with more than one electron to an ensemble of monoelectronic orbitals.

$$\hat{h}_i^{KS}\phi_i(\boldsymbol{r}) = \varepsilon_i\phi_i(\boldsymbol{r}) \qquad \textbf{Eq.42}$$

Where $\hat{h}_i^{KS}$ is the Kohn-Sham operator, $\boldsymbol{\phi_i(r)}$ are the Kohn-Sham wave functions for the non-interacting particles and ε_i is the eigenvalue of energy associated to the eigenfunction $\boldsymbol{\phi_i(r)}$.

The equation resembles the Schrödinger equation, but the electrons do not interact between themselves. This is artificial system which differs from the real one but has the same behavior, and the Kohn-Sham equations' eigenvalue of energy are equal to Schrödinger equations' eigenvalue of energy.

The formulation of the energy proposed by Kohn and Sham is a sum of different components: the kinetic energy (T_k), the electrostatic attraction electron-nucleus (E_{Ne}), the Coulombian term (J) and the exchange-correlation term (E_{XC}).

$$E[\rho] = T_k[\rho] + E_{Ne}[\rho] + J[\rho] + E_{XC}[\rho] \qquad \textbf{Eq.43}$$

HF calculations can be used to obtain electrostatic attraction between electron and nucleus (E_{Ne}) and the electrostatic repulsion between electrons (J) , because their definitions are the same in the two methods; the electron kinetic energy however has been redefined as:

$$T_k[\rho] = -\tfrac{1}{2}\Sigma_{i=1}^{N} \int \phi_i^*(\boldsymbol{r})\nabla^2\phi_i(\boldsymbol{r})d\boldsymbol{r} \qquad \textbf{Eq.44}$$

Where $\boldsymbol{\phi_i(r)}$ are the Kohn-Sham non-interacting particles wave functions and are the eigenfunctions of the Kohn-Sham eigenvalue equation (**Eq.42**). The functional expressed in **Eq.43** is determined less than the exchange-correlation energy (E_{XC}), whose representation determines the quality of the DFT. It is possible to define the Kohn-Sham operator as:

$$h_i^{KS} = -\tfrac{1}{2}\nabla^2 - \Sigma_{k=1}^{Nn}\frac{Z_k}{|r_i - r_k|} - \int \frac{\rho(r')}{r_i - r'}d\boldsymbol{r'} + V_{XC} \qquad \textbf{Eq.45}$$

Where V_{XC} is the exchange-correlation term for one electron and is represented as:

$$V_{XC} = \frac{\delta E_{XC}}{\delta \rho} \qquad \textbf{Eq.46}$$

Where E_{XC} is the expectation value of the energy for a monodeterminal wave function, solution of **Eq.42**. Because the analytical form of $E_{XC}[\rho]$ cannot be determined, approximations must be used. The most important is separation of this functional into a sum of contributes.

$$E_{XC}[\rho] = E_X[\rho] + E_C[\rho] \qquad \textbf{Eq.47}$$

A vast array of DFT methods have been developed for the calculation of the exchange functional ($E_X[\rho]$) and correlation functional ($E_C[\rho]$). They can be classified as *local methods*, where only the electron density is used and *non local method* or *generalized gradient corrected*, where is the gradient of the electron density is used as well.

2.3. Molecular docking

Molecular docking techniques try to predict the structure of a complex formed from the interactions between two molecules, usually a small ligand and a receptor or an enzyme. This technique is broadly used to predict the binding modes of pharmaceutical compounds on their biological targets and can be employed at multiple stages of the process of drug design

Many degrees of freedom are associated with the docking problem:

- six degree of translational and rotational freedom of one of the two interacting partners with respect to the other;
- all the internal conformational degrees of each molecules, making the calculation very computationally expensive if all of them are considered.

Even if this problem can be tackled manually by using accurate computer graphics and can be quite effective if the operator has a good hypothesis of the possible binding mode, generally deriving from prior knowledge of the binding mode of a similar ligand. However, a plethora of single crystal x-ray studies has shown that

very similar drugs can prefer very different binding modes. As result, automatic protocols are preferred as they can be less biased and consider more binding modes.

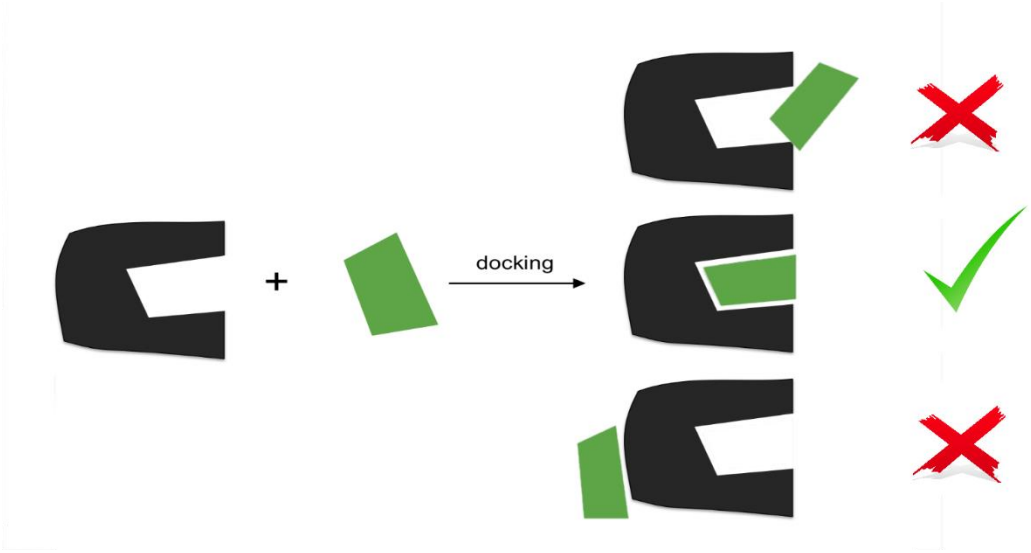


Figure 5 - Simple graphical representation of the goal of molecular docking.

2.3.1. Types of molecular docking techniques

A vast array of algorithms has been developed through the years to tackle the problem of generating the most reasonable complex geometry and differ in the number of degrees of freedom considered. Rigid-body docking refers to techniques that consider only the six rotational and translational degrees, ignoring the conformational degrees of freedom and effectively considering the two interacting entities as rigid. Their molecular surface is mapped using various methods which create an array of overlapping spheres of radii of the magnitude of atomic radii (probes), which touch the molecular surface at two points only; the smaller the probe, the higher the resolution of the surface. Subsequently, evaluation of their shape complementarity is performed and possible binding modes are generated.

These methods based on the lock and key model of the interaction between a ligand and a receptor. This type of docking usually cannot produce accurate result because the interactions between a ligand and a receptor are the same which govern the secondary structures of the receptor. As result, the conformation of the latter undergoes a transformation upon binding (induced fit model) and the ligand changes its conformation as well to enhance the interaction with its partner.

Flexible docking methods consider the two entities as flexible and differ in the number of conformational degrees of freedom considered. For the sake of lowering the computational cost, only the flexibility of the ligand is usually considered.

Figure 6 - Overview of the various types of docking techniques.

Considering all the conformational degrees of freedoms of both ligand and receptor would result in the best possible solutions for the complex geometry; thanks to the current computing power, MD simulations can be performed to this end. However, they cannot explore a large range of binding modes, except for small ligands, due to the magnitude of the energy barriers that separate them that is usually too high to be overcome during the length of a simulation. In addition, they are very computationally expensive and cannot be used efficiently on a large array of molecules.

Nowadays, rigid-body docking and methods that consider only the ligand flexibility are used for large screening of macromolecules' databases, while methods which consider the full extent of the conformational space of both ligand and receptor are used only to refine the structures obtained with more primitive algorithms. Recently, methods which consider protein flexibility only locally in the binding site have been developed, but they are still far from their effective application.

2.3.2. Methods for docking candidates' generation and scoring

Usually, docking algorithms produce a vast array of possible structures, a step called *docking candidates generation*, and valuates their binding energy them according to a scoring function, a step called *docking candidates scoring*. At some point in time, all the broadly used computational methods for the study of the conformational space of molecules have been included in docking algorithms, such as Monte Carlo simulations, genetic algorithm-based methods, distance geometry and incremental construction of the ligand. As result, finding common grounds and theory for the pose generation is very difficult and only the methods used in the case of the docking tools that have been employed in this work will be analyzed. Nevertheless, the following are the methods that are used by the majority of modern docking tools:

- Rigid 3D transformations with a series of conformers
  - The most common method, which involves roto-translational transformations of the ligand's coordinates. To take into consideration ligand flexibility, multiple conformers are docked for a single molecule. Most techniques generate conformers internally before the actual docking phase, rather than accepting an ensemble of conformers as inputs. Every docking software of this family fall into one of two sub-categories: i) *brute force enumeration* of the transformation space and ii) *local shape feature matching*[4].
    - Brute force algorithms search the entire 6-dimensional transformation space of the ligand and use FFT (Fast Fourier Transform,) for fast enumeration of the translations.
    - Local shape feature matching algorithms direct the exploration of the roto-translational space of the ligand by matching its surface feature to the features of the interacting partner. This results in much faster computation.
- Incremental construction
  - The broad philosophy of fragment based docking methods can be described as dividing the ligand into separate portions or fragments, docking the fragments, followed by the linking of fragments. These methods require subjective decisions on the importance of the various

functional groups in the ligand, which can result in the omission of possible solutions, due to assumptions made about the potential energy landscape[5].

- Genetic Algorithms
  - Techniques that mimic the process of evolution by manipulating a collection of data structures called *chromosomes*. Each of these structures corresponds to a possible solution to the docking problem, i.e., a possible ligand orientation within the protein binding site where each degree of freedom corresponds to a *gene* . Each chromosome is assigned a fitness score based on the relative merit of that solution according to a scoring function[6]. The population of solutions evolves through the use of genetic operators:
    - mutations: the value of a gene is randomly changed.
    - crossovers: a set of genes is exchanged between parent chromosomes.
    - migrations: motion of individual genes from one sub-population to another[5].

The scoring functions that are built inside docking tools approximate the binding free energy and are computationally cheaper than various techniques used to valuate this energy with high accuracy. As result, they can be used to screen large scale databases in short time. Fortunately, in this case it is possible to divide the scoring functions into three broad families[7]:

- Force-field
  - (electrostatic + vdW (+ solvation))
  - Based on physical atomic interactions like van der Waals interactions, electrostatic interactions and bond lengths, bond angles and torsions.
- Empirical (often combined with Ffs)
  - The binding energies of a complex can be approximated by a sum of individual uncorrelated terms. The coefficients of the various terms involved in calculation of binding energy are obtained from regression

analysis using experimentally determined binding energies or potentially from X-ray structural information.

- ❖ Knowledge-based
  - ? (compare interactions to some reference set)
  - ? The functions use statistical analysis on crystal structures of complexes to obtain the interatomic contact frequencies between the protein and the ligand based on the presumption that stronger an interaction is, the greater the frequency of its occurrence will be.

2.3.3. PatchDock: fast rigid-body docking based on computer vision

A local shape feature matching algorithm which was inspired by object recognition and image segmentation techniques used in computer vision[4]. Given two molecules, their surfaces are divided into patches according to the surface curvature and the patches are superimposed using shape matching algorithms to generate docking candidates. In addition to docking small molecules on proteins, this software can be used for protein-protein docking as well and performed favorably in CAPRI evaluations[8]. The algorithm has three major stages:

- ❖ Molecular Shape Representation
  - ? The surface of the molecule is generated using the MS program which generates a high density Connolly surface[9,10]. A sparce surface representation is generated as well[11], which consists of critical points named *caps*, *pits* and *belts*. The latter correspond to the projections (perpendicular to the molecular surface) of the centers of, respectively, convex, concave and saddle areas of the Connolly surface.

    These points are used to divide the surface of the molecule into patches of almost equal area of three types: *convex*, *concave* and *flat* patches[4].

    A graph is obtained by connecting critical points close to each other: each pit point can be connected by an edge to at most three caps and three belts. Each belt point is connected to two corresponding caps. A probe is placed at each critical point and the fraction of the probe that occupies the solvent-excluded volume of the molecule, calculated on

the basis of the Connolly surface, is the *shape function* which intersect with the solvent excluded surface of the molecule, i.e. the Connolly surface, is the *shape function* of the point. Depending on its value, the point is marked as *knob, hole* or *flat*. Surface patches are then generated by combining points of the same type that correspond to a connected subgraph of the overall molecular graph[4]. Patches are created with similar sizes, independently of their type. If the user desires, the patches are then filtered, so that only patches with 'hot spot' residues[12] are used in the actual docking process.

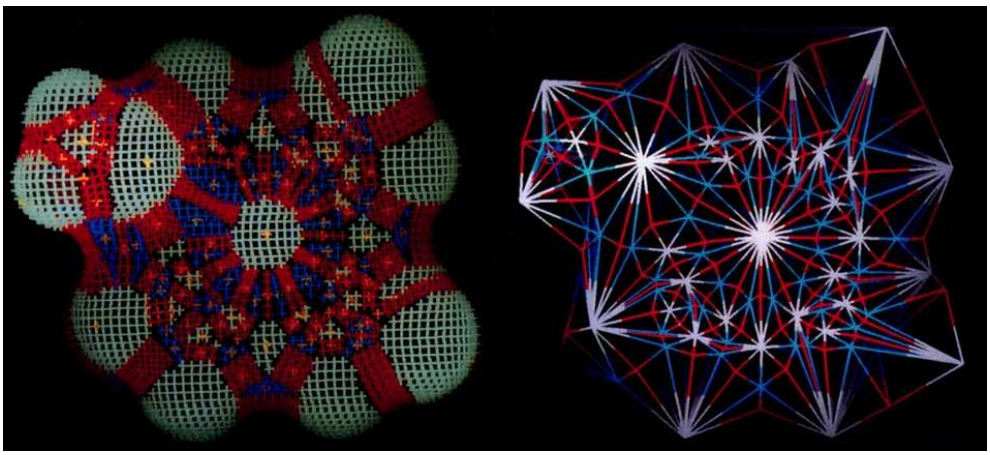


*Figure 7 - **Left**: Connolly's dots (generated with the MS program and the critical points on the heme surface. All points drawn as small crosses. Colors: light-green=convex faces; blue=concave faces; red=saddle-shaped faces; yellow, critical points. **Right**: The criticalpoints connected in a triangle mesh. Colors: white=caps; blue=pits; red=belts[11].*

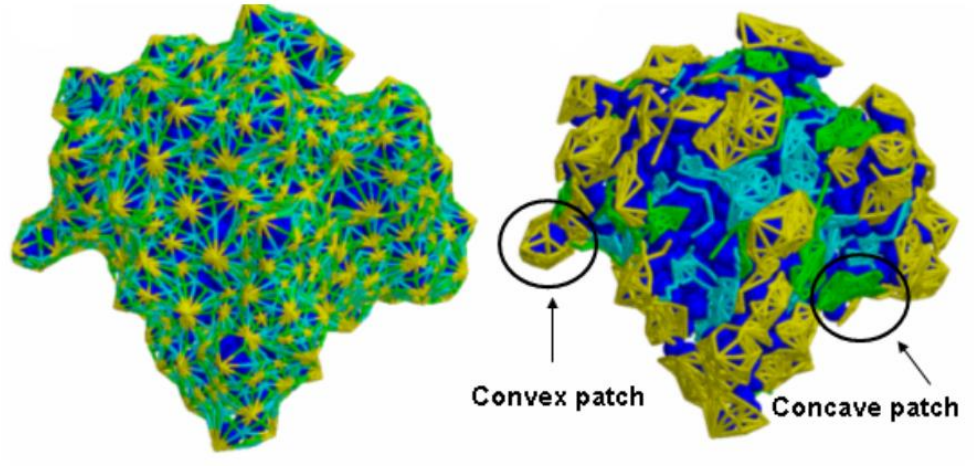


*Figure 8 – **Left**: surface topology graphs for trypsin inhibitor (PDB id 1BA7). The caps (yellow), belts (green) and pits (lightblue) are connected with edges. **Right**: resulting geometric patches: the patches are in light colors and the protein is dark[4].*

- Surface Patch Matching
  - Based on the principle that knob patches should match hole patches and flat patches can match any patch. Two techniques are used for matching[4]:
    - Single Patch Matching: one patch from the receptor is matched with one patch from the ligand. This type of matching is used for docking of small ligands, like drugs or peptides which have a small area of contact.
    - Patch-Pair Matching: two patches from the receptor are matched with two patches from the ligand. This is employed in protein-protein docking since it results in a larger area of contact at the binding site

    For the actual generation of complexes a combination of Computer Vision motivated Geometric Hashing and Pose-Clustering is used[4] in both cases. A certain degree of surface penetration is allowed to account to protein and ligand flexibility. RMSD (root-mean-square deviation) clustering is performed on the docking candidates and redundant docking solutions do not advance to the scoring step

- Filtering and Scoring
  - The docking candidates are examined and structures with unacceptable penetrations of the atoms discarded. The remaining candidates are ranked according to a geometric shape complementarity score. Given the fact that van der Waals interactions scales with the area of contact between the interacting molecule, this shape complementarity score evaluates them indirectly. In addition, atomic desolvation energies are also taken into account. In the case of protein structures, the atoms of each residues are assigned one of 18 atom-types deduced from crystallographic structures of proteins[13] (as result, this scoring function is also knowledge-based); regarding *non-standard* molecules, approximated versions of the same atom-types are used.

2.3.4. ZDOCK 2.1: rigid-body protein-protein docking that reward continuous areas of contact

It is a protein-protein docking software that belongs to the *brute force enumeration* family and uses a Fast Fourier Transform algorithm to rapidly explore the roto-translational space of a protein [14] (which will be called 'ligand' for the sake of simplicity from here on) relative to its interacting partner.

Contrary to PatchDock and similar docking methods, it is a *grid-based* docking algorithm, ,i.e., it use a surface description that does not include explicit information on surface curvature[14]. Instead, the receptor and ligand atoms are placed in a grid and a number of points of said grid that surround, but do not overlap with any atoms of the structures, is used to describe their surfaces. Roto-translational transformations of the ligand coordinates are computed and each transformation is assigned a surface complementarity score based on the number of grid points that overlaps with the grid points of the receptor surface; a penalty for overlapping grid points that correspond to atoms centers in the two proteins. This type of score is called Grid-based Shape Complementarity (GSC)[14].

ZDOCK 2.1 uses an upgraded version of scoring function called Pairwise Shape Complementarity (PSC) which uses two complex functions to describe the receptor and the ligand grid points; the real part of each function is used for computing the favorable component of the interaction, while the imaginary part is used to penalize steric clashes between atoms on the two partners.

The two components can be briefly summarized as this:

- Favorable component
  - For each grid point of the receptor, PSC computes the total number of receptor atoms within a distance cutoff, which depends on the atoms' van der Waals radii, and assigns that number to the grid point if it lies outside the solvent-excluding surface of the protein (calculated on the basis of the van der Waals radii of the atoms), 0 otherwise.

    Grid points of the ligand are assigned a value equal to 1 if they are the closest point to a ligand atom, 0 otherwise.

❖ Penalty component

? For both the ligand and the receptor, grid points that lie in the solvent-excluding surface of the protein (calculated on the basis of the van der Waals radii of the atoms) receive a penalty of value of 3, while grid points that lie in the core of the structure receive a value of 9. On the contrary, grip points in the open space around the solvent-excluding surface receive no penalty value.

$$\mathrm{Re}[R_{PSC}(l,m,n)] = \begin{cases} \text{number of receptor atoms within (D + receptor atom radius)} & \text{open space} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathrm{Re}[L_{PSC}(l,m,n)] = \begin{cases} 1 & \text{if this grid is the nearest grid of a ligand atom} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathrm{Im}[R_{PSC}(l,m,n)] = \mathrm{Im}[L_{PSC}(l,m,n)] = \begin{cases} 3 & \text{solvent } \textit{excluding} \text{ surface of the protein} \\ 9 & \text{protein core} \\ 0 & \text{open space} \end{cases}$$

Figure 9 - Real (Re) and imaginary (Im) parts of the functions that describe receptor ($R_{PSC}(l,m,n)$) and ligand ($L_{PSC}(l,m,n)$) grid points[14].

The score of a transformation is obtained by adding the favorable values of the overlapping grid points of the receptor and the ligand, minus the clash penalty which is computed by combining the penalty values of said overlapping grid points. In particular, core-core, surface-core, or surface-surface grid point overlap results in an overall penalty of -9*9=-81, -3*9=-27 and -3*3=-9. In other words, overlaps involving surface grid points are penalized only moderately to take into consideration protein structural flexibility[14].

This method effectively rewards all close atomic contacts between the receptor and the ligand, i.e. it maximizes the number of receptor atoms close to each ligand atom and vice-versa. Since neighboring atoms in one protein tend to make contacts with the same atoms in the other protein, PSC rewards continuous surface patches at the

binding site, lowering the number of false-positive predictions since the latter are usually characterized by large, yet non continuous, areas of contact[14].

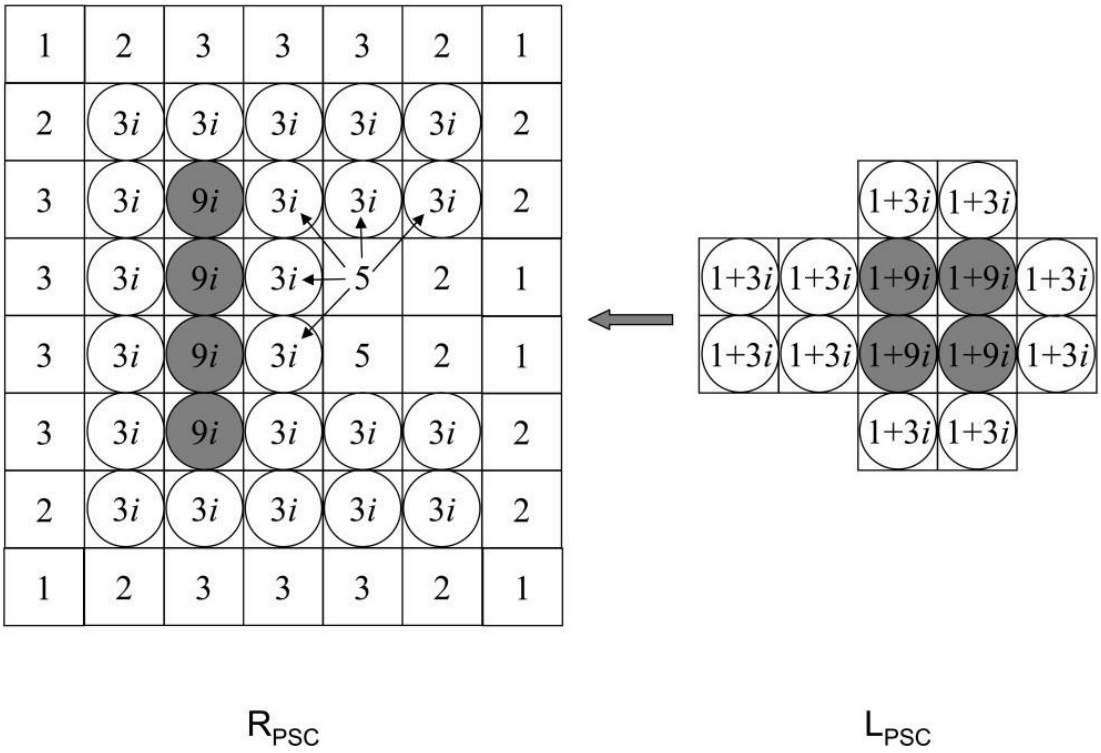


Figure 10 - 2D schematic illustration for the discrete functions RPSC and LPSC used in PSC. Protein atoms are indicated using circles, with open circles indicating surface atoms and shaded circles indicating core atoms. For clarity, a grid spacing that equals atom diameter has been used and grid points whose values are 0 have been omitted from the figure. The block arrow indicates the direction of translation for the ligand in order to achieve the optimal shape complementarity score. For each grid point in the open space of RSC, the number of atoms within a distance cutoff is computed (the cutoff has been set to be 1.5 times atom diameters for illustration purposes). Small arrows point out the five atoms that are within the distance cutoff of a grid point and thus contribute to its score of 5[14].

No other terms play a role in this scoring function except shape complementarity. Since the latter is directly proportional to van der Waals interaction, as previously stated, it is exceptionally effective in describing the interactions between hydrophobic chemical entities.

In addition, PSC is impartial to the receptor/ligand assignment of input proteins, in contrast with other shape complementarity scoring functions that perform better when the protein with the concave binding site is designated as the receptor, a decision that cannot be made when the binding site is unknown.

2.4. References

[1] J. Srinivasan, T. E. Cheatham, P. Cieplak, P. A. Kollman, D. A. Case, *Journal of the American Chemical Society* **1998**, *120*, 9401–9409.

[2] P. Hohenberg, W. Kohn, *Phys. Rev.* **1964**, *136*, B864–B871.

[3] W. Kohn, L. J. Sham, *Phys. Rev.* **1965**, *140*, A1133–A1138.

[4] D. Duhovny, R. Nussinov, H. J. Wolfson, in *Algorithms in Bioinformatics* (Eds.: R. Guigó, D. Gusfield), Springer, Berlin, Heidelberg, **2002**, pp. 185–200.

[5] R. D. Taylor, P. J. Jewsbury, J. W. Essex, *J Comput Aided Mol Des* **2002**, *16*, 151–166.

[6] G. Jones, P. Willett, R. C. Glen, A. R. Leach, R. Taylor, *Journal of Molecular Biology* **1997**, *267*, 727–748.

[7] A. Sethi, K. Joshi, K. Sasikala, M. Alvala, *Drug Discovery and Development - New Advances* **2019**, DOI 10.5772/intechopen.85991.

[8] P. Kangueane, C. Nilofer, in *Protein-Protein and Domain-Domain Interactions*, Springer Singapore, Singapore, **2018**, pp. 161–168.

[9] M. L. Connolly, *Journal of Applied Crystallography* **1983**, *16*, 548–558.

[10] M. L. Connolly, *Science* **1983**, *221*, 709–713.

[11] S. L. Lin, R. Nussinov, D. Fischer, H. J. Wolfson, *Proteins: Structure, Function, and Bioinformatics* **1994**, *18*, 94–101.

[12] Z. Hu, B. Ma, H. Wolfson, R. Nussinov, *Proteins: Structure, Function, and Bioinformatics* **2000**, *39*, 331–342.

[13] C. Zhang, G. Vasmatzis, J. L. Cornette, C. DeLisi, *Journal of Molecular Biology* **1997**, *267*, 707–726.

[14] R. Chen, Z. Weng, *Proteins* **2003**, *51*, 397–408.

3. Porphine and phthalocyanine

3.1 Porphyrins, porphine and phthalocyanines

Porphyrins are a family of naturally occurring compounds characterized by a central scaffold composed of 4 pyrrolic sub-units that are linked by 4 sp^2 methine group to make a macrocycle; the scaffold is known as *porphine*. This scaffold is planar thanks to the high degree of electron delocalization, that causes strong absorption in the visible spectrum. As result, the molecules that belong to this family are characterized by strong coloration which was at the origin of their name: *porphura* means 'purple' in Greek. The scaffold can be substituted at the bridge methine atoms (*meso* position) or at the carbon atoms of the pyrrolic sub-units that are not bonded to the methine carbons (β position). The compounds that result from the substitution in these two positions are called porphyrins.

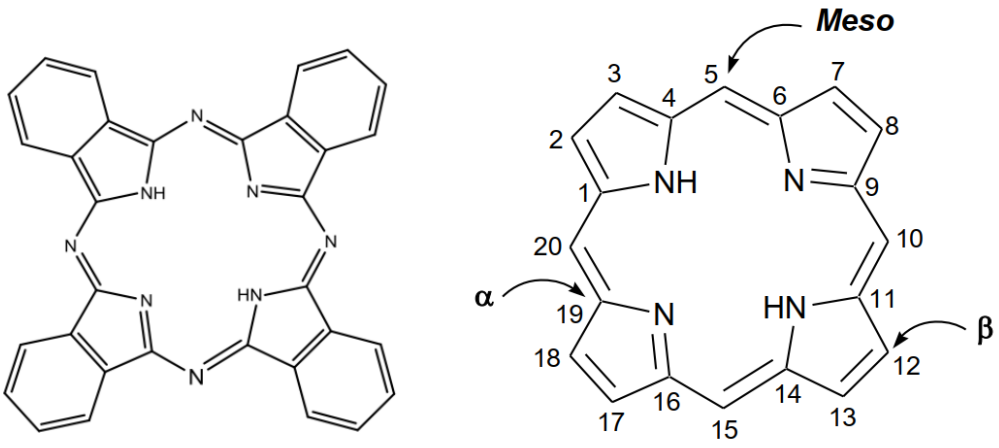


Figure 1 - Left: phthalocyanine; Right: porphine scaffold with the nomenclature of the atom positions.

Phthalocyanines are related to porphyrins and are characterized by a planar conjugated macrocycle of 4 iso-indole sub-units linked by 4 conjugated nitrogen atoms. With respect to the porphine scaffold, the addition of a benzene ring fused to each β carbon of the pyrrolic units causes an improvement of the absorption in the high wavelength region of the visible spectrum (670-780 nm), improving tissue

penetration, and brings the extinction coefficient of the molecule almost two orders of magnitude higher than that of porphyrins[1].
Thanks to their strong absorption in the visible spectrum, both classes of molecules have been investigated as *photosensitizers* (PS) for *photodynamic therapy*.

3.2 A brief history of PDT and an overview of the technique

The correlation between photodynamic activity and positive therapeutic outcomes is known since millennia. Ancient Egyptian, Indian and Chinese civilizations used light to treat various diseases, including psoriasis, rickets, vitiligo and skin cancer[2], but only at the beginning of the last century the scientific basis of this association was uncovered[3]. One of the biggest contributors was Niels Finsen who was awarded the 1903 Nobel Prize for his work on use of phototherapy to treat the skin manifestation of tuberculosis, a very common and deadly condition of his time. In 1912, Meyer-Betz showed that the compound hematoporphyrin (Hp) isolated from hemoglobin was characterized by strong photodynamic properties and localized preferably in tumor tissue. A purified version called hematoporphyrin derivative (HpD) was able to reach even better tumor localization and a more refined purification by Dougherty in the 1980s led to the compound porfimer sodium, commercially known as Photofrin. HpD and Photofrin were the first PS that were tested for clinical applications and in 1995 FDA approved the use of Photofrin as the first PS for the therapy against certain tumors[1].
Modern PDT is a non-invasive treatment for various types of tumors that revolves around photosensitizers, molecules that in their excited states are capable of producing cytotoxic agents through photochemical reactions while being harmless in their ground state. By irradiation of the PS molecule, it is possible to produce a controlled amount of reactive oxygen species (ROS) that are capable of damaging cell structures. If the photosensitizer has been accumulated more within the tumor tissues as compared to normal tissues, this results in localized tumor destruction without damaging healthy surrounding cells, since ROS have a short half-life, therefore limited range of action.

3.2.1 Mechanisms of ROS production and indirect immune response

When a photosensitizer agent is irradiated by light of the correct wavelength, it is elevated from the ground state (S_0) into a short-lived, electronically excited state (S_n). Via *internal conversion* (IC), the excited PS decays through a number of vibrational sub-levels (S_n') to populate the first excited singlet state (S_1)[1]. From the S_1 state, the PS quickly relaxes to the S_0 ground state since S->S transitions are allowed according to Spin Selection Rules, releasing the absorbed energy via fluorescence. The excited S_1 state can also undergo *intersystem crossing* (ISC) and reach the first excited T_1 state, a spin-forbidden process, from which it can decay to the ground state via phosphorescence. Since decay from the T_1 state to the S_0 is a spin-forbidden process as well, the half-life of the T_1 state is considerably longer than the S_1 state (10^{-3} to 1 second compared to 10^{-9}-10^{-6} seconds)[1]

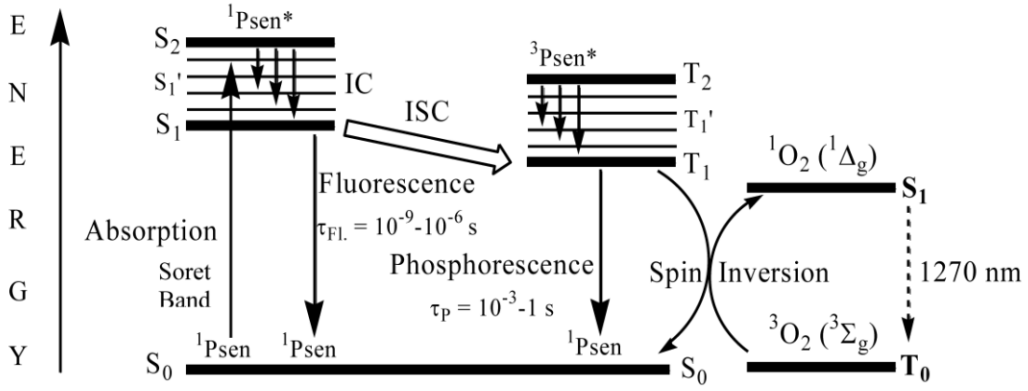


Figure 2 - simplified Jablonski diagram of a photosensitizer molecule.[1]

As result, the PS in the T1 state can interact with the surrounding environment and elicit damage according to two processes. The first process consists in one-electron oxido-reduction reaction with biomolecules in the surroundings, which causes the production of free radical intermediates that interact with molecular oxygen $^3O_2$ to generate various ROS (Type I). Alternatively, the T_1 state PS can interact directly with molecular oxygen and cause its conversion to singlet oxygen $^1O_2$ by energy transfer.[4] Singlet oxygen has a typical lifetime of approximately 40ns in biological systems and it is considered the main cause of cellular damage elicited by PDT[1,3].

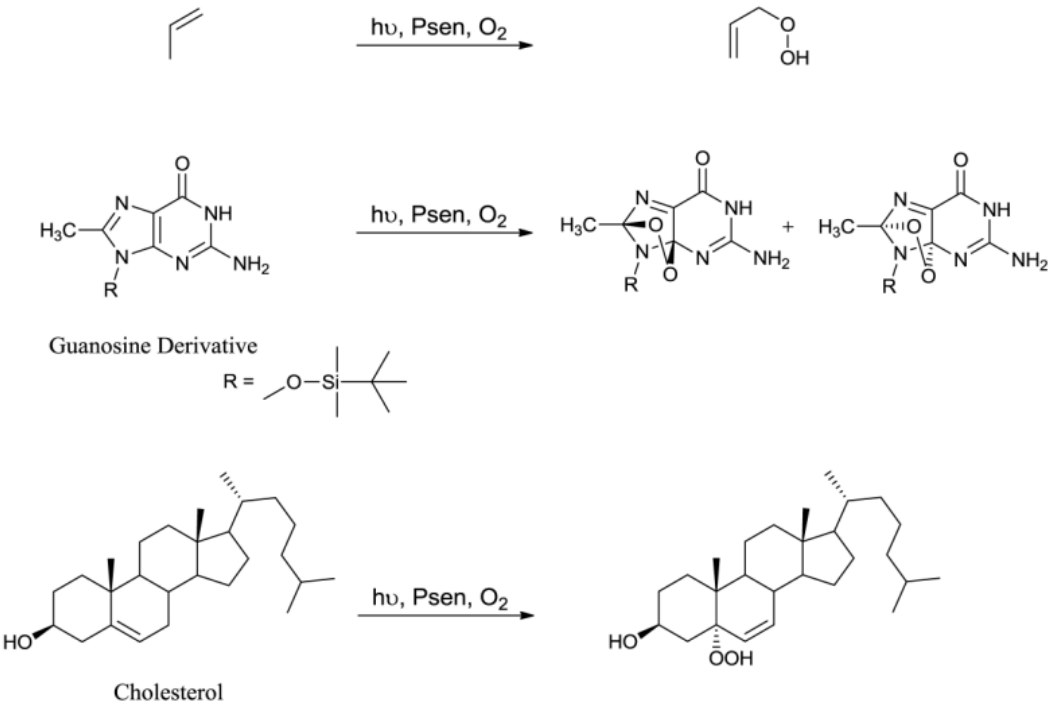


Figure 3 - Examples of the reactions at the basis of singlet oxygen cytotoxicity.[1]

In addition to the Type I and Type II processes, a growing body of evidence suggests that the antitumoral effects of PDT are also mediated by indirect stimulation of inflammatory and immune responses [5].

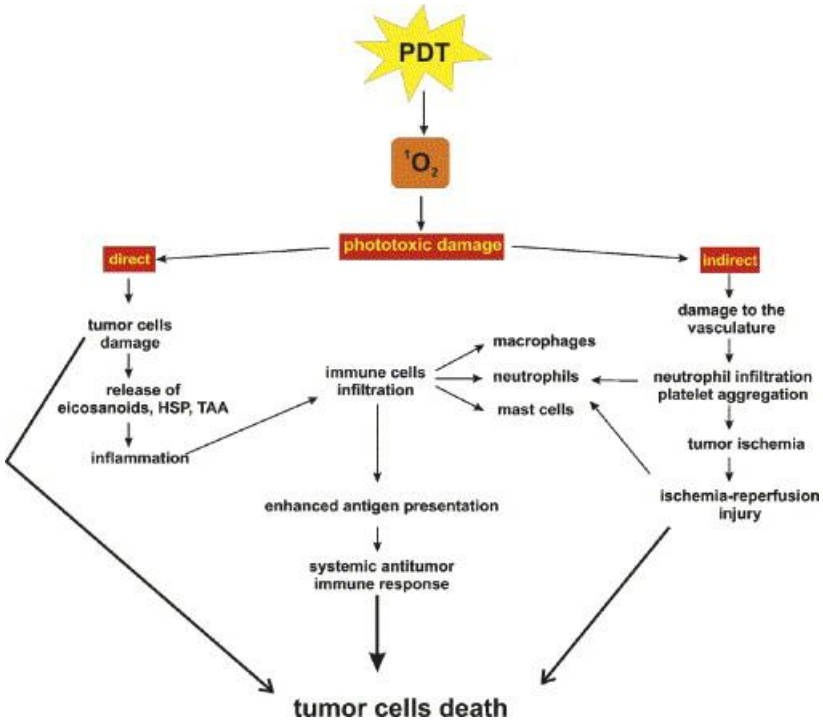


Figure 4 - The mechanisms of antitumor effects triggered by PDT: direct and indirect processes.[5]

The oxidative stress caused by photodynamic therapy triggers a series of protective responses by the organism, including the expression of heat shock proteins, and transcription factors such as NF-κB and AP-1. PDT induce rapid infiltration of neutrophils and macrophages into tumor tissue and there is evidence of increased antitumor effects when PDT is combined with non-specific immunostimulatory substances. [5]

3.2.2 The outcomes of a PDT attack on a tumor tissue: cell death or growth inhibition

Apoptosis is a physiological process that is used by organisms for the regulation of tissue development and homeostasis. It is a process of controlled cell suicide that is regulated by both intracellular and extracellular signals that dismantles the cell into its components that are phagocitated by macrophages and neighboring cells[6]. This results in a limited leakage of intracellular material, preventing inflammation. On the other hand, necrosis is an uncontrolled cell death that is caused from high levels of cell damage and results in cell membrane tearing and tissue inflammation due to the release of intracellular material. Although it has been demonstrated that PDT can induced both types of processes, in many cases apoptosis is the induced mechanism[6].

In addition to the destruction of tumor tissue, PDT can also be used to suppress cell growth. If PDT is carried out in optimal conditions of light, oxygen and PS concentration (high dose PDT) cell death, mainly from apoptosis, is observed; when one of these components is limiting ("low dose PDT"), PDT suppress the cell cycle at the G_2/M checkpoint.[4]

3.2.3 Efficacy of PDT and PS's evolution and challenges

Two factors determine the efficacy of a PS: wavelength of max absorption and specificity of cell uptake/tissue localization.

An 'ideal' photosensitizer for in vivo application is a maximum light absorption in the red region of the visible spectrum (650–780 nm), to ensure high tissue penetration and avoid interference by endogenous pigments, mainly haemoglobin, while maintain the energy level which is necessary for the generation of cytotoxic species

and $^1O_2$ [3,7]. It is important to note that, although the energy of singlet oxygen corresponds to the energy of 1270 nm, wavelengths longer than 800 nm are rarely used for PDT owing to high scattering in tissues.[3]

Because the half-life of singlet oxygen in biological systems is <0.04 μs, and, therefore, the radius of the action of singlet oxygen is <0.02 μm, tumor localization of the photosensitizer is an important factor that determines PDT efficacy[2]. Drug localization is known to be determined by vascular permeability and interstitial diffusion, which depend on molecular size, configuration, charge, and hydrophilic or lipophilic property of the compound, as well as physiological properties of blood vessels. Binding of the drug with various components of the tissue can also influence its transport and retention in tumors.[2].

As result, over the decades various new generations of photosensitizers were designed. The agent Photofrin, the first-generation PS, had well documented shortcoming, namely a prolonged skin photosensitization in patients up to 4-6 weeks after treatment and weak long-wavelength absorption (630 nm) which translated into poor tissue penetration and energetics.

A second generation of PS was designed with the goal to improve the photochemical characteristic of the first generation; molecules belonging to this generation are for the most parts porphyrins or related molecules such as phthalocyanines. In particular, second-gen PS have high extinction coefficients and quantum yields, strong long-wavelengths absorption (660-700 nm far red and 700-850 nm near infrared bands) and tissue penetration, partial selective tissue accumulation and present minimal toxicity in the absence of light[3].

A third generation is currently in development focusing on the improvement of: i) the poor solubility of previous generation photosensitizers, that prevents their intravenous delivery, ii) the selective targeting of cells and tissue and iii) aggregation phenomena that quench photodynamic properties[1]. This is done by conjugation with carrier molecules such as antibodies directed to tumor-associated antigens or vascular antigens, such as the ED-B domain[2], or Low-Density lipoproteins, sugars and serum albumin[1].

3.3 In silico tools to support photosensitizers research

As it has highlighted in the previous paragraph, the new trends in the development of more performing photosensitizers is the design of novel agents with improved cell and tissue specificity. This has been achieved with the conjugation with carrier biomolecules that have intrinsic targeting abilities, such as antibodies, or that can be functionalized with targeting moieties such as proteins. In addition, the interactions between the PS and the various components of the tissue can have an impact in its retention and subsequent specificity of action.

The widespread in silico drug design tool can be a powerful ally in this regard since it allows to quickly scan the interactions between a single 'fixed' chemical entity and a large library of compounds. Usually the 'fixed' chemical entity is represented by a biomolecule of interest, such as receptor, and the library comprises small active molecules. By reversing this relation, that is, by screening a large structural database of biomolecules against a single small molecule, when can easily identify possible cellular targets for PS and proteins that can serve as carriers.

3.4 General reverse screening algorithm overview

As previously stated, reverse screening consists in screening a library of 3D structures of biomolecules, usually proteins, against the structure of a molecule of interest, such as a drug-like molecule or a small nanomaterial, against a library of biomolecules, usually proteins. It is the exact opposite of the common screening protocol used in In Silico Drug Design where a large library of small compounds (referred to as 'ligands') is screened against a target biomolecule, usually a receptor which plays a key role in biological processes such tumor growth or cell development (regardless of its biological function, biomolecules are called 'receptors' in docking jargon). For the sake of clarity, the molecule that belongs to the structural database that is screened will be called 'ligand' and the molecule that is screened against will be called 'receptor', regardless of their relative sizes.

The reverse docking algorithm is comprised of three main stages that are applied to each biomolecule of the library of targets: *pose generation, pose scoring and ranking*. *Pose generation* is the act of building a reasonable structure of the non-covalent complex between the two interacting partners, called *pose* or *docking candidate*. There are a plethora of methods used to infer how the two partners can interact but all belongs to one of two broad categories: i) *local shape feature matching* and ii) *brute force* enumeration of the transformation space[8], as it has been outlined in the Theory of Computational Methods section. Docking algorithms build many possible structures of the non-covalent complex and discard only those that give rise to steric clashes between atoms.

Pose scoring is the act of inferring the binding energy of a specific complex structure using a mathematical equation called *scoring function*. This step is performed by common docking programs in an approximate, yet fast manner. In the pharmaceutical industry, docking tools are usually employed to filter out the compounds of the library which cannot interact favorably with the target biomolecule and to identify a group of promising compounds; accurate investigation of the binding energy is then carried out at the experimental level only for the latter. Therefore, the scoring function has been designed with speed over accuracy in mind. This step is repeated for each pose inferred in the previous step.

Ranking is the act of creating a scoreboard of the entries in the library according to the score of the best scoring pose.

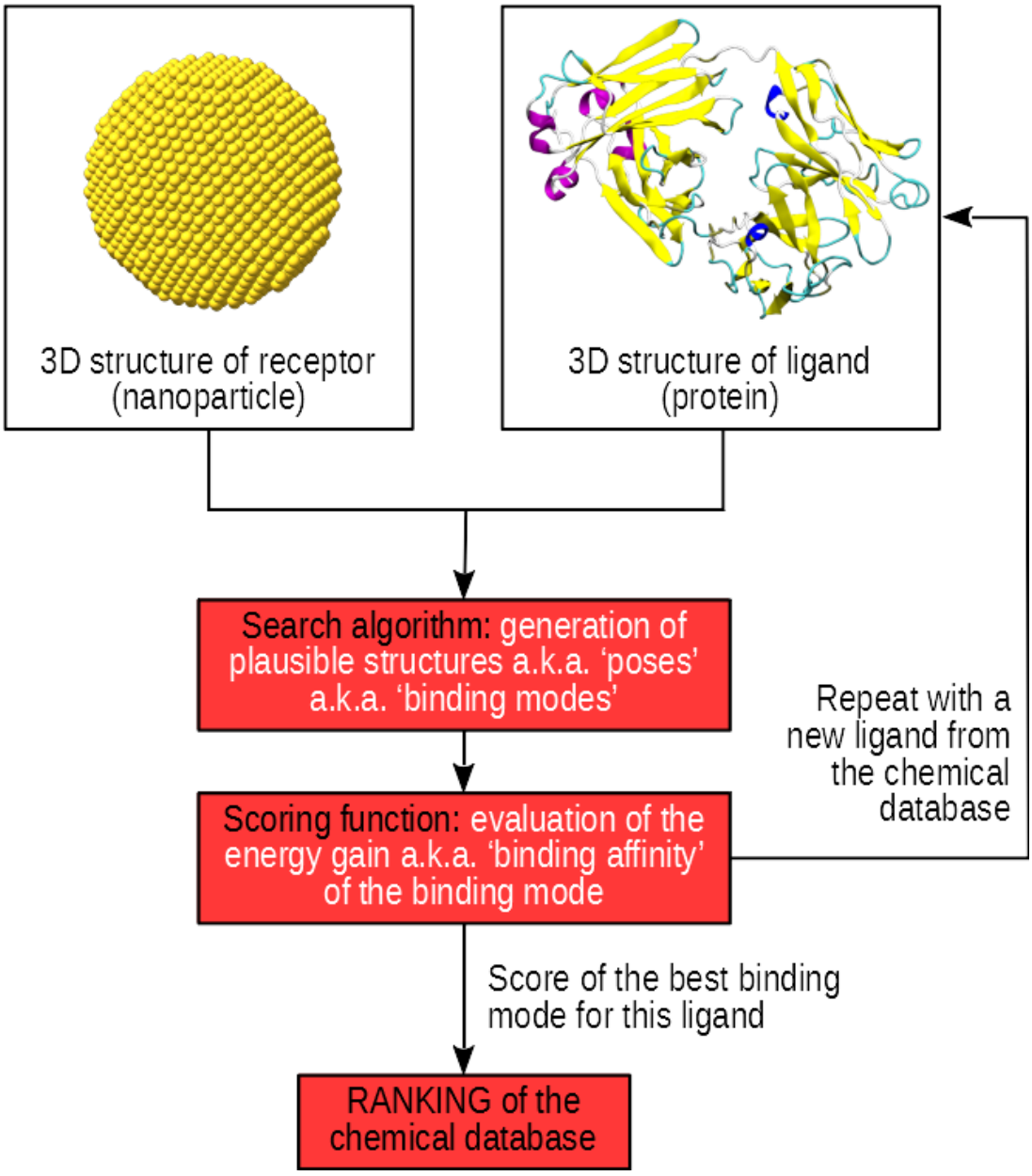


Figure 5 - Graphical representation of the reverse screening algorithm.

One of the main causes of the poor predictive ability of docking techniques is the difficulty to properly consider the flexibility of the two interacting entities, such as the change in conformation of the biomolecule, a phenomenon called *induced fit*. Briefly, since intramolecular non-bonded interactions that govern secondary and tertiary structures of proteins and biomolecular aggregates are fundamentally

identical to the non-bonded interactions that give rise to ligand-receptor complexes, upon binding a ligand changes its conformation, if flexible, and the conformation of the receptor to maximize the interaction energy. The advantage of investigating small nanoparticles is their low flexibility that significantly simplifies the prediction of the docking candidates; as result, it has been possible to accurately predict the binding site of the buckminsterfullerene on lysozyme[9], which was later confirmed by experimental measures[10], by using simple rigid docking tools.

In addition, molecules with a high hydrophobic character are good candidates as well, since electrostatics and polar solvation contributions play a secondary role in the binding process and it is possible to approximate their contributions with simple and fast models, such as the Generalized Born model for solvation.

The simplest form of a reverse screening algorithm consists in using the built-in scoring function of the docking program. Proprietary scoring functions are usually *empirical* or *knowledge-based*, as it has been outlined in the 'Theory of Computational Methods' section, and can perform quite well for peptides or small molecules without an exotic structure or that are similar to common drugs and active compounds.

3.5 Reverse screening of porphine: less is better

Two separate reverse screening protocols were carried out to investigate the interactions between porphine and a database of 3D structures of proteins of potential therapeutic interest, the Potential Drug Targe Database (PDTD[11]), which comprises 1040 unique entries ranging from enzymes to ion channels. This database has already been used in many other reverse screening investigations by our research group and provides a common reference among them.

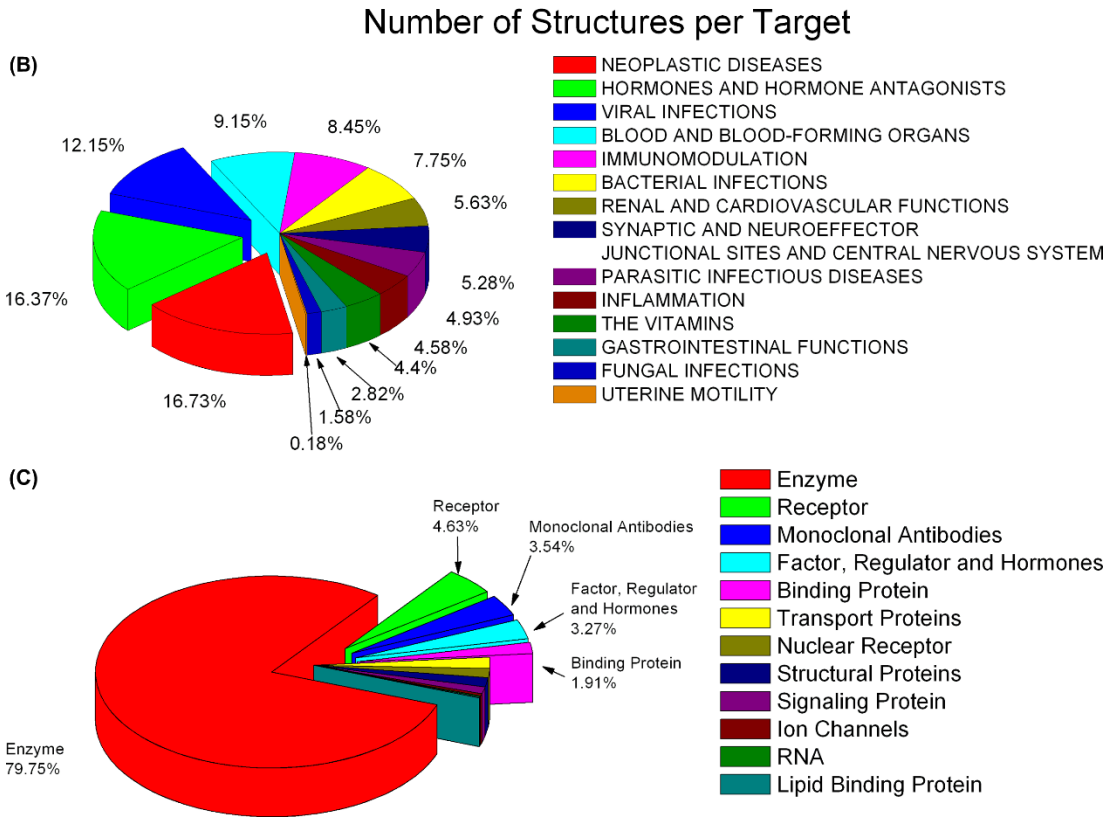


Figure 6 - Functional and biochemical classifications of PDTD protein entries. (B) Distribution of drug targets according to their therapeutic areas. (C) Distribution of drug targets according to their biochemical criteria. [11]

3.5.1 Comparison of the algorithms of the two protocols

Both protocols use the free docking program PatchDock[12] which belongs to the local shape feature matching family of docking programs. Briefly, the peculiarity of this program is that it uses object recognition and image segmentation techniques commonly used in computer vision and 3D graphics to divide the surfaces of the two interacting entities in patches of concave, convex or flat curvature. Complementary patches are matched, the entire structure of the two entities is built according to the orientation of the matched patches and the complex is checked for steric clashes. Flexibility is accounted for only by allowing a certain degree of penetration of the ligand structure into the receptor structure. Although this approximate treatment of the induced fit phenomenon and the ligand flexibility is detrimental in the investigation of common drug-like molecules, it can be easily overcome by using rigid molecules, while retaining the high performance of the local matching algorithm.

The difference between the two protocols resides in the pose scoring step. The first protocol only uses the built-in scoring function of PatchDock which scores the poses according to shape complementarity and the atomic desolvation energy[13]. In the case of highly hydrophobic molecules such as fullerenes van der Waals interactions play the main role in the formation of non-covalent complexes with proteins[14]. Van der Waals interactions are proportionate to the degree of contact between the two interacting molecules, which scales with the shape complementarity between the two structures. A scoring function which takes strongly into consideration shape complementarity is therefore optimal for this class of compounds.

The second protocol improves upon the scoring process of the first one in 2 ways:

i) by optimizing the structure of the binding site of each docking candidate by using Molecular Mechanics energy-minimizations algorithms to take into consideration the induced fit phenomenon;

ii) by re-scoring the newly optimized docking candidates with a forcefield-based scoring function to take into consideration electrostatic contributions as well and improve the treatment of solvation energies. A new scoreboard is subsequently generated based on the score of the best docking candidate for each protein.

Energy minimization is performed for all the residues within 5 Å of every atom of the ligand. After an initial coarse optimization with 100 steps of steepest descent algorithm, 150 steps of conjugate gradient is performed to finely adjust the local protein structure. To lower the computational cost involved in minimizing tens of thousands of structures, minimizations are performed in vacuum, electrostatics are not treated with Particle Mesh Ewald and a 12 Å cut-off is employed for non-bonded interactions.

As scoring function, we chose to employ MM-PBSA method[15]; it is an efficient algorithm that obtains binding energies in a solvated environment from the estimation of the binding energy in vacuum and the solvation energies of the receptor, the ligand and their complex; the latter are computed using an implicit solvent model. Unlike the built-in scoring function of PatchDock which assigns scores with adimensional units, the MM-PBSA protocol estimates a realistic energy component, the binding energy, in Kcal/mol units. Although it lacks the accuracy to

generate absolute values of the binding energy (unlike other, more computationally expensive methods), the MM-PBSA analysis produces an accurate relative scale of the binding interaction across the poses of a single protein and across the whole protein database in a fast, efficient manner which is perfectly suitable for an *in silico* screening protocol. To perform MM-GBSA analysis, we used the Python-based script MMPBSA.py[16] included in the AmberTools16 installation[17]. With the same rationale, we chose to use the Generalized Born method for the implicit solvation treatment instead of the more accurate and computationally demanding Poisson-Boltzmann method. In particular, we chose model II of a pair of modified GB models developed by A. Onufriev, D. Bashford and D.A. Case[18], which agrees better with the Poisson-Boltzmann treatment in calculating the electrostatic part of the solvation free energy. Atom types and atomic partial charges for the proteins are assigned automatically according to the AMBER forcefield *ff14SB*[19] by using the tool *tleap* from the molecular mechanics suite Ambertools16[17]. Ambertools16 is the open-source version of the AMBER16 suite; although it lacks GPU-acceleration support and the high-performance *pmemd* molecular mechanics simulation engine, it supports almost all the types of calculations of the complete version and allows the whole protocol to be completely *open-source* and accessible to scientist in academia and industry alike. Regarding the ligand, while atom types are assigned automatically according to the General Amber ForceField (GAFF)[20] by using the tool *antechamber*, also from the Ambertools16 suite, automatic estimation of partial charges is not reliable. As result, the latter are calculated at the quantum mechanical level in accordance with the method used in the derivation of the AMBER family of forcefields. The ligand structure is optimized with the program Gaussian16[21] at the HF level with the 6-31G* basis set [22] and population analysis using the Merz-Singh-Kollman (MK) scheme [23] is performed. The MK scheme fits atomic charges to reproduce the molecular electrostatic potential (MEP), an observable property of the molecule, at a number of points around atoms of the structure; the charge grid produced during this process is then used by the tool *antechamber* to derive RESP charges. The major weaknessess of common electrostatic potential derived (ESP) charges are conformational dependence and difficult transferability between common functional groups in related molecules; the fitting method used to derive RESP charges allows to overcome these limitations[24].

Based on the scoring obtained by using MM-GBSA, the best pose is identified once again and its binding energy value is used to represent the relative protein inside a new scoreboard.

This round of binding pocket optimization and rescoring is performed only on the proteins that ranks in the first half of the scoreboard generated by the first protocol. Proteins at the bottom of the first scoreboard do not possess any binding pocket with a sufficiently compatible shape. Since shape complementarity plays a key role in the binding process, they cannot reach a good binding interaction even after the optimization and rescoring steps above.

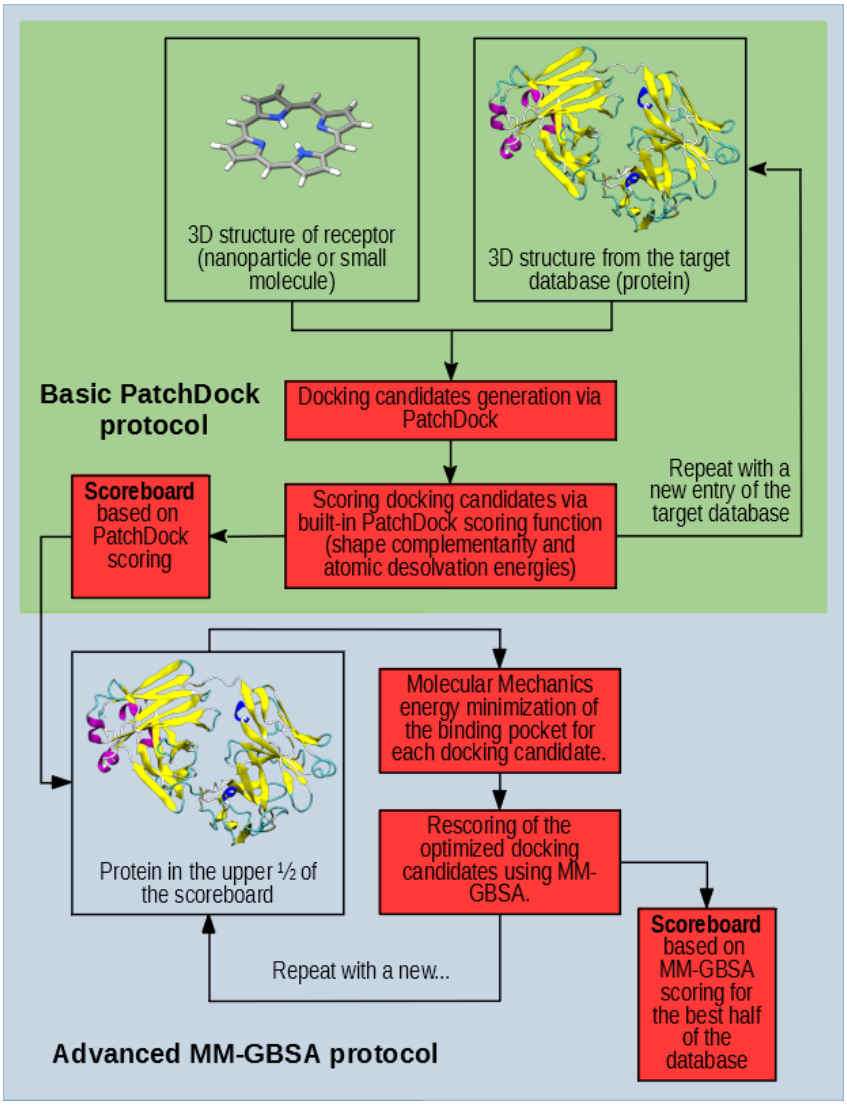


Figure 7 - Graphical representation of the workflows of the two reverse screening algorithms.

Since both algorithms involve large amounts of calculations to be performed and files to be generated for each protein of the database, Bash scripts were written to manage the various parts of the screening and automate the entire investigation apart from the quantum mechanical calculations for the ligand, which were performed beforehand.

3.5.2 Comparison of the reliability of prediction

By applying the two protocols to the PDTD, two scoreboards where obtained. To gauge the accuracy of the two protocols, we checked which part of the hemeproteins of the database (which were stripped of the heme prosthetic group before the screening process) was identified as the best binding pocket for the ligand and how well they ranked in the scoreboard. Although heme groups vary in structure across proteins, they all share a common scaffold: porphine. The cavity of the original heme group was tailored to stabilize the heme porphine scaffold after eons of evolution. An accurate scoring protocol should identify the cavity of the heme group as the favoured binding site of the porphine on the protein and a should position the latter high in the scoreboard. From the comparison of the two resulting scoreboards, it is clear that while the protocol that leverages only on the built-in scoring function correctly places the hemeproteins among the most interacting biomolecules of the database, while the more advanced protocol based on MM-PBSA fails to do so. The basic protocol always selects the heme cavity as the favored binding site on the hemeproteins, while the advanced protocol is not as reliable, as we can see in the case of Escherichia Coli succinate dehydrogenase (PDB identifier: 1NEK) (Figure 4)

	PatchDock score function		MM-GBSA scoring	
	PDB identifier	Protein name	PDB identifier	Protein name
1st	8CAT	Beef liver catalase	1PSO	Pepsin 3A
2nd	1OG5	Cytochrome P450 2C9	1OOQ	Nitroreductase
3rd	2F9Q	Cytochrome P450 2D6	1ZZE	Aldehyde reductase II
4	1SPG	Hemoglobin	1YVB	Falcipain 2
5	1K74	Retinoic acid receptor RXR-α	1WWC	Tropomyosin receptor kinase 2
6	1BVY	Cytochrome P450 BM-3	1D8T	Elongation factor
7	1NEK	Succinate:quinone oxydoreductase	1PHD	Cytochrome P450-CAM
8	1PSO	Pepsin 3A	2B3K	Plasmogen Activator Inhibitor-1
9	1I8O	Purine Nucleoside Phosphorilase	5TLN	Thermolysin
10	1E55	Beta-Glucosidase	1J3H	cAMP-dependent protein kinase,
11	1HN4	Prophospholipase A2	1XOS	cAMP phosphodiesterase 4B
12	1BBP	Bilin binding Protein	1QKM	Estrogen Receptor β
13	1VID	Catechol o-Methyltransferase	2BE1	Ser/Thr-protein IRE1
14	1OIQ	Cell Division Protein Kinase 2	1PPM	Penicillopepsin
15	1QIJ	Acetylcolinesterase	1CTT	CytidineDeaminase
16	1VE9	D-aminoacid oxidase	1G12	Peptidyl Metaloendopeptidase-Lys
17	1PQ2	Cytochrome P450 2C8	1APV	Penicillopepsin
18	1TVR	HIV-1 reverse transcriptase	2ER6	Endothiapepsin
19	2IFB	Intestinal fatty acid binding protein	1D6U	Copper amine oxidase
20	1DIS	Dihydrofolate reductase	1LMO	Lisozyme

Table 1 - First 20 positions of the scoreboards obtained with the two reverse screening protocols.

A shape based scoring function can measure correctly the interaction between ligand and receptor when the molecule is characterized by low polarity and low flexibility, such as porphine. An accurate calculation of electrostatic and solvation contributions is not necessary thanks to the low polarity and the low flexibility of the structure which solves the problem of predicting conformational changes upon binding. The MM-GBSA scoring function for these molecules, is biased by the approximations to lower their computational cost. As consequence, this scoring function yields worse results in the case of molecules that are well represented by less advanced methods.

Following the results of this comparison, the same basic protocol was used for the reverse screening analysis of the PDTD against phthalocyanine, since it is characterized by a similar structure.

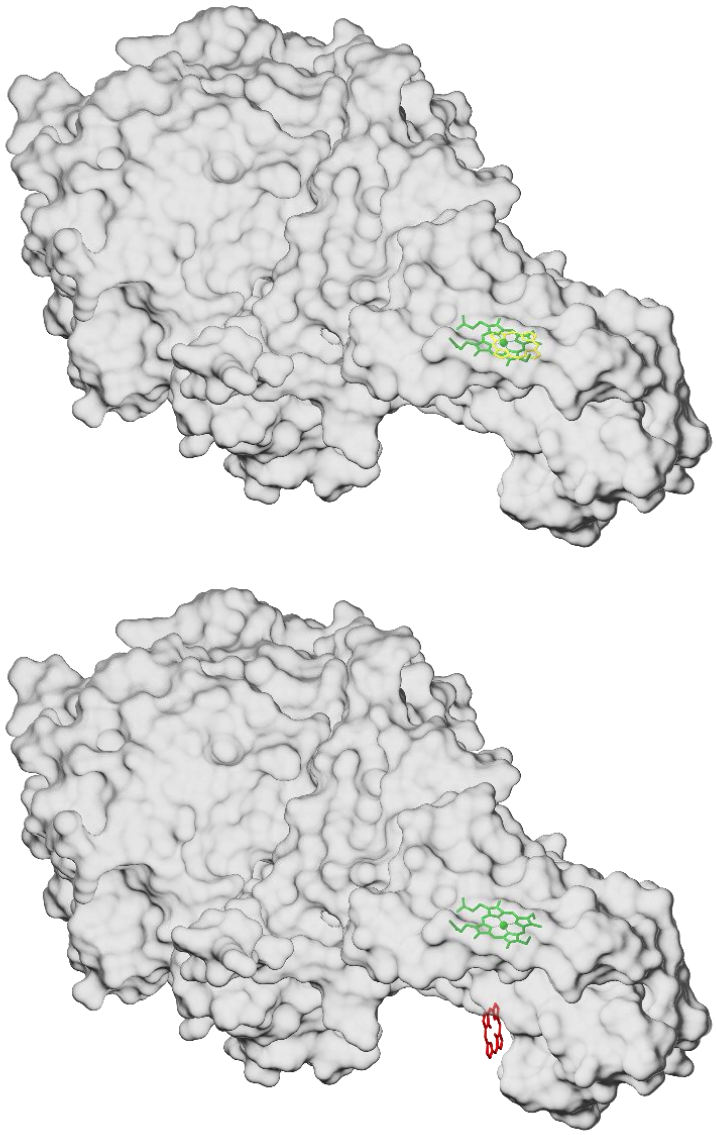


Figure 8 - Comparison of the reliability of the binding site prediction. The protocol A identifies the cavity of the native heme group (green) as the favored binding site of porphine (yellow) on E. Coli succinate dehydrogenase. Protocol B places the favored binding site (red) of the surface of the protein.

3.6 Analysis of the scoreboards: solving old problems and finding new therapeutic targets

Screening the PDTD against porphine and phthalocyanine yielded two relative scales of the interactions between each one of the latter molecules and the proteins in the database. From the analysis of these scoreboards is possible to identify targets for photodynamic therapy or other pharmacological applications.

Every computational chemistry investigation is performed with a certain degree of approximation due to the sheer number of variables involved, which scales with the amount of particles that are considered. This is especially true for screening protocols since they must trade accuracy for speed. For these reasons, only proteins in the top 10% of the scoreboards have been considered as reliable predictions and have been reported in Table 1 and Table 2 of the Appendix A. In the following section, an analysis of the literature on the most interesting proteins from a pharmacological point of view is presented.

3.6.1 Carrier proteins for therapeutic applications

One of the main limitations of common photosensitizing (PS) agents is their high lipophilic character and non-specific cellular and tissue uptake. A solution is to coordinate a PS agent with a carrier protein, which will grant solubility and a favorable ADME profile. Functionalization of the protein with moieties which recognize specific types of cells can grant specific cellular uptake, lowering necessary doses and unwanted side-effects in the process. The best candidates for this role are *carrier-protein*s, which carry hydrophobic substances, such as hormones, in the body and are characterized by a lipophilic binding pocket to accommodate the latter.

Rank 12 and 20 of the porphine scoreboard are occupied by two carrier proteins, albeit employed in different biological processes[25,26]: they are respectively bilin binding protein (BBP) from Pieris brassicae with rank (PDB identifier: 1BBP) and intestinal fatty-acid-binding protein from Rattus Norvegicus (PDB id: 2IFB). In both cases the reverse docking protocol places the porphine molecule in the lipophilic binding pocket of the proteins.

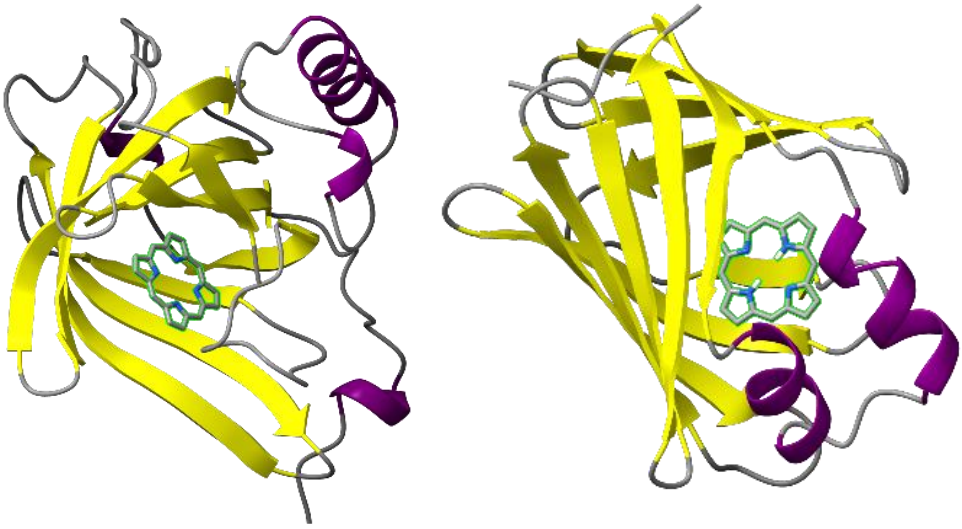


Figure 9 - Left: 1BBP ; Right: 2IFB

Rank 15 of the phthalocyanine scoreboard is occupied by yeast oxysterol binding protein Osh4 (PDB id: 1ZHY). Oxysterol-binding proteins (OSBP) are lipid-binding proteins that are conserved from yeast to humans. They are implicated in many cellular processes, among which the regulation of the homeostasis of sterol, thanks to a hydrophobic pocket that binds a single sterol molecule[27]. The lipophilic binding pocket is once again identified as the favored binding site of phthalocyanine.

Figure 10 - 1ZH1

3.6.2 Targets for photodynamic therapy

Starting with the porphine scoreboard, rank 14 is occupied by the human cyclin-dependent kinase 2, also known as 'cell division protein kinase 2' or CDK2 (PDB id: 1OIQ). The eukaryotic cell cycle is characterized by checkpoints to ensure its correct progression These checkpoints are implemented through the regulation of the activity of cyclin-dependent kinase. Cyclin-dependent kinase 2 (CDK2) regulation determines the progression into the S- and M-phases of the cell cycle and it is critically associated with tumor growth in a number of cancer types.[28]

Figure 11 - 1OIQ

Rank 19 of the porphine scoreboard is taken by the human immunodeficiency virus type 1 (HIV-1) reverse transcriptase (RT), an important target for chemotherapeutic agents used in the treatment of Acquired Immune Deficiency Syndrome (AIDS)[29] (PDB id: 1TVR). HIV is a retrovirus, therefore uses its reverse transcriptase protein to translate its RNA genetic material into DNA; the latter is integrated into the host cell genome and read along the original DNA strands, leading to the replication of the virus. Irradiation of porphine while complexed with the HIV RT protein can lead to the neutralization of the latter, effectively blocking virus replication.

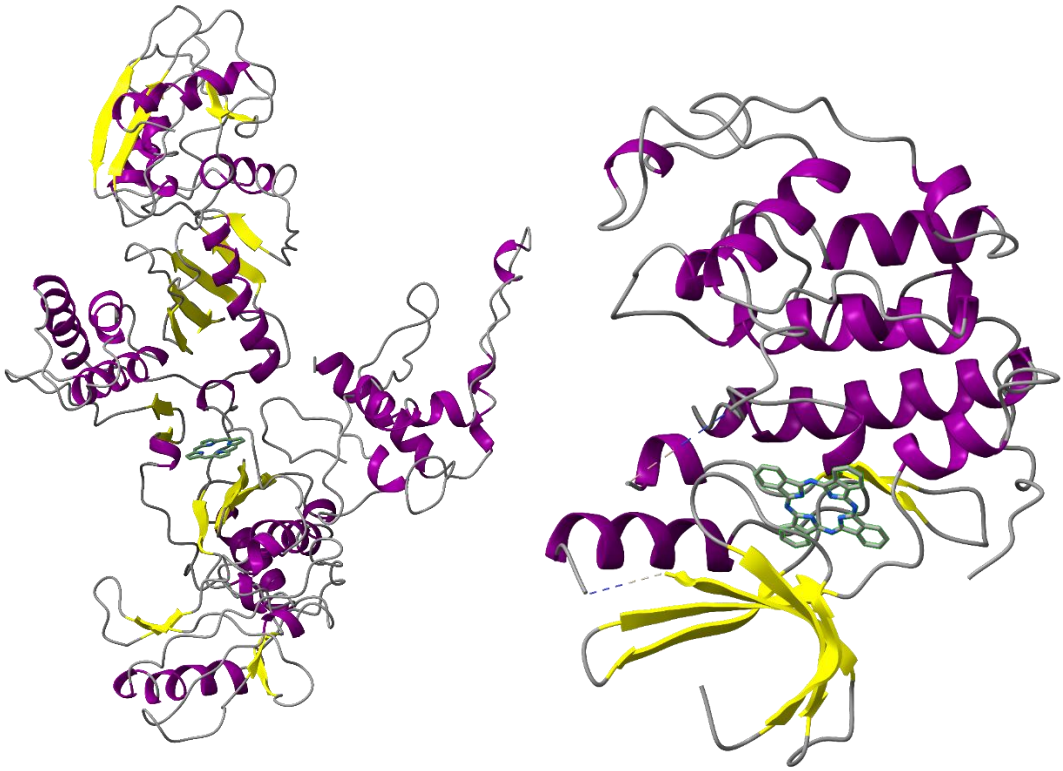


Figure 12 - Left: 1TVR ; Right: 1GII

Moving to the phatlocyanine scoreboard, rank 37 is taken by of by a mimic of the cyclin-dependent kinase 4 (PDB id: 1GII). Since the CDK family of proteins has a high structural homology, a CDK2 protein was synthesized with the ATP binding pocket of CDK4 in place of its native one to help in the design of CDK4-specific inhibitors. Genetic alteration of one or more components of the p16(INK4A)-CDK4,6/cyclin D-retinoblastoma pathway is found in more than half of all human cancers, making CDK4 a promising target for anticancer drugs[30]. CDK4 has additional space inside its ATP binding pocket, making molecules with large substituents selective against it instead of CDK2. As result, CDK4 is not in the top 10% portion of the porphine scoreboard, while CDK2 ranks only 65th in the phthalocyanine scoreboard. These results suggest that phthalocyanine could be employed as a selective PDT agent against CDK4, while porphine might selectively target CDK2.

Ranks 34 and 38 are occupied respectively by cyclooxygenase-1, also known as COX-1, (PDB id:1CQE) and cyclooxygenase-2, also known as COX-2 (PDB id: 1CX2), two isoforms of the membrain protein cyclooxygenase. While COX-1 is constitutively expressed in most tissue and is responsible for the physiological production of

prostaglandins, COX-2 expression is induced by inflammatory conditions and is responsible for a spike in production of prostaglandins[31]. High expression of COX-2 is also characteristic of human tumor neovasculature and of neoplastic cells present in human colon, breast, prostate, and lung cancer tissue[32]; furthermore, inhibition of COX-2 by the drug celecoxib has been demonstrated to suppress growth of colon and lung tumors [32], suggesting that COX-2 can be an effective target for tumor photodynamic therapy.

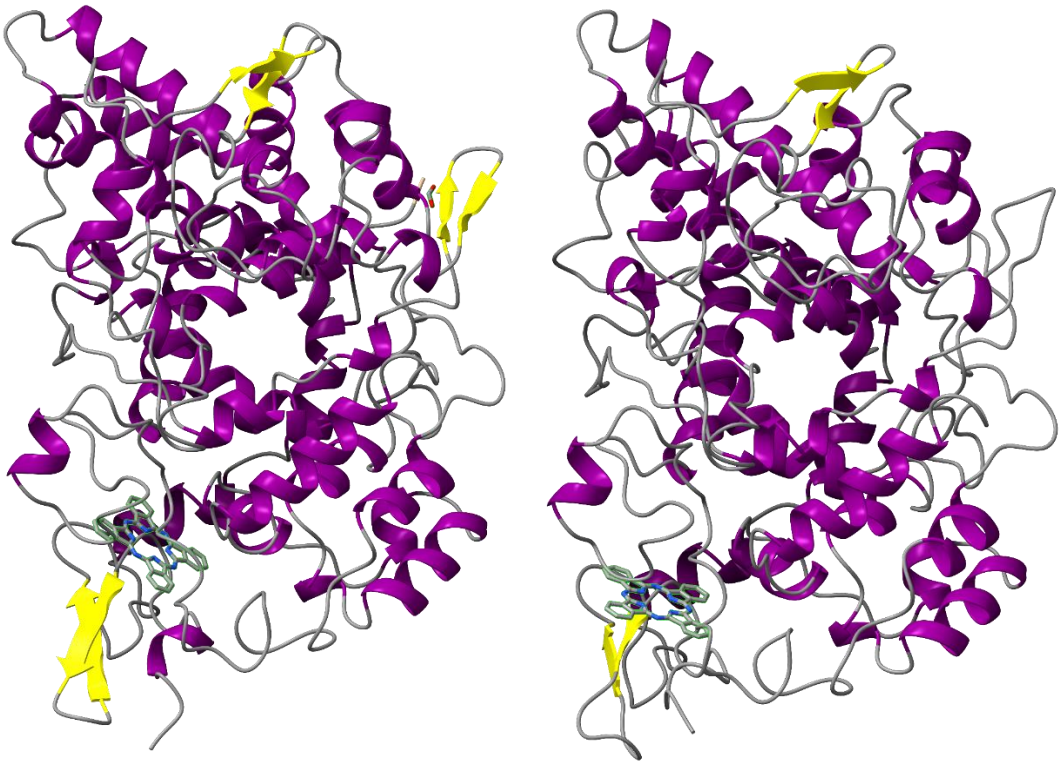


Figure 13 - Left: 1CQE ; Right 1CX2

3.7 References

[1] L. B. Josefsen, R. W. Boyle, *Theranostics* **2012**, *2*, 916–966.

[2] D. E. J. G. J. Dolmans, D. Fukumura, R. K. Jain, *Nature Reviews Cancer* **2003**, *3*, 380–387.

[3] L. M. Moreira, F. V. dos Santos, J. P. Lyon, M. Maftoum-Costa, C. Pacheco-Soares, N. S. da Silva, *Aust. J. Chem.* **2008**, *61*, 741–754.

[4] J. Piette, C. Volanti, A. Vantieghem, J.-Y. Matroule, Y. Habraken, P. Agostinis, *Biochem Pharmacol* **2003**, *66*, 1651–1659.

[5] D. Nowis, T. Stokłosa, M. Legat, T. Issat, M. Jakóbisiak, J. Gołąb, *Photodiagnosis Photodyn Ther* **2005**, *2*, 283–298.

[6] N. L. Oleinick, R. L. Morris, I. Belichenko, *Photochemical & Photobiological Sciences* **2002**, *1*, 1–21.

[7] J. P. Celli, B. Q. Spring, I. Rizvi, C. L. Evans, K. S. Samkoe, S. Verma, B. W. Pogue, T. Hasan, *Chem. Rev.* **2010**, *110*, 2795–2838.

[8] D. Duhovny, R. Nussinov, H. J. Wolfson, in *Algorithms in Bioinformatics* (Eds.: R. Guigó, D. Gusfield), Springer, Berlin, Heidelberg, **2002**, pp. 185–200.

[9] M. Calvaresi, F. Zerbetto, *ACS Nano* **2010**, *4*, 2283–2299.

[10] M. Calvaresi, F. Arnesano, S. Bonacchi, A. Bottoni, V. Calò, S. Conte, G. Falini, S. Fermani, M. Losacco, M. Montalti, G. Natile, L. Prodi, F. Sparla, F. Zerbetto, *ACS Nano* **2014**, *8*, 1871–1877.

[11] Z. Gao, H. Li, H. Zhang, X. Liu, L. Kang, X. Luo, W. Zhu, K. Chen, X. Wang, H. Jiang, *BMC Bioinformatics* **2008**, *9*, 104.

[12] D. Schneidman-Duhovny, Y. Inbar, R. Nussinov, H. J. Wolfson, *Nucleic Acids Res* **2005**, *33*, W363–W367.

[13] C. Zhang, G. Vasmatzis, J. L. Cornette, C. DeLisi, *Journal of Molecular Biology* **1997**, *267*, 707–726.

[14] M. Calvaresi, A. Bottoni, F. Zerbetto, *J. Phys. Chem. C* **2015**, *119*, 28077–28082.

[15] J. Srinivasan, T. E. Cheatham, P. Cieplak, P. A. Kollman, D. A. Case, *Journal of the American Chemical Society* **1998**, *120*, 9401–9409.

[16] B. R. Miller, T. D. McGee, J. M. Swails, N. Homeyer, H. Gohlke, A. E. Roitberg, *J. Chem. Theory Comput.* **2012**, *8*, 3314–3321.

[17] D. A. Case, R. M. Betz, D. S. Cerutti, T. E. C. III, T. A. Darden, R. E. Duke, T. J. Giese, H. Gohlke, A. W. Goetz, N. Homeyer, S. Izadi, P. Janowski, J. Kaus, A. Kovalenko, T. S. Lee, S. LeGrand, P. Li, C. Lin, T. Luchko, R. Luo, B. Madej, D. Mermelstein, K. M. Merz, G. Monard, H. Nguyen, H. T. Nguyen, I. Omelyan, A. Onufriev, D. R. Roe, A. Roitberg, C. Sagui, C. L. Simmerling, W. M. Botello-Smith, J. Swails, R. C. Walker, J. Wang, R. M. Wolf, X. Wu, L. Xiao, P. A. Kollman, *AMBER 16 and AmberTools16*, University Of California, San Francisco, **2016**.

[18] A. Onufriev, D. Bashford, D. A. Case, *Proteins* **2004**, *55*, 383–394.

[19] J. A. Maier, C. Martinez, K. Kasavajhala, L. Wickstrom, K. E. Hauser, C. Simmerling, *J. Chem. Theory Comput.* **2015**, *11*, 3696–3713.

[20] J. Wang, R. M. Wolf, J. W. Caldwell, P. A. Kollman, D. A. Case, *Journal of Computational Chemistry* **2004**, *25*, 1157–1174.

[21] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, D. J. Fox, *Gaussian 16*, Gaussian Inc., Wallingford CT, **2016**.

[22] G. A. Petersson, A. Bennett, T. G. Tensfeldt, M. A. Al-Laham, W. A. Shirley, J. Mantzaris, *J. Chem. Phys.* **1988**, *89*, 2193–2218.

[23] U. C. Singh, P. A. Kollman, *Journal of Computational Chemistry* **1984**, *5*, 129–145.

[24] C. I. Bayly, P. Cieplak, W. Cornell, P. A. Kollman, *J. Phys. Chem.* **1993**, *97*, 10269–10280.

[25] R. Huber, M. Schneider, I. Mayr, R. Müller, R. Deutzmann, F. Suter, H. Zuber, H. Falk, H. Kayser, *J Mol Biol* **1987**, *198*, 499–513.

[26] J. C. Sacchettini, J. I. Gordon, L. J. Banaszak, *J Mol Biol* **1989**, *208*, 327–339.

[27] S. Raychaudhuri, W. A. Prinz, *Annu Rev Cell Dev Biol* **2010**, *26*, 157–177.

[28] S. Tadesse, E. C. Caldon, W. Tilley, S. Wang, *J Med Chem* **2019**, *62*, 4233–4251.

[29] H. C. Castro, N. I. V. Loureiro, M. Pujol-Luz, A. M. T. Souza, M. G. Albuquerque, D. O. Santos, L. M. Cabral, I. C. Frugulhetti, C. R. Rodrigues, *Curr Med Chem* **2006**, *13*, 313–324.

[30] M. Ikuta, K. Kamata, K. Fukasawa, T. Honma, T. Machida, H. Hirai, I. Suzuki-Takahashi, T. Hayama, S. Nishimura, *J. Biol. Chem.* **2001**, *276*, 27548–27554.

[31] R. G. Kurumbail, A. M. Stevens, J. K. Gierse, J. J. McDonald, R. A. Stegeman, J. Y. Pak, D. Gildehaus, J. M. Iyashiro, T. D. Penning, K. Seibert, P. C. Isakson, W. C. Stallings, *Nature* **1996**, *384*, 644–648.

[32] J. L. Masferrer, K. M. Leahy, A. T. Koki, B. S. Zweifel, S. L. Settle, B. M. Woerner, D. A. Edwards, A. G. Flickinger, R. J. Moore, K. Seibert, *Cancer Res* **2000**, *60*, 1306–1311.

4. Gadofullerenes

4.1. Fullerenes: geometry, redox and optical properties

Fullerenes are one of the carbon allotropes and consist of a cage of carbon atoms arranged in 6- and 5-members ring and characterized by high symmetry. They were first discovered in 1985 by Kroto et al.[1] who characterized the first member of this family (C_{60}-Ih), also known as 'buckminsterfullerene', which is the smallest fullerene to obey the isolated pentagon rule (IPR)[2]; IPR states that a fullerene structure is stable if each pentagon is completely surrounded by hexagons. The carbon atoms are conjugated and electrons are delocalized on the whole surface of the cage. Compared to a graphene sheet, the curvature of the molecule causes the C-atoms to be pyramidalized and rehybridization of the sp2 σ and π orbitals occurs[3]. As result, orbital in fullerenes exhibit significant s-character and extend further outside than inside the carbon cage[4]. This also applies to the low lying π* orbitals, resulting in high electron affinity. Coupled with the energy strain release that reduction reactions brings to the structure, since carbanions favor pyramidalized geometries, C_{60} can perform up to six successive reversible one-electron reductions.

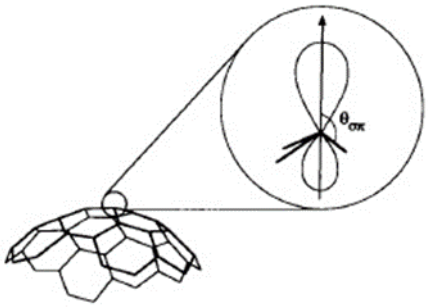


Figure 1 - π-orbital axis vector in POAV analysis, highlighting how π-orbitals stretch utside the carbon cage[4].

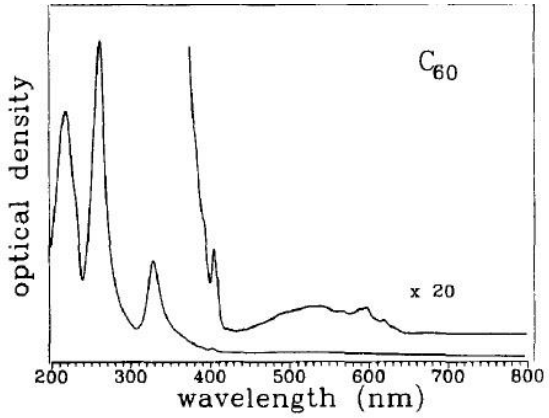


Figure 2 - UV-Vis absorption spectra of C_{60} in hexane[5].

Fullerenes absorb strongly in the UV and moderately in the visible regions[5]. The predominant decay mode is intersystem crossing to triplets and the process proceeds with quantum yields of almost unity[6]. Triplet C_{60} can be reduced to $C_{60}^{\cdot -}$ in the presence of an electron donor or be quenched by interacting to molecular oxygen $^3O_2$, which is converted by energy transfer to singlet oxygen $^1O_2$ in the process[7].

4.2. Filling the void: endohedral metallofullerenes

Atoms belonging to group 2 and 3 of the lanthanide series can be encapsulated in the spherical void inside the carbon cage during fullerene synthesis, resulting in endohedral metallofullerenes[8]. Encapsulation is more common for bigger fullerenes, especially C_{82}.

Studies have demonstrated that the metal atoms are centered in the cage but positioned close to the carbon cage, due to a strong metal-cage interaction[8]. Substantial electron transfer was confirmed to take place from the encaged metal atom to the carbon cage, a phenomenon known as 'intrafullerene electron transfer'. Electron Spin Resonance studies conducted by Johnson et al.[9] on $La@C_{82}$ concluded that this electron transfer caused the La atom in the cage to have 3+ charge, yielding a formal state charge of $La^{3+}@C_{82}^{3-}$. A consequence of this phenomenon is that the IPR in not respected in every endohedral metallofullerene species[8].

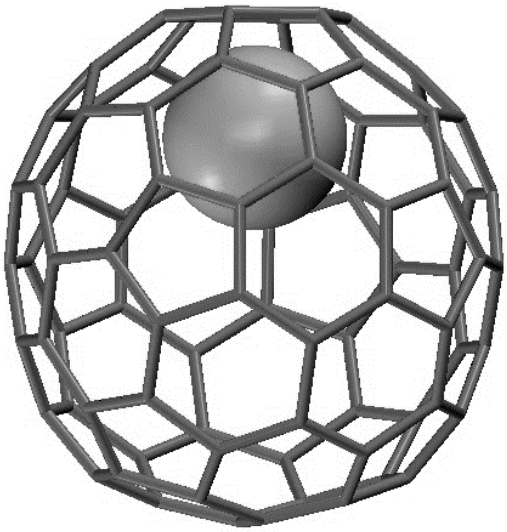


Figure 3 - Gd@C82

Intrafullerene electron transfer also changes the absorption in the UV-Vis-NIR spectrum with respect to their empty fullerene analogs. While absorption of empty fullerenes is relatively weak in the long wavelength portion of the visible spectrum and in the NIR region, metallofullerenes have long tails down to 1500 nm characterized by peaks that may be related to intrafullerene electron transfers from the endohedral metal atom to the cage[8].

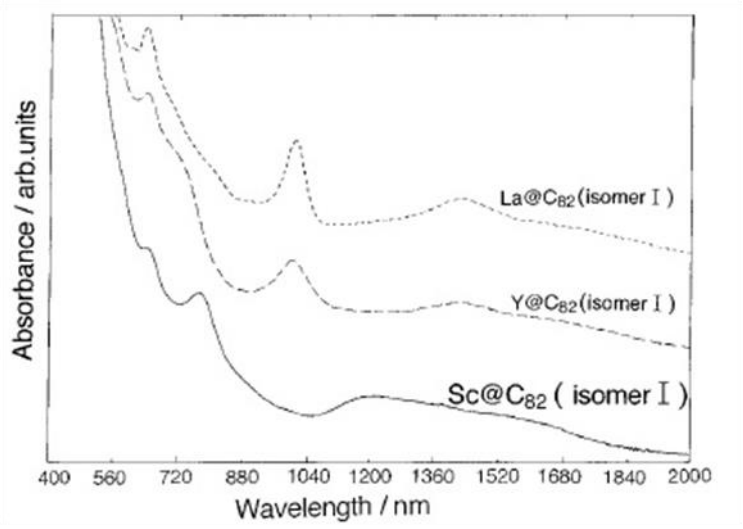


Figure 4 - UV-Vis-NIR absorption spectra for M@C82 molecules[8].

4.3. Therapeutical applications

Fullerenes and their derivatives found application in many fields. In particular, they have been studied extensively in nanomedicine for therapeutic or imaging techniques in the fight against cancer[10].

As already explored in the previous chapter of this work, photodynamic therapy is a promising technique that in the right conditions allows to destroy cancer tissues while eliciting minimum damage to the healthy surroundings. Thanks to their properties, fullerenes can act as efficient photosensitizing agents. Contrary to common PS agents, while singlet oxygen is effectively generated in non-polar environments (Type I process), the electron transfer mechanism (Type II process) is favored in physiological conditions. Guanosine [11] and other reducing agents can act as donors and reduce the fullerene to the radical anion that generates superoxide anion radical $O_2^{\bullet -}$ by interaction with molecular oxygen. $O_2^{\bullet -}$ is converted to hydroxyl radical ·OH, that is responsible for the DNA-cleavage activity of fullerenes[12].

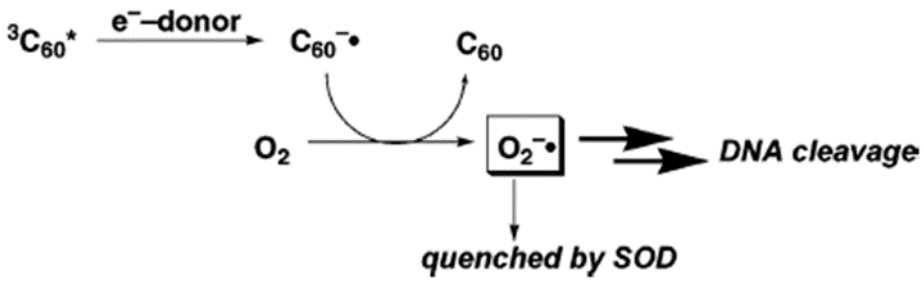


Figure 5 - DNA-cleavage mechanism of photoexcited C_{60}[12].

Certain types of fullerene can be used alongside chemotherapeutic agents to enhance their action. By mediating the penetration of the antitumoral drug through the cancer cell membrane, $[Gd@C_{82}(OH)_{22}]_n$ has been shown to suppress the growth of cisplatin-resistant tumors following cisplatin inoculation both *in vitro* and *in vivo* experiments [13].

In addition, it has been reported that certain functionalized fullerenes can inhibit the tumor lifecycle through several mechanisms. $Gd@C_{82}(OH)_{22}$ can elicit beneficial immune system response [14] and can downregulate more than 10 angiogenic factors, decreasing tumor microvessels density by more than 40% after a two-week treatment in mice[15]. Similar downregulation results were obtained with $C_{60}(OH)_{20}$ as well [16]. Thanks to their high strain energy and LUMO orbitals projected to the outside of the carbon cage, fullerenes are the world's most efficient radical scavengers [11] and can be used to neutralize reactive oxygen species, that during carcinogenesis damage cellular structures directly and promote tumor-associated angiogenesis [16]. Moreover, inoculation of C_{60} nanocrystal water suspension causes the appearance of autophagic features in HeLa cells [17].

Fullerenes can also be used as a combined photothermal and photoacoustic agent thank to the high symmetry of the structure. By irradiation with low-intensity ($< 10^2$ W cm^{-2}) continuous-wave NIR, carbon cage structure of certain fullerenes species can be distorted resulting in heating to the ignition temperature, a phenomenon called 'photothermal ablation'. However, according to studies conducted by Krishna et al. [18] the resulting damage of the tumor tissue can be explained only with a synergy with the phenomenon known as 'acoustic explosion', which has been reported for Single Walled Carbon Nanotubes as well [19]. Briefly, expansion during heating by irradiation and the following compression translated in pressure

differences in the physiological medium, effectively creating strong shockwaves that can damage cell structures.

4.4. Application in diagnostics and imaging

Gadofullerenes are a promising alternative to common commercial contrast agents. Magnetic resonance imaging (MRI) is a noninvasive diagnostic technique that provides physiological and anatomical information. It must be used in conjunction with contrast agents must be to improve sensitivity and resolution to better distinguish diseased tissues from healthy tissues [10]. The MRI signal is enhanced by using paramagnetic metal ions, especially Gd^{3+}, which interact with the water molecule in physiological mediums to reduce proton relaxation times. In commercially available contrast agents, the metal ion is coordinated to polydentate ligands, such as diethylenetriamine-penta-acetic acid (DTPA), to stabilize it and prevent its release *in vivo*, which would poison of the patient. Even though great advances have been made in the design of safe contrast agents, ion release in vivo has not been solved entirely. The use of gadofullerenes, that is, endohedral metallofullerenes that have a Gd atom trapped inside the carbon cage, represent a solution to this problem since the metal ions cannot escape the carbon cage under any circumstance. In addition, gadofullerenes induce stronger proton relaxivity than chelated contrast agents thanks to intrafullerene electron transfer, which translates achieving the same level of contrast while using lower concentrations [20]. In particular, $Gd@(C_{82})(OH)_n$ has a very strong ability of reducing proton relaxation times T_1 and T_2 both *in vivo* and *in vitro* and the observed r_1 values are more than 20 times higher than those of Gd-DTPA (commercially known as Magnevist)[20].

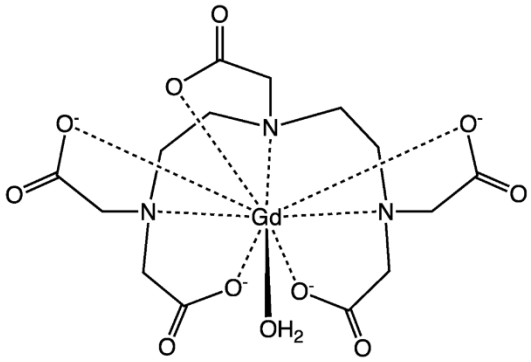


Figure 6 - Gd-DTPA contrast agent, marketed as Magnevist by Bayer Schering Pharma.

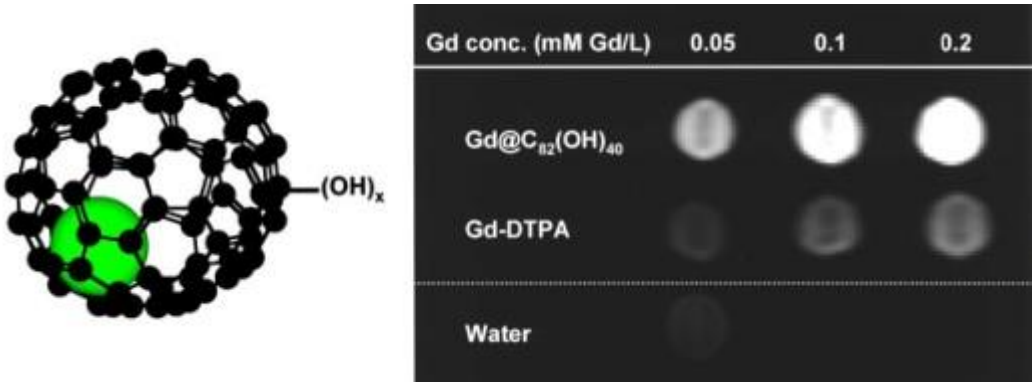


*Figure 7 – **Left**: $Gd@C_{82}(OH)_n$; **Right**: TI-weighted MRI of $Gd@C_{82}(OH)_{40}$ and Gd-DTPA phantom [10].*

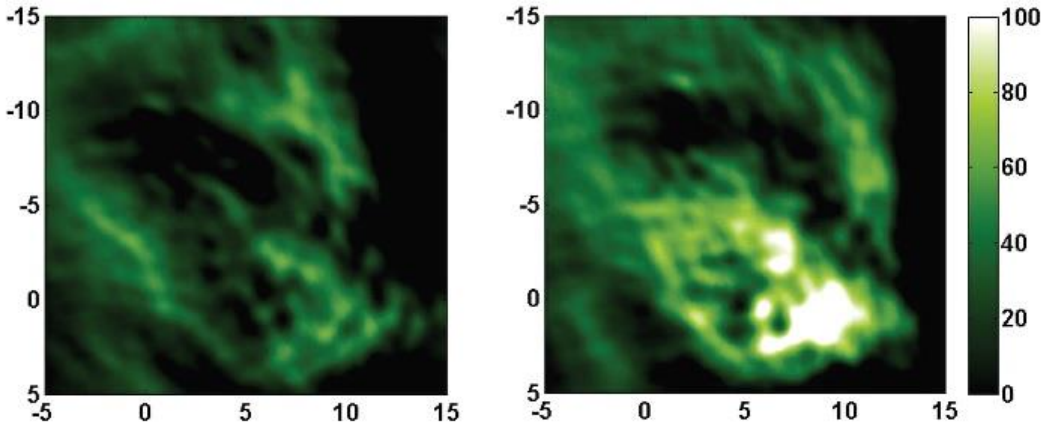


Figure 8 - Photoacustic imaging produces images with excellent contrast between tumor regions and healthy tissues on tumor-bearing nude mice after intratumoral injection of polyhydroxylated fullerenes (PHF) and PHF-containing nanoparticles and following irradiation with a 785 nm pulsed laser [18].

The phenomenon of photoacoustic explosion can also be exploited for imaging purposes if kept under control. Irradiating certain functionalized fullerenes with the same low-intensity (< 10^2 W cm^{-2}) near infrared (NIR) laser used in the photothermal ablation and photoacoustic explosion technique but in a pulsed modality leads to a faint pop once the irradiation is interrupted [18]. These acoustic waves, which are generated by thermoelastic expansion, can be detected using ultrasonic transducers to produce images. Sound scatters 1000 times lesser than light the acoustic signal propagates much longer in biological issue without significant attenuation [21], leading to excellent sensitivity.

4.5. Obstacles to the application in clinical settings

Widespread application of fullerenes and gadofullerenes in nanomedicine is held back by a series of physico-chemical and pharmacokinetics shortcomings. In particular:

- Poor water solubility and low biocompatibility;
- Physiological environment and aggregation phenomena affect their pharmacologically useful properties and toxicity;
- Non-specific cellular uptake.

Several methodologies have been employed to enhance hydrophilicity and overcome this limitation were developed and the most broadly used are: i) encapsulation or micro-encapsulation in special carriers like cyclodextrins, micelles and liposomes, ii) water suspensions produced with the help of co-solvents, defined as nano or colloidal fullerenes (nC_{60}), iii) chemical functionalization of the cage surface with hydrophilic groups [11]. However, commercially available fullerenes and endohedral metallofullerenes, either functionalized or in suspension, still tend to aggregate in water and form clusters of nanoparticles instead of remaining monomolecularly disperse [22,23].

Aggregation due to the poor water solubility can deactivate fullerenes in regard to their therapeutic activity, such as in PDT [10]. Another study suggested that the degree of dispersion affected the antioxidant potential of C_{60} [24], a hypothesis later confirmed by Yin et al., who concluded that the extend of aggregation and radical-scavenging capability were correlated [22].

Regarding fullerene toxicity and its correlation to surface functionalization and aggregation, caution must be exercised in the analysis of the literature since translating the correlation deducted from *in vitro* studies to *in vivo* effects might not be possible. This has been shown by two studies by Sayes et al., one *in vitro* [25] and one *in vivo* [26], which reached opposite conclusions. In addition, there is a difference between the *in vitro* models and the and the *in vivo* model with respect to the target organ under investigation, which suggest that fullerenes' toxicity might depend on the tissue or cells targets [27]. Regardless, a growing body of evidence suggest that fullerene toxicity is correlated to aggregation phenomena. *In vitro* studies showed

that fullerenes enhance the production of proinflammatory mediators such as cytokines TNF-α and IL-8 [27]; the results show that concentration, surface derivatization and the biological environment, which determine the degree of aggregation of the molecule, play a role in the inflammatory potential of fullerenes[27]. *In vitro* studies showed that nC_{60} exerts cytotoxicity even at very low concentrations through enhanced ROS production, subsequent lipid peroxidation and cell membrane damage while PHF, which are less prone to aggregation, are incapable of stimulating ROS production [25]. Genotoxicity studies *in vitro* showed that colloidal suspensions with bigger clusters caused more severe DNA damage than suspensions with smaller cluster sizes, which cause damage nonetheless [28]; the authors however concluded that other factors other than cluster size play a role, such as the amount of molecular hydrated $C_{60}\cdot(H_2O)_n$, that might leak trough the cell membrane and damage DNA directly by redox reactions. Another study by Lyon et al. reached similar conclusions [29].

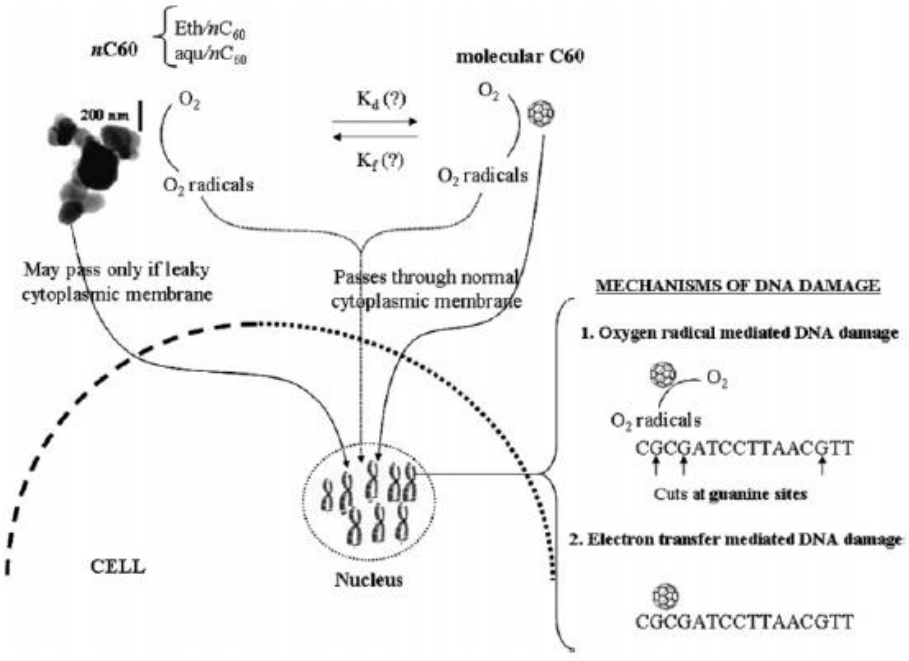


Figure 9 - Proposed mechanism of genotoxicity of $_nC_{60}$ [28].

Although fullerenes are not characterized by specific cellular and tissue uptake, the degree of lipophilicity affects in which organ they mainly accumulate and how fast they are cleared from the body. When injected intravenously, more lipophilic species, like the fullerene derivative from the study conducted by Yamago et al. [30], tend to

accumulate mainly in the liver, followed by kidney, spleen and lungs, and show high retention rates.

More hydrophilic derivatives, such as PHF, have a wider tissue distribution following intravenous injection and are quickly distributed to all organs, mainly in kidneys, liver and bone, and more than half the dose is secreted through urine within 72 h [31]. In most studies, the uptake in brain was found to be negligible [10].

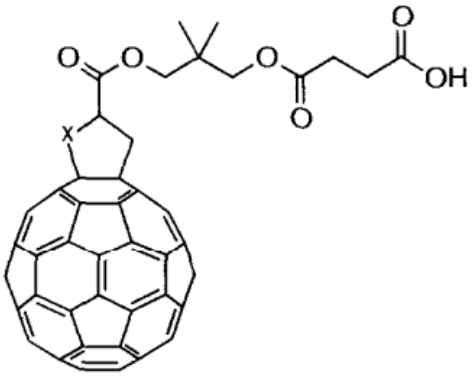


Figure 10 - The water soluble investigated by Yamago et al. The derivative accumulated mainly in the liver with a peak dose of 91.7 % after 16 h and after 160 h only 5.4% was eliminated in the feces [30].

4.6. A new solution: protein carriers

Proteins have shown potential as In recent times conjugating fullerenes with proteins has been proposed as a new way to tackle the limitations presented in the previous paragraph. The first demonstration of the ability of fullerenes to interact with proteins dates back in 1993: following the report by Schinazi et al. that water-soluble fullerene derivatives exhibited antiviral activity against HIV without eliciting cytotoxicity [32], Friedman et al. demonstrated that the inhibition was caused by the insertion of the fullerene molecule into the active site of the HIV protease. The active site of the enzyme has a cylindrical shape with an inner radius comparable to the radius of C_{60} and it is lined almost exclusively with hydrophobic residues, which results in a strong hydrophobic interaction with the carbon cage [33].

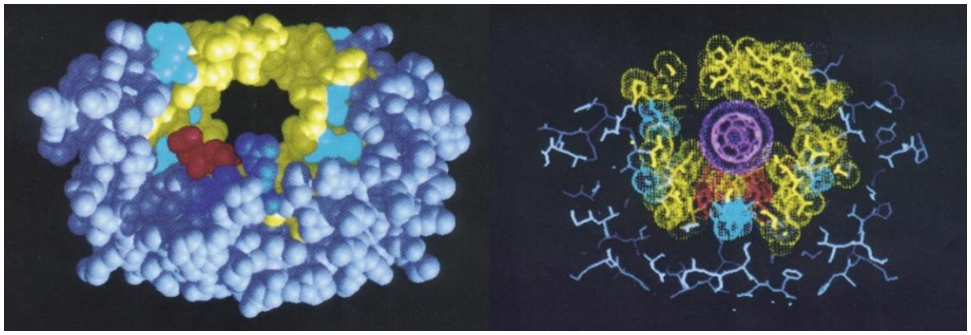


*Figure 11 - **Left**: "Front" view of HIV-1 protease; **Right**: same view with the top scoring C60 orientation shown. Hydrophobic residues in the active site are colored yellow [33].*

More recently, Calvaresi et el. demonstrated with ^{1}H-^{15}N NMR experiments that proteins can form a stochiometric 1:1 adducts with C_{60}, preventing the process of aggregation and granting solubility in aqueous solutions [34]. Proteins have already been proposed as carrier systems for active molecules[35]. Ligand-binding proteins bind their designated ligands with high selectivity and affinity and their binding properties are affected by environmental stimuli and conditions, leading to precise controlled release; these are all properties of the ideal carrier system. Formation of adducts with the right protein can solve the intrinsic limitation of fullerenes and gadofullerenes, paving the road to their application in clinical settings.

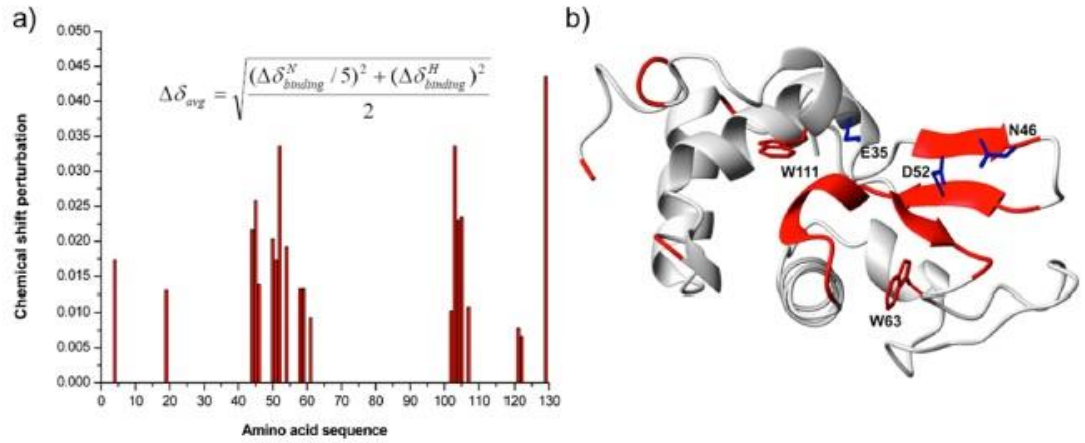


Figure 12 - NMR chemical shift perturbation analysis of LSZ upon interaction with C_{60}. a) Weighted average chemical shift differences of cross-peaks in the ^{1}H, ^{15}N HSQC spectra of free and bound LSZ. b) 3D representation of the residues undergoing chemical shift changes (red region) upon C_{60} binding [34].

In both cases, *in silico drug design* tools drove the discovery. Friedeman et al. used the virtual docking program *DOCK3*[36] to predict the best binding mode of C_{60} on the HIV protease, which was later confirmed by experimental evidence. Calvaresi et al. discovered the favorable interaction between lysozyme and C_{60} by first using the basic reverse screening docking protocol based on *PatchDock* [37], that was introduced in the previous chapter, to screen the PDTD [38] against the C_{60} structure [39]. The protocol correctly predicted that the active site of the protein would be the favored binding site of C_{60}.

The reverse screening docking protocols outlined in the previous chapter therefore were used to identify suitable protein carrier and theranostic targets for Gd@C_{82}, the smallest stable and most studied gadofullerene.

4.7. Gadofullerenes: more is better

In the previous chapter we concluded that for drug-like, rigid, hydrophobic molecules, such as the photosensitizers porphine and phthalocyanine, the built-in scoring function of the docking software PatchDock was more effective in predicting the correct binding site.

4.7.1. Pristine fullerenes and PatchDock: a match made in heaven

Fullerenes possess an exotic molecular structure that differentiates strongly from a typical drug-like molecule, so predictions of their mode of interaction based on experimental data of small active molecule or structural knowledge of their adducts with proteins should have poor quality. However, they are characterized by high rigidity and a very strong lipophilic character, since they do not possess fixed partial charges thank to the extreme delocalization of the electrons and lacks internal differences in terms of electronegativity. It has been demonstrated by Calvaresi et al. that Van der Waals interaction do indeed govern the binding between fullerenes and protein [40].

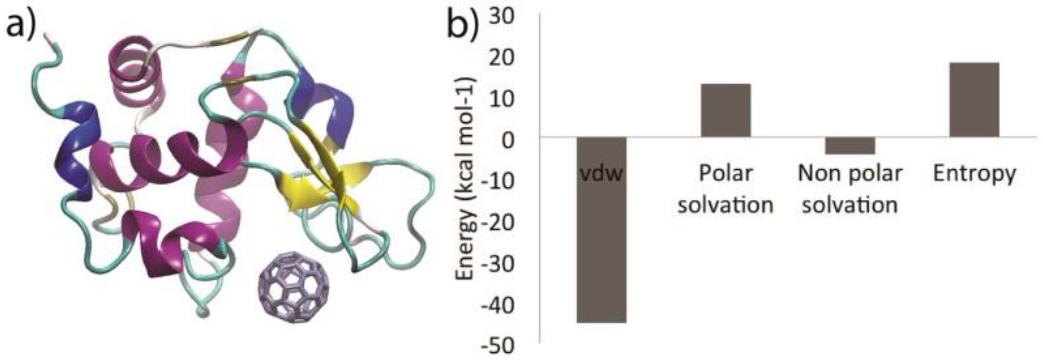


Figure 13 - (a) Binding pocket of C60 in lysozyme. (b) Energy components of $\Delta G_{binding}$[40].

The results were obtained by performing 200 ns of Molecular Dynamics simulation in explicit solvent of the adduct lysozyme-C_{60} that was predicted with the basic reverse screening protocol[39] and later confirmed by experimental evidence[34]. The binding energy contributions were obtained by using the MM-GBSA method[41] to analyze the evolution of the system coordinates during the simulation. Van der Waals interactions are the main contributors to the binding (-45,1 kcal mol^{-1}) and are

marginally assisted by non-polar solvation contributions (-4,3 kcal mol^{-1}), that arise from losing unfavorable interactions with water molecules, as it is in case of C_{60} and the hydrophobic residues in the binding pocket. On the other hand, the binding of C_{60} prevents polar residues in the binding pocket to interact favorably with the water environment and limits the mobility of all the residues in contact with it; these behaviors translate in detrimental contributions to the binding in the form of polar solvation (12,8 kcal mol^{-1}) and entropy (18,1 kcal mol^{-1}). However, detrimental contributions are overshadowed by favorable vdW interactions and C_{60} binds firmly. Van der Waals scales directly with the area of contact between the interacting partner, as it can be seen by simply comparing the boiling points of linear alkanes with the boiling points of branched alkanes with the same number of atoms, The obvious consequence is that the higher is the degree of shape complementarity between two molecules, the higher the strength of the resulting van der Waals interactions. Since PatchDock generates poses on the basis of shape complementarity alone and strongly considers the latter in the scoring stage, it was able to correctly identify the binding pocket and overcome the theoretical limitations that were presented at the beginning of this paragraph.

4.7.2. Reverse screening, episode gadofullerene: the revenge of MM-GBSA

Can we draw the same conclusions for gadofullerenes? Although gadofullerenes are rigid molecules with a strong hydrophobic character they are characterized by the phenomenon of intrafullerene electron transfer, that grants the carbon cage a net charge opposite of that of the metal ion trapped inside [9]. Since this electrostatic contribution might play a key role in the interactions with the binding site, we decided to do a comparative binding investigation between lysozyme-C_{60} and its gadofullerene counterpart, lysozyme-Gd@C_{60} [42].

Although the 3D structure of Gd@C_{60} has been obtained through chemical functionalization of the carbon cage[43] , the functional groups can interfere with the process of intrafullerene charge transfer and have an impact on the position of inner metal atom and the resulting partial charges on the surface. Therefore, we decided to obtain the 3D structure of Gd@C_{60} by positioning the Gd atom in various points inside C_{60} and perform quantum mechanical optimization with a range of values of

ground state multiplicity to identify the most stable geometry-multiplicity pair. The molecule was modelled ad the Density Functional Theory (DFT) level. Relativistic Effective Core Potentials (RECP) have been widely used in combination with DFT to represent endohedral metallofullerenes. In particular, we used the pure GGA functional PBE[44] in combination with the effective core potential triple split basis set (CEP-121G)[45] to describe the Gd atom and the 6-31G* basis set [46] to describe the carbon atoms. This combination produced results in good agreement with experimental data, in particular septet–nonet gap and Gd–C distance, in an extensive benchmark from Dai et al., who investigated various combinations of DFT types and RECP[47]. To ensure that the obtained structures are minima of the potential energy surface of $Gd@C_{60}$ frequency calculations were carried out. All calculations were performed with the QM suite *Gaussian09*[48]. In line with previous studies, the ground state is a septet (S=3), with the Gd atom positioned close to one of the hexagonal faces of the carbon cage [49].

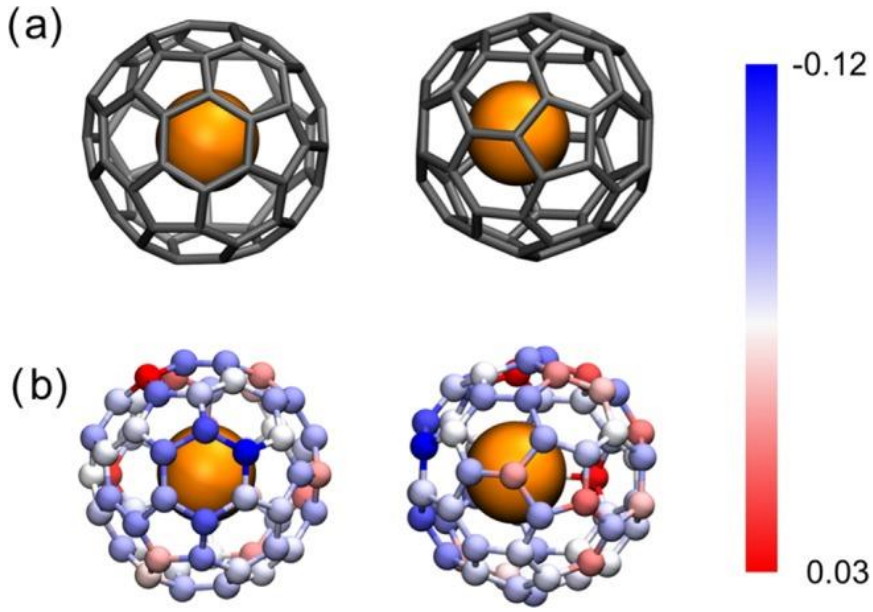


Figure 14 - (a) QM optimized structure of $Gd@C_{60}$; (b) partial charges of the carbon cage (negative charges in blue, positive charges in red). Since che Gd atom is located close to one hexagonal face of C_{60}, intrafullerene electron transfer generates polarized the carbon cage: atoms in the hemisphere close to the metal ion acquire negative charges while atoms in the opposite hemisphere acquire positive charges.

For this structure, ESP charges were computed according to the Merz-Singh-Kollman (MK) scheme[50] which were used in combination with AMBER FF and the TIP3P water model, both of which were used in the reference study of this comparative[40], in a previous study on the water exchange dynamics of Gd-DOTA complexes[51]. Using the RESP charge model, which is the designated charge model of AMBER FF, was not possible since current algorithms cannot handle fullerenes structures. Ring identification is an historical problem of chemoinformatics[52] and since fullerenes are comprised of many fused rings they pose an extreme challenge to algorithms such as the one employed in RESP calculations. However, given the high rigidity and symmetry of fullerenes and their derivatives, charge conformational dependence is not an issue; therefore, ESP are perfectly adequate in representing the interactions with water molecules and protein structures.

To account for electrostatic contributions to the binding, we used the advanced protocol that was presented in the previous chapter, to identify the binding pocket of $Gd@C_{60}$ on lysozyme. Briefly, docking candidates were generated using PatchDock, their binding pocket was optimized using MM energy minimization algorithms and scoring was performed with MM-GBSA (details in Table 1).

Binding pocket minimization	
Target of optimization	All protein residues within 5 Å from ligand
Steps of steepest descent	100
Steps of conjugate gradient	150
Solvent	Vacuum
Electrostatic treatment	Cut-off (no PME)
Non-bonded interaction cut-off	12 Å
Molecular Mechanics – Poisson-Boltzmann (Generalized Born) Surface Area	
Solvation model	igb5[53]

Table 1 - Computational details for the MM refinement and MM-GBSA scoring parts of the advanced docking protocol.

The only difference with the protocol employed for porphine and phthalocyanine, was the use of the package AMBER 12[54] and its tools, in particular the Python script MMPBSA.py[55] for the scoring phase, and that we modelled the protein using ff12SB[54] to better mimic the simulation conditions of the previous study[40]. The carbon cage atoms were still modelled using the Generalized Amber ForceField

(GAFF)[56] while the Lennard-Jones potential parameters for the non-bonded interactions of the Gd atom were taken from the Gd-DOTA study[51].

The 3D structure of the best docking candidate was used as the initial coordinates for 200 ns of MD simulation with TIP3P explicit solvent [57], with the same simulation parameters and protocol of the reference study[40], which are summarized in Table 2.

Energy minimization	
Steps of steepest descent	1000
Non-bonded iteractions cut-off	12 Å
Particle Mesh Ewald	off
NVT equilibration	
Timestep	2 fs
Duration	50 ps
Bond and angle restraints	SHAKE, only h-bonds
Thermostat	Berendsen
Target temperature	298 K
Particle Mesh Ewald	on
Non-bonded interactions cut-off (= cut off for the direct space sum of PME)	10 Å
NPT equilibration	
Timestep	2 fs
Duration	50 ps
Bond and angle restraints	SHAKE, only h-bonds
Thermostat	Berendsen
Target temperature	298 K
Barostat	Berendsen, isotropic
Target pressure	1 atm
Particle Mesh Ewald	on
Non-bonded interaction cut-off (= …)	10 Å
Equilibration in production MD conditions	
Timestep	2 fs
Duration	400 ps
Bond and angle restraints	SHAKE, only h-bonds
Thermostat	Andersen
Target temperature	298 K
Barostat	Berendsen, isotropic
Target pressure	1 atm
Particle Mesh Ewald	on
Non-bonded interaction cut-off (= …)	10 Å
Production MD	
Parameters are identical to previous step except for:	
Duration	200 ns
Coordinates are saved every	2 ps

Table 2 - Molecular Dynamics simulation protocol

The Molecular Dynamics trajectory was analyzed using the same MM-GBSA procedure of the reference paper (igb=5, like in the scoring part of the docking protocol), complete with normal mode analysis to calculate the entropic contribution to the binding. Binding calculation have been performed with MMPBSA.py[55].

The binding energy for the adduct lysozyme-Gd@C_{60} was evaluated to be -18,7 kcal mol^{-1}, a value not significantly different from the value of the binding energy of lysozyme-C_{60}, -18,5 kcal mol^{-1}; since the latter adduct has been confirmed by experimental evidence, this result confirms that gadofullerenes can interact with proteins to form stable complexes and the latter can be exploited as carrier systems.

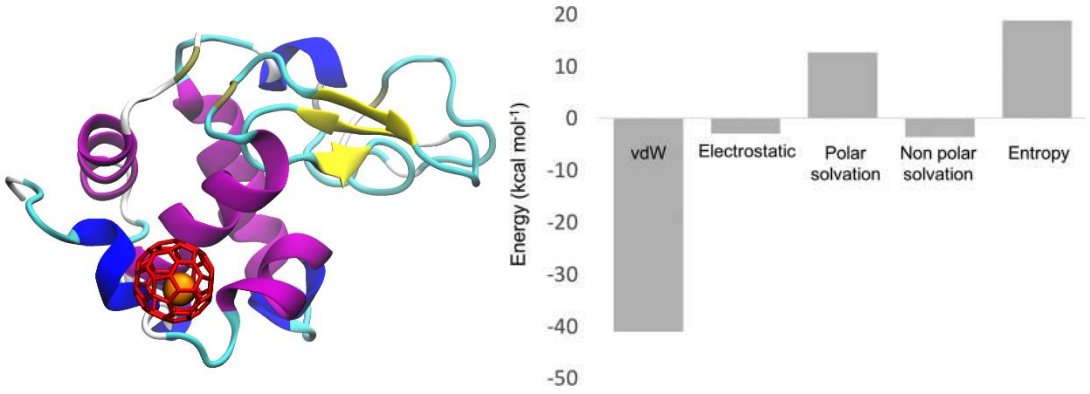
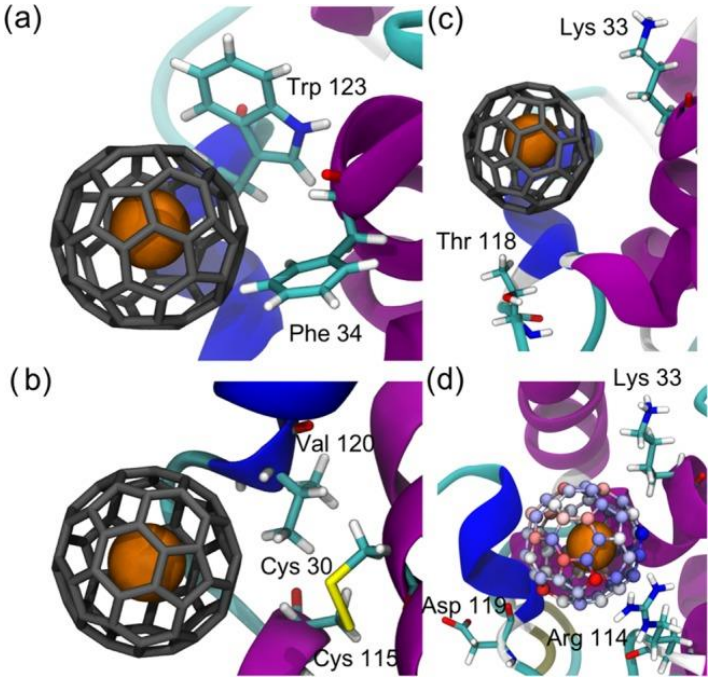


*Figure 15 - **Left**: Gd@C_{60} in the favored binding site on lysozyme; **Right**: Energetic contribution to the binding[42].*

By analyzing the binding energy components, we can see that electrostatic interactions contribute to the binding (-3.0 kcal mol^{-1}) along with non-polar solvation (-3,7 kcal mol^{-1}), but both are overshadowed by van der Waals interactions, which are the driving force to the binding once again (-41,1 kcal mol^{-1}) even if the surface of Gd@C_{60} is highly charged. Polar solvation (12,6 kcal mol^{-1}) and entropy (16,5 kcal mol^{-1}) are once again detrimental to the binding.

Protein residues can interact in various ways with carbon nanomaterials, such as π-π stacking, hydrophobic interactions, surfactant-like interactions and electrostatic interactions, which are all well represented in the case of this protein-gadofullerene adduct (figure 16). While the first three types of interactions are shared with pristine fullerenes, electrostatics are a prerogative to endohedral metallofullerenes, therefore will be analyzed more in depth.



Residue	*ΔG binding (kcal mol^{-1})*
Trp 123	-6,9
Arg 114	-6,6
Val 120	-4,0
Thr 118	-3,1
Phe 34	-2,9
Lys 33	-2,7
Cys 30	-2,7
Ala 122	-2,6
Cis 115	-2,4
Asp 119	-2,4

Figure 16 -
***Left**: types of interactions between lysozyme residues and Gd@C$_{60}$:*
(a) π-π stacking (Trp 123: sandwich like, Phe 34: T-shaped), (b) hydrophobic,
(c) surfactant-like, (d) electrostatic[42];
***Right**: Residues that contribute with more than 2 Kcal mol $^{-1}$ to the binging.*

As already stated, the phenomenon of intrafullerene electron transfer creates a negatively charge hemisphere close to the Gd atom and a positively charge on the opposite side of the carbon cage. An inverse charge distribution characterizes the binding pocket, with the positively charged Arg 114 and Lys 33 located on the opposite side of the binding site with respect to Asp 199. During the MD simulation Gd@C$_{60}$ orients itself to minimize the electrostatic repulsions and interacts strongly with the charged residues of the binding pocket; all three are in the top 10 most interacting residues, with Arg 114 almost taking the crown.

To better understand the impact on the binding energy of the complementarity in charge distribution between a gadofullerene and its binding pocket, we placed the Gd@C$_{60}$ in the binding pocket of C$_{60}$, that is the active site of lysozyme, and repeated the MD simulation and MM-GBSA analysis of the trajectory using the same parameters presented in Table 2. The resulting binding energy is -4,8 kcal mol^{-1}, less than 1/3 with respect to the predicted binding site. Because Gd@C$_{60}$ has the same volume and shape of its pristine counterpart, PatchDock would have placed it in the

same binding site of C_{60}, since it does not consider electrostatic contributions in both the pose generation and scoring phases.

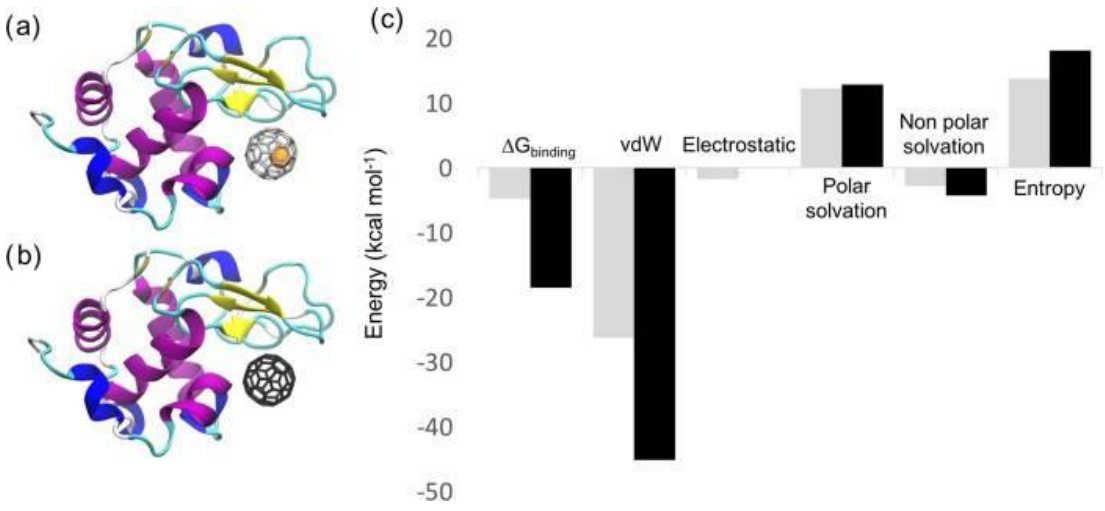


Figure 17 - (a) Binding of Gd@C_{60} in the C_{60} binding pocket. (b) Binding of C_{60} in the C_{60} binding pocket Gd@C_{60}. (c) Total binding energy ($\Delta G_{binding}$) and energy components of $\Delta G_{binding}$ of Gd@C_{60} (in gray) and C_{60} (in black) with lysozyme, in the C_{60} binding pocket[42].

The most interesting values are the worse Van der Waals interactions and the more favorable entropic contribution, which are connected. If we analyze the mobility during the simulation of C_{60} with Gd@C_{60} we can notice that while the pristine fullerene is hold firmly in place in the binding pocket, the gadofullerenes moves along the crevice-like active site of lysozyme and outside of it. This enhanced mobility prevents the residues in the binding pocket from gluing themselves to the carbon cage and maximize van der Waals interactions and at the same time allows them to move more freely, resulting in a less detrimental entropic contribution. On the other hand, Gd@C_{60}'s mobility is severely reduced when it is placed in the binding site that was predicted by the advanced reverse screening protocol and strong stabilizing interactions can take place.

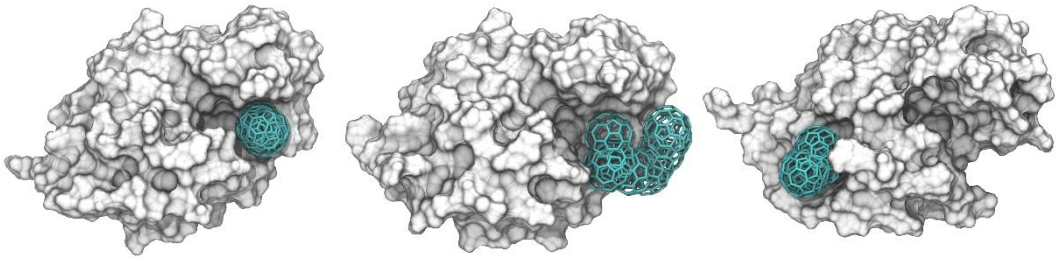


Figure 18 – From left to right: carbon cage position during 200 ns of MD for C60 in its binding site, @C60 in C60 binding site, Gd@C60 in its predicted binding site [42].

The origin of this mobility lies in the electrostatic interactions. Although their net contribution to the binding is low, it is actually the combination of many small stabilizing and destabilizing contributions that partially cancel each other. In a binding site characterized by a charge distribution complementary to the gadofullerenes surface, they contribute to its stabilization; in a binding site with a disordered distribution of charged residues, they cause a continuous series of kicks toward the gadofullerenes which prevents residues from adhering to the carbon cage and form stable van der Waals interactions.

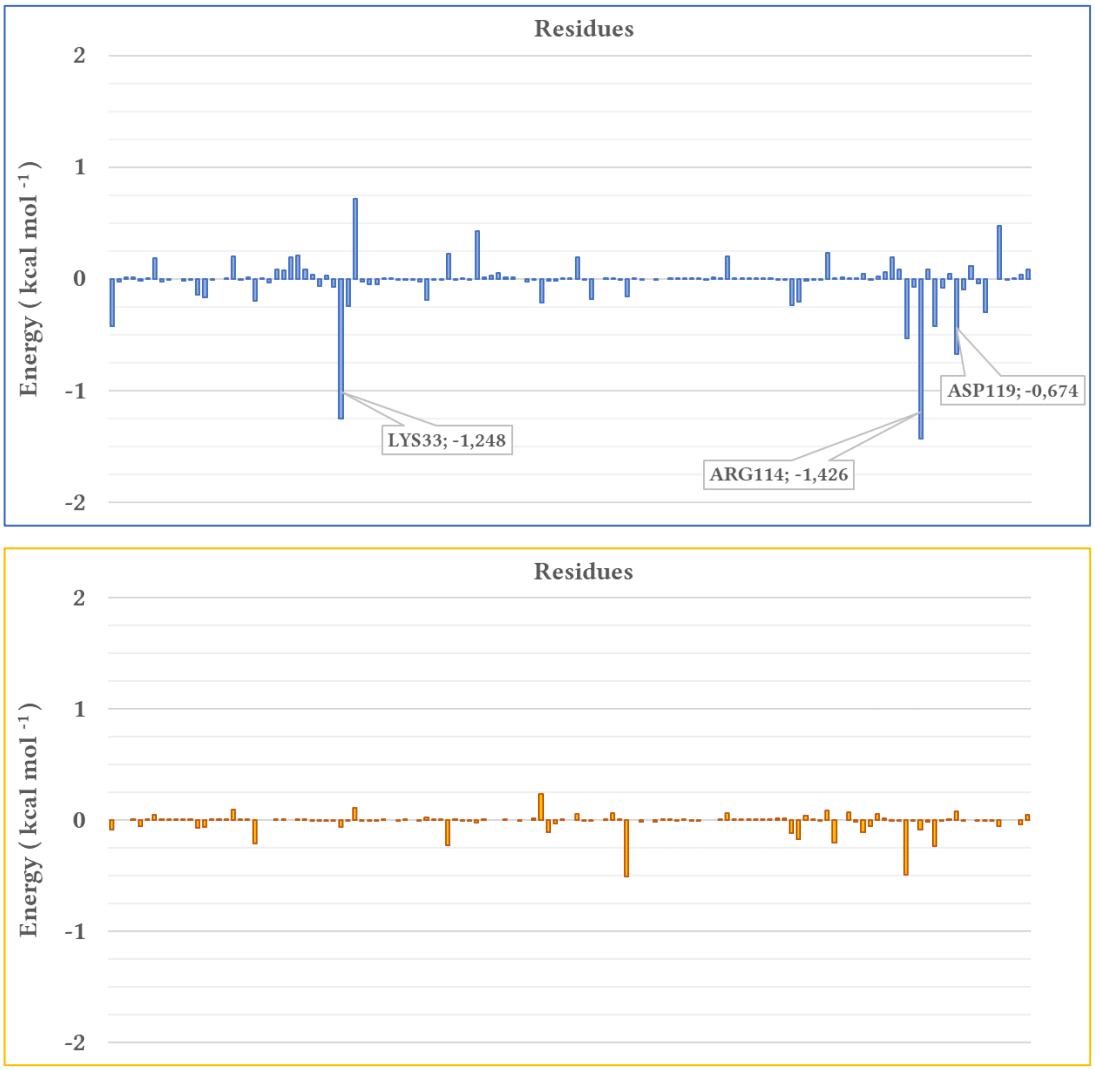


Figure 19 – Electrostatic interactions per residue for:
***Top**: Gd@C_{60} in the predicted binding site;*
***Bottom**: Gd@C_{60} in C_{60} bindind site.*
In both cases, every single contribution is lower than 1,5 kcal mol^{-1} but depending on how residues are localized in the binding site they determine a successful binding.

In conclusion, we found that electrostatic contributions play an important, albeit indirect, role in the binding of gadofullerenes on proteins since they affect the formation of stable van der Waals interaction, which are the driving force for the binding. As result, only binding sites with *Janus*-like charge distributions can interact favorably. In the case of gadofullerenes only the advanced protocol based on the combination of PatchDock pose generation and MM-GBSA scoring can properly take into consideration electrostatic charges and accurately predict the favored binding site.

4.8. Reverse screening investigation of Gd@C$_{82}$

As already stated, only the advanced protocol that combines PatchDock and MM-GBSA can accurately predict the binding site of gadofullerenes. Therefore, it has been applied for screening the PDTD[38] against Gd@C$_{82}$, the most promising gadofullerene for therapeutic and diagnostic applications.

Since Gd@C$_{82}$ has been successfully isolated and its structure is well known, it was not necessary to perform the geometry-multiplicity screening that was done for Gd@C$_{60}$. Gd@C$_{82}$ possess C_{2v} symmetry, with the Gd atom positioned along the C_2 axis near the intersecting hexagonal face and with a Gd-C distance equal to 2,49 Å [47,58,59]; its ground state is a septet (S=3)[60].

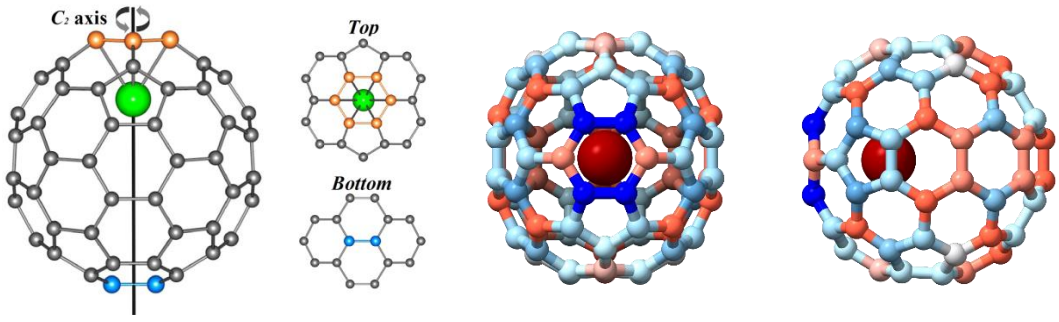


Figure 20 - From left to right: C_{2v} symmetry of Gd@C$_{82}$ with Gd atom on the C_2 axis [47]; ESP charge distribution looking along the C_2 axis; ESP charge distribution looking perpendicular to the C_2 axis

Nevertheless, to apply the protocol QM calculations must be performed to compute the ESP charges that will be used in the pose refinement and scoring phases. Once again, we used the pure GGA functional PBE[44] in combination with the effective core potential triple split basis set (CEP-121G)[45] to describe the Gd atom and the 6-

31G* basis set[46] to describe the carbon atoms, in accordance to the previous thermodynamic comparison and the benchmarks from Dai et al.[47]; population analysis was performed using the Merz-Singh-Kollman scheme[50]. To describe the docking candidates at the Molecular Mechanics level, we used the more recent ff14SB AMBER forcefield[61] for the protein residues, in line with the reverse screening inverstigation of porphine and phthalocyanine, the Generalized Amber ForceField (GAFF)[56] for the carbon cage and the previous Lennard-Jones potential parameters[51] for the Gd atom. The simulation parameters used in the docking candidates' refinement and scoring phases are reported in Table 1. Parameter assignment, topology and coordinate files generation and MM-GBSA analysis have been performed using the tools included in the AmberTools16 installation[62] as it was described in the previous chapter.

4.9. Analysis of the scoreboards: solving old problems and finding new therapeutic targets

The scoreboard of the interactions between Gd@C82 and the PDTD database entries that reached the second stage of the protocol will be analyzed to identify possible carrier proteins or targets for the therapeutical and diagnostic techniques presented at the beginning of the chapter. To filter out false positives caused by the intrinsic approximations of computational techniques, only the first 100 positions (which roughly represent the top 10% of the entire PDTD) have been considered as reliable predictions and have been reported in Table 3 of the Appendix A. In the following section, an analysis of the literature on the most interesting proteins from a pharmacological point of view is presented.

4.9.1. Imaging and theranostic targets

Rank 1 is occupied by the voltage-gated potassium channel (PDB id: 1JVM), which ranked first also in the reverse screening of PDTD against C_{60}; the binding site of both molecules is buried inside the pore. Since C_{60} was confirmed to act as an ion channel blocker[63] and shares its binding site with Gd@C_{82}, it is safe to assume that the latter can hamper ion mobility as well. In addition, the poor rank of the Chloride

Intracellular Channel 4 (ClIC4) (PDB id: 2AHE, rank 638) suggests that Gd@C82 might discriminate between K^+ and Cl^- channels and bind selectively in the same way that was confirmed for C_{60}[63]. and be employed as a selective ion channel blocker. Finally, since gadofullerenes which can enhance proton relaxivity up to 20 times more than commercial contrast agents[20], $Gd@C_{82}$ could be used to achieve cellular level MRI, by releasing in a controlled way in close proximity to potassium ion channels. Cellular MRI is getting more and more attention from the research community [64,65] as it can allow to detect, track, and quantify cells in vivo and over time, a feat that can have profound clinical implications.

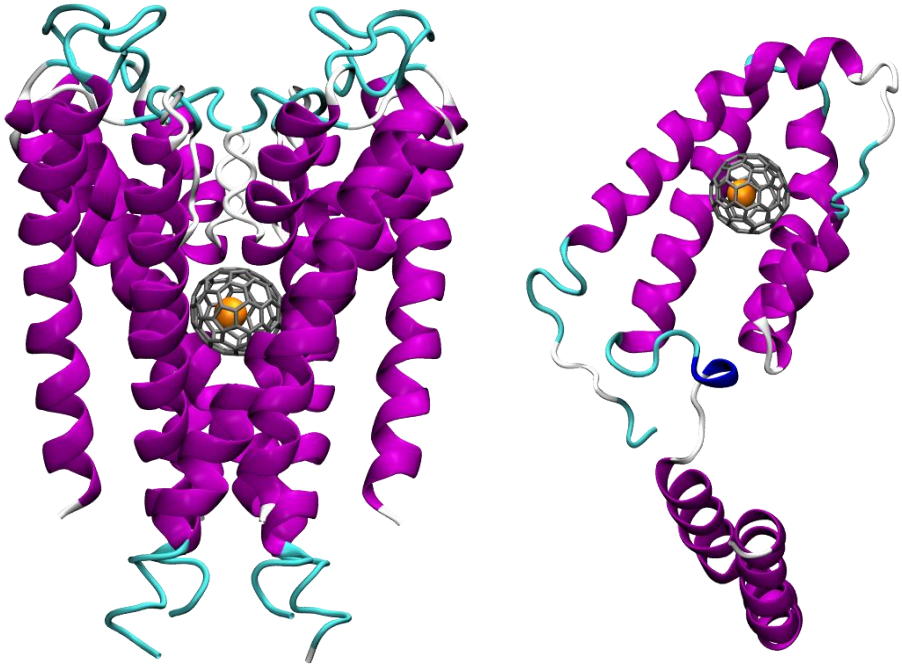


*Figure 21 - **Left**: voltage gated K^+ channel (PDB id: 1JVM); **Right**: human interleukin-10 (PDB id: 2ILK)*

Rank 24 is occupied by the human interleukin-10 (IL-10) (PDB id: 2ILK). IL-10 is predominantly an anti-inflammatory cytokine produced by the majority of immune cells that limits immune responses during infection, allergy, and autoimmunity, preventing damage to the host. However, chronic infections can arise if IL-10 exerts too much immune suppression[66]. By releasing $Gd@C_{82}$ in a site of ongoing inflammation, it would be possible to create images of its extension with high sensitivity and contrast; on the other end, by using its photodynamic properties it would be possible to neutralize IL-10 proteins, preventing the onset of chronic infections.

4.9.2. Targets for photodynamic and photothermal/photoacoustic therapy

Rank 16 is taken by the inosine-5'-monophosphate dehydrogenase (IMPDH) (PDB id: 1ME8), which catalyzes the de novo biosynthesis of guanine nucleotides. B and T lymphocytes require sufficient levels of guanosine to initiate a proliferative response against a mitogen or antigen[67]; as result, inhibition of IMPDH leads to immunosuppression by decreasing guanine nucleotides that are required for the proliferation of lymphocytes[68]. In addition, IMPDH activity is enhanced in rapidly proliferating human leukemic cell lines, solid tumor tissues, and other replicating cell types, making IMPDH a target for cancer as well as immunosuppressive chemotherapy[67].

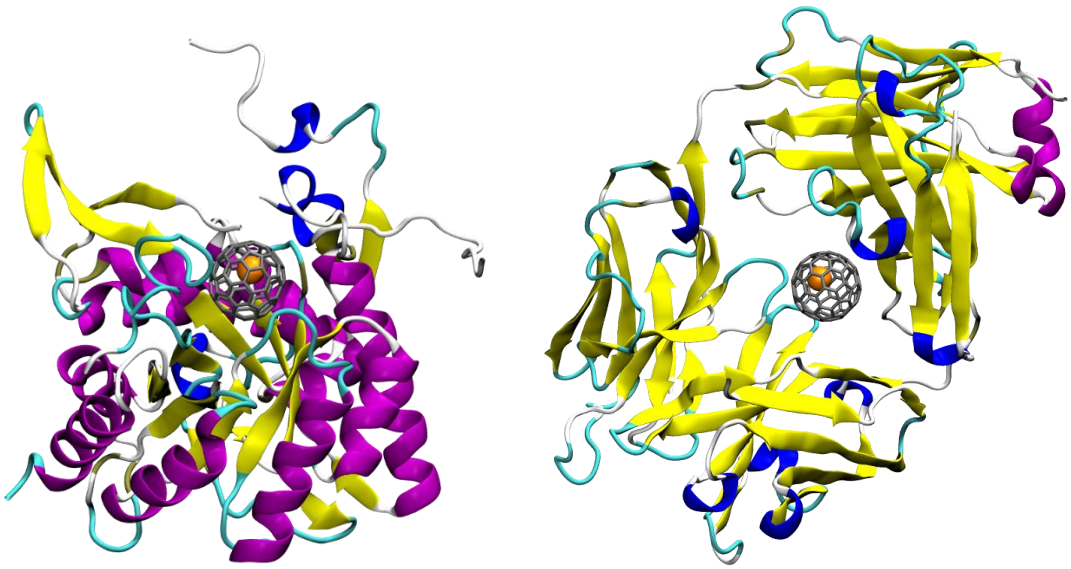


*Figure 22 - **Left**: inosine-5'-monophosphate dehydrogenase (PDB id: 1ME8); **Right**: monoclonal anti-tumor antibody Br96 (PDB id: 1CLY)*

4.9.3. Innovative solutions in cancer immunoassays

Rank 30 is occupied by the monoclonal anti-tumor antibody Br96 (PDB id: 1CLY). Br96 binds selectively and with high affinity Le^y (Lewis Y), a carbohydrate determinant that is expressed at high levels on many human carcinomas[69,70]. Combining Br96 high selectivity and Gd@C82 superior enhancement of MRI signals can lead to high-sensitivity labelled antibodies for immunoassay that would be able to detect traces amounts of cancer in blood plasma. This would allow the early identification of tumor progression in patients, with profound clinical consequences.

4.9.4. Protein carriers for theranostic platforms

Rank 2 is occupied by the human pregnane X receptor (PDB id: 1ILH). PXR is a nuclear receptor that is expressed in all mammalian species' liver and intestine, front line organs involved in the absorption, distribution, metabolism, and elimination of xenobiotics and endobiotics[71]. It has a large, spherical ligand binding cavity that allows it to interact with a wide range of hydrophobic chemicals. PXR serves as a generalized sensor of hydrophobic toxins, in contrast to the majority of nuclear receptors that selectively binds their physiological ligands Thus, unlike other nuclear receptors that interact selectively with their physiological ligands[72]. It triggers the cellular response to xenobiotics, including induction of enzymes involved in drug oxidation and conjugation, as well as induction of xenobiotic and endobiotic transporters[71]. The characteristics of the binding site along with the high rank suggest that it could be a stable carrier platform for Gd@C_{82}. In addition, a growing body of evidence suggest that PXR activation accelerates cancer cell growth and drug resistance[73], making it a potential target for cancer treatment. The strong interaction with Gd@C82 is particularly important since only a few PXR inhibitors have been identified[74].

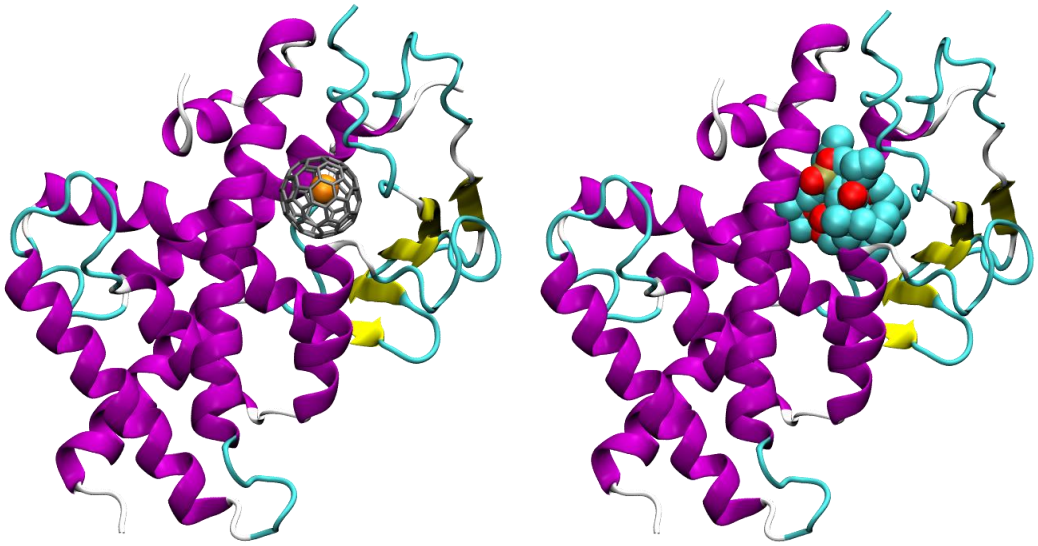


Figure 23 - Human pregnane X receptor (PDB id: 1ILH).
***Left**: adduct with Gd@C82; **Right**: adduct with hydrophobic ligand*

4.10. References

[1] H. W. Kroto, J. R. Heath, S. C. O'Brien, R. F. Curl, R. E. Smalley, *Nature* **1985**, *318*, 162–163.

[2] H. W. Kroto, *Nature* **1987**, *329*, 529–531.

[3] A. Hirsch, Ed. , *Fullerenes and Related Structures*, Springer-Verlag, Berlin Heidelberg, **1999**.

[4] R. C. Haddon, *Science* **1993**, *261*, 1545–1550.

[5] A. W. Jensen, S. R. Wilson, D. I. Schuster, *Bioorganic & Medicinal Chemistry* **1996**, *4*, 767–779.

[6] P. M. Allemand, G. Srdanov, A. Koch, K. Khemani, F. Wudl, Y. Rubin, F. Diederich, M. M. Alvarez, S. J. Anz, R. L. Whetten, *J. Am. Chem. Soc.* **1991**, *113*, 2780–2781.

[7] J. W. Arbogast, A. P. Darmanyan, C. S. Foote, F. N. Diederich, R. L. Whetten, Y. Rubin, M. M. Alvarez, S. J. Anz, *J. Phys. Chem.* **1991**, *95*, 11–12.

[8] H. Shinohara, *Reports on Progress in Physics* **2000**, *63*, 843–892.

[9] R. D. Johnson, M. S. de Vries, J. Salem, D. S. Bethune, C. S. Yannoni, *Nature* **1992**, *355*, 239–240.

[10] Z. Chen, L. Ma, Y. Liu, C. Chen, *Theranostics* **2012**, *2*, 238–250.

[11] R. Bakry, R. M. Vallant, M. Najam-ul-Haq, M. Rainer, Z. Szabo, C. W. Huck, G. K. Bonn, *Int J Nanomedicine* **2007**, *2*, 639–649.

[12] Y. Yamakoshi, N. Umezawa, A. Ryu, K. Arakane, N. Miyata, Y. Goda, T. Masumizu, T. Nagano, *J. Am. Chem. Soc.* **2003**, *125*, 12803–12809.

[13] X.-J. Liang, H. Meng, Y. Wang, H. He, J. Meng, J. Lu, P. C. Wang, Y. Zhao, X. Gao, B. Sun, C. Chen, G. Xing, D. Shen, M. M. Gottesman, Y. Wu, J. Yin, L. Jia, *PNAS* **2010**, *107*, 7449–7454.

[14] Y. Liu, F. Jiao, Y. Qiu, W. Li, F. Lao, G. Zhou, B. Sun, G. Xing, J. Dong, Y. Zhao, Z. Chai, C. Chen, *Biomaterials* **2009**, *30*, 3934–3945.

[15] H. Meng, G. Xing, B. Sun, F. Zhao, H. Lei, W. Li, Y. Song, Z. Chen, H. Yuan, X. Wang, J. Long, C. Chen, X. Liang, N. Zhang, Z. Chai, Y. Zhao, *ACS Nano* **2010**, *4*, 2773–2783.

[16] F. Jiao, Y. Liu, Y. Qu, W. Li, G. Zhou, C. Ge, Y. Li, B. Sun, C. Chen, *Carbon* **2010**, *48*, 2231–2243.

[17] Q. Zhang, W. Yang, N. Man, F. Zheng, Y. Shen, K. Sun, Y. Li, L.-P. Wen, *Autophagy* **2009**, *5*, 1107–1117.

[18] V. Krishna, A. Singh, P. Sharma, N. Iwakuma, Q. Wang, Q. Zhang, J. Knapik, H. Jiang, S. R. Grobmyer, B. Koopman, B. Moudgil, *Small* **2010**, *6*, 2236–2241.

[19] P. M. Ajayan, M. Terrones, A. de la Guardia, V. Huc, N. Grobert, B. Q. Wei, H. Lezec, G. Ramanath, T. W. Ebbesen, *Science* **2002**, *296*, 705–705.

[20] H. Kato, Y. Kanazawa, M. Okumura, A. Taninaka, T. Yokawa, H. Shinohara, *J. Am. Chem. Soc.* **2003**, *125*, 4391–4397.

[21] A. B. E. Attia, G. Balasundaram, M. Moothanchery, U. S. Dinish, R. Bi, V. Ntziachristos, M. Olivo, *Photoacoustics* **2019**, *16*, 100144.

[22] J.-J. Yin, F. Lao, P. P. Fu, W. G. Wamer, Y. Zhao, P. C. Wang, Y. Qiu, B. Sun, G. Xing, J. Dong, X.-J. Liang, C. Chen, *Biomaterials* **2009**, *30*, 611–621.

[23] S. Samal, K. E. Geckeler, *Chem. Commun.* **2001**, 2224–2225.

[24] N. Gharbi, M. Pressac, M. Hadchouel, H. Szwarc, S. R. Wilson, F. Moussa, *Nano Lett.* **2005**, *5*, 2578–2585.

[25] C. M. Sayes, J. D. Fortner, W. Guo, D. Lyon, A. M. Boyd, K. D. Ausman, Y. J. Tao, B. Sitharaman, L. J. Wilson, J. B. Hughes, J. L. West, V. L. Colvin, *Nano Lett.* **2004**, *4*, 1881–1887.

[26] C. M. Sayes, A. A. Marchione, K. L. Reed, D. B. Warheit, *Nano Lett.* **2007**, *7*, 2399–2406.

[27] H. J. Johnston, G. R. Hutchison, F. M. Christensen, K. Aschberger, V. Stone, *Toxicological Sciences* **2010**, *114*, 162–182.

[28] A. Dhawan, J. S. Taurozzi, A. K. Pandey, W. Shan, S. M. Miller, S. A. Hashsham, V. V. Tarabara, *Environ. Sci. Technol.* **2006**, *40*, 7394–7401.

[29] D. Y. Lyon, L. K. Adams, J. C. Falkner, P. J. J. Alvarez, *Environ. Sci. Technol.* **2006**, *40*, 4360–4366.

[30] S. Yamago, H. Tokuyama, E. Nakamura, K. Kikuchi, S. Kananishi, K. Sueki, H. Nakahara, S. Enomoto, F. Ambe, *Cell chemical biology* **1995**, *2*, 385–389.

[31] Z. Q. Ji, H. Sun, H. Wang, Q. Xie, Y. Liu, Z. Wang, *J Nanopart Res* **2006**, *8*, 53–63.

[32] R. F. Schinazi, R. Sijbesma, G. Srdanov, C. L. Hill, F. Wudl, *Antimicrobial Agents and Chemotherapy* **1993**, *37*, 1707–1710.

[33] S. H. Friedman, D. L. DeCamp, R. P. Sijbesma, G. Srdanov, F. Wudl, G. L. Kenyon, *J. Am. Chem. Soc.* **1993**, *115*, 6506–6509.

[34] M. Calvaresi, F. Arnesano, S. Bonacchi, A. Bottoni, V. Calò, S. Conte, G. Falini, S. Fermani, M. Losacco, M. Montalti, G. Natile, L. Prodi, F. Sparla, F. Zerbetto, *ACS Nano* **2014**, *8*, 1871–1877.

[35] F. A. de Wolf, G. M. Brett, *Pharmacol Rev* **2000**, *52*, 207–236.

[36] E. C. Meng, B. K. Shoichet, I. D. Kuntz, *Journal of Computational Chemistry* **1992**, *13*, 505–524.

[37] D. Duhovny, R. Nussinov, H. J. Wolfson, in *Algorithms in Bioinformatics* (Eds.: R. Guigó, D. Gusfield), Springer, Berlin, Heidelberg, **2002**, pp. 185–200.

[38] Z. Gao, H. Li, H. Zhang, X. Liu, L. Kang, X. Luo, W. Zhu, K. Chen, X. Wang, H. Jiang, *BMC Bioinformatics* **2008**, *9*, 104.

[39] M. Calvaresi, F. Zerbetto, *ACS Nano* **2010**, *4*, 2283–2299.

[40] M. Calvaresi, A. Bottoni, F. Zerbetto, *J. Phys. Chem. C* **2015**, *119*, 28077–28082.

[41] J. Srinivasan, T. E. Cheatham, P. Cieplak, P. A. Kollman, D. A. Case, *Journal of the American Chemical Society* **1998**, *120*, 9401–9409.

[42] F. Bologna, E. J. Mattioli, A. Bottoni, F. Zerbetto, M. Calvaresi, *ACS Omega* **2018**, *3*, 13782–13789.

[43] A. Nakagawa, M. Nishino, H. Niwa, K. Ishino, Z. Wang, H. Omachi, K. Furukawa, T. Yamaguchi, T. Kato, S. Bandow, J. Rio, C. Ewels, S. Aoyagi, H. Shinohara, *Nature Communications* **2018**, *9*, 3073.

[44] J. P. Perdew, K. Burke, M. Ernzerhof, *Phys. Rev. Lett.* **1996**, *77*, 3865–3868.

[45] T. R. Cundari, W. J. Stevens, *J. Chem. Phys.* **1993**, *98*, 5555–5565.

[46] G. A. Petersson, A. Bennett, T. G. Tensfeldt, M. A. Al-Laham, W. A. Shirley, J. Mantzaris, *J. Chem. Phys.* **1988**, *89*, 2193–2218.

[47] X. Dai, Y. Gao, M. Xin, Z. Wang, R. Zhou, *J. Chem. Phys.* **2014**, *141*, 244306.

[48] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery Jr., J. E. Peralta, F. Ogliaro, M. Bearpark, J. J. Heyd, E. Brothers, K. N. Kudin, V. N. Staroverov, T. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, D. J. Fox, *Gaussian 09*, Gaussian Inc., Wallingford CT, **2009**.

[49] J. Lu, W. N. Mei, Y. Gao, X. Zeng, M. Jing, G. Li, R. Sabirianov, Z. Gao, L. You, J. Xu, D. Yu, H. Ye, *Chemical Physics Letters* **2006**, *425*, 82–84.

[50] U. C. Singh, P. A. Kollman, *Journal of Computational Chemistry* **1984**, *5*, 129–145.

[51] R. J. Dimelow, N. A. Burton, I. H. Hillier, *Phys. Chem. Chem. Phys.* **2007**, *9*, 1318–1323.

[52] F. Berger, C. Flamm, P. M. Gleiss, J. Leydold, P. F. Stadler, *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 323–331.

[53] A. Onufriev, D. Bashford, D. A. Case, *Proteins* **2004**, *55*, 383–394.

[54] D. A. Case, T. A. Darden, T. E. Cheatham III, C. L. Simmerling, J. Wang, R. E. Duke, R. Luo, R. C. Walker, W. Zhang, K. M. Merz, B. Roberts, S. Hayik, A. Roitberg, G. Seabra, J. Swails, A. W. Götz, I. Kolossváry, K. F. Wong, F. Paesani, J. Vanicek, R. M. Wolf, J. Liu, X. Wu, S. R. Brozell, T. Steinbrecher, H. Gohlke, Q. Cai, X. Ye, J. Wang, J. M.-Hsieh, G. Cui, D. R. Roe, D. H. Mathews, M. G. Seetin, R. Salomon-Ferrer, C. Sagui, V. Babin, T. Luchko, S. Gusarov, A. Kovalenko, P. A. Kollman, *AMBER 12*, University Of California, San Francisco, **2012**.

[55] B. R. Miller, T. D. McGee, J. M. Swails, N. Homeyer, H. Gohlke, A. E. Roitberg, *J. Chem. Theory Comput.* **2012**, *8*, 3314–3321.

[56] J. Wang, R. M. Wolf, J. W. Caldwell, P. A. Kollman, D. A. Case, *Journal of Computational Chemistry* **2004**, *25*, 1157–1174.

[57] W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey, M. L. Klein, *J. Chem. Phys.* **1983**, *79*, 926–935.

[58] M. Suzuki, X. Lu, S. Sato, H. Nikawa, N. Mizorogi, Z. Slanina, T. Tsuchiya, S. Nagase, T. Akasaka, *Inorg. Chem.* **2012**, *51*, 5270–5273.

[59] T. Akasaka, T. Kono, Y. Takematsu, H. Nikawa, T. Nakahodo, T. Wakahara, M. O. Ishitsuka, T. Tsuchiya, Y. Maeda, M. T. H. Liu, K. Yoza, T. Kato, K. Yamamoto, N. Mizorogi, Z. Slanina, S. Nagase, *J. Am. Chem. Soc.* **2008**, *130*, 12840–12841.

[60] K. Furukawa, S. Okubo, H. Kato, H. Shinohara, T. Kato, *J. Phys. Chem. A* **2003**, *107*, 10933–10937.

[61] J. A. Maier, C. Martinez, K. Kasavajhala, L. Wickstrom, K. E. Hauser, C. Simmerling, *J. Chem. Theory Comput.* **2015**, *11*, 3696–3713.

[62] D. A. Case, R. M. Betz, D. S. Cerutti, T. E. C. III, T. A. Darden, R. E. Duke, T. J. Giese, H. Gohlke, A. W. Goetz, N. Homeyer, S. Izadi, P. Janowski, J. Kaus, A. Kovalenko, T. S. Lee, S. LeGrand, P. Li, C. Lin, T. Luchko, R. Luo, B. Madej, D. Mermelstein, K. M. Merz, G. Monard, H. Nguyen, H. T. Nguyen, I. Omelyan, A. Onufriev, D. R. Roe, A. Roitberg, C. Sagui, C. L. Simmerling, W. M. Botello-Smith, J. Swails, R. C. Walker, J. Wang, R. M. Wolf, X. Wu, L. Xiao, P. A. Kollman, *AMBER 16 and AmberTools16*, University Of California, San Francisco, **2016**.

[63] K. H. Park, M. Chhowalla, Z. Iqbal, F. Sesti, *J. Biol. Chem.* **2003**, *278*, 50212–50216.

[64] A. V. Makela, D. H. Murrell, K. M. Parkins, J. Kara, J. M. Gaudet, P. J. Foster, *Topics in Magnetic Resonance Imaging* **2016**, *25*, 177–186.

[65] Z. Zhang, N. Mascheri, R. Dharmakumar, D. Li, *Cytotherapy* **2008**, *10*, 575–586.

[66] A. Howes, L. Gabryšová, A. O'Garra, in *Reference Module in Biomedical Sciences*, Elsevier, **2014**.

[67] M. D. Sintchak, M. A. Fleming, O. Futer, S. A. Raybuck, S. P. Chambers, P. R. Caron, M. A. Murcko, K. P. Wilson, *Cell* **1996**, *85*, 921–930.

[68] J. Jain, S. J. Almquist, P. J. Ford, D. Shlyakhter, Y. Wang, E. Nimmesgern, U. A. Germann, *Biochem Pharmacol* **2004**, *67*, 767–776.

[69] I. Hellström, H. J. Garrigues, U. Garrigues, K. E. Hellström, *Cancer Res* **1990**, *50*, 2183–2190.

[70] I. Hellström, H. J. Garrigues, U. Garrigues, K. E. Hellström, *J Cell Biol* **1994**, *125*, 129–142.

[71] X. Ma, J. R. Idle, F. J. Gonzalez, *Expert Opin Drug Metab Toxicol* **2008**, *4*, 895–908.

[72] S. A. Kliewer, B. Goodwin, T. M. Willson, *Endocrine Reviews* **2002**, *23*, 687–702.

[73] D. Gupta, M. Venkatesh, H. Wang, S. Kim, M. Sinz, G. L. Goldberg, K. Whitney, C. Longley, S. Mani, *Clin Cancer Res* **2008**, *14*, 5332–5340.

[74] S. Mani, W. Dou, M. R. Redinbo, *Drug Metab Rev* **2013**, *45*, 60–72.

5. Graphene

5.1. Graphene: size matters

Up to this point, the size of the compounds that we investigated using the reverse screening protocol were comparable or slightly bigger than the average drug or active compound. The docking software that we repurposed for reverse screening, PatchDock[1], and the tools for the automated parametrization of non-standard molecules from the AMBER16 suite[2] had been designed to work with molecules of this size. Therefore, up until now it was all a matter of coordinating their execution with relatively simple automation tools, such as Bash scripts.

The game changes when we want to apply the same protocol to real nano systems, such as gold, silver, and silica nanoparticles or 2D materials such as graphene and phosphorene. For a particle is considered a "true" NP if at least one of its dimensions lies within the size range of 1–100 nm [3]. Gd@C_{82} largest dimension is 7,7 Å, which corresponds to the distance between the hexagonal face in front of the Gd atom and the opposing C-C bond along the C2v axis. The largest dimension of porphine is the distance between two β-carbons of two opposing pyrrolic sub-units, which corresponds to 8,45 Å. Only phthalocyanine possess a dimension larger than the minimum requisite of 1 nm, with a 13,03 Å distance between the furthest carbons of two opposing iso-indole sub-units. However, most nanoparticles that have been approved for clinical use or are actively been tested are larger than 10 nm [3–5]. The difference in dimensions is even more dramatic in the case of 2D materials, since their aggregates usually reach dimensions in the order of hundreds of nm [6–8].

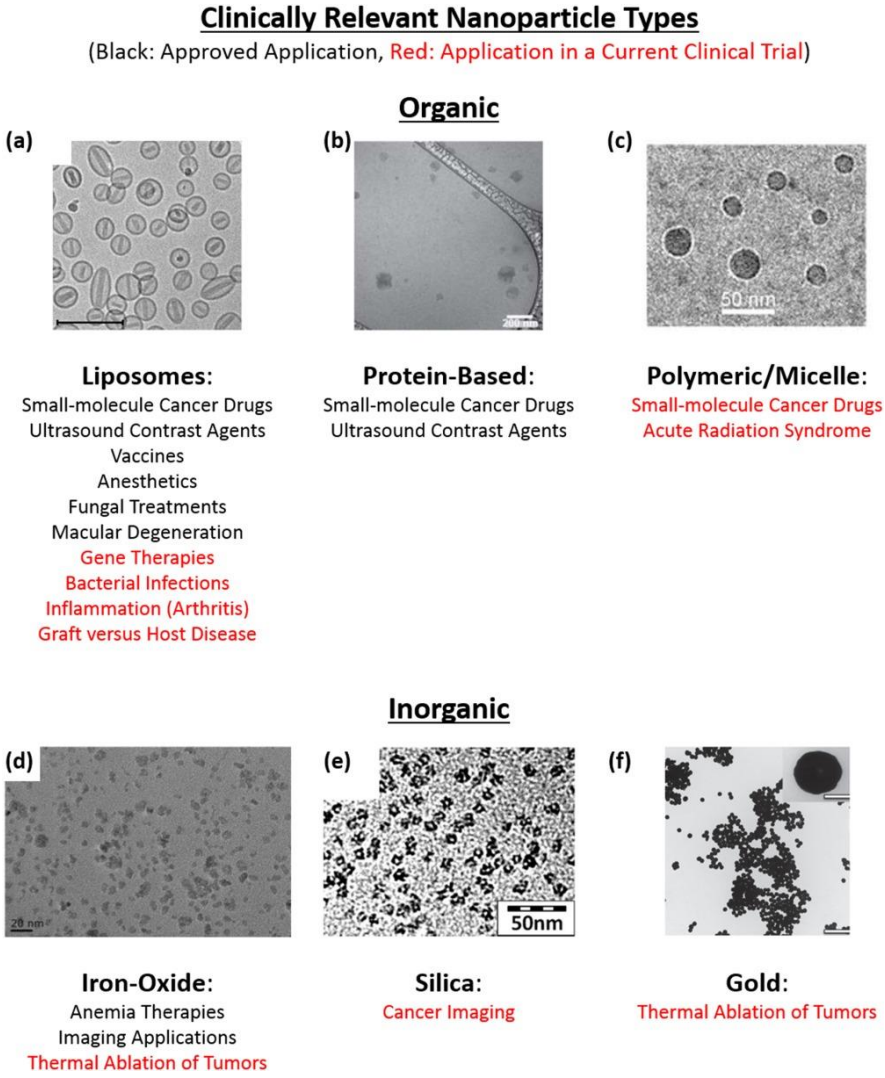


Figure 1 - Clinically relevant nanoparticles. Organic and inorganic nanoparticles have been approved for a variety of clinical indications (black text) and are being investigated in current clinical studies for additional indications (red text). Examples included (a) Doxil (200 nm scale bar), (b) Abraxane (200 nm scale bar), (c) CRLX101 (50 nm scale bar), (d) Feraheme (20 nm scale bar), (e) early iteration of Cornell Dots (50 nm scale bar), and (f) gold nanoshells (inset: 100 nm scale bar, main figure: 1,000 nm scale bar) from Nanospectra, makers of AuroLase)[5].

Although it is not necessary to represent every atom of the nanomaterial to gauge its interactions with a biomolecule, it is still important to represent the curvature of the nano-system correctly to use docking tools, since they all consider shape-complementarity between the two interacting entities, although in different ways. This is especially important for hydrophobic nanomaterials, such as graphene, since they interact with biomolecules *via* van der Waals interactions which scale with shape complementarity, an incorrect representation of the protein-nanomaterial interface would be significantly detrimental to the scoring phase of any protocol.

Regarding the overall size of the nanoparticle/material representation, it is necessary to represent the nanomaterial at equal in size than the interacting biomolecule to prevent two problem: incorrect pose generation and underestimation of solvation energies. Without atoms to generate unfavorable steric interactions, the space that corresponds to the 'imaginary' continuation of the material surface is seen as available empty space by docking algorithms during the translational space search of the ligand, resulting in the generation of docking candidates where parts of the biomolecule cross the 'imaginary' material surface.

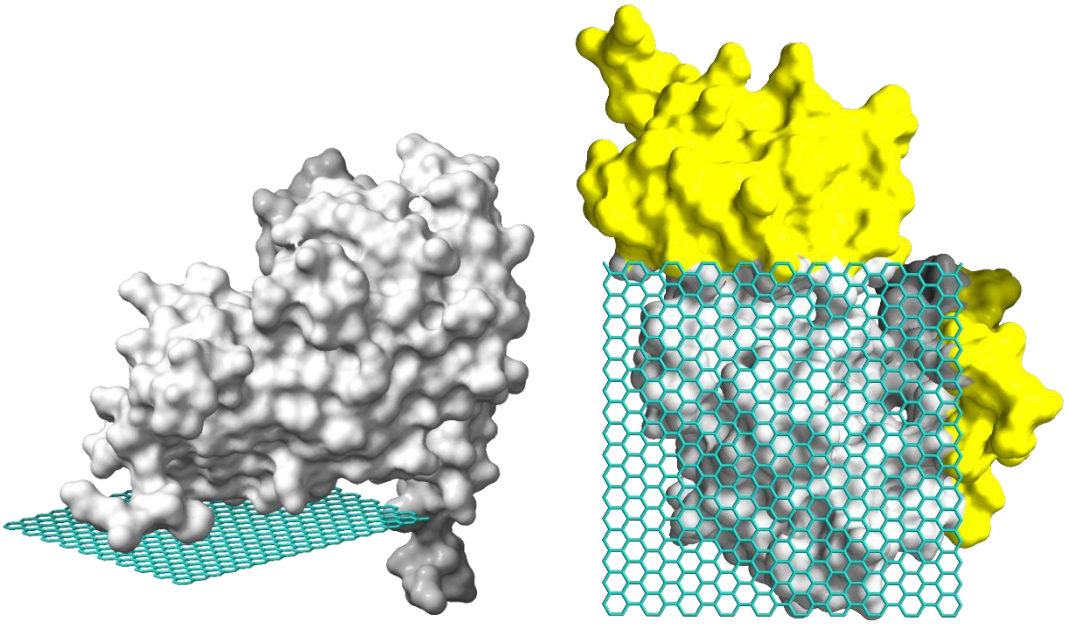


*Figure 2 – **Left**: Example of incorrect docking candidates' generation. Due to the limited amount of material atoms explicitly represented, the docking algorithm (PatchDock) generated a docking candidate that in a real-world scenario would not exist because of the penetration of the protein inside the surface of the nanomaterial aggregate. **Right**: although this 'face' of the protein is flat and would be in contact with an ideal real-size graphene surface, the parts directly not on top of the explicit nanoparticle atoms (yellow) are seen as surrounded by the solvent environment and their solvation contribution to the binding is not computed.*

In addition, even in the case of a proper docking candidate without steric clash with the 'imaginary' material surface, the forced desolvation caused by the latter is not considered for the residues that do not stand atop the explicit representation of the material surface. Because of the large area of contact that usually occurs in protein-nanoparticle interactions, this term is of paramount importance.

5.2. Impact of structure size on pose generation and scoring

The necessity of represent the nanoparticle with similar sizes with respect to the protein results has a big impact in both the docking candidates' generation and scoring phases of a hypothetical reverse screening protocol.

Regarding pose generation, the majority of docking tools available has been designed for screening small active molecules against large biomolecules; as result, they are unable to store and handle as many variables as the number of x,y,z coordinates of each atoms of a second protein-sized entity. In addition, many of them are based on *local shape feature matching* which generates docking candidates that are characterized by perfectly matching areas of contact at the expense of their extension, instead of candidates with a less complementary but more extended area of contact, which would be obtain better scores in the following Molecular Mechanics-based scoring phase. PatchDock[1] is capable of handling the coordinates of two protein-sized molecules since it has been designed to investigate antibody-antigen interactions as well. However, even in those scenarios performs local shape feature matching on the assumption that the area of contact for antigen-antibody recognition are relatively small and perfectly complementary.

Regarding the scoring phase, scoring functions based on Molecular Mechanics (MM) are the only feasible option, since nanoparticles differs strongly from drug-like molecules, even in the absence of partial charges, and accurate computation of solvation energies is necessary due to the large scale desolvation upon binding. However, automated tools for the parametrization of molecules that differ from proteins, nucleic acids and the most common lipids and sugar, known as *non-standard molecules*, can process molecules made of few hundred atoms at most. This is the case of the tool *antechamber* of the AmberTools16 suite[2], that was employed in the previous chapters, and the CGenFF[9] online server of the CHARMM suite[10]. These packages: i) assign Molecular Mechanics parameters that are compatible with the respective families of macromolecular forcefields, which are the most widespread for the representation of proteins in Molecular Mechanics; ii) build topology and parameters files which are compatible with the tools of their respective simulation

suites. All these steps are done in an automated way and can be easily integrated into automated protocols like the ones shown in the previous chapters.

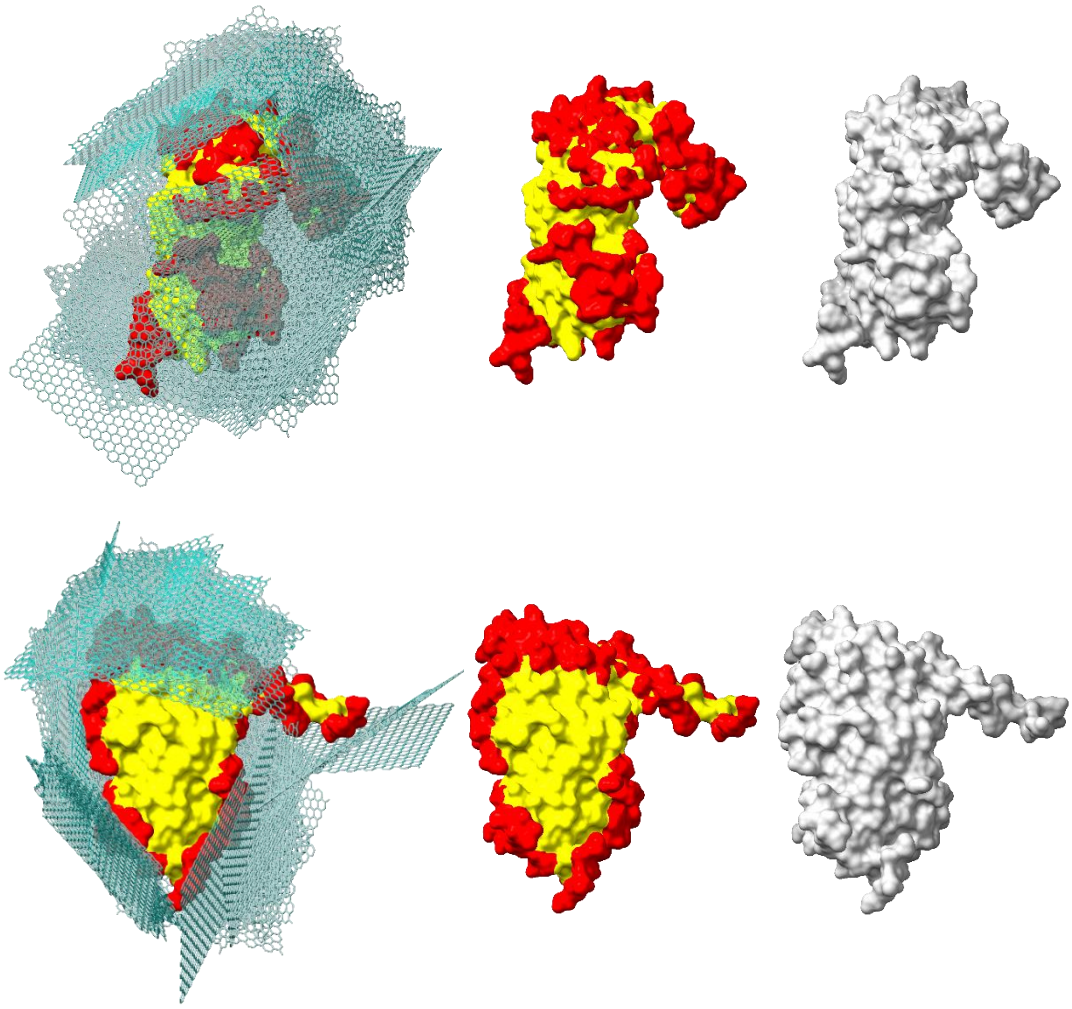


Figure 3 - Consequences of local shape feature matching in PatchDock: large areas with slight irregularities are ignored even if they are complementary to a large portion of the other molecule. ***Top****: side view of a protein surface with a large flat portion that has not been selected for the generation of poses with graphene;* ***Bottom****: point of view perpendicular to the unrecognized flat portion of the surface.* ***Red****: area in contact with at least one docking candidate in red;* ***Yellow****: area not at the interface in any docking candidate.* ***Each level, from left to right****: highlighted protein surface with the coordinates of graphene in every pose generated; highlighted protein surface, protein surface*

As result, it is impossible to employ the same protocols that have been previously outlined to real size nanoparticles. A new protocol must be designed that:

- can store a large number of coordinates for both the interacting entities;
- generates docking candidates based on shape complementarity alone;
- docking candidates' generation is not based on *local shape feature matching;* but of *brute-force enumeration of the transformation space;*

- ❖ parametrize large non-standard molecules in an automated way;
- ❖ accurately computes solvation energies.

5.3. Upgrading the reverse screening protocol

To meet these goals, PatchDock[1] has been replaced with protein-protein docking software ZDOCK 2.1[11]. ZDOCK 2.1 is a rigid-body grid-based docking algorithms that generates docking candidates *via* brute-force enumeration using a Fast Fourier Transform algorithm to quickly explore the translational space of the ligand. Just like PatchDock, ZDOCK 2.1 only considers the shape complementarity of the interacting entities in both pose generation and scoring. More modern algorithms for protein-protein docking use built-in atom types and residue classifiers to match residues with complementary non-bonded interactions in the pose generation phase and employ experimental-based or knowledge-based scoring[12]. Although, these features allow them to reach a higher predictive power regarding protein-protein interactions[12], it prevents their application to chemical entities other than proteins while ZDOCK 2.1 can be applied to any system since the only chemical information that it requires are the van der Waals radius of the interacting atoms. In addition, its scoring function, referred as Pairwise Shape Complementarity, rewards docking candidates with a continuous area of contact between the two interacting entities, giving it an edge in the investigation of protein-nanoparticle interactions. However, ZDOCK 2.1 is not free of disadvantages:

- ❖ it is free only for academic or non-profit associations, preventing the application of this protocol in industrial settings;
- ❖ it is not open-source;
- ❖ the available precompiled versions do not support parallelization, neither *via* OpenMP, nor *via* MPI, and do not support acceleration using GPGPU;
- ❖ It allows a certain degree of penetration of the ligand surface into the receptor surface to account flexibility to some degree. This is especially problematic in the MM-based scoring phase since atomic superimpositions result in spikes of positive, i.e., destabilizing, non-bonded interaction energies.

It is possible to overcome the performance related limitations by running several instances of the protocol at the same time, each on a separate core of a CPU; since

PatchDock does not support parallelization as well, the protocols described in the precious chapters have been used in the same way. The surface penetration issue is solved by describing the nanomaterial with few additional layers other than the surface-atoms layer, by doubling the van der Waals radius for the latter and by adding a 'mock' surface layer on top of it. The 'mock' atoms are vertical projections at vdW radius x2 distance of the surface atoms, unlike the atoms of the additional layers below, and possess the same doubled vdW radius of the surface atoms. Since the algorithm rewards docking candidates without superimposition between core atoms, i.e., atoms that cannot be access by a solvent probe, this effectively limits the surface penetration at the level of the 'mock' atom layer, preserving the real surface layer which is used alone in the following scoring phase.

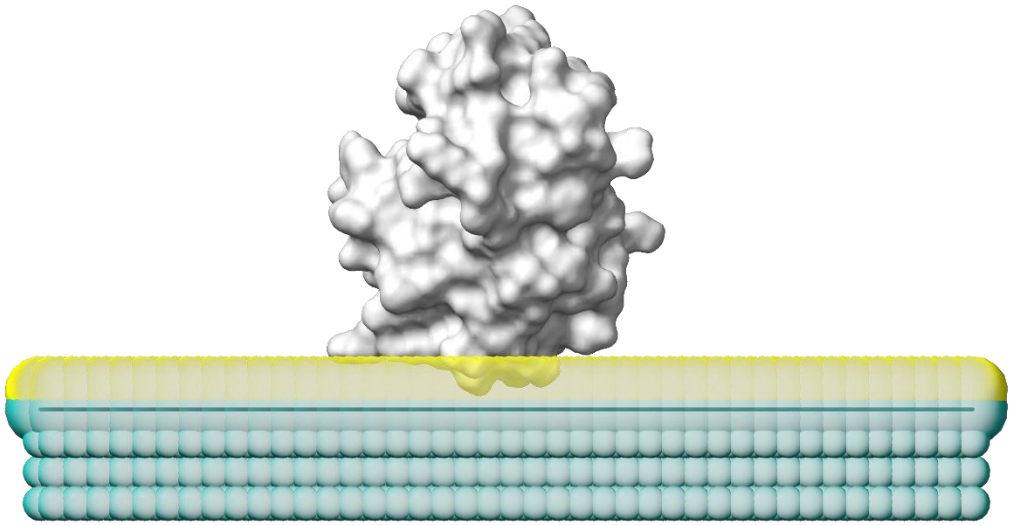


Figure 4 - ZDOCK 2.1 penetration is solved by adding a 'mock' surface atom layer (yellow) and by tuning the van der Waals radius of the nanoparticle atoms. The van der Waals radii of the atoms belonging to the real surface and the 'mock' surface are doubled to prevent steric clashes with the atoms of the real surface layer. All layers are represented as transparent van der Waals spheres; in addition, the nuclei of the real surface layer are represented as 'sticks'.

In addition, since nanoparticles and 2D materials have regular surfaces characterized by constant curvature, docking programs tend to dock the protein structures in several positions on the latter. This behavior results in a large number of docking candidates that are completely identical in terms of relative orientation of the biomolecule with respect to the nanoparticle surface. To partially prevent this issue,

it is necessary to specify a group of atoms in the middle of the nanoparticle surface as the nanoparticle binding site in the input files of ZDOCK2.1.

To solve the issue of the automated generation of parameters and topology's files, we decided to partially move away from the AmberTools16[2] suite and incorporate in the protocol GROMACS 5.1[13], an Free and Open-Source (FOSS) suite for Molecular Mechanics calculations that is used worldwide on HPC infrastructures thanks to its high performance and flexibility. GROMACS has access to the tool *x2top* which allows the creation of parameters and topology files for large non-standard molecules if provided with the rules for atom-type assignment on the basis of connectivity and atomic number of the neighboring atoms. *x2top* is a primitive tool that cannot assigning correct atom-types if simply given a set of atomic coordinates structure, unlike the *antechamber* tool and the CGenFF online server. However, it is very effective for nanoparticles since they are comprised of only few different atom-types, which simplifies the creation of the atom-types assignment rules. In addition, GROMACS topology and coordinates file are easily editable using common Bash text editing tools, such as *sed*, *awk* etc., thanks to their formatting. As result, it is possible to effectively create, manipulate and combine molecular structure in ways that are impossible with AMBER format files except by using the tools included in the installation that do not support large non-standard molecules. An exception is represented by the tools *CPPTRAJ*[14] and *ParmED*[15,16] which are included in AmberTools installations, are capable of processing said molecules in certain conditions and have been included in the new protocol as result. Ambertools16 *sander* molecular simulation engine will still be used for the optional Molecular Mechanics energy minimization of the unbound protein structures, as well as the built-in Python-based MM-GBSA scripts[17].

Regarding the choice of scoring function, we decided to continue using MM-GBSA since it is a good tradeoff between predictive accuracy and speed of computation. However, given the importance of solvation contribution, it is used as scoring function for every protein of the database. As an additional layer of accuracy, the protocol supports Molecular Mechanics-Poisson Boltzmann Surface Area (MM-PBSA) calculations *via* the FOSS tool *g_mmpbsa*[18]. *g_mmpbsa* was developed as part of the Open Source Drug Discovery (OSDD) consortium and implements the MM-

PBSA approach by combining subroutines from GROMACS and the *apbs* package[19], a powerful FOSS solver of the Poisson-Boltzmann equations.

g_mmpbsa is available as

- source code;
- precompiled package that includes a built-in version of *pbsa* with OpenMP support;
- precompiled package to be linked to an external installation of *pbsa (*the latter can be compiled with MPI and be employed on HPC architectures).

Each of the options above is available with GROMACS subroutines taken from versions 4.5, 4.6, 5.0 and 5.1. We decided to employ the precompiled version with built-in *apbs* subroutines and the latest available GROMACS subroutines, i.e., version 5.1, since MPI support is only meaningful on multi-node servers.

The first version of the protocol was devised during my research period abroad in the laboratory of Professor Dario Greco, Tampere University. The goal was to investigate the interactions between gold, silver and silica ultrasmall nanoparticles (d<5nm) and the proteins expressed by a series of genes whose expression was perturbated after specific human cell lines were exposed to said nanoparticles. The resulting relative binding energy values would have been used as an additional descriptor of the gene in a co-expression network analysis of the cell's transcriptome. In addition, since the protein expressed by the perturbated genes were not part of a curated structural database, the original version of the protocol used an auxiliary script that given a list of proteins, each with its own list of PDB identifiers, was able to: i) download the structure file with the best resolution for each of them; ii) "clean" them, i.e., removing headers, ligand/solvent/ions coordinates, changing old residue names with modern ones etc. This original version used exclusively GROMACS 5.1[13] tools, *g_mmpbsa*[18] and ZDOCK2.1[20] in addition to Bash tools, such as *sed*, *awk* and *grep*. Although its application was successful, it was unable to optimize large protein structures since GROMACS does not support multi-core energy minimizations in vacuum or implicit solvent and relied only on the computationally expensive MM-PBSA scoring function.

5.4. Analysis of the upgraded protocol for 2D materials

The version that will be conceptually presented in this paragraph is an upgrade on the original; it solves the previous limitations by integrating tools from AmberTools16[2] and is better suited in the investigation of 2D materials such as graphene. The protocol is divided into two scripts:

- the *pre-screening* script that optionally optimizes the protein structures via MM energy minimization, maps their surfaces for ZDOCK2.1[20] and optionally calculates their solvation energies using Poisson-Boltzmann with *g_mmpbsa*[18];
- the *screening* script that generates docking candidates, performs a clustering analysis of said candidates and computes the binding energies of the best structures for each cluster, optionally with MM-PBSA via *g_mmpbsa*[18] instead of the default MM-GBSA via *MMPBSA.py*[17] from AmberTools16[2].

Energy minimization has been deemed optional since it does not change the overall shape of the protein significantly.

In line with the reverse screening protocols described in the previous chapters, the docking candidates are described at the Molecular Mechanics level with the ff14SB AMBER forcefield for the protein residues and the Generalized Amber ForceField (GAFF) for the nanoparticle atoms.

The protein structure energy minimization and the docking candidates' scoring step via MM-GBSA are performed using the simulation parameters reported in Table 1.

Protein structure minimization	
Target of optimization	Entire protein
Steps of steepest descent	250
Steps of conjugate gradient	250
Solvent	Implicit, igb5[21]
Electrostatic treatment	Cut-off (no PME)
Non-bonded interaction cut-off	20 Å
Molecular Mechanics – Poisson-Boltzmann (Generalized Born) Surface Area	
Solvation model	igb5[21]

Table 1 - Computational details for the MM refinement and MM-GBSA scoring parts of the advanced docking protocol.

The parameters for the optional MM-PBSA computation were chosen in line with the benchmark results of the *g_mmpbsa* paper[18] and are reported in Table 1; the actual *apbs* keywords that must be specified in the input files are reported between in the second column.

Grid		
Factor to multiply molecular dimensions to obtain coarse-grid dimension	cfac	2
Fine mesh spacing	gridspace	0.5 Å
Distance to add to molecular dimensions to get fine-grid dimensions	fadd	20 Å
Polar solvation calculation		
PB equation to solve	PBsolver	lpbe
Type of boundary condition to solve PB equation	bcfl	mdh
Positive ions:		
Charge	pcharge	1
Radius (Å)	prad	0.95 Å
Concentration (M)	pconc	0.150 M
Negative ions:		
Charge	ncharge	1
Radius (Å)	nrad	1.81 Å
Concentration (M)	nconc	0.150 M
Dielectric constant:		
Solute	pdie	2
Solvent	sdie	80
Vacuum	vdie	1
Solvent probe radius	srad	1.4
Method used to map biomolecular charges on grid	chgm	spl4
Model to construct dielectric/ionic boundary	srfm	smol
Value for cubic spline window	swin	0.30
Number of grid points per Å2	sdens	10
Temperature (K)	temp	300
Polar solvation calculation		
Surface tension (γ) (KJ $mol^{-1}Å^{-2}$)	gamma	0.0226778
Probe radius (Å)	sasrad	1.4
Offset (KJ mol^{-1})	sasaconst	3.84928

Table 2 - PBSA parameters employed in the protocol. The second column contains the specific keywords that must be passed to the apbs subroutines from the g_mmpbsa input file.

Conversion between AMBER and GROMACS files has been performed using a combination of Bash editing tools, such as *sed*, *awk*, *grep*, and the FOSS tools *ACPYPE*[22], *ParmED*[16,17] and *CPPTRAJ*[14].

Although the protocol was successfully tested on a selected ensemble of proteins, it will only be applied for the investigation of the PDTD[23] in the near future.

Each protein of the database must have a separate sub-directory in the *working directory*, i.e., the directory from which the scripts must be launched. Each protein sub-directory must contain the protein 3D structure in .pdb format; the name of the folder must be identical to the name of its .pdb structure file.

The nanoparticle must have its own sub-directory of the *working* directory and the user must manually prepare the following files for each nanoparticle:

- np.gro
  - GROMACS coordinate file of nanoparticle, with 3-digit precision. Can be generated from a .pdb file using the GROMACS tool *editconf.*
- np_sur.pdb
  - Nanoparticle surface for ZDOCK2.1 generated with *mark_sur*, a binary provided with the ZDOCK 2.1 package.
- forcefield.itp
  - Nanoparticle parameters, compatible with AMBER macromolecular forcefields, in GROMACS format. The name cannot be changed since GROMACS uses it as the default name for every forcefield included in its installation and its subroutines search for it. Unfortunately, no tools are available for the automated creation of this file, so the user must manually convert the parameters to GROMACS format according to the documentation of the latter.
- np.itp
  - GROMACS "include topology" file, which contains atom-names, atom-types and connectivity of the nanoparticle but not parameters or simulation directives. It is generated using the tool *x2top.*
- np_polar.xvg
  - Optional, polar solvation energy of the unbounded nanoparticle for MM-PBSA scoring, computed using *g_mmpbsa.*
- np_apolar.xvg
  - Optional, apolar solvation energy of the unbounded nanoparticle for MM-PBSA scoring, computed using *g_mmpbsa.*

The first and last two files must be stored in the nanoparticle sub-directory, the remaining two in a directory that must be specified by the user in the script body before execution. The script also assumes that:

- the following auxilliary functions are stored in home/$USER/utils:
  - aesthetics.sh
  - error_functions.sh
  - pdb_editing.sh
- the sub-directory containing ZDOCK2.1 binaries is located in the standard location for pre-compiled binaries, i.e., home/$USER/opt

The full *pre-screening* and *screening* scripts are available in Appendix B, alongside the auxiliary Bash functions mentioned before. Although this protocol is supposed to be used for the investigation of an already curated structural database, such as the PDTD[23], the workflow of the *database-maker* script will be presented in this paragraph; however, the full script is also available inside Appendix B.
In the following graphical representations of the algorithm, each block is a different part of the script that can be 'activated' by the user in the first section of the script body. The scripts have been created with a modular structure to allow maximum flexibility for the user and help in case of errors. Regarding the latter, the scripts are characterized by a series of debugging functions that stop the script if certain conditions are not met at key passages; in addition, these debugging functions generate useful log file that can help in the identification of the problems.

The following color code has been applied to each block according to its role:

- RED = Protein structures' list, editing, optimization.
- YELLOW = Creation of files necessary for docking candidates' generation and docking generation itself.
- GREEN = Creation of files that describe the docking candidates' structure and clustering.
- BLUE = Computation of scoring components (solvation energies and molecular mechanics energy) and generation of the relative files.

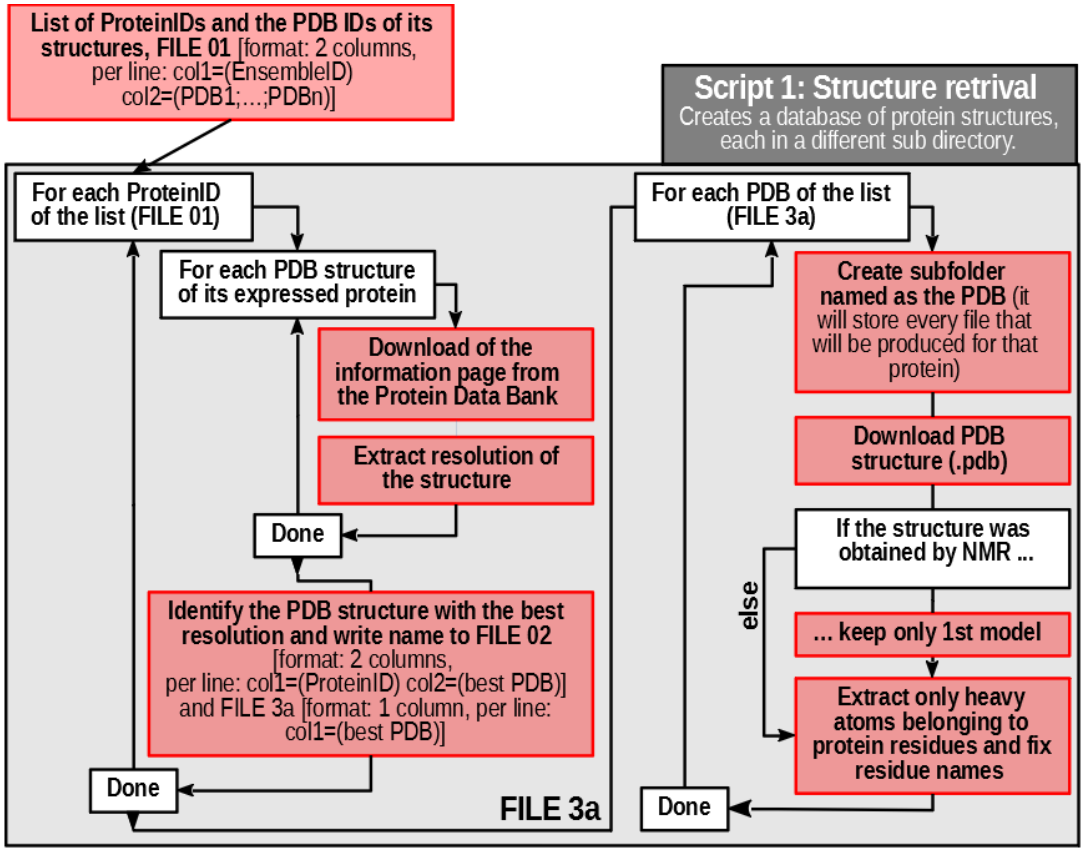


Figure 5 - Graphical representation of the algorithm of the database-maker script. In contrast to what was said previously, each block does not correspond to a part that can be turned on and off by the user but has been separated for the sake of clarity. Instead, the user can control the execution of each 'for loop'

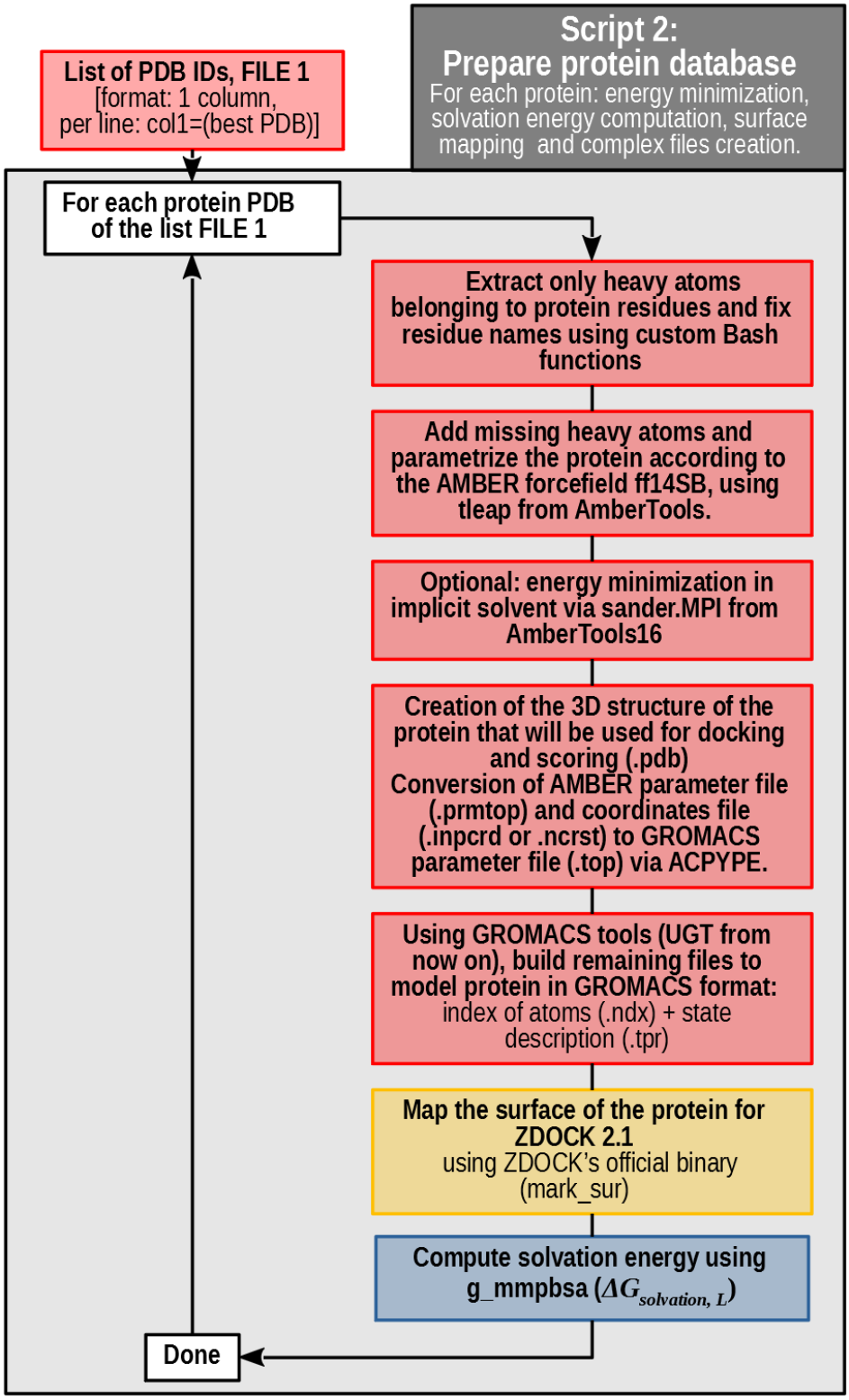


Figure 6 - Graphical representation of the algorithm of the pre-screening script. In particular, the solvation energy of the protein is computed in a separate moment with respect to the solvation energy of each docking candidate, since it is constant among the various candidate because the structure is kept rigid. As result, in the next script only the solvation energy of the complex as a whole will be computed instead of computing the solvation energies of each interacting partner as well, saving precious computational time.

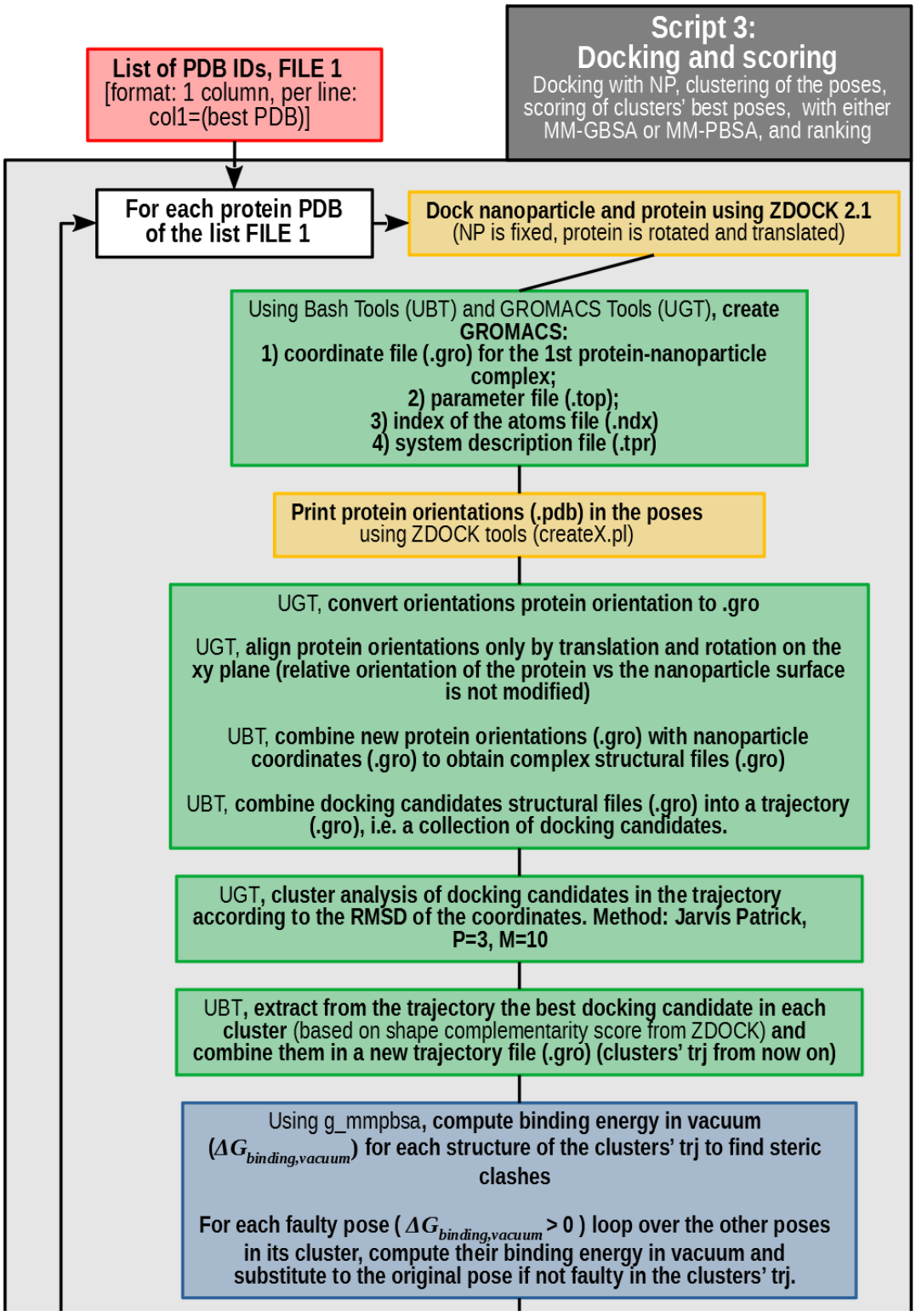


Figure 7 - Graphical representation of the algorithm of the screening script (part1)

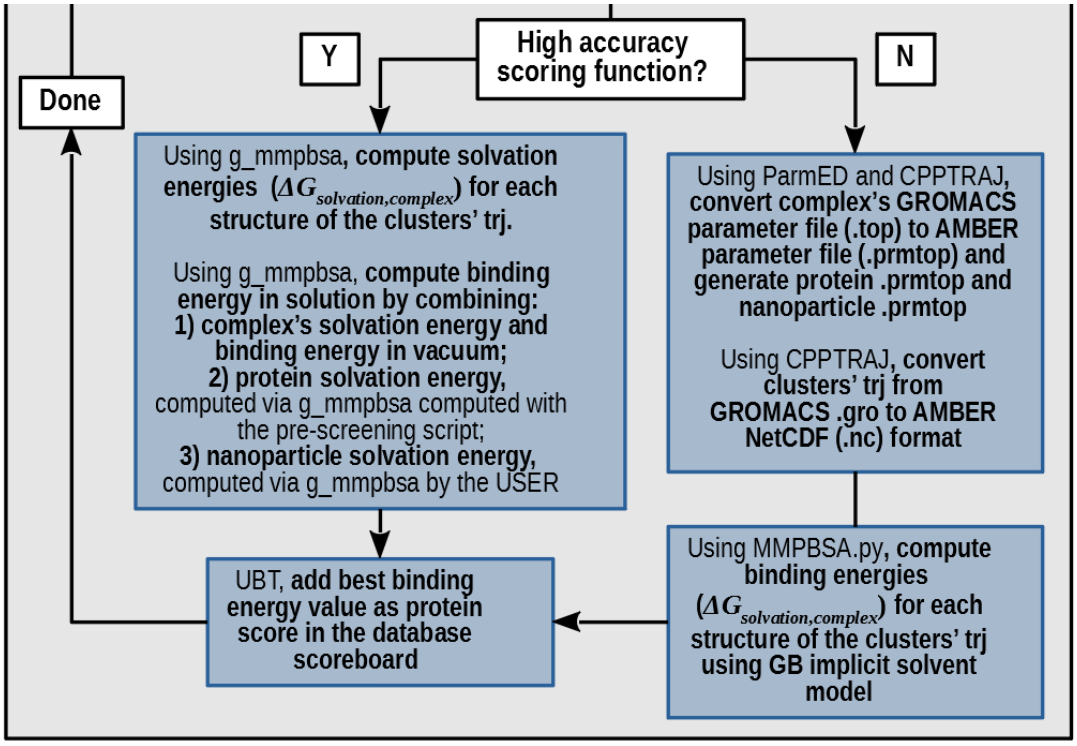


Figure 8 - Graphical representation of the algorithm of the pre-screening script (part 2)

5.5. References

[1] D. Duhovny, R. Nussinov, H. J. Wolfson, in *Algorithms in Bioinformatics* (Eds.: R. Guigó, D. Gusfield), Springer, Berlin, Heidelberg, **2002**, pp. 185–200.

[2] D. A. Case, R. M. Betz, D. S. Cerutti, T. E. C. III, T. A. Darden, R. E. Duke, T. J. Giese, H. Gohlke, A. W. Goetz, N. Homeyer, S. Izadi, P. Janowski, J. Kaus, A. Kovalenko, T. S. Lee, S. LeGrand, P. Li, C. Lin, T. Luchko, R. Luo, B. Madej, D. Mermelstein, K. M. Merz, G. Monard, H. Nguyen, H. T. Nguyen, I. Omelyan, A. Onufriev, D. R. Roe, A. Roitberg, C. Sagui, C. L. Simmerling, W. M. Botello-Smith, J. Swails, R. C. Walker, J. Wang, R. M. Wolf, X. Wu, L. Xiao, P. A. Kollman, *AMBER 16 and AmberTools16*, University Of California, San Francisco, **2016**.

[3] L. Yildirimer, N. T. K. Thanh, M. Loizidou, A. M. Seifalian, *Nano Today* **2011**, *6*, 585–607.

[4] C. L. Ventola, *P T* **2017**, *42*, 742–755.

[5] A. C. Anselmo, S. Mitragotri, *Bioengineering & Translational Medicine* **2016**, *1*, 10–29.

[6] J. T. Choi, D. H. Kim, K. S. Ryu, H. Lee, H. M. Jeong, C. M. Shin, J. H. Kim, B. K. Kim, *Macromol. Res.* **2011**, *19*, 809–814.

[7] J. Amaro-Gahete, A. Benítez, R. Otero, D. Esquivel, C. Jiménez-Sanchidrián, J. Morales, Á. Caballero, F. J. Romero-Salguero, *Nanomaterials* **2019**, *9*, 152.

[8] D. Li, M. B. Müller, S. Gilje, R. B. Kaner, G. G. Wallace, *Nature Nanotechnology* **2008**, *3*, 101–105.

[9] K. Vanommeslaeghe, E. Hatcher, C. Acharya, S. Kundu, S. Zhong, J. Shim, E. Darian, O. Guvench, P. Lopes, I. Vorobyov, A. D. Mackerell, *Journal of Computational Chemistry* **2010**, *31*, 671–690.

[10] B. R. Brooks, C. L. Brooks, A. D. Mackerell, L. Nilsson, R. J. Petrella, B. Roux, Y. Won, G. Archontis, C. Bartels, S. Boresch, A. Caflisch, L. Caves, Q. Cui, A. R. Dinner, M. Feig, S. Fischer, J. Gao, M. Hodoscek, W. Im, K. Kuczera, T. Lazaridis, J. Ma, V. Ovchinnikov, E. Paci, R. W. Pastor, C. B. Post, J. Z. Pu, M. Schaefer, B. Tidor, R. M. Venable, H. L. Woodcock, X. Wu, W. Yang, D. M. York, M. Karplus, *Journal of Computational Chemistry* **2009**, *30*, 1545–1614.

[11] R. Chen, Z. Weng, *Proteins* **2003**, *51*, 397–408.

[12] P. Kangueane, C. Nilofer, in *Protein-Protein and Domain-Domain Interactions*, Springer Singapore, Singapore, **2018**, pp. 161–168.

[13] M. J. Abraham, D. van der Spoel, E. Lindahl, B. Hess, the GROMACS development team, *GROMACS 5.1.5*, **2017**.

[14] D. R. Roe, T. E. Cheatham, *J. Chem. Theory Comput.* **2013**, *9*, 3084–3095.

[15] J. M. Swails, C. Klein, M. R. Shirts, *ParmED*, **n.d.**

[16] M. R. Shirts, C. Klein, J. M. Swails, J. Yin, M. K. Gilson, D. L. Mobley, D. A. Case, E. D. Zhong, *J Comput Aided Mol Des* **2017**, *31*, 147–161.

[17] B. R. Miller, T. D. McGee, J. M. Swails, N. Homeyer, H. Gohlke, A. E. Roitberg, *J. Chem. Theory Comput.* **2012**, *8*, 3314–3321.

[18] R. Kumari, R. Kumar, Open Source Drug Discovery Consortium, A. Lynn, *J. Chem. Inf. Model.* **2014**, *54*, 1951–1962.

[19] E. Jurrus, D. Engel, K. Star, K. Monson, J. Brandi, L. E. Felberg, D. H. Brookes, L. Wilson, J. Chen, K. Liles, M. Chun, P. Li, D. W. Gohara, T. Dolinsky, R. Konecny, D. R. Koes, J. E. Nielsen, T. Head-Gordon, W. Geng, R. Krasny, G.-W. Wei, M. J. Holst, J. A. McCammon, N. A. Baker, *Protein Science* **2018**, *27*, 112–128.

[20] R. Chen, L. Li, Z. Weng, *Proteins: Structure, Function, and Bioinformatics* **2003**, *52*, 80–87.

[21] A. Onufriev, D. Bashford, D. A. Case, *Proteins* **2004**, *55*, 383–394.

[22] A. W. Sousa da Silva, W. F. Vranken, *BMC Research Notes* **2012**, *5*, 367.

[23] Z. Gao, H. Li, H. Zhang, X. Liu, L. Kang, X. Luo, W. Zhu, K. Chen, X. Wang, H. Jiang, *BMC Bioinformatics* **2008**, *9*, 104.

6. Conclusions

Molecular docking, a common *in silico drug design* tool, has been successfully repurposed for investigating the interaction between a single molecule of interest and a collection of biomolecules. The new protocol, called *reverse screening*, is particularly effective in the case rigid and hydrophobic molecules since the shortcomings of molecular docking itself do not have a significant impact. Reverse screening protocols can help in the identification of new therapeutical targets, predicting toxicological effects and unwanted interactions or the design of new therapeutic platforms based on the conjugation between a synthetic compound and a protein carrier.

In this work, reverse screening has been applied to porphine and phthalocyanine, two chemically related photosensitizers employed in photodynamic therapy, and $Gd@C_{82}$, the most promising endohedral gadofullerene for theranostic applications. The results of this work suggest that in the case of rigid, hydrophobic molecules with structures that are fundamentally similar to drugs or active compounds, such as porphine and phthalocyanine, built-in scoring functions of docking tools that take strongly in consideration shape complementarity are more accurate than more flexible and advanced methods based on the law of physics (force-field based scoring functions). This is due to the fact that they are based on chemico-physical and structural data of drug-protein adducts, which are abundant, while every computational method carries a certain amount of approximations and inaccuracy.

In the case of more 'exotic' molecules, such as $Gd@C_{82}$, built-in scoring functions are incapable of identifying the correct mode of interaction since they differ strongly from the molecules for which the scoring function has been tuned and perfected. Force-field based scoring functions offer the necessary flexibility, albeit at a higher more computational cost.

While it is theoretically possible to use reverse screening protocols to investigate the interactions between nanoparticles and biomolecules, a current hot-topic in research, the sheer number of atoms that must be considered and the nanoparticle size result in a plethora of problems since docking tools have been designed with small active molecules in mind. The same applies in the case of force-field based tools as well

since they have been designed to handle large molecular entities only in the case of biomolecules, such as proteins and nucleic acids. Nevertheless, by repurposing protein-protein docking tools and combining them with force-field bases scoring functions it has been possible to investigate the interactions between gold, silver and silica nanoparticles and proteins. Although this initial version of the protocol carried many shortcomings, a new protocol has been devised and presented in the last chapter. This new protocol has shown promising results in the case of hydrophobic 2D materials and will be applied in the near future for the reverse screening investigation of graphene.

7. Appendix A: reverse screening scoreboards

7.1. Porphine scoreboard

Rank	PDB_ID	Name
1	8CAT	Beef Liver Catalase
2	1OG5	Cytochrome P450 2C9
3	2F9Q	Cytochrome P450 2D6
4	1SPG	Carbonmonoxy Hemoglobin
5	1K74	Retinoic Acid Receptor RXR-Alpha
6	1BVY	Protein (Cytochrome P450 BM-3)
7	1NEK	Succinate Dehydrogenase
8	1PSO	Human Pepsin
9	1I8O	Cytochrome C2
10	1E55	Beta-Glucosidase
11	1HN4	Prophospholipase A2
12	1BBP	Bilin Binding Protein
13	1VID	Catechol O-Methyltransferase
14	1OIQ	Protein Kinase 2
15	1QIJ	Acetylcholinesterase
16	1VE9	D-Amino Acid Oxidase
17	1PQ2	Human Cytochrome P450 2C8
18	1R9O	Human Cytochrome P450 2C9
19	1TVR	HIV-1 Reverse Transcriptase
20	2IFB	Intestinal Fatty Acid Binding Protein
21	1DIS	Dihydrofolate Reductase
22	1A7C	Human Plasminogen Activator Inhibitor Type-1
23	1PHD	Cytochrome P450-CAM
24	1K7L	Peroxisome Proliferator Activated Receptor Alpha
25	1E86	Cytochrome C'
26	1IB1	Serotonin N-Acetyltransferase
27	1Z57	Dual Specificity Protein Kinase CLK1
28	1EFR	Mitochondrial F1-Atpase Subunit Alpha
29	1ACM	Aspartate Carbamoyltransferase
30	1Y0S	Peroxisome Proliferator Activated Receptor Delta
31	1FTK	Glutamate Receptor Subunit 2
32	1R0P	Hepatocyte Growth Factor Receptor
33	2MBR	Uridine Diphospho-N-Acetylenolpyruvylglucosamine Reductase
34	1K6W	Cytosine Deaminase
35	1Q5M	Prostaglandin-E2 9-Reductase

36	3MDE	Medium Chain Acyl-CoA Dehydrogenase
37	1Y79	Dipeptidyl Carboxypeptidase
38	1PHA	Cytochrome P450-CAM
39	1H9U	Human Retinoid X Receptor, Beta
40	1CR1	DNA Primase/Helicase
41	1GOX	(S)-2-Hydroxy-Acid Oxidase, Peroxisomal
42	2BRO	Serine/Threonine-Protein Kinase Chk1
43	1EET	Hiv-1 Reverse Transcriptase
44	2CPP	Cytochrome P450-CAM
45	1P4G	Glycogen Phosphorylase, Muscle Form
46	1E1F	Beta-Glucosidase
47	8CPP	Cytochrome P450-CAM
48	2HCK	Hematopoietic Cell Kinase HCK
49	1OIT	Cell Division Protein Kinase 2
50	1EFA	Lac Repressor
51	1AEV	Cytochrome C Peroxidase
52	3DFR	Dihydrofolate Reductase
53	1SEZ	Protoporphyrinogen Oxidase, Mitochondrial
54	1PYY	Penicillin-Binding Protein 2X, Streptococcus Pneumoniae
55	7CPP	Cytochrome P450-CAM
56	5CPP	Cytochrome P450-CAM
57	1QD6	Outer Membrane Phospholipase (OMPLA)
58	6COX	Cyclooxygenase-2
59	3HSC	Heat-Shock Cognate 70KD Protein
60	1WKD	tRNA-Guanine Transglycosylase
61	1AEE	Cytochrome C Peroxidase
62	1IKX	Mutant Hiv-1 Reverse Transcriptase
63	1P0N	Isopentenyl-Diphosphate Delta-Isomerase
64	1I7G	Peroxisome Proliferator Activated Receptor Alpha
65	1OIR	Cell Division Protein Kinase 2
66	1ZHY	KES1 Protein
67	2J0D	Cytochrome P450 3A4
68	1AC8	Cytochrome C Peroxidase
69	1K8Q	Triacylglycerol Lipase, Gastric
70	6CPP	Cytochrome P450-CAM
71	1JQ9	Phospholipase A2
72	2DDH	Acyl-CoA Oxidase
73	1CQE	Prostaglandin H2 Synthase-1
74	1D3G	Dihydroorotate Dehydrogenase
75	1BU5	Flavodoxin
76	1KAE	Histidinol Dehydrogenase
77	1DIY	Prostaglandin H2 Synthase-1
78	1SQ5	Pantothenate Kinase
79	1QPB	Pyruvate Decarboxylase (Form B)
80	2A3L	AMP Deaminase
81	1PXX	Prostaglandin G/H Synthase 2

82	1RA2	Dihydrofolate Reductase
83	1PBD	P-Hydroxybenzoate Hydroxylase
84	1K02	Nadph Dehydrogenase 1
85	1AEU	Cytochrome C Peroxidase
86	1AEO	Cytochrome C Peroxidase
87	1AEN	Cytochrome C Peroxidase
88	1AC4	Cytochrome C Peroxidase
89	1HT8	Prostaglandin H2 Synthase-1
90	1MCS	Immunoglobulin Lambda Dimer Mcg (Light Chain)
91	1IJE	Elongation Factor 1-Alpha
92	1F8N	Lipoxygenase-1
93	1DBM	Anti-Progesterone Antibody DB3
94	1IBJ	Cystathionine Beta-Lyase
95	1D8R	Dihydrofolate Reductase
96	1ACJ	Acetylcholinesterase
97	1PHG	Cytochrome P450-CAM
98	1AEQ	Cytochrome C Peroxidase
99	1AEM	Cytochrome C Peroxidase
100	1AEK	Cytochrome C Peroxidase

7.2. Phthalocyanine scoreboard

Rank	PDB_ID	Name
1	2F9Q	Cytochrome P450 2D6
2	2J0D	Cytochrome P450 3A4
3	8CAT	Beef Liver Catalase
4	1PHA	Cytochrome P450-CAM
5	7CPP	Cytochrome P450-CAM
6	2CPP	Cytochrome P450-CAM
7	5CPP	Cytochrome P450-CAM
8	1PHG	Cytochrome P450-CAM
9	8CPP	Cytochrome P450-CAM
10	1BVY	Cytochrome P450 BM-3
11	1NEK	Succinate Dehydrogenase
12	1PHD	Cytochrome P450-CAM
13	1Z57	Dual Specificity Protein Kinase CLK1
14	2F4B	Peroxisome Proliferator-Activated Receptor Gamma
15	1ZHY	KES1 Protein
16	1SPG	Carbonmonoxy Hemoglobin
17	1AEE	Cytochrome C Peroxidase
18	1K8Q	Triacylglycerol Lipase, Gastric
19	1AEQ	Cytochrome C Peroxidase
20	1AEM	Cytochrome C Peroxidase

21	1AEK	Cytochrome C Peroxidase
22	1AEH	Cytochrome C Peroxidase
23	1AEG	Cytochrome C Peroxidase
24	1AEF	Cytochrome C Peroxidase
25	1AED	Cytochrome C Peroxidase
26	1AEB	Cytochrome C Peroxidase
27	1R18	Protein-L-Isoaspartate(D-Aspartate)-O-Methyltransferase
28	1AC8	Cytochrome C Peroxidase
29	6CPP	Cytochrome P450-CAM
30	1AEU	Cytochrome C Peroxidase
31	1AEO	Cytochrome C Peroxidase
32	1AEN	Cytochrome C Peroxidase
33	1AC4	Cytochrome C Peroxidase
34	1CQE	Prostaglandin H2 Synthase-1
35	1AEV	Cytochrome C Peroxidase
36	1ACM	Aspartate Carbamoyltransferase
37	1GII	Cell Division Protein Kinase 2
38	1CX2	Cyclooxygenase-2
39	1QIJ	Acetylcholinesterase
40	1OSH	Bile Acid Receptor
41	1K7L	Peroxisome Proliferator Activated Receptor Alpha
42	1BBP	Bilin Binding Protein
43	1AET	Cytochrome C Peroxidase
44	1STC	cAMP-Dependent Protein Kinase
45	1ANG	Human Angiogenin
46	1OIT	Cell Division Protein Kinase 2
47	3HSC	Heat-Shock Cognate 70KD Protein
48	1PXX	Prostaglandin G/H Synthase 2
49	1P0N	Isopentenyl-Diphosphate Delta-Isomerase
50	3DFR	Dihydrofolate Reductase
51	1HT8	Prostaglandin H2 Synthase-1
52	1ED5	Nitric Oxide Synthase
53	1XKK	Epidermal Growth Factor Receptor
54	1DIS	Dihydrofolate Reductase
55	1GIH	Cell Division Protein Kinase 2
56	1Y79	Dipeptidyl Carboxypeptidase
57	1TPL	Tyrosine Phenol-Lyase
58	2BKH	Reverse-Direction Myosin Motor (Myosin VI)
59	3MDE	Medium Chain Acyl-CoA Dehydrogenase
60	6COX	Cyclooxygenase-2
61	1DTL	Cardiac Troponin C
62	1PYY	Penicillin-Binding Protein 2X , Streptococcus Pneumoniae
63	1OOQ	Oxygen-Insensitive NAD(P)H Nitroreductase
64	1I7G	Peroxisome Proliferator Activated Receptor Alpha
65	1OIQ	Cell Division Protein Kinase 2
66	1P4M	Riboflavin Kinase

67	1HMP	Hypoxanthine Guanine Phosphoribosyl-Transferase
68	1W07	Acyl-CoA Oxidase
69	1V8B	S-Adenosyl-L-Homocysteine Hydrolase
70	1F8N	Lipoxygenase-1
71	1VFR	NAD(P)H:FMN Oxidoreductase
72	1P4G	Glycogen Phosphorylase
73	1HP7	Alpha-1-Antitrypsin
74	1OIR	Cell Division Protein Kinase 2
75	1SEZ	Protoporphyrinogen Oxidase, Mitochondrial
76	1R9O	Cytochrome P450 2C9
77	1DAR	Elongation Factor G
78	2A3L	Adenosine 5'-Monophosphate Deaminase
79	1HN4	Prophospholipase A2
80	1EFR	Mitochondrial F1-Atpase
81	1PS9	2,4-Dienoyl-CoA Reductase
82	1K6W	Cytosine Deaminase
83	1FM6	Retinoic Acid Receptor Rxr-Alpha
84	1XEB	Acyl-CoA N-Acyltransferase
85	2R07	Human Rhinovirus 14 Coat Protein
86	1DIY	Prostaglandin H2 Synthase-1
87	1APU	Aspartyl Proteinase Penicillopepsin
88	1KAE	Histidinol Dehydrogenase
89	1PPL	Penicillopepsin
90	1HRI	Human Rhinovirus 14 Coat Protein
91	1B5L	Interferon Tau
92	1PQ2	Cytochrome P450 2C8
93	1D6M	DNA Topoisomerase III
94	1Y0S	Peroxisome Proliferator Activated Receptor Delta
95	1VKG	Histone Deacetylase 8
96	1SZ2	Glucokinase
97	1OG5	Cytochrome P450 2C9
98	1Q3E	Potassium/Sodium Hyperpolarization-Activated Cyclic Nucleotide-Gated Channel 2
99	1APT	Aspartyl Proteinase Penicillopepsin
100	2BX8	Serum Albumin

7.3. $Gd@C_{82}$ scoreboard

Rank	PDB_ID	Name
1	1JVM	Voltage-Gated Potassium Channel
2	1ILH	Human Pregnane X Receptor
3	1TCO	Serine/Threonine Phosphatase B2
4	1REQ	Methylmalonyl-CoA Mutase
5	1ZZE	Aldehyde Reductase II
6	1J99	Alcohol Sulfotransferase
7	1DHJ	Dihydrofolate Reductase
8	1Y79	Peptidyl-Dipeptidase
9	1FT2	Protein Farnesyltransferase
10	4DFR	Dihydrofolate Reductase
11	1F3O	ATP-Binding Cassette
12	1RA2	Dihydrofolate Reductase
13	1I9C	Glutamate Mutase
14	1W07	Acyl-CoA Oxidase
15	1EVZ	Glycerol-3-Phosphate Dehydrogenase
16	1ME8	Inosine-5'-Monophosphate Dehydrogenase
17	2SIM	LT2 Neuraminidase, Salmonella Typhimurium
18	1SG0	Quinone Reductase 2
19	1DHT	Estrogenic 17-Beta Hydroxysteroid Dehydrogenase
20	1VJT	Alpha-Glucosidase
21	1SZ2	Glucokinase
22	2CG5	L-Aminoadipate-Semialdehyde Dehydrogenase-Phosphopantetheinyl Transferase
23	1GP6	Anthocyanidin Synthase
24	2ILK	Interleukin-10
25	1W6K	Oxidosqualene Cyclase (Lanosterol Synthase)
26	1ICP	12-Oxophytodienoate Reductase 1
27	1QU4	Ornithine Decarboxylase
28	1MO7	Sodium/Potassium-Transporting Atpase Alpha-1 Chain
29	1OS5	Hepatitis C Virus NS5B RNA Polymerase
30	1CLY	Br96 Fab
31	1OPM	Peptidylglycine Alpha-Hydroxylating Monooxygenase
32	1ED5	Nitric Oxide Synthase
33	1SEZ	Protoporphyrinogen Oxidase, Mitochondrial
34	1MCR	Immunoglobulin Lambda Dimer Mcg (Light Chain)
35	1GNX	Beta-Glucosidase
36	1CB7	Glutamate Mutase
37	1U4O	L-Lactate Dehydrogenase
38	1JCN	Inosine Monophosphate Dehydrogenase Type-I
39	1PGP	6-Phosphogluconate Dehydrogenase
40	1DID	D-Xylose Isomerase
41	1EFR	Mitochondrial F1-Atpase Subunit Alpha
42	1LGR	Glutamine Synthetase, Salmonella Typhimurium

43	1TYP	Trypanothione Reductase
44	1DIE	D-Xylose Isomerase
45	1JJC	Phenylalanyl-tRNA Synthetase Alpha Chain
46	2J0D	Cytochrome P450 3A4
47	1DMW	Phenylalanine Hydroxylase
48	1IFX	NH(3)-Dependent NAD(+) Synthetase
49	1AL8	Glycolate Oxidase
50	2AYO	Ubiquitin Carboxyl-Terminal Hydrolase 14
51	1K7Y	Methionine Synthase
52	1KAE	Histidinol Dehydrogenase
53	1C9C	Aspartate Aminotransferase
54	1TSD	Thymidylate Synthase
55	1WRR	Urate Oxidase
56	1SFT	Alanine Racemase
57	2LGS	Glutamine Synthetase
58	1PW2	Cell Division Protein Kinase 2
59	1VKG	Histone Deacetylase 8
60	1EWK	Metabotropic Glutamate Receptor Subtype 1
61	1D2V	Myeloperoxidase
62	1E7S	GDP-Fucose Synthetase
63	1QIJ	Acetylcholinesterase
64	1RT6	HIV-1 Reverse Transcriptase
65	1BVY	Cytochrome P450 BM-3
66	1OIR	Cell Division Protein Kinase 2
67	1H28	Cell Division Protein Kinase 2
68	1JH7	Cyclic Phosphodiesterase
69	1DIA	Methylenetetrahydrofolate Dehydrogenase/Cyclohydrolase
70	1HKV	Diaminopimelate Decarboxylase
71	1TVR	HIV-1 Reverse Transcriptase
72	1D2T	Acid Phosphatase
73	1H5U	Glycogen Phosphorylase
74	1F0I	Phospholipase D
75	1DGD	Dialkylglycine Decarboxylase
76	1Q0N	Hydroxymethyldihydropteridine Pyrophosphokinase, Escherichia Coli
77	1HHI	Human Class I MHC Molecule HLA-A2
78	1JKH	Mutant HIV-1 Reverse Transcriptase
79	1JND	Imaginal Disc Growth Factor-2
80	1GPB	Glycogen Phosphorylase B
81	2BCE	Cholesterol Esterase
82	1HVB	D-alanyl-D-alanine Carboxypeptidase/Transpeptidase, Streptomyces Sp. R61
83	1IYH	Hematopoietic Prostaglandin D Synthase
84	1I8D	Riboflavin Synthase
85	1NEK	Succinate Dehydrogenase Flavoprotein Subunit
86	1HS6	Leukotriene A-4 Hydrolase
87	1J32	Aspartate Aminotransferase
88	1NM8	Carnitine O-Acetyltransferase

89	1OIU	Cell Division Protein Kinase 2
90	1CNF	Nitrate Reductase
91	1USH	5'-Nucleotidase
92	1E6U	GDP-Fucose Synthetase
93	2AK3	Adenylate Kinase Isoenzyme-3
94	1HT8	Prostaglandin H2 Synthase-1
95	1DWD	Human Thrombin
96	2HHA	Dipeptidyl Peptidase-4
97	2F9Q	Cytochrome P450 2D6
98	1DJE	8-Amino-7-Oxonanoate Synthase
99	1PG5	Aspartate Carbamoyltransferase
100	1JYS	MTA/SAH Nucleosidase

8. Appendix B: code for nanoparticle reverse screening

8.1. Database-maker Bash script

```bash
#!/bin/bash

# WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
#
# Users must have recent versions of GNU AWK, sed, grep.
# It is assumed that the following auxilliary functions are stored in a sub-directory of home
# called 'utils': i) aesthetics.sh ii) error_functions.sh iii) pdb_editing.sh
#
# WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING


# SPECIFICATION OF THE ARGUMENTS PROVIDED TO THE SCRIPT ######################################

input=$1
action=$2


# SPECIFICATION OF WHICH PARTS OF THE SCRIPTS TO RUN #########################################

bestPDBsearch=n
#####
retrival=n


# KEY VARIABLES FOR THE SCRIPT ###############################################################

# KEY DIRECTORIES
home=$PWD

# KEY VARIABLES
# Debugging
launch_time=$( date '+%y%m%d_%H%M%S' )
error_suffix=$( basename $0 | sed 's/\.sh//')

# KEY PATHs and LOADING OF UTILS FUNCTIONS
. /home/$USER/utils_byBobo/pdb_editing.sh
. /home/$USER/utils_byBobo/aesthetics.sh


# BEGIN ######################################################################################

if [ $# -lt 1 ]; then echo '
input=$1
action=$2=DEL
'
fi

echo -ne "\n°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°\n"
if [ ! $# -lt 2 ]; then echo -ne " Starting at: $launch_time \n\n"; fi
echo -ne " PARTS THAT WILL BE RUN:\n"
echo -ne " bestPDB_identification=$bestPDBsearch structure_retrival=$retrival\n"
echo -ne "°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°\n\n"

if [ $# -lt 1 ]; then exit; fi


if [ $bestPDBsearch = y ]; then gene_list=$input
else proteins=$input
fi


##############################################################################################
# Identification of the best PDB structure available for each protein of the list           #
if [ $bestPDBsearch = y ]; then                                                             #
```

```
if [ -e $home/"$gene_list"_bestPDBs ]; then rm $home/"$gene_list"_best_res_PDBs; fi
genes_n=$(wc -l < $gene_list)
for ((i=1 ; i<=$genes_n ; i++))
do
    gene=$( head -$i $gene_list | tail -1 | awk '{print $1}' )
    percentage_output $i $genes_n $gene

    declare -A p_PDBs_res
    p_PDBs=( $( head -$i $gene_list | tail -1 | awk '{print $2}' | sed 's/;/ /g' ) )
    for j in ${p_PDBs[@]}
    do
        let exe_lineno=$LINENO+1
        wget "https://www.rcsb.org/structure/$j" > stdout 2> stderr
        __generic_output $j $j $home $exe_lineno $launch_time $error_suffix
        __del_stderr_stdout

        res=$(grep -o --max-count=1 '<strong>Resolution: <\/strong>.\...' $j | \
            head -1 | sed 's#<strong>Resolution: </strong>##')
        p_PDBs_res[$j]=$res
        rm $home/$j
    done
    best_p_PDB=$( for k in ${!p_PDBs_res[@]}
    do
        echo "$k ${p_PDBs_res[$k]}"
    done | sort -n -k2 | head -1 | awk '{print $1}' )
    echo "$gene     $best_p_PDB" >> $home/"$gene_list"_best_res
    unset p_PDBs_res
done
awk '{print $2}' $home/"$gene_list"_best_res | sort | uniq > $home/zdock_protein_list

fi
#############################################################################################
# Download of the best PDB structures identified at the previous step                       #
if [ $retrival = y ];then                                                                   #

if [ $bestPDBsearch = y ]; then mapfile -t proteins_arr < $home/zdock_protein_list
else mapfile -t proteins_arr < $proteins
fi

for i in ${proteins_arr[@]}
do
    for k in ${!proteins_arr[@]};do if [ ${proteins_arr[$k]} = $j ];then index=$k;fi;done
    percentage_output $index ${#proteins_arr[@]} $j

    p_dir=$home/$i

    if [ ! -z $action ] && [ $action = DEL ]; then
        if [ -d $p_dir ]; then rm -r $p_dir; fi
    else
        if [ ! -d $p_dir ]; then mkdir $p_dir; fi
        cd $p_diri

        if [ ! -s $p_dir/"$i"_og_dirty.pdb ]; then

            let exe_lineno=$LINENO+1
            wget https://files.rcsb.org/download/$i.pdb \
                                > $p_dir/stdout 2> $p_dir/stderr
            __generic_output $p_dir/$j.pdb $j $home $exe_lineno \
                                $launch_time $error_suffix
            __del_stderr_stdout

            let exe_lineno=$LINENO+1
            wget https://www.rcsb.org/structure/$i   \
                                > $p_dir/stdout 2> $p_dir/stderr
            __generic_output $p_dir/$j $j $home $exe_lineno \
                                $launch_time $error_suffix
            __del_stderr_stdout

            mv $p_dir/$i.pdb $p_dir/"$i"_og_dirty.pdb
            mv $p_dir/$i "$i"_page

        fi

        if [ ! -s $p_dir/"$i"_noH_noEm.pdb ]; then

            let exe_lineno=$LINENO+1
            nmr=$(grep -o -m 1 'SOLUTION NMR' $p_dir/"$i"_page | head -1)
            if [ ! -z "$nmr" ]; then
                start='^MODEL[[:space:]]*\<1\>'
                end='^MODEL[[:space:]]*\<2\>'
                sed -n '/'"$start"'/,/'"$end"'/{//!p;}' \
                                        $p_dir/"$i"_og_dirty.pdb |\
                    grep '^ATOM[[:space:]]*[[:digit:]]' |\
```

```
                    grep -v '     H' > $p_dir/"$i"_kinda_clean.pdb
            else
                grep '^ATOM[[:space:]]*[[:digit:]]' \
                                        $p_dir/"$i"_og_dirty.pdb |\
                    grep -v '     H' \
                    > $p_dir/"$i"_kinda_clean.pdb
            fi
            __generic_output $p_dir/"$i"_kinda_clean.pdb $j $home $exe_lineno \
                             $launch_time $error_suffix

            if [ ! -d $p_dir/og_toSave ]; then mkdir $p_dir/og_toSave; fi

            mv $p_dir/"$i"_kinda_clean.pdb $p_dir/og_toSave/"$i"_noH_noEm.pdb
            __extractFF14SBprotRes $p_dir/og_toSave/"$j"_noH_noEm.pdb
            __extractHeavyAtoms $p_dir/og_toSave/"$j"_noH_noEm.pdb
            __removeExtraHeavyAtoms $p_dir/og_toSave/"$j"_noH_noEm.pdb
            __repairFckupResNames $p_dir/og_toSave/"$j"_noH_noEm.pdb
            __repairFckupAtomNames $p_dir/og_toSave/"$j"_noH_noEm.pdb
            __insertTER $p_dir/og_toSave/"$j"_noH_noEm.pdb

        fi
    fi
done

fi
#############################################################################################


cd $home
echo -ne " Database creation done at: $(date '+%y%m%d_%H%M%S')\n\n"
```

8.2. Pre-screening Bash script

```
#!/bin/bash

# WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
#
# Users must have a GROMACS 5.1 and AmberTools16 installations in their paths
# and recent versions of GNU AWK, sed, grep.
# Each protein entry of the database must be located in a separate sub-directory
# of the working directory (i.e., the directory where this script is launched), which
# must be named as the PDB id of the protein. Each sub-directory must contain a .pdb file
# of the protein which will be 'cleaned' in the firs step of the script. IF the USER has
# created the database using the script 'database-maker.sh', the cleaning step MUST be
# skipped. It is assumed that the following auxilliary functions are stored in a sub-directory
# of home called 'utils': i) aesthetics.sh ii) error_functions.sh iii) pdb_editing.sh
#
# WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING


# SPECIFICATION OF THE ARGUMENTS PROVIDED TO THE SCRIPT ###################################

proteins=$1
n_proc=$2

if [ -d $PWD/$proteins ]; then proteins_arr=( $proteins )
else mapfile -t proteins_arr < $PWD/$proteins
fi

if [ -z $n_proc ]; then n_proc=2; fi


# SPECIFICATION OF WHICH PARTS OF THE SCRIPTS TO RUN #######################################

tooBig=n
#####
clean_p=n
#####
use_Em_structures=n
#####
tleap=n
#####
Em=n
#####
conversion_to_gmx=n
#####
```

```
tpr_ndx=n
#####
surf=n
#####
pbsa=n

if [ $use_Em_structures = n ] && [ $Em = y ]; then
    echo "°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°"
    echo -ne " ABORTING: you are asking to minimize the system and \n "
    echo -ne " to NOT use the minimized structures in the analysis. \n "
    echo -ne " It doesn't make sense. \n\n"
    exit 1
fi


# KEY VARIABLES FOR THE SCRIPT ###############################################################

# KEY DIRECTORIES
home=$PWD
# Folder with everything we need for g_mmpbsa scoring
mmpbsa_files_dir=$home/mmpbsa_files

# KEY VARIABLES
# Solute dielectric constant (g_mmpbsa only)
pdie=$(grep -m 1 '^pdie' $mmpbsa_files_dir/pbsa.mdp | awk '{print $3}')
# Debugging
launch_time=$( date '+%y%m%d_%H%M%S' )
error_suffix=$( basename $0 | sed 's/\.sh//')

# KEY PATHs and LOADING OF UTILS FUNCTIONS
. /home/$USER/utils/aesthetics.sh
. /home/$USER/utils/error_functions.sh
. /home/$USER/utils/pdb_editing.sh


# BEGIN #####################################################################################

if [ $# -lt 1 ]; then echo '
proteins=$1
n_proc=$2
'
fi

echo -ne "\n°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°\n"
if [ ! $# -lt 2 ]; then echo -ne " Starting at: $launch_time \n\n"; fi
echo -ne " PARTS THAT WILL BE RUN:\n"
echo -ne " clean_p=$clean_p use_Em_structures=$use_Em_structures do_min=$Em \n"
echo -ne " convert_coord_params_to_gmx=$conversion_to_gmx make_tpr_ndx_files=$tpr_ndx \n"
echo -ne " build_surface=$surf do_pbsa=$pbsa \n"
echo -ne " \n PARAMETERS USED:\n"
echo -ne " pdie=$pdie\n"
echo -ne "°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°\n\n"

if [ $# -lt 1 ]; then exit; fi


for j in ${proteins_arr[@]}
do
    # Console output on the progression of the loop
    for k in ${!proteins_arr[@]}; do if [ ${proteins_arr[$k]} = $j ]; then index=$k; fi; done
    percentage_output $index ${#proteins_arr[@]} $j

    # Declaration of the key folders for the protein
    # 1)
    p_dir=$home/$j

    # 2)
    if [ $use_Em_structures = y ]; then calc_dir=$p_dir/dock_onEm
        else calc_dir=$p_dir/dock; fi
    if [ ! -d $calc_dir ]; then mkdir $calc_dir ; fi
    cd $p_dir


##############################################################################################
# Cleaning the protein structural file                                                       #
if [ $clean_p = y ]; then                                                                    #

    if [ ! -d $p_dir/og_toSave ]; then mkdir $p_dir/og_toSave; fi

    mv $p_dir/$j.pdb $p_dir/og_toSave/"$j"_noH_noEm.pdb
    __extractFF14SBprotRes $p_dir/og_toSave/"$j"_noH_noEm.pdb
    __extractHeavyAtoms $p_dir/og_toSave/"$j"_noH_noEm.pdb
    __removeExtraHeavyAtoms $p_dir/og_toSave/"$j"_noH_noEm.pdb
```

```
    __repairFckupResNames $p_dir/og_toSave/"$j"_noH_noEm.pdb
    __repairFckupAtomNames $p_dir/og_toSave/"$j"_noH_noEm.pdb
    __insertTER $p_dir/og_toSave/"$j"_noH_noEm.pdb

fi
#############################################################################################
# Creating .prmtop and .inpcrd for the PDB structure as it is without energy minimization   #
# in order to fill in missing residues                                                      #
if [ $tleap = y ]; then

    # Using tleap in AmberTools to fill in missing heavy atoms (otherwise make gmx will fail)
    tleap_pdb_input="$j"_noH_noEm_wTER.pdb
    tleap_inpcrd_output="$j"_noEm.inpcrd
    cat > $p_dir/tleap.in <<- EOF
        source leaprc.protein.ff14SB
        in = loadpdb $p_dir/og_toSave/$tleap_pdb_input
        set default PBRadii bondi
        saveamberparm in $p_dir/$j.prmtop $p_dir/$tleap_inpcrd_output
        quit
EOF
    let exe_lineno=$LINENO+1
    tleap -f $p_dir/tleap.in > $p_dir/stdout 2> $p_dir/stderr
    __generic_output $p_dir/$j.prmtop $j $home $exe_lineno $launch_time $error_suffix 2
    __del_stderr_stdout

fi
#############################################################################################
# Moving to the folder where the files needed for the docking procedure will be stored.     #
    cd $calc_dir

#############################################################################################
# Energy minimization of the protein structure before surface creation for docking          #
# and PBSA calculations (optional, yet recommended if reverse screening of small ligands).  #
if [ $Em = y ]; then

    # Creation of the input file
    cat > $calc_dir/min.in <<- EOF
        Energy minimization with large cut-off to limit approximations
         &cntrl
          imin   = 1,
          maxcyc = 500,
          ncyc   = 250,
          ntb    = 0,
          igb    = 7,
          cut    = 20
         /
EOF

    # Energy minimization of the protein structure using sander from AmberTools
    SECONDS=0
    let exe_lineno=$LINENO+1
    if [ $n_proc -gt 1 ]; then
        mpirun -n $n_proc sander.MPI -O \
            -i $calc_dir/min.in -o $calc_dir/"$j"_min.out \
            -p $p_dir/$j.prmtop -c $p_dir/"$j"_noEm.inpcrd \
                        -r $calc_dir/$j.ncrst \
            > $calc_dir/stdout 2> $calc_dir/stderr
    else
        sander -O \
            -i $calc_dir/min.in -o $calc_dir/"$j"_min.out \
            -p $p_dir/$j.prmtop -c $p_dir/"$j"_noEm.inpcrd \
                        -r $calc_dir/$j.ncrst \
            > $calc_dir/stdout 2> $calc_dir/stderr
    fi

    __generic_output $calc_dir/$j.ncrst $j $home $exe_lineno $launch_time $error_suffix 2
    __del_stderr_stdout

    echo "$j Energy minimization of protein structure  on $n_proc cores completed in
$(($SECONDS/60)) m $(($SECONDS%60)) s" \
        > $calc_dir/perf_report_minimization

fi
#############################################################################################
# Conversion to GMX files via ACPYPE                                                        #
if [ $conversion_to_gmx = y ]; then                                                         #

    if [ $use_Em_structures = y ]; then
        let exe_lineno=$LINENO+1
        acpype.py -p $p_dir/$j.prmtop  -x $calc_dir/$j.ncrst \
            > $calc_dir/stdout 2> $calc_dir/stderr
    else
        let exe_lineno=$LINENO+1
```

```
        acpype.py -p $p_dir/$j.prmtop  -x $p_dir/"$j"_noEm.inpcrd \
            > $calc_dir/stdout 2> $calc_dir/stderr
    fi

    __generic_output $calc_dir/"$j"_GMX.top $j $home $exe_lineno $launch_time $error_suffix 2
    __del_stderr_stdout

    mv $calc_dir/"$j"_GMX.gro $calc_dir/$j.gro
    mv $calc_dir/"$j"_GMX.top $calc_dir/$j.top

    # Removing H atoms for subsequent surface creation and for docking
    if [ $use_Em_structures = y ]; then
        ambpdb -p $p_dir/$j.prmtop -c $calc_dir/$j.ncrst > $calc_dir/$j.pdb
    else
        ambpdb -p $p_dir/$j.prmtop -c $p_dir/"$j"_noEm.inpcrd > $calc_dir/$j.pdb
    fi
    egrep -v '^ATOM[[:blank:]]*[[:digit:]]*[[:blank:]]*H|^END' $calc_dir/$j.pdb \
            > $calc_dir/"$j"_noH.pdb
    egrep -v 'TER'  $calc_dir/$j.pdb \
            > $calc_dir/"$j"_noTER.pdb

fi
#############################################################################################
####
# Creation of the additional gmx files which will be necessary in the PBSA calculations
#
if [ $tpr_ndx = y ]; then
#
    cat > $calc_dir/em.mdp <<-EOF
        integrator  = steep
        emtol       = 100.0
        emstep      = 0.01
        nsteps      = 50000

        cutoff-scheme   = Verlet
        ns_type     = grid
        nstlist     = 1

        rlist       = 2.0
        rvdw    = 2.0
        DispCorr    = Ener
        rcoulomb    = 2.0
        coulombtype = cut-off

        pbc     = xyz
EOF

    let exe_lineno=$LINENO+1
    gmx51 grompp -f $calc_dir/em.mdp -c $calc_dir/$j.gro \
        -p $calc_dir/$j.top -po $calc_dir/"$j"_tpr_mdout -o $calc_dir/$j.tpr \
        > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 2
    __del_stderr_stdout

    let exe_lineno=$LINENO+1
    echo q | gmx51 make_ndx -f $calc_dir/$j.gro -o $calc_dir/$j.ndx \
        > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 2
    __del_stderr_stdout

fi
#############################################################################################
# Marking the surface of the protein                                                        #
if [ $surf = y ]; then                                                                      #

    # You need to be in the directory where mark_sur is located in order to launch it
    # and inside that dir you also need to have the uniCHARMM file

    if [ ! -s $calc_dir/save/"$j"_noH_sur.pdb ]; then

    cd $home
    let exe_lineno=$LINENO+1
    ./mark_sur $calc_dir/"$j"_noH.pdb $calc_dir/"$j"_noH_sur.pdb \
            > $calc_dir/stdout 2> $calc_dir/stderr
    __generic_stderr $PWD $j $home $exe_lineno \
        $launch_time $error_suffix 2
    __generic_output $calc_dir/"$j"_noH_sur.pdb $j $home $exe_lineno \
        $launch_time $error_suffix 2
    __del_stderr_stdout

    fi
    cd $calc_dir
```

```
fi
#############################################################################################################
# PBSA analysis with g_mmpbsa                                                                               #
if [ $pbsa = y ]; then                                                                                      #

    SECONDS=0
    export OMP_NUM_THREADS=$n_proc
    let exe_lineno=$LINENO+1
    echo 0 | g_mmpbsa -f $calc_dir/$j.gro -s $calc_dir/$j.tpr \
        -n $calc_dir/$j.ndx -i $mmpbsa_files_dir/pbsa.mdp \
        -nodiff -nomme -pbsa \
        -pol $calc_dir/"$j"_polar.xvg \
        -apol $calc_dir/"$j"_apolar.xvg \
        > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 2
    __del_stderr_stdout

    echo "$j Solvation energy computation on $n_proc cores completed in $(($SECONDS/60)) m
$(($SECONDS%60)) s" \
        > $calc_dir/perf_report_solvation

fi
#############################################################################################################
####

done

cd $home
echo -ne " Pre-screening preparations done at: $(date '+%y%m%d_%H%M%S')\n Let's start docking!
\n\n"
```

8.3. Screening Bash script

```
#!/bin/bash

# WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING
#
# Users must have a GROMACS 5.1 and AmberTools16 installations in their paths
# and recent versions of GNU AWK, sed, grep.
# Each protein entry of the database must be located in a separate sub-directory
# of the working directory (i.e., the directory where this script is launched), which
# must be named as the PDB id of the protein.
# WARNING Before running this script, the USER must run 'pre-screening_amberSander.sh' to
properly prepare the protein's sub-directory structure and necessary files.
# Each nanoparticle must have its own sub-directory of the working directory and the USER must
# prepare the following files for each nanoparticle manually:
#  > np.gro (GROMACS coords file of NP with 3-digit precision);
#  > np_sur.pdb
#   (NP surface for ZDOCK2.1 generated with 'mark_sur', a binary provided with ZDOCK 2.1)
#  > forcefield.itp
#   (material params compatible with AMBER macromolecular forcefields, in GROMACS format)
#  > np.itp
#   (GROMACS "include topology" file containing atomnames, atomtypes and connectivity
#    of the NP, generated using the GROMACS tool x2top)
#  > np_polar.xvg + np_apolar.xvg
#   (optional, polar and apolar solvation energy of the unbound nanoparticle
#        for MM-PBSA scoring)
# The first and last 2 files must be stored in the nanoparticle sub-directory, the remaining 2
# the directory of the forcefield that will be used to describe the nanoparticle during the
# screening phase. The latter must be placed as sub-directory of the nanoparticle
# sub-directory. The name must be defined by the user by editing the value of the variable
# $mat_ff under '# KEY VARIABLES' at the beginning of the script.
# It is assumed that the following auxilliary functions are stored in home/$USER/utils:
# i) aesthetics.sh ii) error_functions.sh iii) pdb_editing.sh ; and that ZDOCK2.1
# sub-directory is located in the standard location for pre-compiled binaries,
# i.e.,home/$USER/opt
#
# WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING WARNING


# SPECIFICATION OF THE ARGUMENTS PROVIDED TO THE SCRIPT ###################################

proteins=$1
np=$2
```

```
n_proc=$3

if [ -d $PWD/$proteins ]; then proteins_arr=( $proteins )
else mapfile -t proteins_arr < $PWD/$proteins
fi

if [ -z $n_proc ]; then n_proc=2; fi


# SPECIFICATION OF WHICH PARTS OF THE SCRIPTS TO RUN #########################################

use_Em_structures=n
#####
docking=n
#####
com_gmx_files=n
    com_top=y
    com_gro=y
    com_tpr=y
#####
printing=n
#####
aligning=n
#####
clustering=n
#####
extraction=n
#####
clash_removal=n
#####
scoring=n
    fast_gb=y
        file_gen=y


# KEY VARIABLES FOR THE SCRIPT ###############################################################

# KEY DIRECTORIES
home=$PWD
# Folder with everything we need for g_mmpbsa scoring
mmpbsa_files_dir=$home/mmpbsa_files
# Nanoparticle/material folder
np_dir=$home/$np

# KEY VARIABLES
# Poses to print and process
n_poses=200
# Forcefield to describe the nanoparticle
# (must be equal to the name of the sub-dir in the np dir that constains the relative params)
mat_ff=gaff
# Jarvis clustering parameters
jarvis_m=10
jarvis_p=3
cluster_cutoff=1
# Van der Waals radius for unrecognized atoms and solute dielectric constant (g_mmpbsa only)
rvdw=0.191
pdie=$(grep -m 1 '^pdie' $mmpbsa_files_dir/pbsa.mdp | awk '{print $3}')
# Debugging
launch_time=$( date '+%y%m%d_%H%M%S' )
error_suffix=$( basename -s .sh $0 )

# KEY PATHs and LOADING OF UTILS FUNCTIONS
export PATH=${PATH}:/home/$USER/opt/zdock2.1_mod
. /home/$USER/utils/aesthetics.sh
. /home/$USER/utils/error_functions.sh
. /home/$USER/utils/pdb_editing.sh


# BEGIN ######################################################################################

if [ $# -lt 2 ]; then echo '
proteins=$1
np=$2
n_proc=$3
'
fi

echo -ne "\n°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°°\n"
if [ ! $# -lt 2 ]; then echo -ne " Starting at: $launch_time \n\n"; fi
echo -ne " PARTS THAT WILL BE RUN:\n"
echo -ne " docking=$docking\n"
echo -ne " make_complex=$com_gmx_files ( top=$com_top gro=$com_gro tpr=$com_tpr )\n"
echo -ne " printing=$printing aligning=$aligning clustering=$clustering\n"
```

```
echo -ne " extraction=$extraction clash_removal=$clash_removal\n"
echo -ne " scoring=$scoring (fast_gb=$fast_gb, file_gen=$file_gen)\n"
echo -ne " \n PARAMETERS USED:\n"
echo -ne " mat_ff=$mat_ff\n n_poses=$n_poses\n"
echo -ne " jarvis_m=$jarvis_m jarvis_p=$jarvis_p cluster_cutoff=$cluster_cutoff\n"
echo -ne " rvdw=$rvdw pdie=$pdie\n"
echo -ne "^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^\n\n"

if [ $# -lt 2 ]; then exit; fi


echo -ne " Docking of $np \n\n"
# Number of atoms of the structure (useful later on)
np_atoms=$(( $(wc -l < $np_dir/np.gro)-3 ))

# Polar and apolar solvation components for later scoring of the complexes
np_PB=$(tail -1 $np_dir/np_polar.xvg| awk '{print $2}')
np_sasa=$(tail -1 $np_dir/np_apolar.xvg| awk '{print $2}')
np_sav=$(tail -1 $np_dir/np_apolar.xvg| awk '{print $3}')
np_wca=$(tail -1 $np_dir/np_apolar.xvg| awk '{print $4}')


# Beginning of the iteration of the list of proteins
for j in ${proteins_arr[@]}
do
    # Console output on the progression of the loop
    for k in ${!proteins_arr[@]}; do if [ ${proteins_arr[$k]} = $j ]; then index=$k; fi; done
    percentage_output $index ${#proteins_arr[@]} $j

    # Declaration of the key folders for the protein
    # 1)
    p_dir=$home/$j
    if [ $use_Em_structures = y ]; then calc_dir=$p_dir/dock_onEm
        else calc_dir=$p_dir/dock; fi
    cd $calc_dir
    # Number of atoms ot the protein and of the resulting complex (useful later on).
    # I substract 3 lines from the count because they do not contain atom coords.
    # Above, I didn't do it because the NP .gro is created by hand and it doen't have
    # those lines.
    p_atoms=$(( $(wc -l < $calc_dir/$j.gro)-3 ))
    com_atoms=$(( $p_atoms+$np_atoms ))

    # Polar and apolar solvation components for later scoring of the complexes
    p_PB=$(tail -1 $calc_dir/"$j"_polar.xvg| awk '{print $2}')
    p_sasa=$(tail -1 $calc_dir/"$j"_apolar.xvg| awk '{print $2}')
    p_sav=$(tail -1 $calc_dir/"$j"_apolar.xvg| awk '{print $3}')
    p_wca=$(tail -1 $calc_dir/"$j"_apolar.xvg| awk '{print $4}')


##############################################################################################
# Shape complementarity docking process                                                      #
if [ $docking = y ]; then                                                                    #


    SECONDS=0
    # For whatever reason, if the path of the files is too long the executable
        # won't be able to find a so-called file 'b.all' (which is nowhere to be found).
    # Probabily the executable doesn't allocate enough memory for a full path...
    # so we have to use this stupid relative paths here.
    let exe_lineno=$LINENO+1
    zdock -o $np.out -R ../../$np/np_sur.pdb -L save/"$j"_noH_sur.pdb -N $n_poses \
            > $calc_dir/stdout 2> $calc_dir/stderr
    __generic_output $calc_dir/$np.out $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout

    echo "$j 1 Docking completed in $(($SECONDS/60)) m $(($SECONDS%60)) s" \
        > $calc_dir/perf_report_"$np"_docking

    # Saving the time consuming docking results in the save folder made
    # with the pre-screening-prot script
    let exe_lineno=$LINENO+1
    __generic_output $calc_dir/save $j $home $exe_lineno $launch_time $error_suffix 1
    chmod 777 $calc_dir/save/
    cp $calc_dir/$np.out $calc_dir/save/$np.out
    chmod 555 $calc_dir/save/

fi
##############################################################################################
# Generation of proper GROMACS structural, topology and system files using Bash tools        #
if [ $com_gmx_files = y ]; then                                                              #

if [ $com_gro = y ]; then #-----------------------------------------------------------------
```

```
    # Extraction of the protein coordinates of the 1st docking candidate in .pdb format
    let exe_lineno=$LINENO+1
    createX.pl $calc_dir/$np.out 1 1 > $calc_dir/stdout 2> $calc_dir/stderr
    __generic_stderr $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout

    # Conversion of the latter in .gro format
    let exe_lineno=$LINENO+1
    gmx51 editconf -f $calc_dir/"$np"_1_p.pdb -o $calc_dir/"$np"_1_p.gro \
            > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout

    # Construction of the coordinates file (.gro) of the complex
    let exe_lineno=$LINENO+1
    cat <( echo 'Gradirei Ravioli Oppure Maccheroni Alla Carbonara Stasera!' ) \
            <( echo $com_atoms ) \
            <( head -n -1 $calc_dir/"$np"_1_p.gro | tail -n +3 ) \
            <( head -n -1 $np_dir/np.gro | tail -n +3 ) \
            $np_dir/box_size.txt \
            > $calc_dir/"$np"_1_com.gro
    com_gro_len=$(wc -l < $calc_dir/"$np"_1_com.gro)
    if [ $com_gro_len -eq $(($com_atoms+3)) ]; then echo 'OK' > $calc_dir/check ; fi
    __generic_output $calc_dir/check $j $home $exe_lineno $launch_time $error_suffix 1
    __del_check

fi #------------------------------------------------------------------------------------------

if [ $com_top = y ]; then #-------------------------------------------------------------------

    # Construction of the parameters file (.top)
    let exe_lineno=$LINENO+1
    upper_limit=$(egrep -o -m 1 -n '\[ atomtypes \]' $calc_dir/$j.top \
        | gawk -F: '{print $1}')
    lower_limit=$(egrep -o -m 1 -n '\[ system \]' $calc_dir/$j.top \
        | gawk -F: '{print $1}')
    let corrected_lower_limit=$lower_limit-$upper_limit

    cat <(echo "include \"$np_dir/$mat_ff.ff/forcefield.itp\"") \
            <(echo "include \"$np_dir/$mat_ff.ff/sol_ions_ffnonbonded.itp\"") \
            <(tail -n +$upper_limit $calc_dir/$j.top | head -n $corrected_lower_limit) \
            <(echo "include \"$np_dir/$mat_ff.ff/np.itp\"") \
            <(echo "include \"$np_dir/$mat_ff.ff/tip3p.itp\"") \
            <(echo "include \"$np_dir/$mat_ff.ff/ions.itp\"") \
            > $calc_dir/"$np"_com.top
    cat <<- EOF >> $calc_dir/"$np"_com.top
    [ system ]
    ; Name
    $j on $np

    [ molecules ]
    ; Compound           #mols
    $j                   1
EOF
    cat $np_dir/molecules_section.txt >> $calc_dir/"$np"_com.top
    prot_itp_len=$(tail -n +$upper_limit $calc_dir/$j.top | \
            head -n $corrected_lower_limit | wc -l)
    com_top_len=$(wc -l < $calc_dir/"$np"_com.top)
    if [ $com_top_len -gt $prot_itp_len ]; then echo 'OK' > $calc_dir/check ; fi
    __generic_output $calc_dir/check $j $home $exe_lineno $launch_time $error_suffix 1
    __del_check

fi #------------------------------------------------------------------------------------------

if [ $com_tpr = y ]; then #-------------------------------------------------------------------

    # Construction of the complex index file (.ndx)
    let exe_lineno=$LINENO+1
    gmx51 make_ndx -f $calc_dir/"$np"_1_com.gro -o $calc_dir/"$np"_com.ndx \
            < $np_dir/make_ndx.txt \
            > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout

    # Construction of the complex .tpr file for clustering and g_mmpbsa computations
    cat > $calc_dir/em.mdp <<- EOF
        ; RUN CONTROL
        integrator      = steep
        nsteps          = 10000
        emtol           = 10

        ; CONSTRAINTS
        constraints     = none
```

```
        constraint-algorithm    = lincs
        lincs-iter      = 4

        ; NS SEARCH AND NON-BONDED INTERACTIONS
        nstlist             = 0
        cutoff-scheme       = group
        ns_type             = simple
        rlist           = 0
        coulombtype     = cutoff
        rcoulomb        = 0
        vdwtype         = cutoff
        rvdw        = 0
        pbc         = no
        DispCorr        = no
EOF
    let exe_lineno=$LINENO+1
    gmx51 grompp -f $calc_dir/em.mdp -c $calc_dir/"$np"_1_com.gro \
        -p $calc_dir/"$np"_com.top -n $calc_dir/"$np"_com.ndx \
        -po $calc_dir/"$np"_com_tpr_mdout -o $calc_dir/"$np"_com.tpr \
        > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout
    rm $calc_dir/"$np"_1_com.gro

fi #-------------------------------------------------------------------------------------------

fi
#############################################################################################
# Printing the protein coordinates ALONE of the docking candidates in .pdb format           #
if [ $printing = y ]; then                                                                  #

    let exe_lineno=$LINENO+1
    createX.pl $calc_dir/$np.out 2 $n_poses > $calc_dir/stdout 2> $calc_dir/stderr
    __generic_stderr $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout

fi
#############################################################################################
# Aligning the protein coordinates of the docking candidates by translation+rotation        #
# on the xy plane. This results in more accurate clustering later on and doesn't change the #
# orientation of the protein structures on the 2D material surface since it extends         #
# on the xy plane                                                                           #
if [ $aligning = y ]; then                                                                  #

    # Creation of the reference protein coordinates (.tpr) for the rmsd fitting
        # in the xy plane. There should be already a protein .tpr file (for the calculation
        # of the protein solvation energy) but it doesn't contain the right coordinates
        # for the rms fitting.
    if [ ! -s $calc_dir/"$np"_1_p.gro ]; then
        # Extraction of the protein coords of 1st docking candidate in .pdb format
        let exe_lineno=$LINENO+1
        createX.pl $calc_dir/$np.out 1 1 > $calc_dir/stdout 2> $calc_dir/stderr
        __generic_stderr $PWD $j $home $exe_lineno $launch_time $error_suffix 1
        __del_stderr_stdout

        # Conversion of the latter in .gro format
        let exe_lineno=$LINENO+1
        gmx51 editconf -f $calc_dir/"$np"_1_p.pdb -o $calc_dir/"$np"_1_p.gro \
                > $calc_dir/stdout 2> $calc_dir/stderr
        __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 1
        __del_stderr_stdout
    fi

    # Mock .mdp file for grompp
    cat > $calc_dir/em.mdp <<- EOF
        ; RUN CONTROL
        integrator      = steep
        nsteps          = 10000
        emtol           = 10

        ; CONSTRAINTS
        constraints     = none
        constraint-algorithm    = lincs
        lincs-iter      = 4

        ; NS SEARCH AND NON-BONDED INTERACTIONS
        nstlist             = 0
        cutoff-scheme       = group
        ns_type             = simple
        rlist           = 0
        coulombtype     = cutoff
        rcoulomb        = 0
        vdwtype         = cutoff
```

```
        rvdw        = 0
        pbc         = no
        DispCorr        = no
EOF

    # Creation of the reference protein coords in .tpr format
    let exe_lineno=$LINENO+1
    gmx51 grompp -f $calc_dir/em.mdp -c $calc_dir/"$np"_1_p.gro \
        -p $calc_dir/$j.top -n $calc_dir/$j.ndx \
        -po $calc_dir/"$np"_p_ref_tpr_mdout -o $calc_dir/"$np"_p_ref.tpr \
        > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout
    rm $calc_dir/"$np"_1_p.gro

    # For each of those docking candidates ...
    for ((k=1 ; k<=$n_poses ; k++))
    do

        # ... conversion from .pdb to .gro format ...
        let exe_lineno=$LINENO+1
        gmx51 editconf -f $calc_dir/"$np"_"$k"_p.pdb -o $calc_dir/"$np"_"$k"_p.gro \
                > $calc_dir/stdout 2> $calc_dir/stderr
        __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 2
        __del_stderr_stdout
        rm $calc_dir/"$np"_"$k"_p.pdb

        # ... rmsd fitting in the xy plane ...
        let exe_lineno=$LINENO+1
        echo 0 0 | gmx51 trjconv -f $calc_dir/"$np"_"$k"_p.gro \
                -s $calc_dir/"$np"_p_ref.tpr -fit rotxy+transxy \
                -o $calc_dir/"$np"_"$k"_p_fit.gro \
                > $calc_dir/stdout 2> $calc_dir/stderr
        __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 2
        __del_stderr_stdout
        rm $calc_dir/"$np"_"$k"_p.gro

        # ... construction of the .gro file of the complex and
                # addition to an aligned .gro trj
        let exe_lineno=$LINENO+1
        cat <( echo "Gradirei Ravioli Oppure Maccheroni Alla Carbonara Stasera! t= $k.0" ) \
                <( echo $com_atoms ) \
                <( head -n -1 $calc_dir/"$np"_"$k"_p_fit.gro | tail -n +3 ) \
                <( head -n -1 $np_dir/np.gro | tail -n +3 ) \
                $np_dir/box_size.txt \
                > $calc_dir/"$np"_"$k"_com_fit.gro
        com_gro_len=$(wc -l < $calc_dir/"$np"_"$k"_com_fit.gro)
        if [ $com_gro_len -eq $(($com_atoms+3)) ]; then echo 'OK' > $calc_dir/check ; fi
        __generic_output $calc_dir/check $j $home $exe_lineno $launch_time $error_suffix 2
        __del_check
        cat $calc_dir/"$np"_"$k"_com_fit.gro >> $calc_dir/"$np"_coms_fit.gro
        rm $calc_dir/"$np"_"$k"_p_fit.gro $calc_dir/"$np"_"$k"_com_fit.gro
    done
fi
#############################################################################################################
# Clustering of the docking candidates after alignment of the protein structure to identify  #
# the most representing structure for each binding mode and reduce computational costs       #
if [ $clustering = y ]; then                                                                 #

    SECONDS=0
    let exe_lineno=$LINENO+1
    echo 2 | gmx51 cluster -f  $calc_dir/"$np"_coms_fit.gro -s $calc_dir/"$np"_com.tpr \
            -o $calc_dir/"$np"_coms_clusters.xpm -g $calc_dir/"$np"_coms_clusters.log \
            -dist $calc_dir/"$np"_coms_clusters.xvg \
            -minstruct 2 -nofit \
            -method jarvis-patrick -M $jarvis_m -P $jarvis_p \
            > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout

    echo "$j 2 Clustering completed in $(($SECONDS/60)) m $(($SECONDS%60)) s" \
        > $calc_dir/perf_report_"$np"_clustering

fi
#############################################################################################################
# Combination of the representative structures, one for each binding mode,                  #
# into a collection for which the binding energy will be computed                           #
if [ $extraction = y ]; then                                                                 #

    # Selection of the best structure per cluster (lower the number, better from
    # the POV of shape complementarity) ...
    best_per_cluster=( $(grep '^[[:space:]]*[[:digit:]]' $calc_dir/"$np"_coms_clusters.log |\
            awk -F '|' '{print $4}' | awk '{print $1}') )
```

```
    # ... and immediate fusion in single trajectory file
    let exe_lineno=$LINENO+1
    if [ -s $calc_dir/"$np"_coms_clusters.gro ]; then
                rm $calc_dir/"$np"_coms_clusters.gro; fi
    for k in ${best_per_cluster[@]}
    do
        grep "Carbonara Stasera! t= "$k"\.0" -A$(($com_atoms+2)) \
                                $calc_dir/"$np"_coms_fit.gro \
                                >> $calc_dir/"$np"_coms_clusters.gro
    done

    file_len=$(wc -l < $calc_dir/"$np"_coms_clusters.gro)
    if [ $file_len -eq $((($com_atoms+3)*${#best_per_cluster[@]})) ];
    then echo 'OK!' > $calc_dir/check; fi
    __generic_output $calc_dir/check $j $home $exe_lineno $launch_time $error_suffix 1
    __del_check
    cp $calc_dir/"$np"_coms_clusters.gro $calc_dir/"$np"_coms_clusters_og.gro

fi
##############################################################################################
# Checking if the representative structures of the collection contain steric clashes and     #
# eventual identification of alternative structures                                          #
if [ $clash_removal = y ]; then                                                              #

    best_per_cluster=( $(grep '^[[:space:]]*[[:digit:]]' \
        $calc_dir/"$np"_coms_clusters.log | awk -F '|' '{print $4}' | \
        awk '{print $1}') )

    export OMP_NUM_THREADS=$n_proc
    SECONDS=0

    # Differential Molecular Mechanics energy
    # (used here to identify steric clashes and later for the MMPBSA scoring)
    if [ -s $calc_dir/"$np"_coms_clusters_og.gro ]; then
        cp $calc_dir/"$np"_coms_clusters_og.gro $calc_dir/"$np"_coms_clusters.gro; fi
    if [ -s $calc_dir/"$np"_com_mme_og.xvg ]; then rm $calc_dir/"$np"_com_mme_og.xvg; fi
    let exe_lineno=$LINENO+1
    echo 1 2 | g_mmpbsa -f $calc_dir/"$np"_coms_clusters.gro -s $calc_dir/"$np"_com.tpr \
        -n $calc_dir/"$np"_com.ndx -mm $calc_dir/"$np"_com_mme.xvg -pdie $pdie \
        > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout

    # Taking only the complexes without steric clashes (VdW interaction > 0) ...

    # ... first we identify them ...
    if [ -e $calc_dir/noclash_list ]; then rm $calc_dir/noclash_list; fi
    if [ -e $calc_dir/clash_list ]; then rm $calc_dir/clash_list; fi
    let exe_lineno=$LINENO+1
    for k in ${best_per_cluster[@]}
    do
        vdw=$( grep "^[[:space:]]*$k\.000" $calc_dir/"$np"_com_mme.xvg \
            | awk 'BEGIN{FIELDWIDTHS="75 15 30"}{print $2}' | sed 's/\.//')
        if [ $vdw -lt 0 ]; then
            echo "$k" >> $calc_dir/noclash_list
        elif [ $vdw -gt 0 ]; then
            echo "$k" >> $calc_dir/clash_list
        fi
    done

    if [ ! -s $calc_dir/noclash_list ]; then
    if [[ ! $(wc -l < $calc_dir/clash_list) -eq ${#best_per_cluster[@]} ]]; then
    __generic_output $calc_dir/noclash_list $j $home $exe_lineno \
                $launch_time $error_suffix 1
    fi
    fi

    if [ -s $calc_dir/clash_list ]; then
        if [ ! -s $calc_dir/"$np"_com_mme_og.xvg ]; then
            cp $calc_dir/"$np"_com_mme.xvg $calc_dir/"$np"_com_mme_og.xvg
        fi

        if [ ! -s $calc_dir/"$np"_coms_clusters_og.gro ]; then
            cp $calc_dir/"$np"_coms_clusters.gro $calc_dir/"$np"_coms_clusters_og.gro
        fi

        # ... then we find alternatives to the ones with steric clashes
        mapfile -t clashes < $calc_dir/clash_list
        for k in ${clashes[@]}; do

            cluster_n=$(grep "^[[:space:]]*[[:digit:]]* |.*|.*|[[:space:]]*\<$k\>" \
                    $calc_dir/"$np"_coms_clusters.log | awk -F '|' '{print $1}' |\
```

```
                    sed 's/ //g' )
            next_cluster=$(($cluster_n+1))
            pattern1="^[[:blank:]]*\<$cluster_n\> "
            pattern2="^[[:blank:]]*\<$next_cluster\> "
            alts=( $(sed -n "/$pattern1/,/$pattern2/{/$pattern2/!p}" \
                    $calc_dir/"$np"_coms_clusters.log | cut -d'|' -f4 | sed "s/\<$k\>//") )

            for alt in ${alts[@]}; do
                grep "Carbonara Stasera! t= "$alt"\.0" -A$(($com_atoms+2)) \
                        $calc_dir/"$np"_coms_fit.gro > $calc_dir/"$np"_"$k"_com_alt.gro
                let exe_lineno=$LINENO+1
                echo 1 2 | g_mmpbsa -f $calc_dir/"$np"_"$k"_com_alt.gro \
                        -s $calc_dir/"$np"_com.tpr -n $calc_dir/"$np"_com.ndx \
                        -mm $calc_dir/"$np"_"$k"_alt_com_mme.xvg -pdie $pdie \
                        > $calc_dir/stdout 2> $calc_dir/stderr
                __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 3
                __del_stderr_stdout

                new_vdw=$( tail -1 $calc_dir/"$np"_"$k"_alt_com_mme.xvg |\
                         awk 'BEGIN{FIELDWIDTHS="75 15 30"}{print $2}' |\
                         sed 's/\.//')
                if [ $new_vdw -lt 0 ]; then
                    cat $calc_dir/"$np"_"$k"_com_alt.gro \
                            >> $calc_dir/"$np"_coms_clusters.gro
                    tail -1 $calc_dir/"$np"_"$k"_alt_com_mme.xvg \
                            >> $calc_dir/"$np"_com_mme.xvg
                    echo $alt >> $calc_dir/noclash_list
                    continue 2
                else
                    tail -1 $calc_dir/"$np"_"$k"_alt_com_mme.xvg \
                        >> $calc_dir/"$np"_com_mme_clash.xvg
                fi
                rm $calc_dir/"$np"_"$k"_com_alt.gro $calc_dir/"$np"_"$k"_alt_com_mme.xvg
            done

        done
    fi

    if [ ! -s $calc_dir/noclash_list ]; then
        __generic_output $calc_dir/noclash_list $j $home $exe_lineno \
                $launch_time $error_suffix 1
    fi

    # ... then we create the files to house their properties ...
    if [ -e $calc_dir/"$np"_coms_clusters_noclash.gro ]; then
        rm $calc_dir/"$np"_coms_clusters_noclash.gro
    fi
    grep -v '^[[:space:]]*[[:digit:]]' $calc_dir/"$np"_com_mme.xvg \
        > $calc_dir/"$np"_com_mme_noclash.xvg

    # ... finally, we fill the latters.
    let exe_lineno=$LINENO+1
    mapfile -t noclash_com < $calc_dir/noclash_list
    for k in ${noclash_com[@]}
    do
        grep "^[[:space:]]*$k\.000" $calc_dir/"$np"_com_mme.xvg \
            >> $calc_dir/"$np"_com_mme_noclash.xvg
        grep "Carbonara Stasera! t= "$k"\.0" -A$(($com_atoms+2)) \
            $calc_dir/"$np"_coms_clusters.gro \
            >> $calc_dir/"$np"_coms_clusters_noclash.gro
    done
    __generic_output $calc_dir/"$np"_coms_clusters_noclash.gro $j $home $exe_lineno \
            $launch_time $error_suffix 1
    __del_stderr_stdout

    # Finally we can delete this monstrocity of a file (we have a small backup)
    rm $calc_dir/"$np"_coms_fit.gro

fi
#############################################################################################
# Actual scoring phase of the representative structures with no steric clash.               #
# If the USER chose the more accurate MMPBSA method, the Molecular Mechanics Energy that was #
# computed in the steric clash identification step will be used in the binding energy       #
# derivation for each non-clashing structure and will not be computed again                 #
if [ $scoring = y ]; then                                                                   #

if [ $fast_gb = y ]; then #-------------------------------------------------------------------

if [ $file_gen = y ]; then #------------------------------------------------------------------

    # Converting complex topology from .top format to .prmtop format using ParmED
    head -$(($com_atoms+3)) $calc_dir/"$np"_coms_clusters_noclash.gro \
```

```
            >  $calc_dir/"$np"_coms_clusters_noclash_1.gro
    cat > $calc_dir/parmed.in <<-EOF
        gromber $calc_dir/"$np"_com.top $calc_dir/"$np"_coms_clusters_noclash_1.gro radii
mbondi2 topdir $np_dir/$mat_ff.ff
        parmout $calc_dir/"$np"_com.prmtop
        go
EOF
    let exe_lineno=$LINENO+1
    parmed -i $calc_dir/parmed.in > $calc_dir/stdout 2> $calc_dir/stderr
    __generic_output $calc_dir/"$np"_com.prmtop $j $home $exe_lineno \
            $launch_time $error_suffix 1
    __del_stderr_stdout
    rm $calc_dir/parmed.in

    # Using CPPTRAJ to strip complex .prmtop of protein and NP data
    # to obtain ligand .prmtop ...
    cat > $calc_dir/cpptraj.in <<- EOF
        parm $calc_dir/"$np"_com.prmtop
        parmstrip :GRA
        parmwrite out $calc_dir/"$np"_com_p.prmtop
        go
        quit
EOF
    let exe_lineno=$LINENO+1
    cpptraj -i $calc_dir/cpptraj.in > $calc_dir/stdout 2> $calc_dir/stderr
    __generic_output $calc_dir/"$np"_com_p.prmtop $j $home $exe_lineno \
            $launch_time $error_suffix 1
    __del_stderr_stdout

    # ... and receptor .prmtop
    cat > $calc_dir/cpptraj.in <<- EOF
        parm $calc_dir/"$np"_com.prmtop
        parmstrip !:GRA
        parmwrite out $calc_dir/"$np"_com_np.prmtop
        go
        quit
EOF
    let exe_lineno=$LINENO+1
    cpptraj -i $calc_dir/cpptraj.in > $calc_dir/stdout 2> $calc_dir/stderr
    __generic_output $calc_dir/"$np"_com_np.prmtop $j $home $exe_lineno \
            $launch_time $error_suffix 1
    __del_stderr_stdout

    # Using CPPTRAJ, convert ASCII .gro trajectory of complexes with no clashes
    # to binary NetCDF
    cat > $calc_dir/cpptraj.in <<- EOF
        parm $calc_dir/"$np"_com.prmtop
        trajin $calc_dir/"$np"_coms_clusters_noclash.gro
        trajout $calc_dir/"$np"_coms_clusters_noclash.nc nobox
        go
        quit
EOF
    let exe_lineno=$LINENO+1
    cpptraj -i $calc_dir/cpptraj.in > $calc_dir/stdout 2> $calc_dir/stderr
    __generic_output $calc_dir/"$np"_coms_clusters_noclash.nc $j $home $exe_lineno \
            $launch_time $error_suffix 1
    __del_stderr_stdout
    rm $calc_dir/cpptraj.in

fi #-------------------------------------------------------------------------------------------

    # Enter mmgbsa sub-directory
    mmgbsa_dir=$calc_dir/mmgbsa
    if [ ! -d $mmgbsa_dir ]; then mkdir $mmgbsa_dir; fi
    cd $mmgbsa_dir

    # For every complex without clashes compute MMGBSA
    for ((k=1;k<=$(wc -l < $calc_dir/noclash_list);k++))
    do
        pose_n=$(head -$k $calc_dir/noclash_list | tail -1 | grep -o '[[:digit:]]*')
        cat > $mmgbsa_dir/mmgbsa.in <<- EOF
            mmgbsa on trajectory frames (docking pose $pose_n)
            &general
            startframe=$k, endframe=$k, interval=1,
            verbose=1, netcdf=1, keep_files=0, use_sander=1,
            /
            &gb
            igb=5, saltcon=0.100,
            /
EOF
        if [ -s $mmgbsa_dir/mmgbsa.out ]; then rm $mmgbsa_dir/mmgbsa.out; fi
        let exe_lineno=$LINENO+1
```

```
        MMPBSA.py -O -i $mmgbsa_dir/mmgbsa.in -o $mmgbsa_dir/mmgbsa_$pose_n.out \
                -cp $calc_dir/"$np"_com.prmtop -rp $calc_dir/"$np"_com_np.prmtop \
                -lp $calc_dir/"$np"_com_p.prmtop -y $calc_dir/"$np"_coms_clusters_noclash.nc \
                > $mmgbsa_dir/stdout 2> $mmgbsa_dir/stderr
        __generic_stderr $PWD $j $home $exe_lineno $launch_time $error_suffix 2
        __del_stderr_stdout
        pose_bind_ener=$(grep 'DELTA TOTAL' $mmgbsa_dir/mmgbsa_$pose_n.out | awk '{print $3}')
        echo "$pose_n $pose_bind_ener" >> $mmgbsa_dir/mmgbsa.out
    done
    rm $mmgbsa_dir/mmgbsa.in

    # Identify best pose from the POV of MMGBSA, print structure and
    # add its value to the scoreboard as the overall protein score
    best_pose=$(sort -n -k 2 $mmgbsa_dir/mmgbsa.out | head -1 | awk '{print $1}')
    grep "Carbonara Stasera! t= $best_pose\.0" -A$(($com_atoms+2)) \
            $calc_dir/"$np"_coms_clusters_noclash.gro \
            > $mmgbsa_dir/"$np"_"$best_pose"_best_pose.gro
    let exe_lineno=$LINENO+1
    gmx51 editconf -f $mmgbsa_dir/"$np"_"$best_pose"_best_pose.gro \
            -o $mmgbsa_dir/"$np"_"$best_pose"_best_pose.pdb \
            > $mmgbsa_dir/stdout 2> $mmgbsa_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout
    best_pose_bind_ener=$(sort -n -k 2 $mmgbsa_dir/mmgbsa.out | head -1 | awk '{print $2}')
    if [ -s $home/"$np"_mmgbsa_scores ] && [ ! -z $(grep -o "^$j" $home/"$np"_mmgbsa_scores) ]
    then sed -i "/^$j/d" $home/"$np"_mmgbsa_scores; fi
    echo "$j  |  $best_pose_bind_ener" >> $home/"$np"_mmgbsa_scores

else #-----------------------------------------------------------------------------------------

    # Polar and non polar PBSA energy of the complex structure
    let exe_lineno=$LINENO+1
    echo 0 | g_mmpbsa -f $calc_dir/"$np"_coms_clusters_noclash.gro \
        -s $calc_dir/"$np"_com.tpr -n $calc_dir/"$np"_com.ndx -i $mmpbsa_files_dir/pbsa.mdp \
        -nodiff -nomme -pbsa -rvdw $rvdw \
        -pol $calc_dir/"$np"_com_polar_nodiff.xvg \
        -apol $calc_dir/"$np"_com_apolar_nodiff.xvg \
        > $calc_dir/stdout 2> $calc_dir/stderr
    __gmx_error $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout

    echo "$j 3 Scoring completed in $(($SECONDS/60)) m $(($SECONDS%60)) s" \
        > $calc_dir/perf_report_"$np"_scoring

    # Creation of polar and nonpolar PBSA output files for the complex to be filled
    sed "s/AAAAA/$np/g" $mmpbsa_files_dir/header_polar \
        | sed "s/BBBBB/$j/g" > $calc_dir/"$np"_com_polar.xvg
    sed "s/AAAAA/$np/g" $mmpbsa_files_dir/header_apolar \
        | sed "s/BBBBB/$j/g" > $calc_dir/"$np"_com_apolar.xvg

    # Extraction of PBSA outputs for each one of the analyzed clusters
    mapfile -t clusters < $calc_dir/noclash_list
    for k in ${clusters[@]}
    do
        # Filling the polar and nonpolar PBSA output files for the complex

        # Poisson-Boltzman
        com_PB=$(grep "^[[:space:]]*$k\.000" $calc_dir/"$np"_com_polar_nodiff.xvg \
            | awk '{print $2}')
        echo "$k   $np_PB   $p_PB   $com_PB" >> $calc_dir/"$np"_com_polar.xvg

        # Solvent Accessible Surface Area
        com_sasa=$(grep "^[[:space:]]*$k\.000" $calc_dir/"$np"_com_apolar_nodiff.xvg \
            | awk '{print $2}')
        sasa_entries="$np_sasa   $p_sasa   $com_sasa"

        # Solvent Accessible Volume
        com_sav=$(grep "^[[:space:]]*$k\.000" $calc_dir/"$np"_com_apolar_nodiff.xvg \
            | awk '{print $3}')
        sav_entries="$np_sav   $p_sav   $com_sav"

        # WCA
        com_wca=$(grep "^[[:space:]]*$k\.000" $calc_dir/"$np"_com_apolar_nodiff.xvg \
            | awk '{print $4}')
        wca_entries="$np_wca   $p_wca   $com_wca"

        echo "$k   $sasa_entries   $sav_entries   $wca_entries" \
            >> $calc_dir/"$np"_com_apolar.xvg
    done

    # Python script to calculate binding energies
    let exe_lineno=$LINENO+1
```

```
    MmPbSaStat.py -m $calc_dir/"$np"_com_mme_noclash.xvg \
        -p $calc_dir/"$np"_com_polar.xvg \
        -a $calc_dir/"$np"_com_apolar.xvg \
        -of $calc_dir/"$np"_binding_energies.dat \
        -os $calc_dir/"$np"_binding_energies_avg.dat \
        > $calc_dir/stdout 2> $calc_dir/stderr
    __generic_stderr $PWD $j $home $exe_lineno $launch_time $error_suffix 1
    __del_stderr_stdout

    # Writing best result to ranking file
    best_deltaG=$( tail -n +4  $calc_dir/"$np"_binding_energies.dat | awk '{print $17}' \
        | sort -k1 -n | head -1 )
    echo "$j     $best_deltaG" >> $home/"$np"_mmpbsa_scores

fi #-------------------------------------------------------------------------------------------

fi
#############################################################################################

done

cd $home
echo -ne " Screening completed at: $(date '+%y%m%d_%H%M%S') \n\n"
```

8.4. Auxiliary Bash function: aesthetics.sh

```
#!/bin/bash

percentage_output () {
    # This function creates a completion bar that fills during the computation
    let normal_index=$1+1
    percent=$( awk 'BEGIN {printf "%.0f", ('"$normal_index"'*100/'"$2"')}' )
    if   [ $percent -lt 10 ];  then echo -ne " #                       (1%) $3 \r"
    elif [ $percent -lt 20 ];  then echo -ne " ##                     (10%) $3 \r"
    elif [ $percent -lt 30 ];  then echo -ne " ####                   (20%) $3 \r"
    elif [ $percent -lt 40 ];  then echo -ne " ######                 (30%) $3 \r"
    elif [ $percent -lt 50 ];  then echo -ne " ########               (40%) $3 \r"
    elif [ $percent -lt 60 ];  then echo -ne " ##########             (50%) $3 \r"
    elif [ $percent -lt 70 ];  then echo -ne " ############           (60%) $3 \r"
    elif [ $percent -lt 80 ];  then echo -ne " ##############         (70%) $3 \r"
    elif [ $percent -lt 90 ];  then echo -ne " ################       (80%) $3 \r"
    elif [ $percent -lt 100 ]; then echo -ne " ##################     (90%) $3 \r"
    elif [ $percent -eq 100 ]; then echo -ne " #################### (100%) $3 \n\n"
    fi
}
```

8.5. Auxiliary Bash function: error_functions.sh

```
#!/bin/bash

__gmx_error () {
    # Function to check generic error messages in GROMACS stderr (must be redirected to file)
    local workingDir=$1
    local loopID=$2
    local whereToWriteErrorLog=$3
    local problematicLine=$4
    local launchT=$5
    local scriptID=$6
    local exitHowManyLoops=$7 ; if [ -z $exitHowManyLoops ]; then exitHowManyLoops=1; fi

    local error=$(egrep -v 'gromacs\.org|more information' $1/stderr |\
          tac | egrep -m 1 -o '^Fatal error|^Error')
    if [ ! -z "$error" ]; then
        echo -ne "$loopID\n\n" \
            >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo -ne "Crash at line: $problematicLine\n" \
            >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        egrep -v 'gromacs\.org|more information' $1/stderr | tac |\
```

```
            awk "/-----/ && ++n == 1, /$error/" | tac \
            >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo -ne "\n\n" >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo "$loopID   $problematicLine" >>  $whereToWriteErrorLog/err_"$scriptID"_"$launchT"
        continue $exitHowManyLoops
    fi
}

__gmx_badEnergy () {
    # Function to check bad_energy error messages in GROMACS stderr (must be redirected to
file)
    local gmxOutput=$1
    local loopID=$2
    local whereToWriteErrorLog=$3
    local problematicLine=$4
    local launchT=$5
    local scriptID=$6
    local exitHowManyLoops=$7 ; if [ -z $exitHowManyLoops ]; then exitHowManyLoops=1; fi

    local ener=$(tac $1 | grep -m 1 'Potential Energy' | awk '{printf "%.0f",
$problematicLine}')
    if [ $ener -gt 0 ]; then
        echo -ne "$loopID\n\n" \
                 >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo -ne "Crash at line: $problematicLine\nBad MM energy!!" \
            >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo -ne "-----------------------------------------------------------------\n\n\n" \
            >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo "$loopID   $problematicLine" >>  $whereToWriteErrorLog/err_"$scriptID"_"$launchT"
        continue $exitHowManyLoops
    fi
}

__generic_stderr () {
    # Function to retrive the errors of a generic program that writes them into stderr
    # (must be redirected to file) and if everything goes smoothly doesn't write anything
    # to STDERR
    local workingDir=$1
    local loopID=$2
    local whereToWriteErrorLog=$3
    local problematicLine=$4
    local launchT=$5
    local scriptID=$6
    local exitHowManyLoops=$7; if [ -z $exitHowManyLoops ]; then exitHowManyLoops=1; fi

    if [ -s $workingDir/stderr ]; then
            echo -ne "$loopID\n\n" \
                 >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo -ne "Crash at line: $problematicLine\n" \
                 >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        cat $workingDir/stderr >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo -ne "-----------------------------------------------------------------\n\n\n" \
                 >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo "$loopID   $problematicLine" \
                 >>  $whereToWriteErrorLog/err_"$scriptID"_"$launchT"
        continue $exitHowManyLoops
    fi
}

__generic_output () {
    # Function for checking if the supposed output of a program exists and is not empty
    # (for programs with ectic error messages behaviour)
    local fileThatMustExist=$1
    local loopID=$2
    local whereToWriteErrorLog=$3
    local problematicLine=$4
    local launchT=$5
    local scriptID=$6
    local exitHowManyLoops=$7 ; if [ -z $exitHowManyLoops ]; then exitHowManyLoops=1; fi

    if [ -s $1 ]; then : ; else
        echo -ne "$loopID\n\n" \
            >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo -ne "Crash at line: $problematicLine\nFile $1 doesn't exist!!\n" \
            >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo -ne "-----------------------------------------------------------------\n\n\n" \
            >> $whereToWriteErrorLog/err_"$scriptID"_"$launchT"_detailed
        echo "$loopID   $problematicLine" \
              >>  $whereToWriteErrorLog/err_"$scriptID"_"$launchT"
        continue $exitHowManyLoops
```

```
    fi
}

__del_stderr_stdout () {
    # Function for removing stdout and stderr after passing an error check
    local dir=$1
    if [ -z $dir ]; then dir=$PWD; fi
    if [ -s "$dir/stderr" ]; then rm $dir/stderr; fi
    if [ -s "$dir/stdout" ]; then rm $dir/stdout; fi
}

__del_check () {
    # Function for removing a temp file that was created to test a condition on certain files,
    # to be used after the test if positive.
    local tempFileToDestroy=$1
    local dir=$2
    if [ -z $tempFileToDestroy ]; then tempFileToDestroy=check
    elif [ ! $(dirname $tempFileToDestroy) = '.' ]; then dir=$(dirname $tempFileToDestroy)
    fi
    if [ -z $dir ]; then dir=$PWD; fi
    if [ -s "$dir/$tempFileToDestroy" ]; then rm $dir/$tempFileToDestroy; fi
}
```

8.6. Auxiliary Bash function: pdb_editing.sh

```
#!/bin/bash

__pdbget () {
    # Function to download a .pdb structure from the Protein Data Bank website
    local input=$1
    local target=$( tr [:upper:] [:lower:] <<< $input )
        wget https://www.rcsb.org/pdb/files/$target.pdb
}

__insertTER () {
    # Function to insert a TER line between 2 non-consecutive residues
    # Especially useful for tleap that otherwise creates long impossible bonds
    local input=$1
    local filename=$(basename -s .pdb $input)
    local dir=$(dirname $input)
    local output=$dir/"$filename"_wTER.pdb

    local prevl_end='[[:digit:]][[:blank:]]*\n'
    local atom='ATOM[[:blank:]]*[[:digit:]]*[[:blank:]]*[[:alnum:]]*'
    local resname='[[:blank:]]*[[:alnum:]]*[[:blank:]][[:blank:][:upper:]][[:blank:]]*'
    local xcoord='[[:blank:]]*-*[[:digit:]]*\.[[:digit:]]\{3\}'
    local ter='TER\n'

    if [ -s $output ]; then rm $output ; fi

    local chainnumbers=( $(cut -c 22 $input | uniq) )
    if [ -z ${chainnumbers[0]} ]
    then
        cp $input $output
        local resnumbers=( $(cut -c 23-26 $input | uniq) )
        for i in ${!resnumbers[@]}
        do
            # First you have to discard the first index which is equal to 0
            if [ ! $i -eq 0 ]; then
                local j=$(( $i-1 ))
                local difference=$(( ${resnumbers[$i]}-${resnumbers[$j]} ))
                if [ $difference -gt 1 ]; then
                    local line=$atom$resname${resnumbers[$i]}$xcoord
                    sed -i -z "s/$line/$ter&/" $output
                fi
            fi
        done
    else
        for k in ${!chainnumbers[@]}
        do
            grep "^.\{21\}${chainnumbers[$k]}" $input > $dir/chain$k.pdb

            local resnumbers=( $(cut -c 23-26 $dir/chain$k.pdb | uniq) )
            for i in ${!resnumbers[@]}
            do
                # First you have to discard the first index which is equal to 0
```

```
                if [ ! $i -eq 0 ]; then
                    local j=$(( $i-1 ))
                    local difference=$(( ${resnumbers[$i]}-${resnumbers[$j]} ))
                    if [ $difference -gt 1 ]; then
                        local line=$atom$resname${resnumbers[$i]}$xcoord
                        sed -i -z "s/$line/$ter&/" $dir/chain$k.pdb
                    fi
                fi
            done
            cat $dir/chain$k.pdb >> $output
            echo 'TER' >>$output
            rm $dir/chain$k.pdb
        done
    fi

}

__extractPROTres () {
    # Function that extracts from a .pdb file only the atoms that belong to residues with
    # standard AA names
    local input=$1
    local dir=$(dirname $input)

    local aliphatic='ALA|LEU|ILE|MET|VAL|'
    local aromatic='PHE|TYR|TRP|'
    local polar_neutral='ASN|GLN|CYS|SER|THR|'
    local acidic='ASP|GLU|'
    local basic='ARG|HIS|LYS|'
    local unique='GLY|PRO|'
    local capping='ACE|NME|'
    local fuckedUpNames='CYH|CSH|CSS|CYX|ILU|PR0|PRZ|TRY'
    local aa_list=$aliphatic$aromatic$polar_neutral$acidic$basic$unique$capping$fuckedUpNames
    egrep "$aa_list" $input > $dir/int_extractPROTres
    mv $dir/int_extractPROTres $input
}

__extractFF14SBprotRes () {
    # Function that extracts from a .pdb file only the atoms that belong to residues with
    # AA names that are recognized by the tool tleap of AmberTools and for which AMBER FF
    # have parameters
    local input=$1
    local dir=$(dirname $input)

    local amino_p1='ALA|ARG|ASH|ASN|ASP|CYM|CYS|CYX|GLH|GLN|GLU|GLY|HID|HIE|HIS|'
    local amino_p2='HIP|HYP|ILE|LEU|LYN|LYS|MET|PHE|PRO|SER|THR|TRP|TYR|VAL|'
        local amino=$amino_p1$amino_p2
    local aminoct_p1='CALA|CARG|CASN|CASP|CCYS|CCYX|CGLN|CGLU|CGLY|CHID|CHIE|CHIP|CHYP|'
    local aminoct_p2='CILE|CLEU|CLYS|CMET|CPHE|CPRO|CSER|CTHR|CTRP|CTYR|CVAL|NHE|NME|'
        local aminoct=$aminoct_p1$aminoct_p2
    local aminont_p1='ACE|NALA|NARG|NASN|NASP|NCYS|NCYX|NGLN|NGLU|NGLY|NHID|NHIE|'
    local aminont_p2='NHIP|NILE|NLEU|NLYS|NMET|NPHE|NPRO|NSER|NTHR|NTRP|NTYR|NVAL|'
        local aminont=$aminont_p1$aminont_p2
    local fckdUpNames='CYH|CSH|CSS|CYX|ILU|PR0|PRZ|TRY'
    local terminations=TER
    local amino_list=$amino$aminoct$aminont$fckdUpNames$terminations
    egrep "$amino_list" $input > $dir/int_extractPROTres
    mv $dir/int_extractPROTres $input
}

__extractHeavyAtoms () {
    # This function removes hydrogen atoms and lone pairs (rare, but out there to mess you up)
    local input=$1
    local dir=$(dirname $input)

    local atom='^ATOM[[:blank:]]*[[:digit:]]*[[:blank:]]*'
    local div='|'
    local hydrogens='H'
    local lone_pairs='LPG'
    egrep -v "$atom$hydrogens$div$atom$lone_pairs" $input \
        > $dir/int_extractPdbHeavyAtoms
    mv $dir/int_extractPdbHeavyAtoms $input
}

__removeExtraHeavyAtoms () {
    # This function removes heavy atoms that belonged to covalent links to ligands and
    # were incorrectly assigned to the standard aminoacid
    local input=$1
    local dir=$(dirname $input)

    local atom='^ATOM[[:blank:]]*[[:digit:]]*[[:blank:]]*'
    local div='|'
    local phosph_p='P'
```

```
    local phosph_o='O.P'
    local phosph_c="C.\'"
    egrep -v "$atom$phosph_p$div$atom$phosph_o$div$atom$phosph_c" $input \
        > $dir/int_removeExtraHeavyAtoms
    mv $dir/int_removeExtraHeavyAtoms $input
}


__repairFckupResNames () {
    # This function fixes some old nomenclature of the following residues
    sed -i 's/CYH/CYS/g' $1
    sed -i 's/CSH/CYS/g' $1
    sed -i 's/CSS/CYS/g' $1
    sed -i 's/CYX/CYS/g' $1
    sed -i 's/ILU/ILE/g' $1
    sed -i 's/PR0/PRO/g' $1
    sed -i 's/PRZ/PRO/g' $1
    sed -i 's/TRY/TRP/g' $1
}

__repairFckupAtomNames () {
    # Thi function fixes some non-canon nomenclature of the following atoms
    sed -i 's/OT1/O  /' $1
    sed -i 's/OT2/OXT/' $1
    sed -i 's/CD  ILE/CD1 ILE/' $1
    sed -i 's/S   MET/SD  MET/' $1
    #sed -i 's/ [[:digit:]]H   VAL/  H   VAL' $i
}

__split () {
    # Split file into n_chunks smaller files. No line is broken in the process and the smaller
    # files are named like the input file plus '_' and a double digit number starting from
'01'.
    local input=$1
    local output="$input"_
    local n_chunks=$2
    local lines_per_chunck=$(( $(wc -l < $input) / $n_chunks ))
    local modulus=$(( $(wc -l < $input) % $n_chunks ))

    split --numeric-suffixes=1 --lines=$lines_per_chunck $input "$output"
    echo $modulus
    if [ $modulus -gt 0 ]; then
            if [ $n_chunks -lt 9 ]; then mapfile -t remainders < "$output"0$(( $n_chunks + 1
))
        else  mapfile -t remainders < "$output"$(( $n_chunks + 1 ))
        fi
        for i in ${!remainders[@]}
        do
            if [ $i -lt 9 ]; then echo ${remainders[$i]} >> "$output"0$(($i+1))
            else echo ${remainders[$i]} >> "$output"$(($i+1))
            fi
        done
            if [ $n_chunks -lt 9 ]; then rm "$output"0$(( $n_chunks + 1 ))
        else rm "$output"$(( $n_chunks + 1 ))
        fi
    fi

}
```

Grazie di cuore a tutti coloro che mi hanno sostenuto in questi 3 lunghi anni, i più difficili della mia vita fino ad ora. Grazie alla mia famiglia, i miei cari amici, nuovi e antichi, e alla mia meravigliosa fidanzata. È solo grazie a voi tutti che sono arrivato a questo traguardo.