Bioinformatics, 34(1), 2018, 49–55
doi: 10.1093/bioinformatics/btx555
Advance Access Publication Date: 4 September 2017
Original Paper

Structural bioinformatics

LightDock: a new multi-scale approach to protein–protein docking

Brian Jiménez-García[1], Jorge Roel-Touris[1], Miguel Romero-Durana[1], Miquel Vidal[1], Daniel Jiménez-González[1,2] and Juan Fernández-Recio[1,3,*]

[1]Life Sciences Department, Barcelona Supercomputing Center (BSC), 08034 Barcelona, Spain, [2]Department of Computer Architecture, Universitat Politècnica de Catalunya, 08034 Barcelona, Spain and [3]Structural Biology Unit, IBMB-CSIC, 08028 Barcelona, Spain

*To whom correspondence should be addressed.

Associate Editor: Alfonso Valencia

Received on May 25, 2017; revised on July 21, 2017; editorial decision on August 30, 2017; accepted on September 1, 2017

Abstract

Motivation: Computational prediction of protein–protein complex structure by docking can provide structural and mechanistic insights for protein interactions of biomedical interest. However, current methods struggle with difficult cases, such as those involving flexible proteins, low-affinity complexes or transient interactions. A major challenge is how to efficiently sample the structural and energetic landscape of the association at different resolution levels, given that each scoring function is often highly coupled to a specific type of search method. Thus, new methodologies capable of accommodating multi-scale conformational flexibility and scoring are strongly needed.

Results: We describe here a new multi-scale protein–protein docking methodology, LightDock, capable of accommodating conformational flexibility and a variety of scoring functions at different resolution levels. Implicit use of normal modes during the search and atomic/coarse-grained combined scoring functions yielded improved predictive results with respect to state-of-the-art rigid-body docking, especially in flexible cases.

Availability and implementation: The source code of the software and installation instructions are available for download at https://life.bsc.es/pid/lightdock/.

Contact: juanf@bsc.es

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Protein–protein interactions are involved in virtually all cellular processes, such as protein expression regulation, cell-cycle control or immune response, among many others (Eisenberg *et al.*, 2000). Characterizing such interactions at atomic level is of paramount importance to better understand pathological conditions at molecular level. However, structural data at atomic resolution is only available for a tiny fraction of the estimated number of protein–protein complexes in human (Mosca *et al.*, 2013; Stumpf *et al.*, 2008; Venkatesan *et al.*, 2009). In this context, computational docking is being increasingly applied for the structural modeling of protein–protein interactions, aiming to complement experimental methods.

From a technical point of view, the docking problem presents two main challenges: the efficient sampling of the conformational and orientation space in search of near-native structures (sampling), and the identification of such near-native structures among the many models generated (scoring) (Moal and Bates, 2010). In most of the cases, the applicability of a given scoring function is strongly dependent on the sampling approaches used. The widely used Fast-Fourier Transform (FFT) based methods can efficiently generate geometrically complementary rigid-body docking poses (Gabb *et al.*, 1997; Katchalski-Katzir *et al.*, 1992). Their main advantage is their high computational speed, which can be even further accelerated by using graphics processing units (GPU) (Ritchie and Venkatraman,

2010). However, the inclusion of new scoring schemes within the FFT approach is difficult, since any extra atomic pairwise scoring function needs to be defined as one or more additional 3D grids, usually at a higher computational cost. Thus very often, it is more efficient to use external scoring functions, such as that in pyDock (Cheng *et al.*, 2007). Another major limitation of the FFT grid-based methods is that they cannot explicitly consider conformational flexibility, although recently reported new developments could alleviate some of these limitations (Padhorny *et al.*, 2016).

Other docking methods are based on explicit representation of the interacting proteins, using a larger variety of scoring functions at atomic or coarser-grained level. However, in the majority of the cases, these scoring functions are highly coupled to a specific sampling protocol. In addition, the computational cost of conformational search in atomistic representation is high, so in practice, these methods usually consist in an initial rigid-body docking search, followed by an additional flexible refinement step (Dominguez *et al.*, 2003; Fernández-Recio *et al.*, 2003; Schueler-Furman *et al.*, 2005). A few docking procedures consider flexibility during the entire search phase, using a reduced representation of the conformational search space (Li *et al.*, 2010; May and Zacharias, 2007; Zacharias, 2003).

The development of new scoring functions that can be independently applied to different sets of docking models generated by a variety of docking methods is an active area of research (Brenke *et al.*, 2012; Moal *et al.*, 2013a,b; Schneidman-Duhovny *et al.*, 2012). However, as above mentioned, the use of new scoring functions in docking has been traditionally limited by the type of sampling method. On the one hand, grid-based docking search methods have difficulties in efficiently including energy-based scoring functions. On the other hand, molecular dynamics, minimization or Monte-Carlo sampling methods usually are linked to a specific force-field and cannot easily accept new scoring schemes. It is thus necessary the development of new sampling schemes in docking that can use multi-scale representation of the proteins, accept flexibility at different degrees and accommodate a large variety of new scoring functions.

In this context, Swarm Intelligence (SI) is a family of the artificial intelligence algorithms inspired by emergent systems in nature, which can perform a more efficient search in a complex space, quite independently on the scoring function to optimize. Basically, those algorithms make use of simple agents that interact locally in a decentralized way, and whose interactions lead to complex emergent patterns or systems in nature, e.g. fish schooling or termite mounds. SI algorithms have been applied to protein–protein docking, such as Particle Swarm Optimization (PSO) in SwarmDock (Li *et al.*, 2010). Another algorithm is Glowworm Swarm Optimization (GSO) (Krishnanand and Ghose, 2009a), a bio-inspired algorithm from the SI family, which is based in the concept that in nature, glowworms are being attracted by other mates depending on the quantity of emitted light. This metaphor is used by the GSO algorithm for simultaneously capturing multiple local optima in multimodal functions. Each agent in the algorithm, a glowworm, carries out a quantity of luciferin which encodes the actual fitness of the position of the agent in the explored search space. The algorithm has been applied to many different problems (Huang and Zhou, 2011; Krishnanand and Ghose, 2009b; Liao *et al.*, 2011), but not explicitly to protein–protein docking. GSO has some advantages over PSO (Krishnanand and Ghose, 2009a). First, while PSO was initially designed for capturing global minima or maxima, GSO was also intended for capturing multimodal local. This property is especially relevant when exploring the protein–protein docking energetic landscape, which tends to be very noisy. This can be overcome in ad-hoc PSO implementations, such as in SwarmDock (Li *et al.*, 2010), which has additional features efficiently adapted to the docking problem and uses multiple trajectories to avoid focusing only on a single global minimum. Moreover, in GSO the number of captured minima or maxima is proportional to the number of defined agents, while this is not true in PSO, which poses a major drawback in systems which are required to scale. On the contrary, the major drawback of GSO over PSO is the computation time, which tends to be one order of magnitude higher.

Here in this work we show that GSO can capture the multiple local and global energetic minima of the docking energetic landscape, independently from the force-field used. The new method shows robust performance in very noisy environments, and good scalability (an interesting property in high-performance computing architectures), and has been devised as a protein–protein docking framework for fast-prototyping and testing of new scoring functions.

2 Materials and methods

2.1 LightDock: GSO algorithm applied to protein–protein docking

The agents in the GSO algorithm are defined as glowworms which carry a luminescent quantity called *luciferin*. At each step of the simulation, the quantity of luciferin l depends on the evaluation of the complex energy by the user-defined scoring S function in the actual search space x and the previous value of the luciferin based on the trajectory of the given glowworm (Eq. 1). Decay of the quantity of luciferin is controlled by the ρ variable, and γ represents the enhancement constant, i.e. how much affects the actual evaluation of the energy in the luciferin quantity.

$$l_i(t+1) = (1-\rho)\cdot l_i(t) + \gamma \cdot S(x_i(t+1)) \quad (1)$$

In LightDock, these parameters are defined by default as: $\rho = 0.4$, $\gamma = 0.6$, initial luciferin $l(0) = 5.0$ (Krishnanand and Ghose, 2009a). Each glowworm g_i initially represents a specific position in the translational and rotational space of the ligand (Eq. 2), where t_x, t_y and t_z are the components of the vector $v_{origin-ligand_{center}}$ and q_w, q_x, q_y and q_z are the components of the quaternion that represents the ligand rotation in the four-dimensional quaternions space. The use of quaternions needs fewer variables than rotation matrices, and avoids the known gimbal lock problem of sampling based on Euler angles or polar coordinates (Shoemake and Ken, 1985).

$$g_i = \left[t_x, t_y, t_z, q_w, q_x, q_y, q_z\right] \quad (2)$$

In addition, the framework has the capability of using the anisotropic network model (ANM) (Atilgan *et al.*, 2001; Doruker *et al.*, 2000) to introduce a certain degree of backbone flexibility during the protein–protein binding process. In this case, each glowworm agent represents, in addition to a translation/rotation ligand position, the extent of deformation along each non-trivial normal mode for the receptor, *nr*, and the ligand, *nl*, in the optimization vector (Eq. 3). The number of normal modes is customizable for the receptor, R, and the ligand, L.

$$g_i = \left[t_x, t_y, t_z, q_w, q_x, q_y, q_z, nr_{1..R}, nl_{1..L}\right] \quad (3)$$

ANM is implemented in the LightDock framework via the ProDy Python library (Bakan *et al.*, 2011). The ANM model is calculated on the $C\alpha$ atoms of the backbone of both receptor and ligand and then extended to the rest of atoms for each residue. By default, we considered the first ten non-trivial normal modes ($R = L = 10$) because of the good compromise between the percentage of recovery

in the interface as seen in (Moal and Bates, 2010) (55% in ten normal modes versus 44% for the first five non-trivial normal modes) and the computation time required.

2.2 Initial receptor/ligand models (glowworms)

Each independent simulation in a LightDock run will contain a fixed number of receptor/ligand models (glowworm swarm) in which the randomly defined ligand positions will cover a given region around the receptor. The initial ligand positions can show a certain overlapping between some of the swarms so that taking all together they will cover all regions around the receptor. The use of independent simulations from different swarms has important advantages. First, only the glowworms within the same swarm can see each other. In this way, the agents can only sample a localized region of the receptor and thus can maximize the acquired information by the swarm in this specific region of the search space. Second, it makes the algorithm to be embarrassingly parallel, with no need of communication between parallel executions and facilitates the optimal execution of the algorithm in high-performance computing architectures or small clusters. Finally, by selecting the swarms centers to be used in the simulation, it offers the opportunity to the users to avoid regions that are known in advance not to be likely involved in binding, i.e. transmembrane domains, as opposed to many FFT-based methods where this filtering has to be performed *a posteriori*.

The setup of the initial glowworm swarms is as follows. Initially, a fixed number of initial swarm centers N_s (by default 400) are defined around the receptor, by using the spiral method (Rakhmanov *et al.*, 1994), and are projected using a ray-tracing technique to find the closest atom from the receptor at the distance of the maximum radius of the ligand. To guarantee a correct sampling over the surface, a certain density of these centers is needed (Supplementary Methods 1.1). For each initial swarm center, glowworms are defined by randomly positioning the ligands (by default 300) so that their center of coordinates are placed within a 10 Å radius sphere from the given swarm center (Fig. 1). LightDock framework can also support the use of precalculated ligand poses generated by FTDock (Gabb *et al.*, 1997) (Supplementary Methods 1.2).

If ANM model is considered, deformational extents for receptor and ligand are randomly generated from a Gaussian distribution with $\mu = 4.0$ and $\sigma = 3.0$. To minimize over-fitting, these values were tested against a small set of only four complexes of the Protein–Protein Benchmark 3.0 (Hwang *et al.*, 2008) that were classified as rigid in the mentioned benchmark. Intuitively, a relatively large value of σ is required to ensure some variability, but μ centered in 0.0 does not seem to be a good choice according to our tests (data not shown), since the range of the normal mode extents generated is not sufficient to recover unbound-bound conformational changes. Other methods as ATTRACT (de Vries and Zacharias, 2013) and SwarmDock (Moal and Bates, 2010) reported similar values for the deformational extents.

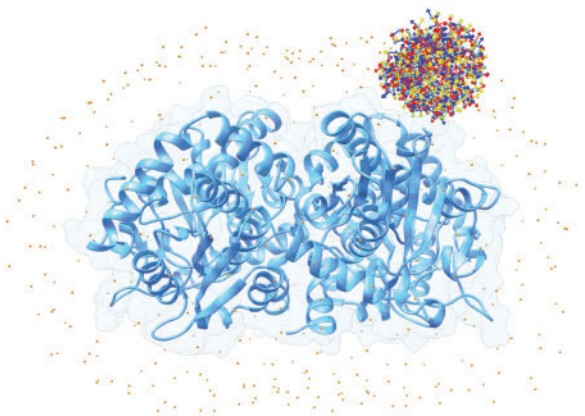

Fig. 1. Initial glowworm swarms together with initial ligand positions. Tryptophan synthase $\alpha(2)\beta(2)$ complex (PDB code 1WDW). The receptor is shown in blue, 300 ligand random positions for a given glowworm swarm are represented using a three-axis arrows model (red, yellow and blue represent the *x*, *y* and *z* orthogonal axis), showing their initial translation and rotation. Orange points over the surface of the receptor represent the 400 initial swarm centers

2.3 GSO sampling

As above described, sets of initial receptor/ligand putative models (glowworms) are defined for their use in independent simulations. Each given glowworm g_i will move towards the best-scoring (luciferin) neighbor glowworm g_j with a given probability p_{ij} (Eq. 4) (Krishnanand and Ghose, 2009a),

$$p_{ij}(t) = \frac{l_j(t) - l_i(t)}{\sum_{k \in N_i(t)} l_k(t) - l_i(t)} \tag{4}$$

where the number of neighbor glowworms (N_i) of glowworm g_i is defined by its *vision range* distance (initially $r_d^i = 5.0$ Å), limited by the *maximum number of neighbors* (by default $N_{max} = 5$).

The distance in the search space between two receptor/ligand models (glowworms) used to update this list of neighbors (N_i) is computed as that between the centers of the minimum ellipsoids of the ligands (translation and rotation of the receptors does not vary). Other definitions of distance based on RMSD did not improve sampling (Supplementary Methods 1.3). The *vision range* of each glowworm r_d^i is dynamically updated at each step (Eq. 5) (Krishnanand and Ghose, 2009a) up to a *maximum vision range* (by default $r_s = 20.0$ Å),

$$r_d^i(t+1) = \; min\{r_s,\; max\{0, r_d^i(t) + \beta(n_t - |N_i(t)|)\}\} \tag{5}$$

where the β parameter indicates how the *vision range* depends on the number of neighbors in the GSO algorithm (by default $\beta = 0.16$).

The evolution from one ligand pose (initial glowworm g_i) towards another one (target glowworm g_j) is composed of two different movements: a translation in the Cartesian space and a rotation in the space of the quaternions. Within the translational space, a new pose will be built from the initial pose by applying a number from the interval (0, 1) as defined in the *translation step* variable (by default 0.5) to the translation vector t_{ij} between g_i and g_j. As for the rotational movement, the movement in the quaternion space is calculated using the spherical linear interpolation (SLERP) (Morrison and Jack, 1992) between the quaternion components of g_i and g_j with a default step of 0.5. In the case of using the ANM representation, a simple interpolation in Euclidean space with a step of 0.5 will be included in both receptor and ligand values. All of these step values can be changed by the users.

2.4 Scoring functions

The movement of the different agents though the search space is driven by the fitness of the scoring function *S* (which defines the quantity of luciferin; Eq. 1). The GSO algorithm is able to optimize the function as long as the agents are uniformly distributed along the search space. In that sense, the optimization method is independent from the search space and makes the strategy valid for any

scoring function used. LightDock framework offers the possibility to add new scoring functions abstracting the way of how molecules are considered. In particular, users can easily specify their own protein models (full atoms or coarse grained) through the *Adapter* class. In the movement step, the model will be rotated and translated and there will be a new class coded by the user, the evaluation module, the one in charge of evaluating the fitness of the scoring function. To demonstrate the possibilities of the framework regarding further extension, nine scoring functions have been implemented (see Results).

The program allows the combination of two or more scoring functions, even if they are defined at different resolution levels. It only requires from the user a file containing, for each line, the name of the scoring function already implemented in LightDock and the weight of the function. For each simulation step, each scoring function is evaluated and the scoring function S is the result of the linear combination of the selected individual scoring functions.

2.5 Clustering of final docking poses

The resulting models from each independent simulation (by default 300) are merged and clustered. Clustering plays an essential role in the final success rate independently of the scoring function applied, since it removes redundant models. We applied a simple clustering procedure based on the Basic Sequential Algorithmic Scheme (BSAS) algorithm (Theodoridis and Koutroumbas, 2008), which is devised to be able to discard redundant poses with a ligand RMSD below 4 Å. First, the best docking pose, in terms of energy, is identified, thus establishing the first sub-cluster. Then, and sequentially, the rest of receptor-ligand complexes are structurally evaluated against the already clustered poses. If their ligand RMSD is within 4 Å from any of the cluster representatives, they will be included in that cluster, otherwise they will establish a new one. The final representative of each cluster corresponds to the structure with the best energy.

Another hierarchical method (Supplementary Methods 1.4) was tested on the cases of the Protein–Protein Docking Benchmark version 5.0 (Vreven *et al.*, 2015), but it yielded worse performance based on the ratio of near native solutions versus the number of total predictions.

3 Results and discussion

3.1 Overall predictive performance of LightDock

The predictive performance of LightDock was tested on the Protein–Protein Docking Benchmark 5.0, composed of a total of 230 complexes. The predictive success rates were based on the percentage of cases in which at least one near-native solution was found within the top N solutions ($N = 10, 100$), as ranked according to the corresponding scoring function. Near-native solutions were defined as those ones with a ligand RMSD < 10 Å with respect to the ligand position in the reference structure (when receptor molecules are superimposed). We tested the performance of LightDock (using default parameters; see Methods) with DFIRE (Zhou and Zhou, 2002) scoring function (LightDock-DFIRE), as well as that of LightDock with a faster implementation of the pyDock (Cheng *et al.*, 2007) scoring function (Supplementary Methods 1.6) called pyDockLite (LightDock-pyDockLite). For each docking case, LightDock generated a total of 120 000 poses, which were clustered as described in the Methods section. After clustering, the final number of docking models obtained by LightDock-pyDockLite ranged between 600 (PDB 1CLV) and 6387 (PDB 1DE4), and near-native poses were found in 70% of the cases. In LightDock-DFIRE, the total number of docking models ranged between 748 (PDB 1CLV) and 6713 (PDB 1AKJ), and near-native poses were found in 75% of the cases.

As a further test, docking simulations on the same complex using different scoring functions were combined in order to capture different near-native predictions. With this purpose, all the models independently generated by LightDock-DFIRE or by LightDock-pyDockLite were merged and re-scored by pyDock scoring function (i.e. combination of LightDock-DFIRE/pyDock and LightDock-pyDockLite/pyDock). The scoring function in pyDock has shown excellent performance in the scorers round of the CAPRI community-wide experiment (Lensink *et al.*, 2016; Pallara *et al.*, 2013), and it is sufficiently fast not to become an overhead in the total computation time of LightDock.

As can be seen in Figure 2A, the use of pyDockLite scoring function within LightDock showed better success rates for the top 10 docking solutions than when using the DFIRE scoring function. The performance of LightDock-pyDockLite is only slightly worse than that of pyDock applied on FTDock docking models (FTDock/pyDock), as in pyDock server (Jiménez-García *et al.*, 2013). For the top 100 success rates (Supplementary Fig. S3A), this difference in performance between LightDock-pyDockLite and LightDock-DFIRE scoring functions is higher, and interestingly, LightDock-pyDockLite top 100 success rate is even slightly better than that of the standard FTDock/pyDock.

Interestingly, the number of successful cases after pyDock rescoring increased for both methods. The improvement was more evident for LightDock-DFIRE models, which after re-scoring with pyDock (LightDock-DFIRE/pyDock), achieved success rates similar to LightDock-pyDockLite. This shows that the differences in the success rates when using pyDockLite or DFIRE as scoring function during the search mainly depended on the scoring of the resulting

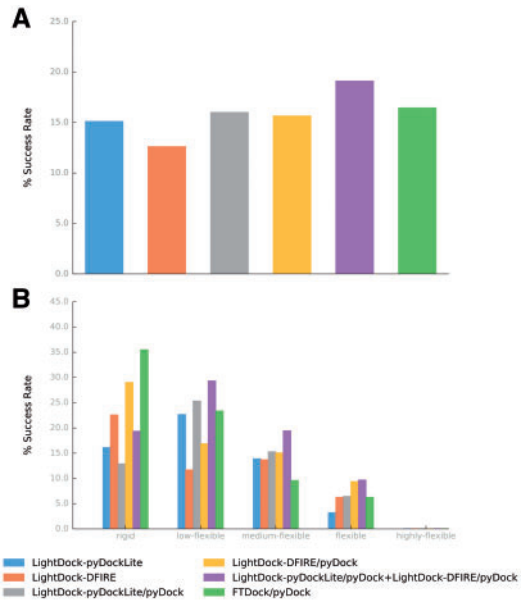

Fig. 2. Predictive success rates for LightDock on the Protein–Protein Docking Benchmark 5.0, $n = 230$. (**A**) Success rates for the top 10 docking models are shown for: LightDock-pyDockLite (blue), LightDock-DFIRE (orange), LightDock-pyDockLite/pyDock (grey), LightDock-DFIRE/pyDock (yellow), combination of LightDock-pyDockLite/pyDock and LightDock-DFIRE/pyDock (purple). For comparison, the performance of the standard protein–protein docking protocols FTDock/pyDock (green) and ZDock 3.0.2 (red) are shown. (**B**) Top 10 success rates are shown according to unbound-to-bound conformational changes

models, and not on the search algorithm itself, given that sampling, even with DFIRE, was able to provide good models that were later identified by pyDock re-scoring. When combining the docking models obtained from the two LightDock versions, and subsequent re-scoring by pyDock, global success rates (19% for top 10; 44% for top 100) slightly improved with respect to the individual simulations, and were even better than those of standard pyDock on FTDock models (Fig. 2A). To explore whether these results by LightDock were due to the above mentioned clustering step, we applied the same clustering method to FTDock docking poses prior to pyDock scoring, but the results did not significantly change (data not shown). For the sake of comparison, we checked that top 10 performance of state-of-the-art ZDock 3.0.2 (Pierce *et al.*, 2011) was only slightly better than the combination of LightDock-pyDockLite/pyDock and LightDock-DFIRE/pyDock (Fig. 2A), but top 100 performance was clearly worse (Supplementary Fig. S3A). However, we should note that this small improvement comes at the expense of doubling the computational cost, since two independent simulations are needed.

Preliminary tests on the use of two scoring functions during search have shown an improvement over the results when using the individual scoring schemes (data not shown). Although further analyses are needed, this opens new possibilities for the efficient combination of different multi-scale models within LightDock protocol.

3.2 LightDock is more efficient in flexible cases

It is interesting to analyze whether the performance of LightDock (with different scoring functions) depends on the flexibility of the interacting proteins. For that, we have classified the cases, according to the RMSD of the interface Cα atoms (I-RMSDCα) between the unbound and bound states (as defined in the Protein–Protein Docking Benchmark 5.0), in the following categories: rigid (I-RMSDCα < 0.5 Å), low-flexible (0.5 Å < I-RMSDCα < 1.0 Å), medium-flexible (1.0 Å < I-RMSDCα < 2.0 Å), flexible (2.0 Å < I-RMSDCα < 3.0 Å) and highly flexible (I-RMSDCα > 3.0 Å). LightDock-pyDockLite performs better in the low-flexible cases (Fig. 2B), while the standard FTDock/pyDock or ZDock protocols were more successful in the rigid cases. The introduction of the ANM representation is probably improving the predictions in the more flexible cases, but at the expense of worsening the results in the rigid cases (due to the introduction of some noise in the already good geometries). Strikingly, LightDock-DFIRE showed its best results in the rigid cases, as in rigid-body FTDock/pyDock. It seems that the DFIRE scoring function cannot take advantage of the ANM model in the more flexible cases, perhaps due to the more coarse-grained character of the potentials. When both approaches are rescored with pyDock, these tendencies remain, which suggests that the scoring function imposed some differences in the ANM-based conformational search. Results for top 100 show a similar fashion compared to top 10 (Supplementary Fig. S3).

The use of ANM-based flexibility aims to provide better predicted models. To evaluate this, we tested a version of LightDock that did not use the ANM model, being thus completely rigid-body sampling, on a heterogeneous set of 30 complexes (6 rigid, 17 low-flexible, 5 medium-flexible and 2 flexible) from the Protein–Protein Benchmark 5.0. The success rates were much worse (10% for top 10; 20% for top 100; as compared to 17 and 27%, respectively, when using ANM and LightDock-DFIRE option). Interestingly, the analysis by category of flexibility shows that there is no difference between the use of ANM in the rigid-body class (17% for top 10 and top 100, using or not ANM), but the difference of success rate comes from an improvement in the low-flexible and medium-flexible categories for both top 10 and top 100 results. This improvement provided by ANM is in the same range as that reported for other state of the art methods that use normal mode analysis (Moal and Bates, 2010; de Vries and Zacharias, 2013).

3.3 Extending the framework to multi-scale

Seven additional scoring functions have been implemented in the framework (see Supplementary Methods 1.5 for more details on defining new scoring functions) as a demonstration of the capabilities of LightDock for being extended with new scoring functions: DFIRE2 (Yang and Zhou, 2008), MJ3h (Miyazawa *et al.*, 1999), PISA (Viswanath *et al.*, 2013), TOBI (Tobi and Bahar, 2005), SIPPER (Pons *et al.*, 2011), a truncated van der Waals scoring as defined in pyDock (Cheng *et al.*, 2007) and the SwarmDock scoring energy (Moal and Bates, 2010) with electrostatics and van der Waals charges from AMBER force-field. Several other options are supported by the framework. For instance, local energy optimization using a non-gradient algorithm has been implemented. For each swarm and each step, the best glowworm in terms of scoring energy is minimized using this non-gradient algorithm. This strategy should help the algorithm to converge in fewer steps (data not shown).

On the other hand, the LightDock framework includes the option of using pre-calculated conformational ensembles, in which case each structure for receptor and ligand is identified by a unique identifier that is added to the optimization vector. For the future, a clearer strategy to define the distance between two conformers is needed so that it can be more efficiently used when one of the glowworms is moving towards the other one. The search could be optimized by maintaining a global list of the most successful or used conformers for receptor and ligand, and then use it to define a probability for selecting a given conformer.

Multi-scale chained simulations are currently supported by the framework. One possible strategy is to perform a first run of the LightDock protocol using a given scoring function and then, after identifying the best energy wells, the predictions could be expanded by a new LightDock run, using the same scoring function or a different one, with finer sampling parameters for instance. In this way, a first quick run could be performed with a coarse-grained force-field, which can be followed by a more accurate refinement using a full-atom scoring function. As mentioned before, LightDock also supports the use of multiple weighted scoring functions upon search, which opens the protocol to the use of multi-scale models at the sampling process. For example, coarse-grained models could be combined with full-atomistic models for a better sampling of the energetic landscape. This approach would be only limited by computational resources.

Finally, the framework includes more than 200 unit tests and more than 10 regression tests from point to point to guarantee a good testing coverage of the code, and additional usage examples to users who aim to extend the framework.

3.4 Computational performance

Optimizations at the level of the scoring function (the most time-consuming part) were performed using the Python C extensions mechanism. The average computation time for all the 230 complexes in the Protein–Protein Docking Benchmark 5.0 using DFIRE scoring function and 400 CPU cores (1 core per swarm) is of 1.5 h, while for pyDockLite scoring function is of 2.0 h in the same conditions. For demonstration purposes, some scoring functions are provided in native Python, Cython (www.cython.org) and Python/C

versions. In addition, LightDock is implemented using multicore and MPI Python libraries, and the algorithm is embarrassingly parallel, which means that can ideally scale proportional to the number of CPU cores used.

4 Conclusions

We have presented here a new protein–protein docking protocol called LightDock, which is based on the GSO algorithm for sampling the translational and rotational space of protein–protein docking, and ANM representation for the inclusion of flexibility. LightDock aims to be a publicly available framework for testing and developing new scoring strategies for protein–protein docking. The use of pyDockLite scoring function during the search provides comparable success rates to state-of-the-art protocols, and the combination with additional functions, like DFIRE, can further improve the predictions. This multi-scale docking framework has capabilities for the use of many different scoring functions (alone or in combination) and the inclusion of flexibility at different resolution levels.

Acknowledgements

We give our thanks to Iain H. Moal for his invaluable help in many discussions on normal mode analysis, PSO and protein–protein docking in general. We are grateful to the Joint BSC-CRG-IRB Research Program in Computational Biology.

Funding

B.J-G was supported by a FPI fellowship from the Spanish Ministry of Economy and Competitiveness. This work was supported by grants BIO2013-48213-R, SEV-2015-0493, TIN2015-65316-P, and BIO2016-79930-R from the Spanish Ministry of Economy and Competitiveness, GA 687698 from the EU H2020 program, and 2014-SGR-1051 from Universitats i Empresa program of Generalitat de Catalunya.

Conflict of Interest: none declared.

References

Atilgan,A.R. *et al*. (2001) Anisotropy of fluctuation dynamics of proteins with an elastic network model. *Biophys. J*., **80**, 505–515.

Bakan,A. *et al*. (2011) ProDy: protein dynamics inferred from theory and experiments. *Bioinformatics*, **27**, 1575–1577.

Brenke,R. *et al*. (2012) Application of asymmetric statistical potentials to antibody-protein docking. *Bioinformatics*, **28**, 2608–2614.

Cheng,T.M.-K. *et al*. (2007) pyDock: electrostatics and desolvation for effective scoring of rigid-body protein–protein docking. *Proteins*, **68**, 503–515.

Dominguez,C. *et al*. (2003) HADDOCK: a protein−protein docking approach based on biochemical or biophysical information. *J. Am. Chem. Soc*., **125**, 1731–1737.

Doruker,P. *et al*. (2000) Dynamics of proteins predicted by molecular dynamics simulations and analytical approaches: application to alpha-amylase inhibitor. *Proteins*, **40**, 512–524.

Eisenberg,D. *et al*. (2000) Protein function in the post-genomic era. *Nature*, **405**, 823–826.

Fernández-Recio,J. *et al*. (2003) ICM-DISCO docking by global energy optimization with fully flexible side-chains. *Proteins*, **52**, 113–117.

Gabb,H.A. *et al*. (1997) Modelling protein docking using shape complementarity, electrostatics and biochemical information. *J. Mol. Biol*., **272**, 106–120.

Huang,Z. and Zhou,Y. (2011) Using Glowworm Swarm optimization algorithm for clustering analysis. *J. Converg. Inf. Technol*., **6**, 78–85.

Hwang,H. *et al*. (2008) Protein–protein docking benchmark version 3.0. *Proteins Struct. Funct. Bioinf*., **73**, 705–709.

Jiménez-García,B. *et al*. (2013) pyDockWEB: a web server for rigid-body protein–protein docking using electrostatics and desolvation scoring. *Bioinformatics*, **29**, 1698–1699.

Katchalski-Katzir,E. *et al*. (1992) Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques. *Proc. Natl. Acad. Sci. USA*, **89**, 2195–2199.

Krishnanand,K.N. and Ghose,D. (2009a) Glowworm Swarm optimization for simultaneous capture of multiple local optima of multimodal functions. *Swarm Intell*., **3**, 87–124.

Krishnanand,K.N. and Ghose,D. (2009b) A Glowworm Swarm optimization based multirobot system for signal source localization. *Stud. Comput. Intell*., **177**, 49–68.

Lensink,M.F. *et al*. (2016) Prediction of homo- and hetero-protein complexes by protein docking and template-based modeling: a CASP-CAPRI experiment. *Proteins*, **84**, 323–348.

Liao,W.-H. *et al*. (2011) A sensor deployment approach using glowworm swarm optimization algorithm in wireless sensor networks. *Expert Syst. Appl*., **38**, 12180–12188.

Li,X. *et al*. (2010) Detection and refinement of encounter complexes for protein–protein docking: taking account of macromolecular crowding. *Proteins*, **78**, 3189–3196.

May,A. and Zacharias,M. (2007) Energy minimization in low-frequency normal modes to efficiently allow for global flexibility during systematic protein–protein docking. *Proteins*, **70**, 794–809.

Miyazawa,S. *et al*. (1999) Self-consistent estimation of inter-residue protein contact energies based on an equilibrium mixture approximation of residues. *Proteins Struct. Funct. Genet*., **34**, 49–68.

Moal,I.H. *et al*. (2013a) Scoring functions for protein–protein interactions. *Curr. Opin. Struct. Biol*., **23**, 862–867.

Moal,I.H. *et al*. (2013b) The scoring of poses in protein–protein docking: current capabilities and future directions. *BMC Bioinformatics*, **14**, 286.

Moal,I.H. and Bates,P.A. (2010) SwarmDock and the use of normal modes in protein–protein docking. *Int. J. Mol. Sci*., **11**, 3623–3648.

Morrison,J. and Jack,M. (1992) QUATERNION INTERPOLATION WITH EXTRA SPINS. In: *Graphics Gems III (IBM Version)*., pp. 96–97.

Mosca,R. *et al*. (2013) Interactome3D: adding structural details to protein networks. *Nat. Methods*, **10**, 47–53.

Padhorny,D. *et al*. (2016) Protein–protein docking by fast generalized Fourier transforms on 5D rotational manifolds. *Proc. Natl. Acad. Sci. USA*, **113**, E4286–E4293. *A*.,

Pallara,C. *et al*. (2013) Expanding the frontiers of protein–protein modeling: from docking and scoring to binding affinity predictions and other challenges. *Proteins*, **81**, 2192–2200.

Pierce,B.G. *et al*. (2011) Accelerating protein docking in ZDOCK using an advanced 3D convolution library. *PLoS One*, **6**, e24657.

Pons,C. *et al*. (2011) Scoring by intermolecular pairwise propensities of exposed residues (SIPPER): a new efficient potential for protein–protein docking. *J. Chem. Inf. Model*., **51**, 370–377.

Rakhmanov,E.A. *et al*. (1994) Minimal discrete energy on the sphere. *Math. Res. Lett*., **1**, 647–662.

Ritchie,D.W. and Venkatraman,V. (2010) Ultra-fast FFT protein docking on graphics processors. *Bioinformatics*, **26**, 2398–2405.

Schneidman-Duhovny,D. *et al*. (2012) A method for integrative structure determination of protein–protein complexes. *Bioinformatics*, **28**, 3282–3289.

Schueler-Furman,O. *et al*. (2005) Progress in protein–protein docking: atomic resolution predictions in the CAPRI experiment using RosettaDock with an improved treatment of side-chain flexibility. *Proteins*, **60**, 187–194.

Shoemake,K., Ken,S. (1985) Animating rotation with quaternion curves. *ACM SIGGRAPH Comput. Graph*., **19**, 245–254.

Stumpf,M.P.H. *et al*. (2008) Estimating the size of the human interactome. *Proc. Natl. Acad. Sci. USA*, **105**, 6959–6964.

Theodoridis,S. and Koutroumbas,K. (2008) *Pattern Recognition*. Elsevier Academic Press, USA, UK.

Tobi,D. and Bahar,I. (2005) Optimal design of protein docking potentials: efficiency and limitations. *Proteins*, **62**, 970–981.

Venkatesan,K. *et al*. (2009) An empirical framework for binary interactome mapping. *Nat. Methods*, **6**, 83–90.

Viswanath,S. *et al*. (2013) Improving ranking of models for protein complexes with side chain modeling and atomic potentials. *Proteins*, **81**, 592–606.

Vreven,T. *et al*. (2015) Updates to the integrated protein–protein interaction benchmarks: docking benchmark version 5 and affinity benchmark version 2. *J. Mol. Biol*., **427**, 3031–3041.

de Vries,S. and Zacharias,M. (2013) Flexible docking and refinement with a coarse-grained protein model using ATTRACT. *Proteins*, **81**, 2167–2174.

Yang,Y. and Zhou,Y. (2008) Ab initio folding of terminal segments with secondary structures reveals the fine difference between two closely related all-atom statistical energy functions. *Protein Sci*., **17**, 1212–1219.

Zacharias,M. (2003) Protein–protein docking with a reduced protein model accounting for side-chain flexibility. *Protein Sci*., **12**, 1271–1282.

Zhou,H. and Zhou,Y. (2002) Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. *Protein Sci*., **11**, 2714–2726.