PROTEINS
STRUCTURE ■ FUNCTION ■ BIOINFORMATICS

Present and future challenges and limitations in protein–protein docking

Carles Pons,[1,2] Solène Grosdidier,[1] Albert Solernou,[1] Laura Pérez-Cano,[1] and Juan Fernández-Recio[1]*

[1] Barcelona Supercomputing Center (BSC), Jordi Girona 29, 08034 Barcelona, Spain

[2] National Institute of Bioinformatics (INB), Computational Bioinformatics, Jordi Girona 29, 08034 Barcelona, Spain

ABSTRACT

The study of protein–protein interactions that are involved in essential life processes can largely benefit from the recent upraising of computational docking approaches. Predicting the structure of a protein–protein complex from their separate components is still a highly challenging task, but the field is rapidly improving. Recent advances in sampling algorithms and rigid-body scoring functions allow to produce, at least for some cases, high quality docking models that are perfectly suitable for biological and functional annotations, as it has been shown in the CAPRI blind tests. However, important challenges still remain in docking prediction. For example, in cases with significant mobility, such as multidomain proteins, fully unrestricted rigid-body docking approaches are clearly insufficient so they need to be combined with restraints derived from domain–domain linker residues, evolutionary information, or binding site predictions. Other challenging cases are weak or transient interactions, such as those between proteins involved in electron transfer, where the existence of alternative bound orientations and encounter complexes complicates the binding energy landscape. Docking methods also struggle when using *in silico* structural models for the interacting subunits. Bringing these challenges to a practical point of view, we have studied here the limitations of our docking and energy-based scoring approach, and have analyzed different parameters to overcome the limitations and improve the docking performance. For that, we have used the standard benchmark and some practical cases from CAPRI. Based on these results, we have devised a protocol to estimate the success of a given docking run.

Proteins 2010; 78:95–108.

Key words: protein–protein docking; Fast Fourier transform; molecular recognition; desolvation; CAPRI.

INTRODUCTION

Physico–chemical and structural studies of molecular recognition between proteins are essential to understand protein function and hence life processes. Given the experimental difficulties for having information on protein–protein interactions at atomic level, computational docking is increasingly used as a complementary tool to predict the structure of a specific complex formed by two given interacting proteins. Although in recent years the field has experienced a rapid improvement, major challenges remain, such as the treatment of flexibility and reliable scoring (see recent reviews on protein–protein docking).[1–4] Most of the available docking methods treat the interacting proteins as rigid-bodies during the whole process, or at least in a first stage. The majority of rigid-body docking methods are based on an exhaustive sampling of the rotational and translational space in search for geometric (or even additional scoring parameters) surface correlation [mainly through FFT (Fast Fourier Transform), spherical polar Fourier, or geometric hashing algorithms]. Two of the most known FFT-based programs are FTDock[5] and ZDOCK, for which several improved versions have been reported.[6–9] Other successful geometric-based docking methods are Hex[10] or MolFit.[11] In addition, a significant number of rigid-body docking methods are based on energy sampling (usually through minimization, molecular dynamics or Monte-Carlo) and/or scoring. This approach has been very successful in the CAPRI experiments (http://www.ebi.ac.uk/msd-srv/capri/). One of the most successful methods in the first two CAPRI editions was ICM-DISCO,[12] which used a Monte-Carlo rigid-body search with grid-based potentials and an essential evaluation step based on electrostatics and desolvation.[13,14] This evaluation scheme was later implemented in pyDock[15] in order to rescore docking sets generated by other different methods, which yielded top results as scorer in the most recent CAPRI edition.[16] Other successful methods also used energy evaluation during or after the docking generation phase, like Haddock,[17] ClusPro/SmoothDock,[18,19] RosettaDock,[20] or ATTRACT.[21]

The authors state no conflict of interest.
Grant sponsor: The Spanish Ministry of Science; Grant number: BIO2008-02882
*Correspondence to: Juan Fernandez-Recio, Barcelona Supercomputing Center (BSC), Jordi Girona 29, 08034 Barcelona, Spain. E-mail: juanf@bsc.es
Received 7 April 2009; Revised 4 July 2009; Accepted 16 July 2009
Published online 5 August 2009 in Wiley InterScience (www.interscience.wiley.com).
DOI: 10.1002/prot.22564

In spite of this variety of methods, there are still many cases that are particularly challenging and pose limitations to current docking approaches. Based on the results of the recent CAPRI experiments,[22–24] the difficult cases for docking involve: (i) large movements upon binding; (ii) weak or transient binding; and (iii) unavailability of X-ray structure for one or both subunits. In these difficult cases, the predictive success rates drop considerably, especially when there is no previous knowledge of possible interface residues. This subject has been recently discussed, describing different possible reasons for poor predictive rates (such as large induced fit, interface size, etc.)[25,26] and analyzing the limitations of current docking procedures.[27] The field needs to improve not only in optimizing or developing new docking methods adapted to these challenging cases but also in identifying these problematic cases and evaluating the reliability of the predictions. Here, we will focus on our docking and scoring methods and how they cope with current challenges in protein–protein docking. We will evaluate the limitations of the rigid-body approach, discuss the importance of the scoring function in order to overcome some of the known docking challenges, and show our attempts to identify successful and/or difficult cases, always from a practical point of view. The specific aspects we aim to explore here include: the choice of optimal parameters for rigid-body sampling and scoring; the number and quality of near-native solutions that are needed during sampling for the optimal efficiency of our scoring function; the effect of flexibility on the docking results and how to overcome the rigid-body hypothesis; the dependence of sampling on the size of the complex; the use of the scoring function to predict energy-related aspects; the use of homology-based models for docking; and the definition of a reliability index to evaluate the docking predictions. For that, we have analyzed our docking protocol by running new calculations on standard benchmarks with different parameters and conditions, and also by giving new light to previous calculations performed in realistic conditions thanks to the blind test provided by the CAPRI experiment.

METHODS

Docking and scoring

The generation of rigid-body docking poses was performed with the Fast Fourier Transform (FFT) program FTDock.[5] To evaluate the effect of sampling on the scoring results, we tested here different running conditions of FTDock: with or without electrostatics, and using either 0.7 or 1.2 Å grid resolution.

We used pyDock[15] for the optimal scoring of FFT-based rigid-body docking sets. The pyDock scoring function is composed of Coulombic electrostatics with a distance-dependent dielectric constant, ASA-based desolvation with atomic solvation parameters previously optimized for rigid-body docking, and an optional term for van der Waals energy (with 0.1 weighing factor and truncated to +1.0 kcal/mol to allow certain overlap of the structures). We tested here the use of pyDock with and without van der Waals energy term. This scoring function has proven to be the best for several targets of the CAPRI experiment.[16] For the CAPRI experiment, in addition to FTDock, we also used ZDOCK 2.1[8] to generate the rigid-body docking poses. Then the docking sets from both programs were scored by pyDock and the best 10 models submitted. For some of the CAPRI targets, additional information about possible interface residues was included in the final scoring as distance restraints with the pyDockRST module.[28]

When using the X-ray structures of the interacting proteins from the PDB, their 3D coordinates were automatically prepared for docking and scoring with the pyDock module "setup." Basically, they were checked for incomplete side chains, which were rebuilt by SCWRL 3.0.[29] In addition, the residues with missing backbone atoms were removed (usually incomplete N-terminal or C-terminal residues). Cofactors, ions, and other heteroatoms were excluded from docking and scoring calculations.

Benchmark

We used here the standard protein–protein Weng's benchmark 2.0,[30] composed of 84 cases in which the structures of the bound and unbound subunits are known. The orientation of the initial structures were randomized (the structures in the benchmark are provided as superimposed to the X-ray structure of the complex, which can yield artificially better results in some cases). The docking results were evaluated by comparing the coordinates of the docking poses with the X-ray structure of the complex. A near-native solution was defined as a docking pose with ligand RMSD <10 Å (RMSD was calculated for the ligand c-alpha atoms with respect to the equivalent ones in the X-ray structure of the complex, after superimposing unbound and bound receptor molecules; this is very similar to the definition of "acceptable" model by CAPRI criteria).[22] The success rate is defined as the percentage of cases in which a near-native solution is found within the top N docking poses, as sorted by pyDock. We calculated these success rates for different N values and the results were typically shown as success rate plots. We paid special attention to success rate for top 10, given that this is a reasonable number of models that can be proposed and later experimentally checked in a realistic situation (moreover, it is actually the number of models submitted and assessed in CAPRI).

Minimization

We used TINKER[31] here to improve the rigid-body docking solutions and, especially, to reduce the number of clashing atoms. For that, we used the "minimize" command in TINKER, with AMBER94 forcefield,[32] and "solvateterm" option with implicit solvation GBSA.[33] The hydrogen atoms were automatically added by the program. When there was any sequence gap in the structure, we manually removed the artificial covalent bond created by TINKER, and distance restraints were defined instead in order to keep the gap boundary atoms to within 0.5 Å distance from the original positions. The typical minimization time for each structure is around 30 min. The convergence criterion is defined by:

$$\frac{|\bigtriangledown E|}{\sqrt{3N}} = 0.5 \qquad (1)$$

where $\bigtriangledown E$ is the gradient of the energy of the system and N the number of degrees of freedom.

RESULTS AND DISCUSSION

The first challenge: optimal parameters for rigid-body sampling and scoring

Clearly, scoring is essential in order to obtain good results in rigid-body docking. We recently reported an energy-based scoring function (pyDock) that was successfully applied to docking sets generated by different methods.[15] One of the methods we tested was FTDock, running on a fixed set of parameters for the sake of simplicity: grid resolution of 1.2 Å and no electrostatics.[15] However, given that scoring itself depends on sampling, even subtle changes in the way the docking poses are generated could introduce differences in the scoring results. Indeed, in previous studies, we used different sampling programs such as ZDOCK or Hex and we could confirm that the results of the scoring can vary depending on the method (data not shown). Then, to check whether we can obtain better results for rigid-body docking and pyDock scoring, we have explored here alternative running conditions for FTDock: (i) with and without electrostatics; and (ii) grid resolution 0.7 or 1.2 Å.

In Figure 1(A) we can see the success rates of pyDock on scoring the docking poses generated by each set of FTDock parameters for a standard protein docking benchmark,[30] when van der Waals energy is included. Figure 1(B) shows the success rates for the same sets when pyDock does not include van der Waals energy. Basically, we can conclude that, while the use of van der Waals does not significantly affect the global success rates, for low ranks, the results without van der Waals are slightly worse. Thus, to reduce the complexity of the analysis, from now on, we will continue our analysis

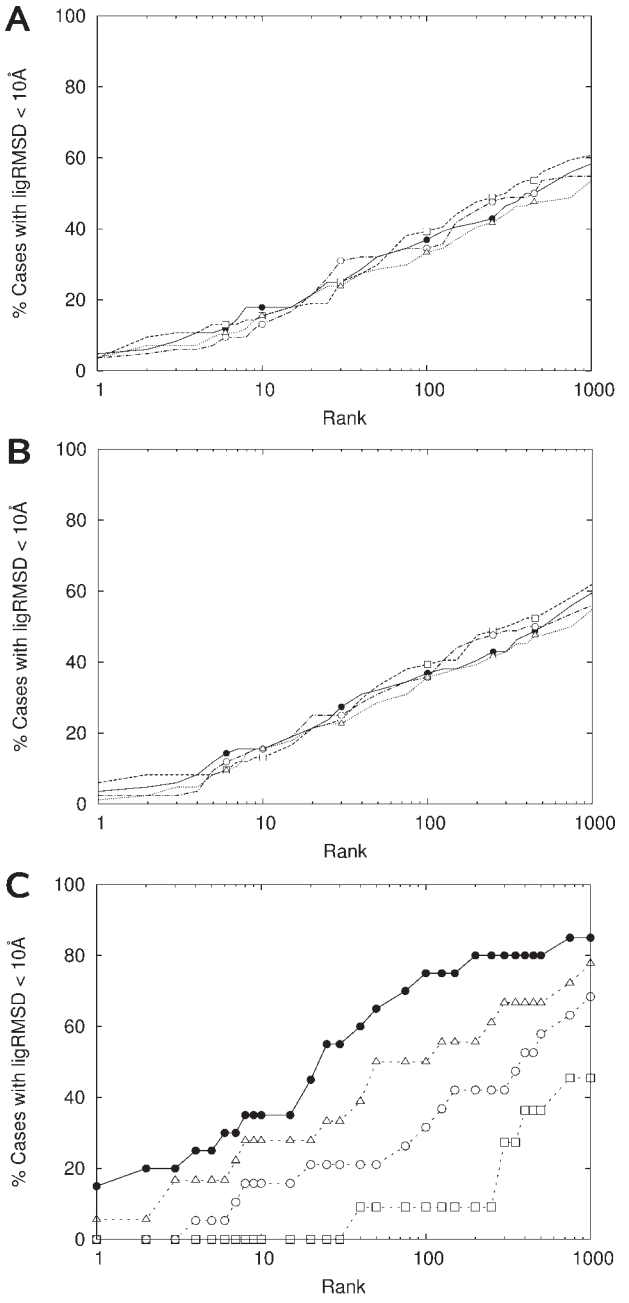

Figure 1

(**A**) pyDock success rates (% of cases with a near-native solution with rank within threshold specified in abscissas) in the docking sets generated by different FTDock versions: electrostatics and 0.7 Å grid resolution (solid line with filled circles); electrostatics and 1.2 Å grid resolution (dotted line with open triangles); no electrostatics and 0.7 Å grid resolution (dashed line with open squares); no electrostatics and 1.2 Å grid resolution (dashed line with open circles). (**B**) Success rates of pyDock without van der Waals (same conditions as in A). (**C**) pyDock success rates for cases grouped by the number of generated near-native solutions by FTDock (electrostatics and 0.7 Å grid resolution): cases with more than 10 near-native solutions are shown in solid line with filled circles; cases with 5 to 9 near-native solutions in dashed line with open triangles; cases with 2 to 4 near-native solutions in dashed line with open circles; cases with only one near-native solution are in dashed line with open squares.

Table I
Docking Results on Weng's Benchmark 2.0 According to Different Parameters and for Cases Grouped by Different Conditions

	Cases with near-native solutions (%)[a]	Number of near-native solutions[b]	Success rate top 10 (%)[c]
FTDock parameters			
ELE 0.7 Å	81	8.0	18 (22)
noELE 0.7 Å	85	8.5	16 (19)
ELE 1.2 Å	73	5.9	16 (22)
noELE 1.2 Å	77	6.4	13 (17)
Difficulty			
"Rigid-body"	89	8.8	24 (27)
Medium	62	5.6	0 (0)
Difficult	50	1.3	0 (0)
Unbound-bound RMSD (Å)			
<0.5	83	12.8	58 (70)
0.5–1.0	96	7.9	18 (19)
1.0–1.5	92	7.6	13 (14)
1.5–2.0	50	5.2	0 (0)
>2.0	38	1.3	0 (0)
Complex type			
Enzyme-inhibitor	100	13.4	39 (39)
Antibody-antigen	73	5.4	14 (19)
Other	74	5.1	8 (11)
Grid size (in cell units)			
<150	100	17.7	33 (33)
150–200	93	7.7	21 (23)
200–250	74	4.1	13 (18)
>250	30	4.0	0 (0)
Docking mean energy (DME)			
<0	75	9.7	50 (67)
0–5	88	8.1	25 (29)
5–10	83	9.5	17 (20)
10–15	79	7.6	18 (23)
15–20	82	5.2	9 (11)
>20	75	6.7	0 (0)

[a]Percentage of cases with at least one near-native solution (defined by ligand RMSD < 10 Å with respect to reference complex).
[b]Average number of near-native solutions per case (considering only those cases with near-native solutions).
[c]pyDock (including van der Waals) success rate for top 10 (i.e., percentage of cases in which a near-native solution is found within the 10 lowest-scoring poses; in brackets, the same parameter but considering only those cases with near-native solutions).

with the van der Waals energy included in the scoring (weighed by 0.1, see Methods).

As for the different FTDock settings, there were no major global differences, but we could see some trends. For instance, if we specifically focus on the success rates for finding a near-native solution with rank 10 or below, the best results were obtained with ELE 0.7 Å, while the worse ones came from NoELE 1.2 Å [Fig. 1(A), Table I]. Next we tried to find whether these success rates could be explained by differences in FTDock sampling that might affect the number and quality of the near-native docking solutions generated with each parameter set.

First, we tested the efficacy of FTDock in generating near-native solutions. Table I shows, for each set of parameters, the percentage of cases in which there is at least one near-native solution within the 10,000 generated docking poses. We found here small variations among the results from the different FTDock settings, ranging from NoELE 0.7 Å, which generated near-native solutions in 85% of the cases, to ELE 1.2 Å, which generated near-native solutions in only 73% of the cases. In general, 0.7 Å resolution (with and without electrostatics) gave better sampling: there were more cases with near-native solutions, and in addition, these cases had in average more near-native solutions (Table I). Thereby, different conditions in the sampling process could affect the overall FTDock + pyDock success rates: the higher the number of near-native solutions generated by FTDock (i.e., with 0.7 Å resolution), the better the overall success rates expected by pyDock. Indeed, this is what we saw for rank values of 10 or below [Fig. 1(A)]. In particular for top 10, the best results were obtained with electrostatics and a resolution of 0.7 Å (for other rank values, a resolution of 0.7 Å gave always better rates, but dependence on the use of electrostatics seemed more random). In the above line of reasoning, we see in Figure 1(C) that for a given set of FTDock parameters (ELE 0.7 Å is shown here, but same applies to the other sets of parameters) success rates strongly depended on the number of near-native solutions generated per case. For those cases with more than 10 near-native solutions, the success rates were much better, while the results were clearly worse when the number of near-native solutions decreased. Actually, when only one near-native solution was generated by FTDock, pyDock was never able to identify it within the top 10 solutions.

In conclusion, we found in this analysis that at high resolution FTDock generates more cases with solution and higher number of solutions per case. The existence of a significant number of near-native solutions in the docking pool is essential for pyDock efficacy; therefore, success rates are higher at 0.7 Å resolution. In addition, we observed that success rates for top 10 are better with electrostatics, and although the use of electrostatics increases the FTDock computational times (originally ranging from 30 min to 10 h) to almost double, this is not dramatic when compared with the cost of pyDock scoring (typically between 10 and 20 times longer). Therefore, given the improvement in the success rates and the low relative computational cost, we decided to focus our analysis on the docking sets generated by FTDock ELE 0.7 Å.

The challenge of flexibility

Conformational flexibility upon binding is one of the clear limitations of current rigid-body docking approaches, so we have explored here how our protocols are dealing with this. The complexes in the benchmark were previously classified according to the expected difficulty for docking as "rigid-body," medium, or difficult.[30] Our results shown in Table I indicate that FTDock gener-

ates more near-native solutions for the "rigid-body" type cases: we found at least one near-native solution in 89% of the cases, with an average of 8.8 near-native solutions per case. For comparison, this percentage slightly decreased to 62% for the medium difficulty type cases (with an average of 5.6 near-native solutions per case). Finally, 50% of the difficult type cases had at least one near-native solution, but the average number of near-native solutions per case dramatically dropped to 1.3.

Next, we wanted to check whether the pyDock scoring results would also depend on the difficulty of the complex. As observed in FTDock sampling, the results were much better for the "rigid-body" type cases [Fig. 2(A)]. As we can see in Table I, the overall FTDock + pyDock success rate for top 10 in these cases was significantly above the success rate of the complete benchmark. Expectedly, if we only consider the cases with at least one near-native solution, success rate increased. However, success rates for the medium and difficult cases were really poor: no near-native solution was found within rank 10 in any of those cases. The poor docking results for the difficult cases can be explained from the fact that, although FTDock found near-native solutions in 50% of the cases, it only generated 1.3 near-native solutions per case. We have discussed above [Fig. 1(C)] that this is too low for pyDock to be efficient; therefore, the reason for these low predictive rates seems to be FTDock failure to generate sufficient number of near-native solutions (poor sampling). In comparison, for the medium type cases, FTDock generated an average of 5.6 near-native solutions per case. We have seen above [Fig. 1(C)] that with these values, docking success rates should be above average. However, for these cases, pyDock was unable to identify any of the near-native solutions within rank 10, which indicates that the difficulty for the medium type cases comes from the scoring rather than from the sampling process.

We have further analyzed these categories from a flexibility point of view. Although one of the categories is called "rigid-body," this classification was previously defined according to the docking results on a standard benchmark, not based on the actual unbound-bound difference of the subunits.[30] For a more precise classification in terms of flexibility, we have calculated the RMSD of the C-alpha atoms of the unbound receptor and ligand when they are separately superimposed onto those of the bound structure. In only one target (PDB code 1H1V), it is technically impossible to find any near-native solution within our criterion, given that the RMSD of the unbound ligand optimally superimposed onto the bound one is >10 Å. In the remaining cases, we could potentially have obtained near-native solutions even for the difficult cases, so we must find other reasons for the poor results in this category. We have analyzed whether the difficulty-based classification and/or the docking success rates are related to the unbound-bound

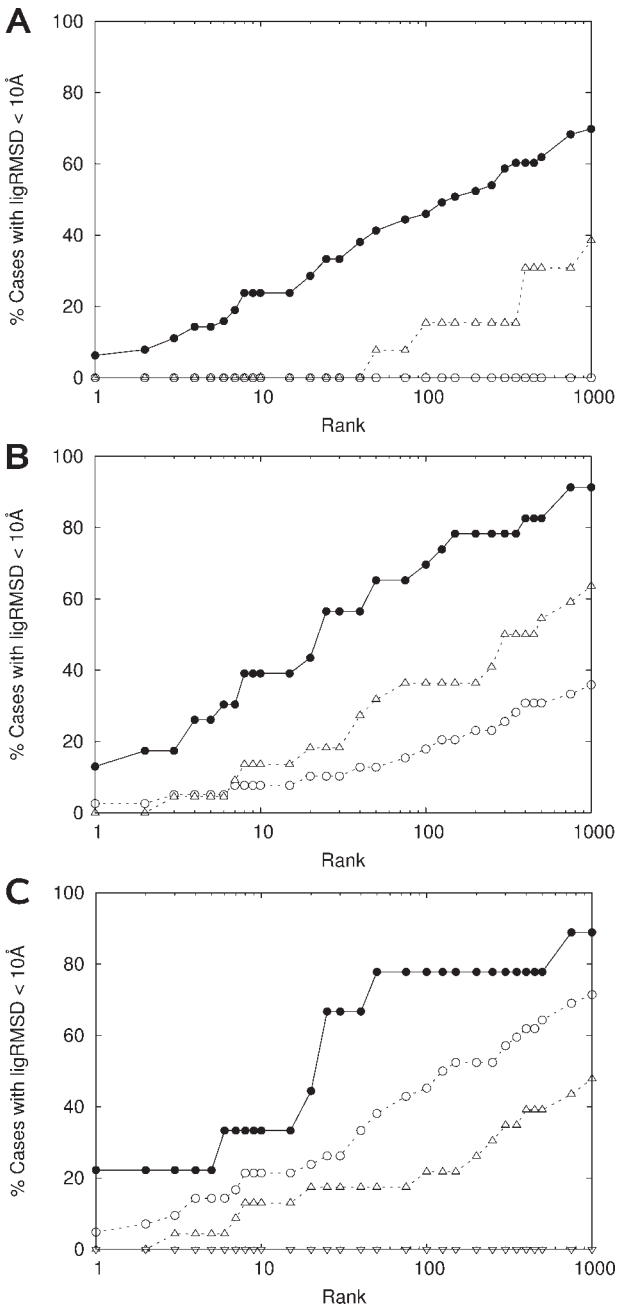

Figure 2

(**A**) pyDock success rates according to the case difficulty as defined in the Weng benchmark: rigid-body (solid line with filled circles), medium difficulty (dashed line with open triangles), and difficult cases (dashed line with open circles). (**B**) pyDock success rates according to the type of complex as defined in the Weng benchmark: enzyme/inhibitors (solid line with filled circles), antibody/antigen (dashed line with open triangles), and "other" kind of complexes (dashed line with open circles). (**C**) pyDock success rates grouping the cases by the grid size defined by FTDock: below 150 (solid line with filled circles), from 150 to 199 (dashed line with open circles), from 200 to 249 (dashed line with open triangles) and above 250 grid cells (dashed line with open inverted triangles).

flexibility. The average RMSD of receptor and ligand (after individual superimposition of unbound-bound molecules) ranges from 0.1 to 3.3 Å in the "rigid-body" cases, from 0.8 to 2.3 Å in the medium cases, and from 1.6 to 7.8 Å in the difficult cases. In the difficult type cases, we can find larger RMSD values, but also some values within the same ranges as in the other categories. It is clear that these categories, previously defined according to the docking results on a benchmark,[30] do not reflect well the amount of conformational variability upon binding. Therefore, we have reclassified here the docking cases in order to describe better their flexibility upon binding. Table I shows the docking results according to the average RMSD of receptor and ligand after optimal superposition. The docking results are clearly better for those cases with small movement upon binding (for RMSD $<$0.5 Å, the success rate for finding a near-native solution within top 10 is 58%), which we could consider as true rigid-body cases. For higher RMSD values, the results deteriorate rapidly, and indeed, for those with RMSD $>$ 1.5 Å the top 10 success rate drops to zero.

In cases with significant induced fit or conformational movement upon binding, a rigid-body geometry-based sampling is having evident difficulties, as docking orientations close to the native one may not have perfect surface complementarity. Moreover, even if some near-native orientations are generated, scoring them correctly will be difficult due to the non-native conformation of the interfaces (which can only keep a small fraction of the native interactions). We clearly need to improve sampling in these flexible cases, which are probably the most important to predict from a biological point of view (as many of the signaling complexes undergo significant conformational movements upon binding). The challenge is clearly the treatment of flexibility. In these cases, finer rigid-body sampling is not going to improve the success rates, as it will only introduce more false-positives that the scoring function will not be able to discriminate. It could be possible to improve the scoring function (e.g., softer potentials, optimized parameters for flexible side-chains, etc.), but we believe it would be more reasonable to start improving flexible sampling. In summary, these cases represent the limit for the rigid-body approach, and improvement on rigid-body sampling for them will not probably be reflected on better success rates unless we introduce flexibility in the sampling step.

We also analyzed the results according to the complex type (previously classified as enzyme-inhibitor, antibody-antigen, and other).[30] On the enzyme-inhibitor type, FTDock generated near-native solutions for all the cases, and pyDock success rate (39% for top 10) was significantly better than that of the global benchmark. For the antibody-antigen cases, FTDock generated near-native solutions in 73% of the cases (with an average of 5.4 solutions per case), and success rate (14% for top 10) was close to that of the global benchmark. However, for the "other" type of complexes, although the number of cases with solution and the average number of near-native solutions were similar to those of the antibody-antigen type, pyDock success rates were well below those of the whole benchmark. This shows that pyDock is quite efficient with enzyme-inhibitor [Fig. 2(B)] and even with antibody-antigen cases (other automatic docking methods usually fail for this type of complexes), whereas it is less efficient with the "other" type of complexes. Again, in the enzyme-inhibitor and antibody-antigen types of complexes, a better sampling is expected to improve success rates, but for the "other" type of cases, we would need to improve something more than the sampling. Actually, in the "other" type of complexes there are more cases of the "difficult" type, corresponding to cases with large conformational changes upon binding. Here, introduction of flexibility during the sampling phase would be essential. These results are somehow related to the ones obtained by case difficulty and flexibility (above discussed).

Size matters, large complexes are difficult to predict

We also analyzed here the docking results according to the size of the grid generated by FTDock ELE 0.7 Å to represent the proteins. The grid size (in number of cells) is defined by:

$$\text{grid size} = \frac{2 \cdot r_{\text{rec}}^{\max} + 2 \cdot r_{\text{lig}}^{\max} + 1}{\text{gridres}} \qquad (2)$$

where $r_{\text{rec}}^{\max}$ is the maximum radius of the receptor (distance from the center of coordinates of the receptor protein to its farthest atom), $r_{\text{lig}}^{\max}$ the maximum radius of the ligand, and gridres the grid resolution (0.7 Å in our case). As can be seen in Figure 2(C), the prediction rates drop dramatically for cases with large grid size. And the best docking results are clearly in cases with grid sizes below 150: success rate of 33% for top 10 using FTDock with electrostatics and 0.7 Å. For these complexes formed by small proteins, not only does FTDock generate near-native solutions for all the cases but also the average number of near-native solutions is quite high. Because of this, pyDock success rates for these cases are better than those of the whole benchmark. For the cases with size between 150 and 200, FTDock efficiency in generating near-native solutions is close to average, and consequently, pyDock success rates are also around average. For cases with grid size between 200 and 250, FTDock efficiency in generating near-native solutions starts to decrease and the same trend is seen for pyDock success rates. Finally, for large complexes ($>$250 grid size), FTDock generates near-native solutions for just a few cases, and therefore, pyDock results are quite poor (0% success rate for top 10). Thus, from this analysis, we can

conclude that the dependence of the docking success rates on the grid size is very much related to sampling issues. It seems reasonable to think that a better sampling would be needed for the complexes formed by large proteins in order to improve the docking results. Given that grid resolution is kept fixed for all cases, the key aspect is the rotational angular resolution. This value is also kept fixed for all cases, but since the effective surface exploration additionally depends on the distance from the surface to the center of coordinates, it could happen that a given angular resolution value might give satisfactory surface sampling in small proteins, but be insufficient for the large ones. The logical approach would be to increase the angular resolution for larger proteins. To test this, we have rerun FTDock with increased angular resolution (from the default resolution of 12° to 9°, thus increasing the number of rotations from 9240 to 21,440) for the "rigid-body" type cases (we focus here only on these cases to avoid other possible factors affecting the results) with grid size between 150 and 250 (we have excluded cases with grid size >250 because they would need a significantly higher number of rotations). Indeed, the results indicate an improvement in sampling. Before, there were five failed cases that did not have any near-native solution (PDB codes 1BJ1, 1FC2, 1GHQ, 1K4C, and 2QFW). All of them had at least one near-native solution after increasing angular resolution. The case 2QFW (grid size 230) is certainly remarkable: there was not any near-native solution with the original angular resolution, but we found one ranked 3 after increasing the angular resolution. However, if we focus not only on the failed cases but also on all of them, the pyDock success rate for the top 10 solutions is practically the same independently on the angular resolution. The problem is that the number of false positives also increases with the number of rotations, which is a limitation for scoring. In conclusion, when using FFT-based methods for generating rigid-body docking orientations, it would be essential not only to increase angular resolution for larger proteins (i.e., more rotations for cases >150 grid size) but also to use some criteria to improve enrichment in near-native solutions.

Can binding affinity be estimated from docking?

Estimating the binding energy between two given proteins would be essential in order to identify whether these proteins interact or not, a critical point for the study of systems biology and protein interaction networks. However, binding energy cannot be accurately estimated from the crystallographic structure of the complex, let alone from the docking results. What was previously found is that high affinity complexes seem to give better docking results. As an example, a past CAPRI test[22] nicely showed that, among the three complexes formed by α-amylase and different camel antibodies (targets T4, T5, and T6, respectively), most of the groups succeeded only on T6, indeed the one with the strongest experimental affinity (T6 K_d = 3.5 n*M*, compared with T4 K_d = 25 n*M* and T5 K_d = 235 n*M*).[34] In that case, there was a clear correlation between docking results and binding affinity, suggesting that docking is more reliable for high affinity complexes. However, an even more interesting question is to know if the reliability of the docking results can be estimated from the scoring values of the resulting solutions.

We evaluated in this work whether the pyDock docking energy of the generated decoys could give us any hint about the success of the docking predictions. For that, we computed the mean pyDock scoring value of all the 10,000 docking poses generated by FTDock (including good and bad docking poses), which we call "Docking Mean Energy" (DME). As can be seen in Table I, we have classified the docking cases in different categories according to their DME values (the standard errors of these DME values ranged from 0.07 to 0.18, so the possibility of misclassifying a case is very small). We clearly see that the success rates strongly depend on the DME value. For those cases where the DME value is below 0.0 kcal/mol, the success rate for the top 10 solutions is 50%, well above the average. Good success rates are also found for those cases with DME values between 0.0 and 5.0 kcal/mol. For DME values between 5 and 15 kcal/mol, success rates are around average, and for DME >15 kcal/mol, success rates drop significantly. Interestingly, neither the percentage of cases with at least one near-native docking solution nor the average number of near-native solutions per case seem to depend on the DME value (Table I). Actually, these values are similar, within noise, to the global values. This clearly indicates that the good success rates in the cases with lower DME values cannot be explained because of good FTDock sampling. On the contrary, these results are consequently related to the ability of the pyDock scoring function. Moreover, if we consider only those cases with at least one near-native solution, the success rates for those with DME < 0.0 kcal/mol increases up to 67%, which is quite impressive. In other words, if FTDock had generated near-native solutions for more of these cases, pyDock might probably have detected them. In summary, these cases in which the docking decoys have in average lower pyDock scoring values are also the ones in which pyDock is very efficient in identifying near-native docking solutions, independently on the number of them generated by FTDock.

Figure 3 shows the dependence of the success rate on DME, with the percentage of cases that have DME values greater than the given cutoff in abscissas. This can help to define a level of confidence on the docking results for a given simulation.

As explained above, the DME values computed from all the docking poses are very robust (the standard errors of the DME values are very small). We further checked

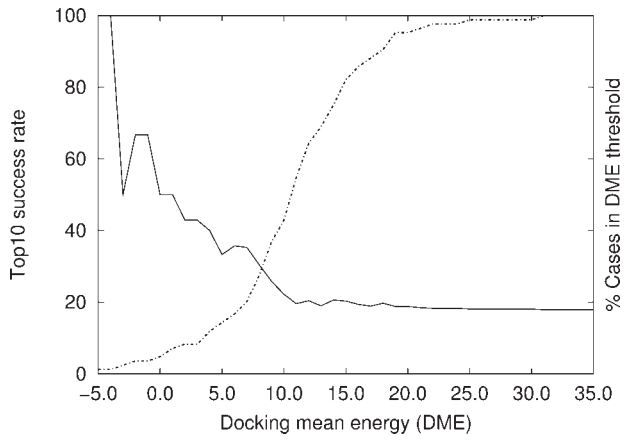

Figure 3

pyDock success rates for the top 10 decoys (solid line) considering the cases with docking mean energy (DME) within the threshold specified in abscissas (the % of cases with DME within the threshold is represented in dashed line).

that these values did not depend on the sample size: the DME values computed from subsets of 1000 docking poses, randomly selected from the complete sets, were not significantly different from the original DME values (consequently, they were able to classify the docking cases into the same categories). However, when the DME values were computed, not from random sub-sets, but from the lowest-energy N docking decoys (N = 1, 10, 100, 1000, 5000), the docking cases were reclassified in different groups, none of which had better success rates than those formed with the original DME values (data not shown). It is surprising that the mean energy of all docking poses (or a random sub-set) is a better indicator of the docking success than the mean energy of the best-scored solutions. In principle, one would expect that in a good docking case, a number of near-native docking solutions with lower pyDock energy should be generated. In this situation, the mean energy of the top 10, 100, or even 1000 docking poses should be better in the good docking cases. On the contrary, we observed that the best predictor for the docking success is the mean energy of the whole sampled docking landscape. This is actually an indicator of the quality of the general ensemble of poses generated by FTDock (i.e., not just the near-native ones or those around the real binding areas), and it might reflect the abundance of poses with better shape complementarity (which in turn will have better pyDock energy).

However, in spite of the clear correlation between pyDock energy values and the docking success rates, as well as that between docking success and experimental binding energy (at least for some cases), we could not find any correlation between pyDock energy values and the experimental affinities in the few cases in which the latter are available (data not shown). This could be due to the lack of sufficient data for reliable statistics (for instance, we could not find any experimental binding energy data for cases with DME $<$ 0.0 kcal/mol, which are the most successful ones for pyDock).

Interacting subunits with no available 3D structure: the challenge of using models in docking

One of the main challenges in docking is the use of models for the interacting subunits during the simulations. Although most of the reported methods have been optimized and benchmarked on cases in which there are available X-ray or NMR structures for the unbound subunits, in a realistic situation it is quite usual not to have the structure for one or both interacting proteins available. In these cases, one could build a homology-based model of the proteins of unknown structure and use it for docking. The performance will consequently drop, as the uncertainty in the model is added to the intrinsic noise of the docking results. However, to our knowledge, there is not any systematic benchmark studying the use of models in docking. Fortunately, we can define a small blind test formed by recent CAPRI cases in which one or two of the interacting proteins needed to be modeled (Table II). We have analyzed here the performance of pyDock, previously run in blind conditions. The results indicate that the rigid-body approach struggles when using homology-based models. Indeed, from the six targets in predictors in which at least one of the subunits needed to be modeled (T24, T28, T35, T36, T37, T38), we managed to submit a correct docking model for only one case (T35). Interestingly, this case was a domain–domain interaction, with flexible interdomain linker as restraint. This confirm our recent results from a different benchmark showing that docking can be successfully applied to this type of domain–domain cases and that the domain modeling does not significantly decrease the success rates.[35] The bottom line is that, in spite of the inferior performance in cases in which one or both subunits need to be modeled, the docking results can dramatically improve with a minimum of information used as distance restraints.

A practical challenge from CAPRI: choice of optimal parameters in a realistic situation

In addition to the standard docking benchmark, the CAPRI experiment provides an excellent test set formed by realistic docking problems. To keep the blind conditions, we have analyzed here the docking models we previously generated for CAPRI targets 24 to 38 (excluding targets 33 and 34, as they were protein-RNA docking cases). The official assessment for targets T24 to T28 has already been published.[24] However, although targets T29 to T38 have been assessed (except T31) and the results

Table II
Results of pyDock on CAPRI Predictors Experiment (Cases Assessed by the Organization)

Target	Type[a]	PDB[b]	CAPRI submission[c]	pyDock (vdw)[d]			pyDock (no vdw)[d]		
				Rank[e]	RMSD[f]	Method[g]	Rank[e]	RMSD[f]	Method[g]
T24	UH	2J59	No vdw	27	9.5	ftdock	20	9.5	ftdock
T25	UB	2J59	vdw	3	**	zdock	12	8.8	ftdock
T26	UU	2HQS	No vdw	40 (3)	8.7	zdock	19 (3)	8.7	zdock
T27	UU	2O25	vdw	977 (5)	5.6	ftdock	703 (7)	5.6	ftdock
T28	HH	2ONI	No vdw	No hits	—	—	No hits	—	—
T29	UB	2VDU	vdw	1	**	zdock	1	**	zdock
T30	UU	2REX	No vdw	344	8.6	ftdock	122	8.6	ftdock
T32	UU	3BX1	vdw RST	55 (5)[h]	8.7	ftdock	54 (1)[h]	4.1	zdock
T35	HH[i]	—	vdw TET	1	*	zdock	3	*	zdock
T36	HB	—	vdw	145	>0.139	ftdock	107	>0.139	zdock
T37	UH	—	No vdw	273	0.408	ftdock	315	0.408	ftdock
T38	UH	—	No vdw	275	0.294	ftdock	98	0.176	zdock

[a]Target type. U: unbound; B: bound; H: homology-based model.
[b]PDB code of the reference complex, used to calculate RMSD of the docking solutions. For targets T35 to T38, no structure is available for the reference complex.
[c]Parameters used in our submission to CAPRI. vdw: van der Waals included in the pyDock scoring function (weighed by 0.1); no vdw: scoring without van der Waals; RST: available information used as distance restraints; TET: used end-to-end linker distance for domain-domain docking.
[d]Results of scoring with pyDock (with or without van der Waals, as indicated).
[e]Best rank of a near-native solution (ligand RMSD < 10 Å) or the best acceptable/medium/high model as assessed by CAPRI. In brackets, rank of the best solution that we would have found if we had used restraints from available information on interface residues.
[f]Ligand RMSD of the best ranked near-native solution from the reference complex structure. If it has been assessed by CAPRI, we show the official classification: acceptable (*), medium (**), or high (***). For some cases in which complex structure is not yet available (T35 to T38), RMSD cannot be calculated. Therefore, we have evaluated our docking solutions by the fraction of the native contacts (fnat) available from the CAPRI web (http://www.ebi.ac.uk/msd-srv/capri). Based on the fnat values of the acceptable solutions for each case (for predictors and scorers), we have estimated a cutoff to define acceptable solution. For T35 and T36 (same reference complex) we have used fnat = 0.139 as the minimal value for an acceptable solution (for T35 our rank 1 solution was assessed by CAPRI as an acceptable solution). For T37 we have used fnat = 0.3 (this cutoff gives similar number of acceptable solutions as the ones known to be in uploaders; see Table III). For T38 there was not any acceptable solution with fnat officially calculated, so we chose the minimal fnat that we considered for T35 (0.139).
[g]FFT-based method that generated the best-ranked near-native solution.
[h]In our CAPRI submission, we used BASI residue Tyr87 as the only restraint ligand residue for the pyDockRST module. If we had used in addition the BASI residues Thr89, Ser93, and Glu95, which we also found as possible restraint residues (but they were not sufficiently clear to include them), we would have found a medium solution with the rank shown in brackets.
[i]T35 is a domain-domain case in which each domain monomer is built by homology.

are publicly available (http://www.ebi.ac.uk/msd-srv/capri), some of the structures are not yet released so we will not give further details on them. The calculations presented here are not only the models submitted to CAPRI, but the whole set of calculations run in strictly blind conditions with different sets of parameters. This is a good representation of the type of decisions that one faces when the parameters analyzed and optimized on a standard benchmark need to be translated to a realistic situation.

The first decision is whether to include van der Waals in the pyDock scoring function or not. We have seen in the standard benchmark that using van der Waals does not make any significant difference on the global success rates, but of course, using van der Waals could be advantageous for some specific complexes but not for others. In addition, in the CAPRI experiment, sometimes it is provided the bound conformation of one of the subunits and/or some information on putative interface residues. But none of these situations were considered in our standard benchmark. Therefore, our initial criteria for using or not van der Waals in the scoring function in real docking problems or in the CAPRI experiment is usually as follows:

1. When at least one of the subunits is given in the bound conformation, we include van der Waals term in the scoring function. The rationale is that since interface atoms for one of the molecules are in the correct conformation, near-native orientations should have fewer clashes and perhaps better van der Waals energy. However, this situation is less likely to happen in a real case scenario.
2. For the rest of targets, when at least one of the subunits needs to be built by homology, we do not use van der Waals, because near-native orientations might contain more clashes and thus van der Waals term would just introduce more noise.
3. Finally, when both proteins are given in the unbound conformation, in principle, the use of van der Waals should not make a big difference, so by default it is not used. However, sometimes we arbitrarily decide to include it (based on other available information: mutational data, ODA,[36] and NIP[37] predictions, clustering results, etc.).

We have analyzed here our results in CAPRI (both as predictors, where we have to generate and submit models, and as scorers, where we have to score the models

Table III
Results of pyDock on CAPRI Scorers Experiment (Cases with Near-native Solutions in the Uploaded Set)

					pyDock (vdw)[e]			pyDock (no vdw)[e]		
Target	Type[a]	PDB[b]	CAPRI submission[c]	Number of near-native solutions[d]	Original rank[f]	Min. rank ele/vdw truncated[g]	Min. rank ele/vdw not truncated[h]	Original rank[f]	Min. rank ele/vdw truncated[g]	Min. rank ele/vdw not truncated[h]
T25	UB	2J59	vdw	47	2	2	1	5	3	1
T26	UU	2HQS	no vdw RST	129	2	1	8	2	1	3
T27	UU	2O25	vdw	155	3	47	14	4	45	5
T29	UB	2VDU	vdw	167	3	4	6	12	3	5
T32	UU	3BX1	vdw RST	15	203(5)[i]	98(2)[i]	11 (1)[i]	194 (4)[i]	57 (1)[i]	69 (4)[i]
T35	HH	—	vdw TET	3	78	63	113	38	55	107
T37	UH	—	no vdw	76	23	36	29	106	52	25

[a]Target type. U: unbound; B: bound; H: homology-based model.
[b]PDB code of the reference complex, used to calculate RMSD of the docking solutions. For targets T35 and T37, no structure is available for the reference complex. In these cases, we have evaluated our solutions by fnat values (see Table II). For T35, we have used fnat = 0.139 as the minimal value for an acceptable solution. For T37, we have used fnat = 0.3 (see Table II).
[c]Parameters used in our submission to CAPRI. vdw: van der Waals included in the pyDock scoring function (weighed by 0.1); no vdw: scoring without van der Waals; RST: available information used as distance restraints; TET: used end-to-end linker distance for domain-domain docking.
[d]Number of near-native solutions in each uploaded set (as assessed by CAPRI organizers as acceptable, medium, or high).
[e]Results of scoring with pyDock (with or without van der Waals, as indicated).
[f]Best rank of a near-native solution (ligand RMSD < 10 Å) or the best acceptable/medium/high model as assessed by CAPRI, in the original uploaded set.
[g]Best rank of a near-native solution after minimization and scoring by pyDock with electrostatics and van der Waals (in their case) truncated as usual.
[h]Best rank of a near-native solution after minimization and scoring by pyDock with electrostatics and van der Waals (in their case) not truncated.
[i]In our CAPRI submission we used BASI residue Tyr87 as the only restraint ligand residue for the pyDockRST module. If we had used in addition the BASI residues Thr89, Ser93 and Glu95, which we also found as possible restraint residues (but they were not sufficiently clear to include them), we would have found a medium solution with the rank shown in brackets.

uploaded by other participants) to check whether these criteria are correct in completely blind conditions (Table II). For instance, from the results as predictors for targets with at least one bound subunit (T25, T29, T36), we can confirm that the inclusion of van der Waals was the right decision, especially for T25, for which it made a significant difference (rank 3 with van der Waals vs. rank 12 without van der Waals). For the other targets, the results practically did not change (for T29 we had rank 1 either with or without van der Waals; for T36 we had similar poor rank >100 either with or without van der Waals). In scorers (Table III), for T25 and T29, we had indeed better results with van der Waals (rank 2 and 3, respectively) than without it (rank 5 and 12, respectively). Target T36 is not shown in Table III because there were no near-native solutions in the uploaded set.

For the targets in which at least one of the subunits needed to be built by homology (T24, T28, T37, T38), the decision of not using van der Waals was also correct, especially for T24 (rank 20 without van der Waals vs. rank 27 with van der Waals; Table II). For the rest of the targets, the results practically did not change: for T28, we could not generate any near-native solution anyway; for T37, we had similar poor rank >100 either with or without van der Waals; for T38, we had similar poor rank either with van der Waals (rank >100) or without it (rank 98). However, in scorers (Table III), T37 had worse results without van der Waals (rank >100) than with it (rank 23), perhaps because some of the solutions in the uploaded set had been minimized by the corresponding uploading groups and were thus more appropriate for using van der Waals. Targets T24, T28, and T38 are not shown in Table III because there were no near-native solutions in the uploaded set.

In T35, we had to build interacting subunits by homology, but we have considered this case as a separate category because it was a domain–domain interacting case, for which we could use the constraint of the interdomain linker. This was done with our pyDockTET[35] protocol, which uses tethering from the expected end-to-end distance of the interdomain linker. We decided to use van der Waals for this case, in spite of involving homology-based modeling of the domains, because the top solutions (sorted by energy alone) correlated better with the expected interdomain linker end-to-end distance. Moreover, it also produced more varied docking orientations and fewer poses with impossible geometries (i.e., in which the linker would need to enter through the protein). Although the difference was not dramatic, we had better results (rank 1) with van der Waals than without it (rank 3), as seen in Table II. In scorers (Table III), we failed to identify any acceptable model for this target, but this could be due to the existence of only three near-native solutions within the uploaded set, indeed too few for pyDock to be efficient (as discussed in previous sections) and for any meaningful discussion about the use of van der Waals.

Finally, for the cases in which both subunits were unbound (T26, T27, T30, T32), the use of van der Waals was an arbitrary decision. For T26, we did not use van der Waals, and the best near-native solution

had rank 19 (compared with rank 40 with van der Waals). However, if we had used the available experimental data as distance restraints with our module pyDockRST[28] (as we did as scorers for the same target), we would have obtained an acceptable solution with rank 3 (either with or without van der Waals). For T27, we used van der Waals (because the resulting orientations seemed more consistent with other analyses, such as ODA and NIP), and the results were as poor as without van der Waals (rank >100). However, we failed again to use the available data as distance restraints. If we had used them, we would have obtained an acceptable solution with rank 5 (compared with rank 7 without van der Waals). For T30, we did not use van der Waals, and obtained slightly better results than with van der Waals, but both with rank >100. For T32 (Savinase/BASI complex), we used van der Waals because top solutions (sorted by energy alone) correlated better with the information available for this complex (BASI residue Y87 was previously reported as a possible interface residue). We also used this information to include a pseudo-energy term with the module pyDockRST,[28] based on distance restraints for this residue, obtaining a near-native solution ranked 55 (rank 53 without van der Waals). Later we realized that we could have obtained an acceptable model as rank 5 with van der Waals, or rank 1 (!) without van der Waals, by including additional residues as restraints (we found that BASI residues Thr89, Ser93, Glu95 were also previously reported as possible interface residues, but we preferred not to use them because we could not clearly evaluate their importance for the interaction). In scorers (Table III), the criterion for using van der Waals was the same as in predictors. Thus, we did not use van der Waals for target T26, though we would have obtained the same good results (rank 2) as with van der Waals. For target T27, we used van der Waals, obtaining an acceptable solution with rank 3 (rank 4 without van der Waals). For target T32, results were similarly poor with or without van der Waals (rank >100). Target 30 is not shown in Table III because the uploaded set did not have any near-native solution. In general, the choice of using or not van der Waals was the correct one, but in any case it did not dramatically alter the final results.

The first conclusion from the analysis of our predictions in CAPRI is that, in virtually all cases, we took the correct decision on whether to include van der Waals or not in the scoring function. In addition, the results indicate that we can treat homology-based docking as unbound, that is, the use of van der Waals is optional and does not drastically affect the results. But most importantly, if there is any available experimental information on the complex, it can be used not only for the final scoring in order to improve the results, but also to help to decide whether to use van der Waals or not.

Beyond rigid-body scoring: to minimize or not to minimize, that is the question

Rigid-body docking decoys have usually a significant number of interatomic clashes and impossible geometries. This is not a critical problem as these interfaces can be easily relaxed with molecular mechanics tools. We typically use TINKER to minimize rigid-body docking structures before submitting them to CAPRI (Methods). For instance, in the CAPRI scoring experiment (Table III), we typically score with pyDock the uploaded sets as they are provided (a minimal file setup is performed to rebuild missing side-chains, remove duplicated residues if any, etc.), and then we minimize the resulting 10 lowest-scoring models. Thus, the goal of this minimization is mostly to reduce the number of clashes (as defined by CAPRI), in order to generate more realistic models, otherwise they could be rejected by CAPRI on the grounds of excessive clashes. As a further test, we have minimized here all the models of the uploaded sets. For instance, as can be seen in Figure 4(A,B), the number of clashes in the best solution found in target T32 (rank 203) are dramatically reduced after minimization, from 26 to 9 clashes. Furthermore, in order to check here if the reduction of clashes after minimization can also improve the ranking of the near-native solutions, we have proceeded to score with pyDock all the minimized docking models of the uploaded sets. The scoring of minimized structures has different characteristics than that of rigid-body docking poses and constitutes a challenging problem itself. Here, we have limited our approach to adapt the pyDock function as follows. The desolvation contribution has been calculated as the difference between the ASA-based solvation energy of the minimized docking pose, minus the solvation of the original unbound subunits (we have also evaluated the effect of subtracting the solvation of the separated subunits directly taken from the minimized docking pose, with worse results; data not shown). On the other side, van der Waals and electrostatics in pyDock scoring function are truncated to allow some clashes in the rigid-body approach (Methods). Initially, we have applied here the same function to the minimized structures (Table III). But dealing with minimized conformations, perhaps we do not need the tolerance to clashes provided by truncated electrostatics and van der Waals energy terms. The results in Table III indicate that full (not truncated) electrostatics and van der Waals terms yield similar scoring to that obtained by the truncated energy terms, though it could be considered better from some points of view. For example, in target T32, the best solution found before minimization was ranked 203 with van der Waals included in the scoring, while it became rank 11 after TINKER minimization and pyDock scoring with full electrostatics and van der Waals. For target T25, the best solution before minimization (rank 2) achieved rank 1 after minimization [Fig. 4(C,D)].

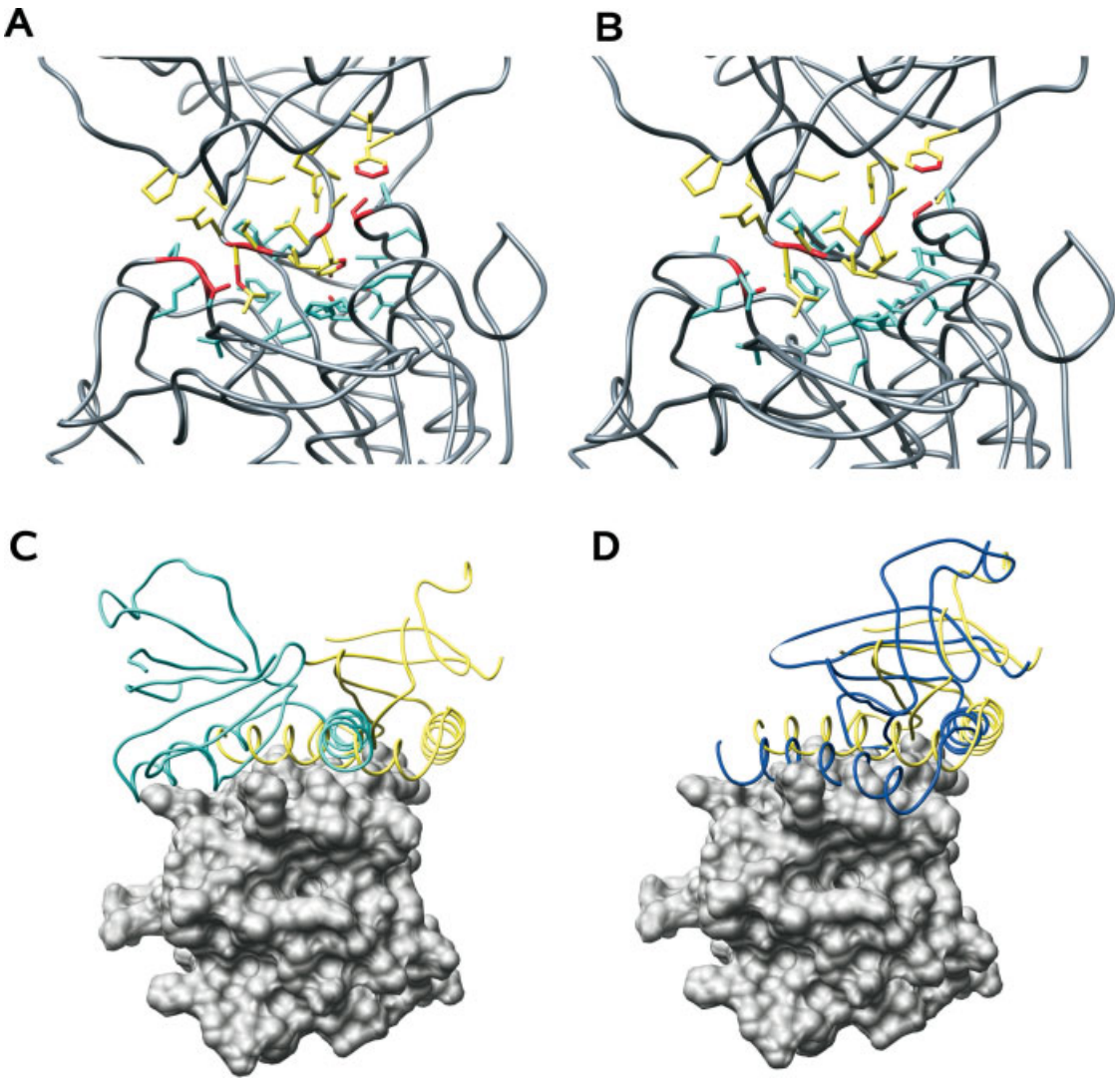

Figure 4

(**A**) Best-ranked near-native solution for scorer target T32 (rank 203, ligand RMSD 6.3 Å), with 26 clashing atoms (in red); receptor and ligand interface residues are shown in sticks (cyan and yellow, respectively). (**B**) The same near-native solution is shown after minimization (rank 11, ligand RMSD 6.3 Å), with nine clashing atoms. (**C**) pyDock (with van der Waals) rank 1 solution (ligand RMSD 30.6 Å) for scorer target T25 before minimization (predicted ligand in cyan ribbon); for comparison, the position of the ligand in the native complex is shown in yellow (PDB code 2J59). (**D**) pyDock (with van der Waals) rank 1 solution (ligand RMSD 8.0 Å) for the same target T25 after minimization.

However, on the other side, the best solution in target T27 worsened from rank 3 to rank 14, and in target T26, it moved from rank 2 to rank 8. Thus, though we observe a significant improvement after minimization in some cases, in others the advantage is not convincing. We believe more work must be done to adapt the scoring function to these new minimized, clash-free structures.

The ultimate challenge: can we estimate *a priori* the reliability of a docking case?

We have seen above that the best success rates are found in the docking cases involving small proteins and in those with low DME. Integrating these two criteria could be a good way of estimating the success of a docking case. For instance, among the cases with grid size $\leq$200 and DME $<$ 4.0 kcal/mol in our benchmark, as many as 57% of them have a near-native solution within the top 10 decoys. Moreover, with stricter DME thresholds the success rates are even better. Figure 5 shows the three cases with grid size $<$200 and DME $<$ 0.5 kcal/mol: 1TMQ, 1UDI, and 1BVN, in which a near-native solution is ranked 1, 1, and 2, respectively (success rate 100%).

These criteria also apply to our CAPRI results. In 100% of the cases with grid size $<$200 and DME $<$ 10.0 kcal/mol, that is, T25, T26, T32, and T35, we submitted

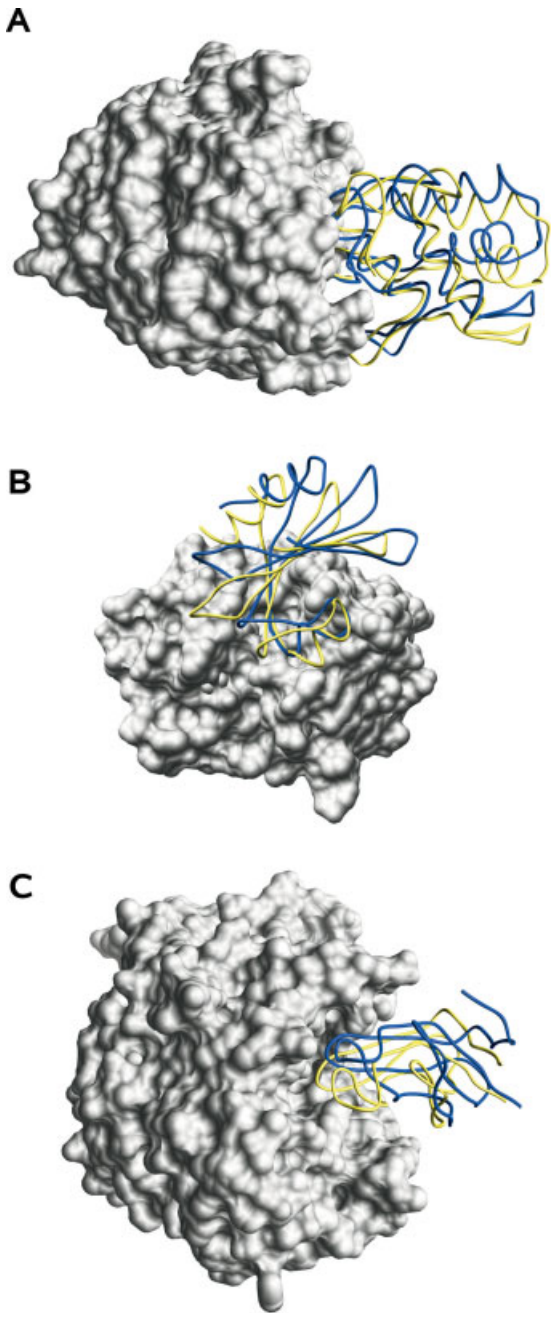

Figure 5

pyDock predictions on cases with grid size <200 and docking mean energy (DME) < 0.5 kcal/mol: (**A**) 1TMQ, rank 1 (ligand RMSD 3.5 Å); (**B**) 1UDI, rank 1 (ligand RMSD 5.6 Å); and (**C**) 1BVN, rank 2 (ligand RMSD 7.4 Å). Predicted ligand is shown in cyan ribbon; ligand in the native complex is shown in yellow ribbon.

a good model as either predictor or scorers (or would have submitted it with the appropriate restraints). Moreover, the cases with grid size >200 (T28, T30, T37, T38) had quite bad predictive results (Tables II and III). From all these analyses, we have found the above criteria to estimate the outcome of our docking predictions in realistic situations.

CONCLUSIONS

We have analyzed different conditions for FFT-based rigid-body sampling (FTDock) and energy-based scoring (pyDock) on a standard benchmark. One of the findings is that the use of electrostatics and high-resolution in FTDock sampling gave better results than our previous protocol.[15] In addition, we have seen in the blind test provided by the CAPRI experiment that it is better to use van der Waals if any of the docking subunits is in the bound conformation, while for unbound or homology-based subunits, the use of van der Waals does not make a major difference. A rule of thumb would be not to use van der Waals, unless there is available information about the interface that would permit the evaluation of the best scoring option. In general, when we include restraints in the final scoring, it seems that the use of van der Waals might be slightly advantageous. We also noticed that most of our good predictions (near-native rank <10) were generated by ZDOCK. Finally, we can improve our predictions with a minimization step, which reduces the number of clashing atoms and improves the rank of the near-native solutions. The main conclusion is that we can estimate the reliability of a given docking case according to simple parameters such as grid size and DME. Docking cases involving small proteins and showing low DME have almost 60% probability of success, while cases involving large proteins and high docking mean energy have always bad predictive results. Intermediate grid size and DME values give different estimates of docking success. Nevertheless, conformational flexibility is one of the most important determinants for docking success, and more effort should be put not only on the development of new algorithms to overcome the rigid-body docking and scoring approach but also on estimating the conformational flexibility upon binding for a given case. Protein–protein docking is still very challenging from many points of view, but knowing in advance the difficult cases to predict will save time and will help to develop new algorithmic solutions and docking tools.

ACKNOWLEDGMENTS

We would like to pay tribute to the late Dr. Angel R. Ortiz not only for his renowned scientific contributions but also for his uninterested friendship, never-ending scientific and non-scientific conversations, and for being a source of inspiration for people around.

REFERENCES

1. Smith GR, Sternberg MJE. Prediction of protein-protein interactions by docking methods. Curr Opin Struct Biol 2002;12:28–35.

2. Bonvin AMJJ. Flexible protein-protein docking. Curr Opin Struct Biol 2006;16:194–200.
3. Gray JJ. High-resolution protein-protein docking. Curr Opin Struct Biol 2006;16:183–193.
4. Ritchie DW. Recent progress and future directions in protein-protein docking. Curr Protein Pept Sci 2008;9:1–15.
5. Gabb HA, Jackson RM, Sternberg MJ. Modelling protein docking using shape complementarity, electrostatics and biochemical information. J Mol Biol 1997;272:106–120.
6. Chen R, Weng Z. Docking unbound proteins using shape complementarity, desolvation, and electrostatics. Proteins 2002;47:281–294.
7. Chen R, Li L, Weng Z. ZDOCK: an initial-stage protein-docking algorithm. Proteins 2003;52:80–87.
8. Chen R, Weng Z. A novel shape complementarity scoring function for protein-protein docking. Proteins 2003;51:397–408.
9. Mintseris J, Pierce B, Wiehe K, Anderson R, Chen R, Weng Z. Integrating statistical pair potentials into protein complex prediction. Proteins 2007;69:511–520.
10. Ritchie DW, Kemp GJ. Protein docking using spherical polar Fourier correlations. Proteins 2000;39:178–194.
11. Katchalski-Katzir E, Shariv I, Eisenstein M, Friesem AA, Aflalo C, Vakser IA. Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques. Proc Natl Acad Sci USA 1992;89:2195–2199.
12. Fernández-Recio J, Totrov M, Abagyan R. Soft protein-protein docking in internal coordinates. Protein Sci 2002;11:280–291.
13. Fernández-Recio J, Totrov M, Abagyan R. ICM-DISCO docking by global energy optimization with fully flexible side-chains. Proteins 2003;52:113–117.
14. Fernandez-Recio J, Abagyan R, Totrov M. Improving CAPRI predictions: optimized desolvation for rigid-body docking. Proteins 2005;60:308–313.
15. Cheng TM, Blundell TL, Fernández-Recio J. pyDock: electrostatics and desolvation for effective scoring of rigid-body protein-protein docking. Proteins 2007;68:503–515.
16. Grosdidier S, Pons C, Solernou A, Fernández-Recio J. Prediction and scoring of docking poses with pyDock. Proteins 2007;69:852–858.
17. Dominguez C, Boelens R, Bonvin AMJJ. HADDOCK: a protein-protein docking approach based on biochemical or biophysical information. J Am Chem Soc 2003;125:1731–1737.
18. Camacho CJ. Modeling side-chains using molecular dynamics improve recognition of binding region in CAPRI targets. Proteins 2005;60:245–251.
19. Camacho CJ, Gatchell DW. Successful discrimination of protein interactions. Proteins 2003;52:92–97.
20. Gray JJ, Moughon S, Wang C, Schueler-Furman O, Kuhlman B, Rohl CA, Baker D. Protein-protein docking with simultaneous optimization of rigid-body displacement and side-chain conformations. J Mol Biol 2003;331:281–299.
21. Zacharias M. ATTRACT: protein-protein docking in CAPRI using a reduced protein model. Proteins 2005;60:252–256.
22. Méndez R, Leplae R, De Maria L, Wodak SJ. Assessment of blind predictions of protein-protein interactions: current status of docking methods. Proteins 2003;52:51–67.
23. Méndez R, Leplae R, Lensink MF, Wodak SJ. Assessment of CAPRI predictions in rounds 3–5 shows progress in docking procedures. Proteins 2005;60:150–169.
24. Lensink MF, Méndez R, Wodak SJ. Docking and scoring protein complexes: CAPRI 3rd Edition. Proteins 2007;69:704–718.
25. Vajda S, Camacho CJ. Protein-protein docking: is the glass half-full or half-empty? Trends Biotechnol 2004;22:110–116.
26. Vajda S. Classification of protein complexes based on docking difficulty. Proteins 2005;60:176–180.
27. Kowalsman N, Eisenstein M. Inherent limitations in protein-protein docking procedures. Bioinformatics 2007;23:421–426.
28. Chelliah V, Blundell TL, Fernández-Recio J. Efficient restraints for protein-protein docking by comparison of observed amino acid substitution patterns with those predicted from local environment. J Mol Biol 2006;357:1669–1682.
29. Canutescu AA, Shelenkov AA, Dunbrack RL Jr. A graph-theory algorithm for rapid protein side-chain prediction. Protein Sci 2003; 12:2001–2014.
30. Mintseris J, Wiehe K, Pierce B, Anderson R, Chen R, Janin J, Weng Z. Protein-Protein Docking Benchmark 2.0: an update. Proteins 2005; 60:214–216.
31. Ponder JW, Richards FM. An efficient Newton-like method for molecular mechanics energy minimization of large molecules. J Comput Chem 1987;8:1016–1024.
32. Cornell WD, Cieplak P, Bayly CI, Gould IR, Merz KM, Ferguson DM, Spellmeyer DC, Fox T, Caldwell JW, Kollman PA. A second generation force field for the simulation of proteins, nucleic acids and organic molecules. J Am Chem Soc 1995;117:5179–5197.
33. Still WC, Tempczyk A, Hawley RC, Hendrickson T. Semianalytical treatment of solvation for molecular mechanics and dynamics. J Am Chem Soc 1990;16:6127–6129.
34. Desmyter A, Spinelli S, Payan F, Lauwereys M, Wyns L, Muyldermans S, Cambillau C. Three camelid VHH domains in complex with porcine pancreatic alpha-amylase. Inhibition and versatility of binding topology. J Biol Chem 2002;277:23645–23650.
35. Cheng TMK, Blundell TL, Fernández-Recio J. Structural assembly of two-domain proteins by rigid-body docking. BMC Bioinformatics 2008;9:441.
36. Fernández-Recio J, Totrov M, Skorodumov C, Abagyan R. Optimal docking area: a new method for predicting protein-protein interaction sites. Proteins 2005;58:134–143.
37. Fernández-Recio J, Totrov M, Abagyan R. Identification of protein-protein interaction sites from docking energy landscapes. J Mol Biol 2004;335:843–865.