RDPSOVina: the random drift particle swarm optimization for protein-ligand docking

Li, J., Li, C., Sun, J. & Palade, V.

Original citation & hyperlink:
Li, J, Li, C, Sun, J & Palade, V 2022, 'RDPSOVina: the random drift particle swarm optimization for protein-ligand docking', Journal of Computer-Aided Molecular Design, vol. 36, pp. 415-425.
https://doi.org/10.1007/s10822-022-00455-4

DOI 10.1007/s10822-022-00455-4
ISSN 0920–654X
ESSN 1573-4951

Publisher: Springer

RDPSOVina: The random drift particle swarm optimization for protein-ligand docking

Jinxing Li,[1] Chao Li,[1] Jun Sun[*1] and Vasile Palade[2]

1 Department of Computer Science and Technology, Jiangnan University, No.1800, Lihu Avenue, Wuxi, Jiangsu, PR

2 Centre for Computational Science and Mathematical Modelling, Coventry University, Priory Street, Coventry, UK, CV15FB

Abstract

Protein-ligand docking is of great importance to drug design, since it can predict the binding affinity between ligand and protein, and guide the synthesis direction of the lead compounds. Over the past few decades, various docking programs have been developed, some of them employing novel optimization algorithms. However, most of those methods cannot simultaneously achieve both good efficiency and accuracy. Therefore, it is worthwhile to pour the efforts into the development of a docking program with fast speed and high quality of the solutions obtained.

The research presented in this paper, based on the docking scheme of Vina, developed a novel docking program called RDPSOVina. The RDPSOVina employed a novel search algorithm but the same scoring function of Vina. It utilizes the random drift particle swarm optimization (RDPSO) algorithm as the global search algorithm, implements the local search with small probability, and applies Markov chain mutation to the particles' personal best positions in order to harvest more potential-candidates. To prove the outstanding docking performance in RDPSOVina, we performed the re-docking and cross-docking experiments on two PDBbind datasets and the Sutherland-crossdock-set, respectively. The RDPSOVina exhibited superior protein-ligand docking accuracy and better cross-docking prediction with higher operation efficiency than most of the compared methods. The developed RDPSOVina is available at https://github.com/li-jin-xing/RDPSOVina.

Keywords protein-ligand docking, random drift particle swarm optimization, AutoDock Vina

Introduction

The protein-ligand docking plays a crucial role in the structure-based drug molecule design [1]. Depending on the docking results, the designed ligand compounds can be rationally predicted whether to exert their pharmacological effects in the biological body or not. Therefore, by means of the docking programs, researchers can screen libraries of compounds to find a potential drug candidate (lead compound). However, a discovery of a new lead compound generally requires the screening of thousands of compounds, which is very time-consuming. Moreover, if the docking simulations are significantly different from the real binding situations, valuable candidates might be neglected. Hence, it is indispensable for drug development to develop a docking program that is both accurate and efficient.

Based on the given three-dimensional structure of the ligand and protein in the specified box space, protein-ligand docking can predict the binding-conformation and scoring-energy of compound ligands to the appropriate protein-target binding site. In this process, the search algorithm tunes the parameter combination of position, orientation and torsion angle in order to search for the most suitable ligand conformation binding to the protein. The scoring function serves as objective search function whose value is used to evaluate the strength of intermolecular interaction between a protein and a ligand [2]. During the past few decades, various docking software packages have been developed using novel scoring functions or search algorithms, such as Autodock4 [3], AutoDock Vina (Vina) [4], PSO@AutoDock [5], SODock [6] and FIPSDock [7]. Among these docking methods, Vina has outstanding docking performance, which organically combined the global Markov Chain Monte Carlo (MCMC) method and local Broyden-Fletcher-Goldfard-Shanno (BFGS) optimization [4].

Generally, the search algorithm will significantly affect the performance of the docking tool. Furthermore, a global search method can considerably improve the performance of a docking algorithm, since the global search method quickly locates the potential regions of full search space, and provides promising solutions for the local optimizer to further seek more optimal solutions. In this paper, we mainly concentrate on the improvement for the global search method of Vina. In the Vina global search method, Monte Carlo method was adopted for the large-scale sampling of molecule conformations, and Markov Chain (MC) mutation updates searched solutions to generate new molecular conformations.

Up to now, most development strategies for the Vina global search algorithm are based on the swarm-intelligence methods inspired by nature, such as particle swarm optimization (PSO) and grey wolf optimization (GWO). The PSO stems from the behavioral simulation of a bird flock searching for food [8]. In 2015, Tai et al. substituted the MC update method of Vina with the particles' update scheme in PSO. This improved version of Vina is called PSOVina [9]. And then, in 2016 and 2018 respectively, he induced two novel strategies into PSOVina, namely two-stage local search [10] and a chaos-embedded parameterization [11], which significantly enhanced the performance of PSOVina. The GWO simulates the hunting process of grey wolves and their social hierarchy, and it can generate a competitive optimization performance against PSO [12]. In 2020, Wong et al. applied the GWO to the global search algorithm of Vina and presented the GWOVina [13]. As a result, the GWOVina has comparable docking behavior with Vina, but 2 to 7 times faster than it. Although these two docking programs have improved the docking performance of Vina, they still have limitations regarding the docking efficiency or accuracy.

In this paper, for obtaining a superior docking performance than Vina, we utilized a novel swarm-intelligence optimization method, the random drift particle swarm optimization (RDPSO), as the global search method to adjust the molecular conformations. Furthermore, the MC mutation of Vina is also implemented in RDPSOVina to further improve the search performance and yield more potential-candidates. The newly developed program still employed the scoring function of Vina, called RDPSOVina. Evaluated on two datasets of re-docking and a cross-docking dataset, the experimental results indicate that the RDPSOVina exhibits an outstanding docking accuracy with remarkable running efficiency compared to other methods.

Methods

Particle Swarm Optimization

Particle swarm optimization (PSO) simulates the social behavior of bird flocks during searching for food in an area [14]. In a swarm with M particles, every particle has N dimensions, a certain velocity $V_{i,n} = (V_{i,n}^1, V_{i,n}^2, ..., V_{i,n}^N)$ and a current position $X_{i,n} = (X_{i,n}^1, X_{i,n}^2, ..., X_{i,n}^N)$ that represents a potential solution of search problem (i and j subject to $1 \le i \le M$ and $1 \le j \le N$, respectively). The best solution located by the ith particle so far, named the personal best (pbest) position $P_{i,n} = (P_{i,n}^1, P_{i,n}^2, ..., P_{i,n}^N)$, and the best of all the pbest positions is called as the global best (gbest) position $G_{i,n} = (G_{i,n}^1, G_{i,n}^2, ..., G_{i,n}^N)$. Guided by the pbest and gbest positions, particles are constantly updated the current positions to find better solutions, expressed as

$$\begin{aligned} V_{i,n+1}^j = \omega \cdot V_{i,n}^j + c_1 \cdot r1_{i,n}^j \cdot (P_{i,n}^j - X_{i,n}^j) \\ + c_2 \cdot r2_{i,n}^j \cdot (G_n^j - X_{i,n}^j) \end{aligned} \tag{1}$$

$$X_{i,n+1}^j = X_{i,n}^j + V_{i,n+1}^j \tag{2}$$

where ω is an inertia weight proposed by Shi and Eberhart [8]. $r1_{i,n}^j$ and $r2_{i,n}^j$ are random numbers from 0 to 1. c_1 and c_2 are acceleration factors and used to adjust the particle's moving distance to $P_{i,n}$ and G_n, respectively. In the canonical PSO, there are some drawbacks. When $P_{i,n}$ and G_n are very close, particles might be trapped in a local optimum. What's more, if $P_{i,n}$ and G_n locate the opposite directions of a particle's current position, the particle may oscillate between $P_{i,n}$ and G_n, and cause hard convergence to the whole swarm. In order to deal with these problems, researchers combined $P_{i,n}$ and G_n as a novel attractor $p_{i,n}$ and developed the update equation of particle's velocity [15]. The developed PSO is illustrated as:

$$p_{i,n}^j = \frac{c_1 \cdot r1_{i,n}^j \cdot P_{i,n}^j + c_2 \cdot r2_{i,n}^j G_n^j}{c_1 \cdot r1_{i,n}^j + c_2 \cdot r2_{i,n}^j} \tag{3}$$

$$V_{i,n+1}^j = \omega \cdot V_{i,n}^j + (c_1 + c_2) \cdot R_{i,n}^j \cdot (p_{i,n}^j - X_{i,n}^j) \tag{4}$$

$$X_{i,n+1}^j = X_{i,n}^j + V_{i,n+1}^j \tag{5}$$

Random Drift Particle Swarm Optimization

The random drift particle swarm optimization (RDPSO) assumed that the particle's behavior is similar to an electron moving in a metal conductor within an external electric field [16]. The electrons in this condition are deemed to have random thermal motion influenced by entropy and drift motion driven by external electric field force. Thus, the update of particle velocity become the superposition of the thermal motions and the drift motions corresponding to the random velocity ($VR_{i,n+1}^{j}$) and the drift velocity ($VD_{i,n+1}^{j}$), respectively.

$$V_{i,n+1}^{j} = VR_{i,n+1}^{j} + VD_{i,n+1}^{j} \tag{6}$$

It is assumed that random thermal velocity obeys the Maxwell velocity distribution law, the $VR_{i,n+1}^{j}$ can be expressed as

$$VR_{i,n+1}^{j} = \alpha \left| C_{n}^{j} - X_{i,n}^{j} \right| \varphi_{i,n+1}^{j} \tag{7}$$

where $\alpha > 0$ is the thermal coefficient and C_{n}^{j} is the component in the jth dimension of the mean best (mbest) position $C_{n} = (C_{n}^{1}, C_{n}^{2}, ..., C_{n}^{N})$, defined as the mean of all the pbest positions. $\varphi_{i,n+1}^{j}$ is a random number that is subject to the standardized normal distribution. The drift movement is the directional movement towards an attractor $p_{i,n} = (p_{i,n}^{1}, p_{i,n}^{2}, ..., p_{i,n}^{N})$, described as

$$VD_{i,n+1}^{j} = \beta \left(p_{i,n}^{j} - X_{i,n}^{j} \right) \tag{8}$$

where $\beta > 0$ is the drift coefficient. Eventually, the update equation of particles is expressed as

$$V_{i,n+1}^{j} = \alpha \left| C_{n}^{j} - X_{i,n}^{j} \right| \phi_{i,n+1}^{j} + \beta \left(p_{i,n}^{j} - X_{i,n}^{j} \right) \tag{9}$$

It is recommended that α linearly decreases from 0.9 to 0.3 and β is specified as 1.45 [16]. The main difference of the update equation between RDPSO and the developed PSO is that RDPSO substitutes the inertia part of PSO with the random thermal motivation [17]. It will make the RDPSO have better search performance than PSO, and the main reason is that the thermal motivation has greater randomness, providing more opportunity for particles to escape the local optima in the later stage of the search process.

The Search Algorithm Used in RDPSOVina

The most time-consuming component in Vina is the local search, namely, Broyden-Fletcher-Goldfard-Shanno (BFGS) [18], which iteratively calculates an approximation to the Hessian matrix to determine the descent direction. In order to save the computational time, in RDPSOVina, we adopted a strategy of applying the BFGS with a small probability to each particle. To be specific, in each iteration, each particle has 6% probability of using both RDPSO and BFGS to update the current positions, while for the remaining 94% probability, it merely utilizes the RDPSO to update its position. The small probability is set as 6%, which is built upon the same setting of the genetic algorithm in Autodock4. In our previous experiments, the 6% is the most appropriate value, and a bigger or smaller value makes a poor docking performance. Moreover, comparing with the two-stage strategy used in the PSOVina [10], this small probability strategy can more effectively decrease the time consumption of the local search. In PSOVina, a rough local search is utilized to determine the feasibility of solutions as the first stage, and then BFGS as the second stage is implemented on the potential ones. In fact, the rough local search is a partial execution of BFGS that decreases the total iteration of BFGS by 10 times. Another noteworthy difference of algorithmic implementations between the RDPSOVina and the PSOVina is that the results of the local search, in RDPSOVina, are updated to the current positions and pbest positions, while in the PSOVina, are only able to be substituted for the pbest positions. The utilized strategy in our RDPSOVina will not lead to a large fitness gap between current positions and pbest's like PSOVina does, so that particles can search for the better solutions in suitable areas.

During the search process of RDPSOVina, if the gbest position is not updated for a long time, it will be difficult for the algorithm to converge to a remarkable docking solution. To solve this problem, when the successive steps without updating gbest are more than 100 steps, we employ the MC mutation for updating the best three pbest positions as a substitute for the RPDSO execution. Additionally, to obtain as good a final output as possible, the best three pbest positions are forced to further apply MC mutation at the later stage of search process (over 90% of the total iterations).

At the end of each algorithmic iteration [4,11,13], a specific searched solution is selected according to the Metropolis principle to put into an output container (Metropolis-sampling step) for further processing. In RDPSOVina, the specific candidate is a random pbest position chosen for further employing the MC mutation before the Metropolis-sampling step. This sampling process is different from that applied in PSOVina, in which only the gbest position in each iteration is used as the input for the Metropolis-sampling step. Obviously, our proposed sampling process can generally generate more diverse solutions in the output container than that used in PSOVina, and can thus output more conformations when the program is terminated.

The termination condition of the proposed search algorithm is when it reaches the maximum number of iterations. In RDPSOVina, the maximum number of iterations is specified as 6% of total step number in Vina. Using the 6% is the result of comprehensive consideration of algorithmic convergence property and docking time efficiency. Moreover, the 6% can be changed with a newly added parameter called ratio_steps.

Table 1 The search algorithm of RDPSOVina

```
Algorithmic procedure
Begin
  Initialize current position $X_{i,0}$ of swarm particles, pbest position $P_{i,0}$
  and gbest position $G_0$. Evaluate the fitness $Y_{i,0}$ of current position $X_{i,0}$. C
  counts the steps for no updating gbest position;
  n=0
  while n<max_step do
    if (phrase=1)
      for i=1 to M do
        if (rand (0,1) < 0.06) // 6% particles to use BFGS
          full BFGS ($X_{i,n}$);
          update $X_{i,n}$, $Y_{i,n}$;
        else
          $Y_{i,n}$=score($X_{i,n}$); // score(.) is scoring function
        end if
        update $P_{i,n+1}$ and $G_{n+1}$;
        update $X_{i,n+1}$ by RDPSO;
      end for
    end if
    if (phrase=2)
        pick-top-three ($P_{j,n}$); // use the top three $P_{j,n}$
        markov mutation ($P_{j,n}$);
        two-stage BFGS ($P_{j,n}$);
        update $P_{j,n+1}$, $G_{n+1}$;
    end if
    m=rand-int (1, m); // m is random integer from 1 to M
    if ($P_{m,n+1}$ satisfied metropolis principle)
        further improve $P_{m,n+1}$ and output it;
    end if
    if (C<3 and phrase=2)
        phrase=1;
    end if
    if ((C>100 and phrase=1) or n>0.9*max_steps)
        phrase=2;
    end if
    if (score($G_{n+1}$)–score($G_n$)<0.0001)
        C=C+1;
    else
        C=0
    end if
    n=n+1;
  end while
End
```

Scoring Function

The scoring function in Vina is also employed in our proposed program. Vina employs a semi-empirical scoring function inspired by the X-score function [19], used to score the binding affinity between a protein and a ligand. The semi-empirical scoring function can be regarded as the combination of intermolecular (c_{inter}) and intramolecular (c_{intra}) contributions. In Vina, it is calculated as the summation of distance-dependent interactions $f_{t_i t_j}(r_{ij})$, shown as:

$$c{=}c_{\text{inter}}+c_{\text{intra}}=\sum_{i<j} f_{t_i t_j}(r_{ij}) \tag{10}$$

where c is the object function of search algorithm and r_{ij} is the interatomic distance. Moreover, the value of c_{inter} is the final energy output in Vina, namely binding energy, and it is calculated by the following:

$$c_{\text{inter}}=\sum_{i<j} h_{t_i t_j}(d_{ij}) \tag{11}$$

where d_{ij} is the surface distance defined as $d_{ij}=r_{ij}-R_i-R_j$. R_i is the van der Waal radius of the ith atom. $h_{t_i t_j}(d_{ij})$ is the interatomic interaction and involves polar interaction, nonpolar-nonpolar contacts, repulsion forces, hydrogen bonds, solvation interactions and so on [2], calculated as:

$$h_{t_i t_j}(d_{ij})=\left\{\begin{array}{c} w_1\cdot \mathrm{Gauss}_1(d_{ij})+ \\ w_2\cdot \mathrm{Gauss}_2(d_{ij})+ \\ w_3\cdot \mathit{Repulsion}(d_{ij})+ \\ w_4\cdot \mathit{Hydrophobic}(d_{ij})+ \\ w_5\cdot \mathit{HBond}(d_{ij}) \end{array}\right. \tag{12}$$

As in equation (12), the outcome of $h_{t_i t_j}(d_{ij})$ in Vina is the accumulation of five component items, and each item is significantly weighted according to its contributions for the protein-ligand binding interaction. The first three given items compose the steric interactions, and the existence of the fourth hydrophobic and the fifth hydrogen interactions depend on the atom types [20].

Datasets and Experimental Setups

To effectively estimate the docking performance, two basic docking patterns are commonly used: re-docking and cross-docking. Re-docking represents the ability to reproduce the position, orientation and geometry of the associated ligand binding to a receptor, and the ligand and receptor are separated from the same given complex. Cross-docking refers to the prediction capability for crossed docking cases between ligands and proteins isolated from multiple co-crystallized complexes of the same protein.

To evaluate the performance for re-docking, we used two datasets of PDBbind and they can be downloaded from http://www.PDBbind-cn.org. The first one is PDBbind core set v.2016, which is included in CASF-2016 [21]. The core set provides 285 groups of structural information of protein-ligand complexes, which are the crystal structures with the resolution of less than 0.25nm measured by the real experiments. The second one is PDBbind refined set v.2020. The refined set consists of 5316 protein-ligand complexes. But, one complex called 4bps cannot generate its protein structural file, so 5315 test cases were used in our experiments.

Our re-docking experiments were developed in the following three steps: prepare the proteins and ligands, generate the configuration files, and execute docking simulations. The PDBQT files of protein and ligand were attained through the prepare_protein4.py and prepare_ligand4.py in AutoDock Tools. The involved parameters for docking were specified as follows: the box shape was a cubic with the size of 22.5Å×22.5Å×22.5Å; the box center was set as the geometric center of the crystal ligand; the exhaustiveness parameter was set to 8; the number of particles for each exhaustiveness was 12 and thus the size of particle population was 96; CPU number was 10. The other parameters were all default. Our docking experiments were carried out on a multiple-nodes cluster server with 240 Intel(R) Xeon(R) E5–2620 CPU core 2.40 GHz processors based on a Centos 7.6.1810 Linux platform.

In the cross-docking experiments, we utilized the 8 protein-target families of CDK2, ESR1, F2, MAPK14, MMP8, MMP13, PDE4B and PDE5A in Sutherland-crossdock-set [22]. All the testing items of them are listed in the appendix reported by Dr. Jeff Sutherland [22]. Considering the relatively limited power of computation, we picked 30 items each for CDK2 (82 structures) and F2 (72 structures) and employed all the items available in the Protein Data Bank [23] for the rest of the families (<30 structures).

In the preparatory phase of cross-docking materials, we used the PyMOL to align all the complexes and separate the ligand and receptor from each complex, added the hydrogens to the ligands and receptors due to their lack of most hydrogens, retained the metal ions belonging to the binding pocket (in MMP8, MMP13, PDE4B and PDE5A) and removed other components not belonging to the pocket, such as water and other irrelevant solvent.

Evaluation Criteria

The common indicator for docking accuracy is the standard root-mean-square deviation (RMSD), which calculates the difference between predicted and co-crystallized structure. Generally, when the difference is less than 2Å, the obtained conformation is considered as a success. In the re-docking comparison, for each test case of PDBbind core set, we implemented 10 groups of repetitive docking experiments and calculated their average RMSD. The PDBbind refined set has too many test cases, so the docking test was only implemented once to save time. Besides, the calculated errors of RMSDs was estimated by 2000 rounds of bootstrapping computations [24]. To demonstrate whether there is a significant difference between two docking methods, the Wilcoxon signed-rank test [25] was utilized, and it was undertaken at a 5% level of significance in this paper.

For the cross-docking target families, cross-docking results are categorized as three clusters: docking success, scoring failure and sampling failure [7]. If the corresponding best-scored conformation is successfully docked, the docking program gets a docking success. A scoring failure signifies that all the successful docking structures found by a docking method are not the best-scored ones. When the RMSDs of ligand conformations obtained by a docking method are all more than 2Å, this docking result is considered as a sampling failure.

Results and Discussion

Re-Docking Results of PDBbind Core Set in Terms of Binding Energy, Accuracy, Efficiency and Stability.

As mentioned above, five individual items compose the scoring function of Vina, and the total result of their contributions is the binding energy. Table 2 shows each component contribution (for binding energy) and the binding energy, in terms of docking prediction conformations and the crystalline ligand. Vina had the best binding energy, followed by GWOVina and RDPSOVina, and then PSOVina. When it comes to the component contributions, except for the contribution of the repulsion item, the compared docking programs yielded the same order as the binding energy, that is, RDPSOVina surpassed PSOVina but followed behand Vina and GWOVina.

It is noteworthy that using the docking tools is aimed at obtaining the real simulations for crystalline ligand structure, so the docking energy should be as close as possible to the crystal energy. However, the total energy of ligand crystallization was far behind the energies of docking results, since most component energies of the crystal ligand were all worse than those of the docking results. It is mainly caused by two aspects. On the one hand, the scoring function in Vina was significantly empirically weighted, which most likely makes the crystal structure without a minimum energy score. On the other hand, no matter whether the scoring function is appropriate or not, the search algorithm always struggles for a minimum result of the objective function and generates as low energy scores as possible.

According to the listed energies in Table 2, the search capability ranking of the global methods was MCMC > GWO > RDPSO > PSO. Although the RDPSO found worse binding energy than GWO and MCMC, it does not mean that RDPSOVina will generate worse docking accuracy than Vina and PSOVina, since the crystalline ligand has the worst binding energy among the compared results.

Table 2 The component and binding energies of docking predictions and crystalline ligand for 285 test cases. (kcal/mol)

	Vina	PSO	GWO	RDPSO	Crystalli
		Vina	Vina	Vina	zation
Gauss1	-2.497	-2.361	-2.496	-2.480	-2.414
Gauss2	-5.045	-4.942	-5.044	-5.042	-4.936
Repulsion	1.548	1.430	1.550	1.529	2.267
Hydrophobic	-1.072	-1.040	-1.069	-1.065	-0.977
Hydrogen	-1.614	-1.471	-1.607	-1.600	-1.685
Energy	**-8.679**	**-8.384**	**-8.667**	**-8.657**	**-7.745**

Table 3 Statistical results of the best-Scored RMSD obtained by all compared docking programs for 285 test cases.

	Mean RMSD (Å)	Mean Succ [a]	P-value [b]	Mean Time (s)
Vina	2.52±0.32	64.35%	$9.7e^{-7}$	39.77
PSOVina	2.57±0.34	63.30%	0.0016	4.33
GWOVina	2.41±0.32	65.71%	0.0036	8.25
RDPSOVina	**2.27±0.31**	**68.24%**	--	**6.35**

[a] The mean success rate is the ratio of the number of successful dockings divided by the total number of docking executions.

[b] P-value is calculated by comparing two sets of mean RMSD results obtained by the RDPSOVina and another docking method, respectively.

Table 3 illustrates the overall accuracy obtained by all compared docking programs. The Mean RMSD and the mean success rate of docking indicate that the RDPSOVina was able to achieve the most accurate docking prediction, followed by GWOVina and Vina, and finally PSOVina. The P-values show that the small difference in precision between RDPSOVina and other docking programs (an increase of 3.89%, 4.94% or 2.52%) is statistically significant. To further analyze the docking accuracy of the predicted conformations obtained by all compared methods, we divided the best-scored conformations of 285 test cases into six RMSD regions of 0.0Å - 0.5Å, 0.5Å - 1.0Å, 1.0Å - 1.5Å, 1.5Å - 2.0Å, 2.0Å - 3.0Å and > 3.0Å, and counted the number of test cases in each classified region, as displayed in Fig. 1. According to Fig. 1, RDPSOVina, Vina, GWOVina and PSOVina had the largest number of test cases on the RMSD intervals of 0Å-0.5Å, 0.5Å-1.0Å, 1.0Å-1.5Å and >2.0Å, respectively. Among all the compared docking programs, the RDPSOVina concentrated on the smallest RMSD region of 0Å-0.5Å and had the least docking failures of more than 2Å. Therefore, it can be summarized that the RDPSOVina can obtain the closest conformation prediction to crystallization.

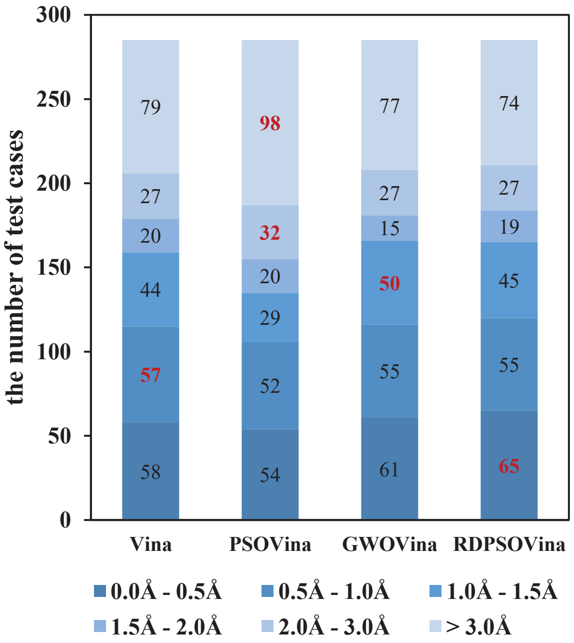


Fig. 1 Cumulative histograms of the best-scored RMSD obtained by Vina, PSOVina, GWOVina and RDPSOVina

As for the docking efficiency, take the popular Vina as the reference. RDPSOVina spent 6.35s at six-folds acceleration than Vina, while PSOVina and GWOVina were around 9 times and 5 times faster than Vina, respectively, according to the mean running time listed in the last column of Table 3. As mentioned above, the total number of iterations in RDPSOVina is 6% of that in Vina. Unlike the RDPSOVina, the search procedure of PSOVina is terminated only when it shows convergence (the gbest position has stopped updating for 350 iterations) or reaches the total steps of Vina [9]. In most cases, PSOVina stops early, and thus the number of running iterations in PSOVina is generally smaller than the total iterative number of RDPSOVina. Therefore, compared to the PSOVina, although the RDPSOVina consumes less time on the local search stage, the larger number of running iterations in RDPSOVina still leads to the slightly longer docking time, as shown by the time results in Table 3.

Subsequently, we focused on the performance of each docking method for the test cases with different number of torsions (i.e., different search dimensions), as shown in Fig. 2. The histograms in Fig. 2 illustrate the average RMSD of best-scored structures for different number of torsions. Except the classes with 1, 2, 5 and 6 torsions, RDPSOVina had the best docking accuracy when

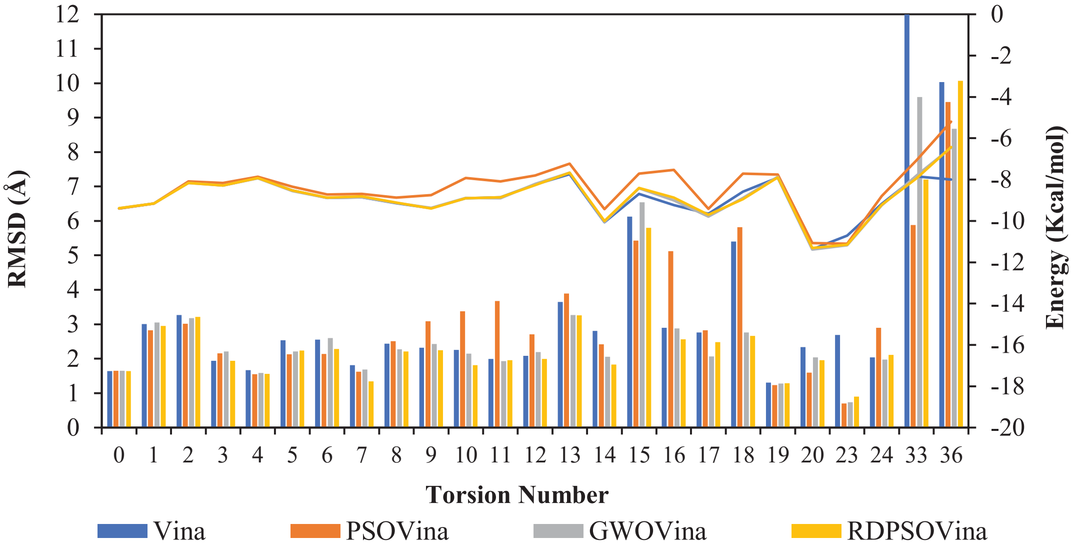


Fig. 2 Mean RMSD and mean energy comparison based on different number of torsions. Left axis illustrates the y-coordinate of the histograms, and the y-axis of the polylines corresponds to the right axis.

the number of torsions is less than 14. When the number of torsions is larger than 14, there only are 22 test cases for the docking evaluation. Owing that the randomness of search algorithm can make the docking results uncertain, the relatively small number of test cases might lead to lose the statistical significance for the docking evaluation. Nevertheless, with respect to most cases of larger than 14 torsions, the RDPSOVina obtained the best or second best result among all the compared docking methods. Overall, RDPSOVina has obtained stably better docking performance for the test cases with different number of torsions than the other compared approaches.

Docking Performance with Different Sizes of Optimization Population and Search Boxes.

In swarm intelligence optimization, the population size can considerably affect the search accuracy and efficiency. To evaluate the docking performance with different size of population, we conducted one group of docking experiment on PDBbind core set using PSOVina, GWOVina and RDPSOVina. According to Fig. 3, the RDPSOVina obviously performed better than PSOVina when the population size was less than 256, but showed comparable docking accuracy to PSOVina when the number of particles was larger than 192. GWOVina obtained worse docking accuracy than RDPSOVina, irrespective of the number of particles used. Overall, the RDPSOVina is more robust than PSOVina and GWOVina under the different population size conditions.

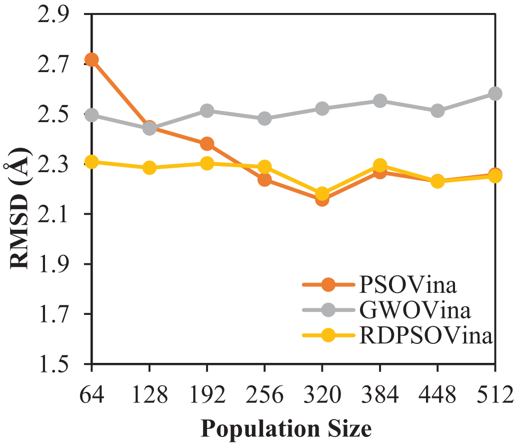


Fig. 3 The accuracy comparison of docking programs with different population size.

The size of search space is also an important factor of influencing the algorithmic search accuracy. To detect the algorithmic performance with different sized search space, the docking programs were implemented in three cubic boxes with sides of 22.5 Å, 27 Å and 36 Å. The setting of 22.5 Å is default. When the side length exceeds 27 Å, the Vina and Vina-based programs will throw a warning that the box setting is too large. The 36 Å refers to the biggest cubic setting in the Feinstein's experiments that were conducted to find an optimal box size [26]. Fig. 4 shows the docking accuracy of all the compared programs with three sizes of cubic boxes. In each box with a certain size, the RDPSOVina can obtain the better RMSD results than the other compared docking programs. It indicates that the RDPSOVina can obtain the best docking accuracy among the compared docking programs. Observing the RMSD fluctuations of the same docking method in different cubic boxes, we can conclude the docking stability of a program with enlarging the box size. The PSOVina had the worst stability of docking accuracy among all the compared programs when the box size was increased. The RDPSOVina and GWOVina has the similar docking stability, and a little better than Vina's. To sum up, the RDPSO can obtain stable and excellent docking results than other compared methods, when expanding the seach space.

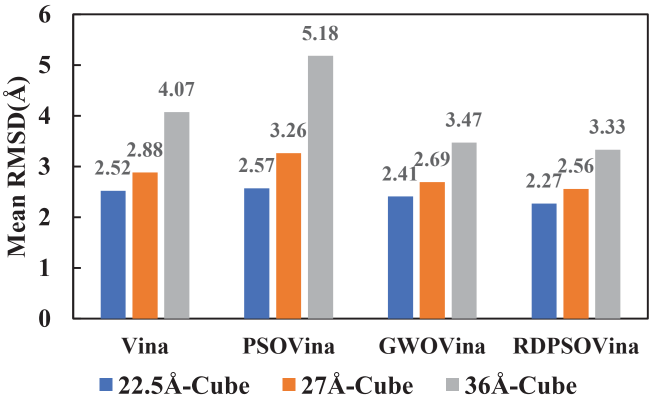


Fig. 4 The accuracy comparison of docking programs with different sizes of search boxes.

Re-Docking Results of PDBbind Refined Set v.2020 in Terms of Accuracy, Energy, Efficiency and Stability.

Table 4 Statistical results of all the compared docking programs for PDBbind Refined set v.2020.

	Mean RMSD (Å)	Mean Succ	Mean Energy[a]	Mean Time (s)	Throughput [b]
Vina	2.95	56.52%	**-8.326**	28.4	304
PSOVina	2.92	55.88%	-7.965	5.7	1516
GWOVina	2.85	57.38%	-8.296	11.1	778
RDPSOVina	**2.63**	**60.55%**	-8.299	6.5	1330
RDPSOVina1	**2.66**	**59.76%**	-8.293	5.7	1516

[a] the mean value of best-scored energy for 5315 test cases.
[b] the expected throughput of structures/day on a single cpu.

The re-docking performances of different programs were also tested on another PDBbind dataset, i.e., refined set v.2020, in which there are 5315 test cases used. All the tested experiments were employed on 10 CPUs and the compared results are illustrated in Table 4. With regards to mean RMSD and mean success rate, the RDPSOVina performed the best, followed by GWOVina, and finally PSOVina and Vina. In term of mean energy, the RDPSOVina performed only inferior to Vina, and better than the GWOVina and PSOVina. According to the mean running time, the RDPSOVina run was only slower to PSOVina, but faster than GWOVina and Vina. These compared performances accord with the results tested on PDBbind core set. According to the mean time of running a test case, we can calculate the expected quantity (throughput) of molecular structures output by each docking program in one day on a single cpu. The last column of Table 4 gives the expected throughput of docking programs, when 10 CPUs were in parallel. The throughput of PSOVina and RDPSOVina are significantly better than the other docking programs. The RDPSVina can output 1330 structures on a single CPU in one day, and using our settings of 10 CPUs, so it can give about 13300 structures in one day .

In the results Table 3 and Table 4, the PSOVina used less docking time and obtained worse accuracy than the RDPSOVina. Thus, it's hard to elucidate that the RDPSO has better global search capability than PSO in the docking task. In order to explain the problem, the comparison of the docking performance need to be employed with the same computation time. Therefore, we decreased the total number of iterations of the RDPSO, and ensured that the RDPSOVina have the same computation time as PSOVina. The experimental results are illustrated as the last row of Table 5. The RDPSOVina with decreased iterations (RDPSOVina1) achieved lower RMSD, higher success rate and better energy than the PSOVina, GWOVina and Vina. Thus, a conclusion can be drawn that the RDPSO had better global search ability than the PSO in terms of protein-ligand docking in Vina.

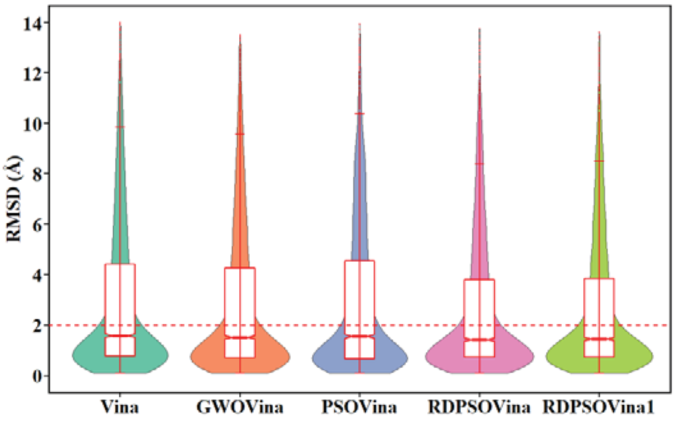


Fig. 5 The box-and-whisker and violin plots for the average RMSD of 5315 test cases.

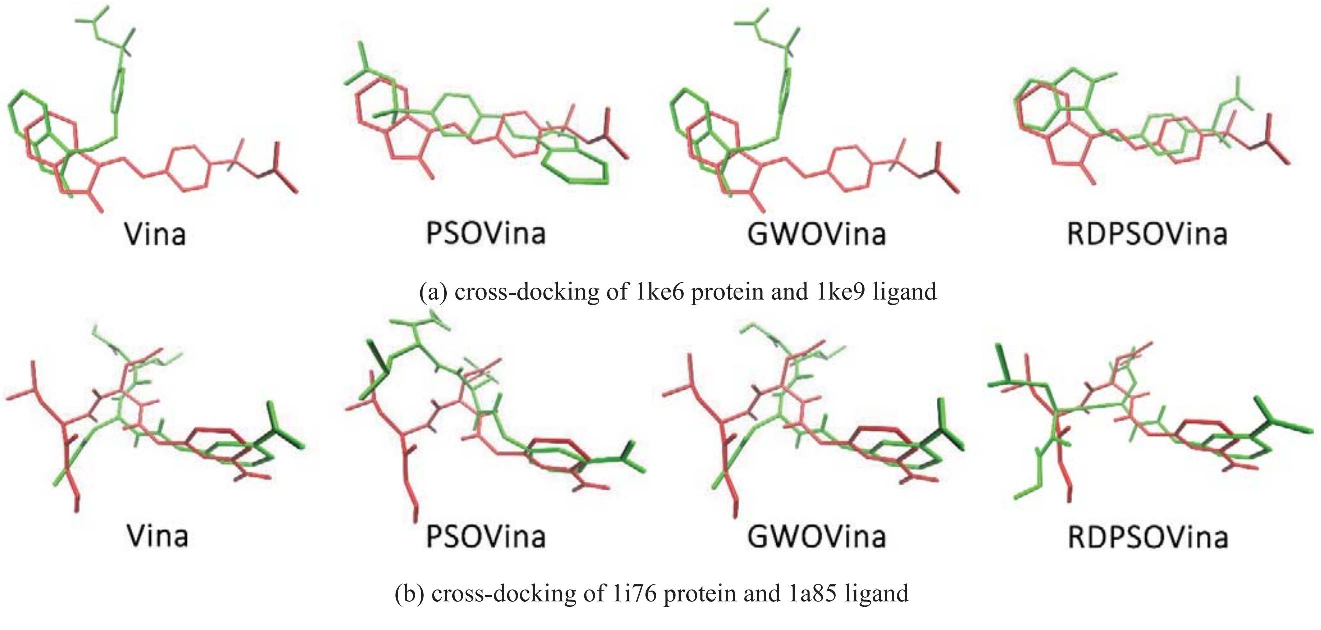
The Fig. 5 illustrates the RMSD distribution of tested results on PDBbind refined set, which indicates the docking stability of program. The more concentrated the RMSD distribution of the docking results is, the more stably the corresponding docking method performs. In Fig. 5, the boxes and the whiskers illustrate the overall RMSD distribution of docking results. From the violin plots of Fig.5, the distribution detail of 5315 best-scored RMSD can be observed. In the RDPSOVina, the docking effectiveness of using full iterations were slightly better than, but almost the same as that using decreased iterations, illustrated as the results of Table 4. The above docking expression of RDPSOVina also reflected in the plots of RDPSOVina and RDPSOVina1 in Fig. 5. Compared with other docking programs, the RDPSOVina obtained more concentrated distribution of RMSD results, demonstrated by the shorter box height and whisker length. GWOVina exhibited the comparable distribution to Vina, which can be reflected from their similar boxes' and violins' shapes. The box plot of PSOVina had the longest height among all the docking programs, which means that the docking stability of PSOVina is worse than that of the other methods. Moreover, through the Fig. 5, the docking accuracy of program can also be expressed. In Fig. 5, a horizontal line of 2 Å is used to divide the violins into two parts. The width on a certain RMSD for a violin plot represents the occurrence frequency of the corresponding RMSD values in all RMSD results. Below this 2 Å line, the largest violin area of RDPSOVina means that it obtained the largest number of docking success among all the compared approaches. On the other hand, the violin area above the 2 Å line of PSOVina was larger than those of the others, which means the PSOVina obtained the most docking failure cases.

Overall, the RDPSOVina had the outstanding docking accuracy, excellent search ability for an optimal energy, high running efficiency and remarkable docking stability. From the algorithmic perspective, there are some reasons why the RDPSOVina can generate these superior docking performances. First of all, the RDPSO has better convergence than the PSO and MCMC, which will give RDPSOVina high efficiency. Compared with the RDPSOVina, the faster speed in PSOVina attributes to its setting of algorithmic early termination. Secondly, in the RDPSOVina, we also use markov mutation to improve the search ability of algorithm and ensure the gbest position to be updated as soon as possible. Thirdly, in each iteration, the particles of using BFGS typically stay at better positions than the rest partcles, this will benefit for the diversity development of particle population.

Comparison Results of Cross-Docking Accuracy.

The statistical results of cross-docking are displayed in Table 5. Obviously, PSOVina and RDPSOVina were better than Vina and GWOVina in terms of the cross-docking performance, which indicates that PSO-based search methods are more suitable for handling cross-docking tasks than MCMC and GWO. The PSOVina exhibited the remarkable accuracy of cross-docking on the relatively easy cross-docking targets, namely ESR1, PDE4B and PDE5A. For the remaining relatively complicated cross-docking tasks, the RDPSOVina was able to obtain better cross-docking performance than the PSOVina. Overall, the RDPSOVina has more advantages than the other docking methods in dealing with the cross-docking problems, which can be concluded from the average values of success rate for all test case in Table5.

It should be point out that cross-docking can be utilized to find the best holo structure of a target protein. As illustrated by Shamsara [27], the protein structure with larger number of docking success has a higher probability to obtain the better



(a) cross-docking of 1ke6 protein and 1ke9 ligand

(b) cross-docking of 1i76 protein and 1a85 ligand

Fig. 6 Docking pose comparison based on different docking programs. The red-colored structure is the co-crystallized structure of the ligand.

performance for pose prediction and virtual screening when it is used as the docking receptor. In the Figure 1 of Support Information, according to the comparison of four rightmost numbers for the same receptor obtained by four docking methods, it can be demonstrated that the best holo protein structures for CDK2, ESR1, F2, MAPK14, MMP8, MMP13, PDE4B and PDE5A are the receptors from the crystalized complexes 1ke6, 1qkt, 1ype, 1ouy, 1i76, 1xud, 1xm4 and 1udt, respectively.

Table 5. The docking success rate obtained by Vina and Vina-based programs for each protein family.

Protein Family	Vina	PSOVina	GWOVina	RDPSOVina
CDK2	12.22%	13.44%	12.22%	**13.78%**
ESR1	40.70%	**45.45%**	41.25%	42.15%
F2	31.89%	27.78%	32.89%	**34.78%**
MAPK14	18.56%	20.64%	19.04%	**20.80%**
MMP8	23.98%	27.04%	26.53%	**30.10%**
MMP13	22.45%	26.53%	22.45%	**30.61%**
PDE4B	25.44%	**31.95%**	25.44%	27.22%
PDE5A	31.25%	**46.88%**	37.50%	37.50%
Average [a]	24.53%	25.68%	25.36%	**27.02%**

[a] The mean success rate averaged over all the test cases of 8 protein families.

To further compare the effectiveness of different cross-docking methods, we chose the best holo proteins of both CDK2 and MMP8 (i.e., 1ke6 and 1i76) and visualized their cross-docking results with another ligand (1ke9 and 1a85), as illustrated in Fig. 6. With respect to Fig. 6(a), RDPSOVina yielded the best docking of 2.78 Å; Vina and GWOVina obtained the similar binding pose of 7.37 Å and 7.29 Å; PSOVina reversed the geometry of crystallized ligand and generated the worst RMSD of 9.05 Å. With regards to the example on MMP8, the docked results of Vina, PSOVina and GWOVina were very different from the crystallization structure. The RDPSOVina obtained the most similar conformation to the crystallization. The afore-mentioned analysis indicates that the RDPSOVina can obtain better performance for the cross-docking tasks than other docking methods.

Conclusions

Protein-ligand docking plays an essential role in modern drug development. To further enhance the docking performance, in this study, we developed a docking program with a novel search algorithm, named RDPSOVina. The RDPSOVina utilizes the RDPSO as the global search algorithm, it implements the BFGS with a 6% probability to save docking time, and it applies MC mutation to harvest more potential-candidates. Our experimental results revealed that the RDPSOVina obtains the best accuracy and stability among all the compared docking programs, and it can run with higher efficiency than Vina and GWOVina. Furthermore, it can achieve the best cross-docking accuracy, especially for the relatively hard protein families. In the future, some studies will be provoked by our success of RDPSO in protein-ligand docking to investigate the RDPSO properties on more complicated docking problem, such as protein structure prediction and blind docking.

Acknowledgments

The authors acknowledge the support and facilities provided for this research by the Engineering Laboratory of Pattern Recognition and Computing Intelligence of the Jiangsu Province, China.

Funding

This work was supported in part by the National Natural Science Foundation of China (Projects Numbers: 61673194, 61672263, 61672265) and in part by the National First-class Discipline Program of Light Industry Technology and Engineering (Project Number: LITE2018–25).

References

[1] X.Y. Meng, H.X. Zhang, M. Mezei, and M. Cui, "Molecular Docking: A Powerful Approach for Structure-Based Drug Discovery," *Curr. Comput. - Aid. Drug* vol. 7: pp. 146-157, 2016.

[2] M.R. Koebel, G. Schmadeke, R.G. Posner, and S. Sirimulla, "AutoDock VinaXB: implementation of XBSF, new empirical halogen bond scoring function, into AutoDock Vina," *J. Cheminformatics* vol. 8: pp. 27, 2016.

[3] G. Morris, D. Goodsell, R. Halliday, R. Huey, W. Hart, R. Belew, and A. Olson, "Automated Docking Using a Lamarckian Genetic Algorithm and Empirical Binding Free Energy Function," *J. Comput. Chem.* vol. 19: pp. 1639-1662, 1998.

[4] O. Trott and A.J. Olson, "AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading," *J. Comput. Chem.* vol. 31: pp. 455-61, 2010.

[5] V. Namasivayam and R. Günther, "pso@autodock: a fast flexible molecular docking program based on Swarm intelligence," *Chem. Biol. Drug. Des.* vol. 70: pp. 475-484, 2010.

[6] H.M. Chen, B.F. Liu, H.L. Huang, S.F. Hwang, and S.Y. Ho, "SODOCK: swarm optimization for highly flexible protein-ligand docking," *J. Comput. Chem.* vol. 28: pp. 612-623, 2010.

[7] Y. Liu, L. Zhao, W. Li, D. Zhao, M. Song, and Y. Yang, "FIPSDock: A new molecular docking technique driven by fully informed swarm optimization algorithm," *J. Comput. Chem.* vol. 34: pp. 67-75, 2012.

[8] Eberhart and S. Yuhui, "Particle swarm optimization: developments, applications and resources," *Proceedings of the 2001 Congress on Evolutionary Computation,* Seoul, Korea (South), pp. 81-86 vol. 1, 2001.

[9] M.C. Ng, S. Fong, and S.W. Siu, "PSOVina: The hybrid particle swarm optimization algorithm for protein-ligand docking," *J. Bioinf. Comput. Biol.* vol. 13: pp. 1541007, 2015.

[10] H.K. Tai, H. Lin, and S.W.I. Siu, "Improving the efficiency of PSOVina for protein-ligand docking by two-stage local search," *2016 IEEE Congress on Evolutionary Computation,* Vancouver, BC, Canada, pp. 770-777, 2016.

[11] H.K. Tai, S.A. Jusoh, and S.W.I. Siu, "Chaos-embedded particle swarm optimization approach for protein-ligand docking and virtual screening," *J. Cheminformatics* vol. 10: pp. 62, 2018.

[12] S. Mirjalili, S.M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Adv. Eng. Softw.* vol. 69: pp. 46-61, 2014.

[13] K.M. Wong, H.K. Tai, and S. Siu, "GWOVina: A grey wolf optimization approach to rigid and flexible receptor docking," *Chem. Biol. Drug. Des.* vol. 97: pp. 97-110, 2020.

[14] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *International Conference on Neural Networks,* pp. 1942-1948, 2002.

[15] Y.J. Gong, J.J. Li, Y. Zhou, L. Yun, S.H. Chung, Y.H. Shi, and J. Zhang, "Genetic Learning Particle Swarm Optimization," *IEEE Transactions on Cybernetics* vol. 46: pp. 2277-2290, 2017.

[16] J. Sun, X. Wu, V. Palade, W. Fang, and Y. Shi, "Random drift particle swarm optimization algorithm: convergence analysis and parameter selection," *Mach. Learn.* vol. 101: pp. 345-376, 2015.

[17] F. Yi, Z. Chen, and J. Sun, "Random Drift Particle Swarm Optimisation Algorithm for Highly Flexible Protein-ligand Docking," *J. Theor. Biol.* vol. 457: pp. 180-189, 2018.

[18] S.D. Handoko, X. Ouyang, C.T.T. Su, C.K. Kwoh, and Y.S. Ong, "QuickVina: Accelerating AutoDock Vina Using Gradient-Based Heuristics for Global Optimization," *IEEE/ACM Trans. Comput. Biol. Bioinformatics* vol. 9: pp. 1266–1272, 2012.

[19] R. Wang, L. Lai, and S. Wang, "Further development and validation of empirical scoring functions for structure-based binding affinity prediction," *J. Comput. Aid. Molec. Design* vol. 16: pp. 11-26, 2002.

[20] R. Quiroga and M.A. Villarreal, "Vinardo: A Scoring Function Based on Autodock Vina Improves Scoring, Docking, and Virtual Screening," *PLoS One* vol. 11: pp. e0155183, 2016.

[21] M. Su, Q. Yang, Y. Du, G. Feng, Z. Liu, Y. Li, and R. Wang, "Comparative Assessment of Scoring Functions: The CASF-2016 Update," *J. Chem. Inf. Model.* vol. 59: pp. 895-913, 2019.

[22] J.J. Sutherland, R.K. Nandigam, J.A. Erickson, and M. Vieth, "Lessons in Molecular Recognition. 2. Assessing and Improving Cross-Docking Accuracy," *J. Chem. Inf. Model.* vol. 47: pp. 2293-2302, 2007.

[23] H.M. Berman, T. Battistuz, T.N. Bhat, W.F. Bluhm, P.E. Bourne, K. Burkhardt, Z. Feng, G.L. Gilliland, L. Iype, and S. Jain, "The Protein Data Bank," *Nucleic Acids Res.* vol. 28: pp. 235-242, 2003.

[24] P. Hawkins, G.L. Warren, A.G. Skillman, and A. Nicholls, "How to do an evaluation: pitfalls and traps," *J. Comput. Aid. Molec. Design* vol. 22: pp. 179-190, 2008.

[25] S.M. Taheri and G. Hesamian, "A generalization of the Wilcoxon signed-rank test and its applications," *Stat. Pap.* vol. 54: pp. 457-470, 2013.

[26] W.P. Feinstein and M. Brylinski, "Calculating an optimal box size for ligand docking and virtual screening against experimental and predicted binding pockets," *J. Cheminformatics* vol. 7: pp. 1-10, 2015.

[27] J. Shamsara, "CrossDocker: a tool for performing cross-docking using Autodock Vina," *Springerplus* vol. 5: pp. 344, 2016.