MSLDOCK: Multi-Swarm Optimization for Flexible Ligand Docking and Virtual Screening

Chao Li, Jun Sun and Vasile Palade

Original citation & hyperlink:
Li, C., Sun, J. and Palade, V., 2021. MSLDOCK: Multi-Swarm Optimization for Flexible Ligand Docking and Virtual Screening. *Journal of Chemical Information and Modeling*, *61*(3), pp.1500-1515.
https://doi.org/10.1021/acs.jcim.0c01358

DOI 10.1021/acs.jcim.0c01358
ISSN 1549-9596
ESSN 1549-960X

Publisher: American Chemical Society

MSLDOCK: Multi-Swarm Optimization for Flexible Ligand Docking and Virtual Screening

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

Address: [1] Department of Computer Science and Technology, Jiangnan University, No.1800, Lihu Avenue, Wuxi, Jiangsu 214122, PR China, [2] Faculty of Engineering and Computing, Coventry University, Priory Street, Coventry, CV1 5FB, UK

E-mail: Chao Li - lcmeteor@hotmail.com; Jun Sun* - junsun@jiangnan.edu.cn; Vasile Palade - ab5839@coventry.ac.uk
*Corresponding author

Abstract

Autodock and its various variants are widely utilized docking approaches which adopt optimization methods as search algorithms for flexible ligand docking and virtual screening. However, many of them have their limitations, such as poor accuracy for dockings with highly flexible ligands, and low docking efficiency. In this paper, a multi-swarm optimization algorithm integrated with Autodock environment is proposed to design a high-performance and high-efficiency docking program, namely MSLDOCK. The search algorithm is a combination of the random drift particle swarm optimization with a novel multi-swarm strategy and the Solis and Wets local search method with a modified implementation. Due to the algorithm's structure, MSLDOCK also has a multithread mode. The experimental results reveal that MSLDOCK outperforms other two Autodock-based approaches in many aspects, such as self-docking, cross-docking and virtual screening accuracies as well as docking efficiency. Moreover, compared with three non-Autodock-based docking programs, MSLDOCK can be a reliable choice for

self-docking and virtual screening, especially for dealing with highly flexible ligand docking problems. The source code of MSLDOCK can be downloaded for free from https://github.com/lcmeteor/MSLDOCK.

1. Introduction

Automated protein-ligand docking methods are effective tools in drug design[1], aiming to predict the experimental binding modes and affinities of small molecules within the binding site of particular receptor targets[2]. Among them, one of the most conventionally adopted ways in protein-ligand docking is flexible ligand docking, in which ligands are treated as articulated objects while proteins are rigid. A flexible ligand docking problem can be solved by identifying the translation, orientation and conformation of the ligand relative to the active site of the protein. Its implementation is based on two components: the optimization search algorithm and the scoring function[3]. The optimization search algorithm is a tool to find suitable docked poses of the ligand within a certain area around the binding site, and the scoring function is used to approximately evaluate the binding energy of docking conformations found by the algorithm.

During the past few decades, several software packages, such as Autodock[4], Autodock Vina[5] (referred to as Vina), DOCK[6], LeDock[7], GOLD[8], Glide[9], Surflex-Dock[10], etc., were developed to solve the flexible ligand docking problems. Among them, Autodock is a widely used one, since it is an open source software and can be easily implemented. It adopts a Lamarckian genetic algorithm (LGA), i.e., a hybrid of the

genetic algorithm (GA) and the Solis and Wets local search (SWLS) method[11], as its default search algorithm, and a semi-empirical energy function as its scoring function[4]. In this paper, we mainly focus on design of a new search algorithm for Autodock, since an effective search algorithm can generally provide the docking software a high probability of accurately estimating the ligand-binding affinity or finding a conformation close to the co-crystalized ligand pose with less computational time, which can in turn enhance both the accuracy and efficiency for ligand pose prediction and virtual screening.

Since the release of Autodock, various search algorithms have been proposed for flexible ligand docking. Namasivayam and Günther[12] utilized two variants of particle swarm optimization (PSO), namely varCPSO and varCPSO-Ls, in Autodock for rapid docking with highly flexible ligands. SODOCK[13] adopts a hybrid search algorithm, which combines the PSO algorithm using the neighboorhood topology with the SWLS method, to solve highly flexible ligand docking problems. This hybrid search algorithm, named as Lamarckian PSO (LPSO), has already been integrated into the latest version (4.2.6) of Autodock. FIPSDock[14] implements a variant of the fully informed particle swarm (FIPS) optimization method in Autodock. The experimental results in ref 14 revealed that FIPSDock might be more suitable for highly flexible ligand docking than conventional genetic algorithm-based algorithms. The CEPGA[15], the GA with crossover elitist preservation, was integrated with Autodock 4.2.6 for protein-ligand docking. This variant of GA could keep the elite individuals of the last generation and make the crossover more efficient and robust. In ref 16, a novel optimization algorithm called fitness learning-based artificial bee colony with proximity stimuli (FlABCps) was

implemented in the Autodock environment, showing its superior docking performance when compared with other four search algorithms. In AutoDockFR[17], Ravindranath et al. introduced a new GA and a customized scoring function, to make this variant of Autodock to be more suitable for modeling with receptor flexibility than Autodock4 and Vina. The integration of Autodock with jMetalCpp was made in ref 18, which provides both single- and multi-objective algorithms to solve docking problems. A more recent version of Autodock, known as FWADOCK[19], utilizes an improved fireworks optimization method as its search algorithm, in which the diversity is maintained effectively to avoid premature convergence, and thus yields improve the docking performance. In addition, some nature-inspired docking methods not based on the Autodock program were also proposed in recent years. For example, PLANTS[20] is a protein-ligand docking system based on the ant colony optimization and two empirical scoring functions, namely $\mathrm{PLANTS_{CHEMPLP}}$ and $\mathrm{PLANTS_{PLP}}$. PSOVina[21] and GWOVina[22] are two variants of Vina program that utilize chaos-embedded particle swarm optimization and grey wolf optimization as their search algorithms, respectively. Both PSOVina and GWOVina can obtain superior performance in terms of docking efficiency.

Most of the improved search algorithms coupled with Autodock software and other docking programs still have their limitations, such as poor accuracy for highly flexible ligand docking and relatively low docking efficiency[23,24], although in some cases the docking accuracy of the conformations found by these programs are better than those found by Autodock-default (referred to as Autodock-d), in which LGA used as the search

algorithm. Therefore, in order to design a search algorithm with high performance and high efficiency for flexible ligand docking based on the Autodock docking environment, a novel hybrid search algorithm is proposed in this paper. This algorithm is a combination of the random drift particle swarm optimization (RDPSO) with a novel proposed multi-swarm strategy and the SWLS method with a modified implementation. Unlike most of the multi-swarm strategy, for examples, widely used dynamic structure[25-27] and master-slaver model[28-31], the novel multi-swarm strategy is more suitable for dealing with docking problems. It divides the whole swarm of the RDPSO into several equal-sized sub-swarms, with a feature exchange method designed to ensure enough information exchange between sub-swarms. The hybrid algorithm is named as the multi-swarm Lamarckian RDPSO (MSLRDPSO) and is employed in the latest version of Autodock (version 4.2.6), with the corresponding improved docking software called MSLDOCK. Compared with the previously proposed algorithms, the advantage of MSLRDPSO is illustrated below:

- The RDPSO algorithm generally has a better search performance than many optimization algorithms[32;]
- The multi-swarm strategy, especially when the feature exchange method is used, can help maintain the diversity of the entire swarm, thereby enhancing the robustness of the algorithm;
- Compared with some docking search algorithms including LGA and LPSO, MSLRDPSO is much more efficient because of the multi-swarm structure;
- The implementation of the SWLS method has been modified in MSLRDPSO,

to adapt to the multi-swarm structure, further improving the diversity of each sub-swarm;

- Due to the multi-swarm structure, it is easy for the search process of MSLRDPSO to be parallelized in terms of sub-swarm numbers, and thus a multithread mode of MSLDOCK, i.e., MSLDOCK-M, is proposed for the purpose of parallelizing single docking tasks.

2. Methods

2.1 Random drift particle swarm optimization

Random Drift Particle Swarm Optimization (RDPSO)[32] is a heuristic algorithm, which is usually used to solve non-continuous, complex and global optimization problems. In this work, RDPSO is used as the main body of the proposed hybrid algorithm to solve docking problems. This algorithm is motivated by the trajectory analysis of the canonical PSO in ref 33 and the free electron model in metal conductors placed in an external electric field[34]. In a RDPSO with M individuals, each individual has N dimensions, with the current position vector and the velocity vector of particle i at the n^{th} iteration represented as $X_{i,n} = (X_{i,n}^1, X_{i,n}^2, \cdots, X_{i,n}^N)$ and $V_{i,n} = (V_{i,n}^1, V_{i,n}^2, \cdots, V_{i,n}^N)$, respectively. The previous best position of particle i according to the fitness value is expressed as $P_{i,n} = (P_{i,n}^1, P_{i,n}^2, \cdots, P_{i,n}^N)$, called the personal best (pbest) position, and the best one of the pbest positions of all the particles according to the fitness values is $G_n = (G_n^1, G_n^2, \cdots, G_n^N)$, called the global best (gbest) position. Each particle in RDPSO moves according to the following equations

$$V_{i,n+1}^{j} = \alpha|C_n^j - X_{i,n}^j|\varphi_{i,n+1}^j + \beta(p_{i,n}^j - X_{i,n}^j) \quad (1)$$

$$X_{i,n+1}^{j} = X_{i,n}^j + V_{i,n+1}^j \quad (2)$$

In equation (1), the first term is the random velocity simulating the thermal motion of an electron, where $\alpha > 0$ is a parameter called the thermal coefficient, C_n is the mean best (mbest) position defined by the mean of the pbest positions of all the particles, and $\varphi_{i,n+1}^j$ is the sequence of random number subject to standard normal distribution. The second term is the drift velocity denoting the drift motion, where $\beta > 0$ is another parameter named the drift coefficient, and $p_{i,n} = (p_{i,n}^1, p_{i,n}^2, \cdots, p_{i,n}^N)$ is the local focus of particle i in canonical PSO, which can be expressed as[33]:

$$p_{i,n}^j = \gamma_{i,n}^j P_{i,n}^j + (1 - \gamma_{i,n}^j) G_n^j, \qquad \gamma_{i,n}^j \sim U(0,1) \quad (3)$$

With respect to the setting of two algorithmic parameters in RDPSO, α and β are set to decrease linearly from 0.9 to 0 and 1.45 to 1, respectively. Unlike the parameter setting recommended in ref 32, this parameter setting can make the algorithm search in a small range, enhancing the exploitation ability of particles, which meets the requirements for high precision of the results in docking problems.

2.2 Multi-swarm strategy and the feature exchange method

MSLRDPSO utilizes a novel multi-swarm strategy, in which the whole swarm is divided into several equal-sized sub-swarms. Obviously, the number of particles in each sub-swarm is much smaller than that of the entire swarm. Thus, in order to ensure that each sub-swarm does not easily prematurely converge, a feature exchange method is used every certain number of energy function evaluations to ensure the information exchanged

effectively and efficiently between different sub-swarms for the purpose of maintaining their diversities. Between two feature exchange operations, all particles in each sub-swarm implement their own RDPSO search process following equations (1) and (2). Before describing the feature exchange method, the definition of feature dimension is described in detail below.

For a problem with an N-dimensional search space, the entire swarm is divided into several sub-swarms represented as S_1 to S_T, with the corresponding pbest populations，each of which is the set of pbest positions of all the particles in each sub-swarm, expressed as B_1 to B_T, respectively, where T is the number of sub-swarms. In each sub-swarm's pbest population, several specific dimensions are marked as features. A feature marking the j^{th} $(1 \le j \le N)$ dimension is represented as F_j, and if the marked dimension belongs to B_i $(1 \le i \le T)$, such a feature can be further represented as F_j^i. Note that each F_j^i contains the j^{th} components of all pbest positions in B_i, so F_j^i is actually the j^{th} column vector in B_i. The features in all sub-swarms should meet the following conditions:

- The number of features in each sub-swarm should be N/T (suppose N is divisible by T, if not, the first sub-swarm will share equally the remaining features). Hence, the entire swarm will have N dimensions being marked as features;
- For any two features in two different sub-swarms, i.e., F_{j1}^{i1} and F_{j2}^{i2} $(i1 \ne i2, 1 \le j1,\ j2 \le N)$, it means $j1 \ne j2$. Hence, two different sub-swarms cannot mark the same dimension as features.

According to the above conditions, it can be concluded that for the entire swarm, if the sequence numbers of all the dimensions marked as features are put together, a sequence of integers 1 to N can be obtained. Furthermore, condition 1 guarantees that the number of dimensions marked as features in each sub-swarm is almost the same, which implies that the features are distributed as equal as possible in all sub-swarms.

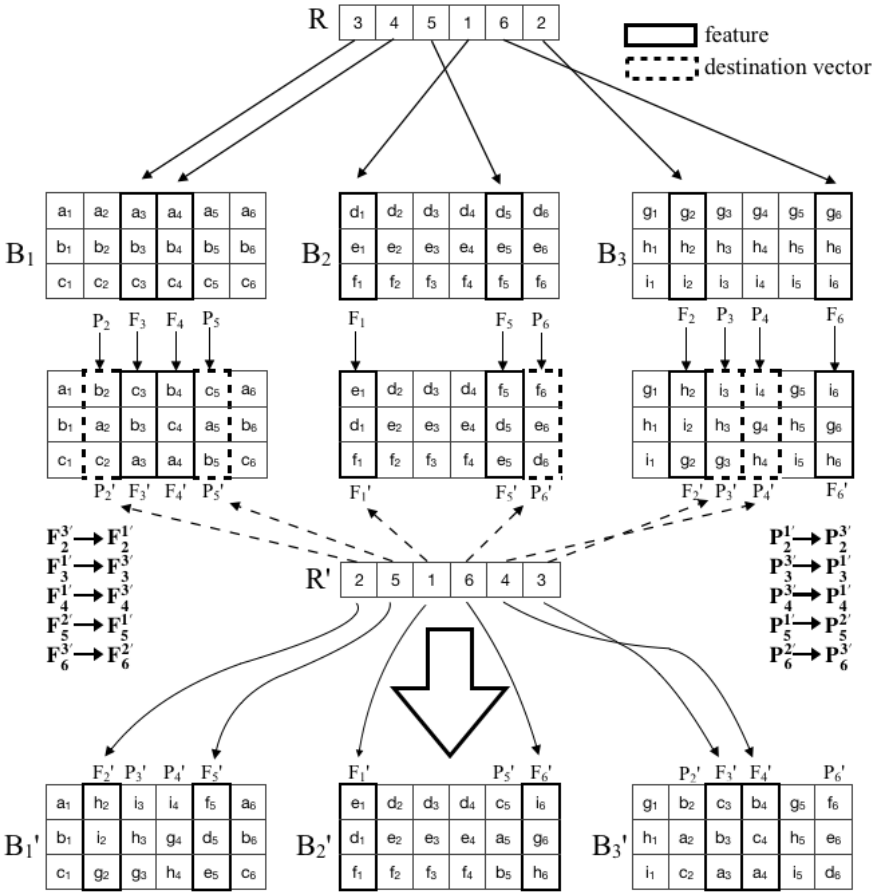


Figure 1. Example of one feature exchange operation.

Based on this definition, the feature exchange method is used to exchange feature dimensions between different sub-swarms to guarantee enough information exchange. In Figure 1, there are 3 pbest matrixes B_1, B_2 and B_3 of three sub-swarms, and the dimension of the problem is 6. The vector R is generated by shuffling integers from 1 to 6, and the elements in R serially represent the positions of the column vectors

regarded as features in $\{B_1, B_2, B_3\}$ (see where the arrows from R point to in Figure 1). Each pbest matrix has two features, as all the features should be distributed evenly. When the feature exchange operation begins, R' is generated, whose elements indicate the new locations of all the features in the next generation. The column vectors in the destinations are marked as $P_j(1 \leq j \leq 6)$, who need to exchange with the features. In order to exchange information between different rows in each pbest matrix, the elements in all features and destination vectors should be shuffled (generating F_j', P_j' $(1 \leq j \leq 6)$). Then the corresponding column vectors F_j' and P_j' should be exchanged one by one, moving F_j' to their destinations and P_j' to the original positions of F_j. Note that there is a certain possibility that one or more features fail to change sub-swarms in one feature exchange operation (e.g. F_1' in B_2), but the shuffle inside the features still needs to be done in this situation. Combining these migrated vectors with the other elements in each pbest matrix, three new pbest matrixes B_1', B_2' and B_3' are obtained, with the positions of their features corresponding to the numbers in R'.

After the information exchanges between different sub-swarms, it is obvious that the new pbest positions may not be better than the original ones, so they should be evaluated after every exchange and the better ones are retained. After such evaluations, with respect to the pbest population in each sub-swarm, it probably contains better information from other sub-swarms, which definitely improves its diversity and leads the particles to search more effectively. Note that our preliminary experiments showed the number of energy function evaluations required between two feature exchange methods was $30M$, which is the appropriate setting to MSLDOCK for flexible ligand docking.

2.3 The local search method implementation

Unlike in LGA and LPSO, the implementation of SWLS in MSLRDPSO has been modified, which can be adapted to the multi-swarm structure to enhance local optimization, and can further improve the diversity of each sub-swarm during the search process. After all the sub-swarms finished one-time RDPSO iterative process, the novel implementation gives a certain probability of performing the local search method on the best particle (the particle with the best fitness value in the current RDPSO iteration) in each sub-swarm. In this paper, the local search probability is set to $1/T$, which means that for the entire swarm, the expected number of particles to perform the SWLS method in each generation is 1. The other setting of SWLS is the same as that applied in LPSO, which can be referred to ref 13.

2.4 MSLRDPSO and MSLDOCK

The proposed MSLRDPSO algorithm combines the RDPSO with the multi-swarm strategy and the SWLS method with a modified implementation. Figure 2(A) illustrates the normal procedure of the MSLRDPSO algorithm (serial mode). In MSLRDPSO, one generation is composed of T iterations with each iteration being the normal RDPSO iteration executed by a sub-swarm, or composed of T iterations with each iteration being the hybrid of the normal RDPSO iteration and one local search process executed by each sub-swarm. As shown in Figure 2(A), all sub-swarms execute their own search process sequentially within one generation in the serial mode of MSLRDPSO. The pseudocode of the serial mode of MSLRDPSO is shown below in Algorithm 1.

Algorithm 1: MSLRDPSO-serial ($T, M, N, X_{min}, X_{max}, neval_{max}, exn$)

```
pexn = 0; /* pexn records the count of evaluations of the previous information exchange */
nexn = exn; /* exn is number of evaluations required between two feature exchanges */
R = randperm(N); /* randperm is a function to make sequence {1,2 ⋯ , N} out of order */
Distribute particles to each sub-swarm;
Set the corresponding dimensions as features in each sub-swarm according to the R vector;
Compute C_0 and find G_0 among P_{i,0} in each sub-swarm;
While neval < neval_max do /* neval is the current number of evaluations */
    lsf = rand() < 1/T /* local search flag, if true, do local search in each sub-swarm */
    For sub-swarm from 1 to T do
        For particle i in each sub-swarm do
            calculate new velocity V_{i,n+1} using (7);
            fix V_{i,n+1} if it is out of range;
            calculate new position X_{i,n+1} using (8);
            evaluate the objective function value f(X_{i,n+1});
        End for
        If lsf == true /* do local search */
            Apply the SWLS method to the best X_{i,n+1} in each sub-swarm;
        End if
        Update P_{i,n+1}, C_{t,n+1} and G_{t,n+1}
    End for
    If neval ≥ nexn /* do feature exchange */
        R' = randperm(N);
        Do feature exchange according to R and R' to generate all tP_{i,n};
        For particle i from 1 to M do
            Compare f(tP_{i,n+1}) with f(P_{i,n}) and the one with better fitness replace P_{i,n};
        End for
        R' = R;
        pexn = nexn;
        nexn = pexn + exn;
    End if
End while
```

Meanwhile, due to the structure of the multi-swarm strategy and the modified implementation of SWLS method, the search process of RDPSO and SWLS within each sub-swarm can be performed simultaneously, that is, the parallel mode of MSLRDPSO, as shown in Figure 2(B). In the parallel mode, with the help of OpenMP interface, each thread is designed to load one sub-swarm so that all the sub-swarms can be performed

simultaneously. After each generation and each feature exchange operation, the sub-swarms that have already finished their own work should wait for others. The parallel mode cannot maintain complete synchronization between sub-swarms mainly due to the uncertain searching time of each SWLS application[4,13] (see the local search in Figure 2(B)), but such a small idle time during the entire search process is acceptable.

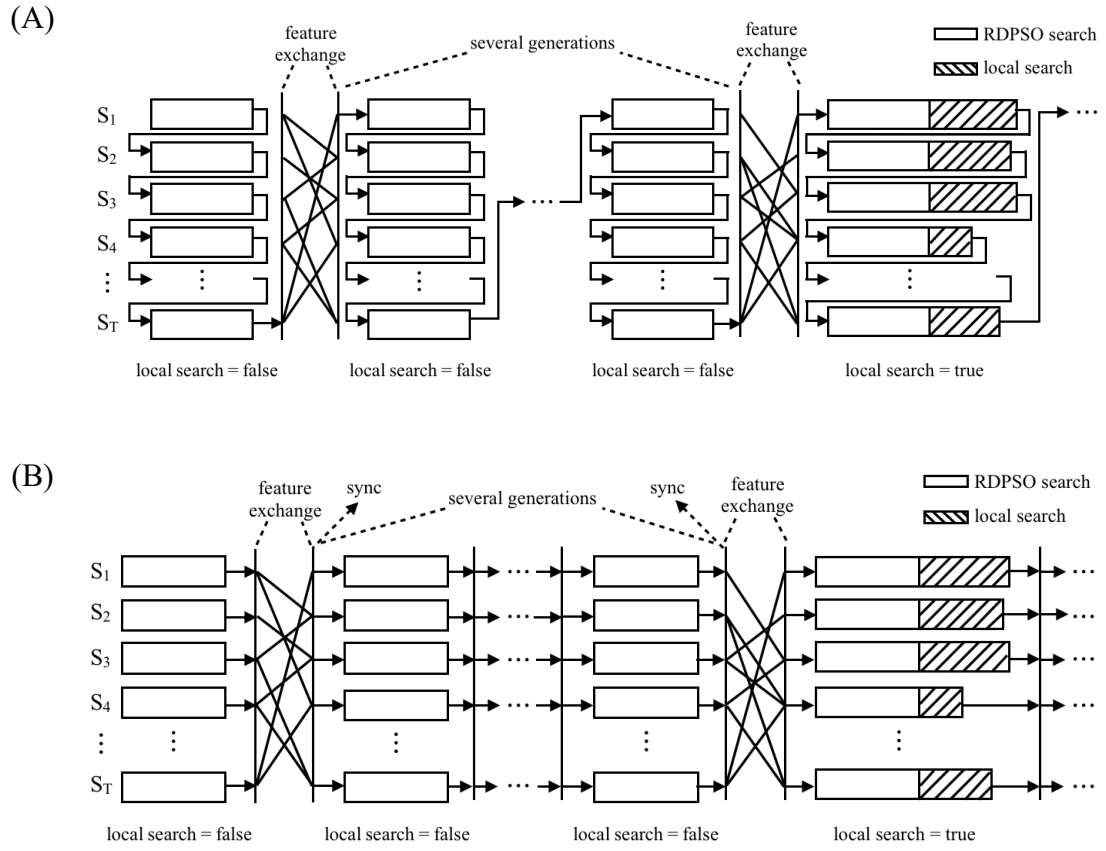


Figure 2. The procedures of the serial mode and parallel mode of MSLRDPSO. (A) The procedure of the serial mode of MSLRDPSO. (B) The procedure of the parallel mode of MSLRDPSO.

The normal mode and multithread mode of MSLDOCK are the serial mode and parallel mode of MSLRDPSO integrated with the docking environment and scoring function of Autodock 4.2.6, respectively. To make the program be adapt to the OpenMP interface, some files in the original version of Autodock has been modified in

MSLDOCK, especially those related to scoring function calculation and local search, since they are not thread-safe.

3. Experimental Setups

3.1 Datasets

In this paper, the experimental section comprises three parts: self-docking, cross-docking and virtual screening. By examining the proposed docking program from different perspectives, the performance of MSLDOCK can be comprehensively evaluated.

Two datasets were used for evaluating self-docking performance in this paper. One is a subset of the PDBbind dataset[35]. This dataset is carefully selected from the PDBbind refined set through a systematic, non-redundant sampling procedure, which is used as the dataset in the CASF benchmark and is named as the PDBbind coreset. The dataset included in the latest version of the CASF benchmark (CASF-2016[36]), which was published in 2019, was employed in this paper. All the 285 complexes in this PDBbind coreset were employed in our experiments, and the number of torsions of the ligands in these complexes ranges from 0 to 36. The protein and ligand files of the PDBbind coreset can be download from http://www.pdbbind-cn.org/casf.asp. Another one is the widely-used GOLD benchmark dataset[21,37,38], which originally contains 134 test cases and are available at https://www.ccdc.cam.ac.uk/support-and-resources/Downloads/. Eliminating the test cases for which the compared docking programs cannot do successful docking preparations (e.g. failed to generate .pdbqt files by AutodockTools

within MGLTools 1.5.6 when docking with Autodock, or failed to generate .mae files by Schrodinger 2018-1 when docking with Glide), we used a set of 103 complexes from GOLD dataset for performance testing in this paper. The names of these test cases are listed in Table S1. The number of torsions of the ligands in these complexes ranges from 0 to 28. Therefore, the two datasets for self-docking experiments totally contain 388 test cases, with their search dimensions ranging from 7 to 43. With the increase of dimensions of the search space, the algorithmic performance can be comprehensively evaluated.

In cross-docking experiments, we used a collection of 20 protein-ligand complexes for two protein targets (CDK2 and MAPK14) to benchmark the performance of MSLDOCK. For each target, 10 protein conformer structures were selected from the Sutherland-crossdock-set[39] and the names of these complexes are listed in Table S2. All the 20 complexes were downloaded from Protein Data Bank (PDB)[40] and aligned by using PyMol. Then for each complex, the co-crystalized ligand and protein were saved as two individual files. The python scripts in AutodockTools were utilized to add hydrogens to the ligands and receptors for each test case.

When it comes to virtual screening, the database of Useful Decoys-Enhanced (DUD-E)[41] was employed for this purpose. The entire dataset consists of 102 protein targets with known active ligands and computationally generated inactive ligands. The inactive ligands, called decoys, were made to have similar physicochemical properties such as molecular weight, number of rotatable bonds, calculated log P, and hydrogen bond acceptors and donors, but dissimilar 2D topologies from the active ligands so that it is challenging for docking programs to identify real positives from the positive-like ligands.

As docking all the compounds to their targets would require massive amounts of CPU time, we chose four targets with a relatively small number of total compounds from the "Diverse" subset in DUD-E (i.e., ampc, cxcr4, cp3a4 and kif11) as the dataset for the virtual screening experiments.

3.2 Settings of compared docking programs

According to the structure of the proposed multi-swarm strategy, the key parameter of the MSLRDPSO algorithm is T (number of sub-swarms). According to our preliminary experiments (see the PDF file in the supporting information for corresponding results and analysis), MSLDOCK with 2 sub-swarms, i.e., MSLDOCK-s2, outperforms other versions of MSLDOCK in terms of the lowest docked energy (the docked energy is the energy value calculated by the semi-empirical scoring function used in Autodock) but has relatively large standard deviations. Therefore, when there are enough repetitions for a single docking test, MSLDOCK-s2 may be more appropriate to find a docking conformation with lower binding energy and to generate poses with more different energy levels than the MSLDOCK versions with more sub-swarms. On the other hand, among all the compared versions of MSLDOCK, MSLDOCK with 6 sub-swarms, i.e., MSLDOCK-s6, has the smallest standard deviation, and its mean docked energy performance in terms of different number of torsions is one of the best. MSLDOCK-s6 may be suitable for finding good docking results for the "fast" docking problems, that is, docking single test cases with a small number of repetitions. Therefore, MSLDOCK-s2 and MSLDOCK-s6 were chosen to compared with other docking programs in further experiments.

In this work, MSLDOCK-s2 and MSLDOCK-s6 were compared with two docking programs based on Autodock (i.e., Autodock-d and SODOCK), and three other non-Autodock-based docking programs, including Vina[5] (version 1.1.2), LeDock[7] (version 1.0), and Glide[9] (version 7.8, integrated in Schrodinger 2018-1). Autodock-d was chosen since LGA is the default search algorithm in the latest version of Autodock. We selected SODOCK rather than other optimization algorithms integrated with Autodock (e.g. varCPSO-ls and FIPSDock mentioned in the "Introduction" section) since Guo et al.[23] has proved that the SODOCK generally has better docking accuracy and robustness than many optimization algorithms designed for Autodock. Vina was developed in order to improve the docking speed and accuracy of Autodock4, and thus this program was often used to compare with Autodock in many aspects[5,24,42] and was also employed in our experiments. In ref 24, five commercial docking programs and five academic docking programs were compared, and among them, the LeDock program has been proved to have the best prediction accuracy for the poses with the best scores, while Glide was considered to be the most robust program. Therefore, the academic docking program LeDock and the commercial docking program Glide were chosen for performance comparison in our experiments.

In all these docking experiments, only the flexibility of ligands was explored, and the receptors were always kept rigid. For each test case, all the compared programs started with the same structure of the ligand. And the translation, orientation and conformation of the ligand were all randomly initialized within their corresponding ranges. The self-docking experiments were carried out on a workstation with an Intel®

i7-6850K 12-core 3.60GHz processor and totally 128-GB RAM, while the other experiments were run on a workstation with an Intel® i9-9900X 20-core 4.00GHz processor and totally 64-GB RAM. The operating systems of both these two workstations are Ubuntu 16.04. The specific parameter settings of each docking program are listed below.

Autodock-based programs

The implementation of all the Autodock-based programs in our experiments, including MSLDOCK, Autodock-d, and SODOCK, utilized the docking environment and scoring function of Autodock 4.2.6. The detailed explanation of this scoring function can be found in ref 43. For all the experiments, the number of particles was set to 150 and the maximum number of iterations of single SWLS method was set to 300. For virtual screening, the number of energy function evaluations was set to 2.5×10^5 (short length) and each test case was docked for 10 repetitions, since we cannot spend too much time on a single test case in virtual screening. For self-docking and cross-docking experiments, the number of energy function evaluations was set to 2.5×10^6 (medium length) and each test case was docked for 50 repetitions. With enough number of energy function evaluations and docking repetitions, the obtained statistical results can be considered as convincing ones. Other specific parameter settings of Autodock-d and SODOCK can be accessed in ref 4 and ref 13, respectively. As Autodock-based programs do not handle ligands with more than 32 torsions, for larger ligands recompiled versions allowing up to 64 torsions were used.

The PDBQT format files of the receptor and ligand for each test case were generated

by using the AutodockTools. The energy grid maps were calculated with AutoGrid. To make the search area large enough for all the ligands to rotate, the grid size was set to 60×60×60 points (a cube with an edge length of 22.5 Å) for all test cases. The center of the grid map was always set as the center of the reference ligand for each test case.

Autodock Vina

In Autodock Vina[5], the implementation of the global optimizing algorithm is related to Markov chain Monte Carlo algorithm with restart supplemented by random mutating the current solution, while the Broyden-Fletcher-Goldfarb-Shanno algorithm is used as the local optimization method[44]. A superficially physics-based scoring function is adopted in Vina[5].

For each test case, the grid map for Vina was set the same as that for Autodock-based programs. The maximum energy difference between the best binding mode and the worst one reported in the result files was set to 10 kcal/mol, in order to generate enough modes with different energy level. With respect to self-docking and cross-docking experiments, the maximum number of binding modes to generate was set to 20, which is the maximum value can be set in Vina, and each test case was docked with the exhaustiveness set to 56. For virtual screening, 10 docking poses was generated and the exhaustiveness was set to 8 for each test case. These two settings of exhaustiveness correspond to the medium-length and short-length evaluations in Autodock, respectively[42].

LeDock

The LeDock[7] applies an empirical scoring function and the simulated annealing

search algorithm. The LePro (http://www.lephar.com) was used to add hydrogen atoms to receptors and write the input file for LeDock. The search range of LeDock program for each test case was set the same as that of the Autodock-based programs. The number of binding poses to generate for each test case was set to 10 for virtual screening and 50 for both self-docking and cross-docking.

Glide

The Glide[9] program uses a hierarchical series of filters to search for possible locations of the ligand in the active-site region of the receptor. The scoring function adopted in Glide is based on ChenScore[45], but includes a steric-clash term, adds some rewards and penalties.

The LigPrep program in Schrodinger 2018-1 was used to prepare ligands for docking with Glide. The Glide program was used for both grid generation and protein-ligand docking for each test case. The grid center and the box size settings were the same as those used by the Autodock-based programs. Docking precision mode was set to the standard precision (SP) rather than the extra precision (XP) since the docking poses from SP can have more diversity than those from XP[24]. The forcefield used in Glide was OPLS3e. The number of the poses reported for each docking test case was set to 10 for virtual screening and 50 for both self-docking and cross-docking. All the other parameters were set to default values.

3.3 Performance metrics

In this paper, the binding free energy, which was only reported at the end of a docking

test in Autodock, was compared among various Autodock-based programs. The binding free energy should be distinguished from the energy value calculated by the semi-empirical scoring function in Autodock 4.2.6. The energy evaluated by the scoring function includes the intermolecular and intramolecular interaction energies[4], while the binding free energy is only the sum of the intermolecular energy and the torsional free energy, but not including the internal or intramolecular interaction energy of the ligand. The final docking conformations in Autodock-based programs were ranked according to the binding free energy, which is instructive for the selection and specific research of the final conformations[4].

The similarity between the produced conformation and the co-crystallized one is usually accessed by calculating the root mean squared deviation (RMSD) between them. Unlike the default way of calculating the RMSD value in Autodock (considering the symmetry of the conformations, probably working well when the two conformations are very similar[46]), the strictest definition of the RMSD was adopted in this paper by calculating its value using all the heavy atoms of the ligand without considering the symmetry. This means the changes in the conformation, as well as in the position and orientation of the entire ligand within the protein's binding site should be measured. In this paper, two kinds of RMSD were evaluated: one is the best-scored RMSD, which is the RMSD between the reference structure and the conformation with the lowest score or binding free energy; the other is the best-sampled RMSD, which is the RMSD of the conformation closest to the crystal one among all the produced conformations. The conformations with the best-scored RMSD and the best-sampled RMSD found by each

docking program are correspondingly called the best-scored conformation and best-sampled conformation, respectively. These two RMSD-related metrics were widely used for comparing docking programs with different scoring functions[47,48]. A common threshold used to determine whether the crystal structure was successfully reproduced is 2 Å, which is also adopted in this paper.

For self-docking experiments, the results of binding free energy, RMSD and docking speed were evaluated in this paper. Firstly, the correlation coefficients between the binding free energies of the best-scored and experimental conformations were compared among Autodock-based docking programs. Secondly, we evaluated the statistical results of the best-sampled RMSD and the best-scored RMSD for all compared algorithms, including the results for all test cases and those in terms of torsions. The computed errors of RMSDs were estimated via 2000 rounds of bootstrapping calculations[49], and the Wilcoxon sign-ranked test[50] was used to determine whether there was a difference between the mean RMSDs obtained by MSLDOCK and other docking programs at a 5% level of significance. Thirdly, the docking speed was evaluated by comparing the mean of computational time for generating per docking pose taken by each compared docking program. Besides, we also compared the computational time of MSLDOCK and MSLDOCK-M in order to show the speed-up performance of the multithread mode.

Cross-docking refers to the docking of every ligand to every receptor structure obtained from a series of receptor-ligand complex pairs. It could be useful to evaluate the ability of a docking program to reproduce the experimental binding pose of a ligand to a protein target complexed with a different ligand. Moreover, cross-docking analysis

could help to access the efficiency of docking studies to support drug lead identification and structure-activity relationship studies[51]. In this paper, the cross-docking results were compared among Autodock-based docking programs, i.e., MSLDOCK-s2, MSLDOCK-s6, Autodock-d and SODOCK, since these programs utilize the same scoring functions in Autodock 4.2.6. For each cross-docking family, the results obtained by every docking program were classified as three classes: docking success, scoring failure and sampling failure[6]. If the best-scored conformation was successfully docked, the corresponding program got a docking success. A scoring failure means a correct pose was sampled but not scored as the best-scored one. If a docking program failed to get a correct pose after 50 docking trials, it got a sampling failure. In this paper, we mainly focused on the number of the test cases in these three classes to evaluate the cross-docking performance for each Autodock-based docking program.

The performance of a docking method in virtual screening was evaluated based on the list of screened compounds ranked by the estimated binding affinities from low to high. Generally, a more effective docking method for virtual screening means it can make more actives get higher rank in the list. A threshold can be set in the list to classify the compounds with lower affinities as actives and the ones with higher affinities as decoys. By calculating the ratios of true positive fraction over the false positive fraction at different classification thresholds, the receiver operating characteristic (ROC) curve of different docking method can be plotted. One performance metric to access the virtual screening performance of a docking program is to calculate the area under the ROC curve (AUC-ROC). An AUC-ROC value of 1.0 indicates perfect classification, whereas a value

of 0.5 indicates random prediction[21]. Another metric used in this work for virtual screening is the enrichment factor (EF). The EF value is to measure the quality of the predicted top-$x\%$ ligands in the list, since the drug discovery research mainly considers the top-ranked ligands from the virtual screening results for further investigation. The value of $\mathrm{EF}_{x\%}$ is computed as[21]:

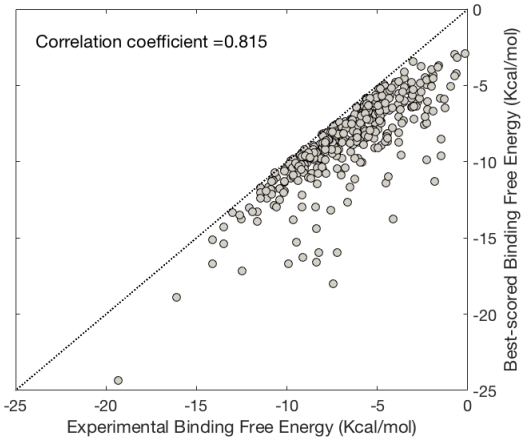
$$\mathrm{EF}_{x\%} = \frac{\text{actives at } x\%}{\text{total actives}} \Big/ \frac{\text{ligands at } x\%}{\text{total ligands}} \tag{4}$$

In this paper, the values of $\mathrm{EF}_{1\%}$ and $\mathrm{EF}_{10\%}$ obtained by all tested docking programs were accessed. Besides, since the parameter settings in virtual screening was different from those in self-docking and cross-docking experiments, we also evaluated the mean time of screening per docking ligand taken by every docking program for each target to show the virtual screening efficiency.

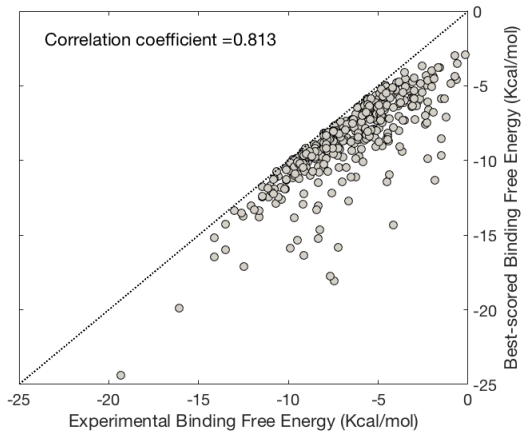
4. Results and Discussion

4.1 Comparison of self-docking results in terms of energy, accuracy and efficiency

(A)



(B)



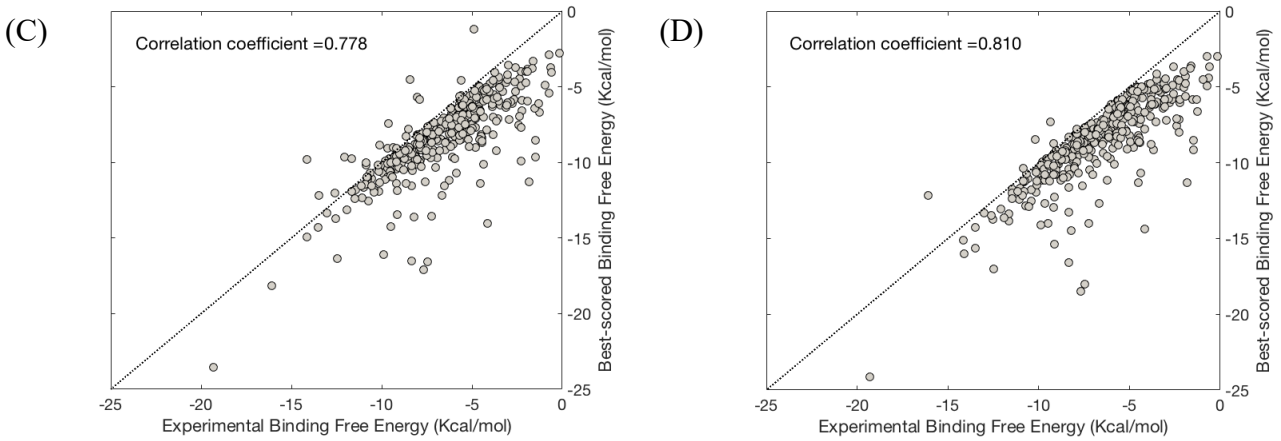


Figure 3. The scatter plots of the experimental binding free energies and best-scored ones for all test cases obtained by (A) MSLDOCK-s2, (B) MSLDOCK-s6, (C) Autodock-d, (D) SODOCK.

Figure 3 shows the scatter plots of the experimental binding free energies and best-scored ones obtained by each docking program for all test cases. The correlation coefficients in Figure 3 indicate that the MSLDOCK-s2 has the best accuracy of binding free energy estimation, followed by MSLDOCK-s6 and SODOCK, and finally Autodock-d. More specifically, the difference between the correlation coefficients of the docking programs which adopt PSO algorithms as their search methods (MSLDOCK-s2, MSLDOCK-s6 and SODOCK) is relatively small. In addition, the distributions of many points in Figure 3(a) and Figure 3(b) are very similar, verifying that the number of sub-swarms in MSLDOCK slightly impacts the binding affinity estimation of the proposed docking program.

Table 1. The statistical results of the best-scored RMSD obtained by all compared algorithms for each dataset and all test cases

	Mean RMSD for PDBbind (Å)	Mean RMSD for GOLD (Å)	Mean RMSD for all[1] (Å)	P-value-s2[2]	P-value-s6[3]	Succ[4]
MSLDOCK-s2	2.423 ± 0.295	2.332 ± 0.422	2.399 ± 0.246		0.076	248
MSLDOCK-s6	2.512 ± 0.293	2.411 ± 0.465	2.485 ± 0.259	0.076		241

Autodock-d	3.359 ± 0.335	3.177 ± 0.519	3.311 ± 0.294	1.309e-29	1.075e-28	180
SODOCK	3.121 ± 0.341	3.273 ± 0.537	3.162 ± 0.305	3.260e-15	6.643e-11	203
Vina	2.545 ± 0.331	2.657 ± 0.514	2.575 ± 0.276	0.450	0.247	247
LeDock	3.270 ± 0.334	4.122 ± 0.597	3.496 ± 0.296	2.767e-12	9.528e-11	166
Glide	2.273 ± 0.295	2.662 ± 0.513	2.376 ± 0.265	0.111	0.050	244

[1] Mean best-scored RMSD and the corresponding bootstrapping errors for all test cases

[2] P-value for the results by the corresponding docking program and MSLDOCK-s2 for all test cases

[3] P-value for the results by the corresponding docking program and MSLDOCK-s6 for all test cases

[4] The number of the test cases for which the corresponding docking method can obtain a best-scored RMSD lower than 2Å

Table 2. The statistical results of the best-sampled RMSD obtained by all compared algorithms for each dataset and all test cases

	Mean RMSD for PDBbind (Å)	**Mean RMSD for GOLD (Å)**	**Mean RMSD for all[1] (Å)**	**P-value-s2[2]**	**P-value-s6[3]**	**Succ[4]**
MSLDOCK-s2	1.061 ± 0.111	1.042 ± 0.135	1.056 ± 0.089		6.883e-04	353
MSLDOCK-s6	1.102 ± 0.134	1.107 ± 0.170	1.103 ± 0.107	6.883e-04		346
Autodock-d	1.627 ± 0.192	1.593 ± 0.274	1.618 ± 0.153	1.089e-22	4.961e-24	293
SODOCK	1.383 ± 0.159	1.556 ± 0.285	1.429 ± 0.142	7.663e-12	4.570e-19	306
Vina	1.101 ± 0.141	1.412 ± 0.305	1.184 ± 0.124	0.299	0.084	346
LeDock	1.212 ± 0.168	2.472 ± 0.347	2.168 ± 0.169	4.274e-33	2.594e-31	230
Glide	1.142 ± 0.160	1.587 ± 0.362	1.260 ± 0.154	0.031	0.144	322

[1] Mean best-sampled RMSD and the corresponding bootstrapping errors for all test cases

[2] P-value for the results by the corresponding docking program and MSLDOCK-s2 for all test cases

[3] P-value for the results by the corresponding docking program and MSLDOCK-s6 for all test cases

[4] The number of the test cases for which the corresponding docking method can obtain a best-sampled RMSD lower than 2Å

In order to evaluate the self-docking accuracy in terms of RMSD, the MSLDOCK-s2 and MSLDOCK-s6 were compared with Autodock-d, SODOCK, Vina, LeDock, and Glide, irrespective of their score or binding free energy. Some statistical results in terms of best-scored RMSD and best-sampled RMSD obtained by these programs are illustrated in Table 1 and Table 2, respectively. The results for the test cases in each dataset reveal that MSLDOCK-s2 and MSLDOCK-s6 are the best two docking programs for almost all criterion in Table 1 and Table 2. The only exception is that for the mean best-scored RMSD results on the PDBbind coreset, Glide is better than the two

MSLDOCK versions. The similar phenomenon can also be observed in the results for all test cases, that is, Glide obtained the smallest best-scored RMSD, followed by two MSLDOCK versions and Vina, and then the other three programs. However, the P-values (calculated by using Wilcoxon signed-rank test, see section 3.3) in Table 1 indicate that there is no significant difference between the results of the mean best-scored RMSD by Glide, Vina and two MSLDOCK versions for all test cases. Moreover, the number of the successful docking test cases (the "Succ" criteria) by these four programs in Table 1 are also very similar, which verifies that the best-scored docking accuracies of MSLDOCK, Vina and Glide are equivalent. On the other hand, the advantages of the two MSLDOCK versions over the others in terms of best-sampled RMSD (in Table 2) are more obvious than those in terms of best-scored RMSD (in Table 1). Although the P-values in Table 2 shows that Vina can be comparable to the two MSLDOCK versions and Glide can get similar results to MSLDOCK-s6 in terms of best-sampled RMSD, the mean RMSD values and the number of successful docking test cases obtained by the two MSLDOCK versions are the first- and second-best results among all the compared programs. Besides, unlike the comparison results based on the binding free energy accuracy obtained by the four Autodock-based programs, the overall results of RMSD by two MSLDOCK versions are significantly better than those by Autodock-d and SODOCK, verifying the effectiveness of the MSLRDPSO algorithm. With respect to the errors of the mean RMSDs in two tables, MSLDOCK-s2 and MSLDCOK-s6 are the two smallest ones and both have relatively large gaps with the others, which indicates that the MSLDOCK program definitely has the best robustness among all compared programs in terms of

RMSD.

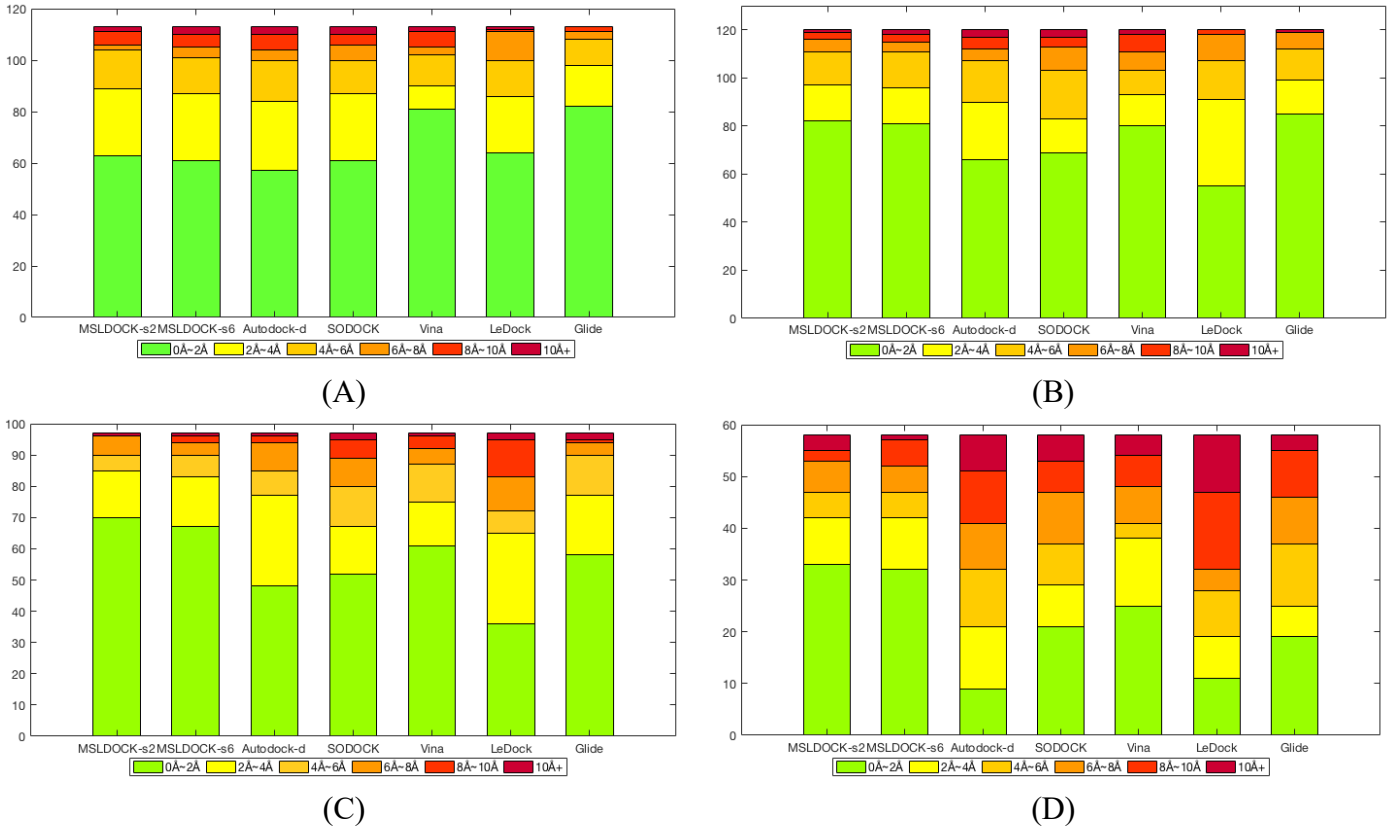
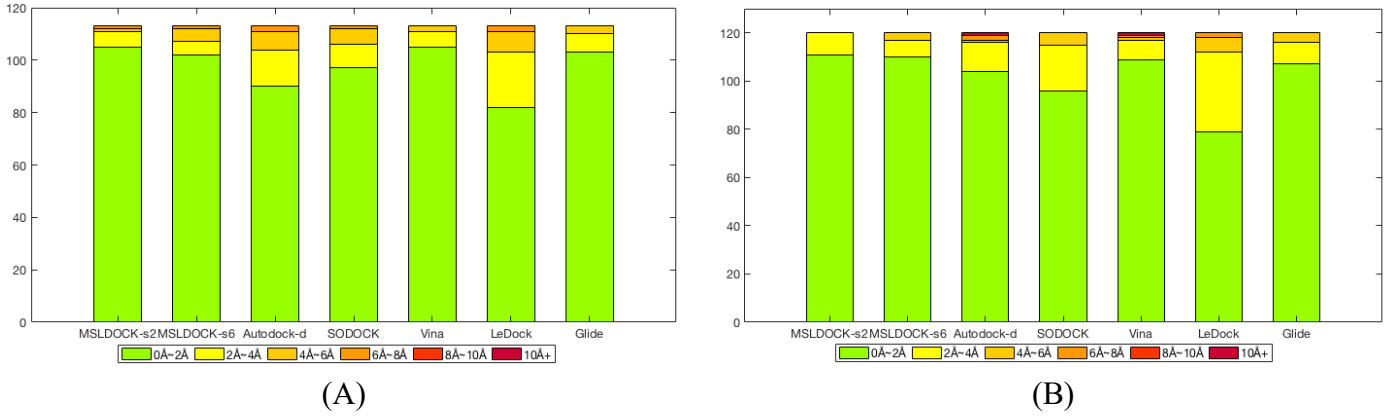
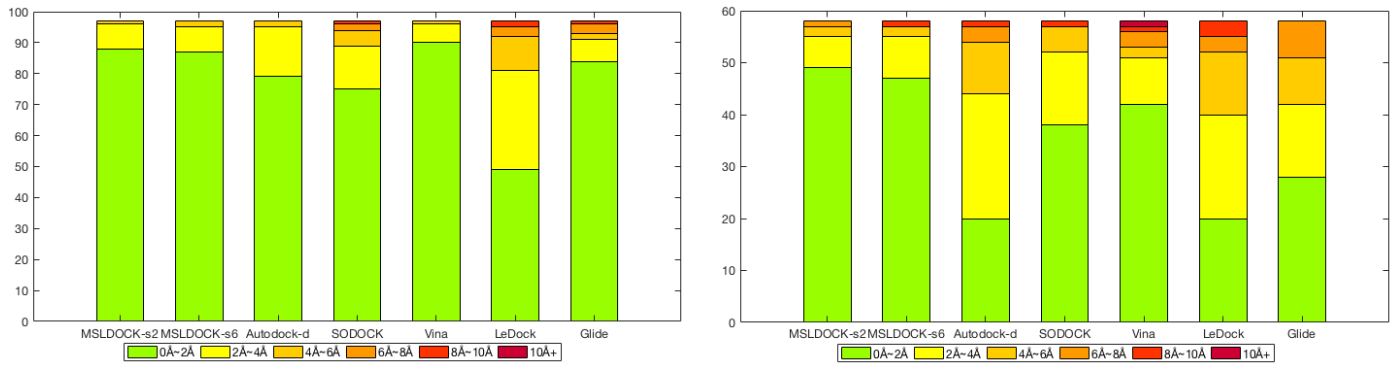


Figure 4. The best-scored RMSD performance in terms of torsions. (A) 113 test cases with torsions between 0 to 4. (B) 120 test cases with torsions between 5 to 7. (C) 97 test cases with torsions between 8 to 12. (D) 58 test cases with torsions between 13 to 36.



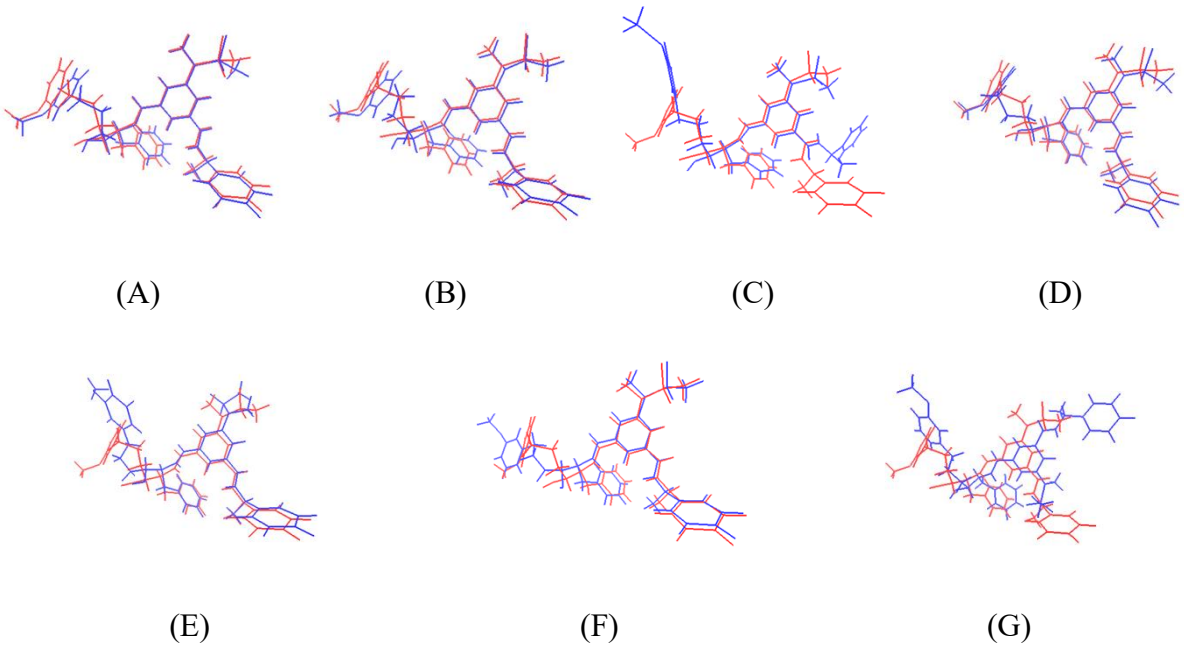


(C) (D)

Figure 5. The best-sampled RMSD performance in terms of torsions. (A) 113 test cases with torsions between 0 to 4. (B) 120 test cases with torsions between 5 to 7. (C) 97 test cases with torsions between 8 to 12. (D) 58 test cases with torsions between 13 to 36.

In addition to the overall results of RMSD, we also analyzed the docking accuracy performance in terms of torsions obtained by these compared docking programs, with the results shown in Figure 4 and Figure 5. For the best-scored RMSD in Figure 4, Glide and Vina may be the best two choices for the test cases no larger than 4 torsions, since they can both find the most successful dockings (RMSD less than 2 Å) among all docking programs, with fewer docking results of large RMSDs. For the same reason, the two versions of MSLDOCK are superior to others for the test cases with more than 7 torsions in Figure 4 and for those with more than 12 torsions in Figure 5. To show the effectiveness of the proposed docking program for highly flexible ligand docking more clearly, we lay out in Figure 6 the best-scored docked ligand conformations found by all compared docking programs for 2vkm with 20 torsions as an example. With respect to all the other torsions' classes in Figure 4 and Figure 5, namely the classes with torsions between 5 to 7 in Figure 4 and the ones with torsions no larger than 12 in Figure 5,

MSLDOCK-s2 and MSLDOCK-s6 are both good choices.



(A) (B) (C) (D)

(E) (F) (G)

Figure 6. The best-scored docked ligand conformations for 2vkm (20 torsions) obtained by all compared docking programs. The red-colored ligand is the conformation from co-crystallized complex and the blue-colored ligand is the docked pose. (A) MSLDOCK-s2 (RMSD equals to 0.66 A°). (B) MSLDOCK-s6 (RMSD equals to 0.84 A°). (C) Autodock-d (RMSD equals to 4.19 A°). (D) SODOCK (RMSD equals to 0.83 A°). (E) Vina (RMSD equals to 2.45 A°). (F) LeDock (RMSD equals to 2.01 A°). (G) Glide (RMSD equals to 5.54 A°).

More specifically, according to the number of the successful dockings in Figure 4, it is clear that both Vina and Glide outperform MSLDOCK when the number of torsions is no larger than 4, while for the larger number of torsions, MSLDOCK is comparable to Vina and Glide (torsions between 4 and 7) or better than them (torsions larger than 7). With respect to the number of the successful dockings in Figure 5, the advantage of two MSLDOCK versions over Vina and Glide is more obvious than that in Figure 4, that is,

MSLDOCK-s2, MSLDOCK-s6, Vina, and Glide are comparable for the torsions no larger than 12, and for the torsions greater than 12, the two versions of MSLDOCK are still better ones. Comparing the results obtained by two versions of MSLDOCK for both best-scored and best-sampled RMSDs, MSLDOCK-s6 can be comparable to MSLDOCK-s2 for the less flexible ligand test cases and is a little worse than MSLDOCK-s2 for the highly flexible ligand ones. With the increase of the number of torsions, the advantage of both these two MSLDOCK versions over the other two Autodock-based programs, i.e., SODOCK and Autodock-d, become more apparent. For LeDock, it can only get comparable results to Autodock in terms of best-scored RMSD, and finds the least successful dockings in almost every class in terms of best-sampled RMSD.

Table 3. Mean time for generating per docking pose taken by each aforementioned docking program (unit: second)

	Mean time for all test cases	Mean time for test cases with torsions no more than 13	Mean time for test cases with torsions more than 13
MSLDOCK-s2	27.193	19.880	68.802
MSLDOCK-s6	30.418	23.114	71.974
Autodock-d	78.897	55.715	210.797
SODOCK	85.147	61.150	221.681
Vina	58.841	30.229	221.637
LeDock	2.444	1.895	5.570
Glide	6.364	2.545	28.096

In order to evaluate the docking efficiency of all aforementioned docking programs, Table 3 lists the mean of computational time for generating per docking pose taken by each docking program. Overall, the MSLDOCK has better docking efficiency than Autodock-d, SODOCK and Vina but worse than LeDock and Glide. The slight advantage

of MSLDOCK-s2 over MSLDOCK-s6 probably caused by the low probability of moving features to other sub-swarms when applying the feature exchange method in MSLDOCK-s2, resulting in the fact that not many features need to be shuffled. Comparing the MSLDOCK versions with the other two Autodock-based programs, we find that it takes by the MSLDOCK versions only approximately 35% of time taken by Autodock-d and SODOCK to execute one docking trial. This is because that the main structure of MSLRDPSO is less complex than those of the other two Autodock-based algorithms: with respect to LGA, in each iteration every particle should exchange its elements from genotype to phenotype[4], which takes much time; for the LPSO, the neighborhood topology[13] makes the search algorithm time-consuming due to its frequent search for the best pbest within each particle's neighborhood. The mean time for all test cases taken by Vina is about twice of those by the two MSLDOCK versions, but it is very different for the test cases with small number of torsions and for those with highly flexible ligands. The reason is that the total steps for searching each docking pose in Vina are associated with the torsion numbers, and thus Vina is forced to do long-step search for these test cases. Glide and LeDock both consume much less time than MSLDOCK, illustrating their good docking efficiency. However, considering the results in terms of RMSD obtained by MSLDOCK, Glide and LeDock, we can conclude that MSLDOCK still can be a competitive docking program due to its superior docking accuracy, especially for the highly flexible ligand docking problems.

Table 4. Statistical results of the computational time for the multithread mode of MSLDOCK-s2 and MSLDOCK-s6 for all test cases

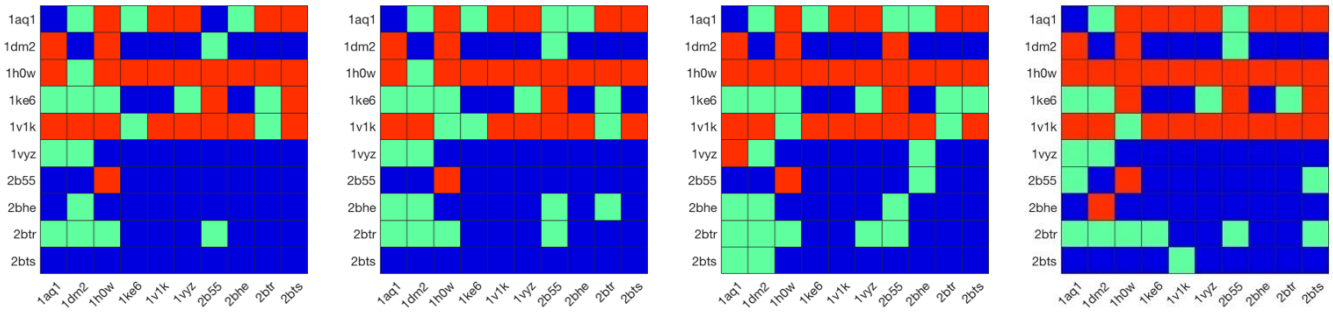
	MSLDOCK-s2	**MSLDOCK-s6**

Mean time of MSLDOCK-M (unit: second)	16.577	5.733
Speedup ratio of MSLDOCK-M relative to MSLDOCK	164.04%	530.55%
Percentage of the real speedup ratio to the desirable ratio for MSLDOCK-M	82.02%	88.43%

Furthermore, some statistics of the computational time for MSLDOCK-M are recorded in Table 4. The speedup ratio cannot be close to the desirable value due to not only the incomplete synchronization between sub-swarms for the parallel mode of MSLRDPSO, but also the parallelization for only the calculation part of the scoring function (no parallelization for other parts of the docking program such as program initialization and analysis of final results). The latter reason also leads to the effect that with a greater number of sub-swarms (number of threads), the scoring function calculation consumes less computational time, and thus the percentages of the real speedup ratio to the desirable ratio for MSLDOCK-M-s6 are a little better than those for MSLDOCK-M-s2 in Table 4.

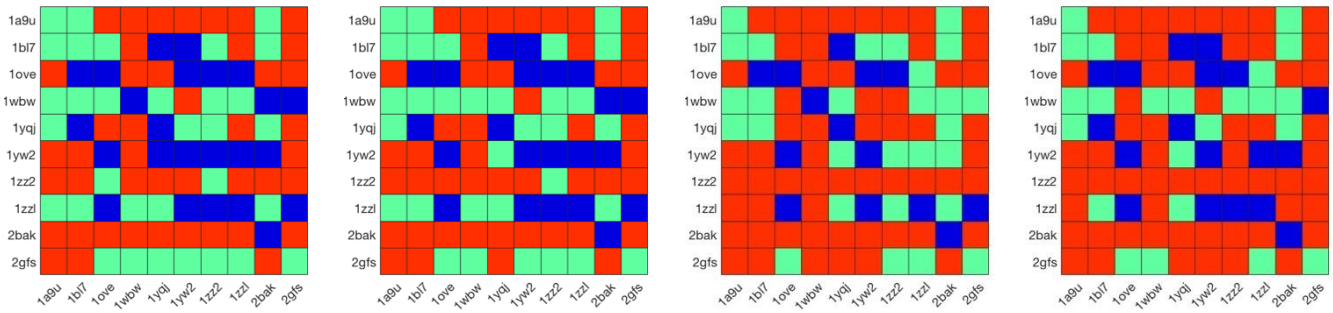
Summarizing the above analysis in terms of binding affinities, docking accuracy and efficiency for self-docking, it can be concluded that the MSLDOCK is better than all the compared programs for most of the evaluation criteria, especially for test cases with highly flexible ligands. The two exceptions are that MSLDOCK performs relatively worse than Glide and Vina in terms of best-scored RMSD for the test cases with small number of torsions, and its docking efficiency is not as good as Glide and LeDock.

4.2 Comparison of cross-docking accuracy



MSLDOCK-s2 MSLDOCK-s6 Autodock-d SODOCK

(A)



MSLDOCK-s2 MSLDOCK-s6 Autodock-d SODOCK

(B)

Figure 7. Cross-docking results obtained by MSLDOCK-s2, MSLDOCK-s6, Autodock-d and SODOCK for (A) CDK2 family and (B) MAPK14 family. Bottom stacked bar plots indicate outcomes for all ligands with a given receptor.

Table 5. The results of cross-docking accuracy for Autodock-based programs

Protein family		MSLDOCK-s2	MSLDOCK-s6	Autodock-d	SODOCK
CDK2	Docking success	54	51	46	48
	Scoring failure	19	24	25	19
	Sampling failure	27	25	29	33
MAPK14	Docking success	24	22	14	18
	Scoring failure	32	32	28	23
	Sampling failure	44	46	58	59

Figure 7 shows the heatmaps of the cross-docking results for two families obtained by four Autodock-based programs. In Figure 7, the matrix rows and columns correspond to a given ligand or receptor and are identified by PDB codes, and the diagonal entries indicate self-docking. Docking outcomes are classified as sampling failure (red), scoring

failure (green) and docking success (blue). Table 5 records the number of the test cases in each class. Overall, MSLDOCK-s2 can get the best results for these two cross-docking families, followed by MSLDOCK-s6 and SODOCK, and Autodock-d is the worst one. This implies that the PSO algorithms may be more suitable in handling cross-docking problems than the genetic algorithms, and the proposed algorithm can get better cross-docking performance than SODOCK which is based on the canonical PSO algorithm. In addition, the little advantage of MSLDOCK-s2 over MSLDOCK-s6 is similar to that in the self-docking results, which demonstrates the MSLDOCK with fewer sub-swarms is more suitable for experiments with a large number of docking repetitions. With respect to the results for each cross-docking family, the superiority of two MSLDOCK versions over Autodock-d and SODOCK for the MAPK14 family are more obvious than that for the CDK2. This indicates that the proposed docking program can obtain good results not only for a relatively easy cross-docking family (CDK2), but also for a hard one (MAPK14).

4.3 Comparison of virtual screening accuracy and screening speed

The ROC curves obtained by all the compared docking programs are illustrated in Figure 8, and the corresponding AUC-ROC values are presented in Table 6. The ROC curves in Figure 8 demonstrate that the Autodock-based docking programs generated very similar ranking list only except the list by SODOCK for kif11. However, the results in Table 6 verify that the two MSLDOCK versions perform better than Autodock-d and SODOCK in terms of the average AUC-ROC values. On the other hand, the results obtained by Vina, LeDock and Glide are different from those obtained by the Autodock-

based programs. Specifically, Vina performs slightly better than Autodock-based programs on cp3a4 and kif11 but shows a little worse performance on ampc and cxcr4. LeDock cannot get a good screening ranking list on ampc, while for other targets it is comparable to other docking methods. Glide is better than the Autodock-based programs according to the ROC curves on almost all targets expect the cxcr4. However, the average AUC-ROC values in Table 6 for the three non-Autodock-based programs indicate that the two MSLDOCK versions are better than Vina and LeDock, and that MSLDOCK-s6 is slightly better than Glide, while MSLDOCK-s2 is a little worse than it.

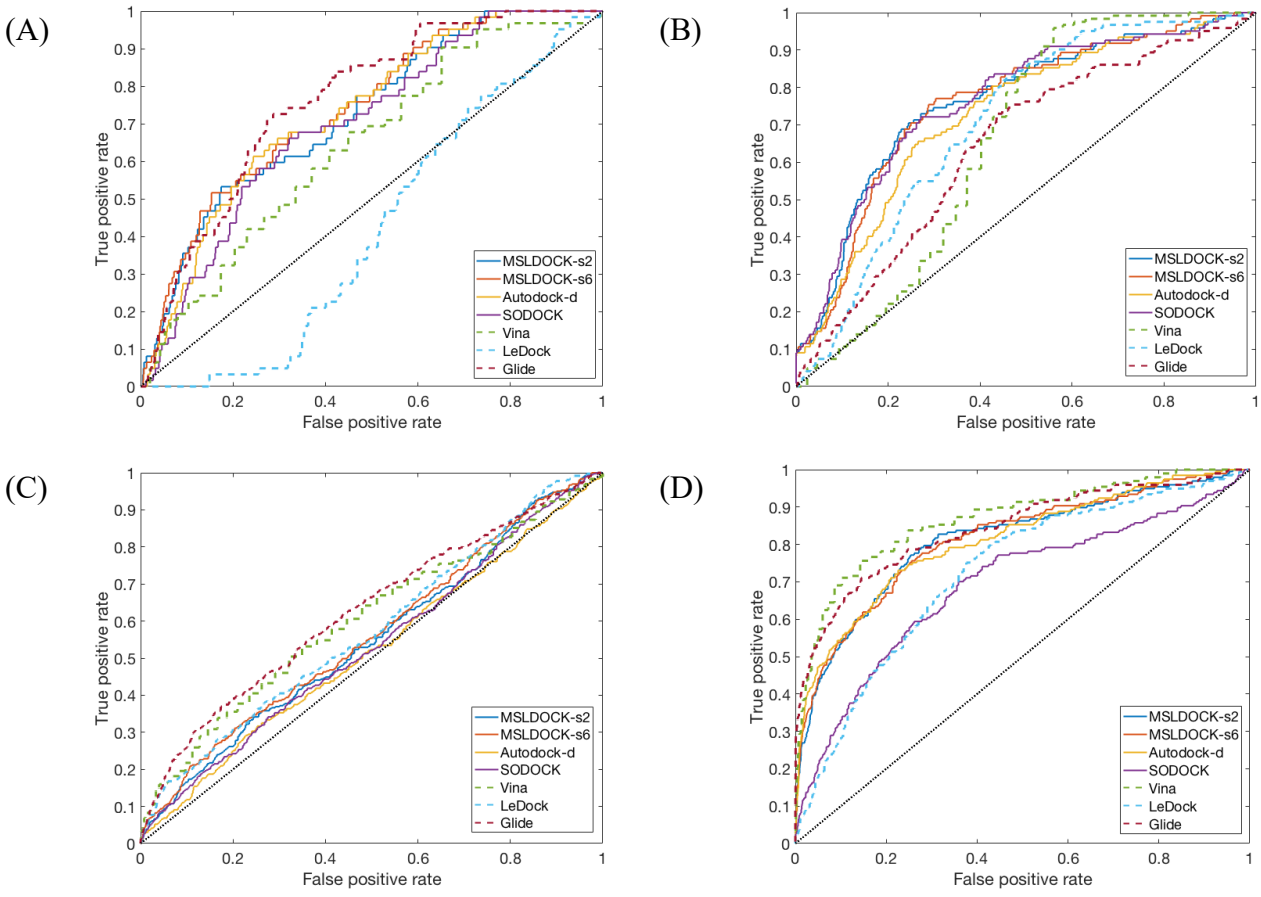


Figure 8. ROC curves of virtual screening four targets using all compared docking programs. (A) ROC curves for ampc. (B) ROC curves for cxcr4. (C) ROC curves for cp3a4. (D) ROC curves for kif11.

Table 6. The AUC-ROC values of virtual screening four targets in DUD-E dataset using all compared programs

	ampc	cxcr4	cp3a4	kif11	Average
MSLDOCK-s2	0.726	0.762	0.548	0.810	0.712
MSLDOCK-s6	0.741	0.759	0.562	0.813	0.719
Autodock-d	0.731	0.729	0.518	0.810	0.697
SODOCK	0.701	0.764	0.533	0.693	0.673
Vina	0.642	0.659	0.608	0.867	0.694
LeDock	0.419	0.708	0.575	0.724	0.607
Glide	0.761	0.641	0.625	0.840	0.717

Table 7. The EF values of virtual screening four targets in DUD-E dataset using all compared programs

		ampc	cxcr4	cp3a4	kif11	Average
MSLDOCK-s2	$EF_{1\%}$	6.374	10.765	2.756	15.756	8.913
	$EF_{10\%}$	3.553	2.948	1.653	5.177	3.333
MSLDOCK-s6	$EF_{1\%}$	4.781	10.765	4.409	15.756	8.928
	$EF_{10\%}$	3.392	2.456	1.791	5.228	3.217
Autodock-d	$EF_{1\%}$	0.000	9.109	2.756	14.739	6.651
	$EF_{10\%}$	2.746	2.620	1.185	5.278	2.957
SODOCK	$EF_{1\%}$	0.000	9.937	2.756	7.116	4.952
	$EF_{10\%}$	2.423	3.275	1.460	3.147	2.576
Vina	$EF_{1\%}$	0.000	0.000	6.963	20.330	6.823
	$EF_{10\%}$	1.938	1.064	1.978	6.446	2.857
LeDock	$EF_{1\%}$	0.000	1.656	3.582	3.558	2.199
	$EF_{10\%}$	0.000	1.474	1.929	2.690	1.523
Glide	$EF_{1\%}$	0.000	4.109	4.973	29.008	9.523
	$EF_{10\%}$	3.061	1.639	2.548	6.090	3.335

With respect to the EF values shown in Table 7, the average results indicate that the Glide program can obtain the best results for both the $EF_{1\%}$ and $EF_{10\%}$ criteria, followed by MSLDOCK-s6 and MSLDOCK-s2, then Vina and Autodock-d, and finally SODOCK and LeDock. The performance gaps between two MSLDOCK versions and Glide are much smaller than those between the two MSLDOCK versions and the other docking programs, which indicates that the proposed docking methods are one of the best screening programs in terms of the EF values. Specifically, the MSLDOCK-s2 and

MSLDOCK-s6 both perform better than all the other docking programs on ampc and cxcr4, and can get the closest results to Vina and Glide on cp3a4 and kif11 among the four Autodock-based programs. It should be pointed out that robustness of the two MSLDOCK versions are better than those of all the other docking programs on the $EF_{1\%}$ criterion, since for the ampc target, MSLDOCK-s2 and MSLDOCK-s6 are able to obtain good results on $EF_{1\%}$ values (6.374 for MSLDOCK-s2 and 4.781 for MSLDOCK-s6, respectively), while for all the other docking programs the $EF_{1\%}$ values are all 0. This is very important in drug discovery research, since people mainly consider the top-ranked ligands from the virtual screening results, and the good robustness of the $EF_{1\%}$ value of the MSLDOCK program can thus lead users to screen the actives very efficiently for many targets.

Comparing both the AUC-ROC and EF values among all the Autodock-based algorithms, we found that MSLDOCK-s6 has the best results for all the average criteria in Table 6 and 7, followed by MSLDOCK-s2, Autodock-d and finally SODOCK. This indicates that the proposed multi-swarm strategy and the random drift mechanism in RDPSO can really improve the virtual screening performance of the optimization algorithms. Moreover, the superiority of the MSLDOCK-s6 over MSLDOCK-s2 on virtual screening accuracy demonstrates that the MSLDOCK with larger number of sub-swarms is more suitable for docking with fewer repetitions due to its higher robustness.

With respect to the screening speed for each target, the mean time of docking per ligand taken by all compared programs are illustrated in Table 8. The average results in Table 8 reveal that the MSLDOCK-s2 and MSLDOCK-s6 has the third and fourth fastest

screening time among all the single-thread docking programs, only a little slower than Glide and LeDock, but much better than the other compared programs, which is similar to the results of docking-speed comparison in self-docking. The high efficiency along with the superior screening accuracy makes the MSLDOCK program to be one of best choices for virtual screening, for it is comparable to Glide but much better than all the other compared docking programs.

Table 8. The mean screening time of docking per ligand for the four targets in DUD-E dataset using all compared programs (unit: second)

	ampc	cxcr4	cp3a4	kif11	Average
MSLDOCK-s2	2.530	4.902	5.536	4.396	4.341
MSLDOCK-s6	2.771	5.180	5.778	4.698	4.607
Autodock-d	6.875	13.759	15.496	12.146	12.069
SODOCK	7.282	14.934	16.904	13.455	13.144
Vina[1]	5.844	11.865	16.359	11.531	11.340
LeDock	2.916	3.642	4.284	3.833	3.669
Glide	0.837	1.893	4.632	1.100	2.116

[1] For all the tested targets in virtual screening, Vina ran in single-threaded mode

5. Conclusions

The MSLDOCK docking program has been proposed in this paper to provide a high-performance and high-efficiency method for solving flexible ligand docking problems. According to the experimental results and corresponding analysis for comparing MSLDOCK with two Autodock-based and three other widely used docking programs on self-docking, cross-docking and virtual screening, some conclusions can be drawn as follows:

- MSLDOCK can be a reliable choice compared to many Autodock-based docking programs, since it is better than the default version of Autodock and a

high-robust and high-accuracy program named SODOCK on almost all compared evaluation criteria for self-docking, cross-docking and virtual screening;

- In terms of both the docking accuracy and docking efficiency, MSLDOCK outperforms Vina, LeDock and Glide in many aspects, especially in self-docking accuracy with highly flexible ligands. Although the docking time taken by MSLDOCK is not as little as that taken by Glide and LeDock, the MSLDOCK program is still considered to be competitive with the non-Autodock-based docking programs;
- For test cases with less flexible ligands (torsions no larger than 4) in self-docking, Vina and Glide perform better than MSLDOCK in terms of the best-scored RMSD. This demonstrates that for some test cases with relatively small number of torsions, MSLDOCK is not the best choice to obtain a good enough RMSD for the best-scored conformation, which is the limitation of MSLDOCK;
- MSLDOCK-s2 is more suitable for docking a single test case with many repetitions. The corresponding examples shown in this paper are that MSLDOCK-s2 outperforms MSLDOCK-s6 in terms of both self-docking and cross-docking accuracies when the repetitions for a single test case is 50. With fewer sub-swarms and enough docking repetitions, MSLDOCK-s2 has stronger ability to find conformations with better best-scored RMSDs than MSLDOCK-s6, especially for highly flexible ligand docking problems.
- MSLDOCK-s6 should be appropriate for docking a single test case with a small

number of repetitions, such as virtual screening. Due to the high robustness of the MSLDOCK with a relatively large number of sub-swarms, MSLDOCK-s6 generally has a higher probability of obtaining good docking results within a few docking repetitions than MSLDOCK-s2.

In light of the above conclusions, we recommend here a selection of the number of sub-swarms for MSLDOCK, in order to obtain good docking results by using MSLDOCK program for most of the flexile ligand docking problems. It is suggested to set 2 sub-swarms for docking a single test case with many repetitions, and to set 6 or more sub-swarms for “fast” docking problems such as virtual screening. It should be pointed out that the default number of sub-swarms is set to 6 in current version of MSLDOCK. The reasons are that this parameter setting can help MSLDOCK obtain remarkable results in the case where only a few docking repetitions can be executed, and that MSLDOCK-s6 can find docking poses with only a little worse RMSDs than MSLDOCK-s2 when there are many repetitions for a single docking test. The source code of MSLDOCK can be downloaded for free from https://github.com/lcmeteor/MSLDOCK.

Additionally, as shown in Table 4, MSLDOCK-M has a good speedup ratio in terms of the number of sub-swarms for running single docking tasks in parallel. This multithread mode of MSLDOCK can be used to complete a docking with a fairly highly flexible ligand in short time, or to quickly find appropriate settings of docking parameters.

In our future work, we intend to evaluate in detail how the change of sub-swarm numbers influences the docking results, or make full use of the multithread mode to

rapidly evaluate the setting of the number of sub-swarms so that MSLDOCK can automatically find the suitable choice for every specific docking problem. We will also further access the docking performance of MSLDOCK in terms of the atom number of ligands, since to some extent, the number of atoms for a ligand can really affect the docking accuracy and efficiency. Besides, we will integrate MSLRDPSO with other scoring functions and/or other docking programs to examine the potential of the proposed search algorithm.

Supporting Information

- The self-docking results obtained by all compared programs for each test case (Table S1); The cross-docking results obtained by the compared Autodock-based programs for each test case (Table S2); The virtual screening results obtained by all compared programs for each test case (Table S3) (ZIP)
- Search performance comparison on MSLDOCK versions with different number of sub-swarms (PDF)

Corresponding Author Information

E-mail: junsun@jiangnan.edu.cn

Notes

The authors declare no competing financial interest.

Acknowledgements

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) López-Camacho, E.; Godoy, M. J. G.; García-Nieto, J., Nebro, A. J.; Aldana-Montes, J. F. Solving Molecular Flexible Docking Problems with Metaheuristics: A Comparative Study. *Appl. Soft Comput.* **2015**, *28*, 379-393.

(2) F Sousa, S.; MFSA Cerqueira, N.; A Fernandes, P.; Joao Ramos, M. Virtual Screening in Drug Design and Development. *Comb. Chem. High Throughput Screen* **2010**, *13*, 442-453.

(3) Heo, L.; Shin, W. H.; Lee, M. S.; Seok, C. GalaxySite: Ligand-Binding-Site Prediction by Using Molecular Docking. *Nucleic Acids Res.* **2014**, *42*, 210-214.

(4) Morris, G. M.; Goodsell, D. S.; Halliday, R. S.; Huey, R.; Hart, W. E.; Belew, R. K.; Olson, A. J. Automated Docking Using a Lamarckian Genetic Algorithm and an Empirical Binding Free Energy Function. *J. Comput. Chem.* **1998**, *19*, 1639-1662.

(5) Trott, O.; Olson, A. J. AutoDock Vina: Improving the Speed and Accuracy of Docking with a New Scoring Function, Efficient Optimization, and Multithreading. *J. Comput. Chem.* **2010**, *31*, 455-461.

(6) Allen, W. J.; Balius, T. E.; Mukherjee, S.; Brozell, S. R.; Moustakas, D. T.; Lang, P. T.; Case, D. A.; Kuntz, I. D.; Rizzo, R. C. DOCK 6: Impact of New Features and Current Docking Performance. *J. Comput. Chem.* **2015**, *36*, 1132-1156.

(7) Zhao H.; Caflisch A. Discovery of ZAP70 Inhibitors by High-Throughput Docking into a Conformation of Its Kinase Domain Generated by Molecular Dynamics. *Bioorganic Med. Chem. Lett.* **2013**, *23*, 5721-5726.

(8) Jones G.; Willett P.; Glen R. C. Molecular Recognition of Receptor Sites Using a Genetic Algorithm with a Description of Desolvation. *J. Mol. Biol.* **1995**, *245*, 43-53.

(9) Friesner, R. A.; Banks, J. L.; Murphy, R. B.; Halgren, T. A.; Klicic, J. J.; Mainz, D. T.; Repasky M. P.; Knoll E. H.; Shelley M.; Perry J. K.; Shaw, D. E.; Francis P.; Shenkin P. S. Glide: A New Approach for Rapid, Accurate Docking And Scoring. 1. Method and Assessment of Docking Accuracy. *J. Med. Chem.* **2004**, *47*, 1739-1749.

(10) Jain A. N. Surflex-Dock 2.1: Robust Performance from Ligand Energetic Modeling, Ring Flexibility, and Knowledge-Based Search. *J. Comput. Aided Mol. Des.* **2007**, *21*, 281-306.

(11) Solis, F. J.; Wets, R. J. B. Minimization by Random Search Techniques. *Math. Oper. Res.*, **1981**, *6*, 19-30.

(12) Namasivayam V.; Günther R. PSO@ AUTODOCK: A Fast Flexible Molecular Docking Program Based on Swarm Intelligence. *Chem. Biol. Drug Des.* **2007**, *70*, 475-484.

(13) Chen, H. M.; Liu, B. F.; Huang, H. L.; Hwang, S. F.; Ho, S. Y. SODOCK: Swarm Optimization for Highly Flexible Protein-Ligand Docking. *J. Comput. Chem.* **2007**, *28*, 612-623.

(14) Liu, Y.; Zhao, L.; Li, W.; Zhao, D.; Song, M.; Yang, Y. FIPSDock: A New Molecular Docking Technique Driven by Fully Informed Swarm Optimization Algorithm. *J. Comput. Chem.* **2013**, *34*, 67-75.

(15) Guan, B.; Zhang, C.; Ning, J. Genetic Algorithm with A Crossover Elitist Preservation Mechanism for Protein-Ligand Docking. *AMB Express* **2017**, *7*, 1-11.

(16) Uehara, S.; Fujimoto, K. J.; Tanaka, S. Protein-Ligand Docking Using Fitness Learning-Based Artificial Bee Colony with Proximity Stimuli. *Phys. Chem. Chem. Phys.* **2015**, *17*, 16412-16417.

(17) Ravindranath, P. A.; Forli, S.; Goodsell, D. S.; Olson, A. J.; Sanner, M. F. AutoDockFR: Advances in Protein-Ligand Docking with Explicitly Specified Binding Site Flexibility. *PLoS Comput. Biol.* **2015**, *11*, e1004586.

(18) López-Camacho E.; García Godoy M. J.; Nebro A. J.; Aldana-Montes, J. F. jMetalCpp: Optimizing Molecular Docking Problems with a C++ Metaheuristic Framework. *Bioinformatics* **2014**, *30*, 437-438.

(19) Liu, Z.; Jiang, D.; Zhang, C.; Zhao, H.; Zhao, Q.; Zhang, B. A Novel Fireworks Algorithm for the Protein-Ligand Docking on the AutoDock. *Mob. Netw. Appl.* **2019**, 1-12.

(20) Korb, O.; Stutzle, T.; Exner, T. E. Empirical Scoring Functions for Advanced Protein-Ligand Docking with PLANTS. *J. Chem. Inf. Model.* **2009**, *49*, 84-96.

(21) Tai, H. K.; Jusoh, S. A.; Siu, S. W. Chaos-Embedded Particle Swarm Optimization Approach for Protein-Ligand Docking and Virtual Screening. *J. Cheminformatics* **2018**, *10*, 1-13.

(22) Wong, K. M.; Tai, H. K.; Siu, S. W. GWOVina: A Grey Wolf Optimization Approach to Rigid and Flexible Receptor Docking. *Chem. Biol. Drug. Des.* **2021**, *97*, 97-110.

(23) Guo, L.; Yan, Z.; Zheng, X.; Hu, L.; Yang, Y.; Wang, J. A Comparison of Various Optimization Algorithms of Protein-Ligand Docking Programs by Fitness Accuracy. *J Mol. Model.* **2014**, *20*, 2251.

(24) Wang, Z.; Sun, H.; Yao, X.; Li, D.; Xu, L.; Li, Y.; Tian S.; Hou, T. Comprehensive Evaluation of Ten Docking Programs on a Diverse Set of Protein-Ligand Complexes: The Prediction Accuracy of Sampling Power and Scoring Power. *Phys. Chem. Chem. Phys.* **2016**, *18*, 12964-12975.

(25) Liang, J. J.; Suganthan, P. N. Dynamic Multi-Swarm Particle Swarm Optimizer. In Proceedings

2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005. IEEE. **2005**, 124-129.

(26) Xu, X.; Tang, Y.; Li, J.; Hua, C.; Guan, X. Dynamic Multi-Swarm Particle Swarm Optimizer with Cooperative Learning Strategy. *Appl. Soft Comput.* **2015**, *29*, 169-183.

(27) Chang, R. I.; Hsu, H. M.; Lin, S. Y.; Chang, C. C.; Ho, J. M. Query-Based Learning for Dynamic Particle Swarm Optimization. *IEEE Access* **2017**, *5*, 7648-7658.

(28) Niu, B.; Zhu, Y.; He, X.; Shen, H. A Multi-Swarm Optimizer Based Fuzzy Modeling Approach for Dynamic Systems Processing. *Neurocomputing* **2008**, *71*, 1436-1448.

(29) Fan, S. K. S.; Chang, J. M. Dynamic Multi-Swarm Particle Swarm Optimizer Using Parallel Pc Cluster Systems for Global Optimization of Large-Scale Multimodal Functions. *Eng. Optim.* **2010**, *42*, 431-451.

(30) Parsopoulos, K. E. Parallel Cooperative Micro-Particle Swarm Optimization: A Master-Slave Model. *Appl. Soft Comput.* **2012**, *12*, 3552-3579.

(31) Gülcü, Ş.; Kodaz, H. A Novel Parallel Multi-Swarm Algorithm Based on Comprehensive Learning Particle Swarm Optimization. *Eng. Appl. Artif. Intell.* **2015**, *45*, 33-45.

(32) Sun, J.; Wu, X.; Palade, V.; Fang, W.; Shi, Y. Random Drift Particle Swarm Optimization Algorithm: Convergence Analysis and Parameter Selection. *Mach. Learn.* **2015**, *101*, 345-376.

(33) Clerc, M.; Kennedy, J. The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space. *IEEE Trans. Evol. Comput.* **2002**, *6*, 58-73.

(34) Omar, M.A. Elementary Solid State Physics: Principles and Applications. Addison-Wesley,1993.

(35) Wang, R.; Fang, X.; Lu, Y.; Wang, S. The PDBbind Database: Collection of Binding Affinities for Protein-Ligand Complexes with Known Three-Dimensional Structures. *J. Med. Chem.* **2004**, *47*, 2977-2980.

(36) Su M.; Yang Q.; Du Y.; Feng G.; Liu Z.; Li Y.; Wang R. Comparative Assessment of Scoring Functions: The CASF-2016 Update. *J. Chem. Inf. Model.* **2019**, *59*, 895-913.

(37) Nissink, J. W. M.; Murray, C.; Hartshorn, M.; Verdonk, M. L.; Cole, J. C.; Taylor, R. A New Test Set for Validating Predictions of Protein-Ligand Interaction. *Proteins*. **2002**, *49*, 457-471.

(38) Thomsen, R.; Christensen, M. H. MolDock: A New Technique for High-Accuracy Molecular Docking. *J. Med. Chem.* **2006**, *49*, 3315-3321.

(39) Sutherland, J. J.; Nandigam, R. K.; Erickson, J. A.; Vieth, M. Lessons in Molecular Recognition. 2. Assessing and Improving Cross-Docking Accuracy. *J. Chem. Inf. Model.* **2007**, *47*, 2293-2302.

(40) Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov I. N.; Bourne, P. E. The Protein Data Bank. *Nucleic Acids Res.* **2000**, *28*, 235-242.

(41) Mysinger, M. M.; Carchia, M.; Irwin, J. J.; Shoichet, B. K. Directory of Useful Decoys, Enhanced (DUD-E): Better Ligands and Decoys for Better Benchmarking. *J. Med. Chem.* **2012**, *55*, 6582-6594.

(42) Nguyen, N. T.; Nguyen, T. H.; Pham, T. N. H.; Huy, N. T.; Bay, M. V.; Pham, M. Q.; Nam, P. C.; Vu, V. V.; Ngo, S. T. Autodock Vina Adopts More Accurate Binding Poses but Autodock4 Forms Better Binding Affinity. *J. Chem. Inf. Model.* **2019**, *60*, 204-211.

(43) Hill A.D.; Reilly P.J. Scoring Functions for AutoDock. In *Glycoinformatics*; Lütteke T., Frank M., Eds.; Humana Press: New York, 2015; pp. 467-474.

(44) Dolezal, R.; Ramalho, T. C.; França, T. C.; Kuca, K. Parallel Flexible Molecular Docking in Computational Chemistry on High Performance Computing Clusters. In *Computational Collective Intelligence*. Springer, Cham, 2015; pp 418-427.

(45) Eldridge, M. D.; Murray, C. W.; Auton., T. R.; Paolini, G. V.; Mee, R. P. Empirical Scoring Functions: I. The Development of a Fast Empirical Scoring Function to Estimate The Binding Affinity

of Ligands in Receptor Complexes. *J. Comput. Aided Mol. Des.* **1997**, *11*, 425–445.

(46) Forli, W.; Halliday, S.; Belew, R.; Olson, A. J. AutoDock Version 4.2. **2012**.

(47) Hauser A. S.; Windshügel B. LEADS-PEP: A Benchmark Data Set for Assessment of Peptide Docking Performance. *J. Chem. Inf. Model.* **2016**, *56*, 188-200.

(48) Devaurs, D.; Antunes, D. A.; Hall-Swan, S.; Mitchell, N.; Moll, M.; Lizée, G.; Kavraki, L. E. Using Parallelized Incremental Meta-Docking Can Solve the Conformational Sampling Issue When Docking Large Ligands to Proteins. *J. Mol. Cell Biol*. **2019**, *20*, 42.

(49) Hawkins, P. C. D.; Warren, G. L.; Skillman, A. G.; Nicholls, A. How to Do an Evaluation: Pitfalls and Traps. *J. Comput. Aided Mol. Des*. **2008**, *22*, 179–190.

(50) Taheri, S. M.; Hesamian, G. A Generalization of The Wilcoxon Signed-Rank Test and Its Applications. *Stat. Pap*. **2013**, *54*, 457-470.

(51) Verdonk, M. L.; Mortenson, P. N.; Hall, R. J.; Hartshorn, M. J.; Murray, C. W. Protein-Ligand Docking Against Non-Native Protein Conformers. *J. Chem. Inf. Model.* **2008**, *48*, 2214-2225.

Table of Contents graphic