Supporting Information

Flexible CDOCKER: Hybrid Searching Algorithm and Scoring Function with Side Chain Conformational Entropy

Yujin Wu† and Charles L. Brooks III*,†,‡

†*Department of Chemistry, University of Michigan, Ann Arbor, 48109, Michigan, USA*

‡*Biophysics Program, University of Michigan, Ann Arbor, 48109, Michigan, USA*

E-mail: brookscl@umich.edu

MD Based Simulated Annealing Parameters

The parameters for the MD based simulated annealing for both flexible docking and rigid docking are listed in the Table S1.

Table S1: Soft-core potentials used in rigid and flexible receptor docking.

name	$E^*_{\max}$(vdw)	$E^*_{\max}$(att)	$E^*_{\max}$(rep)
soft-core potential I	0.6	-0.4	8.0
soft-core potential II	3.0	-20.0	40.0
soft-core potential III	15.0	-120.0	-2.0
soft-core potential IV	10000	-10000	10000

*$E_{\max}$(vdw), $E_{\max}$(att) and $E_{\max}$(rep) in the unit of kcal/mol are parameters for the van der Waals, electrostatic attractive, and electrostatic repulsive interactions, respectively. In flexible receptor docking, these soft core interactions were used both for the rigid receptor region of the protein target as well as the flexible side chains.

Datasets for Docking

PDB Entries Used in Cross-docking Experiments

The complete list of the protein and RNA PDB codes are included in the Table S2. For each complex used in this study, the receptor flexible residues are listed in the supplementary documentary (flexible-residues.xlsx).

Table S2: Protein and RNA codes used for cross-docking experiment.

PDE10A							
2OUN	2OUQ	2OUR	2OUY	3SN7	3SNI	3UI7	3WI2
3WS8	3WS9	4FCB	4FCD	4LLJ	4LLX	4LM1	4LM2
4LM4	4MRW	4MRZ	4MS0	4MSA	4MSC	4MSE	4MSH
4MSN	4WN1	4XY2	4YQH	4YS7	5C1W	5C28	5C29
5C2A	5C2E	5C2H	5DH5	5K9R	5XUJ	5ZNL	6IJH
6KDX	6KE0	6MSA	6MSC				
T4 L99A							
181L	182L	183L	184L	185L	186L	187L	188L
1NHB	2OU0	2RAY	2RAZ	2RB0	2RB1	3DMZ	3DN0
3DN1	3DN3	3DN6	4W53	4W55	4W58	4W59	
T4 M102Q							
1LGW	1LGX	1LI2	1LI3	1LI6	1OV7	1OVH	1OVJ
1OWY	1OWZ	2RBN	2RBR	2RBS	3HT7	3HT8	3HTB
3HTF	3HU8	3HUA	5JWT	5JWV			
DHFR							
1AOE	1DG5	1DR3	1IA1	1BOZ	1DG7	1IA2	2CD2
3DFR	1DLR	1HFP					
Thrombin							
1A4W	1A61	1C5Z	1D3D	1D4P	1D6W	1FPC	1G32
1GHW	1GI9	1GJ4	1GJ5	1K1I	1K22		
Riboswitch							
2XO1	2XO0	2XNW	2XNZ				

Binders and Non-binders

The binders and non-binders for the T4 L99A and T4 L99A/M102Q systems are summarized by the Shoichet group and can be found at `http://www.bkslab.org/projects/model\_systems`.

RNA Side Chain Dihedral Angle

The increasing number of RNA crystal structures enables a structure-based approach to the discovery of new RNA-binding ligands. We define the sugar and phosphate group in the nucleotide monophosphate as the fixed backbone of RNA and base is considered as the flexible side chain. Thus, only one dihedral angle χ is presented in all 4 bases. Instead of using IUPAC-IUB convention, we rotate the dihedral angle one degree at a time and record the total energy for each nucleotide monophosphate. This result is shown in Figure S1 and the conformational states of the determined by nucleotide monophosphate dihedral angle are summarized in Table S3.

Table S3: Conformational State Determined by Dihedral Angle, χ

Nucleotide monophsphate	State 1	State 2
ADE, GUA	$-91 \sim 175^\circ$	$-180 \sim -91^\circ$ or $175 \sim 180^\circ$
URA, CYT	$-170 \sim 137^\circ$	$-180 \sim -170^\circ$ or $137 \sim 180^\circ$

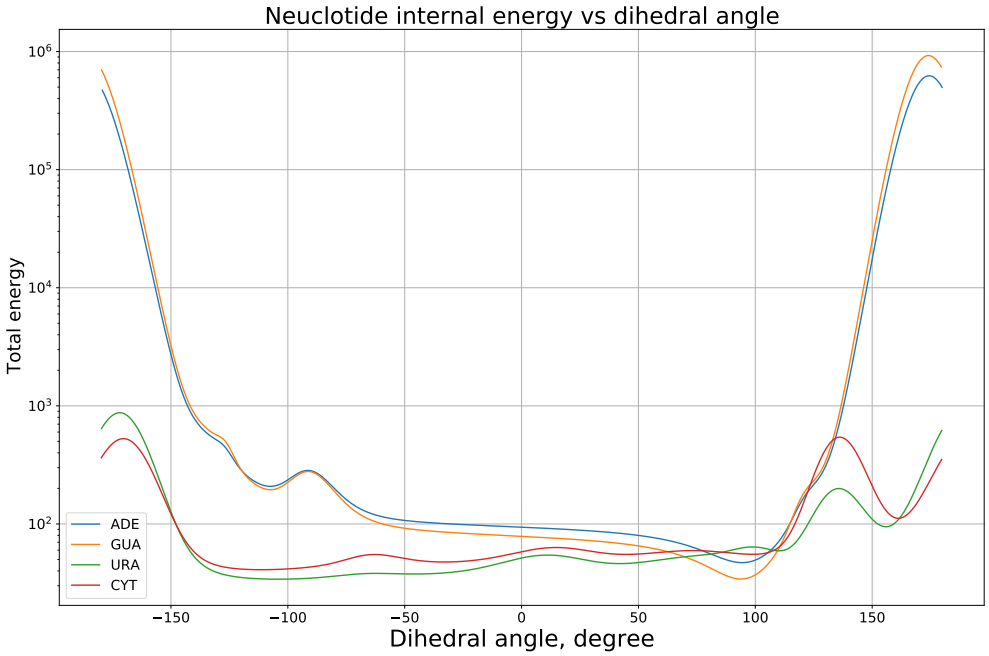

Figure S1: Total energy of nucleotide monophosphate as a function of dihedral angle. **(A)** The y-axis is in logistic scale. **(B)** Large total energy value is resulted from unfavored dihedral angle.

Pose Prediction Accuracy for different RMSD Cutoffs

Cumulative pose prediction accuracy with different RMSD radius cutoff for Flexible CDOCKER, Rigid CDOCKER and AutoDock Vina are plotted in Figures S2, S3 and S4

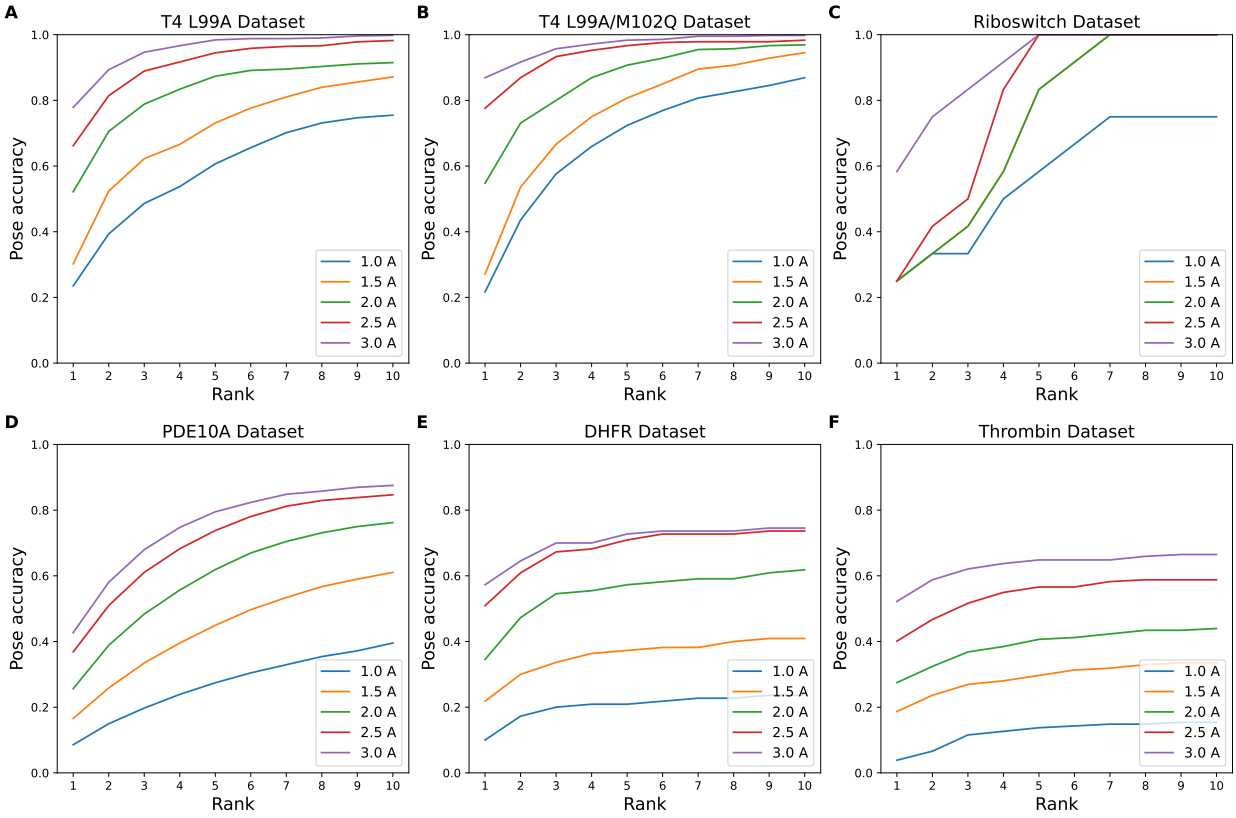

Figure S2: Cumulative pose prediction accuracy for the (A) T4 L99A dataset, the (B) T4 L99A/M102Q dataset, the (C) Riboswitch dataset, the (D) PDE10A dataset, the (E) DHFR dataset and the (F) Thrombin dataset using Flexible CDOCKER. RMSD cutoff is set to be 1, 1.5, 2, 2.5 and 3 Å. A rank of N means the correct docking pose is within the top N solutions.

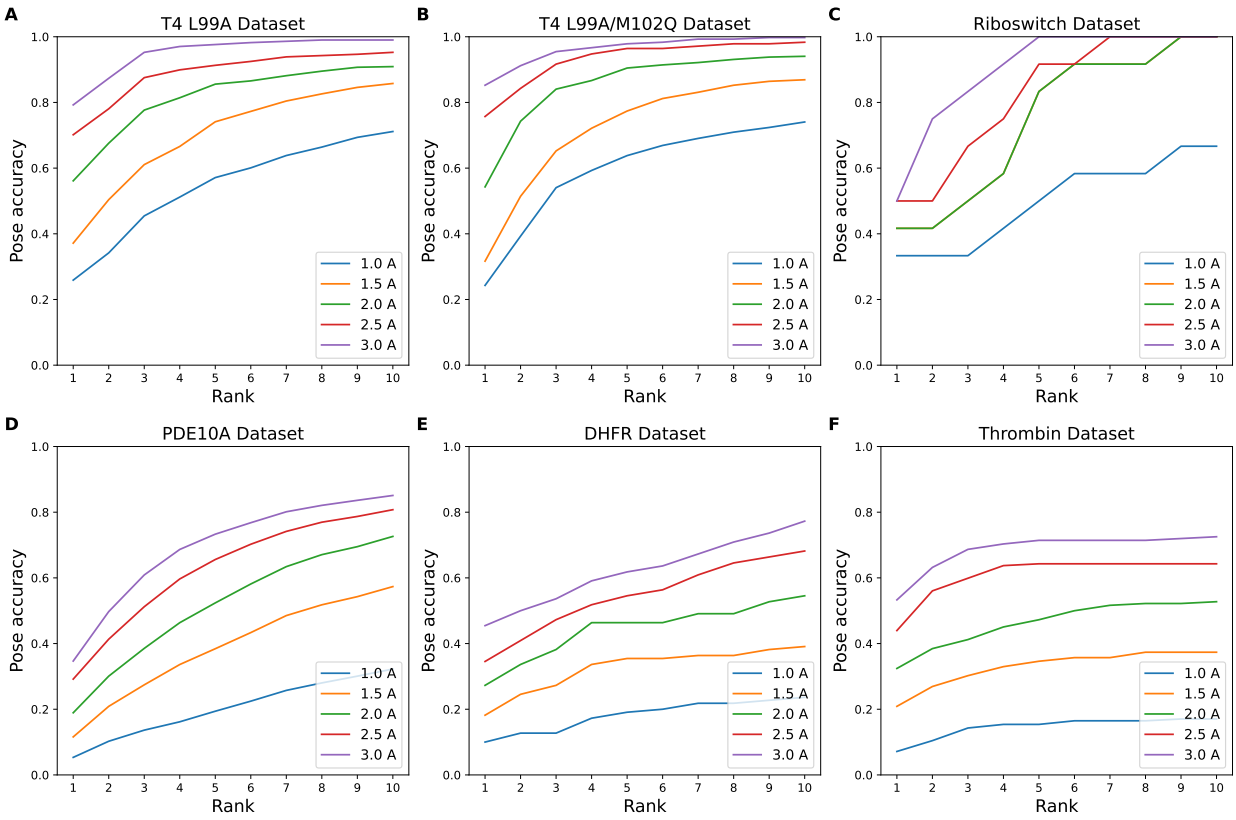

Figure S3: Cumulative pose prediction accuracy for the (A) T4 L99A dataset, the (B) T4 L99A/M102Q dataset, the (C) Riboswitch dataset, the (D) PDE10A dataset, the (E) DHFR dataset and the (F) Thrombin dataset using Rigid CDOCKER. RMSD cutoff is set to be 1, 1.5, 2, 2.5 and 3 Å. A rank of N means the correct docking pose is within the top N solutions.

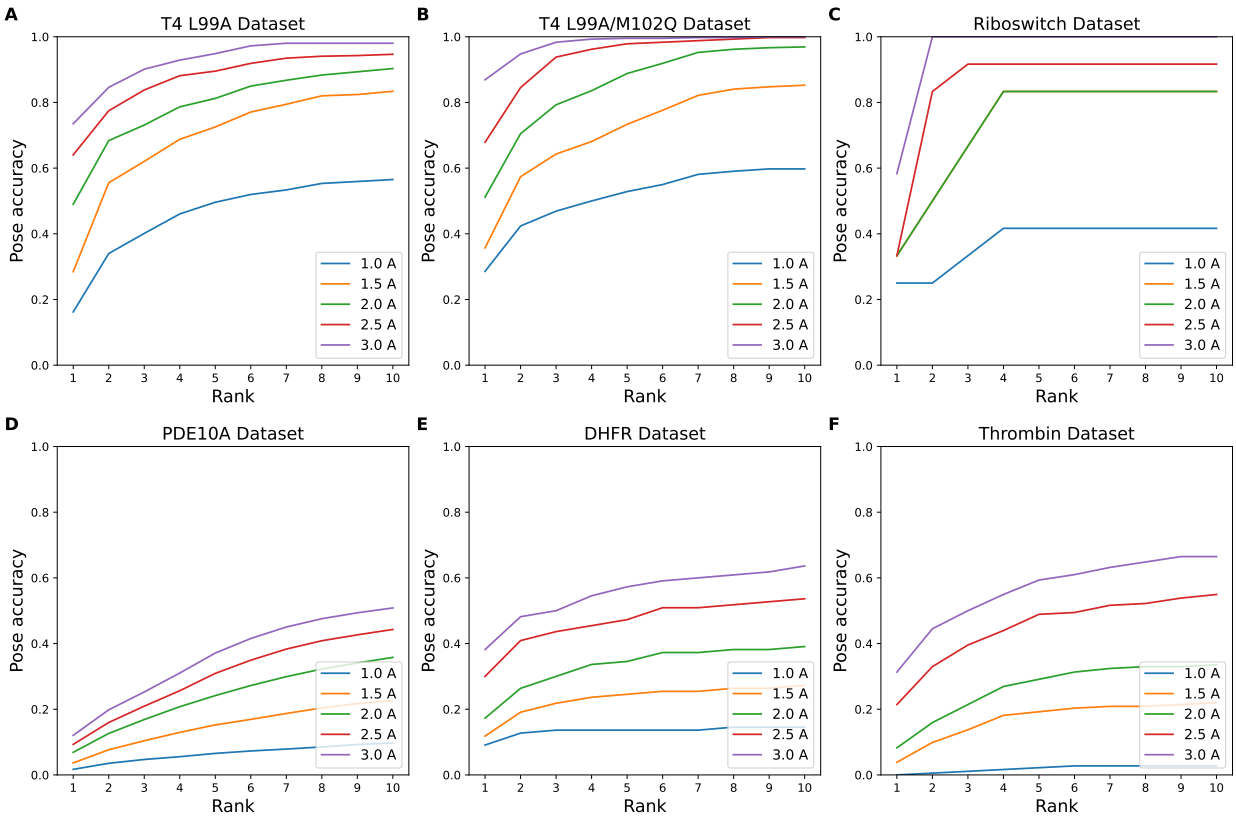

Figure S4: Cumulative pose prediction accuracy for the (A) T4 L99A dataset, the (B) T4 L99A/M102Q dataset, the (C) Riboswitch dataset, the (D) PDE10A dataset, the (E) DHFR dataset and the (F) Thrombin dataset using AutoDock Vina. RMSD cutoff is set to be 1, 1.5, 2, 2.5 and 3 Å. A rank of N means the correct docking pose is within the top N solutions.

Flexible Docking Setup and Results for the DUD-E Subsets

Three target receptors from the DUD-E datasets are selected to test the performance of virtual screening. The flexible side chains are determined based on the ligand and receptor structure provided in the DUD-E sets (4 Å cutoff). The number of flexible side chains for MCR, GCR and ANDR is 17, 17 and 15 respectively. The selected flexible side chains are listed in Table S4. Since multiple copies of the flexible atoms (receptor flexible side chains and ligand atoms) are stored in the GPU for parallel simulated annealing. Because of larger number of flexible side chains in these three receptors, using 500 docking trials per generation might exceed GPU memory size for large ligands and cause abnormal termination of docking. Thus, we reduced the docking trials to 300 per generation for each ligand.

Table S4: Flexible Side Chains Selection for Receptor Targets MCR, GCR and ANDR

Receptor	Flexible side chains
MCR	LEU766, LEU769, ASN770, ALA773, GLN776, TRP806, MET807, SER810, ARG817, PHE829, MET845, MET852, PHE941, CYS942, THR945, VAL954, PHE956
GCR	MET36, LEU39, ASN40, LEU42, GLN46, MET77, MET80, ALA83, LEU84, ARG87, PHE99, GLN118, MET122, TYR211, THR215, ILE223, PHE225
ANDR	LEU701, ASN705, LEU707, TRP741, MET742, MET745, MET749, ARG752, PHE764, MET780, LEU873, PHE876, THR877, PHE891, MET895

Because the enolate moiety and halogen atoms within 5 member aromatic rings are not supported by ParamChem, a small number of the compounds were not successfully parameterized. The number of actives and decoys used in the virtual screening test are listed in Table S5. Total number of actives and decoys used in the original DUD-E test are listed in the parentheses. The receiver operating characteristic ROC curve is plotted in Figure S5.

Table S5: Number of actives and decoys used in the virtual screening test

Receptor Name	Number of actives	Number of decoys
MCR	84 (94)	5089 (5150)
GCR	238 (258)	14711 (15000)
ANDR	233 (269)	14166 (14350)

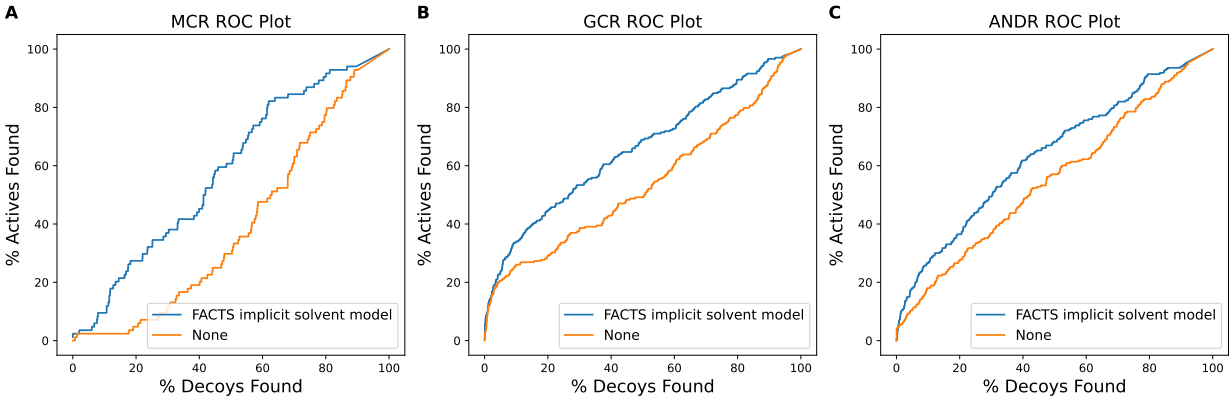

Figure S5: ROC curve for receptor target (A) MCR, (B) GCR and (C) ANDR.

Scripts and Files

We provide the essential CHARMM scripts in the folder Dataset/ to reproduce the flexible receptor docking results. There is also an example of flexible receptor docking of the GCR in the folder ./Dataset/Example/. Inside the folder Example/, the folder pdb/ and str/ contain the ligand PDB, topology and paramter files. The file list under the folder run/ is the name of compounds used for docking. To run flexible docking, one just needs to copy the slurm file Script/dock/dock.slm to the folder run/ and submit the job request, assuming a slurm-based queuing environment is being used. The docked poses are saved in the folder run/cd_result/. The file Script/dock/ener.slm will use FACTS implicit solvent model (ener.inp) to score the docked pose. If one wants to dock more compounds, one just needs to append the list file with the compound name and add the corresponding PDB, topology and parameter files to the folder pdb/ and str/.

Docking Workflow

Minimize Ligand in Vacuum Before Docking

Ligands are minimized in vacuum using CHARMM before the docking experiments with the following script:

```
!! Minimization
mini sd nstep 50 tolenr 0.01
mini abnr nstep 50 tolenr 0.005
```

Flexible Receptor Docking Procedures

The general flexible receptor docking workflow is performed using the following procedure:

1. Preparation of receptor flexible side chains.
2. Ligand initial placement.
3. Generate the intermediate generation based on the results from previous generation.
4. Generate the next generation based on the intermediate generation with mutation operator described in this study.

This section shows the key scripts developed and used in this study.

1. Preparation of receptor flexible side chains The first step of flexible receptor docking is to define flexible side chains. For each receptor-ligand holo complex in a cross-docking dataset with N crystal structures, we have only one unique definition of the flexible side chains for each holo-complex based on the distance between ligand heavy atoms and the corresponding own target structure side chain atoms. We considered the receptor side chains with at least one pairwise interaction within 4 Å of any of the ligand heavy atoms as flexible side chains. The following CHARMM script shows the selection of 17 flexible side chains used for the flexible receptor docking experiment of GCR.

```
!! Define binding pocket
define flexchain select -
       ( resid  36 -
       .or. resid  39 -
       .or. resid  40 -
       .or. resid  42 -
       .or. resid  46 -
       .or. resid  77 -
       .or. resid  80 -
       .or. resid  83 -
       .or. resid  84 -
       .or. resid  87 -
       .or. resid  99 -
       .or. resid  118 -
       .or. resid  122 -
       .or. resid  211 -
       .or. resid  215 -
       .or. resid  223 -
       .or. resid 225 ) end
```

In the real application, we note that one could incorporate more structures with varied ligand interfaces to the receptor if they were available, thereby expanding the size of the flexible side chain region to accommodate the additional knowledge about the targeted receptor. If one does not have any knowledge of the bound state, one could easily define the flexible side chain selection through CHARMM. One could manually define or change the flexible side chains through CHARMM (i.e., change residue id in the CHARMM script).

2. Ligand initial placement After defining receptor flexible residues, the next step of docking is the ligand initial placement. We use Open Babel (obrotamer). After a random

conformer is generated, a random translation (maximum ±2 Å) and rotation (maximum 360∘) is performed on this random conformer. The following CHARMM script shows how to utilize Open Babel and CHARMM script to generate ligand random initial coordinates.

```
!! Generate ligand random conformer with Open Babel
system " obrotamer ./ligand.pdb | convpdb.pl -segnames | sed s/PRO0/LIGA/g
> ligand_rotamer.pdb "
read coor pdb name "ligand_rotamer.pdb" resid

!! rotation
calc theta ( ?RANDOM - 0.5 ) * ( ?PI )
calc phi ( ?RANDOM ) * ( ?PI ) * 2
calc XDIR cos(@theta) * cos(@phi)
calc YDIR cos(@theta) * sin(@phi)
calc ZDIR sin(@theta)
calc rphi ( ?RANDOM ) * 360
coor rotate xdir @XDIR ydir @YDIR zdir @ZDIR -
xcen @xcen ycen @ycen zcen @zcen -
phi @rphi select segid LIGA end

!! translation
coor stats select segid LIGA end
calc xtran ( ?RANDOM - 0.5 ) * 2 * 2
calc ytran ( ?RANDOM - 0.5 ) * 2 * 2
calc ztran ( ?RANDOM - 0.5 ) * 2 * 2
coor trans xdir @xtran ydir @ytran zdir @ztran select segid LIGA end
```

An energy cutoff is applied to filter out significant collisions between ligand atoms and

protein atoms due to the random translation and rotation. This process is repeated 500 times to create the initial generation. The complete CHARMM script for the ligand initial placement and minimizing the initial generation is located at ./Script/dock/initial_generation.inp.

3. Generate the intermediate generation based on the results from previous generation After the initial (previous) generation is generated and minimized with molecular dynamic (MD) based simulated annealing, we create the intermediate generation by performing crossover by swapping the ligand-receptor pair. The probability of selecting a given parent is using a roulette-wheel selection method and is based on the population of the ligand clusters After selecting a pair of parents, the probability for this pair of parents to undergo crossover is 0.5. The following python script ultilize python random.choices function to create the intermediate generation.

```
## Import module
import random
import numpy as np
import pandas as pd

################################################################
## Create ligand - protein pairs for next generation
## Top 10 largest cluster
## Combination probability : 0.50
## Generated list is saved as a csv file
## Parameters :
##     count             --> weight of index (population)
##     total             --> total protein ligand number
##     comb_pair         --> pairs for recombination
##     self_pair         --> paris for no recombination
##     tmp1              --> list 1 for recombination
```

```
##     tmp2              --> list 2 for recombination
##     tmp3              --> list for no recombination
#####################################################################

## Read in data
count = np.loadtxt("count", dtype = int)
comb_pair = np.loadtxt("pairs", dtype = int)
self_pair = comb_pair * 2
total = comb_pair * 4
num = np.shape(count)[0]
idx = np.arange(0, num) + 1

## Generate pair list
tmp1 = random.choices(idx, weights = count, k = comb_pair)
tmp2 = random.choices(idx, weights = count, k = comb_pair)
tmp3 = random.choices(idx, weights = count, k = self_pair)

## Create protein ligand list
total_list = np.zeros((total, 2))

for i in np.arange(comb_pair):
     j = 2 * i
     k = 2 * i + 1
     l = self_pair + j
     m = self_pair + k

     total_list[j, 0] = tmp1[i]
```

```
    total_list[k, 0] = tmp2[i]
    total_list[l, 0] = tmp3[j]
    total_list[m, 0] = tmp3[k]

    total_list[j, 1] = tmp2[i]
    total_list[k, 1] = tmp1[i]
    total_list[l, 1] = tmp3[j]
    total_list[m, 1] = tmp3[k]

    i+=1

## Save all data to a csv file
pair_list = pd.DataFrame(total_list, columns = ['Protein', 'Ligand'], dtype = int)
pair_list.to_csv('pair.csv', sep = '')
```

4. Generate the next generation based on the intermediate generation with mutation operator described in this study. After generating the intermidate generation, the second intensification is mutation. The mutation operator is defined as randomly translating (maximum ±2 Å) and rotating (maximum 30∘) the ligand. The probability of mutation for a given individual in the intermediate generation is a function of the difference between the total energy of that individual and the largest total energy value among the 10 parents. The following CHARMM script shows how to generate the next generation of the docking poses based on the intermediate generation created by previous python script. The complete CHARMM script for the ligand initial placement and minimizing the initial generation is located at ./Script/dock/next_generation.inp.

```
!! Looping protein - ligand pairs
set idxnum = 1       ! index of receptor-ligand pair.
set numcopy = 500    ! total number of individuals.
```

```
label initialPose

!! Brief minimization of ligand and flexible side chain with swaping pairs
read coor pdb name "ligand/@IDXNUM.pdb" resid
read coor pdb name "protein/@IDXNUM.pdb" resid
mini sd nstep 50 tolenr 0.01
mini abnr nstep 50 tolenr 0.005
write coor card select segid LIGA end name "ligand_rotamer.crd"
write coor card select all .and. (.not. segid LIGA) end name "tmpprotein.crd"
calc DEner = ?ener - @totalener

!! Compute energy cutoff
!! Mutation basec on different probility (prob)
if @DEner .le. 0 then
   calc prob = 1 - 0.5 * exp(@DEner)
else
   calc prob = 0.5 * exp(-@DEner)
   endif

!! Dimensional analysis
coor stats select segid LIGA end
set xcen ?XAVE
set ycen ?YAVE
set zcen ?ZAVE

!! No mutation
if ?random .le. @prob then
```

```
   incr idxnum by 1
   if @idxnum .le. @numcopy goto initialPose

!! Mutation
else
   label MutationPose
   read coor select segid LIGA end card name "ligand_rotamer.crd"
   read coor select all .and. (.not. segid LIGA) end card name "tmpprotein.crd"

   ! rotation
   calc theta ( ?RANDOM - 0.5 ) * ( ?PI )
   calc phi ( ?RANDOM ) * ( ?PI ) * 2
   calc XDIR cos(@theta) * cos(@phi)
   calc YDIR cos(@theta) * sin(@phi)
   calc ZDIR sin(@theta)
   calc tmpphi ( ?RANDOM ) * 360
   if @tmpphi .lt. 180 then
      calc rphi @tmpphi / 6
   else
      calc rphi 390 - @tmpphi / 6
      endif
   coor rotate xdir @XDIR ydir @YDIR zdir @ZDIR -
   xcen @xcen ycen @ycen zcen @zcen -
   phi @rphi select segid LIGA end

   ! translation
   coor stats select segid LIGA end
```

```
   calc xtran ( ?RANDOM - 0.5 ) * 2 * 2
   calc ytran ( ?RANDOM - 0.5 ) * 2 * 2
   calc ztran ( ?RANDOM - 0.5 ) * 2 * 2
   coor trans xdir @xtran ydir @ytran zdir @ztran select segid LIGA end

   if @idxnum .le. @numcopy goto initialPose
   endif
```

Minimization with the FACTS Implicit Solvent Model

The docked poses are rescored with the FACTS implicit solvent model with a short minimization to better estimate each enthalpy term in the scoring function with the following script. The complete CHARMM script for is located at ./Script/dock/ener.inp.

```
!! Read in docked pose
read coor pdb name "docked_pose/ligand_@I.pdb" resid
read coor pdb name "docked_pose/protein_@I.pdb" resid

!! Set FACTS implicit solvent model parameter
faster on
nbond nbxmod 5 atom cdiel eps 1 shift vatom vdistance vswitch -
      cutnb 14.0 ctofnb 12.0 ctonnb 10.0 e14fac 1.0 wmin 1.5
scalar wmain = radius
facts tcps 22 teps 1 gamm 0.015 tavw conc 0.1 temp 298

!! FACTS MD
cons fix select all .and. ( .not. ( flexchain .or. segid LIGA )) end
mini abnr nstep 1000 tolgrd 0.001
cons fix select none end
```

```
!! Energy term
energy
set PLtotal = ?ener
coor tranlate xdir 400 ydir 400 zdir 400 select segid LIGA end
energy
set PLsep = ?ener
inte select segid PROT end
set protener = ?ener
inte select segid LIGA end
set ligaener = ?ener

!! Output
open unit 41 write form name "energy.dat"
write title unit 41
* @PLtotal @PLsep @protener @ligaener pose_@I
*
```