Check for updates

Chapter 7

# Virtual Ligand Screening Using PL-PatchSurfer2, a Molecular Surface-Based Protein–Ligand Docking Method

**Woong-Hee Shin and Daisuke Kihara**

## Abstract

Virtual screening is a computational technique for predicting a potent binding compound for a receptor protein from a ligand library. It has been a widely used in the drug discovery field to reduce the efforts of medicinal chemists to find hit compounds by experiments.

Here, we introduce our novel structure-based virtual screening program, PL-PatchSurfer, which uses molecular surface representation with the three-dimensional Zernike descriptors, which is an effective mathematical representation for identifying physicochemical complementarities between local surfaces of a target protein and a ligand. The advantage of the surface-patch description is its tolerance on a receptor and compound structure variation. PL-PatchSurfer2 achieves higher accuracy on apo form and computationally modeled receptor structures than conventional structure-based virtual screening programs. Thus, PL-PatchSurfer2 opens up an opportunity for targets that do not have their crystal structures. The program is provided as a stand-alone program at http://kiharalab.org/plps2. We also provide files for two ligand libraries, ChEMBL and ZINC Drug-like.

**Key words** Drug discovery, Molecular surface, Protein–ligand interaction, Three-dimensional Zernike descriptor, Virtual screening, 3DZD

## 1 Introduction

Virtual screening is a computational technique that searches active compounds for a target protein from a large virtual compound library [1]. It has been widely used to help the efforts of medicinal chemists to experimentally test and synthesize a large number of compounds by reducing the chemical space to explore. The technique is classified into two categories: ligand-based virtual screening (LBVS) and structure-based virtual screening (SBVS). LBVS compares the compounds in library with known drugs that have been previously discovered. Therefore, to use LBVS, prior knowledge of the known drugs is required. LBVS methods compare ligands in their 1D [2, 3], 2D [4, 5], or 3D structure representations [6, 7]. On the other hand, SBVS methods use the 3D

Mohini Gore and Umesh B. Jagtap (eds.), *Computational Drug Discovery and Design*, Methods in Molecular Biology, vol. 1762, https://doi.org/10.1007/978-1-4939-7756-7_7, © Springer Science+Business Media, LLC, part of Springer Nature 2018

structure of a target receptor protein and compute complementarities between the ligand binding pocket of the receptor and ligands in a library. The fit of a ligand to the binding pocket of the receptor is measured by estimating the binding energy in protein–ligand docking [8–10] or by evaluating geometrical matching of pharmacophores [11, 12]. One of the most widely used classes of SBVS methods is protein–ligand docking. In protein–ligand docking, interaction between a receptor and a ligand is evaluated by sampling binding poses of the ligand in the pocket and calculating the binding affinity of the poses. Generally the binding affinity is computed with a pairwise atom-based energy function.

Here, we explain how to use our novel SBVS method, PL-PatchSurfer2 [13]. Instead of employing an atomic-based interaction description, this program adopts molecular surface description. Four physicochemical features of molecules, both a receptor and ligands, are calculated and assigned on the surface: geometric shape, the electrostatic potential, hydrogen bonding ability (donors or acceptors), and the hydrophobicity. The complementarity between a pocket and a ligand is calculated by comparing chemical characters of local surface patches. A schematic illustration of PL-PatchSurfer2 is shown in Fig. 1.

The chemical features on each surface patch of a receptor and ligands are converted to three-dimensional Zernike descriptors (3DZD). 3DZD is a rotationally invariant representation of a 3D function in the Euclidean space (i.e., physicochemical properties mapped on the 3D molecular surface), which is essentially a vector

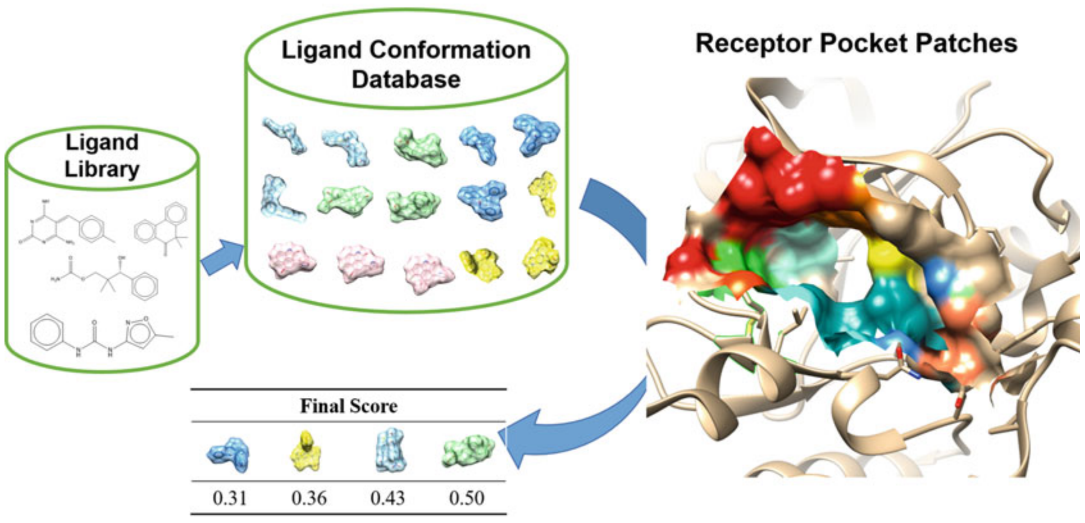


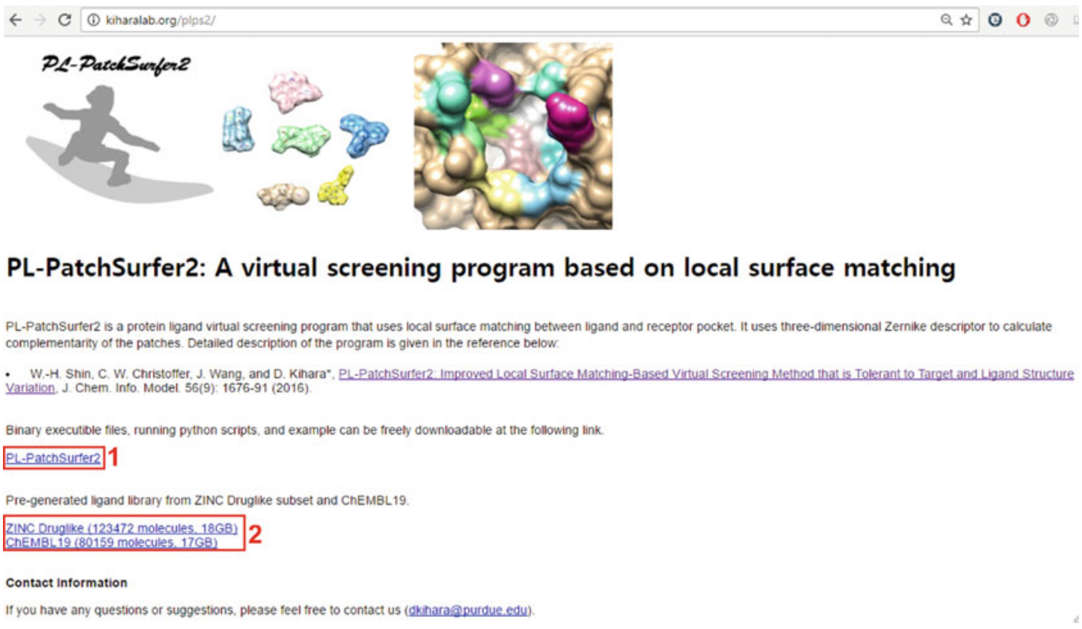
**Fig. 1** Illustration of PL-PatchSurfer2. Multiple three-dimensional conformations of ligands are generated by OMEGA. The surfaces of each ligand in a conformation and a receptor pocket are divided into patches. Local surface patches between the binding pocket of the protein and ligands are matched and the ligands are ranked in an ascending order of their scores

of coefficients from a series expansion of the 3D function into 3D Zernike basis function [14]. Our group has applied 3DZD to solve various structural biology problems, such as ligand similarity calculation [15], pocket–pocket comparison [16], electron microscopy density map comparison [17], and protein–protein docking [18]. An asset of PL-PatchSurfer2 is that it is tolerant to conformational changes of a receptor protein. Thus, the program showed better performance than conventional protein–ligand docking programs including AutoDock Vina [13], when the receptor structure is computationally modeled or in an apo-form, which can be substantially different from the ligand-bound form of the protein.

## 2 Materials

PL-PatchSurfer2 is available for academic users at our lab website, http://kiharalab.org/plps2/ (Fig. 2). The program and associated files are compressed in a file named *PLPS.tar.gz* and can be downloaded from a link shown as label 1 in Fig. 2. To decompress the file, in a GUI interface, right-click and select an option for decompress or double click to decompress the file. In Linux command line, type *tar –zxf PLPS.tar.gz*. All binary files are for the Linux OS.

Decompressing the file creates a directory named *PL-PatchSurfer2*. In the directory, there are four directories and *README* file. *apbs_tool* gives utilities to run APBS [19], which will be described later, *bin* and *scripts* contain executable files and python scripts to



PL-PatchSurfer2: A virtual screening program based on local surface matching

PL-PatchSurfer2 is a protein ligand virtual screening program that uses local surface matching between ligand and receptor pocket. It uses three-dimensional Zernike descriptor to calculate complementarity of the patches. Detailed description of the program is given in the reference below:

- W.-H. Shin, C. W. Christoffer, J. Wang, and D. Kihara*, PL-PatchSurfer2: Improved Local Surface Matching-Based Virtual Screening Method that is Tolerant to Target and Ligand Structure Variation, J. Chem. Info. Model. 56(9): 1676-91 (2016).

Binary executible files, running python scripts, and example can be freely downloadable at the following link.

PL-PatchSurfer2

Pre-generated ligand library from ZINC Druglike subset and ChEMBL19.

ZINC Druglike (123472 molecules, 18GB)
ChEMBL19 (80159 molecules, 17GB)

Contact Information

If you have any questions or suggestions, please feel free to contact us (dkihara@purdue.edu).

**Fig. 2** The PL-PatchSurfer2 webpage

run PL-PatchSurfer2, and *example* provides a step-by-step example of a virtual screening process. The details of the required program and input file will be described in next section.

On the PL-PatchSurfer2 webpage, pregenerated ligand library files are also made available (Label 2 of Fig. 2). *druglike.tar.gz* and *chembl.tar.gz* contain preprocessed files for ~120,000 and ~80,000 compounds, respectively. The libraries can be used for a virtual screening after decompressing file by typing *tar –zxf druglike.tar.gz* or *chembl.tar.gz*. Descriptions of how to use them will be given in Methods.

### 2.1 Input Files and Python Scripts of PL-PatchSurfer2

PL-PatchSurfer2 requires a receptor structure file and ligands files to be screened against. The input receptor structure file should be in the PDB format without any hetero atoms in the HETATM fields. To define a binding pocket of the protein, a cognate ligand that is cocrystallized ligand should also be given in PDB format. Ligand files need to be in the MOL2 format, which contains the atom information, coordinates, and charge information. The final output of the program is a text file that has a ranking of the compounds.

To execute a virtual screening experiment, the following Python scripts will be used, which locate in the *scripts* directory:

prepare_receptor.py: This script takes a protein PDB file and a cognate ligand PDB file as inputs and generates an SSIC file that contains patch information of the protein binding pocket. The format of SSIC file is shown in Subheading 3.

prepare_ligands.py: This script reads ligand MOL2 files, generates multiple conformations for each ligand using OMEGA [20], and produces SSIC files of the ligands. SSIC is a PL-PatchSurfer specific file format and contains information of surface patches of a molecule.

compare_seeds.py: As its name indicates, this script compares a ligand-binding pocket of a receptor and a ligand conformation by the Auction algorithm [21]. It takes SSIC files of the pocket and the ligand as input.

rank_ligands.py: PL-PatchSurfer2 offers two scoring options that evaluate the fit between a binding pocket and a ligand: The Lowest Conformation Score (LCS) and The Boltzmann-weighted Score (BS). LCS ranks ligands using the best scoring conformation of a ligand, whereas BS sorts ligands by a Boltzmann-weighted scoring scheme [13].

### 2.2 Required Programs to Run PL-PatchSurfer2

In order to run PL-PatchSurfer2, five external programs are required: PDB2PQR, APBS, OPEN BABEL, XLOGP3, and OMEGA. A brief explanation of each program is given below. These programs can be replaced alternative ones with the same function.

APBS and PDB2PQR: APBS [19] calculates molecular surface and the electrostatic potential on molecular surface by solving the Poisson-Boltzmann equation. PDB2PQR convert a PDB file to a PQR-format file, an input file of APBS by adding atom charge and radius information to the PDB file. The programs can be obtained from http://www.poissonboltzmann.org/.

XLOGP3: The aim of this program to estimate log*P* value of a molecule [22] and assigns a log*P* value to each atom. PL-PatchSurfer2 calculates a hydrophobic field [23] from atomic log*P* values and assigns the field on a molecular surface. XLOGP3 is downloadable at http://www.sioc-ccbg.ac.cn/skins/ccbgwebsite/software/xlogp3/.

OMEGA: To consider ligand flexibility, we use OMEGA [20] to generate multiple conformations of a ligand molecule from a single MOL2 file. For academic users, a 1-year license is offered, and the program can be obtained from http://eyesopen.com.

OPEN BABEL [24]: This program is for converts file formats of ligand files and used internally in the python scripts. It can be downloaded from http://openbabel.org/

## 3 Methods

PL-PatchSurfer2 first computes SSIC files of a ligand binding pocket of a receptor and ligands, which contain surface patch information. The input receptor structure and ligand structures need to be prepared in the PDB and in the MOL2 format, respectively. Once the SSIC files are computed, the patches are compared between the binding pocket and ligands and the ligands are ranked by the score that evaluates compatibility between the pocket and ligands. The overview of PL-PatchSurfer2 is illustrated in Fig. 3.

### 3.1 Receptor Binding Pocket File Preparation

To run PL-PatchSurfer2, an SSIC file of a ligand binding pocket of a receptor protein needs to be prepared, which contains information about the position, the distribution, and physicochemical properties of surface patches represented in 3DZD. To prepare the SSIC file, three files are required: a receptor structure PDB file, a cognate ligand to define a binding pocket also in the PDB format, and an input file, which lists file locations etc. A receptor PDB file and a cognate ligand file can be obtained by any structure viewer such as UCSF Chimera [25]. Otherwise, it can be obtained using *split_lig.py* in the *script* directory in command window as follows:

```
python split_lig.py [PDB file] [Chain ID] [Ligand ID]
```

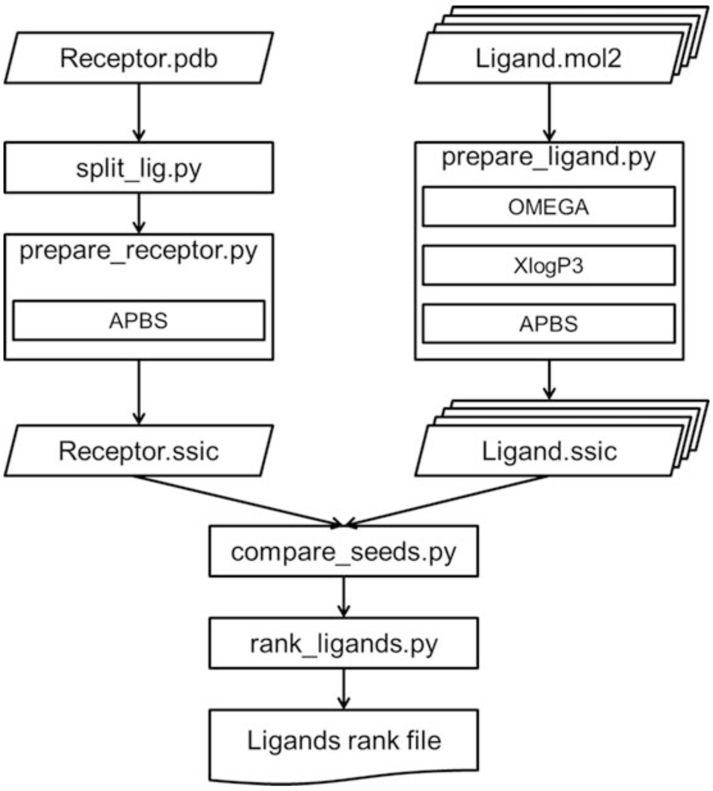


**Fig. 3** Flowchart of PL-PatchSurfer2. The Left side of the chart shows a process for generating an SSIC file for a receptor, while the right side shows the ligand library preparation steps. "split_lig.py" extracts a cocrystallized ligand from a receptor PDB file. "prepare_receptor.py" detects a binding pocket from the extracted ligand, runs APBS to calculate pocket surface and electrostatic potential, and generates a receptor SSIC file. On the ligand side, multiple conformations of a ligand are generated by OMEGA, atomic log*P* values are assigned by XlogP3, and molecular surface and the electrostatic potential are calculated by APBS. All these steps for ligand are executed in "prepare_ligand.py". A complementarity between the receptor and the ligands are calculated by "compare_-seeds.py." "rank_ligands.py" gives the final result, a rank of the ligands in the library

[PDB file] is a name of PDB file that has a receptor and a ligand structure. [Chain ID] should be matched to a chain name in PDB file, and [Ligand ID] should also be matched to a three-letter code of a ligand in the PDB file. The script identifies ligand coordinates in the PDB file and writes them in a file named *xtal-lig.pdb*. The coordinates of the receptor structure without the ligand is written in *rec.pdb*.

The current version of PL-PatchSurfer2 requires a cocrystallized ligand structure with the receptor protein to define a receptor binding pocket. If the receptor structure does not have a bound ligand, such as a computationally modeled structure or an apo structure, the user can provide a potential bound ligand from a homologous protein structure found in the PDB database (*see* **Note 1**) or provide a center position of the putative binding pocket identified by a ligand binding site prediction program, such as VisGrid [26] (*see* **Note 1**).

After splitting the receptor and the ligand into separate PDB files, the SSIC file of the receptor is generated by running *prepar-e_receptor.py* in the *scripts* directory:

```
python prepare_receptor.py [Input file]
```

An example of the input file is shown below (*rec_prep.in* in the *example* directory):

```
PLPS_path ~/project/PL-PatchSurfer2/
PDB2PQR_path ~/apps/pdb2pqr
APBS_path /apps/apbs/apbs-1.4/bin
BABEL_path /usr/bin
receptor_file rec.pdb
ligand_file xtal-lig.pdb
```

The top line, *PLPS_path*, shows the path of PL-PatchSurfer2 is installed. Similarly, following three lines designate the locations of programs. They can be changed to match the user's environment. The last two lines, *receptor_file* and *ligand_file* are file names of the receptor and its cognate ligand.

The output of this script is an SSIC format file named after the input structure file. For example, if the PDB file's name is *rec.pdb*, then the output is named as *rec.ssic*. This file contains information of patches: the coordinates of the patch center, 3DZDs of chemical features (shape, electrostatic potential, hydrogen bonding, and hydrophobicity), and the distribution of geodesic distance of patches. An example of SSIC file is given below:

```
68 72 144 144 144
5.632 10.269 10.834
 0  72  0.16617  0.00000  0.26899  0.27014  0.01290  0.01691  0.20478  0.21539  0.21553
 0.04274 ...
 3  144  0.04947  0.00000  0.07677  0.07714  0.00725  0.00944  0.05392  0.05716  0.05732
 0.02009 ...
 5  144  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000  0.00000
 0.00000 ...
 6  144  0.00337  0.00000  0.00726  0.00726  0.00004  0.00006  0.01014  0.01017  0.01017
 0.00022 ...
68  0  12  10  8  13  5  5  13  8  14  17  17  21  20  15  22  21  25  23  24  20  24  24  28  26  21  30  28
27  28  32  30  27  30  37  31  35  29  28  31  27  26  32  24  21  20  22  22  17  24  27  23  22  19  25  18
17  24  27  16  13  16  12  11  13  17  18  21
```

The first line indicates that the pocket is composed of 68 patches and chemical features, shape, electrostatic potential, hydrogen bonding, and hydrophobicity, are converted to 72-, 144-, 144-, and 144-dimensional 3DZD vectors, respectively. The second line is a (*x*, *y*, *z*) coordinate of the center of the patch. The following four lines starting with *0 72*, *3 144*, *5 144*, and *6 144* show 3DZD vectors of the geometric shape, the electrostatic potential, the hydrogen bonding, and the hydrophobicity. The last line, *11 13 17 18 21*, is a histogram of geodesic distance between the patch the patch center to the other patch centers of the pocket with a bin size of 1.0 Å. *convert_ssic_to_pdb.py* helps visualizing location of patches by generating PDB files that contains the coordinates of patch centers (*see* **Note 2**).

### 3.2 Ligand File Preparation

Similar to the receptor, input files for ligands should be prepared in the SSIC format from their MOL2 format files. A MOL2 file can be converted from a SMILES string, a one-dimensional representation of a molecule, using OpenBabel [24]. Alternatively, it can be also obtained from a public accessible library such as ZINC [27], ChEMBL [28], and PubChem [29].

Ligands files are prepared using *prepare_ligands.py* in the *scripts* directory, in the similar way as the receptor preparation:

```
python prepare_ligands.py [Input file]
```

The input file should have contents shown below (*lig_prep.in* in *example* directory):

```
PLPS_path ~/project/PL-PatchSurfer2/
PDB2PQR_path ~/apps/pdb2pqr
APBS_path /apps/apbs/apbs-1.4/bin
BABEL_path /usr/bin
XLOGP3_path ~/apps/XLOGP3/bin/
OMEGA_path ~/apps/openeye/arch/Ubuntu-12.04-x64/omega
n_conf 50
ligand_file ZINC03833861.mol2
ligand_file ZINC03815630.mol2
```

Up to the fourth line from the top of the file are the locations of the programs: PL-PatchSurfer2, PDB2PQR, APBS, and OPEN BABEL. The fifth and the sixth lines show the locations of programs, XLOGP3 and OMEGA. *n_conf* is a parameter for OMEGA. The program generates multiple conformations of a ligand to reflect its flexibility and *n_conf* determines the maximum number of conformations to be produced. Ligand MOL2 files are listed after that, with *ligand_file* as a header of a line. There is no limit for the number of ligands to process.

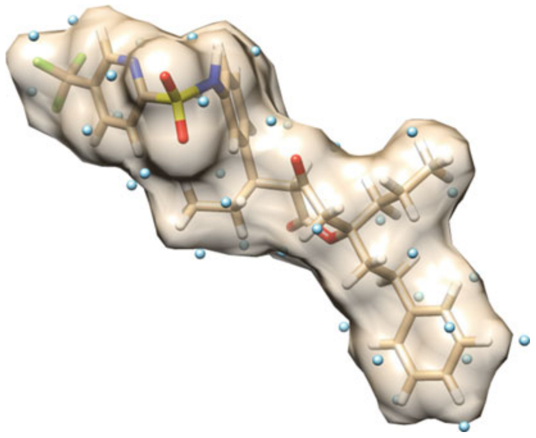


**Fig. 4** Seed points of local surface patches distributed on the surface of a compound, ZINC03815630. There are 35 patches and blue dots show the centers of the patches (seed points)

Running the scripts will generate a directory, for each ligand in the ligand list in the *lig_prep.in* file, each of which contains conformation files in the PDB format and patch information files in the SSIC format. Thus, the number of directories generated equals the number of ligands in the library. Patches of a ligand conformation are generated and distributed along the molecular surface as Fig. 4. An example of a ligand SSIC file is given below. This is the same format as a receptor SSIC file:

```
35 72 144 144 144
8.908 0.088 1.140
0 72 0.14882 0.00000 0.25880 0.25937 0.00286 0.00380 0.22792 0.23418 0.23420
0.01081...
3 144 0.03049 0.00000 0.05685 0.05688 0.00016 0.00043 0.05894 0.05928 0.05928
0.00070...
5 144 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000...
6 144 0.05316 0.00000 0.09245 0.09265 0.00102 0.00136 0.08142 0.08365 0.08366
0.00386...
35 0 4 4 20 7 5 20 28 25 23 23 29 8 20 24 28 28 16 13 12 18 12 14 19 8 12 18 14
28 31 27 16 29 5 17
```

## 3.3 Comparing Patches

After generating SSIC files of a receptor and ligands, complementarities between are measured to find active ligands. In PL-PatchSurfer, the auction algorithm [21] is employed to match surface patches between the protein and each ligand. To compare and identify compatible surface patches between a receptor and ligands, *compare_seeds.py* in *scripts* is executed as follows.

```
python compare_seeds.py [Input file]
PLPS_path ~/project/PL-PatchSurfer2/
receptor_file rec.ssic
n_conf 50
ligand_dir ZINC03833861
ligand_dir ZINC03815630
```

The structure of an input file is shown above. The first line shows the location of PL-PatchSurfer2 is installed, and *receptor_file* is a protein SSIC file prepared in the receptor preparation step. *n_conf* is the number of maximum conformations for each ligand and *ligand_dir* is a directory for ligand conformations and SSIC files generated in the previous step.

Two types output files will be generated by executing the python script. The first set of output files are named as a combination of a receptor name, ligand name, and the conformation number. For example, for a case that a receptor's name is ERa and a ligand's name is estrogen, output file names will be ERa_estrogen_-conf_01.dat for the first conformation of the ligand. If the user runs this script with the same parameter as the provided example, the output file name will be rec_ZINC03815630_conf_01.dat and so on. They are saved in the directory of ligand conformations. An example of the output file is illustrated below.

```
54 19 0.373 0.277 0.227 0.287 0.197 0.127 0.297 0.000 14.097
59 16 0.328 0.349 0.160 0.249 0.294 0.095 0.169 0.000 12.716
60 26 0.412 0.240 0.223 0.273 0.108 0.102 0.190 0.360 14.616
61 7 0.324 0.223 0.178 0.227 0.153 0.111 0.249 0.000 17.601
62 8 0.398 0.235 0.286 0.328 0.189 0.074 0.156 0.000 12.513
65 22 0.382 0.307 0.186 0.259 0.194 0.232 0.199 0.000 8.476
SUM: 8.807 AVG: 0.275 avgRP: 2.915 navgRP: 1.872 AVGSd 0.226 0.142 0.186 0.214
```

The first two columns except for the last line are patch indices of the receptor protein and the ligand in a certain conformation that are paired by the auction algorithm. The next four columns show a total score of matched pairs, and three individual terms of the score (weighted sum of 3DZD difference, geodesic distance distribution difference, and geodesic distance difference between matched pairs). Values in the next four columns show the 3DZD differences (dissimilarity) of the two patches in terms of shape, the electrostatic potential, hydrophobicity, and hydrogen bond term, from left to right, respectively. The last column is the Euclidean distance of the patch centers. The last line summarizes the score of the two patches by averaging three scoring terms used in PL-PatchSurfer2: 3DZD difference, geodesic distance difference, and approximate position difference calculated by the geodesic distance histogram [13].

Another output is a summary file provided for each ligand screened. It is named as a combination of the receptor file name and the ligand name, for example, *rec_ZINC03815630.dat*. The file lists the conformation of the ligand and the four scoring terms.

```
 1 0.37611 0.18108 0.13645 0.45588
 2 0.35956 0.18906 0.12975 0.50000
 3 0.38208 0.18667 0.15510 0.47059
 4 0.39500 0.19485 0.16095 0.42647
 5 0.39779 0.19794 0.13975 0.50000
 6 0.40681 0.19178 0.15785 0.47059
 7 0.40150 0.19709 0.14375 0.50000
 8 0.38156 0.19872 0.13715 0.47059
 9 0.41342 0.18456 0.12550 0.47059
10 0.39468 0.19456 0.13200 0.50000
```

The first column is the conformation index. Following columns are 3DZD difference, APPD (approximate position difference), GRPD (geodesic distance difference), and size difference between the pocket and a ligand in that conformation, respectively. The overall score of a conformation is calculated as a weighted sum of these four values.

## 3.4 Ranking Ligands

The last step of PL-PatchSurfer2 is ranking ligands for the target binding pocket in the receptor. The top ranked ligands are predicted to bind to the target binding pocket. The program provides two options for scoring ligands: the lowest conformer score, which ranks ligands based on the lowest scored conformer among conformations examined, and the Boltzmann weighted scoring, which averages the scores given to different conformations of each ligand by putting exponential weight to each conformation.

$$\text{Boltzmann-Weighted Score }(P, L) = \frac{\sum_{C}^{N_{\text{conf}}} \text{CS}(P, C) \times \exp[-\beta \times \text{CS}(P, C)]}{\sum_{C}^{N_{\text{conf}}} \exp[-\beta \times \text{CS}(P, C)]}$$

where $P$, $L$, and $C$ stand for pocket, ligand, and ligand in a certain conformation, respectively. CS($P$,$C$) is a *Conformer Score*, the score between the pocket $P$ and the ligand in the conformation $C$. $N_{\text{conf}}$ is the number of ligand conformations. $\beta$ determines a weight to be given to each conformer, which is set to 1.

To execute ranking of ligands, run *rank_ligands.py* in the *scripts* directory as follows:

```
python rank_ligands.py [Input file]
```

An example of an input file is illustrated below (*lcs.in* provided in the *example* directory):

```
receptor_file rec.ssic
scoring_type lcs
ligand_dir ZINC03833861
ligand_dir ZINC03815630
output_file lcs.rank
```

*receptor_file*, *ligand_dir*, and *output_file* are the names of the receptor SSIC file, the ligand conformation directories, and the output file that has a ranked list of screened ligands. *scoring_type* designates the type of the scoring function, either *lcs* or *bs*, for the lowest conformer score or the Boltzmann-weighted score, selected by the user.

The output file of *bs* scoring type has two columns. The first column shows the name of the ligands, while the second column shows the score of ligands. The ligands are sorted by the score in the ascending order.

```
ZINC03815630 0.69811
ZINC03833861 0.70995
```

The output of *lcs* scoring function has three columns. Between the columns of ligand names and the score is situated an additional column which shows the index of the conformations of the ligand that gave the lowest score.

```
ZINC03815630 1 0.66714
ZINC03833861  31 0.67820
```

### 3.5 Virtual Screening Using Pregenerated Ligand Sets

On the web site of PL-PatchSurfer2 (http://kiharalab.org/plps2/), we provide two pregenerated ligand sets: Drug-like (*druglike.tar.gz*) and ChEMBL19 (*chembl.tar.gz*). Both sets are preselected sets provided in the ZINC library (http://zinc.docking.org). The Drug-like set is composed of ligands that satisfy "Lipinski's Rule of Five" [30], which are four chemical properties of compounds that are suitable for drugs. The ChEMBL19 dataset was selected from ChEMBL [28], which is an open compound library with bioactivity information collected from medicinal chemistry literature. The pre-selection of the two datasets was performed using the SUBSET 1.0 algorithm [31]. The Drug-like dataset were filtered with 90% Tanimoto similarity cutoff and the ChEMBL19 dataset were filtered with 80% Tanimoto similarity cutoff. The Drug-like and ChEMBL19 set have 123472 and 80159 compounds, respectively.

To use the database, decompress by a command *tar –zxf [tar.gz file]*. In the directory, *gen_input.py* makes input files for comparing patches and ranking ligands. Four parameters should be given to run the python script.

```
python gen_input.py [Receptor SSIC] [PL-PatchSurfer2 path]
[Scoring function] [Rank file]
```

*Receptor SSIC* is the name of the pocket SSIC file. *PL-PatchSurfer2 path* is a location of the program. The user may choose type of scoring function by typing *lcs* for lowest conformer or *bs* for Boltzmann-weighted in [*Scoring function*]. [*Rank file*] is an output file name that will contain the ranking of the ligands in the library. It can be any file name the user wishes to have. The outputs of this python script are *compare_seeds.in* and *rank_ligands.in*, which are input files to run *compare_seeds.py* and *rank_ligands.py*, respectively.

## 3.6 Case Study

In this section, we will show an example of virtual screening using PL-PatchSurfer2. The target protein is a SRC kinase (PDB ID: 2SRC). SRC kinase phosphorylates tyrosine of other proteins [32]. The activation of the SRC pathway is related to colon, liver, lung, breast, and pancreatic cancer [33]. The three-dimensional structure bound with phosphoaminophosphonic acid-adenylate ester, an inhibitor of ATP-dependent phosphorylation, is shown in Fig. 5. The active 60 compounds of this target were mixed with 1740 nonactive compounds in the library. The ratio between actives and decoys are 1:29. Nonactive compounds were randomly chosen from the DUD dataset [34]. In this example, the ligands were scored and ranked by the lowest conformer scoring scheme.

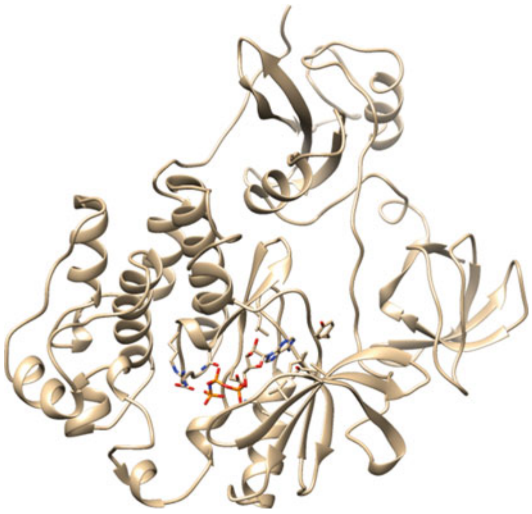


**Fig. 5** The crystal structure of human SRC kinase bound with phosphoaminophosphonic acid-adenylate ester, an analog of ATP (PDB ID: 2SRC)

Table 1
Top 18 ligands ranked by the lowest conformer score

| Rank | Class | ZINC ID | Score |
|---|---|---|---|
| 1 | Active | ZINC03815379 | 0.49045 |
| 2 | Active | ZINC03815551 | 0.51416 |
| 3 | Active | ZINC03815482 | 0.51647 |
| 4 | Active | ZINC03815493 | 0.53378 |
| 5 | Active | ZINC03815508 | 0.53546 |
| 6 | Active | ZINC03815483 | 0.53770 |
| 7 | Active | ZINC03815489 | 0.54398 |
| 8 | Active | ZINC03815307 | 0.54499 |
| 9 | Active | ZINC03815505 | 0.56146 |
| 10 | Active | ZINC03815507 | 0.56395 |
| 11 | Active | ZINC03815525 | 0.56614 |
| 12 | Active | ZINC03815490 | 0.56656 |
| 13 | Decoy | ZINC02196955 | 0.56680 |
| 14 | Active | ZINC03815503 | 0.56779 |
| 15 | Active | ZINC03832351 | 0.57083 |
| 16 | Decoy | ZINC03631303 | 0.57389 |
| 17 | Decoy | ZINC00766404 | 0.57602 |
| 18 | Active | ZINC03815545 | 0.57719 |

Table 1 shows top 18 (1%) ranked ligands. Two-dimensional structures of the molecules are shown in Fig. 6. Among top 18 molecules, 15 were active compounds. The enrichment factor at 1% (top 18 molecules) is 25 means that PL-PatchSurfer2 finds active compounds 25 times more than random selection at top 1%.

## 4 Notes

1. PL-PatchSurfer2 requires a bound ligand in the receptor protein to define a binding pocket. However, computationally modeled structures or receptors in their apo (ligand-free) form do not have one. For a computational protein model, if the model is built by homology modeling based on a template protein that has a bound cognate ligand, superimpose the modeled structure on to the template structure and use the

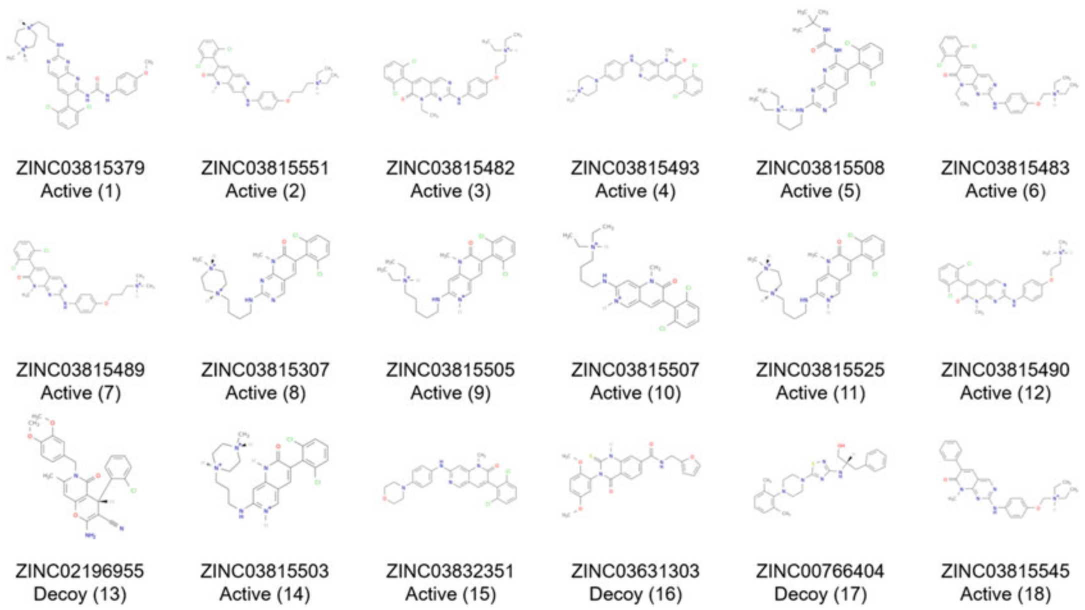


**Fig. 6** Two-dimensional structures of top 1% ligands identified by PL-PatchSurfer2 for SRC kinase. The number indicates the rank of the ligand assigned by PL-PatchSurfer2. ZINC ID is shown for each ligand. The numbers in parentheses are the rank of ligands

cognate ligand of the template structure as the input parameter used in Subheading 3.1.

If there are no homologous structures that have bound ligand, or if apo-form of a binding pocket is used, provide the binding pocket center position in *pkt_cntr.pdb* in the *scripts* directory. Then, replace the line *ligand_file* in *rec_prep.in* from *xtal-lig.pdb* to *pkt_cntr.pdb*.

2. To visualize the patch location on a binding pocket or on a ligand molecule, PL-PatchSurfer2 provides *convert_ssic_to_pdb.py* in the *scripts* directory, a python script that extracts seed point (center of patch) coordinates from the SSIC file. Executing the script gives a PDB format file of seed point coordinates:

```
python convert_ssic_to_pdb.py [SSIC file] [Output PDB file]
```

*Output PDB file* can be any file name the user wish to use for an output file. Use any molecular structure viewer, such as PyMol, and load the output file and a three-dimensional structure file of a molecule (a protein or a ligand), from which SSIC file was generated, to visualize the seed position of patches distributed on the molecular surface.

## Acknowledgment

We acknowledge Dan K. Ntala for proofreading the manuscript. This work was partly supported by grants from the National Science Foundation (IIS1319551). D.K. also acknowledges supports from National Institutes of Health (R01GM097528, R01GM123055) and the National Science Foundation (IOS1127027, DMS1614777).

### References

1. Walters WP, Stahl MT, Murcko MA (1998) Virtual screening—an overview. Drug Discov Today 3(4):160–178
2. Schwartz J, Awale M, Reymond JL (2013) SMIfp (SMILES fingerprint) chemical space for virtual screening and visualization of large databases of organic molecules. J Chem Info Model 53(8):1979–1989
3. Durant JL, Leland BA, Henry DR, Nourse JG (2002) Reoptimization of MDL keys for use in drug discovery. J Chem Inf Comput Sci 42 (6):1273–1280
4. Raymond JW, Gardiner EJ, Willett P (2002) RASCAL: calculation of graph similarity using maximum common edge subgraphs. Comput J 45(6):631–644
5. Bender A, Mussa HY, Glen RC (2004) Similarity searching of chemical databases using atom environment descriptors (MOLPRINT 2D): evaluation of performance. J Chem Inf Comput Sci 44(5):1708–1718
6. Ballester PJ, Richards WG (2007) Ultrafast shape recognition to search compound databases for similar molecular shapes. J Comput Chem 28(10):1711–1723
7. Hawkins PCD, Skillman AG, Nicholls A (2007) Comparison of shape-matching and docking as virtual screening tools. J Med Chem 50(1):74–82
8. Jain AN (2007) Surflex-dock 2.1: robust performance from ligand energetic modeling, ring flexibility, and knowledge-based search. J Comput-Aided Mol Des 21(5):281–306
9. Trott O, Olson AJ (2010) AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. J Comput Chem 31 (2):455–461
10. Allen WJ, Balius TE, Mukherjee S, Brozell SR, Moustakus DT, Lang PT, Case DA, Kuntz ID, Rizzo RC (2015) DOCK 6: impact of new features and current docking performance. J Comput Chem 36(15):1132–1156
11. Leach AR, Gillet VJ, Lewis RA, Taylor R (2010) Three-dimensional pharmacophore methods in drug discovery. J Med Chem 53 (2):539–558
12. Wolber G, Langer T (2005) LigandScout: 3-D pharmacophores derived from protein-bound ligands and their use as virtual screening filters. J Chem Info Model 45(1):160–169
13. Shin WH, Christoffer CW, Wang J, Kihara D (2016) PL-PatchSurfer2: improved local surface matching-based virtual screening method that is tolerant to target and ligand structure variation. J Chem Info Model 56 (9):1676–1691
14. Novotni M, Klein R (2003) 3D Zernike descriptors for content based shape retrieval. In: Proceedings of eighth ACM symposium on solid modeling and applications, Washington, pp 216–225
15. Shin WH, Zhu X, Bures MG, Kihara D (2015) Three-dimensional compound comparison methods and their application in drug discovery. Molecules 20(7):12841–12962
16. Zhu X, Xiong Y, Kihara D (2015) Large-scale binding ligand prediction by improved patch-based method Patch-Surfer2.0. Bioinformatics 31(5):707–713
17. Esquivel-Rodriguez J, Xiong Y, Han X, Gang S, Christoffer CW, Kihara D (2015) Navigating 3D electron microscopy maps with EM-SURFER. BMC Bioinf 16:181
18. Venkatraman V, Yang YD, Sael L, Kihara D (2009) Protein-protein docking using region-based 3D Zernike descriptors. BMC Bioinf 10:407
19. Baker NA, Sept D, Joseph S, Holst MJ, McCammon JA (2001) Electrostatics of nanosystems: application to microtubules and the ribosome. Proc Natl Acad Sci U S A 98 (18):10037–10041
20. Hawkins PCD, Skillman AG, Warren GL, Ellingson BA, Stahl MT (2010) Conformer generation with OMEGA: algorithm and validation using high quality structures from the

protein databank and Cambridge Structural Database. J Chem Info Model 50(4):572–584

21. Sael L, Kihara D (2012) Detecting local ligand-binding site similarity in nonhomologous proteins by surface patch comparison. Proteins 80 (4):1177–1185
22. Cheng T, Zhao Y, Li X, Lin F, Xu Y, Zhang X, Li Y, Wang R (2007) Computation of octanol−water partition coefficients by guiding an additive model with knowledge. J Chem Info Model 47(6):2140–2148
23. Heiden W, Moeckel G, Brickmann J (1993) A new approach to analysis and display of local lipophilicity/hydrophilicity mapped on molecular surfaces. J Comput-Aided Mol Des 7 (5):503–514
24. O'Boyle NM, Banck M, James CA, Morley C, Vandermeersh T, Hutchison GR (2011) Open Babel: an open chemical toolbox. J Cheminf 3:33
25. Pettersen EF, Goddard TD, Huang CC, Couch GS, Greenblatt DM, Meng EC, Ferrin TE (2004) UCSF chimera—a visualization system for exploratory research and analysis. J Comput Chem 25(13):1605–1612
26. Li B, Turuvekere S, Agrawal M, La D, Ramani K, Kihara D (2008) Characterization of local geometry of protein surfaces with the visibility criterion. Proteins 71(2):670–683
27. Irwin JJ, Sterling T, Mysinger MM, Bolstad ES, Coleman E (2012) ZINC: a free tool to discover chemistry for biology. J Chem Inf Model 52(7):1757–1768
28. Bento AP, Gaulton A, Hersey A, Bellis LJ, Chambers J, Davies M, Krüger FA, Light Y, Mak L, McGlinchey S, Nowotka M, Papadatos G, Santos R, Overington JP (2014) The ChEMBL bioactivity database: an update. Nucleic Acids Res 42(D1):D1083–D1090
29. Kim S, Thiessen PA, Bolton EE, Chen J, Fu G, Gindulyte A, Han L, He J, He S, Shoemaker BA, Wang J, Yu B, Zhang J, Bryant SH (2016) PubChem substance and compound databases. Nucleic Acids Res 44(D1):D1202–D1213
30. Lipinski CA (2000) Drug-like properties and the causes of poor solubility and poor permeability. J Pharmacol Toxicol Methods 44 (1):235–249
31. Volgt JH, Blenfalt B, Wang S, Nicklaus MC (2001) Comparison of the NCI open database with seven large chemical structural databases. J Chem Inf Comput Sci 41(3):702–712
32. Wheeler DL, Iida M, Dunn EF (2009) The role of Src in solid tumors. Oncologist 14 (7):667–678
33. Dehm SM, Bonham K (2004) SRC gene expression in human cancer: the role of transcriptional activation. Biochem Cell Biol 82 (2):263–274
34. Huang N, Shoichet BK, Irwin JJ (2006) Benchmarking sets for molecular docking. J Med Chem 49(23):6789–6801