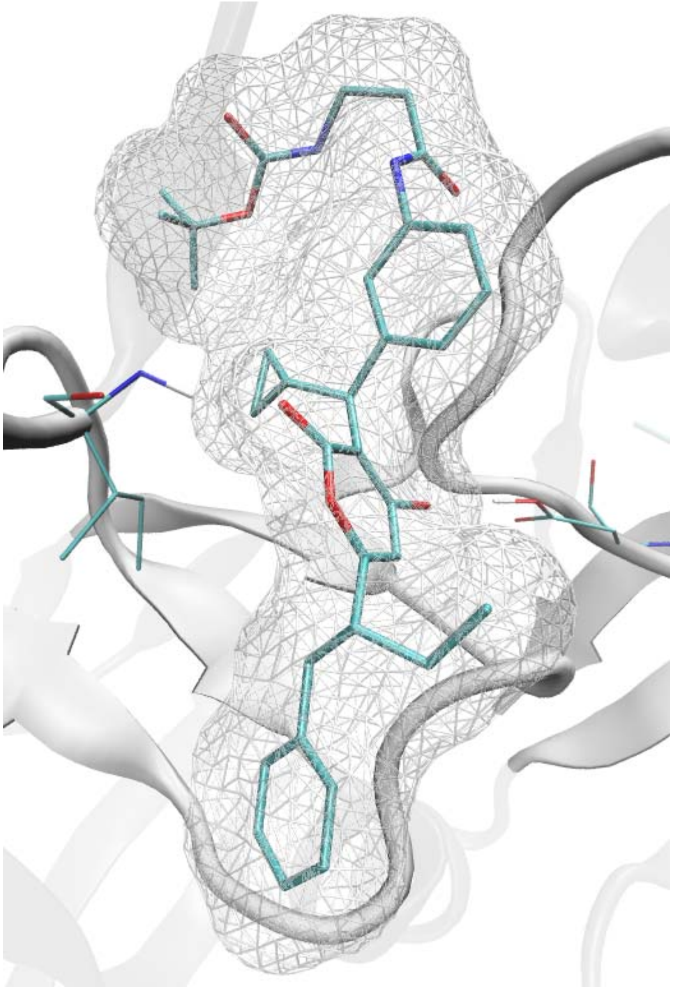
Lead-Finder v.1.1.10

Software for Drug Discovery



Modeling the HIV-1 protease complex with U100313 is a particularly tough case for docking software. Lead-Finder successfully predicts correct ligand position (rendered in licorice) with 1.2Å RMSD as evidenced from X-ray structure (electron density rendered in wireframe).

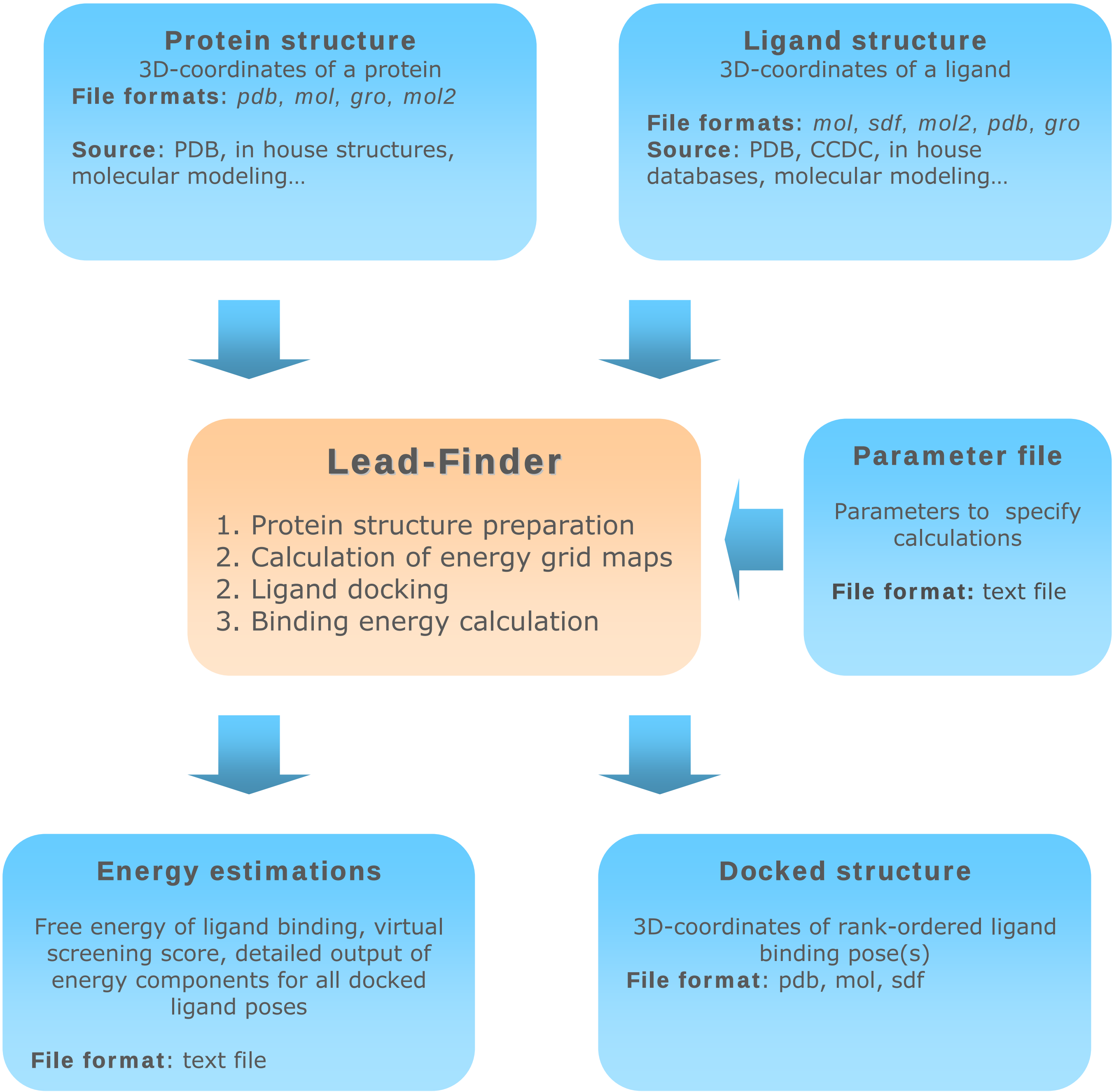
User Manual

Table of contents

Data flow in Lead-Finder

Lead-Finder software is a command line application that is available for Windows and Linux platforms. Lead-Finder takes 3D-coordinates of protein and ligand as input, along with a short text file containing calculation parameters and global settings. Lead-Finder can be used to perform:

- protein structure preparation;
- high-precision protein-ligand docking;
- estimation of the free energy of protein-ligand binding;
- virtual ligand screening to search for potent binders for a given protein target.



Protein structure
3D-coordinates of a protein
File formats: *pdb, mol, gro, mol2*

Source: PDB, in house structures, molecular modeling…

Ligand structure
3D-coordinates of a ligand

File formats: *mol, sdf, mol2, pdb, gro*
Source: PDB, CCDC, in house databases, molecular modeling…

Lead-Finder

1. Protein structure preparation
2. Calculation of energy grid maps
2. Ligand docking
3. Binding energy calculation

Parameter file

Parameters to specify calculations

File format: text file

Energy estimations

Free energy of ligand binding, virtual screening score, detailed output of energy components for all docked ligand poses

File format: text file

Docked structure

3D-coordinates of rank-ordered ligand binding pose(s)
File format: pdb, mol, sdf

Protein Structure And Its Preparation For Docking

- Common file formats[1] such as *pdb*, *gro*, *mol2*, containing 3D-coordinates of heavy atoms and at least functional hydrogen atoms (such as ones attached to N, O, S) are accepted as input for Lead-Finder.

- 3D-structure of a protein can be taken from publicly available or in-house databases, or obtained through molecular modeling techniques. Accuracy of 3D protein structure is crucially important for molecular docking applications. As a rule, high-resolution (<2.5 Å) X-ray crystal structures perform better when alternatives exist.

- Frequently, a protein structure, especially if comes from PDB, will contain no hydrogen atoms. Lead-Finder requires appropriate placement of hydrogen atoms in the protein structure. At least the functional hydrogens must be placed and the following points must be considered: (i) protonation state of a protein depends on pH; (ii) protonation of His should be reviewed as a special case; (iii) protons should be placed on the fittest atom when alternatives exist, such as for chemically equivalent atoms in His, Glu, Asp; (iv) proton orientation may have to be further optimized.

- Lead-Finder can automatically prepare protein structure file by adding hydrogen atoms, selecting ionization states of amino acids, optimizing positions of hydrogens, etc. For this purpose special program called **Model_build** is included in Lead-Finder distribution. Model_build accounts for electrostatic, Van der Waals and hydrogen bonding energy when adding hydrogen atoms to a protein and optimizing their positions, thereby performing high quality automatic structure preparation. Description of electrostatic calculations and optimization algorithms implemented in Model_build can be found in Technology and Bencmarking sections of Lead-Finder internet site.

- Special care should be taken with respect to the defects in experimentally resolved protein structures, such as missed atoms or residues, incorrect bond lengths, angles etc., especially in the proximity to the ligand's binding site[2]. Model_build automatically repairs some of the widespread defects like incorrect aminoacid labels, missing or unresolved side chains of aminoacids.

- Protein cofactors important for ligand binding should be retained within the protein structure. Model_build automatically adds hydrogen atoms to cofactors.

- Any non-intrinsic parts of a protein such as ligand, water molecules, buffer ions, etc. should be removed before docking calculations. Cofactors and (structurally or catalytically important) metal ions bound to protein should be retained; sometimes, conservative structural water molecule(s) known to play crucial role in ligand binding may be retained as well.

[1] Format converting programs such as Obabel, freely available at Hhttp://openbabel.sourceforge.net H, may be used to convert your protein structure into one of the file formats listed above.

[2] The quality of a protein structure may be assessed by a number of internet services available at the PDB site Hwww.rcsb.orgH.

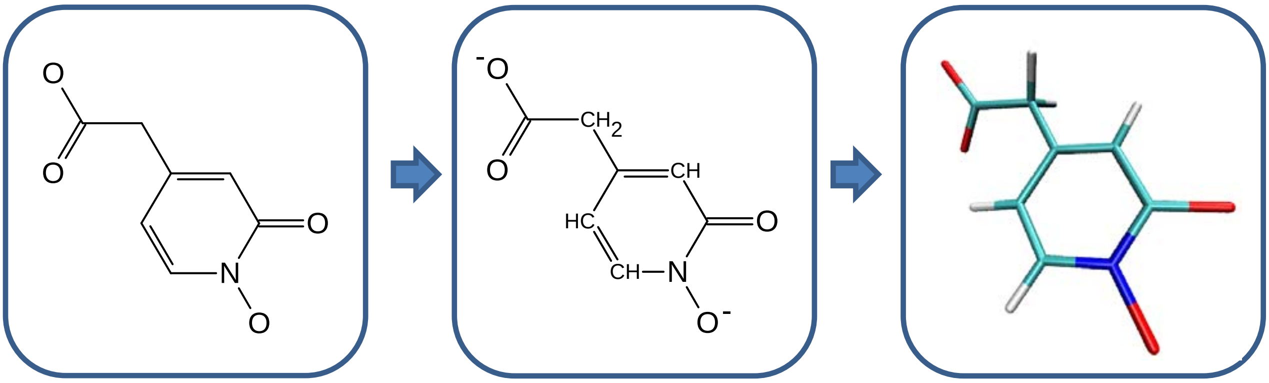
Ligand Structure And Its Preparation For Docking

- Common file formats such as *pdb*, *sdf*, *mol*, *mol2*, *gro* containing 3D-coordinates of heavy atoms and all hydrogen atoms are accepted as input for Lead-Finder.

- 3D-coordinates of a ligand can be taken from publicly available or in-house databases, or obtained through molecular modeling techniques. 2D-structures of ligands must be converted to 3D-coordinates before docking[3].

- Ligand structure must have all hydrogen atoms in correct places. Correct protonation[4] must be performed before docking with Lead-Finder.

- A single *sdf* file containing coordinates for multiple compounds may be used with Lead-Finder. In this case, an additional brief file listing ligand names and corresponding binding energies (dG and VS-score) will be produced for user convenience.



Crude 2D-structure of a ligand

2D-structure with all H-atoms and ionization states adjusted to pH 7. Note that the hydroxyl group bound to the nitrogen atom is ionized!

Complete 3D-structure

[3] ACD/ChemSketch software (Hwww.acdlabs.comH) or Corina Hhttp://www.molecular-networks.com/software/corina/index.htmlH) can be used for 3D-optimization of 2D-structures.

[4] ACD/pKa software (Hwww.acdlabs.comH) can be used for predicting pKa and protonation of ionogenic groups .

Installation

Once the Lead-Finder software license is obtained you get the installation distributive and the HASP key (USB device) that manages your license. HASP device stores information about your license, so when it comes off the license HASP device blocks the program exploitation. Updating license is performed via a simple patch file described below.
The process of software installation under Windows and Linux is provided below.

Installation under Windows

Install the HASP drivers by executing either *haspdinst.exe* or *HASPUserSetup.exe* files located in 'Drivers' directory of your installation distributive.[5] Both *haspdinst.exe* and *HASPUserSetup.exe* recognize your operating system and install the correct driver to the required location. The difference between *haspdinst.exe* and *HASPUserSetup.exe* is that the former is a command line application, while the latter has a graphical user interface.

After you have installed the HASP driver, copy files *lead_finder.exe*, *dock_ff.lib and residue.lib* from the 'Program' directory of your installation distributive to the directory where you'd like to install Lead-Finder. Edit your %PATH% environment variable by adding your installation path to it. Now software is ready for use. Do not forget to keep HASP key inserted when running Lead-Finder.

Installation under Linux

Install the HASP driver (*aksusbd* daemon) either from *rpm* package or by running installation script[5].

RPM packages corresponding to various Linux distributions are provided in the 'Drivers' directory of your installation distributive.[6] You can find there packages specially designed for SuSe and RedHat distributions, however, to our experience, they will work for other RedHat- and Debian-compatible Linux distributions. Also the latest versions of HASP drivers for various Linux distributions are freely available at: http://www.hasp.com/downloads.

Alternatively, you may launch *dinst* script provided in the directory *setup/aksusbd*. This script designed by HASP suppliers will automatically install the optimal driver for your operating system. This script also sets up the *aksusb* daemon to start on boot.

After you have installed HASP driver, you need to have HASP device mounted properly. On newer Linux distributions (SuSe 8.0 and higher, RedHat 9 and higher) it is mounted automatically upon HASP driver installation. You can ensure if the device was mounted by typing:

```
bash –c “if [ -e /proc/bus/usb ]; then echo 1; else echo 0; fi;”
```

and examining the output.
If the HASP device was not mounted automatically, type the following string to mount it:

```
mount -t usbdevfs none /proc/bus/usb
```
[5]

To install **Lead-Finder** software copy files *lead_finder*, *dock_ff.lib and residue.lib* from the *Program* directory of your installation distributive to the directory */usr/local/bin. Please make sure that ownership and access permissions are properly set up.*

Importantly that under Linux you must have *aksusbd* daemon launched before running Lead-Finder software. A*ksusbd* daemon is launched with the following command:
<path>/aksusbd
Contact your system administrator for technical questions.
Do not forget to keep HASP key inserted when running Lead-Finder.
Please note that you will need to install HASP LM in order to communicate with your network HASP key. Please refer to HASP HL manual for the details.

[5] Administrator/root privileges are required for installation.

[6] Additional information on particular drivers for different Linux distributions can be found in *readme.txt* file in the 'Drivers' directory.

Updating software license

To update your license insert HASP key and run program *update_key*. The program will prompt you to retrieve HASP key information (*i*) or to update the key (*u*).

Choose option (*i*), this will generate ‘client-to-vendor’ file, which contains information about the current status of your license.

Send this file to us by e-mail, and you’ll receive file ‘vendor-to-client’ which contains updated license information.

Run program *update_key*, choose option *u* this time and select your updated file. Now your license is renewed.

Configuration Options for Docking

You can set up Lead-Finder tasks by specifying parameters in a ***par*** file. ***par*** file is a human-readable text file containing references to ligand and protein structure, instructions to energy grid map calculations and optional preparation of protein structure. The order of parameters in the parameter file is arbitrary. Single-line comments are allowed after the semi-colon ";" symbol.

Ligand file

ligand = "filename"
Optional parameter; no default value
Example: ligand = "ligand.mol"

Specifies name of a file in *mol, sdf, mol2, pdb or gro* format containing ligand 3D-coordinates. When Lead-Finder is run with *–li as a* command line option, this line of *par* file is ignored, and the file specified after *–li* is used instead.

ligand_reference = "filename"
Optional parameter; no default value
Example: ligand_reference = "ligand_reference.pdb"

Specifies name of a file containing such reference ligand from PDB or other source that is positioned in the same binding site of the same protein and in the same coordinate system. This option is useful when you already have at least one resolved protein-ligand structure and want to dock new ligands to the same protein's site. The reference structure is then used to determine the center and the size of energy grid maps. If the reference ligand coincides with the docked ligand, which may be the case when checking program's performance on experimentally resolved structures, RMSD between the reference and predicted ligand poses will be calculated and saved in a *log* file. Note that reference ligand does not necessarily be a chemically consistent structure; when actual reference ligand is missing, you can take some points in the ligand binding site and write them into reference ligand structure (say in *pdb* format); this may help in determining overall placement of docking solutions.

additional_reference = "filename1"
additional_reference = "filename2"
…
Optional parameter; no default value
Example: additional_reference = "ligand_reference1.pdb"

Specifies coordinates of additional reference ligands (if needed). Additional reference ligands can be used for adequate RMSD calculations of docked ligand pose and reference ligand when a number of symmetric ligand-binding positions are valid. In such case, when a number of corresponding (symmetry related) reference ligand positions are provided, RMSD will be calculated with respect to the nearest reference. Anyway, accuracy of ligand docking does not depend on additional reference ligands.

covalent = A B X
Optional parameter; no default value
Example: covalent = 2 2542 1.542

When ligand is covalently bound to a protein, numbers of bonded atoms must be provided. Specify A as the ligand's bonded atom number, and B as the protein's bonded atom number. The enumeration for ligand and protein starts from zero. Additional parameter X specifies covalent bond length linking ligand to the protein. This parameter is optional (specifying A and B will be enough in principle) but highly recommended, since when ligand coordinates are arbitrary (which is a common case), specified bond length will position ligand correctly with respect to protein. Ignoring X is valid only when ligand coordinates are already compatible with covalent bond constraints.

Protein file

receptor = "filename"
Optional parameter; no default value
Example: receptor = "protein.pdb"

Specify name of a file in *pdb, gro, mol2* formats containing protein coordinates. When Lead-Finder is run with *–mm as a* command line option, this line of *par* file is ignored and the file specified after *–mm* is used instead. Atom names (of amino acid residues, cofactors, metal ions, etc.) listed in protein coordinate file must correspond to standard atom names (contained in *dock_ff.lib* library file).

exclude_water = yes | no
Optional parameter; default value is yes
Example: exclude_water = no

Specify this option to delete water molecules from protein structure file or retain them during docking calculations. This option can be useful when structurally conserved water molecule(s) is(are) necessary for ligand binding.

metal = A B
Optional parameter; no default value
Example: metal = 30 6

When automatic determination of a metal coordination number is ambiguous, it may be useful to explicitly define the metal coordination number. A is the metal atomic number (enumeration starts from zero). B is the metal coordination number that is used in the calculation of a directional potential of metal-ligand interactions. When there are several metal ions in the ligand binding site, each ion can be described by its own string.

Energy grid maps

grid_center = reference | X Y Z
Required parameter; default value is reference
Example: grid_center = 13.5 46.3 10.4

Choose one of the two methods for grid center placement: *reference* – center at the reference ligand if one is used. Alternatively, specify three Cartesian coordinates in *X, Y and Z* (in Å).

grid_spacing = X
Required parameter; default (and recommended) value is 0.375
Example: grid_spacing = 0.375

Distance between neighboring grid points along particular Cartesian direction (in Å).

grid_size = reference X | A B C
Required parameter; no default value
Example: grid_size = 10 10 10

Choose one of the two methods for grid size definition: (1) *reference* – a grid with X Å margin in each dimension from the reference ligand, default value of margin is 6 Å; (2) an explicit definition *grid_size = A B C* with three dimensions (lengths) of the grid box edges (in Å). Alternatively, grid size can be defined by setting the number of grid points with *grid_npoints* parameter instead.

grid_npoints = A B C
Optional parameter; no default value
Example: grid_npoints = 120 60 65

Specify grid size in a number of grid points in each dimension. *grid_npoints and grid_size* parameters are mutually exclusive.

grid_atom_types = auto | common | <individual atoms>
Optional parameter; default value is auto
Example: grid_atom_types = auto

Specify atoms for which grid maps will be calculated: *auto* – automatic extraction of atoms from ligand structure; *common* – all basic atoms (C, A, N, NX, O, S, H, P, F, Cl, Br, I)[7]; also, individual atoms can be specified, for example: *grid_atom_types = C A N NX O H*. Specify atom types with care, since during virtual screening of a big library of compounds different ligand atom types may emerge, so that it is better to calculate more grids than to lack them during high-throughput calculations.

reorient_grid = yes | no
Optional parameter; default value is yes
Example: reorient_grid = yes

Specify this parameter to reorient grids to maximally overlap with the binding site, as determined by the built-in cavity detection algorithm. Reorientation allows reduction of the grid size due to more efficient spatial arrangement.

exclude_volume = V
Optional parameter; default value is 40.5 (the volume of benzene molecule)
Example: exclude_volume = 40.5

Do not consider intramural protein cavities with volume less than V (in Å^3).

Other docking parameters

n_out = N
Optional parameter; default value is 20
Example: n_out = 20

Specify number of predicted docked positions (poses) to be output.

rot_sulfamide = yes | no
Optional parameter; default value is no
Example: rot_sulfamide = yes

Turn on/off the rotation of sulfamide bond during docking.

rot_amide = yes | no
Optional parameter; default value is no
Example: rot_amide = yes

Turn on/off the rotation of amide bond during docking.

rot_conjugated = yes | no
Optional parameter; default value is no
Example: rot_conjugated = no

Turn on/off the rotation of conjugated double or aromatic bonds during docking.

environment = solution | membrane
Optional parameter; default value is solution
Example: environment = membrane

Choose the type of environment surrounding the ligand binding site. Choose *membrane* when ligand binds to the membrane-buried protein site; in this case specially adjusted settings of electrostatic calculations (and other energy terms) will be activated. Default environment is *solution*.

[7] A – is aromatic carbon, NX – is a nitrogen that doesn't form H-bonds (for example, an amide nitrogen).

Command Line Options

You can use command line parameters to set-up Lead-Finder tasks and override some configuration options specified in a ***par*** file. The following command line parameters are accepted:

-f [file.par]
The docking parameter file. Default docking settings will be used if this parameter is not specified. See" for more details.

-li [ligand.mol]
The ligand structure in a *mol, sdf, mol2, pdb or gro* file.

-mm [protein.pdb]
The macromolecule (protein) structure in a *pdb, gro, mol2* file.

-og [grid.bin]
Calculate and save grid maps after docking. The acceptable grid file formats are *.bin* (binary format) and *.map* (human-readable format). Grid files with *.map* extension can be visualized with VMD software, but they are larger in size and are slower to load than the binary files.

-g [grid.bin]
Load grid maps from file and skip grid map calculations.

-o [solutions.pdb]
Output 3D-coordinates (*pdb*) of docked poses.

-os [brief_output.log]
Output only ligand's name and calculated energy (dG and VS score) into a text file. This option may be useful when docking many ligands from an *sdf* file.
-omm [protein.pdb]
Output a protein structure file in *pdb, gro, mol2* formats. The binary file with protein structure can be used subsequently as input for docking as it allows faster processing.

-l [solutions.log]
Output free energy of ligand binding for each docked pose in a human-readable *log* file.
-task [dock, scr]
Specify one of the two available lists of settings for the conformational search algorithm: docking (*dock*) or virtual screening (*scr*). Settings are optimized with respect to docking speed/robustness with screening being the fastest and docking – the most precise (details can be found in the Technology section of Lead-Finder internet site).

-mode [norm, grid, check, dG]
Choose one of the following modes of operation: *norm* - run docking calculations; *grid* – calculate energy grid maps without doing docking; *check* – verbose mode, check the ligand and protein structure at input and diagnose problems; *dG* – calculate free energies of binding.

-h
Print help for using command-line options.

-v
Display information on structure preparation and docking while calculations are in progress.
-debug
Dump of intermediate results, which can be sent to Lead-Finder developers for maintenance purposes.

Example 1: Calculating energy grid maps.

Lead-Finder may be used to calculate energy grid maps only, without doing docking or other tasks. This function is useful when you want to calculate grid maps and store them for future use with other docking experiments, thereby reducing the amount of processing and speeding up docking calculations.

The folder *Example 1* contains a 3D-structure of streptavidin (PDB 1SRE) in *protein.gro* and model settings for energy grid calculations in *1sre.par*. The settings in *1sre.par* are described below:

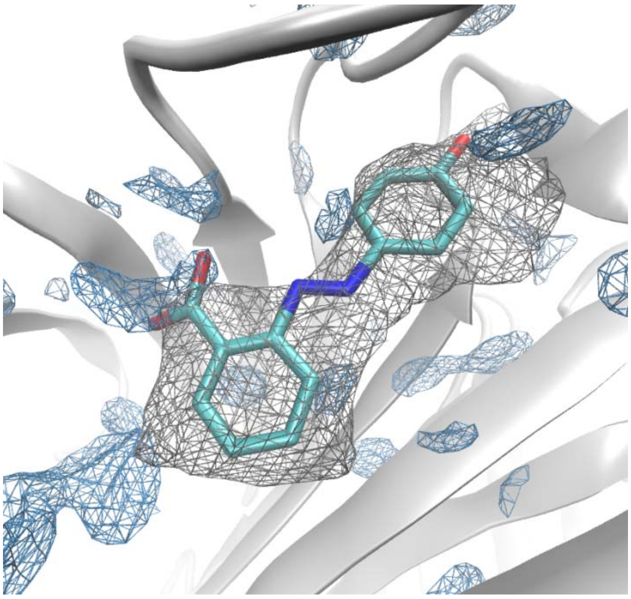
```
receptor = "protein.gro“          ; the protein structure will be taken from "protein.gro" file
exclude_water = yes               ; water molecules will be removed from the protein
                                  structure
grid_center = 33 13 -6            ; set the grid center at (33;13;-6) XYZ coordinates, in Å
grid_size = 15 15 15              ; the grid will span 15 Å in each direction
grid_atom_types = common          ; grid maps will be calculated for all types of atoms
reorient_grid = no                ; do not reorient grid to enable visualization of a ligand
                           with the protein structure
```

Start grid calculations with the following command line:

```
Lead_finder -f 1sre.par -mode grid -og 1sre.map
```

The calculated energy grid maps will be saved in *1sre.map* file. Note that *map* files are loaded to memory sufficiently slower than *bin* files; thus for re-using calculated energy grid maps in future docking experiments it is practical to save them as *bin* files.

The advantage of *map* files (*map* is AutoDock file format) is that they can be visualized with freely available VMD software. Also, this option may be useful in qualitative analysis of ligand binding forces.



This is an example of visualization of grid *map* files with VMD software. The structure of streptavidin, 1SRE, is rendered in ‘new cartoon’, reference ligand (2-((4'-hydroxyphenyl)-azo)benzoic acid) - in licorice, VdW grid - in gray wireframe, H-bond donor grid – in blue wireframe. The reference ligand’s shape seems to fit well into the protein’s binding site and the ligand’s carboxylic group is placed correctly for H-bonding interactions.
Other grid maps (electrostatic, etc.) may be loaded to VMD and visualized as well.

Example 2: Calculating free energy of binding.

Lead-Finder can be used to calculate free energy (ΔG) of ligand binding to protein when structure of the protein-ligand complex is available (from X-ray, NMR, molecular modeling studies, etc.). For this purpose the program should be launched in *-mode dG* and the structures of protein and ligand from the complex should be provided in separate files.

The folder *Example 2* contains a 3D-structure of HIV-1 protease (PDB 1HSG) in *protein.gro* and a structure of the ligand in *ligand.mol*. The coordinates of protein's and ligand's heavy atoms were taken from PDB and necessary hydrogen atoms were added. As a first step, Lead-Finder calculates energy grid maps. Then, it performs optimization of ligand's position and calculates free energy of ligand binding. The file *1hsg.par* contains parameters for the free energy of binding calculations:

```
ligand = "ligand.mol"                    ; the ligand
ligand_reference = "ligand.mol"          ; the reference ligand
receptor = "protein.gro"          ; the protein structure
exclude_water =  yes              ; remove water molecules will from the protein structure
grid_center = reference           ; place the grid center in the geometric center of
                                  the reference ligand
grid_size = reference              ; determine the grid size by the reference ligand
grid_atom_types = auto             ; calculate grid maps for all types of atoms present
                                  in "ligand.mol"
```

Start calculations of free energy of binding with the following command line:

```
Lead_finder -f 1hsg.par -mode dG
```

The free energy of binding and its decomposition to different terms will be saved in text *log* file. The experimental value of the free energy of binding is –13.2 kcal/mol.

Example 3: Ligand docking with pre-calculated grid maps.

When energy grid maps are available (were calculated earlier), they can be directly used for docking without recalculating them. For this purpose key *–g* in the command line should be used to point to the grid map file.
Folder *Example 3* contains streptavidin 3D-structure (*protein.gro*) ready for docking, grid map file (*1sre.map*), biotin structure (*ligand.mol*) and parameter file (*1sre.par*) described in *Example1*.

Docking with pre-calculated grid can be launched with the string:

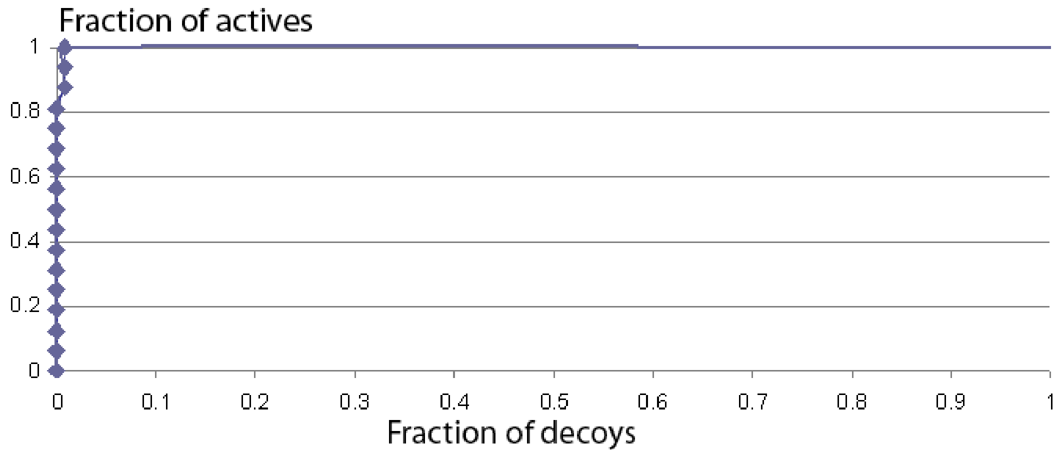
```
Lead_finder -f 1sre.par -task dock -g 1sre.map
```

Calculated ligand poses will be stored in *solution.pdb* file, energies – in *solution.log*.

Example 4: Virtual screening.

Lead-Finder can be used for docking a big number of ligands into single protein thus acting as *in silico* analogue of high-throughput screening. To achieve the fastest docking with Lead-Finder key *–task scr* should be used in the command line.
To perform virtual screening energy grid maps should be calculated first. Then, depending on the configuration of your computational resource (PC, a cluster of PCs, etc.), successive or parallel execution of Lead-Finder should be applied to all ligands of interest. This can be done in many ways, for example using Perl scripting; consult your system administrator how to organize screening in your case.
Folder *Example 4* contains all necessary files to perform virtual screening in a successive mode (on a single PC): 3D-structure of human serum albumin (*protein.gro*) prepared from PDB entry 1B5J; set of decoy ligands (directory *decoys*); set of true albumin ligands (directory *ligands*); parameter file (*screen.par*); and a Perl script (*do_screen.pl* for Linux or *do_screen_win.pl* for Windows), which launches grid calculations and successive docking of all ligands.

Results of screening can be processed (by another script) and gathered in a single file (*stats.txt*), which can be analyzed with standard table processing software (say MS Excel).



Enrichment curve for virtual screening of human serum albumin ligands: X-axis represents the fraction of active albumin ligands (known to bind with albumin with good affinity), Y-axis represents the fraction of decoy ligands. Each point on the curve reveals the ratio of true positives (active ligands) to false positives (decoys) at the certain fraction of the library screened. Totally 16 active ligands and 84 decoys (randomly chosen from the set of drug-like compounds) were used for screening. Results show that in the list of compounds ranged by their calculated affinity only one decoy compound occupies 14-th place, while active ligands occupy 1-13 and 15-17 places. ROC[8] - integral parameter describing the accuracy of virtual screening – equals 0.9987, which is excellent result.

[8] ROC – receiver operating characteristic – corresponds to the area under the normalized enrichment curve in coordinates: fraction of found true active compounds (from 0 to 1) vs fraction of decoys (from 0 to 1). Ideal curve reflecting 100% of true actives found at 0% of decoys gives ROC = 1.00.

Example 5: Preparation of trypsin structure for docking.

This example illustrates the necessary steps of protein and ligand structure preparation for docking.

1. Download trypsin structure (1TNL) from PDB at www.rcsb.org
2. Cut the ligand's (trans-2-phenylcyclopropylamine) coordinates (with any text editor) and save it as a separate file *ligand_reference.pdb*.
3. As far as a number of protein residues, for which alternative protonation states may take place, appear in the ligand's proximity, special care should be taken while adding hydrogen atoms to PDB structure. For this purpose you may launch Model_build program supplied with Lead-Finder distribution

```
Model_build –f 1tnl.pdb –o protein.gro –ph 7
```

Next, ligand structure should be prepared for docking. For this purpose all hydrogen atoms must be added to ligand. ACD software, for example, may be used for adding all hydrogens. Note that correct ionization of the ligand must be performed; this can be done with ACD/pKa module. In principle, any third party software may be used that can correctly add hydrogens to the ligand.

4.Docking may be launched with the string:

```
Lead_finder -f 1tnl.par -o 1tnl_solution.pdb -l 1tnl_energy.log
```

Coordinates of found solutions will be written in *1tnl_solution.pdb* file and energies – in *1tnl_energy.log*.

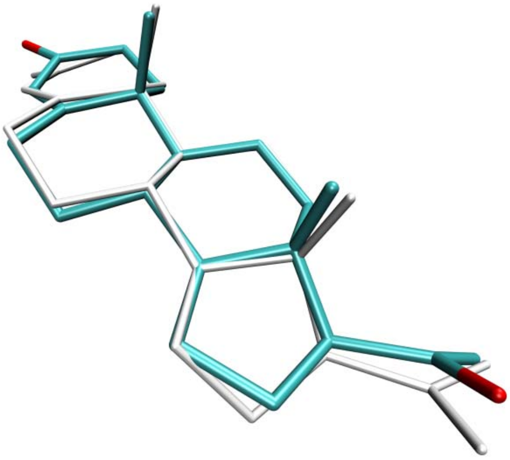
Example 6: Preparation of progesterone structure for docking.

This example outlines the necessity of careful handling with structures from PDB, which may have experimental defects like unresolved amino acid residues in protein or distorted ligand's geometry.

1. Download progesterone receptor structure (1A28) from PDB at www.rcsb.org
2. Cut the ligand's (progesterone) coordinates (with any text editor) from *pdb* file and save it as a separate file *ligand_reference.pdb*. Note that *pdb* file contains two identical protein chains, each having bound ligand. Thus, delete one chain (with its ligand) from the original *pdb* file.
3. In 1A28 structure some of the residues are not resolved experimentally. You can build them with freely available Swiss-PDB Viewer program (or other proper software). Save structure with added residues in a separate file *1a28_corrected.pdb*.
4. As far as a number of protein residues, for which alternative protonation states may take place, appear in the ligand's proximity, special care should be taken in adding hydrogen atoms to PDB structure. You may use Model_build program for this purpose (as described in *Example 5*).
5. Next, convert ligand's coordinates to *mol* file as it was described in *Example 5*.
6. Pay attention to the fact that ligand's geometry in 1A28 structure was initially distorted and due to this obstacle *mol* file contains wrong double bonds. You must correct bond orders before adding hydrogens. This can be done in ACD/ChemSketch (or other proper software) via graphical menu.
7. When bonds in the *mol* file are corrected, hydrogen atoms must be added to the ligand. This can be done as described in *Example 5*.
8. When protein and ligand structures are ready, docking parameter file should be provided (by analogy with *Example 5*) and docking launched with the string:

```
Lead_finder -f 1a28.par -o 1a28_solution.pdb -l 1a28_energy.log
```

Coordinates of solutions will be written in *1a28_solution.pdb* file and energies – in *1a28_energy.log*.



Superposition of deficient ligand (progesterone) structure extracted from PDB (white) and energy-optimized structure (cyan). Seemingly small differences of the overall ligand geometry (almost all bond lengths in PDB structure are distorted by 0.2-0.4 Å) influence docking precision.

Example 7: Preparation of methanol dehydrogenase structure for docking.

This example outlines the necessity of careful handling with target protein structure, which may need to retain non-protein residues (metal ions, water, etc) essential for catalytic competency of a protein.

1. Download methanol dehydrogenase structure (4AAH) from PDB at www.rcsb.org
2. Cut the ligand's (pyrroloquinoline quinone) coordinates (with any text editor) from *pdb* file and save it as a separate file *ligand_reference.pdb*. Delete unnecessary protein chains (B and D) from the original *pdb* file.
3. This enzyme contains catalytically important calcium ion and water molecule (residue number 785), which should be retained in the structure, while other water molecules may be deleted from *pdb* file.
4. Add hydrogen atoms to protein as it was described earlier. Note that orientation of hydrogen atoms in water molecule bound to calcium should be correct. Correction of water orientation can be performed for example with freely available VMD software.
5. Add all hydrogen atoms to ligand as it was described earlier.
6. When protein and ligand structure files are ready, remember that we want to retain catalytically important water molecule for docking, so chose *exclude_water = no* in *par* file.
7. To demonstrate flexibility of grid maps definition in Lead-Finder try setting the grid center explicitly via three Cartesian coordinates. Par file will look like:

```
ligand = "ligand.mol"                ; ligand will be taken from  "ligand.mol‟
ligand_reference = "ligand_reference.pdb "  ; reference ligand will be taken from
                                              "ligand.pdb‟
receptor = "protein.gro"             ; protein will be taken from "protein.gro‟
exclude_water = no                    ; water will not be excluded
grid_center = -15 19 15               ; XYZ coordinates of grid center
grid_size = reference                 ; grid size will be determined by reference ligand
grid_atom_types = auto                ; grid maps will be calculated for all types of atoms
                                     present in "ligand.mol"
```

8. Launch docking with the most precise settings of search algorithm with key *–task dock* standing for the default docking regime:

```
Lead_finder -f 4aah.par -o 4aah_solution.pdb -l 4aah_energy.log -task dock
```

   Coordinates of found solutions will be written in *4aah_solution.pdb* fileand energies – in *4aah_energy.log*.

9. You may want to compare Lead-Finder docking accuracy achieved with different settings of docking search algorithm. Launch docking using already generated structure, grid and parameter files but with different values of key *–task* equal: *dock* (default docking settings), *scr* (fast screening mode).
10. .Compare accuracy of structure prediction (RMSD is calculated in *4aah_energy.log* file for each pose when reference ligand is provided) and speed of calculations (CPU time is also provided in *4aah_energy.log*)[9].

[9] The following results were achieved by us (using AMD Athlon XP 1.8 GHz with 2 GB RAM) : screening mode - 0.74 Å (RMSD), 2 s (CPU time); docking mode - 0.70 Å, 5 s.

Example 8: Docking to cytochrome p450.

This example outlines the necessity of careful handling with target protein structure, which may need to retain cofactor molecules essential for catalytic competency of a protein.

1. Download cytochrome p450cam structure (5CPP) from PDB at www.rcsb.org
2. Cut the ligand's (adamantanone) coordinates (with any text editor) from *pdb* file and save it as a separate file *ligand_reference.pdb*.
3. This enzyme contains catalytically important cofactor – heme, which is essential for cytochrome substrate binding, and must be retained for docking.
4. Add hydrogen atoms to protein as it was described earlier.
5. Add all hydrogen atoms to ligand as it was described above.
6. Use standard docking parameter file (as described in previous examples).
7. Launch docking with key *–task scr* standing for fast screening regime:

```
Lead_finder -f 5cpp.par -o 5cpp_solution.pdb -l 5cpp_energy.log -task scr
```

8. You may want to compare Lead-Finder docking accuracy achieved with different settings of docking search algorithm. Launch docking with different key *–task* values and compare accuracy of structure prediction and speed of calculations (as it was described in *Example 7*)[10].

[10] The following results were achieved by us (using AMD Athlon XP 1.8 GHz with 2 GB RAM): screening mode - 1.16 Å (RMSD), 0.9 s (CPU time); docking mode - 0.60 Å, 2 s.

Glossary

Binding affinity – free energy of protein-ligand binding (kcal/mol).

dG score – Lead-Finder estimation of binding affinity.

Docking – positioning of the ligand in the protein binding site with respect to optimal free energy of binding. For details see Technology section of Lead-Finder internet site.

Energy grid – 3D function placing in accordance a point in Cartesian space and energy of particular type of interaction, which would atom of particular type possess being placed at this point. Correspondingly, grids are calculated for each type of interactions and each atom type.

HASP key – USB device that controls your software license

Virtual screening – rank-ordering of compounds (from particular library) according to estimated binding potency with respect to particular protein. Lead-Finder docks each compound from the library and calculates VS-score for docked ligand, which serves a quantitative measure of ligand activity (binding potency).

VS score - Lead-Finder estimation of ligand activity (binding potency) for rank-ordering ligands during virtual screening. For details see Technology section of Lead-Finder internet site.