



**OpenEye**  
Scientific Software

**SZYBKI**

*Release 1.5.0*

**OpenEye Scientific Software, Inc.**

July 20, 2010



# CONTENTS

<b>1</b>	<b>Front Matter</b>	<b>1</b>
<b>2</b>	<b>Installation and Platform Notes</b>	<b>3</b>
2.1	Licenses . . . . .	3
2.2	General Installation . . . . .	3
2.3	PVM . . . . .	5
2.4	Open MPI . . . . .	6
<b>3</b>	<b>SZYBKI Theory</b>	<b>9</b>
3.1	SZYBKI Theory . . . . .	9
<b>4</b>	<b>Application</b>	<b>11</b>
4.1	Usage . . . . .	11
<b>5</b>	<b>Release Notes</b>	<b>19</b>
5.1	SZYBKI 1.5.0 ( <i>July 2010</i> ) . . . . .	19
5.2	SZYBKI 1.3.4 . . . . .	19
5.3	SZYBKI 1.3.3 . . . . .	20
5.4	SZYBKI 1.3.2 . . . . .	20
5.5	SZYBKI 1.3.1 . . . . .	20
5.6	SZYBKI 1.3.0 . . . . .	21
5.7	SZYBKI 1.2.2 . . . . .	21
	<b>Bibliography</b>	<b>23</b>
	<b>Index</b>	<b>25</b>



# FRONT MATTER

Copyright 1997-2010 OpenEye Scientific Software, Santa Fe, New Mexico. All rights reserved.

All rights reserved. This material contains proprietary information of OpenEye Scientific Software. Use of copyright notice is precautionary only and does not imply publication or disclosure.

The information supplied in this document is believed to be true but no liability is assumed for its use or the infringement of the rights of others resulting from its use. Information in this document is subject to change without notice and does not represent a commitment on the part of OpenEye Scientific Software.

This package is sold/licensed/distributed subject to the condition that it shall not, by way of trade or otherwise, be lent, re-sold, hired out or otherwise circulated without OpenEye Scientific Software's prior consent, in any form of packaging or cover other than that in which it was produced. No part of this manual or accompanying documentation, may be reproduced, stored in a retrieval system on optical or magnetic disk, tape, CD, DVD or other medium, or transmitted in any form or by any means, electronic, mechanical, photocopying recording or otherwise for any purpose other than for the purchaser's personal use without a legal agreement or other written permission granted by OpenEye.

This product should not be used in the planning, construction, maintenance, operation or use of any nuclear facility nor the flight, navigation or communication of aircraft or ground support equipment. OpenEye Scientific software, shall not be liable, in whole or in part, for any claims arising from such use, including death, bankruptcy or outbreak of war.

Windows is a registered trademark of Microsoft Corporation. Apple, OS X, and Macintosh are registered trademarks of Apple Computer, Inc. AIX and IBM are registered trademarks of International Business Machines Corporation. UNIX is a registered trademark of the Open Group. RedHat is a registered trademark of RedHat, Inc. Linux is a registered trademark of Linus Torvalds. SPARC is a registered trademark of SPARC International Inc.

SYBYL is a registered trademark of TRIPOS, Inc. MDL is a registered trademark and ISIS is a trademark of Symyx Technologies, Inc. SMILES, SMARTS, and SMIRKS may be trademarks of Daylight Chemical Information Systems. Macromodel is a trademark of Schrodinger, Inc.

Python is a trademark of the Python Software Foundation. Java is a trademark or registered trademark of Sun Microsystems, Inc. in the U.S. and other countries.

Other products and software packages referenced in this document are trademarks and registered trademarks of their respective vendors or manufacturers.



---

# INSTALLATION AND PLATFORM NOTES

## 2.1 Licenses

To run SZYBKI you will need to obtain a license file for SZYBKI from OpenEye Scientific Software ([business@eyesopen.com](mailto:business@eyesopen.com)). The license file should be pointed to by the environment variable `OE_LICENSE`.

If you intend on running multi-processor SZYBKI via PVM or Open MPI, only the master machine needs access to the license file.

On Windows, the environment variables can be set under the system Control Panel.

## 2.2 General Installation

By default, all OpenEye applications are installed into a single distribution directory tree on the specified machine. The default location for this tree is platform specific and will be detailed below.

The root of the tree (i.e. the `openeye` directory) contains the following subdirectories:

- admin** This directory is intended to contain any administrative scripts and tools associated with the installed applications. Currently, this directory is simply a placeholder on all platforms except for Microsoft Windows, where it contains the uninstaller executables.
- arch** This directory contains the collection of platform specific subdirectories. Each subdirectory contains the actual installed executables and support libraries for the associated platform. In the platform specific subdirectory, there will be a subdirectory for each application and within that will be another subdirectory for each version of that application.
- bin** This directory contains a startup script for each application that has been installed. This script determines at run-time what the current platform is and then calls the appropriate executable in the `arch`. This script enables the easy co-existence of multiple platforms and versions of any OpenEye application in the same distribution tree.
- data** This directory contains all of the associated data for the installed applications. There will be a subdirectory for each installed application and within that subdirectory there will be another subdirectory for each specific version of that application.
- docs** This directory contains all of the documentation associated with the installed applications. There will be a subdirectory for each installed application and within that subdirectory there will be another subdirectory for each specific version of that application.

**examples** This directory contains all of the examples associated with the installed applications. There will be a subdirectory for each installed application and within that subdirectory there will be another subdirectory for each specific version of that application.

The startup script discussed in the section on the `bin` directory above will have the same name as the installed executable with which it is associated. When the script is called, it will attempt to determine the current platform and run the appropriate executable if installed. If an appropriate executable cannot be found, the script will report that information as well as a list of the currently installed platforms. The auto-detection can be overridden by setting one of two environment variables:

- **OE\_ARCH** can be used to specify a colon separated list of compatible distributions for the current platform such as:

```
redhat-RHEL5-x64:redhat-RHEL4-x64
```

Specification of this environment variable overrides the auto-detection process if it is present. If none of the compatible distributions listed are found, the script will fall back to the auto-detection process.

- **APPNAME\_OE\_ARCH** can be used to specify a colon separated list of compatible distributions for a specific application (as specified by changing the **APPNAME** text in the environment variable name) just like **OE\_ARCH** as detailed above.

Specification of this environment variable overrides the **OE\_ARCH** environment variable as well as the auto-detection process. If none of the compatible distributions listed are found, the script will fall back to the **OE\_ARCH** list first and then to the auto-detection process.

Specifying this variable provides a simple way to customize the behavior for individual applications on non-standard platforms.

The startup script also supports a few commandline arguments including:

- |                     |  |
|---------------------|--|
| <b>-path</b>        | Specifying this argument will output the full path of the executable to be run. The executable will not be started if this argument is present.  |
| <b>-print_arch</b>  | Specifying this argument will output the details of the current platform as detected by the script as well as which platform-version of the executable is being run. The executable will be started if this argument is present. |
| <b>-use_version</b> | Specifying this argument followed by a specific version number allows the user to control which released version of the executable to run.   |

## 2.2.1 Linux/Unix

Linux/Unix distributions are provided as a gzipped tarball of the distribution tree described above. Installation is performed by untarring the file in the desired location. Multiple distributions can be installed in the same location without any challenge.

To ensure that the installed applications can be called from the command line, be sure to add the full path of the `openeye/bin` subdirectory to the **PATH** environment variable. For instance, if the distribution was installed into `/usr/local/openeye`, the **PATH** environment variable should contain: `/usr/local/openeye/bin`.

## 2.2.2 Windows

Windows distributions are provided as a standard EXE installer. By default, all OpenEye applications will install into the `C:\OpenEye` directory.

An OpenEye group with an application specific subgroup will be added to the *Start* menu. The application specific subgroup will contain links to the documentation, the uninstaller, as well as to a Windows command shell which has

the appropriate **PATH** settings already defined to allow the user to simply type the executable name at the prompt without concern for where the executable is actually installed.

For GUI applications, a link to the application will be created on the desktop as well as in the application specific subgroup of the *Start* menu.

### 2.2.3 Mac OS X

Mac OS X distributions are provided as a standard *pkg* installer delivered as a *dmg* disk image. By default, all OpenEye applications will install into the `/Applications/OpenEye` directory.

To ensure that the installed applications can be called from the command line in the *Terminal*, be sure to add `/Applications/OpenEye/bin` to the **PATH** environment variable.

For GUI applications, an application bundle which can be clicked on to start, will be present in the `/Applications/OpenEye` directory. This bundle cannot be moved independent of the `OpenEye` directory. For instance, the entire `OpenEye` directory can be moved as one piece, but moving the application bundle or the contents of any of the subdirectories in the `OpenEye` directory may cause the application to not start. However, the bundle can still be dragged into the Dock and run from there without any problem.

## 2.3 PVM

PVM or parallel virtual machine is a freely available library for running processes on more than one processor on one or more machines ([Geist-1994]). SZYBKI can take advantage of PVM to distribute jobs over multiple processors. To do this PVM must be installed on all the machines SZYBKI will be distributed over. The PVM source is freely available from [http://www.csm.ornl.gov/pvm/pvm\\_home.html](http://www.csm.ornl.gov/pvm/pvm_home.html). However many Linux distributions, and some Unix versions, include PVM by default. SZYBKI is built with the current PVM version 3.4.5, but should also work with PVM version 3.4.4. SZYBKI does not support PVM under Windows.

To use SZYBKI with PVM you must do one of the following.

- Place a link to the `szybki` executable in `$PVM_ROOT/bin/$PVM_ARCH`
- Define the environment variable **PVM\_PATH**, which points to the directory where the SZYBKI executable resides.

The environment variables **PVM\_ROOT** and **PVM\_ARCH** should be defined globally as part of the PVM installation. **PVM\_PATH** is generally a user-defined environment variable, and must be defined for all shells (*i.e.*, you can't just define it in the shell you're launching SZYBKI in).

**Note:** There is no specific slave executable. The executable distributed for this program serves as both a master and slave PVM program as well as a single processor version.

### 2.3.1 PVM hosts file

After PVM and SZYBKI are installed, each SZYBKI job run via PVM must be supplied a file (via `-pvmconf`) that describes the hosts and number of CPU's to use. Each line in this file includes the word **host** followed by the hostname of the slave followed by the number of CPU's to use on that slave.

To use 10 slaves on a linux cluster (where nodes are named *linux1*, *linux2*, etc.)

```
host linux1 2
host linux2 2
host linux3 2
```

```
host linux4 2
host linux5 2
```

To use 32 CPU's on a single-image system like an SGI Altix, a single line can be used:

```
host altix 32
```

## 2.3.2 PVM Scaling

PVM jobs involve a certain of network traffic, sending multi-conformer molecules from the master to the slaves and sending results from the slaves back to the master. The single SZYBKI ligand optimization slave jobs are relatively fast compared to the network I/O time, so for scaling beyond 32 CPUs, additional effort is required in setting up a job. For low numbers of CPUs or for divide-and-conquer style applications, using `.oeb.gz` as the dbase file format is fine.

## 2.4 Open MPI

As of version 1.5.0, SZYBKI supports MPI-1, or Message Passing Interface 1, on all platforms except Microsoft Windows, Solaris & AIX. MPI, like PVM, enables distributing SZYBKI runs over multiple machines and processors.

SZYBKI uses the Open MPI implementation of MPI, found at <http://www.open-mpi.org>. Every version of SZYBKI includes a full Open MPI install, so no additional software is needed.

There are two requirements to run SZYBKI under Open MPI.

- Every machine in the cluster must have SZYBKI installed.
- The path to the top level OpenEye `bin` directory must be in the **PATH** environment variable on all machines, and before any other locations that may contain MPI executables (`orted`, `mpirun`, etc).

**Note:** While the OpenEye MPI install must be first in the path, it will not conflict with other installs. OpenEye puts two MPI related scripts in the top level OpenEye `bin` directory. The first, `oempirun`, is a replacement for `mpirun`. Given their different names, there will not be conflict. The `orted` command in the OpenEye `bin` directory is a wrapper script around the `orted` daemon. If MPI was not run with `oempirun` and an OpenEye application, it will fall back to the next instance of `orted` in the path.

### 2.4.1 Using Open MPI

Running SZYBKI under Open MPI makes use of a wrapper script `oempirun` located in the OpenEye `bin` directory.

To run SZYBKI under Open MPI:

```
prompt> oempirun [Open MPI options] szybki [OMEGA Options]
```

For example, to run SZYBKI on 3 hosts `h1`, `h2` and `h3`:

```
prompt> oempirun -np 3 -host h1,h2,h3 szybki my_molecules.mol2
```

The above command line will run 3 SZYBKI processes on the hosts `h1`, `h2` and `h3`. The master machine, which needs a license, would be `h1`. It is important to note that the `szybki` passed to `oempirun` does not take a path.

## 2.4.2 Open MPI License

Most files in this release are marked with the copyrights of the organizations who have edited them. The copyrights below generally reflect members of the Open MPI core team who have contributed code to this release. The copyrights for code used under license from other parties are included in the corresponding files.

Copyright (c) 2004-2008 The Trustees of Indiana University and Indiana University Research and Technology Corporation. All rights reserved.

Copyright (c) 2004-2008 The University of Tennessee and The University of Tennessee Research Foundation. All rights reserved.

Copyright (c) 2004-2008 High Performance Computing Center Stuttgart, University of Stuttgart. All rights reserved.

Copyright (c) 2004-2007 The Regents of the University of California. All rights reserved.

Copyright (c) 2006-2008 Los Alamos National Security, LLC. All rights reserved.

Copyright (c) 2006-2008 Cisco Systems, Inc. All rights reserved.

Copyright (c) 2006-2008 Voltaire, Inc. All rights reserved.

Copyright (c) 2006-2008 Sandia National Laboratories. All rights reserved.

Copyright (c) 2006-2008 Sun Microsystems, Inc. All rights reserved. Use is subject to license terms.

Copyright (c) 2006-2008 The University of Houston. All rights reserved.

Copyright (c) 2006-2008 Myricom, Inc. All rights reserved.

Copyright (c) 2007-2008 UT-Battelle, LLC. All rights reserved.

Copyright (c) 2007-2008 IBM Corporation. All rights reserved.

Copyright (c) 1998-2005 Forschungszentrum Juelich, Juelich Supercomputing Centre, Federal Republic of Germany

Copyright (c) 2005-2008 ZIH, TU Dresden, Federal Republic of Germany

Copyright (c) 2007 Evergrid, Inc. All rights reserved.

Copyright (c) 2008 Institut National de Recherche en Informatique. All rights reserved.

Copyright (c) 2007 Lawrence Livermore National Security, LLC. All rights reserved.

Copyright (c) 2007-2008 Mellanox Technologies. All rights reserved.

Copyright (c) 2006 QLogic Corporation. All rights reserved.

Additional copyrights may follow

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer listed in this license in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holders nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

The copyright holders provide no reassurances that the source code

provided does not infringe any patent, copyright, or any other intellectual property rights of third parties. The copyright holders disclaim any liability to any recipient for claims brought against recipient by any third party for infringement of that parties intellectual property rights.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# SZYBKI THEORY

## 3.1 SZYBKI Theory

### 3.1.1 Force Field

The MMFF potential expression is:

$$V_{MMFF} = \sum_b V_b + \sum_a V_a + \sum_s V_s + \sum_o V_o + \sum_t V_t + \sum_v V_v + \sum_c V_c$$

where the seven terms respectively describe bond stretching (*b*), angle bending (*a*), stretch-bending (*s*), out-of-plane bending (*o*), torsion (*t*), Van der Waals (*v*) and electrostatic (*c*) interactions. Their functional forms are given below.

#### Bond stretching

For a bond *b* between atoms *i* and *j* the stretching potential is:

$$V_b = 143.9325 \frac{k_{ij}}{2} \Delta r_{ij}^2 (1 + c_s^2 \Delta r_{ij} + 7/12 c^2 \Delta r_{ij}^2)$$

where  $k_{ij}$  is the force constant,  $\Delta r_{ij}$  is the difference between actual and reference bond lengths, and  $c_s$  is so called “cubic-stretch” constant for which the value is  $-2/A^{-1}$ .

#### Angle bending

The bending potential of a bond angle *a* made by the bonds between atoms *i, j* and atoms *j, k* is given by:

$$V_a = 0.043844 \frac{k_{ijk}}{2} \Delta \vartheta_{ijk}^2 (1 + c_b \Delta \vartheta_{ijk})$$

where  $k_{ijk}$  is the force constant,  $\Delta \vartheta_{ijk}$  is the difference between actual and reference bond angles, and  $c_b$  is the so called “cubic-bend” constant for which the value is  $-0.4 rad^{-1}$ .

#### Stretch-bend interaction

The coupling between the stretching potential of two bonds forming a bond angle and bending that angle is described by:

$$V_s = 2.5121 (k_{ijk} \Delta r_{ij} + k_{kji} \Delta r_{kj}) \vartheta_{ijk}$$

where  $k_{ijk}$  and  $k_{kji}$  are force constants which couple stretches of *i-j* and *k-j* to the *i-j-k* bend respectively.  $\Delta r$  and  $\vartheta$  are defined above.

### Out-of-plane bending

For a trigonal center  $j$ , the potential of displacement for an atom  $l$  bonded to atom  $j$  out of plane  $i-j-k$  is:

$$V_o = 0.043844 \frac{k_{ijkl}}{2} \chi_{ijkl}^2$$

where  $k_{ijkl}$  is the force constant and  $\chi_{ijkl}$  is an angle formed by the bond  $j-l$  and the plane  $i-j-k$ .

### Torsion interactions

For every four bonded atoms  $i-j-k-l$  the torsion interaction is described by the term:

$$V_t = 0.5(V_1(1 + \cos \Phi) + V_2(1 - \cos 2\Phi) + V_3(1 + \cos 3\Phi))$$

where  $V_1$ ,  $V_2$  and  $V_3$  are constants depending on atoms  $i$ ,  $j$ ,  $k$ ,  $l$ , and  $\Phi$  is the dihedral angle formed by bonds  $i-j$  and  $k-l$ .

### Van der Waals interactions

For a pair of atoms  $i$  and  $j$  separated by three or more bonds, where the distance between them is  $r_{ij}$ , MMFF adopts the following Van der Waals potential:

$$V_v = \epsilon_{ij} \left( \frac{1.07R_{ij}}{r_{ij} + 0.07R_{ij}} \right)^7 \left( \frac{1.12R_{ij}^7}{r_{ij}^7 + 0.12R_{ij}} - 2 \right)$$

where  $R_{ij}$  and  $\epsilon_{ij}$  are defined as follows:

$$R_i = A_i \alpha_i^{0.25}$$

$$R_{ij} = 0.5(R_i + R_j)(1 + B(1 - \exp(-12\gamma_{ij}^2)))$$

$$\gamma_{ij} = (R_i - R_j)/(R_i + R_j)$$

$$\epsilon_{ij} = \frac{181.16G_iG_j\alpha_i\alpha_j}{(\alpha_i/N_i)^{0.5} + (\alpha_j/N_j)^{0.5}} \frac{1}{R_{ij}^6}$$

where  $\alpha_i$  is atomic polarizability of atom  $i$ ,  $B$  is 0.2 or 0.0 if one of the atoms is polar hydrogen,  $N_i$  and  $N_j$  are the Slater-Kirkwood effective numbers of valence electrons,  $G_i$ ,  $G_j$  and  $A_i$  are scale factors.

### Electrostatic interactions

The electrostatic interaction between two charged atoms  $i$  and  $j$  separated by at least three bonds is calculated from the standard Coulombic expression:

$$V_c = f \frac{332.0716q_iq_j}{D(r_{ij} + \delta)}$$

where  $D$  is the dielectric constant for which the default value is 1,  $q_i$  and  $q_j$  are the MMFF partial charges on atoms  $i$  and  $j$ ,  $r_{ij}$  is the interatomic distance,  $\delta$  is the "electrostatic buffering" constant of 0.05Å. Scaling factor  $f$  is 0.75 for 1,4 interactions, and 1.0 otherwise.

# APPLICATION

## 4.1 Usage

### 4.1.1 Command Line Interface

A description of the command line interface can be obtained by executing SZYBKI with the `--help` option.

```
prompt> szybki --help
```

will generate the following output:

```
Help functions:
  szybki --help simple      : Get a list of simple parameters
  szybki --help all        : Get a complete list of parameters
  szybki --help <parameter> : Get detailed help on a parameter
  szybki --help html       : Create an html help file for this program
```

### Parameters

#### **-complex <filename>**

Input file with 3D coordinates of a protein-ligand(s) complex in any format supported by OEChem. All ligands contained in the molecule file `filename` will be optimized one by one in the presence of all others. If in addition the option `-ligands` is used, all external ligands given with that option will be optimized as well. If there is spatial overlap with the internal ligand from the filename file, the internal ligand will be removed.

#### **-fix\_file <filename>**

File filename should contain a list of molecule names followed by atom numbers to be fixed, e.g: molecule1 0 10 molecule2 21 will fix atoms 0 and 10 in molecule1 and atom 21 in molecule2.

#### **-heavy\_rms**

By default all atoms RMSD after optimization is reported in the log file. This option replaces the default with the heavy atoms RMSD.

#### **-ligands <filename>**

Molecular input file name containing 3D coordinates in any format supported by OEChem. In order to comply with the previous SZYBKI releases, an alias `-i` can be used instead of `-ligands`. This option might be used without the `-ligands` or `-i` keys but only when used last or next to last when the last option on the command line is `-out`. Warning: File name: `szybki_out.mol2` is used as the default output file name, so should be avoided as an input file name.

**-loadPG <filename>**

In the case when the electrostatic component of the protein-ligand interaction energy has been pre-calculated on a grid, this option forces SZYBKI to read the grid potential from the file *filename*, and use it for ligand optimization inside the protein.

**-log <pre>**

Prefix of the log file name, *pre.log*. If omitted, the prefix *szybki* is used by default. This option is aliased to *-l*.

**-out <filename>**

Output file name, in any format supported by OEChem, for an optimized ligand. If not specified, *szybki\_out.mol2* or *prefix\_out.mol2* (when *-prefix* is used) will be generated. Alias *-o* can be used instead of *-out*. When options *-ligands* and *-complex* are used together, internal ligands from the protein-ligand complex will be output first. Can be used without the *-out* key when it is last on the command line just after the *-ligands* (or *-i*) option.

**-out\_protein <filename>**

Partially optimized protein will be saved in a file named *filename*.

**-param**

Command line options will be read from the specified file. This file may have been generated from a previous run or may be constructed de novo. The default name of the file is *szybki.param*. Any parameter in the setup file is superseded by the parameter on the command line. For example running *szybki -param szybki.param -i my.pdb* will perform calculations for the molecule *my.pdb* while using all other parameters taken from *szybki.param*.

**-prefix <pn>**

Replaces *szybki* prefix in *.log*, *.rpt* (report), *.status*, *.param* and *\_out.mol2* files, with the input string *pn*.

**-protein <filename>**

Protein input file with 3D coordinates in any format supported by OEChem. Usually works in combination with the *-ligands* option. No removal of pre-existing ligand in file *filename* will be done as in the case of the *-complex* option. When used without *-ligands* option, two scenarios will be followed depending on the presence of ligands in the file *filename*: If ligands are present, all of them will be optimized as in the case of *-complex* option. If not, the optimization of the entire protein will be attempted. The latter is not recommended however, because for larger proteins extreme memory usage can occur causing crashes. This option can be aliased with *-p* for compatibility with previous SZYBKI releases.

**-report**

An output file in tabular form of final energy components for every molecule/conformer will be generated. The name of the file is fixed as: *prefix.rpt* where *prefix* is *szybki* by default or determined by *-prefix* option.

**-reportFile <filename>**

Sets user selected file name, *filename*, for the tab file. Supersedes the *-report* option described above.

**-savePG <filename>**

Saves potential grid in the file named *fn*. Allows a significant saving in CPU time for runs when PB or Coulomb grids are used to optimize a number of ligands inside the same protein.

**-sdtag <string>**

In the case when the output file is in SD format an energy tag can be added. By default, the *string* is empty which means that all relevant tags are added. For single molecules, this means *Total\_energy* and all MMFF terms and solvation terms that are not zero, and for a protein-ligand system this means *Ligand-Protein Energy* and all of its components. The following potential components can be used for *string*, exactly as shown:

- 1.MMFF VdW
- 2.MMFF Coulomb

3.MMFF Bond  
4.MMFF Bend  
5.MMFF StretchBend  
6.MMFF Torsion  
7.MMFF Improper Torsion  
8.L MMFF VdW  
9.L MMFF Coulomb  
10.L MMFF Bond  
11.L MMFF Bend  
12.L MMFF StretchBend  
13.L MMFF Torsion  
14.L MMFF Improper Torsion  
15.P MMFF VdW  
16.P MMFF Coulomb  
17.P MMFF Bond  
18.P MMFF Bend  
19.P MMFF StretchBend  
20.P MMFF Torsion  
21.P MMFF Improper Torsion  
22.PL MMFF VdW  
23.PL MMFF Coulomb  
24.Ligand-Protein Energy  
25.Sheffield Solvation  
26.Constraint Potential  
27.PB Solvent  
28.Area Solvent  
29.\_\_VdW  
30.\_\_Coulomb  
31.\_\_Protein desolv  
32.\_\_Ligand desolv  
33.\_\_Solvent screening  
34.\_\_Grid Coulomb  
35.\_\_Exact Coulomb

If parameter string does not correspond to one of the available energy terms, no tag will be written. Tags starting with L, P, PL and \_\_ are:

- L: ligand intra-molecular terms only

- P: protein intra-molecular terms only
- PL: protein-ligand inter-molecular terms only
- \_\_\_: protein-ligand interaction terms

**-silent**

By default the names of the processed molecules are displayed. The use of the option `-silent` will suppress this output.

**-verbose**

An extensive general output will be generated containing initial and final energy and gradient data, as well as an optimization report. This option is available only for single-processor mode.

## Potential Function Options

**-MMFF94s**

Modified set of MMFF94 parameters developed in 1999 by Halgren ([Halgren-1999-1], [Halgren-1999-2]) will be used.

**-am1bcc**

AM1BCC charges ([Jakalian-2002]) for PB or Sheffield calculated and used for every conformation.

**-exact\_vdw**

By default the VdW protein-ligand interaction is calculated with the use of lookup table in order to speedup the calculations. This option allows to use the exact analytical VdW potential for the optimization of a ligand in the protein binding site.

**-harm\_constr1 k**

Constrained potential of the form:  $kr^2$  will be imposed on all heavy atoms, where  $k$  is the force constant.

**-harm\_constr2 d**

Constrained potential of the form:  $V = k(r - d)^2, r > d$  and  $V = 0, r \leq d$  will be imposed on all heavy atoms, where  $d$  is the constraining distance. Can be used only together with `-harm_constr1`

**-harm\_smarts p**

All atoms which belong to SMARTS pattern  $p$  will be constrained. For example input parameter “-harm\_smarts cO” will result in constraining all aromatic carbon atoms and oxygen atoms bonded to them. Must be used in conjunction with `-harm_constr1`.

**-mod\_vdw**

Regular MMFF *Van der Waals interactions* equation will be replaced with:

$$V_{vdw} = \begin{cases} \epsilon_{ij} \left( \frac{1.07R_{ij}}{r_{ij} + 0.07R_{ij}} \right)^7 \left( \frac{1.12R_{ij}^7}{r_{ij}^7 + 0.12R_{ij}^7} - 2 \right) & \text{for } r_{ij} < R_{ij} \\ -\epsilon_{ij} & \text{for } r_{ij} \geq R_{ij} \end{cases}$$

in which no attractive VdW forces are present. This type of VdW potential prevents so called “hydrophobic collapse”.

**-mol2charges**

During optimization of molecules in solution with the use of the Poisson-Boltzmann solvation model, free energy of solvation and solvent forces can be calculated with the use of atomic partial charges other than MMFF. Option `-mol2charges` allows the use of partial charges read from the input mol2 file. In the case where the ligand is optimized inside a protein, protein-ligand electrostatic interactions (Coulomb and/or Poisson-Boltzmann) could be calculated based on ligand and protein partial charges read from the mol2 input file(s). When input partial charges are not found, the MMFF94 partial charges will be used.

**-neglect\_frozen**

After an optimization with frozen terms, the default behavior calls for a full single-point calculation of the whole

system. The `-neglect_frozen` option will skip this final calculation of the whole system, thus yielding results for only the non-frozen pieces. When there is a large frozen section, this can drastically reduce memory and cpu usage.

#### **-noCoulomb**

Electrostatic terms defined in the *Electrostatic interactions* section will be excluded from the force field potential. This option might be useful to prevent generation of folded structures.

#### **-protein\_dielectric <d>**

The default value for the protein dielectric constant of 1.0 can be changed to a user selected value `d`.

#### **-protein\_elec <m>**

This option provides 4 choices for calculating protein-ligand interaction energies. In all cases the MMFF VdW potential is used. Option `None` (`none`) eliminates electrostatic interactions. Values of `m` set at `ExactCoulomb` and `GridCoulomb` result in the usage of MMFF Coulomb exact potential and digitized on the grid respectively. Option `m = "PB"` provides a more realistic potential which accounts for solvent forces, calculated according to the Poisson-Boltzmann (PB) model at every iteration step. This option requires substantially higher CPU time, particularly for large proteins. By default `m = None`.

#### **-protein\_vdw <r>**

Calculation of VdW protein-ligand interaction energy will be limited to a sphere of radius `r`. The default value is 18.0 angstroms. In many applications a value as small as 10 angstroms can be used with essentially no effect on the final optimized ligand geometry.

#### **-shefA <a>**

parameter `a` in the Sheffield solvation potential given in option `-sheffield`. The default value is: 1.553149 ([Grant-2007]).

#### **-shefB <b>**

parameter `b` in the Sheffield solvation potential given in option `-sheffield`. The default value is: 0.735694 ([Grant-2007]).

#### **-sheffield**

Usage of this option will result in adding additional electrostatic terms of the form:  $\sqrt{(ar^2 + bR_iR_j)}$  where  $q_i, q_j$  are partial charges and  $R_i, R_j$  are Van der Waals radii of atoms `i` and `j` in order to mimic the solution environment ([Grant-2007]). This option might help to preserve unfolded structures during optimization.

#### **-solv\_dielectric <d>**

The default value for solvent dielectric constant used for Poisson-Boltzmann solvation energy calculations is 1. This option allows to change it to user selected value.

#### **-solventCA <s>**

This option can be used only in combination with `-solventPB` or `-sheffield`. It causes inclusion of a molecular surface solvation term (called sometimes cavity solvation term) in the total energy. The recommended value of parameter `s` is 0.025. This option can be aliased with `-solventMA`, for compatibility with previous releases.

#### **-solventPB**

For optimization of small molecules in solution, the electrostatic part of molecule-solvent interactions will be calculated using Poisson-Boltzmann model.

## Optimization Options

#### **-conj**

Conjugate gradient optimization will replace the default quasi-Newton BFGS method.

#### **-fix\_smarts <p>**

All atoms which belong to SMARTS pattern `p` will be fixed. For example input parameter `p = "[!#1]"` will fix

all heavy atoms and optimize all hydrogen atom positions.

**-grad\_conv <c>**

Optimization is terminated when the rms gradient reaches the input value *c* unless is finished earlier because of other reasons. When omitted the default convergence criteria is 0.1 on gradient vector norm:  $\sqrt{\sum_i g_i g_i}$ .

**-largest\_part**

Calculations are performed only for the largest fragment of the noncovalent input complex. For example when the input file contains coordinates for the salt,  $[Large\_cation^+] \cdot [Cl^-]$ , the use of the *-largest\_part* will cause the  $Cl^-$  anion to be ignored. By default calculations are done for the entire complex.

**-max\_iter <m>**

Optimization will be terminated when the number of iteration cycles reaches input number *m*.

**-no\_opt**

A single point calculation will be performed i.e. no optimization of ligands or the protein will be carried out.

**-opt\_cart**

Optimization of Cartesian atomic coordinates will be done. This is the default optimization type for molecules in vacuum or in solution.

**-opt\_solid**

Optimization of ligand position inside the protein receptor with frozen ligand internal ligand coordinates. Option is not valid and ignored without an input protein. Because it is the simplest and less expensive ligand optimization in the protein, it is the default behavior. The option is therefore only formal and redundant.

**-opt\_torsions**

Optimization of free rotor torsions for molecules in vacuum or torsions and translational/rotational degrees of freedom for ligands inside protein receptors. In the case when the ligand has no rotor torsions, rigid-body optimization will be performed.

**-polarH <r>**

This option allows partial flexibility of a protein receptor in the optimization of a ligand inside the protein binding side. All polar protein hydrogens distanced by *r* from the ligand will adjust their position upon energy relaxation. Main chain amide NH hydrogens of peptide groups are not included. The molecular potential used for optimization consists of three components:

1.Ligand intra-molecular MMFF potential

2.Protein-ligand potential

2.1. MMFF terms for interaction between polar protein hydrogens and the ligand

2.2. Interaction between fixed protein atoms and the ligand

3.Protein intra-molecular potential

3.1. MMFF terms involving polar protein hydrogens and atoms up to three bonds apart.

3.2. Interaction between the rest of the protein and flexible polar hydrogens

The sum of components 2.2 and 3.2 might be called “protein-pseudoligand” interaction and is evaluated according to the model selected with the *-protein\_elec* option above.

**-residue <r>**

Similar to polarH. This option allows for flexibility of complete residues having at least one atom within distance *r* of any atom of the ligand. There is no upper value of the *r* parameter, however using a large value will cause even distant residues to be flexible which could lead to poor results. The intention of this option is to allow only residue atoms interacting directly with the ligand to rearrange upon optimization.

**-sideC <r>**

Similar to polarH. This option allows for flexibility of all side chains having at least one atom within distance *r* of any atom of the ligand. Same comment as in the *-residue* above applies with respect to side chains atoms.

**-strip\_water**

Causes removal of water molecules from the input protein when options *-protein* or *-complex* are used.

**Thermodynamics calculations options****-entropy <type>**

Estimation of the entropy of a ligand will be done. Parameter *type* can take three values “None” (default), “QN” (Quasi-Newton Hessian) and “AN” (analytical). Input ligand is assumed in the form of a multiconformation molecule. The environment is determined by the option *-sheffield* (solution), *-protein* (in the binding site) or none of those two indicating a gas-phase.

**PVM Options****-pvmconf <f.txt>**

*f.txt* is a text file which specifies PVM processor configuration. For every host in the cluster it should contain line: *host host\_name n* where *n* is the number of processors on the host.



---

# RELEASE NOTES

## 5.1 SZYBKI 1.5.0 (*July 2010*)

### 5.1.1 New features

- Starting from the current release, certain classes of divalent selenium compounds can be handled by Szybki. Currently only some compounds in which selenium is bonded to the hydrogen or carbon atoms can be processed. Those compounds include selenols (RSeH) and selenides (RSeR1) in which Se is bonded to alkyl or aromatic carbon atoms, with the exception when Se makes a bond with alkyl C which belongs to the 3- or 4-membered ring.
- Previous Szybki releases attempted to assign the “closest” MMFF94 atom type to those atoms for which a unique MMFF94 type could not be found. Starting from the 1.5.0 release this feature is removed, so consequently the input molecules containing such atoms will not be processed.
- Ligand RMSD which is reported in the log file, can optionally refer to the heavy atoms only. The new option used for that purpose is `-heavy_rms`.
- Entropy calculations of a ligand in different environments can be performed based on the methods described in the recent paper of Wlodek *et. al.* ([Wlodek-2010]). This is done with the option `-entropy` described in the section 4.1 of this manual.
- Default VdW protein-ligand potential used for the optimization of a ligand inside the protein binding site is precalculated in the form of a lookup table. This is done for the purpose of speed. The exact VdW protein-ligand potential can be used with the option `-exact_vdw`.

### 5.1.2 Bug fixes

- Memory leak in ligand torsion optimization inside the partially flexible protein receptor was removed.

## 5.2 SZYBKI 1.3.4

### 5.2.1 New features

- Calculation of the constant potential terms at the end of adapted optimization can be optionally eliminated, by the use of the `-neglect_frozen` flag. This feature allows for less memory and cpu usage in the case of partial optimization of large molecules.

- The corresponding atom symbols, bond symbols, and torsion angles are now printed on a line under the matching smarts pattern.

## 5.2.2 Bug fixes

- Input proteins with only partial information about residues were not handled properly. As a result, none of the side chains which were supposed to be made flexible with the use of an option `-sideC` were optimized. This problem was fixed.
- A bug causing memory corruption in the case of ligand optimization in partially flexible protein receptor is fixed.
- Residues with non-standard name `CYX` were not properly handled in the situation when their side chains were included into the flexible part of the protein receptor. This bug has been fixed.

## 5.3 SZYBKI 1.3.3

### 5.3.1 Bug fixes

- Until current release, molecular output files in the `.oeb` format had MMFF aromaticity. In practice, such a situation lead to the visual change between the input and output aromaticity of some molecules. Starting from now OpenEye aromaticity model is used in the `.oeb` output files.

## 5.4 SZYBKI 1.3.2

### 5.4.1 Bug fixes

- A bug which occasionally caused errors in fixing a subset of atoms, was fixed.
- When using PVM, Szybki now runs a mechanism for checking versions used by master and slaves. This mechanism prevents the situation of different versions of Szybki running on the master and slaves.
- Molecules containing atoms for which MMFF types are not properly typed are not processed. In previous versions attempts were made to assign for such atom the closest MMFF type, however this procedure almost always lead to errors. This situation occurred for example in the case of bad `pdb` files. For proteins, better diagnostics (residue name and number, atom name) are included as output.
- Reading protein data in `.oeb` binary format containing additional untagged information (FRED receptor files) is allowed. Until now using such a file in `pvm` mode caused a fatal error.

## 5.5 SZYBKI 1.3.1

### 5.5.1 Bug fixes

- The combination of options `-complex if1` and `-ligands if2` run in the `pvm` mode caused incorrect behavior in the case when the preexisting ligand in the macromolecule (in the file `if1`) overlapped with one or more ligands from the input file `if2`.
- Run time license for the shape toolkit is added. It is required when the combination of options `-complex if1` and `-ligands if2` is used.
- Redundant warnings generated upon `-protein_elec none` and `-strip_water` were removed.

- Spelling in the log file was fixed.
- Year 2008 was added to the Szybki banner.
- In the documentation description for the `-fix_smarts` option an example was added which illustrates the adjustment of hydrogen atom positions.
- Examples of log files generated upon ligand optimization in the partially flexible protein followed by a single point PB calculations are given.

## 5.6 SZYBKI 1.3.0

### 5.6.1 New features

- Optimization using conjugate gradient method was added as an alternative to the default quasi-Newton method with the BFGS Hessian update scheme. In Szybki toolkit this is done by the usage of a new function `void SetOptimizerType(unsigned int type)`. In Szybki application a flag `-conj` has been added for that purpose.
- Default value of the protein dielectric constant is 1. Previous default value for that parameter was 2, however it was not consistent with the default value of MMFF94 force field.
- Protein-electrostatic models in Szybki application are selected with descriptive strings. Available options are:
  - None - no electrostatic potential used (default option)
  - ExactCoulomb - exact MMFF Coulomb potential
  - GridCoulomb - MMFF Coulomb potential on grid
  - PB - PP solvent forces used at every step

### 5.6.2 Bug fixes

- A bug causing crash in the case of combined usage of partial protein flexibility and protein electrostatic model `OEProteinElectrostatics::SolventPBForces` was fixed.
- A bug causing different behavior and/or occasional crash depending of the order of calls: `SetProtein(const OEChem::OEMolBase&)`, `SetRunType(unsigned int)` and `SetResidueFlexibility(double)` has been fixed.
- Modification of the potential function after the usage of coordinate adaptors could occasionally result in crash, for example when function `FixAtoms()` was followed by `SetRemoveCoulombTerms()`. This bug was fixed.
- Running in the `pvm` mode requires now setting the `PVM_PATH` variable, which points at the directory where the Szybki script is located. For example if Szybki was untared in the directory `mydir`, then using `bash` shell one can set the above variable with: `export PVM_PATH=mydir/openeye/bin`

## 5.7 SZYBKI 1.2.2

### 5.7.1 New features

- Two new functions are added to the API:
 

```
void SetResidueFlexibility(double dist); double GetResidueFlexibility() const;
```

 These allow the user to introduce and query for the flexibility of entire residues around the ligand.

## 5.7.2 Bug fixes

- In the case when protein electrostatic model was set at `OEProteinElectrostatics::NoElectrostatics` and partial protein flexibility was allowed with the `void SetSideChainsFlexibility(double)` or `void SetPolarHFlexibility(double)` functions, Coulombic terms describing the interaction between the ligand and flexible protein atoms were included during optimization. This was inconsistent with the meaning of `NoElectrostatics`, therefore those Coulombic terms were eliminated in the current release
- Using the function `void SetSideChainsFlexibility(double)` resulted occasionally in double counting of some residues. This bug was fixed.
- Fixing atoms with a function `bool FixAtoms(std::string)` where the passed string is SMARTS pattern, was based on MMFF aromaticity types. Starting from the current release OE aromaticity model is used.

# BIBLIOGRAPHY

- [Geist-1994] Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Manchek, R. and Sunderam, V., **PVM - Parallel Virtual Machine: A User's Guide and Tutorial for Networked Parallel Computing**; *MIT Press*, 1994
- [Halgren-1996-1] T.A. Halgren, **Merck Molecular Force Field: I. Basis, Form, Scope, Parameterization and Performance of MMFF94**, *Journal of Computational Chemistry*, Vol. 17, No. 5, pp. 490-519, 1996
- [Halgren-1996-2] T.A. Halgren, **Merck Molecular Force Field: II. MMFF94 van der Waals and Electrostatic Parameters for Intermolecular Interactions**, *Journal of Computational Chemistry*, Vol. 17, No. 5, pp. 520-552, 1996
- [Halgren-1996-3] T.A. Halgren, **Merck Molecular Force Field: III. Molecular Geometries and Vibrational Frequencies**, *Journal of Computational Chemistry*, Vol. 17, No. 5, pp. 553-586, 1996
- [Halgren-1996-4] T.A. Halgren, **Merck Molecular Force Field: IV. Conformational Energies and Geometries for MMFF94**, *Journal of Computational Chemistry*, Vol. 17, No. 5, pp. 587-615, 1996
- [Halgren-1996-5] T.A. Halgren, **Merck Molecular Force Field: V. Extension of MMFF94 using Experimental Data, Additional Computational Data and Empirical Rules**, *Journal of Computational Chemistry*, Vol. 17, No. 5, pp. 616-641, 1996
- [Halgren-1999-1] T.A. Halgren, **MMFF VI. MMFF94s Option for Energy Minimization Studies**, *Journal of Computational Chemistry*, Vol. 20, No. 5, pp. 720-729, 1999
- [Halgren-1999-2] T.A. Halgren, **MMFF VII. Characterization of MMFF94, MMFF94s and Other Widely Available Force Fields for Conformational Energies and for Intermolecular Interaction Energies and Geometries**, *Journal of Computational Chemistry*, Vol. 20, No. 5, pp. 730-748, 1999
- [Jakalian-2002] A. Jakalian, D.B. Jack and C.I. Bayly, **Fast, efficient generation of high-quality atomic charges. AM1-BCC model: II. Parameterization and validation**, *J. Comput. Chem.*, Vol. 23, pp. 1623-1641, 2002
- [Grant-2007] J.A. Grant, B.T. Pickup, M.J. Sykes, C.A. Kitchen and A. Nicholls, **A Simple Formula for Dielectric Polarization Energies: The Sheffield Solvation Model**, *Chem. Phys. Letters*, Vol. 441, pp. 163-166, 2007
- [Wlodek-2010] S. Wlodek, A.G. Skillman and A. Nicholls, **Ligand entropy in gas-phase, upon solvation and protein complexation. Fast estimation with Quasi-Newton Hessian**, *J. Chem. Theory Comput.*, accepted for publication, 2010



# INDEX

## Symbols

- MMFF94s
  - szybki command line option, 14
- am1bcc
  - szybki command line option, 14
- complex <filename>
  - szybki command line option, 11
- conj
  - szybki command line option, 15
- entropy <type>
  - szybki command line option, 17
- exact\_vdw
  - szybki command line option, 14
- fix\_file <filename>
  - szybki command line option, 11
- fix\_smarts <p>
  - szybki command line option, 15
- grad\_conv <c>
  - szybki command line option, 16
- harm\_constr1 k
  - szybki command line option, 14
- harm\_constr2 d
  - szybki command line option, 14
- harm\_smarts p
  - szybki command line option, 14
- heavy\_rms
  - szybki command line option, 11
- largest\_part
  - szybki command line option, 16
- ligands <filename>
  - szybki command line option, 11
- loadPG <filename>
  - szybki command line option, 11
- log <pre>
  - szybki command line option, 12
- max\_iter <m>
  - szybki command line option, 16
- mod\_vdw
  - szybki command line option, 14
- mol2charges
  - szybki command line option, 14
- neglect\_frozen
  - szybki command line option, 14
- noCoulomb
  - szybki command line option, 15
- no\_opt
  - szybki command line option, 16
- opt\_cart
  - szybki command line option, 16
- opt\_solid
  - szybki command line option, 16
- opt\_torsions
  - szybki command line option, 16
- out <filename>
  - szybki command line option, 12
- out\_protein <filename>
  - szybki command line option, 12
- param
  - szybki command line option, 12
- polarH <r>
  - szybki command line option, 16
- prefix <pn>
  - szybki command line option, 12
- protein <filename>
  - szybki command line option, 12
- protein\_dielectric <d>
  - szybki command line option, 15
- protein\_elec <m>
  - szybki command line option, 15
- protein\_vdw <r>
  - szybki command line option, 15
- pvmconf <f.txt>
  - szybki command line option, 17
- report
  - szybki command line option, 12
- reportFile <filename>
  - szybki command line option, 12
- residue <r>
  - szybki command line option, 16
- savePG <filename>
  - szybki command line option, 12
- sdtag <string>
  - szybki command line option, 12

-shefA <a>  
     szybki command line option, 15

-shefB <b>  
     szybki command line option, 15

-sheffield  
     szybki command line option, 15

-sideC <r>  
     szybki command line option, 16

-silent  
     szybki command line option, 14

-solv\_dielectric <d>  
     szybki command line option, 15

-solventCA <s>  
     szybki command line option, 15

-solventPB  
     szybki command line option, 15

-strip\_water  
     szybki command line option, 16

-verbose  
     szybki command line option, 14

## A

APPNAME\_OE\_ARCH, 4

## E

environment variable

    APPNAME\_OE\_ARCH, 4  
     OE\_ARCH, 4  
     OE\_LICENSE, 3  
     PATH, 4–6  
     PVM\_ARCH, 5  
     PVM\_PATH, 5  
     PVM\_ROOT, 5

## O

OE\_ARCH, 4

OE\_LICENSE, 3

## P

PATH, 4–6

PVM\_ARCH, 5

PVM\_PATH, 5

PVM\_ROOT, 5

## S

szybki command line option

    -MMFF94s, 14  
     -am1bcc, 14  
     -complex <filename>, 11  
     -conj, 15  
     -entropy <type>, 17  
     -exact\_vdw, 14  
     -fix\_file <filename>, 11  
     -fix\_smarts <p>, 15

    -grad\_conv <c>, 16  
     -harm\_constr1 k, 14  
     -harm\_constr2 d, 14  
     -harm\_smarts p, 14  
     -heavy\_rms, 11  
     -largest\_part, 16  
     -ligands <filename>, 11  
     -loadPG <filename>, 11  
     -log <pre>, 12  
     -max\_iter <m>, 16  
     -mod\_vdw, 14  
     -mol2charges, 14  
     -neglect\_frozen, 14  
     -noCoulomb, 15  
     -no\_opt, 16  
     -opt\_cart, 16  
     -opt\_solid, 16  
     -opt\_torsions, 16  
     -out <filename>, 12  
     -out\_protein <filename>, 12  
     -param, 12  
     -polarH <r>, 16  
     -prefix <pn>, 12  
     -protein <filename>, 12  
     -protein\_dielectric <d>, 15  
     -protein\_elec <m>, 15  
     -protein\_vdw <r>, 15  
     -pvmconf <f.txt>, 17  
     -report, 12  
     -reportFile <filename>, 12  
     -residue <r>, 16  
     -savePG <filename>, 12  
     -sdtag <string>, 12  
     -shefA <a>, 15  
     -shefB <b>, 15  
     -sheffield, 15  
     -sideC <r>, 16  
     -silent, 14  
     -solv\_dielectric <d>, 15  
     -solventCA <s>, 15  
     -solventPB, 15  
     -strip\_water, 16  
     -verbose, 14