LigPrep 3.4

User Manual

Schrödinger Press

May 2015

Contents

Document Conventions

In addition to the use of italics for names of documents, the font conventions that are used in this document are summarized in the table below.

Font	Example	Use
Sans serif	Project Table	Names of GUI features, such as panels, menus, menu items, buttons, and labels
Monospace	`$SCHRODINGER/maestro`	File names, directory names, commands, environment variables, command input and output
Italic	*filename*	Text that the user must replace with a value
Sans serif uppercase	CTRL+H	Keyboard keys

Links to other locations in the current document or to other PDF documents are colored like this: Document Conventions.

In descriptions of command syntax, the following UNIX conventions are used: braces { } enclose a choice of required items, square brackets [ ] enclose optional items, and the bar symbol | separates items in a list from which one item must be chosen. Lines of command syntax that wrap should be interpreted as a single command.

File name, path, and environment variable syntax is generally given with the UNIX conventions. To obtain the Windows conventions, replace the forward slash / with the backslash \ in path or directory names, and replace the $ at the beginning of an environment variable with a % at each end. For example, `$SCHRODINGER/maestro` becomes `%SCHRODINGER%\maestro`.

Keyboard references are given in the Windows convention by default, with Mac equivalents in parentheses, for example CTRL+H (⌘H). Where Mac equivalents are not given, COMMAND should be read in place of CTRL. The convention CTRL-H is not used.

In this document, to *type* text means to type the required text in the specified location, and to *enter* text means to type the required text, then press the ENTER key.

References to literature sources are given in square brackets, like this: [10].

Introduction

LigPrep is a robust collection of tools designed to prepare high quality, all-atom 3D structures for large numbers of drug-like molecules, starting with 2D or 3D structures in SD or Maestro format. The resulting structures can be saved in either SD or Maestro format. The simplest use of LigPrep produces a single, low-energy, 3D structure with correct chiralities for each successfully processed input structure. LigPrep can also produce a number of structures from each input structure with various ionization states, tautomers, stereochemistries, and ring conformations, and eliminate molecules using various criteria including molecular weight or specified numbers and types of functional groups present.

LigPrep can be run from Maestro on UNIX or Windows platforms, or from the command line on UNIX platforms. The `ligprep` script provides an efficient way to use a set of tools for ligand preparation collectively and consistently. Some of the tools may also be used separately. The ligand preparation process typically takes about one second per output structure on a 2 GHz x86-compatible computer. Depending on the nature of the input structures and on the options selected, multiple output structures may be produced from each input structure. LigPrep calculations can be distributed across multiple processors to increase throughput.

1.1 The LigPrep Manual

This manual explains how to use LigPrep, a program suite designed to prepare high quality, all-atom 3D structures for large numbers of drug-like molecules, starting with 2D or 3D structures. The manual is organized as follows:

- Chapter 1 provides an overview of LigPrep and the process it uses to prepare ligands.
- Chapter 2 explains how to run LigPrep from Maestro and from the command line.
- Chapter 3 provides instruction on the use of LigPrep in several scenarios.
- Chapter 4 describes the tools used by LigPrep.
- Chapter 5 describes the `ionizer` and its options, input, and output.
- Chapter 6 describes the `tautomerizer` and how it generates tautomers in detail.

1.2 The LigPrep Process

The LigPrep process consists of a series of steps that perform conversions, apply corrections to the structures, generate variations on the structures, eliminate unwanted structures, and optimize the structures. Many of the steps are optional and are controlled by selecting options in the LigPrep panel or by specifying command-line options. The steps are outlined below. Each step is performed by the script or program mentioned in the step.

While the generation of ionization states and tautomers (Step 6 and Step 7) can be carried out with tools that are part of LigPrep, a separate product, Epik, can be used instead to perform these structural adjustments during a LigPrep run. Epik more rigorously adjusts the tautomerization and ionization states than separate ionizer and tautomerizer treatments. However, it is slower, taking on average around 6 seconds per structure on a 2 GHz Pentium IV processor.

1. Convert the structure format.

   If the input structure file is in SD format, it is converted to Maestro format by `sdconvert`. Parities specified in the SD file are converted into chiralities, which are stored as properties in the Maestro file. If the input structure file is in SMILES format, it is converted to Maestro format by `smiles_to_mae`. The input file should have one SMILES pattern per line. The SMILES pattern can be followed by an optional title string.

2. Select the structures.

   A subset of the input structures can be selected for processing. The selection is done by `maesubset` for Maestro input files and by `sdconvert` for SD input files.

3. Add hydrogen atoms.

   Structures that have implicit hydrogen atoms may need to have hydrogen atoms added before the 3D structures can be minimized. Hydrogen atoms are added in a manner that is consistent with a particular force field. This step is performed by `applyhtreat`, the program used by the Hydrogen Treatment panel in Maestro.

4. Remove unwanted molecules.[1]

   If structures have additional molecules included, such as counter ions in salts and water molecules, these may need to be removed. The `desalter` removes all but the molecule containing the most atoms from each structure.

5. Neutralize charged groups.[1]

   Charged groups must be neutralized before ionization states can be generated. The neutralization is performed by the `neutralizer`, which adds or removes hydrogen ions.

1. Executed by default, but optional in the LigPrep process.

6. Generate ionization states.[2]

   For some applications, it is important that all species that exist in a given pH range are available. In this step, the `ionizer` generates various ionization states for each structure. This step should be preceded by a neutralization step.

7. Generate tautomers.[1]

   As with ionization, the tautomeric states with a significant population may be important for some types of calculations. The `tautomerizer` generates various tautomers for each structure.

8. Filter the structures.[1]

   In this step, structures that match specified conditions can be removed. The condition can be on a property, such as Molecular weight > 1000, or on the structure, such as the presence or absence of a specific functional group. This step is performed by `ligfilter`.

9. Specify chiralities.[1]

   Input structures given in 1D or 2D often have indicated chiralities, and 3D structures many need the chiralities inherent in the geometry retained. Unspecified chiralities must of course be assigned. This step identifies all chiral atoms in the structures and provides options to retain chiralities or generate additional stereochemical isomers with the same molecular formula. The step is performed by `stereoizer`.

10. Generate low-energy ring conformations.[1]

    When ring conformation information is not available, it can be important to specifically generate low-energy ring conformations. Ring confirmations are generated for each structure by `ring_conf`.

11. Remove problematic structures.

    Structures that could cause subsequent processing failures either in the energy minimization of the structures or in other applications are removed by `premin`.

12. Optimize the geometries.

    The geometries of the generated structures are optimized using a restricted version of the MacroModel™ computational program, `bmin`, or a short conformational search is performed to relax the structure into 3 dimensions while strongly encouraging chiral centers to adopt the proper chiralities (if the structure is highly strained).

13. Convert the output file.

    If output in SD format was requested, `sdconvert` is run to perform the conversion.

2. Not executed by default, but can be enabled in the LigPrep process by setting the appropriate options.

1.3 Running Schrödinger Software

Schrödinger applications can be run from a graphical interface or from the command line. The software writes input and output files to a directory (folder) which is termed the *working directory*. If you run applications from the command line, the directory from which you run the application is the working directory for the job.

Linux:

To run any Schrödinger program on a Linux platform, or start a Schrödinger job on a remote host from a Linux platform, you must first set the `SCHRODINGER` environment variable to the installation directory for your Schrödinger software. To set this variable, enter the following command at a shell prompt:

csh/tcsh: `setenv SCHRODINGER` *installation-directory*

bash/ksh: `export SCHRODINGER=`*installation-directory*

Once you have set the `SCHRODINGER` environment variable, you can run programs and utilities with the following commands:

`$SCHRODINGER/`*program* `&`
`$SCHRODINGER/utilities/`*utility* `&`

You can start the Maestro interface with the following command:

```
$SCHRODINGER/maestro &
```

It is usually a good idea to change to the desired working directory before starting the Maestro interface. This directory then becomes the working directory.

Windows:

The primary way of running Schrödinger applications on a Windows platform is from a graphical interface. To start the Maestro interface, double-click on the Maestro icon, on a Maestro project, or on a structure file; or choose Start → All Programs → Schrodinger-2015-2 → Maestro. You do not need to make any settings before starting Maestro or running programs. The default working directory is the Schrodinger folder in your Documents folder.

If you want to run applications from the command line, you can do so in one of the shells that are provided with the installation and have the Schrödinger environment set up:

- Schrödinger Command Prompt—DOS shell.
- Schrödinger Power Shell—Windows Power Shell (if available).

You can open these shells from Start → All Programs → Schrodinger-2015-2. You do not need to include the path to a program or utility when you type the command to run it. If you want access to Unix-style utilities (such as `awk`, `grep`, and `sed`), preface the commands with `sh`, or type `sh` in either of these shells to start a Unix-style shell.

Mac:

The primary way of running Schrödinger software on a Mac is from a graphical interface. To start the Maestro interface, click its icon on the dock. If there is no Maestro icon on the dock, you can put one there by dragging it from the SchrodingerSuite2015-2 folder in your Applications folder. This folder contains icons for all the available interfaces. The default working directory is the Schrodinger folder in your Documents folder (`$HOME/Documents/Schrodinger`).

Running software from the command line is similar to Linux—open a terminal window and run the program. You can also start Maestro from the command line in the same way as on Linux. The default working directory is then the directory from which you start Maestro. You do not need to set the `SCHRODINGER` environment variable, as this is set in your default environment on installation. To set other variables, on OS X 10.7 use the command

`defaults write ~/.MacOSX/environment` *variable* `"`*value*`"`

and on OS X 10.8, 10.9, and 10.10 use the command

`launchctl setenv` *variable* `"`*value*`"`

1.4 Starting Jobs from the Maestro Interface

To run a job from the Maestro interface, you open a panel from one of the menus (e.g. Tasks), make settings, and then submit the job to a host or a queueing system for execution. The panel settings are described in the help topics and in the user manuals. When you have finished making settings, you can use the Job toolbar to start the job.

You can start a job immediately by clicking Run. The job is run on the currently selected host with the current job settings and the job name in the Job name text box. If you want to change the job name, you can edit it in the text box before starting the job. Details of the job settings are reported in the status bar, which is below the Job toolbar.

If you want to change the job settings, such as the host on which to run the job and the number of processors to use, click the Settings button. (You can also click the arrow next to the button and choose Job Settings from the menu that is displayed.)

You can then make the settings in the Job Settings dialog box, and choose to just save the settings by clicking OK, or save the settings and start the job by clicking Run. These settings apply only to jobs that are started from the current panel.

If you want to save the input files for the job but not run it, click the Settings button and choose Write. A dialog box opens in which you can provide the job name, which is used to name the files. The files are written to the current working directory.

The Settings button also allows you to change the panel settings. You can choose Read, to read settings from an input file for the job and apply them to the panel, or you can choose Reset Panel to reset all the panel settings to their default values.

You can also set preferences for all jobs and how the interface interacts with the job at various stages. This is done in the Preferences panel, which you can open at the Jobs section by choosing Preferences from the Settings button menu.

Note: The items present on the Settings menu can vary with the application. The descriptions above cover all of the items.

The icon on the Job Status button shows the status of jobs for the application that belong to the current project. It starts spinning when the first job is successfully launched, and stops spinning when the last job finishes. It changes to an exclamation point if a job is not launched successfully.

Clicking the button shows a small job status window that lists the job name and status for all active jobs submitted for the application from the current project, and a summary message at the bottom. The rows are colored according to the status: yellow for submitted, green for launched, running, or finished, red for incorporated, died, or killed. You can double-click on a row to open the Monitor panel and monitor the job, or click the Monitor button to open the Monitor panel and close the job status window. The job status is updated while the window is open. If a job finishes while the window is open, the job remains displayed but with the new status. Click anywhere outside the window to close it.

Jobs are run under the Job Control facility, which manages the details of starting the job, transferring files, checking on status, and so on. For more information about this facility and how it operates, as well as details of the Job Settings dialog box, see the *Job Control Guide*.

1.5 Other Utilities

A number of utilities that are provided with LigPrep, but are not part of LigPrep itself, might be useful in conjunction with LigPrep. These utilities are available in `$SCHRODINGER/utilities`, and include:

- `proplister`—extracts requested properties from Maestro-formatted files
- `ligfilter`—selects structures from Maestro-formatted files using the properties stored in the files (supersedes `propfilter`)
- `structconvert`—converts files between various formats
- `mol2convert`—converts files between Sybyl Mol2 and other formats
- `pdbconvert`—converts files between Maestro and PDB formats
- `maesubset`—selects a subset of the structures in a Maestro-formatted file
- `sdsubset`—selects a subset of the structures in an SD-formatted file

More information on these utilities is available in the *General Utilities* manual.

1.6 Citing LigPrep in Publications

The use of this product should be acknowledged in publications as:

LigPrep, version 3.4, Schrödinger, LLC, New York, NY, 2015.

Running LigPrep

The main objective of LigPrep is to take 2D or 3D structures and produce corresponding low-energy 3D structures for use in further computational studies. The input and output files can be in SD or Maestro format. LigPrep can also produce multiple output structures for each input structure, a process referred to as *expansion*, by generating variations on the ionization state, tautomers, stereochemistry, and ring conformations.

2.1 Running LigPrep From Maestro

You can set up a LigPrep job from the LigPrep panel in Maestro. To open the LigPrep panel, choose Applications → LigPrep or Tasks → Ligand Preparation in the main window.

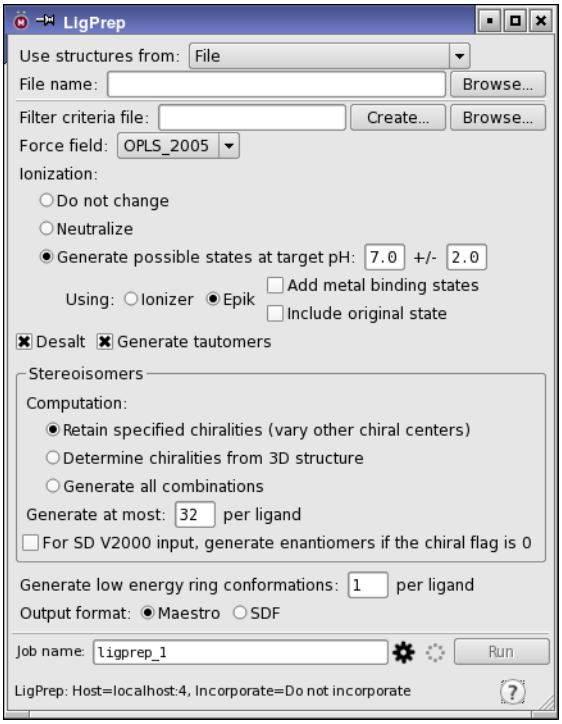

Figure 2.1. The LigPrep panel.

2.1.1 Selecting Structure Input and Output

The upper portion of the LigPrep panel contains settings that control the input structures.

The input for LigPrep is specified with the Use structures from option menu. It can come from the Workspace, selected entries in the Project Table, or from a file. If you select Workspace (included entries) or Project Table (selected entries) as the source of job input, the structures are written to a file in Maestro format, named *jobname*`.mae`. If you select File, you can type a file name in the File name text box, or click Browse to navigate to the file and select it. The file can be in Maestro or SD format, compressed or uncompressed, or it can be in one of two SMILES formats:

- `.smi` file—Text file with one SMILES string and an optional title per line. The title is separated from the SMILES string by a space
- `.csv` file—Comma-separated file with the SMILES string as the first field, title as the second, followed by optional properties

If you decide to use a file for your job input, the results are written to a file in the format you specify under Output format (in the lower left of the panel). The file name is *jobname*`.`*ext*, where *ext* is `maegz` for Maestro format, or `sdf` for SDF format. You can only incorporate results into the project if the format is Maestro.

2.1.2 Setting Filter Criteria

The structures that are produced can be filtered on the basis of various predefined descriptors, using `ligfilter`. These descriptors are counts of various kinds of structural components, including a range of functional groups. You can filter the structures generated by LigPrep using a file that defines conditions on these descriptors.

This file can be created by clicking Create, and constructing the list of criteria in the Ligand Filtering panel. You can also read an existing file by clicking Browse, and navigating to the file in the file selector that opens. The following instructions describe how to create a filter in the Ligand Filtering panel.

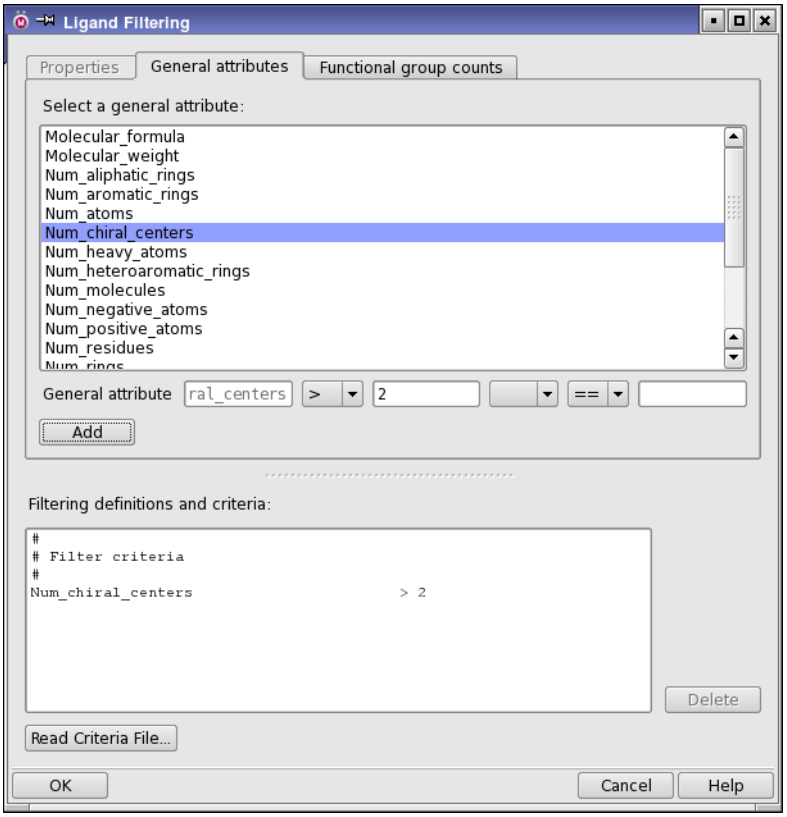

Figure 2.2. The Ligand Filtering panel.

To create a simple filter criterion:

1. Select a descriptor from the Select a general attribute list in the General Attributes tab.

   The descriptor is added to the General attribute text box.

2. Choose a relational operator from the first option menu to the right of the General attribute text box.

3. Enter a value in the text box immediately to the right.

4. Click Add.

   The new criterion is added to the Filtering conditions and criteria text area.

   You can also add criteria in the Functional Group Counts tab in the same way.

To add a second condition, do the following before clicking Add:

1. Choose a Boolean operator from the option menu in the middle.
2. Choose a relational operator from the next option menu.
3. Enter a value in the text box immediately to the right.

Adding the second condition allows you to specify a range of values. For example, the following criterion excludes ligands with molecular weights that are not in the range 150–500:

```
Molecular weight              < 150 OR > 500
```

If you add a criterion when one already exists for a particular attribute, the existing criterion is overwritten with the new one. You can delete individual criteria by selecting them in the list and clicking Delete. The Delete All button clears the entire list.

When you have finished creating the list of criteria, click OK. A temporary file is created for the job and the file name is displayed in the Filter criteria file text box in the LigPrep panel.

Existing filter criteria file can be read (replacing the existing list contents) by clicking Read and navigating to the desired file. You then can modify the criteria list and write out a new file.

An example filter criteria file is given below. Comment lines start with #. The files generated from the panel only have a generic comment at the top, but you can add comments if you want.

```
# Remove molecules that have a molecular weight of greater than 650
Molecular_weight                        > 650
# Remove molecules with too many H-bond acceptor and donor atoms
Acceptor_groups                         > 3
Donor_groups                            > 3
# Remove molecules with fewer than 10 atoms
Num_atoms                               < 10
```

If you want to use other descriptors than those listed above, you can create a filter criteria file from the Ligand Filtering panel in Maestro, which you open from the Tools menu. You can then import this file by clicking Browse in the LigPrep panel to select a filter criteria file. For more information on the Ligand Filtering panel, see Section 10.8 of the *Maestro User Manual*.

For more information on ligfilter, see Section 4.9 on page 47.

2.1.3 Selecting a Force Field

You can choose a force field from the Force field option menu. This force field is used in the MacroModel (`bmin`) step. The choices are OPLS_2005 and MMFFs.

2.1.4 Setting Ionization State Options

The option you select controls the generation of ionization states of the ligand. The ionization states are generated by adding or removing protons from the ligand. Some programs require neutral molecules as input. For these programs, you should select Neutralize. For other programs, such as Glide, it can be useful to generate a range of ionization states that are populated in a given pH range. The Generate possible states at target pH option selects the generation of ionization states that are significantly populated in the specified pH range. If you do not want the ionization states to be altered, choose Do not change. However, even if you select Do not change, deprotonated acids can still be protonated by `applyhtreat` if the formal charges in the acid are incorrect.

To perform the ionization, you can select either the Ionizer or Epik. Epik is a separate product, and must be purchased and installed if you want to use it with LigPrep. The Ionizer is included with LigPrep.

Epik can generate ionization and tautomeric states for molecules in aqueous solution (the default). If the input structure is not probable in the given pH range, it is not included in the output structures by default. If you want to keep it in the output, select Include original state.

Epik can also generate states for molecules bound to metals in metalloproteins. Extra structures are generated and an alternate ranking scheme is used for states of molecules that are bound to metals in a protein. The option to account for metal binding is set by selecting Add metal binding states. These states can be used by Glide when docking ligands to metalloproteins, for example.

2.1.5 Removing Cofactors

Structures from some databases can consist of multiple molecules, one of which is the ligand. In many cases, the extra molecules are water molecules or counter ions. To remove these molecules, select the Desalt option. LigPrep removes all but the largest molecule when this option is selected. In most cases, this procedure removes the correct molecules. If the desired molecule is not the largest, you may have to edit the structure in Maestro before passing it to LigPrep.

2.1.6 Generating Tautomers

For some ligand molecules, a tautomer of the commonly-given structure binds to the active site. To generate up to 8 tautomers per input structure, select Generate Tautomers. LigPrep performs keto-enol tautomerization and the analogous sulfur and nitrogen tautomerizations, and histidine and DNA base tautomerizations. For more information, see Chapter 6.

2.1.7 Setting Stereoisomer Options

The option you select affects the treatment of stereochemical information in the input and the generation of stereoisomers in the output.

- Retain specified chiralities keeps the specified chirality information from the input file and fixes these chiralities for the entire calculation. Chirality information includes parities and bond directions from SD files and the chirality and atom numbering chirality properties from Maestro files. (Maestro chiralities are written only by `sdconvert` and the `stereoizer`.) If the configuration or chirality of a chiral center is not specified, the two possible chiralities are generated internally. This is the default behavior.
- Determine chiralities from 3D structure discards all specified chiralities from the input file and determines the chirality from the 3D geometry (`-g` option of the `ligprep` command). These chiralities are held fixed. For centers whose chirality is indeterminate, the two possible chiralities are generated internally.
- Generate all combinations discards all chirality information from chirality properties in the input file and from the 3d geometry, and internally generates all possible configurations that result from the combination of chiralities on each chiral center (`-ac` option of the `ligprep` command).

For any of these settings, the internally generated stereoisomers are subjected to a filtering step to eliminate some structures that either violate geometric restrictions for fused ring systems or conflict with natural product chiralities (e.g. the pattern of chiralities in steroid frameworks).

You can limit the number of stereoisomers generated for the atoms whose chiralities are not assigned by entering a limit in the Generate at most *N* per ligand text box.

SD files in V2000 format have a field, the “chiral flag”, that determines whether the input structure represents a pair of enantiomers (0) or a stereoisomer (1). If you a want to generate both enantiomers, you can select For SD V2000 input, generate enantiomers if the chiral flag is 0. The default is to generate only the enantiomer represented by the specified chiralities. Chiral centers for which the chirality is not specified are varied, as usual. The setting is stored as a preference, so it is preserved across Maestro sessions.

2.1.8 Generating Ring Conformations

From a 2D structure, it is not immediately obvious which ring conformations give the lowest energy or are preferred for binding to an active site. LigPrep can generate ring conformations and evaluate an approximate conformational energy to determine which conformations are likely to be the lowest in energy. The value in the Generate low energy ring conformations text box determines how many conformations are kept, counting from the lowest in energy.

2.1.9 Reading and Writing Settings

You can save the settings in the panel, including the choice of input and output, by writing them to a LigPrep input file. This file can be used as an alternative to command-line options when running a LigPrep job from the command line. It can also be read in to the LigPrep panel, to set all the options a particular way. This feature is useful if you run LigPrep with nonstandard settings (although these are remembered by the panel), particularly if you use different settings in different circumstances. All the settings for the particular circumstance can be made by reading the appropriate input file. See Section 2.4 on page 18 for more information on the input file and its format.

2.1.10 Submitting the Job

Once you have selected the structural variation options, click Run to run the job with the current job settings. If you want to make settings for the job, such as selecting a host, choose Job Settings from the Settings button to open the Job Settings dialog box (see Figure 2.3) This dialog box is a common interface for submitting jobs from Maestro. For details of its features, see Section 2.2 of the *Job Control Guide*. See Section 1.4 on page 5 for more information on starting jobs from Maestro.

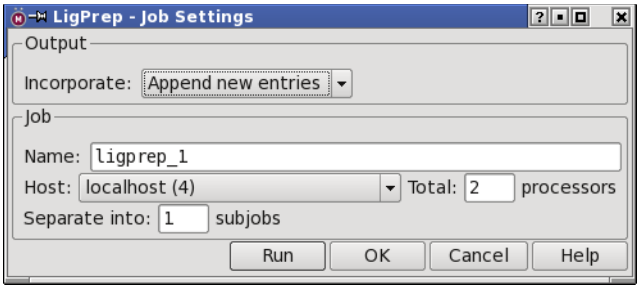

Figure 2.3. The LigPrep - Job Settings dialog box.

- If your job input came from a file, the results are written to a file in the format you specified under Output format (see Section 2.1.1 on page 10). The file name is *jobname*.*ext*, where *ext* is `maegz` for Maestro format or `sdf` for SD format.
- If you want to run the job on a remote host, choose a host from the Host list. You can distribute the job across multiple processors by selecting a multiprocessor host and entering the number of CPUs in the Use column. For more information on hosts, host settings, and configuring this file, see Chapter 7 of the *Installation Guide*.

When you have finished choosing settings, click Run to save the settings and start the job. The LigPrep job is submitted to the selected host. Output for each step of the LigPrep process is stored in a temporary directory. The log file for the job is stored in the job submission directory

(the Maestro current working directory). This file is updated as the job progresses and is displayed in the Monitor panel, which you can use to monitor the job. For more information on job monitoring, see the *Job Control Guide*.

If you want to run the job from the command line, you can choose Write from the Settings button menu to write out an input file (`.inp`), and run `ligprep` with the `-inp` option to specify this file as the source of options.

2.2 The ligprep Command

To run LigPrep from the command line, use the `ligprep` command. The syntax of the `ligprep` command is as follows.

```
ligprep [options ] [ meta-options]
    {-imae|-isd|-ismi|-icsv} input-file {-omae|-osd} output-file
```

You must specify an input structure file using either `-imae` for input files in Maestro format, `-isd` for input files in SD file format, or `-ismi` for input files in SMILES format as a list of SMILES strings (`.smi`) or `-icsv` for input files in CSV format containing a SMILES string with optional title and properties on each line, separated by commas (`.csv`). The output file is specified with `-omae` for Maestro format or `-osd` for SD format. Maestro input and output files can be in uncompressed (`.mae`) or compressed (`.maegz` or `.mae.gz`) form.

The `ligprep` command has a number of options for controlling the stages of the process, which can be displayed by entering the command `ligprep -h`. The standard Job Control options, for selecting the host, defining a scratch directory, and so on, are supported, as are the common `-LOCAL` and `-WAIT` options—see Section 2.3 of the *Job Control Guide* for a complete listing. In particular, you should note the syntax of the `-HOST` option, which is used to specify the list of hosts used for the job. In addition to these options, the `ligprep` command supports a range of other job options for both the control of the job and for distributing jobs over multiple processors.

The `ligprep` command also has a number of *meta-options*, which set a group of options with particular values. You can use options together with meta-options; the options override any values set by the meta-options.

The options can also be specified in an input file that contains keyword-value pairs to set the options—see Section 2.4 on page 18 for details.

The `adjust`, `vary` and `expand` meta-options control the range of structures generated. You can use all three of these meta-options in the same `ligprep` command, but `c`, `i` and `t` must not appear more than once. For instance, you could use `-adjust_t -vary_c –expand_i`, but not `-adjust_t -vary_c –expand_it`.

The retain meta-option can be used to require that at least one structure produced for each input structure retains the characteristics denoted by the suffix. For instance, -retain_t means that the tautomerization state in the input structure will be present in at least one of the output structures for that molecule. The retain meta-option can be used with any of the adjust, vary and expand meta-options.

LigPrep jobs run in the background under Schrödinger's Job Control facility. The job output is normally written to a temporary directory and then copied back to the job submission directory when the job finishes. You can direct the temporary files and the output to the job submission directory with the -LOCAL option of the ligprep command. The .log file is written to the job submission directory and is updated during the run. For more information on the Job Control facility, see the *Job Control Guide*.

2.3 Distributing LigPrep Jobs

You can significantly reduce the amount of time taken by LigPrep to process a large number of ligands by distributing the job across multiple processors. The ligprep command has job options that facilitate this task. You can also use ligprep to split very large input structure files into smaller files that can be processed separately without exceeding the 2 GB file size limit that is in effect on some hosts (see page 25). The number of subjobs may be greater than or equal to the number of processors requested, so that processors run more than one subjob before the job finishes.

Often there is a distinct advantage in setting the number of subjobs to a small multiple (e.g. greater than 3) of the number of processors requested, particularly when the processors differ in inherent speed and load (number of programs contending for CPU time). When a processor finishes a task, it is assigned a new one from the pool of unprocessed tasks until the overall calculation is complete. As a result, processors that return results faster are assigned a greater portion of the overall calculation, which in turn is completed sooner.

For instance, if the input structure file many.sd contains 100,000 structures, then the command:

```
ligprep -NJOBS 20 -SUBHOST big_host:5 -isd many.sd -osd many-out.sd
```

produces 20 subdirectories, each containing a separate 5,000-structure portion of the input file. Up to five subjobs at a time run on the computer big_host until all 20 subjobs are completed.

The ligprep command also supports restarting of subjobs and jobs. If a subjob fails, it is restarted up to the number of times given by -MAX_RETRIES. If there are still failed subjobs when the job finishes, or if the master job fails, you can restart it with the -RESTART option, provided you ran the job with -LOCAL. Each failed subjob is run again; completed subjobs are not rerun.

The -j option is useful for preparing and running a single subjob or a selection of subjobs. If the -nx option is given, the subjob files are prepared but not run. The -j option allows you to rerun subjobs for a failed job that was not run with -LOCAL.

ligprep splits an input ligand file into *njobs* smaller input files, each of which is automatically processed in a separate ligprep calculation. The ligands between *firstlig* and *lastlig* (inclusive) are separated into *njobs* equal-sized batches. ligprep starts up a parent process on the first computer listed after -HOST (or the local host if -HOST is not used). This parent process runs the subjobs as child processes and organizes the output when all subjobs are completed.

The jobname for ligprep is the stem of the output structure file name, formed by removing any .mae, .sd, or .sdf extension. By default, all output structures are placed in the output structure file listed on the command line. If the total size of the output structures would exceed 2 GB, it is better to retain a separate output structure file for each subjob, named *jobname*_subjob_*subjobnum*.mae, by using the -OUTPUT_ORG BY_SUBJOB option.

Behind the scenes, distributed ligprep jobs can involve many files in a number of remote directories (one for each subjob), which you usually do not need to examine. However, sometimes you may wish to examine these files. All these files are saved in a single gzipped tar file called *jobname*.tar.gz. The job files can be extracted using the commands:

```
gunzip jobname.tar.gz
tar -xvf jobname.tar
```

If some of the subjobs failed, for example, you can extract the output and rerun the subjobs, starting them at the stage that failed with the -run_from option. The failed subjobs are noted in the main log file.

2.4 LigPrep Input File

As an alternative to specifying options on the command line, you can set the options in a LigPrep input file. This file is optional, and can be used with the following syntax:

ligprep -inp *filename*

where *filename* is the name of the input file. You can specify options on the command line in addition to the input file. These options take precedence over the settings from the input file.

The input file uses a keyword-value syntax. The keywords are listed in Table 2.1, along with the type of value and the equivalent command-line option. The values accepted by the keyword are the same as for the equivalent option. Options that do not have a corresponding value are set as Booleans in the input file. Boolean values must be specified as True or False.

Table 2.1. LigPrep input file keywords

Keyword	Value Type	Equivalent command line option
Input and output		
INP_MAE	string	-imae
OUT_MAE	string	-omae
INP_SD	string	-isd
OUT_SD	string	-osd
INP_SMI	string	-ismi
General		
APPEND_STRUCTURES	boolean	-a
KEEP_PROPS	boolean	-kp, -keep_props
MAX_ATOMS	integer	-ma
STRUCTS_TO_CONVERT	integer	-n
NO_CLEANUP	boolean	-nc
REMOVE_PROBLEMS	boolean	-np
RUN_ONLY_ONE_STAGE	string	-R
VERBOSE	boolean	-verb
OPTION_TO_PASS	string	-W
Desalting		
NO_DESALTER	boolean	-nd
USE_DESALTER	boolean	-d
Ionization		
EPIK	boolean	-epik
EPIK_FILE	string	-es
EPIK_METAL_BINDING	boolean	-emb, -epik_metal_binding
EPIK_METAL_BINDING_FILE	string	-mbs
IONIZATION	integer	-i
IONIZER_FILE	string	-is
MAX_ION_SITES	integer	-mg
Tautomerization		
EPIK	boolean	-epik

Table 2.1. LigPrep input file keywords

Keyword	Value Type	Equivalent command line option
EPIK_METAL_BINDING	boolean	-emb, -epik_metal_binding
EPIK_METAL_BINDING_FILE	string	-mbs
NO_TAUTOMERIZER	boolean	-nt
MAX_TAUTOMERS	integer	-t
TAUTOMERS_SPEC_FILE	string	-ts
TAUTOMERIZER_PROB	float	-tp
Stereoisomers		
IGNORE_CHIRALITIES	boolean	-ac
RESPECT_CHIRALITIES or DETERMINE_CHIRALITIES	boolean	-g
NO_STEREOIZER	boolean	-ns
NUM_STEREOISOMERS	integer	-s
STRICT_V2000_STEREO	boolean	-strict_v2000_stereo
Ring conformations		
NUM_RING_CONF	integer	-r
RING_CONF_REL_EN	integer	-re
LINEAR_MOLECULE_TREAT	integer	-l
Filtering		
FILTER_FILE	string	-f
FC_FILE	string	-fc
FS_FILE	string	-fs
TRACKING_LABEL	boolean	-lab
LAB_FILTER	string	-labfilter
LIGFILTER_PROPS	boolean	-lp
Final optimization		
FORCE_FIELD	integer	-bff
BMIN_MINI_IN_VAC	boolean	-bvac
BMIN_DISCARD	boolean	-br
BMIN_MINI_ONLY	boolean	-bns

Table 2.1. LigPrep input file keywords

Keyword	Value Type	Equivalent command line option
BMIN_TORS_CONSTRAINTS	integer	-btc
BMIN_TORS_SAMPLING_LEVEL	integer	-bts
Meta-options		
ADJUST	string	-adjust
EXPAND	string	-expand
RETAIN	string	-retain
VARY	string	-vary
UNTANGLE	boolean	-unt
VCS	string	-vcs
Job options		
INTERVAL	integer	-INTERVAL
JOBCTS	integer	-JOBCTS
LOCAL	boolean	-LOCAL
NICE	boolean	-NICE
OUTPUT_ORG	string	-OUTPUT_ORG
STRICT_END	boolean	-STRICT_END
TMPDIR	string	-TMPDIR
WAIT	boolean	-WAIT

2.5 LigPrep Output Files

LigPrep generates many temporary files due to the many stages of the LigPrep process. Most of these files are automatically deleted unless the program is instructed to keep them.

At each stage there is usually one input structure file, one output structure file, and one log file. For some stages the log files are normally empty, while for others they can be large. In general, the output structure file from one stage is used as the input structure for the next stage. When a stage is complete, the input structure file is removed unless it is the first stage in the process, the only stage that has been requested (with the `-R` option), or the `-nc` option (do not clean up files) has been given. If a stage fails in a manner that causes `ligprep` to terminate, none of the files for that stage are deleted. Log files from all stages are retained.

Most intermediate files are named by adding a suffix to the job name. The job name for all tasks in the LigPrep process is the stem of the output structure file name (the file name minus the extension, if present). For example, the name of the output structure file for `sdconvert` is *jobname*`_sdout.mae`. File name suffixes and other files used are described in Table 2.2. Intermediate output files are compressed unless you specify `-mae_uncompressed` on the command line. The output structure file for the last stage of a successful LigPrep run is renamed to the output structure file name specified on the command line.

Table 2.2. File name suffixes

Stage	Log File	Output Structure Files	Other Files
`sdconvert` `maesubset`	`_sd.log` None	`_sdout.maegz` `_msout.maegz`	`-bad.sdf[.gz]` This file contains structures that could not be converted by `sdconvert`.
`applyhtreat`	`_ht.log`	`_htout.maegz`	Output: `_htout-bad.maegz` Contains problematic structures that are not written to the output structure file.
`desalter`	`_ds.log`	`_dsout.maegz`	Output: `_dsout-bad.maegz` Contains problematic structures that are not written to the output structure file.
`neutralizer`	`_nu.log`	`_nuout.maegz`	Output: `_nuout-bad.maegz` Contains problematic structures that are not written to the output structure file.
`ionizer`	`_ion.log`	`_ionout.maegz`	Output: `_ionout-bad.maegz` This file contains structures flagged as problematic by `ionizer`. These structures do not appear in the final LigPrep output structure file.

Table 2.2. File name suffixes (Continued)

Stage	Log File	Output Structure Files	Other Files
tautomerizer	_ta.log	_taout.maegz	Output: _taout-bad.maegz Contains problematic structures that are not written to the output structure file.
epik	_epik.log	_epikout.maegz	Output: _epikout-bad.maegz Contains problematic structures that are not written to the output structure file.
ligfilter	_lf.log	-lfout.maegz	Input: ligfilter.lff. This file contains the conditions used to eliminate molecules and must follow the -f command-line option.
stereoizer	_st.log	_stout.maegz	Output: _stout-bad.maegz Contains problematic structures that are not written to the output structure file.
ring_conf	_rc.log	_rcout.maegz	Output: _rcout_tmpl_att_prob.maegz. This file contains copies of structures with missing ring system templates or missing attachment axial vs. equatorial energies. _rcout_misc_prob.maegz This file contains copies of structures with other types of problems. ring_conf tries to include all structures in the _rcout.maegz file, even those for which copies are saved in these two problem files.
premin	_pr.log	_prout.maegz	Output: _prout_bad.maegz This file contains structures flagged as problematic by premin. These structures do not appear in the final LigPrep output structure file.
bmin	_bm.log	_bmout.maegz	Output: _bmout.ouL This file becomes _bmout.maegz at the end of a successful bmin run. If bmin fails, then the structures processed up to the failure are present in the former file.

Table 2.2. File name suffixes (Continued)

Stage	Log File	Output Structure Files	Other Files
			Output: _bmout-bad.maegz. This file contains structures that bmin eliminated from further processing, usually because of incompatible chiralities or due to severe distortions.
sdconvert	_sdout.log	_sdout.sdf	None
smiles_to_mae	_smi.log	_smiout.maegz	None

If any structures cannot be processed, the structures from the input file are copied to a file named *jobname*-failed.*ext*, where *ext* is the same as for the input file. A report is given at the end of the log file, listing the failures by the structure number in the input file, and specifying the stage in which the failure occurred.

2.6 LigPrep Limitations

The limitations in this sections apply to the LigPrep process as a whole. Information on the limitations of individual tools is given in the sections for each tool.

Target Molecule Classes

LigPrep is intended for use with largely organic, drug-like molecules. While it may be useful for molecules outside this class of compounds it has not been developed for or extensively tested with such systems.

By default some stages of LigPrep pass over molecules containing more than 200 atoms. This limit can be adjusted using the -ma option.

Processing Problems May Eliminate Molecules From the Output Data Set

It might not be possible for the various stages in the LigPrep process to accommodate all input structures. Under such circumstances, these problematic structures might not be retained in the output structure file for that stage and thus might be absent in the final output structure file from the LigPrep process. The molecules that failed are copied to a separate file. To learn more about this issue, see the descriptions of the individual programs within this document.

Problematic structures might also be returned in the output. These are labeled with a text field that describes the processing problems.

File Size Limits

While most computers can work with large files, such as those exceeding 2 GBytes, working with them often puts heavy strains on operating systems and hardware. Maestro files of this size can contain about 150,000 ligand-like molecules. Some LigPrep process and option selections can significantly increase the number of output structures. Given these considerations we recommend processing around 50,000 structures in a single LigPrep run.

The `ligprep` command provides a convenient way to split and process a large structure file. For more information see Section 2.3 on page 17.

Using LigPrep

Although LigPrep contains a variety of options, you will probably use only a subset of them regularly. The options you choose will depend on your goals, the types of input structures you are processing, and the types of subsequent processing that will be carried out. For a complete list of LigPrep options, see Section 2.2 on page 16. A general description of how LigPrep processes structures is provided in Section 1.2 on page 2.

This chapter provides a few examples showing how to use LigPrep. The first section demonstrates the use of the LigPrep panel in Maestro, with input structure files provided in `$SCHRODINGER/macromodel-v`*version*`/ligprep/samples/examples`. A range of command-line examples is given in the second section.

In general, we recommend limiting the size of the input structure file to 20,000 to 50,000 structures. For larger input files, we recommend using `ligprep` to split up and process the structures in smaller batches. Some LigPrep stages do not adjust structures with more than 200 atoms, to avoid spending excessive processor time on molecules that are atypical for ligands. This limit is adjustable using the LigPrep `-ma` option. However, increasing it significantly will result in the processing of ligands larger than LigPrep is tested for, and you can encounter problems, including much slower processing of structures. For information on file size issues, see Section 2.6 on page 24.

Since LigPrep uses Schrödinger's job control facility, jobs are run in the background and the command-line prompt returns immediately after you enter the `ligprep` command. Processing continues until the job has finished, at which time the output files appear in the job submission directory. For more information on the Job Control facility, see the *Job Control Guide*.

3.1 LigPrep Panel Examples

The LigPrep panel in Maestro enables you to prepare and run a variety of ligand preparation jobs. This section provides examples of three ligands and illustrates the effect that LigPrep has on their structure.

3.1.1 Preparing for the Exercises

To run the exercises, you need a working directory in which to store the input and output, and you need to copy the input files from the installation into your working directory. This is done automatically in the Tutorials panel, as described below. To copy the input files manually, just unzip the `ligprep` zip file from the `tutorials` directory of your installation into your working directory.

On Linux, you should first set the `SCHRODINGER` environment variable to the Schrödinger software installation directory, if it is not already set:

csh/tcsh: `setenv SCHRODINGER` *installation-path*

sh/bash/ksh: `export SCHRODINGER=`*installation-path*

If Maestro is not running, start it as follows:

- **Linux:** Enter the following command:

  ```
  $SCHRODINGER/maestro -profile Maestro &
  ```

- **Windows:** Double-click the Maestro icon on the desktop.

  You can also use Start → All Programs → Schrodinger-2015-2 → Maestro.

- **Mac:** Click the Maestro icon on the dock.

  If it is not on the dock, drag it there from the SchrodingerSuites2015-2 folder in your Applications folder, or start Maestro from that folder.

Now that Maestro is running, you can start the setup.

1. Choose Help → Tutorials.

   The Tutorials panel opens.

2. Ensure that the Show tutorials by option menu is set to Product, and the option menu below is labeled Product and set to All.

3. Select LigPrep Panel Examples in the table.

4. Enter the directory that you want to use for the tutorial in the Copy to text box, or click Browse and navigate to the directory.

   If the directory does not exist, it will be created for you, on confirmation. The default is your current working directory.

5. Click Copy.

   The tutorial files are copied to the specified directory, and a progress dialog box is displayed briefly.

If you used the default directory, the files are now in your current working directory, and you can skip the next two steps. Otherwise, you should set the working directory to the place that your tutorial files were copied to.

6. Choose Project → Change Directory.
7. Navigate to the directory you specified for the tutorial files, and click OK.

You can close the Tutorials panel now, and proceed with the exercises.

3.1.2 Generating Variations on 2D Structures

The two structures provided in this example illustrate the variations that are generated with the default LigPrep options.

To import and examine the structures:

1. Click the Import structures button on the Project toolbar.

   

   The Import panel opens.

2. Select the file `2D_variations.sdf`.
3. If the import options are not displayed, click Options.
4. Ensure that Import all structures is selected.
5. Click Open.

   The first structure should be displayed in the Workspace. This structure has two ionizable groups—a carboxylic acid and a pyridine—and a quaternary ammonium counter ion. There are three chiral centers, of which two have chirality information, and the hydrogen atoms are implicit.

6. To verify that this structure is indeed two-dimensional, click the Rotate Y button on the View toolbar, then click the Reset button (or click the Rotate Y button 3 more times).

7. Display the Project Table by clicking the Table button on the Project toolbar.

8. Click the In column for entry 2.

   The second structure should be displayed in the Workspace. This structure also has two ionizable groups—an imidazole ring and a 4-nitrosophenol group. The neutral state of the imidazole ring has two tautomers.

To set up and run the LigPrep job:

1. In the Project Table, click on the `2D_variations` entry group row.
2. In the main window, choose Tasks → Ligand Preparation.

   The LigPrep panel opens.

3. For Use structures from, select Project Table (selected entries).
4. Check that the default options are selected:
   - Ionization: Generate possible states at target pH 7.0 +/- 2.0
   - Using: Epik
   - Desalt
   - Generate tautomers
   - Stereoisomers: Retain specified chiralities
   - Stereoisomers: Generate at most: 32 per ligand
   - Generate low energy ring conformations: 1 per ligand
   - Output format: Maestro
5. Click the Settings button to open the Job Settings dialog box.
6. Under Incorporate, select Append new entries as a new group.
7. Type `variations` in the Name text box.
8. Click Run.

   When the job finishes, there should be 8 new entries in the Project Table in an entry group named variations.

To examine the output structures:

1. In the Project Table, click the In column for entry 3, then control-click the In column for entry 4.

   The two structures generated from the first 2D structure are displayed in the Workspace, superimposed. Note the variations in the structures and the absence of the quaternary ammonium ion, which was removed by the `desalter`.

2. Click the Tile button on the View toolbar.

   

   The structures are displayed side-by-side in the Workspace. Each has a full complement of hydrogen atoms. The carboxylate group is unprotonated because its pK_a lies outside the target pH range of 5 – 9. The pyridine is unprotonated.

3. From the Label All button menu on the Labels toolbar, choose Chirality.

   

   The chiral centers are labeled R or S. Two of the chiral centers are labeled R in both structures: these chiralities were preserved from the input structures. The remaining center is labeled R in one of structure and S in the other. This center did not have chirality information in the input structure, and its chirality has been varied.

4. Click the Tile button on the View toolbar.

   The structures are superimposed again.

5. In the Project Table, click the Row column for entry 5, then shift-click the In column for entry 10.

   The six structures that were generated from the second 2D structure are selected.

6. Choose Entry → Include Only.

   The selected structures are displayed in the Workspace, superimposed. Note the variations in the structures.

7. Click the Fit to Workspace button on the View toolbar.

   

8. Click the In column for entry 5.

   This entry is displayed, and the other entries are undisplayed. The phenol group and the imidazole group are both protonated in this structure.

9. Click the Next button on the ePlayer toolbar. (If the ePlayer toolbar is not displayed, display it from the ePlayer menu.)

   

   The next structure is displayed in the Workspace. The imidazole group is unprotonated.

10. Click the Next button again.

    Note the tautomerization of the imidazole group.

11. Click the Next button three more times to display the remaining structures.

    The last three structures have the phenol group deprotonated.

3.1.3 Generating Ring Conformations of a 3D Structure

The structure in this example has two flexible rings and two chiral atoms. The ring conformations will be varied, but the chiralities and ionization state will be kept.

To import and label the structure:

1. In the main window, choose Project → Close.
2. Click Discard in the dialog box that opens.

   The Workspace and the Project Table are cleared.

3. Click the Import button on the Project toolbar.

   

   The Import panel opens.

4. Select the file `3D_rings.mae` and click Open.

   The 3D structure is displayed in the Workspace.

5. From the Label All button menu on the Labels toolbar, choose Chirality.

   

To set up and run the LigPrep job:

1. If the LigPrep panel is not displayed, choose Tasks → Ligand Preparation in the main window.
2. Under Use structures from, select Workspace (included entries).
3. Under Stereoisomers, select Determine chiralities from 3D structure.
4. Under Ionization, select Do not change.
5. Type `8` in the Generate low energy ring conformations text box.

   The remaining options do not affect the outcome of this job.
6. Type `rings` in the Job name text box.
7. If you are continuing from the previous exercise, click Run. Otherwise, follow the instructions below:

   a. Click the Settings button to open the Job Settings dialog box.

   b. Under Incorporate, select Append new entries as a new group.

   c. Click Run.

   When the job finishes, there should be eight new entries in the Project Table, with the first of the new entries displayed in the Workspace.

To examine the output structures:

1. In the Project Table, click the In column for entry 2, then shift-click the In column for entry 9 to display the structures in the Workspace, superimposed.
2. Click the Tile button on the View toolbar.

   

   The entries are now displayed separately, in a grid.
3. Deselect Transform all tiles and Pick Tiles to Move on the Tile button menu.
4. As necessary, click on each molecule and rotate it (middle mouse button) so that you can see the conformation of each ring.

   Both the 1,4-dioxane ring and the piperidine ring adopt the chair conformation in all molecules. What differs between the molecules is the joining geometry of each ring (axial or equatorial) and the orientation of the hydrogen on the nitrogen in the piperidine (axial or equatorial).

5. In the Project Table panel, right-click on the heading of the Potential Energy-OPLS-2005 property column and choose Workspace Feedback → Add.

   The Workspace feedback text now shows the potential energy. Note that the two molecules in which the rings are both joined axially have the highest energy, and the two in which the rings are both joined equatorially have the lowest energy. Of the remaining molecules, the highest in energy are those in which the piperidine is joined axially. The axial nitrogen conformation is about 2 kJ/mol higher than the equatorial conformation.

3.2 Command-Line Examples

The LigPrep panel in Maestro provides access to the main `ligprep` command options. For finer control of the options, you must run LigPrep from the command line. If you want to prepare a large number of ligands, you might prefer to run LigPrep from the command line. This section provides a number of examples of the `ligprep` command.

3.2.1 Default Operation

LigPrep can be run using the default settings by specifying only the input and output files. Examples follow showing syntax using different input file types; both examples include output files in Maestro format:

```
$SCHRODINGER/ligprep -isd my_2D_cts.sd -omae my_3D_cts.mae
```

In this example, the input file is in SD format.

```
$SCHRODINGER/ligprep -ismi my_1D_cts.smi -omae my_3D_cts.mae
```

In this example, the input file is in SMILES format.

The above commands do the following:

- Adds hydrogen atoms
- Neutralizes many types of charged functional groups
- Generates up to 8 tautomers for each input structure
- Assigns specified chiralities and varies unspecified chiralities for each tautomer
- Generates one set of ring conformations for each stereoisomer
- Minimizes the energy for each stereoisomer

The resulting 3D structures are written to the output file. Only the largest molecule in each input structure is retained.[3] Some of these defaults can be overridden by specifying additional command options.

3. LigPrep removes all molecules except the one with the largest number of atoms in the structure to ensure that co-crystallized counter ions are removed.

Valid combinations of chiralities, recorded as parities or bond directions in the input structure file, are retained in almost all cases. Stereoisomers are generated by varying the chiralities of chiral atoms whose chiralities are not specified. It is generally advisable to produce multiple stereoisomers, especially for fused ring systems, in which some combinations of chiralities and atom-numbering chiralities are unacceptable and may be eliminated from the set of output structures. For Maestro-formatted input structure files the chiralities recorded as properties are respected. To vary the chiralities of all chiral atoms present, use the `-ac` option. If you use this option, it might be desirable or necessary to increase the maximum number of stereoisomers per molecule from its default value of 32 to a higher limit. For example, the option `-s 128` permits up to 128 stereoisomers to be generated.

3.2.2 Producing One Output Structure for Each Input Structure

LigPrep is primarily designed to produce high quality 3D structures and, if possible, to produce multiple output structures for each input structure by introducing variations in the chemical nature of the structures, such as adjusting the stereochemistry or the ionization (protonation) state. However, LigPrep can be used so that it typically produces only one output structure for each input structure.

3.2.2.1 Adjusting Only the Geometry

One extended-conformation, energy-minimized 3D structure can be produced for each 3D input structure from the SD file, `my_3D.sd`, using the command:

```
$SCHRODINGER/ligprep -i 0 -nt -s 1 -g -isd my_3D.sd  -omae
my_improved_3D.mae
```

For the Maestro-formatted input file `my_3D.mae`, use the command

```
$SCHRODINGER/ligprep -i 0 -nt -s 1 -g -imae my_3D.mae -omae
my_improved_3D.mae
```

The options selected with these commands add hydrogens to structures that lack them to produce all atom structures, but neither command should adjust the protonation state of the ionizable groups (`-i 0`). The `-nt` option instructs LigPrep not to adjust the tautomeric state. The `-g` option requests that the chiralities, which are needed for ring conformation generation, be determined from the input geometry. If parts of the molecule are quite distorted, it might not be possible to determine the chiralities of all the chiral atoms unambiguously. In such cases, LigPrep systematically varies the chiralities that could not be determined to generate possible stereoisomers (up to 32 by default). The `-s 1` option instructs LigPrep to generate only one such structure, with all undetermined chiralities set to R. If there is more than one indeterminable chirality, incompatible chiralities can be selected (for instance, in fused ring systems) and the molecule could be eliminated in subsequent processing.

For 2D input structures, the corresponding command would be:

```
$SCHRODINGER/ligprep -i 0 -nt -s 1 -isd my_2D.sd -omae
my_improved_3D.mae
```

This command has the same options as the previous example except that the `-g` option has been removed, and therefore LigPrep looks for stereochemical specifications in the input structure file to determine chiralities. As with the 3D examples, LigPrep normally would vary the unspecified chiralities to generate multiple stereoisomers. As before, the `-s 1` option forces the output of only one stereoisomer. However, since it is quite common for 2D structures to have multiple chiral centers unspecified, the risk of eliminating structures due to incompatible chiralities is much higher than with 3D structures. You might want to compromise and permit the generation of multiple stereoisomers and thus multiple output structures for some input structures.

3.2.2.2 Adjusting the Chemistry

LigPrep can adjust the chemistry of the input structure by removing counter ions (desalting), neutralizing or ionizing ionizable groups, changing between tautomers and generating stereoisomers. Currently it is not possible to guarantee that each input structure will produce only one output structure when adjusting the chiralities or ionization states. A command that might usefully approximate the goal of producing only one output structure per input structure is:

```
$SCHRODINGER/ligprep -s 1 -i 2 -W i,-ph,7.0,-pht,0.0 -t 1 -imae
input.mae -omae output.mae
```

As before, the `-s 1` option forces the generation of only one stereoisomer, and chiral atoms whose chiralities were not previously defined are classified as R. To avoid eliminating molecules in which such chiralities are incompatible, a practical compromise is to permit the generation of several stereoisomers (e.g., `-s 4` to generate up to 4). The desalter is run by default, so only the largest molecule in each structure is retained. The `-t 1` option results in the generation of the tautomer judged to be the most probable one.

The `-i 2` option requests the running of the `ionizer`, and options are passed to it by means of the `-W i,-ph,7.0,-pht,0.0` option. These options request that functional groups with pK_a values less than or equal to 7.0 be deprotonated, while those with pK_a values greater than or equal to 7.0 be protonated. For most input structures, this results in only one output structure. However, for molecules that have functional groups with pK_a values of exactly 7, multiple structures are produced, each with a different protonation state.

3.2.3 Generating Variations

Often it is desirable to generate a number of variations on each input structure. The main benefit is that there is a better chance that the resulting collection of structures includes the "best" form for your particular study. However, as more variations are generated, proportionately more CPU time is consumed, both in LigPrep and in subsequent processing. Another drawback of generating more variations is that improbable states can be generated, which at present are not penalized in subsequent processing.

There are many ways to introduce variations in the structures using LigPrep. Here are a few examples showing how to do this.

3.2.3.1 Generating a Small Set of Low-risk Variations

To generate a few low-risk variations on the input structures:

```
$SCHRODINGER/ligprep -r 1 -s 8 -t 4 -i 2 -W i,-pht,1.0 -imae
input.mae -omae output.mae
```

For each structure in the input structure file, LigPrep generates an extended conformation containing only the lowest energy ring conformation for any rings present, up to 8 stereoisomers, up to 4 tautomers, and ionization states highly populated in the pH range 6.0 to 8.0 (7.0 ± 1.0). Typically, far fewer than 8 stereoisomers and 4 tautomers would be produced.

This command assumes that the available information on the stereoisomers is recorded as R/S properties in the input structure file. If the input structures are 3D and have the correct chiralities, you could use the `-g` option to instruct LigPrep to respect the chiralities present in the 3D geometry.

3.2.3.2 Generating Many Variations

To generate many variations:

```
$SCHRODINGER/ligprep -r 4 -s 64 -t 8 -i 2 -W i,-pht,2.0
-imae input.mae -omae output.mae
```

This command generates many variations for ring conformations, stereoisomers, tautomers, and ionization states. Not all variations are necessarily kept in the output: some may be eliminated because they have poor bonding patterns, others because they lie too high in energy. In a typical LigPrep run with these options, the number of structures increases by about a factor of 8. The actual expansion factor might be smaller or larger, depending on your data set.

3.2.3.3 Using Meta-Options for Convenient Selection of Run Characteristics

For some scenarios, a set of nondefault options is needed. Rather than having to understand and specify each option separately, you can use the `ligprep` command's more intuitive meta-options, which include:

`-unt`	Untangle structures without changing any atoms or bonds.
`-adjust_`*itc*	Adjust to a suitable state.
`-vary_`*itc*	Generate different states.
`-expand_`*itc*	Aggressively generate different states.
`-retain_`*itc*	Retain characteristics of the input structures. Specifying `i` also turns on retention of tautomers.

itc can be replaced by one or more of:

- `c` chiralities
- `i` ionization and neutralization
- `t` tautomerization

The `adjust`, `vary` and `expand` meta-options control the range of structures generated. All three of these meta-options may be used in the same `ligprep` command, but each of `c`, `i` or `t` must only appear in one of them: for instance, you could specify `-adjust_t -vary_c -expand_i`. The `retain` meta-option requires that at least one structure produced for each input structure retains the characteristics specified by the suffix. For instance, `-retain_t` reqires the tautomerization state in the input structure to be present in at least one of the output structures for that molecule. `retain` can be used with any of `adjust`, `vary`, and `expand`. `retain` can be used to produce a range of structural variations while keeping the form in the input structure—something that might be useful in lead optimization.

As an example, the following command permits the generation of moderate variations on the ionization and tautomerization state and significant numbers of stereoisomers, while ensuring that the input tautomeric form appears in the output:

```
$SCHRODINGER/ligprep -vary_it -expand_c -retain_t
    -imae input.mae -omae output.mae
```

This command is equivalent to:

```
$SCHRODINGER/ligprep -i 2 -W i,-pht,1.0 -r 1 -t 2 -s 32
    -imae  input.mae -omae output.mae
```

If other options are given with a meta-option, their values override the values set by the meta-option. For instance,

```
$SCHRODINGER/ligprep -vary_it -expand_c -retain_t -W i,-pht,1.5
    -imae input.mae -omae output.mae
```

extends the pH range from that used by `-vary_i`, ±1 (pH 6-8), to ±1.5 (pH 5.5-8.5).

3.2.4 Filtering Structures

During LigPrep processing, structures matching certain conditions can be removed using `ligfilter`. A file containing a list of these conditions must be created before running LigPrep. For example, to remove molecules with a molecular weight greater than 650 and molecules that have too many hydrogen bond acceptors, you could create a file with the following content:

```
# Remove molecules that have a molecular weight of greater than 650
Molecular_weight                        > 650
# Remove molecules with too many H-bond acceptor and donor atoms
Acceptor_groups                         > 3
Donor_groups                            > 3
```

The following command uses `ligfilter` to filter the input structures using the above file, saved as `filter.lff`:

```
$SCHRODINGER/ligprep -f filter.lff -isd my_2D.sd
-omae my_3D.mae
```

For more information on using `ligfilter` and creating filtering files, see Section 4.9 on page 47.

3.2.5 Running LigPrep on a Remote Host

One advantage of the job control facility is that jobs can be submitted to remote hosts that have been prepared to run Schrödinger software. For instance, to run LigPrep on a host called `fast_cpu`, you can add the `-HOST` option to a command:

```
$SCHRODINGER/ligprep -HOST fast_cpu -s 1 -imae my_2D.mae -omae
my_3D.mae
```

When the LigPrep job finishes on `fast_cpu`, the output files are copied back to the directory from which you started the job. For information on the Job Control facility and on preparing hosts and batch queues to run Schrödinger software, see the *Job Control Guide*.

3.3 Chirality Manipulations Using Maestro and LigPrep

Both Maestro and LigPrep have the ability to manipulate the stereogenic composition of a three-dimensional molecular structure. Maestro includes graphical features that can exchange the chirality of a single center, as well as easily generate the enantiomer of a displayed structure. Correspondingly, a LigPrep preparation has the ability to sample some or all of the chiral centers, in one or multiple input structures, during a given preparative computation. Finally, the combination of the Project Table and LigPrep enables the expansion or inversion of selected subsets of chiral centers in one or more structures easily and quickly.

Simple inversions of a single chiral center, located in a structure displayed by Maestro, can be easily accomplished by using the R/S button on the Build toolbar. Click the button, then pick the atoms in the Workspace whose chirality you want to invert.

You can also use the Chirality tab of the Adjust panel. To open this panel, choose Edit → Adjust → Chirality in the main window. To reverse the chirality of a given stereogenic center, select the chiral atom, then select two other "non-moving" atoms connected to the chiral center. The remaining two groups connected to the selected chiral atom are switched, resulting in an effective inversion of the chirality at the atom. These operations can only be performed on chiral atoms that are not a part of a ring system. To invert the chirality of an individual ring system atom, use LigPrep as described below.

If you want to generate the enantiomer of the structure, click All in the Chirality tab of the Adjust panel. This operation inverts all the chiral centers and produces the enantiomer. The *z*-coordinate values for each atom in the structure are inverted by this procedure.

In addition to Maestro's built-in graphical tools, which modify individual chiral centers and produce enantiomers, LigPrep can also generate and sample alternative chiralities on a broader scale. Generally, in addition to 2D to 3D conversion, LigPrep can sample and expand ionization states, tautomers, ring conformations, and alternative chiralities. For the latter, both Maestro-formatted files and SD-formatted files have the internal ability to indicate the specified chirality at some or all stereogenic centers, usually "R" or "S". LigPrep and the Stereoizer have three basic usage modes that utilize this information:

- Retain specified chiralities—For structures with *specified* chirality information, LigPrep attempts to generate output structures that match the chirality indications at all such stereogenic centers. LigPrep tries to match the specification whether or not the input 3D geometry about the chiral centre (if any) matches this specification. All unspecified chiral centers in the structure are expanded.

- Determine chiralities from 3D structure—Specified chiralities are ignored, and no alternative chiralities are generated unless the chirality cannot be determined from the structure (as is the case for 2D structures). Output structures have the same stereogenic makeup as the input, and all chiral centers have their specification changed to match the input structure.
- Generate all combinations—The input specifications are ignored, and and geometrically feasible stereoisomers (up to the maximum number requested) are generated.

The specifications discussed can be observed in the Project Table as properties with titles such as Chirality 1 and specifications like `9_R_8_13_10_37`. In this example, atom number 9 has the "R" configuration based on the listed atoms with CIP priorities. MDL SD files have their own chirality-indicating format, which is converted to the Maestro format upon importing the SD-formatted structures.

If the chirality specifications were not included in the original structure file, or the molecule was constructed in Maestro, the specifications can be easily assigned by simply running LigPrep with the Stereoizer set to Determine chiralities from 3D structure, and setting the output to Replace existing entries or Append new entries. You may select to turn off the Ionizer, Tautomerizer, and Desalter if you prefer no additional expansion when only specifying chiralities. After incorporation, the chirality specifications are listed in the Project Table.

Once the chirality specifications are listed in the Project Table for the structures of interest, the combination of the Project Table and LigPrep provide a simple, easy method to invert or sample a subset of chiral centers in one or more structures.

To invert a chiral center using LigPrep, simply adjust the specification in the appropriate property cell for the structure of interest to the other "R" or "S" indication, for instance from `9_R_8_13_10_37` to `9_S_8_13_10_37`. You may want to duplicate the entry in the Project Table prior to adjusting the designation. Using the modified entry in a subsequent LigPrep computation with the Retain specified chiralities mode, generates the requested chirality change. The user can make changes to multiple structures by modifying any of the specified chiralities, selecting all of the modified structures in the Project Table, and running the LigPrep preparation with Use structures from set to Selected entries.

In contrast, to sample or expand alternate chiralities once the chiral centers are specified, simply delete or remove the specification from the Project Table cell. A subsequent LigPrep preparation, using the Retain specified chiralities mode, maintains the other, indicated chiralities, but expands the chiralities of the centers with specifications removed. Multiple structures can be expanded in this way using selected entries as the input.

Tools Used by LigPrep

This chapter describes the tools used by LigPrep. A description of the purpose and the syntax is provided for tools that are unique to LigPrep, and a reference to the full description is provided for common utilities. A description of the options can be obtained by running the command with the `-h` option. The options for the tools cannot be used directly as options for `ligprep`. The `ligprep` options can be used to control the behavior or the tools, and for some tools, options can be passed from the `ligprep` script to the tool with the `ligprep -W` option. The descriptions are provided so that you can understand how these tools function in the context of LigPrep, and so that you can use these tools separately from LigPrep.

LigPrep can use a separate product, Epik, instead of the `ionizer` and the `tautomerizer`. For more information on Epik, see the *Epik User Manual*.

4.1 sdconvert

This utility converts between MDL SD, Maestro, and MacroModel format files. See Section 1.4 of the *General Utilities* document for details of this utility. In LigPrep, this utility is ron with the `-bad` option to save structures that could not be converted.

4.2 smiles_to_mae

This utility converts a SMILES file or a CSV file with SMILES strings to a Maestro-formatted file. This utility should be used in conjunction with LigPrep.

Syntax:

`$SCHRODINGER/utilities/smiles_to_mae` [*options*]
{*inputfile*`.smi`|*inputfile*`.csv`} *outputfile*`.mae`

The first argument is taken as the input file and the second as the output file.

If the input file is a SMILES file, each line in the file must consist of the SMILES string, followed by an optional title.

If the input file is a CSV file, the SMILES strings must be in the first column. If the file has a header row, the first column should be labeled `SMILES`, and the title column should be labeled one of `NAME`, `title`, or `s_m_title`. If there is no header row, the second column is taken to be the title. Any other columns are treated as properties, and are read in; if there is no header

row, the properties are labeled as “smiles integer 1”, “smiles real 1”, “smiles string 1”, and so on.

4.3 maesubset

This utility selects a subset of the structures present in a Maestro-formatted file. See Section 2.1 of the *General Utilities* document for details of this utility.

4.4 applyhtreat

The `applyhtreat` utility can be used to delete or add hydrogen atoms and lone pairs to one or more structures in a Maestro-formatted file, consistent with the hydrogen treatment option that you provide. A *hydrogen treatment* is a protocol that determines which atoms are to have hydrogens and lone pairs attached. Several treatments are supplied. Each of these treatments is associated with a particular molecular mechanics force field, but some treatments are suitable for several force fields. See Section 4.1 of the *General Utilities* document for details of this utility.

LigPrep processing requests adjustment of chirality information, otherwise the atom numbers in the chirality properties would remain unchanged. This can result in corruption of the chirality specifications and lead to incorrect results.

4.5 desalter

Individual structures present in many databases contain multiple molecules. In many cases these are co-crystallized components such as counter ions. These components often need to be removed before further calculations are performed. The `desalter` eliminates all molecules except the molecule with the largest number of atoms (including hydrogen atoms, if present) in each structure. If there is more than one molecule containing the largest number of atoms, the molecule with the lowest numbered atom within the structure is retained.

You must have a LigPrep license to run the `desalter`.

Syntax:

`$SCHRODINGER/utilities/desalter [-h] [-v]` *input_file*`.mae` *output_file*`.mae`

4.6 neutralizer

Ionizable groups present in structures could be in any of a number of charge states. Some programs, such as the `ionizer`, do not successfully process input structures containing charged groups or molecules. The `neutralizer` neutralizes functional groups where possible by adding or removing protons. The following replacements and additions are made in the neutralization process:

- Sodium or potassium ions with a single bond are replaced by hydrogens bound to the same atom.
- $(R)_x–NH^+$ is replaced by $(R)_x–N$.
- A proton is added to O^- or S^- atoms bound to C, S or P atoms.
- A proton is added to O^- when it is bound to a N that has a formal charge of 0. Thus, nitro groups and pyridine N-oxides are not protonated, but a hydroxamate is protonated.
- Dummy atoms are removed.
- A proton is removed from $H–O^+R_2$.
- A proton is removed from $H–O–N^+(=O)–R$ and an electron to $O–N^+(=O)–R$, to produce $O^–N^+(=O)–R$.
- A proton is removed from a protonated pyridine N-oxide.
- A proton is added to the nitrogen in sulfonamide $O=S(=O)-N^-$ anions.
- A proton is added to the negatively charged N atom in histidine anions.

You must have a LigPrep license to run the `neutralizer`.

Syntax:

```
$SCHRODINGER/utilities/neutralizer [options] input_file.mae
output_file.mae
```

Limitations:

Some groups, such as quaternary N atoms, are not neutralized.

4.7 ionizer

The `ionizer`, or ionization state expander, produces multiple structures for each input structure with different combinations of ionized states based on the ionizable groups present. By default, the `ionizer` produces both protonated and deprotonated forms of groups that have a

pK_a between 5 and 9. Groups with a higher pK_a are protonated; groups with a lower pK_a are deprotonated. These defaults ensure that the appropriate structures are available for physiological conditions. For more information, see Chapter 5.

The `ionizer` is not called from the `ligprep` script by default. To run the `ionizer` with LigPrep, select Generate Possible States under Ionization in the LigPrep panel, or add the `-i 2` option of the `ligprep` command. The `ionizer` is also called from `ligprep` if you use the `-expand_itc` option. The `ionizer` can be run on its own from the `ligprep` script with the `-R i` option. You can pass options to the `ionizer` from `ligprep` with the `ligprep -W i` option. Not all `ionizer` options are available from `ligprep`; see Section 5.1 on page 61 for more information.

If an input structure contains more than a prescribed number of ionizable groups, the structure is skipped and passed, unchanged, to the output file. To set the limit on the number of ionizable groups, use the `-mg` option of the `ligprep` command. You cannot set the limit to a value greater than 31. The default for running the `ionizer` from `ligprep` is 10.

4.8 tautomerizer

Some isomeric forms of molecules coexist and are interchangeable under physiological conditions. Tautomers constitute an important class of such isomers. For some types of calculations, such as docking with Glide, considering the appropriate tautomeric forms of ligands is nearly as important as examining the appropriate ionization state. LigPrep uses the `tautomerizer` by default to generate probable tautomeric states. To skip the generation of tautomers, clear the Generate Tautomers option in the LigPrep panel of Maestro, or use the `-nt` option of the `ligprep` command.

The `tautomerizer` assigns a probability to each tautomeric form. The tautomers for a given input structure are produced in order of probability, the most probable forms first. By default, up to eight tautomers are generated for each structure. The maximum number of tautomers to generate for each structure can be specified with the `-t` *number* option of the `ligprep` command. In addition, the generation of improbable tautomers can be skipped by setting a threshold probability with the `-tp` *number* option. Only tautomeric forms with overall probabilities higher than this threshold will be produced. The default threshold is 0.01. If all forms have probabilities lower than the threshold, the most probable tautomer is still recorded to avoid eliminating that molecule completely from further consideration.

You must have a LigPrep license to run the `tautomerizer`. For more information on tautomers and the `tautomerizer`, see Chapter 6.

4.9 ligfilter

ligfilter is a versatile utility for filtering molecules based on property values and descriptor counts. LigPrep uses ligfilter to filter molecules, usually to eliminate those matching certain conditions (e.g., molecular weight > 650 amu), although other types of filtering are possible. This section briefly documents ligfilter as it is used within LigPrep. See Section 2.4 of the *General Utilities* document for details of this utility.

ligprep does not call ligfilter unless ligprep is given the -f option, which accepts the name of a file containing criteria to use for filtering. For example, to remove molecules with a molecular weight greater than 650, molecules that have too many hydrogen bond acceptors and donors, and too few atoms, you could create a file with the following content:

```
# Remove molecules that have a molecular weight of greater than 650
Molecular_weight                          > 650
# Remove molecules with too many H-bond acceptor and donor atoms
Acceptor_groups                           > 3
Donor_groups                              > 3
# Remove molecules with fewer than 10 atoms
Num_atoms                                 < 10
```

The spaces between the descriptors and the conditions are not significant.

The following command uses ligfilter to filter the input structures using the above file, saved as ligfilter.lff:

```
$SCHRODINGER/ligprep -f ligfilter.lff -isd input.sd
-omae output.mae
```

It is also possible to use ligparse for filtering, with the -use_ligparse option, though its use is deprecated. ligparse is described in Appendix A. You must have a LigPrep license to run ligparse. If you use ligparse to filter molecules, the -any and -j 5 options are passed to ligparse by default. With these options, if any of the criteria given (-any) match for a particular molecule, the molecule is eliminated (-j 5) from processing. If options are passed to ligparse using the ligprep -W option, the -any and -j 5 options are no longer passed by default, so that other filtering strategies can be used, such as retaining only the molecules that match the criteria specified. The output files from ligparse are renamed by LigPrep. The _cull.mae file is renamed _lpout.mae and the .out file is renamed _lp.out.

4.10 stereoizer

The `stereoizer` has two modes: label mode and stereoisomer mode. In label mode, various stereochemical features in the input structures are labeled, such as atom chiralities (R/S) or double-bonded atom conformations (E/Z for groups of atoms connected by an odd number of bonds or P/M for groups of atoms connected by an even number of bonds). In stereoisomer mode, the `stereoizer` generates multiple output structures for each input structure, based upon the chiral properties of the chiral atoms in the structure. The coordinates of the atoms are not modified in the copies; only the chiral properties included in the structure for the chiral atoms are distinct. The structures produced in stereoisomer mode should be processed by a LigPrep-licensed version of MacroModel to adjust the atomic coordinates to match those required for the various stereoisomers and to eliminate combinations of chiralities that are very unfavorable. To use stereoisomer mode, set the `-n` option. To use the label mode, omit the `-n` option.

In stereoisomer mode, the chiralities of the atoms specified in the input structure or that can be determined from the geometry are retained. Only the chiralities of the other atoms are varied. If the `-a` option is specified, the input chiralities are ignored and the chiralities of all chiral atoms are varied as follows. Stereoisomers are generated by varying chiralities systematically starting from those that can be determined from the input geometry. Each internally generated structure is examined to see if it passes geometric or natural product restrictions on chiralities. Structures passing this stereoisomer filter are recorded in the output structure file. This process of generating stereoisomers is repeated until either all possibilities are sampled or until the maximum number of stereoisomers per input structure is reached.

Stereoisomer filtering is based upon geometric and natural product considerations. In functionalized norbornane both bridging atoms can be chiral. The geometry of these attachments is tightly coupled by geometric considerations so of the four combinations of chiralities, only two are possible. The stereoizer has geometric restrictions on a good number of simple fused ring systems, including norbornane, adamantane, and 2,2,2-bicyclooctane. Natural products (such as steroids and peptides) and their derivatives are often present in databases of drug-like molecules. These compounds often have multiple chiral centers many of which are customarily unspecified. Consequently it is possible to generate many uninteresting stereoisomers and not generate some of the important ones within the maximum number that the stereoizer is to generate. To reduce this problem the stereoizer is able to enforce the geometries of chiral atoms in L alpha amino acids and steroids.

Geometric restrictions on stereoisomers are more generally and strictly enforced. However, depending on the type of natural product and the project at hand, natural product restrictions should or should not be enforced. To deal with these distinctions, each type of stereoisomer filtering is placed in an enforcement class and may also permit violations for chiralities speci-

fied in the input structure. To deal with these nuances the `stereoizer` has two options, `-pat` and `-pat_keep_orig_off`. The former controls the general class of enforcement while the latter can be used to turn off retention of input structures that violate the restrictions on stereoisomers for those patterns that would otherwise permit them. Geometric restrictions are in class 1 while steroids are in class 2 and peptides are in class 3. Steroids and peptides permit violations of the restrictions for chiralities explicitly listed in the input structure. These restrictions apply to chiralities and atom numbering chiralities. For labeling purposes, the atom property NoSrcChirality (`b_st_NoSrcChirality`) is added and set to `true` if the atom is chiral but the chirality was not specified on input.

When LigPrep calls the `stereoizer`, it uses `-n 32`. If you are running LigPrep with more than 5 chiral centers in any structure and want to generate all combinations, use the `-s` option of the `ligprep` command to specify the number of stereoisomers, or the `-n` option of the `stereoizer` command. The syntax is as follows.

```
$SCHRODINGER/utilities/stereoizer [options] input-file.mae
output-file.mae
```

You must have a LigPrep license to run the `stereoizer`.

4.10.1 Stereochemical Properties

Stereochemical properties created and recognized by the `stereoizer` include those related to atom chiralities and to the conformations of doubly bonded series of C and N atoms. Another type of stereochemical information, atom numbering chiralities, is used to help track higher-order stereochemical features like syn and anti 1,4, attachments to cyclohexane.

Chiral atom properties have the form:

A_N_C_N_1_N_2_N_3_N_4

where:

- A_N is the atom number for the chiral atom.
- C specifies the chirality of atom A_N and is either ‘R’ or ‘S’.
- N_1 through N_4 give the atom numbers of the atoms attached to atom A_N in decreasing order of their Cahn-Ingold-Prelog (CIP) priorities. If only 3 atoms are bonded to atom A_N, then _N_4 is missing.

For example, the property 7_S_15_1_8_62 indicates that atom 7 has a ‘S’ chirality and that the four atoms attached to it in order of decreasing CIP priority are: 15, 1, 8, and 62.

Atom numbering chiralities are analogous to chiralities except that the atoms themselves need not be chiral. These specifications are used to describe higher-order stereochemical features

like the relative attachment geometries of groups symmetrically attached to rings. These only originate from bond direction specifications in SD files. The syntax for atom numbering chiralities is the same as for chiralities just described except that the ANR and ANS are used instead of R or S and the bonded atoms are ranked solely based upon their atom numbers.

The isomers of doubly bonded series of C or N atoms with distinct attachments at each end are designated with properties of the form:

$N_1\_N_2\_....\_N_i\_G$

where:

- N_2 through N_{i-1} are the atom numbers of the doubly bonded series of atoms in the order in which they are bonded together.
- N_1 and N_i are the highest priority atoms bonded to N_2 and N_{i-1} respectively, according to the CIP rules.
- G is a character used to designate the overall geometry. If there is an even number of atoms specified then all the atoms are coplanar with N_1 and N_i lying on either the same, G='Z', or opposite, G='E', sides of the doubly bonded atoms. If there is an odd number of atoms then N_1 and N_i lie roughly perpendicular to each other. If on moving from one end of the series of doubly bonded atoms to the other the highest priority atom lies counter clockwise then G='P'. If they lie clockwise then G='M'.

In problematic cases, C or G appear as "???" if the structure was distorted enough to prevent unambiguous designation of the stereochemistry.

The atom property NoSrcChirality (`b_st_NoSrcChirality`) is added and set to `true` if the atom is chiral but the chirality was not specified on input.

4.10.2 Known Limitations

The `stereoizer` is a fairly strict implementation of the standard CIP rules for chiralities and double-bond geometries as described in Prelog, V.; Helmchen, G. Basic Principles of the CIP-System and Proposals for a Revision. *Angew. Chem., Int. Ed. Engl.* **1982,** *21,* 567-583, except that prioritization of atoms by chiralities is not implemented. For instance, for prioritizing groups attached to the terminal atom of double bonded systems of atoms, if the groups are otherwise equivalent the CIP rules indicate that they may be prioritized if they differ in the chiralities of atoms within them. The `stereoizer` does not do this.

In stereoisomer mode, the stereoizer avoids some combinations of chiralities that are not achievable for geometric reasons or are atypical for some types of natural products (e.g. peptides and steroids). The coverage is not exhaustive, and for functionalities outside those covered the `stereoizer` does not attempt to distinguish realistic from unrealistic chiral atoms

combinations. The unrealistic combinations can be largely eliminated in the `bmin` stage of LigPrep processing.

4.11 ring_conf

It can be difficult, or at least computationally demanding, to find low-energy conformations for the flexible ring systems within a structure. The `ring_conf` tool attempts to do so by identifying ring systems in the structure, then seeking suitable ring templates from its collection, and using these templates to generate good ring conformations.

Briefly, `ring_conf` examines the molecule to identify groupings of flexible rings, referred to as ring systems, that require a template. Many aromatic ring systems are recognized as such and no template will be sought for them and the input geometry will be used. However, aromatic rings within ring systems containing flexible rings need special attention. If they share 2 or more atoms with a flexible ring then they influence the conformation of the flexible ring and thus must be considered part of the ring system for template matching. However, if they do not share 2 or more atoms with a flexible ring they are not included in ring system that must be matched by a template. In this way, ring systems in which flexible rings are connected by non-flexible rings that do not overlap with flexible rings may be split into separate ring systems for templating purposes. Once the process of identifying portions of the molecule that require ring system templates are found, templates are sought separately for each such portion. Ring systems for which templates cannot be found retain the input geometry for the ring system. Template matching has some flexibility in that enantiomers may be matched and protonation states of sp^2 or sp^3 nitrogen atoms need not match. Also, amide bonds in rings are matched by C=C if a specific template is not found. The template library has more than 770 templates, many of which have multiple conformations. Templates in the library include inversions of non-quaternary sp^3 nitrogen atoms in their collections of conformations.

After the template identification for the ring systems is complete, the conformational energies that depend on the ring conformation are estimated in a crude manner. The energy of each ring system is estimated separately based upon the ring conformation energies recorded in the template database, the axial-equatorial energies of the attachments and a pair-wise short range repulsion between base atoms for each attachment (the atoms directly bonded to atoms in the rings). The total ring system energy for a particular selection of conformations for all ring systems in a structure is obtained by adding together the energies for these conformations in each ring system.

By default, only the structure with the lowest estimated energy for each input structure is written to the output structure file. However, multiple structures with different ring conformations can be saved (options `-e`, `-n`, and `-N`) using different selection criteria (option `-c`).

While ring_conf is focused on providing conformations for the flexible ring systems in the structure provided, it can also attempt to adjust the conformations of portions of the structures outside such systems, depending on the linear generation method (see the -l option below). We generally recommend using -l 3, which instructs ring_conf to adjust the linear portions of the molecules so that they adopt extended conformations.

The syntax is as follows.

```
$SCHRODINGER/utilities/ring_conf [options] input-file.mae
output-file.mae
```

You must have a LigPrep license to run ring_conf.

Additional output files:

*jobname*_rcout_tmpl_att_prob.mae	Structures for which either templates were missing or the relative energies of axial vs. equatorial geometries for attachments were missing.
*jobname*_rcout_misc_prob.mae	Structures for which ring_conf had other types of problems.

Limitations:

Missing Templates: The collection of templates for flexible ring systems is incomplete. In our trials, typically ring_conf did not find templates for all ring systems present for a small percentage of the valid input structures. However, we expect that the rate of misses will vary dramatically. Flexible ring systems for which templates are not found are simply treated by ring_conf as though they were rigid. At present, ring_conf does not support the ability to directly add templates to the template library; however, you can specify a custom location for additional templates, as described below.

Structures for which templates were not found are written to a Maestro file named *jobname*_rcout_tmpl_att_prob.maegz. This file is copied back to the working directory, or is included in the archive that is copied back to the working directory when running LigPrep. You can use this file to generate templates with the ring_templating utility—see Section 17.8 of the *MacroModel User Manual* for more information. When these templates are placed in your Schrödinger user resource directory, the distribution, or the location specified by the environment variable SCHRODINGER_RING_TEMPLATE_DIR, they are used by ring_conf.

The Maestro property i_lp_ring_sampling_problem is added to the structures, to indicate the cause of failure. A value of 1 means that templates were not found for any of the ring

systems. A value of 2 indicates that templates were found for some ring systems but not others, while 3 indicates an unclassified problem.

Attachment Geometries for Missing Templates: Under some circumstances, attachments to flexible ring systems for which templates were not found are placed on top of each other if a value of 2 or 3 is given for the `-l` option. However, these geometries usually do not cause problems in subsequent processing by `premin` and `bmin` within LigPrep protocol.

Misidentification of Smallest Set of Smallest Rings: In very rare instances, the software that identifies the rings within ring systems, particularly fused ring systems, omits some rings. In such cases, it may not be possible to find a suitable template even if one is present.

Overlapping Atoms: `ring_conf` generates low-energy ring conformations using criteria that depend on features that are local to individual ring systems. The relative positions of portions of a molecule that are not directly associated with a single ring system are not taken into account, and in some instances may lie too close to each other. Within the LigPrep context the follow-up minimization and optional conformational search using MacroModel are expected to eliminate nearly all such problems.

4.12 premin

`premin` is used in filter mode in LigPrep to eliminate structures that could cause severe problems when processed by Schrödinger applications. Ligands that are processed successfully are written to the output structure file, which by default is called *jobname*`-min.mae`, while problematic structures are stored in *jobname*`-bad.mae`. In the LigPrep process, these files are named *jobname*`_prout.mae` and *jobname*`_prout_bad.mae`.

When `premin` is run using a LigPrep license, one MacroModel command file is created, named `filter.com`. Limited customization of the filtering process is possible with a modified `filter.com` file if `premin` is run directly from the command line and the `-custom` option is specified. When `premin` is run using a LigPrep license, only a restricted subset of the internal MacroModel commands (*opcodes*) is supported. The following section lists these opcodes.

The syntax is as follows.

```
$SCHRODINGER/utilities/premin [options] input.mae
```

Here, *input*`.mae` is the input file containing ligand structures.

4.13 MacroModel (bmin)

LigPrep uses MacroModel to enforce the chiral properties listed in each structure while producing an energy-minimized structure. When run under a LigPrep license, MacroModel has a very limited set of commands enabled. These commands, or *opcodes*, are targeted specifically for LigPrep use.

The `CHIP` opcode (see below) is used to instruct MacroModel that this calculation is being performed to assist LigPrep. The overall process is formally a serial MCMM search. Each structure is processed separately and sequentially from the input structure file. The positions of hydrogen atoms attached to chiral atoms are adjusted if they are not originally in an appropriate position.

First, the structure is read in and a small random displacement is made to separately to each atom present. Up to 4 improper torsions are placed on each atom indicated as chiral by a chiral property in the structure to strongly encourage the atom to adopt the chirality specified. This structure is then energy minimized. Then the improper torsional potentials are removed and a second minimization conducted. If the resulting structure has bonds close to their equilibrium length the conformational search is skipped. Otherwise a short conformational search is conducted and the lowest energy structure from the search retained. The same two stage minimization with and without the additional improper torsions is used for each conformation generated during the search. This optional search is used to attempt to correct distorted structures, such as those with connected atoms passing though the central portions of small rings. The lowest energy structure from either the full or truncated search is then checked to see if the requested chiralities were obtained, all bonds lie within 0.25 Å of their ideal length, and all bond angles are greater than 10 degrees. If the structure passes all of these tests then it is added to the output structure file. If not, by default, it is rejected and no structure is recorded in the output structure file. This strategy is used to eliminate unrealistic combination of chiralities, particularly in fused ring systems, which may be generated by the stereoizer.

See the *MacroModel Reference Manual* for `bmin` syntax.

4.13.1 MacroModel Opcodes Enabled for LigPrep

The following opcodes are enabled under a LigPrep license: `READ`, `MINI`, `FFLD`, `BGIN`, `END`, `CONV`, `EXNB`, `MCOP`, `DEBG`, `NANT`, `MCMM`, `MCSS`, `MCNV`, `SOLV`, `BDCO`, `MSYM`, `SPAT`, and `AUTO`. See the *MacroModel Reference Manual* for details. In addition, the `CHIP` and `CHOP` opcodes, described below, are available only with a LigPrep license.

CHIP — CHIrality Protocol

CHIP instructs MacroModel to adjust its behavior appropriately for enforcing chiralities and avoiding common pitfalls encountered for structures that were originally 2D. CHIP is intended to be used within a .com file that would otherwise be a fairly typical MCMM serial search.

CHIP modifies the behavior of the READ, MCMM and MINI opcodes. When CHIP has been specified, structures read in with the READ opcode are examined for chiral atom properties. For each chiral atom specified up to 4 improper torsional potentials are added to encourage the atom to adopt the chirality requested. In addition, each atom in the structure is displaced by a small random displacement to encourage the system to leave saddle points during minimizations. Sometimes hydrogen atoms are placed in positions inconsistent with the chirality of atoms that they are bonded to. When this problem is detected the hydrogen is repositioned to facilitate minimization to the correct geometry.

The behavior of the MCMM opcode is tightly coupled with the MINI opcode. In a serial search, when CHIP has been specified, the search proceeds as follows:

1. The structure is read using READ, whose behavior is modified as described above.
2. An initial minimization is performed on this structure with the additional improper torsional potentials in effect.
3. A second minimization is performed with the additional improper torsional potentials turned off.
4. If the bond lengths deviate from their ideal values by less than a specified value no search is performed and the next step (5) is skipped. Significantly distorted bonds are a good indicator that a portion of the molecule is passing through the center of a small ring.
5. A short MCMM search is performed. Each structure generated is minimized using the same two-stage procedure described in steps 2 and 3.
6. The structure is checked to see if it has adopted the requested chiralities. If so it is saved to the output structure file. If not, by default the structure is not recorded in the output structure file.

arg1 *Chirality enforcement*

−1 Turn on the requirement for LigPrep licensing, but do not turn on chirality enforcement behavior.

0 Turn on the requirement for LigPrep licensing, and turn on chirality enforcement behavior.

arg2 *Recording structures with incorrect chiralities*

−1 Write structures whose chiralities do not match those requested in the output structure file along with structures whose chiralities do match. These structures have the property `b_mmod_Chiralities_Consistent-`*Force_field_name* set to 0 (for FALSE).

0,1 Do not write structures whose chiralities do not match those requested in the output structure file.

arg3 *Search control for strained structures*

Strained structures are defined as those structures with bond lengths differing from the equilibrium bond length by more than the value specified by arg7.

−1 Never perform searches on strained structures.

0,1 Perform an MCMM search on strained structures.

arg4 *Random displacement control*

Displacing atoms by small random amounts is useful in shifting minimizations away from saddle points, a problem that often arises in structures that were originally 2D. The maximum displacement is controlled by arg6.

−1 Do not randomly displace each atom after a new structure has been read in.

0,1 Randomly displace each atom after a new structure has been read in.

arg5 *Chirality enforcement force constant in kJ/mol*

Must be non-negative. The size of the force constant for the improper torsions. Equivalent to arg5 of FXTA. Default value: 1000.0 kJ/mol

arg6 *Maximum size of random displacement in Å*

Must be non-negative. The maximum size of the random change in angstroms. introduced into each of the X,Y, and Z coordinates of each atom in the structure. The random change is chosen with a uniform probability on the range indicated. Default value: 0.05 Å.

arg7 *Extent of bond distortion in Å that triggers a search*

Must be 0 or greater than 0.05. If any bond is distorted away from the equilibrium bond length by more than this amount after the initial minimization then a full con-

formational search is to be performed provided that arg3 is not 1. Default value: 0.25 Å.

arg8 *CHIP verbosity*

0 Normal level of reporting to the `.log` file.

1 Turn on verbose reporting of `CHIP` processing to the `.log` file.

CHOP— CHirality OPtions

The `CHOP` opcode allows you to change some of the behavior associated with the `CHIP` opcode in MacroModel calculations.

arg1 *Saving problematic structures*

Structures that do not have the correct chiralities or are severely distorted are not placed in the output structure file for the `bmin` stage of `ligprep`. This option controls wether these such structures are saved to an auxiliary output file, *jobname*`_bmout-bad.mae`.

0 Do not save problematic structures

!=0 Save problematic structures

arg2 *Scaling factor for conformational search steps if ring_conf could not find a template*

Scale the number of steps in the conformational search if a template could not be found by `ring_conf`. The number of steps is set by arg1 of `MCMM`.

>0 Scale the number of conformational search steps by this value. The default scaling factor is 2.

arg5 *Force constant for the torsional constraints that are used to enforce chiralities*

< 0 Do not use torsional constraints

0 Use the default value of 500.0 kJ/mol

> 0 Use this value (kJ/mol) for the force constant.

arg6 *Maximum average stretch energy for acceptable molecules*

Conformations of molecules containing more than 9 bonds with stretch energies greater than this value are eliminated from consideration.

0 use a value of 9 kJ/mol

< 0 turn off elimination of molecules by average stretch energy

> 0 use this value (in units of kJ/mol)

arg7 *Maximum deviation of a bond angle involving hydrogen*

Conformations in which a bond angle involving at least one hydrogen atom deviates from the ideal bond angle given by the force field by more than this value are eliminated.

0 use a value of 25 degrees

< 0 turn off elimination of conformations by this mechanism

> 0 use this value (in units of degrees)

4.13.2 The LigPrep Command File for MacroModel

`ligprep` creates the `.com` file below for the MacroModel calculations used in the LigPrep process. The overall form of the file is fairly specific for the LigPrep process but follows that for an MCMM serial search.

jobname`_prout.mae`
jobname`_bmout.mae`

```
 DEBG      38      0      0      0     0.0000     0.0000     0.0000     0.0000
 DEBG      51      0      0      0     0.0000     0.0000     0.0000     0.0000
 FFLD      14      2      0      0     4.0000     0.0000     0.0000     0.0000
 BDCO       0      0      0      0    41.5692 99999.0000     0.0000     0.0000
 CHIP       1      0      0      0     0.0000     0.0000     0.0000     0.0000
 CHOP       1      0      0      0     0.0000     0.0000     0.0000     0.0000
 NANT
 READ       0      0      0      0     0.0000     0.0000     0.0000     0.0000
 MCMM      20      0      0      0     0.0000     0.0000     0.0000     0.0000
 MCNV       1     10      0      0     0.0000     0.0000     0.0000     0.0000
 MCSS       2      0      0      0  1000000.0     0.0000     0.0000     0.0000
 MCOP       1      0     20      0     0.0000     1.0000     0.0000     0.0000
 MSYM       0      0      0      0     0.0000     0.0000     0.0000     0.0000
 AUTO                   -1      2               1.0000
 CONV       2      0      0      0     0.0500     0.0000     0.0000     0.0000
 MINI      -9      0    500      0     0.0000     0.0000     0.0000     0.0000
```

4.13.3 MacroModel Limitations and Workarounds

AUTOmatic setup and ring closure failures

Automatic setup for the conformational searches sometimes fails to identify all of the ring closure bonds in a conformational search. When this occurs in a serial search MacroModel abandons the current search and the structure is not included in the output structure file. Processing then proceeds from the next structure in the input structure file, if present.

When the specified chiralities are not obtained the structures may be distorted

The strong improper torsions applied to chiral atoms aggressively try to force the molecules to attain the specified chiralities. If this process fails to produce the correct chiralities the resulting structures may be quite distorted. Such failures usually are the result of physically inconsistent chiralities. By default, these structures are retained, so distorted structures might be written to the output structure file. There is a `ligprep` option (`-br`) that you can use to discard structures that fail to attain the specified chiralities.

Ligprep stopping during bmin processing

`bmin` processing consumes more than 75% of the CPU time required to process a collection of ligands. If the LigPrep process stops while `bmin` is processing ligands, the output structure file for `ligprep` can be empty or contain only one ligand. However, the file *jobname*`_bmout.ouL` should contain essentially all of the ligands successfully processed up to the stage at which `bmin` stopped.

The ionizer Utility

Under physiological conditions, a molecule with acidic or basic groups can exist in a variety of protonation (ionization) states. Ligand structures are usually provided in only one of these states. To ensure that the important ionization states are sampled, LigPrep includes the `ionizer` utility to generate the states that are significantly populated under a prescribed set of conditions.

The Ionizer, or ionization state expander, produces multiple structures for each input structure with different combinations of ionized states based on the ionizable groups present. The `ionizer` makes use of a database of substructure patterns (templates) for ionizable groups, which it matches to the input structure to identify candidates for protonation or deprotonation. If a template for the ionizable group is not found, the group is not ionized.

LigPrep sets only a few of the `ionizer` options. For information on these options, see Section 4.7 on page 45. You can pass a restricted set of `ionizer` options to the `ionizer` with the `ligprep -W i` option. These options are indicated in the option descriptions below.

This chapter describes the use of the `ionizer` as a stand-alone utility.

5.1 Running the ionizer

To run the `ionizer` from the command line, use the following syntax:

```
$SCHRODINGER/utilities/ionizer [options]
```

For a description of the options, run the command with the `-h` option.

5.2 Input Structure Requirements

For the `ionizer` to process all input structures successfully, each structure must satisfy the following requirements:

- Hydrogens must be added.
- Atom types must be correct.
- There must be no formal charge on atoms matched as centers of ionizable groups.

It is not necessary for input structures to be minimized fully, as the Ionizer works in terms of atom types and connectivity. However, bond lengths and bond angles should have reasonable

values: unreasonable structures can cause processing failures. Coincident atomic coordinates can cause serious problems, especially if atoms coincide within an ionizable group.

By default, structures that the Ionizer cannot process are passed unchanged to the output file. It also records these structures in the file *jobname*`-ion-bad.mae` (or the file specified by the `-b` option). If you set the `-strict` option, the `ionizer` stops if it encounters a structure that it cannot process.

5.3 Restricting Ionization State Generation

The Ionizer has two methods for restricting the generation of ion states to ensure that each state has a significant population under the chosen conditions. The default method is to screen values according to a pH range. This is done by specifying a target pH value with the `-ph` option and a threshold value with the `-pht` option. The default pH is 7, and the default threshold is 2. For any ionizable group whose pK_a lies within the threshold value around the target pH value ($|\ pK_a - pH\ | \leq$ threshold), both protonated and deprotonated forms are generated. If the pK_a is smaller than the target pH value by more than the threshold value, only the deprotonated form is generated, and if the pK_a is larger than the target pH value by more than the threshold value, only the protonated form is generated. To generate all possible ionization states, you can set the threshold to a large value.

The alternative method is to use the `-pkt` option to specify a relative threshold for the pK_a values to screen out improbable states. In this method, a structure generated by the `ionizer` is rejected if the maximum pK_a value of any deprotonated group is greater than the minimum pK_a value of any protonated group by the specified threshold.

For example, if the threshold is 3.0 and the structure has two ionizable groups with pK_a values of 6.0 and 10.0, the structure with the first group protonated and the second group deprotonated is rejected. Likewise, if the threshold is 2.0 and the structure has 4 ionizable groups, with pK_a values of 4.0, 5.5, 7.0, and 8.5, the structure with the first and fourth groups deprotonated and the second and third groups protonated is rejected. In this case the maximum pK_a of the deprotonated groups is 8.5 and the minimum pK_a of the protonated groups is 5.5. The difference is greater than the threshold value of 2.0.

This method treats each structure independently and ensures that the totally protonated form and the totally deprotonated form are always included in the output. There is no default value for the threshold, and as for the pH-based method, it is possible to specify a pK_a threshold large enough so that no ion state is rejected.

If you specify the pK_a threshold, you can also specify a maximum number of groups ionized and a maximum total charge, to limit the number of structures generated. Setting the `-mi` option limits the maximum number of ionized groups to the value specified. The default is 4. If

a possible ion state has more individual groups ionized than this value, the state is rejected. This limit applies only to the ionizable groups detected by the Ionizer. Setting the `-mq` option limits the absolute value of the total charge to a value less than or equal to the value specified. The default is 2. If the structure already has a formal charge, this charge is taken into account when the restriction is applied.

5.4 Structural Output

If an input structure contains a large number of ionizable groups, the number of output structures will be very large (2^N combinations) and the `ionizer` will take a long time to run. The `ionizer` has a limit on the number of ionizable groups in any structure that can be set, above which the `ionizer` skips its normal analysis and expansion of the structure and passes it unchanged into the output file. To set this limit, use the `-mg` option. The default value is 15 for standalone use, and 10 for use within LigPrep. You cannot set the limit higher than 31. An input structure that contains close to 31 ionizable groups could take hours or days to be treated by the `ionizer`. You should therefore set this limit well below 31 if you do not want long run times.

In addition, structures that cannot be processed are passed unchanged into the output file and are recorded in a separate file, named *jobname*`-ion-bad.mae` by default. The name of this file can be set with the `-b` option.

5.5 Log File Output

As the Ionizer expands input structures into ionized output structures, it records various status messages in the log file, which by default is named *jobname*`.log`. You can specify the log file name with the `-l` option. If you want to direct the log file to your screen, set the log file name to a hyphen: `-l -`. Examining the log file output can be useful if you want to determine how a particular structure was handled.

Substructure matching: If you use the `-sm` option of the `ionizer` command, the substructure pattern matches are logged on each input structure as it is processed. The information logged for each match includes the line number in the pattern file where the substructure pattern is given, the pattern itself, the ion center atom number where the match is anchored, the acid/base character of the ionizable group, and a comment as to whether the match modifies an earlier match at the same atom.

Final ionization candidate list: If you use the `-sf` option of the `ionizer` command, the final list of ion centers identified on each input structure is logged as it is processed. The information logged for each ion center includes the ion center atom number, the acid/base character of the group, and the pK_a value associated with this group.

Log level: The `-ll` option gives general control over the amount of information provided in the log file. Each level adds information to the previous level.

At level 0, the report on each input structure processed includes the structure number (in the input file), structure title, the number of distinct ionizable groups identified, the number of ionization states to consider generating, and the number of states actually generated after imposition of restrictions.

At level 1, the report contains a line on each ion state generated, showing the state's index and a brief description of the state, listing the original atom numbers in the input molecule which become ionized in the output.

At level 2, the report provides more details which may be helpful in debugging in conjunction with Schrödinger Technical Support, and increases the run time slightly.

For production runs, you should use level 0 if possible and level 1 only if you need to see a list of the ion states generated or skipped.

Skipped states: If you use the `-ss` option of the `ionizer` command, a line is written for each ion state skipped—that is, rejected from generation due to imposition of some restriction. This option affects only log levels greater than 0. If this option is specified when running at level 2, the reasons for rejecting an ion state are shown. You might want to see these reasons spelled out, for example, when adjusting the restriction parameters to see which rejections they lead to in practice.

If structures are skipped because the structure itself is faulty, error messages are printed to the log file. In the default fault-tolerant mode, each set of error messages resulting from input structure problems is followed by a message indicating that the otherwise fatal errors have been tolerated, and the structure passed to the output file unchanged.

5.6 Maestro Properties

When the ionization state changes, most if not all of the properties of the structure are no longer valid. By default, the Ionizer does not copy the properties of the input structures to the output. In some cases, the properties of the parent structure are useful for later screening or identification—for example, QikProp properties. If you want to ensure that properties are inherited by the output structures, you can add the `-kp` option to the command line.

The Ionizer generates some Maestro properties, which are summarized in Table 5.1.

Table 5.1. Maestro properties generated by the Ionizer.

Property Name in Project Table	Name in Maestro Output File	Description
Ion state n	`i_ionizer_Ion_state_n`	Ionization state number. This is an index running from 0 to the number of ionized states, and serves as a unique identifier.
Tot Q	`i_ionizer_Tot_Q`	Total formal charge. This value includes any formal charges in the input structure that have been retained.
Ion centers	`s_ionizer_Ion_centers`	String of atom numbers for new charge centers generated by the `ionizer`.[a]
Ion types	`s_ionizer_Ion_types`	String of MacroModel atom type symbols for new charge centers generated by the `ionizer`.[a]
Ion ctrs in	`s_ionizer_Ion_ctrs_in`	String of atom numbers of the original uncharged atoms in the input structure that correspond to new charge centers in this output structure.
Ion passthru	`s_ionizer_Ion_passthru`	Brief description of the reason an input structure was passed to the output file unchanged. Generated only if an input structure is not processed.
Ionization penalty	`r_ionizer_Ionization_penalty`	This ionization state's total energy penalty[b]
Ionization penalty charging	`r_ionizer_Ionization_penalty_charging`	This ionization state's energy penalty due to charged sites[b]
Ionization penalty neutral	`r_ionizer_Ionization_penalty_neutral`	This ionization state's energy penalty due to neutral sites[b]

a. The strings are empty for the non-ionized state. Neither of these strings is a unique id for the ionization state. It is not uncommon for these strings to match between different ionized variants of the same input.

b. The energy penalties are expressed in kcal/mol units. These properties are not written if running in pK-threshold mode. Details of the calculation are presented below.

Calculation of Ionization State Energy Penalties

For each ionizable group involved in a generated state, we define two energy costs:

Yn: The energy cost of ionizing the group

Zn: The energy cost of keeping the group neutral

Consistent with the Ionizer's other simplified assumptions, we do not account for cooperative effects between multiple groups.

At the given pH, with the group's specified pK_a, if the group is a base, we calculate:

$Yn = RT \ln(10^{pH-pKa} + 1)$

$Zn = RT \ln(10^{pKa-pH} + 1)$

where R = 0.001987207 kcal/(mol K), and T = 298.15 K.

If the group is an acid, we calculate the alternate pair:

$Yn = RT \ln(10^{pKa-pH} + 1)$

$Zn = RT \ln(10^{pH-pKa} + 1)$

Now, for each generated ionization state combination, we sum the Yn for all groups that are ionized, calling the total Y, and we sum the Zn for all groups that are not ionized, calling the total Z.

For small values of the pH threshold (controlled by `-pht`), Y is dominated by the energy cost of ionizing weak acids and bases, while Z is dominated by the energy cost of keeping weak acids or bases neutral. For larger values of the pH threshold, these penalties begin to include the corrections for producing stronger acids and bases in atypical forms.

We also compute P = Y + Z, which is the total energy cost of the ionization state combination.

The P, Y, and Z values are expressed in the energy penalty properties for the output structure:

`r_ionizer_Ionization_penalty`

`r_ionizer_Ionization_penalty_charging`

`r_ionizer_Ionization_penalty_neutral`

5.7 Creating a Customized Patterns File

The file that contains the substructure patterns used to identify ionizable groups can be customized. This file contains substructure patterns for protonating N atoms, and deprotonating NH, OH, and SH groups. The standard patterns file can be found at

```
$SCHRODINGER/mmshare-vversion/data/ionizer.ini
```

If you want to add your own patterns, you should make a copy of the file and edit it. Documentation on the pattern specification commands can be found in comments in this file. The pattern specification commands require use of Schrödinger's linear substructure notation, which is documented in the *Maestro User Manual*.

Once you have made your changes, you can then use the modified file as input to the `ionizer` in the following ways:

- By providing its name with the `-s` option when you run the `ionizer`, or the `-is` option when you run `ligprep`.
- By copying this file to the directory from which you launch an `ionizer` or `ligprep` job. The file must be named `ionizer.ini`.
- By adding this file to your `$HOME/.schrodinger/mmshare` directory. The file must be named `ionizer.ini`, and becomes your global default patterns file.

Note: Correct use of the syntax is not trivial. You might have to experiment for some time to ensure that you have the correct result, and you may need to contact Schrödinger for assistance.

Warning: Do not change the standard patterns file.

5.8 Ionizer Limitations

The limitations described in this section apply only to the Ionizer operation, if used in the LigPrep process, or if used on a stand-alone basis. Limitations of the LigPrep process as a whole are described in Section 2.6 on page 24.

- Local Focus in Adjustment of Protonation States

  The Ionizer adjusts the protonation state of each ionizable group without regard for the protonation state of other ionizable groups found in the molecule. In other words, when deciding which states of one ionizable group to generate, the Ionizer does not take into account changes in the group's effective pK_a due to neighboring protonation states. This can lead to generation of the wrong ionization states for molecules containing more than one ionizable group.

For the specific case of an aliphatic diamine separated by two carbons (for example, piperazines and ethylene diamines) patterns have been introduced in which the pK_a of both nitrogens is given as the average of the two experimental values for the mono and diprotonated species. As a result, when the ionizer is run with the default settings (pH 7 ± 2), four forms are generated: neutral, both monoprotonated forms and the diprotonated species.

Note: Only the monoprotonated forms are favorable near pH 7. Without these patterns, only the diprotonated form would be generated. The ordering of the penalties for this class of system is not accurate. This behavior can be disabled by commenting out the diamine lines in the `ionizer.ini` file.

The erroneously excluded states can be generated by running the Ionizer with non-default pH and/or pH threshold settings, but it can take some trial and error. Of course, running with modified settings to produce previously excluded states of one molecule can generate undesired variants of other input molecules.

- Incompatibility with Certain Force Field Implementations

  The Ionizer may generate structures that cannot be used with certain force field implementations. One known case involves the atom type N5 (N^+ sp^3) and AMBER* (Modified AMBER), which arises when ionized forms are generated outside of "normal" pK_a ranges, particularly the protonation of anilines and aromatic heterocyclic nitrogens. There could also be problems with atom type N4 (N^+ sp^2).

- Rotation of Output Molecules

  In some ionized molecules with protonated nitrogens, the output structure will be rotated with respect to the input structure.

- Upper Limit on Number of Ionizable Groups in Input Molecule

  The program is capable of handling up to 31 ionizable groups in an input molecule (though it would take a long time, and a lot of disk space, to so do). The value 31 is a hard upper limit. This is the maximum value allowed to be specified with the command-line option `-mg` or `-maxgroups`.

The tautomerizer Utility

Tautomers are an important class of isomers that can interconvert under physiological conditions. Tautomeric forms have different chemical properties and interact differently. For instance, one tautomeric form may interact with the active site of a protein more strongly than the other forms. Therefore, for some types of calculations, such as docking with Glide, considering the appropriate tautomeric forms of ligands can be important.

There does not seem to be a universal definition for tautomers. In LigPrep, tautomers are defined as isomers that meet the following conditions:

- In aqueous solution tautomers interconvert rapidly enough to be present as mixtures.
- One or more hydrogen atoms are bound to different atoms and the orders of one or more of the bonds between non-hydrogen atoms differs between tautomers.
- The non-hydrogen atom topology of the structure does not change during these interconversions.

LigPrep's definition of tautomers therefore excludes the aldose-hemiacetal ring opening and closing equilibrium in sugars that are sometimes regarded as tautomerizations. An example of the well-known keto-enol tautomerization is shown in Figure 6.1.

The `tautomerizer` relies on a database of tautomeric templates to guide it in generating tautomers. For more information on this database, see Section 6.2 on page 70.

The `tautomerizer` is not intended to generate all possible tautomers. The collection of groups of tautomers in the database is not exhaustive. As well, each set of tautomeric forms in the database is limited to those tautomers that are expected to have significant populations in aqueous solution, rather than being a comprehensive list. Tautomers in the database are assigned probabilities to assist in focusing on the most highly populated tautomeric forms.

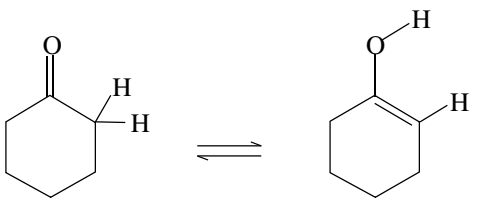

Figure 6.1. Keto-enol tautomerization.

Some types of functional groups such as amino groups strongly perturb tautomeric equilibria. The presence of such groups severely reduces the value of the probabilities recorded in the database which could lead to inappropriate tautomers being generated. To avoid this problem, most patterns exclude the possibility of attachment to such functional groups. In selected cases, new patterns have been added that explicitly include such functionalities in the tautomeric pattern and in the calculation of the probabilities.

The `tautomerizer` provides control over tautomer generation through the command-line options and by permitting users to modify the database.

6.1 Running the tautomerizer

To run the `tautomerizer` from the command line, use the syntax below. The `tautomerizer` is part of the LigPrep collection of programs and requires a LigPrep license.

```
$SCHRODINGER/utilities/tautomerizer [options] input_file.mae
    output_file.mae
```

For a description of the options, run the command with the `-h` option.

The `tautomerizer` can also be run as part of the LigPrep process. When LigPrep is run from the LigPrep panel in Maestro, the default options are used. When it is run from the command line, the defaults can be overridden by options to the `ligprep` command. See Section 2.2 on page 16 and Section 4.8 on page 46 for more information.

6.2 The Tautomer Database

In LigPrep 2.0 the default tautomer database is not accessible to users. However, you can provide your own file to either completely override the default file, or add patterns to it.

Note: The database file format has changed in LigPrep 2.0. The current format will not work with previous versions of the program although earlier versions of the data file may work with LigPrep 2.0, with some modifications.

At the top level of the tautomer database file the following four items can be present: `name`, `clear_standard`, `group_def`, and `tautomer_set`. These items are described in the following sections. Lines beginning with a # are comment lines and are ignored when interpreting the contents of the tautomer database file. Blank lines are also ignored.

6.2.1 name Data Item

`name` specifies the name of the solvent. For example:

```
name: water
```

`water` and `DMSO` are standard names for which the tautomerizer already has information. Currently, the DMSO tautomer information is just a duplicate of that for water.

6.2.2 clear_standard Directive

By default, information in a custom tautomer database file is added to any existing information already available for the solvent specified. Including `clear_standard:` in a tautomer database file clears any values for this solvent accumulated before the current file was read.

6.2.3 group_def Data Structure

The tautomerizer does not support recursive SMARTS. However, a mechanism that supports some of the functionality of recursive SMARTS is provided by the `group_def` data structure. This data structure permits you to define variables that correspond to SMARTS patterns. The variables may be reused in groups and `tautomer_sets` that appear later in the tautomer database file.

Each group contains two items:

`name:` an arbitrary name for the group which is used to reference the group.

`pattern:` The SMARTS pattern for the group. This pattern may refer to previously defined groups using $*groupname*.

Below are some examples of `group_def` data structures:

```
group_def{
        name: Halogens
        pattern: [F,Cl,Br,I]
}

group_def{
        name: Amides
        pattern: [CX3](=[OX1])-[NX3]
}

group_def{
        name: Carbonyls
        pattern: [CX3](=[OX1])
```

```
}

group_def{
        name: Carbonyls_only
        pattern: [$Carbonyls;!$Amides]
}
```

6.2.4 tautomer_set Data Structure

`tautomer_set` data structures define sets of interconvertible tautomers. There are more than 215 tautomer sets available by default for water.

Some examples of `tautomer_set` data structures are given below, and the syntax for the data structures is described following these examples.

Note: The entry for `pattern:` values must be a single line. In the examples below, some of the `pattern:` text wraps to the next line due to formatting constraints within this manual. When creating tautomer data structure files in a text editor, ensure that text-wrapping is turned off, or that margins are set wide enough to accommodate single-line entry for this value.

```
tautomer_set{
    name: single-sided_ket-enol
# From: Handbook of organic chemistry

    tautomer{
      name: enol
      pattern: [CX3](-[#1,$Sub_aC])(-[#1,$Sub_aC])=[CX3](-
[#1,$Sub_carbonyl_C])-[OX2]-[#1]
      probability: 0.00005
    }

    tautomer{
      name: ket
      pattern: [CX4](-[#1])(-[#1,$Sub_aC])(-[#1,$Sub_aC])-[CX3](-
[#1,$Sub_carbonyl_C])=[OX1]
      probability: 0.99995
    }
}

tautomer_set{
    name: double-sided_ket-enol
# From: Handbook of organic chemistry
```

```
    tautomer{
      name: 1enol
      pattern: [CX3](-[#1,$Sub_aC])(-[#1,$Sub_aC])=[CX3](-[CX4](-
[#1])(-[#1,$Sub_aC])(-[#1,$Sub_aC]))-[OX2]-[#1]
      probability: 0.00000001
    }

    tautomer{
      name: ket
      pattern: [CX4](-[#1,$Sub_aC])(-[#1,$Sub_aC])(-[#1])-[CX3](-
[CX4](-[#1])(-[#1,$Sub_aC])(-[#1,$Sub_aC]))=[OX1]
      probability: .99999998
    }

    tautomer{
      name: 2enol
      pattern: [CX4](-[#1,$Sub_aC])(-[#1,$Sub_aC])(-[#1])-
[CX3](=[CX3](-[#1,$Sub_aC])(-[#1,$Sub_aC]))-[OX2]-[#1]
      probability: 0.00000001
    }
}
tautomer_set{
    name: imidazole

    tautomer{
      name: form1
      pattern: c1(~[#1,$Sub_c])n(-[#1,$Sub_n])-c(-
[#1,$Sub_c])=[nX2]c1(~[#1,$Sub_c])
      probability: 0.50
    }

    tautomer{
      name: form2
      pattern: c1(~[#1,$Sub_c])[nX2]=c(-[#1,$Sub_c])-n(-
[#1,$Sub_n])c1(~[#1,$Sub_c])
      probability: 0.50
    }
}
```

Each tautomer set contains a `name:` designator and a number of tautomer structures. The `name:` designator is followed by a space and a contiguous non-blank label to identify the class of tautomers described by the set. The label provided does not affect processing. In the examples below, there are three tautomeric sets: `single-sided_enol-ket`, `double-sided_enol-ket`, and `imidazole`.

The `tautomer` structure describes the properties of one tautomeric form. There are three designators that may be used within a `tautomer` structure: `name:`, `probability:`, and `pattern:`.

The `name:` designator provides a label for the tautomer but does not otherwise affect processing.

The `probability:` designator is used to assign a probability or fractional population of this tautomer within this tautomeric set. In many cases, reliable information on the probability of various tautomeric forms is not available and the values entered in the database are simply educated guesses.

The `pattern:` designator is followed by a contiguous SMARTS-like pattern. A difference between this pattern and a normal SMARTS pattern is that instead of using explicit aromatic bonds, explicit single "–" and double "=" bond designators are used to make the corresponding Lewis structures clear. In addition, these patterns may include references to previously defined groups via the $*group_name* mechanism. Information on SMARTS patterns is provided on the web page: http://www.daylight.com/learn[4]. The SMARTS-like pattern is used to detect the corresponding groups of molecules in the input structures and to permit the tautomerizer to understand how the bonding patterns (Lewis structures) differ between tautomers so that they may be interconverted. For heavy atoms that are expected to carry a formal charge it is advisable to include the charge in the SMARTS pattern. To ensure that the SMARTS patterns are properly interpreted by the `tautomerizer`, the following restrictions must be applied:

- The SMARTS patterns for all tautomers within a tautomer set include the same list of non-hydrogen atoms in the same order.
- All SMARTS patterns must explicitly designate the hydrogens that shift positions in any tautomer within a tautomer set with a `-[#1]` pattern.
- All SMARTS patterns within a tautomer set must contain the same number of explicitly designated mobile hydrogen atoms.
- In both non-aromatic and aromatic portions of the SMARTS pattern, bond orders that change between single and double in any tautomer must be explicitly specified in the SMARTS patterns for all tautomers in a tautomer set.
- In portions of molecules that must be represented by aromatic atom types (e.g., `c` and `n`), only changes in the bond orders of bonds involving `n` atoms in the corresponding Lewis structures are supported. If such a bond changes order in any tautomer in a tautomer set, it must be represented as '`:`' in all the tautomers. See the guanosine tautomer set in the example above.

4. Please see the notice regarding third party programs and third party Web sites on the copyright page at the front of this manual.

- Recursive SMARTS are not supported.
- SMARTS patterns within the same tautomer set must all specify the same overall formal charge.

The database provided with this release contains templates for keto-enol tautomers and their sulfur analogues, imine-enamine tautomers, histidine-like tautomers, tautomers of DNA and RNA bases, and a large number of common heteroaromatic rings containing C, S, O, and N.

6.3 Tautomer Processing

Each structure in the input structure file is processed separately. The processing of each structure by the `tautomerizer` can be divided into two stages: tautomer pattern matching and structure generation.

6.3.1 Tautomer Pattern Matching

The current structure is examined for all matches of all tautomeric patterns in the `tautomerizer` database. All matches are cross-examined to see if the atoms are completely contained within another match. If so, the match containing the smaller number of atoms is eliminated. If they are the same size, then the match pattern found first is retained. This is the only case where the order of tautomers within a tautomer set or the order of the tautomer sets within the `tautomer_list` file affects processing.

In the example above, there are two related SMARTS patterns for keto-enol tautomerization. The `single-sided_enol-ket` tautomer set handles carbonyl groups that can tautomerize only on one side, such as an aldehyde with at least one H atom bonded to the carbon atom adjacent to the carbonyl carbon, while the `double-sided_enol-ket` tautomer set handles carbonyl groups that can tautomerize on either side, such as ketones with at least one H atom bonded to each carbon atom adjacent to the carbonyl carbon atom. Thus structures that match the `double-sided_enol-ket` pattern also match at least one `single-sided_enol-ket` pattern. However, the single-sided matches for this carbonyl are eliminated because they are completely contained within the double-sided pattern. This is important since the double-sided probabilities are different, and matching an enol form in the double-sided case leads to the consideration of the other enol form during the structure generation stage, which would not happen for a single-sided match.

6.3.2 Structure Generation

For each match, all tautomers in the tautomer set should be considered. Structures in which more than one match has been found have multiple locations in the structure that can be tautomerized, and all combinations of tautomers that are compatible should be considered.

Double bonds that can shift to single bonds in any tautomer within a set can shift between E and Z forms, provided that other topological constraints do not prevent this shift.

The `tautomerizer` tries to generate all compatible combinations of all tautomers for each match. For each structure having a high enough probability, all combinations of 180 degree rotations about certain double bonds within each tautomeric pattern are generated. Double bonds are rotated if they involve C or N atoms in which both ends are not attached to two H atoms and in which the two atoms in the bond do not reside in the same ring. No adjustment is made to the probability for the tautomeric form for these double bond rotations. The probability for each structure resulting from this process is estimated as the product of the probabilities for all the tautomer templates used to generate it. The probabilities are normalized amongst all the structures generated for a given input structure.

Tautomers are generated in order of decreasing probability. By default, the maximum number of tautomers generated internally is 128. This limit can be adjusted using the `-m` option of the `tautomerizer` command. Of this collection of tautomers, up to eight tautomers are recorded by default in the output file for each input structure. This limit can be adjusted with the `-n` option of the `tautomerizer` command (`-t` option of the `ligprep` command). The most probable tautomer is always recorded. Except for this tautomer, only tautomers with a probability higher than 0.01 are recorded in the output structure file. This threshold value can be adjusted using the `-p` option of the `tautomerizer` command (`-tp` of the `ligprep` command). If either the `-retain` or the `-retain_lab` option is used to keep a tautomeric form of a molecule that would normally have a probability lower than the threshold probability value, the probability for this form is set to the threshold value.

Note: The probability for forms of a tautomerizable group is not adjusted for other functional groups inside the molecule, but outside the pattern itself. This means that these probabilities, and thus the overall probabilities for molecular forms (particularly when multiple tautomeric sites are being adjusted), are approximate and intended as a guide.

A chiral atom in a tautomer can switch its chirality if one of its tautomeric forms involves a double bond to this atom. The `tautomerizer` does not vary the chiralities of such atoms and does not add chirality properties for such atoms. However, the `stereoizer` can generate these variations. See Section 4.10 on page 48 for more information. In the LigPrep process, the `stereoizer` is run after the `tautomerizer` and the chiralities of such atoms are varied. Some LigPrep options that restrict the generation of stereoisomers may prevent this variation—see Section 2.2 on page 16.

The ligparse Utility

The `ligparse` utility is a tool for examining the structural composition of a molecular data set in a Maestro- or MDL SD-formatted file. `ligparse` can use this structural information to select subsets of compounds within the original data set that meet some user-defined criteria.

Note: This utility has been replaced by `ligfilter` in the LigPrep process. It may be removed from the distribution in a future release.

Note: The input files for `ligfilter` and `ligparse` are not compatible.

A.1 Descriptors and Composite Descriptors

The `ligparse` utility analyzes the composition of a molecular structure by counting structural components, which include the usual organic functional groups. These structural components are represented by SMARTS patterns. A single SMARTS pattern for a given structural component is called a *descriptor*. Linear combinations of SMARTS patterns can be generated to represent more sophisticated structural entities, such as a hydrogen bond donor or acceptor, or metal-ligating groups. These sums of descriptors are called *composite descriptors*. A descriptor is defined by the combination of a SMARTS pattern with its label.

`ligparse` checks each molecule in turn against all descriptors. The algorithm is "greedy"—each atom or bond matches all applicable SMARTS patterns rather than only the most sophisticated or the most simple pattern. For instance, a carboxylic acid group in a molecule would match one instance of [#6;X3]~([O;X1])~[O;X1], two instances of [#6;X3]~[O;X1], and two instances of [#8].

The default descriptor list is stored in the file `$SCHRODINGER/macromodel-v`*version*`/data/str_keys.type`. You can copy and modify the `str_keys.type` file and use it in place of the default with the `-s` command-line option. The `str_keys.type` file is a free-format file with column descriptions as given below.

```
15   H   H  0.00  [n;X2;!H]  nosel  high  -   #  Num heteroaromatic n w.outH
(1) (2) (3) (4)-  (5)------  (6)--  (7)- (8)  (9)----------------------------
```

Only columns (5), (6), and (9) are relevant to this discussion; however, data in the format given above must exist for all columns. In this example, column (5) gives the SMARTS pattern associated with the title in column (9), `Num heteroaromatic n w.outH`. For more information on SMARTS patterns, see http://www.daylight.com. Any string may be used as a descriptor label.

The string `nosel` in column 6 is necessary for atom typing to recognize that this SMARTS pattern is to be used by `ligparse`.

A total of 19 composite descriptors are defined by default in the `composite.types` file in `$SCHRODINGER/macromodel-v`*version*`/data`. They are:

Num rings	Num heteroaromatic rings
Num aromatic rings	Num aliphatic rings
Num rotatable bonds	Num atoms
Molecular weight	Num chiral centers
Num neutral amines	Num amide hydrogens
Num divalent oxygen atoms	Num neutral donor groups
Num charged donor groups	Num neutral acceptor groups
Num charged acceptor groups	Num reactive groups
Num acidic hydrogens	Num donor groups
Num acceptor groups	

Of these composite descriptors, the first eight (Num rings, Num heteroaromatic rings, Num aromatic rings, Num aliphatic rings, Num rotatable bonds, Num atoms, Molecular weight, and Num chiral centers) cannot be adjusted by the user. Other descriptors may be added to the file, and descriptors may be removed from the file. You can specify your own `composite.types` file with the `-c` command-line option. The composite descriptors file is in free format, with a single composite descriptor represented as follows:

```
Num neutral donor groups
    +  Num H-N
    +  Num H-O
    -  Charged amines
    -  Charged imines
    -  Charged pyridine
    -  Charged N-C=N on -N
    -  Charged imidazole
    -  Charged HN-imidazole
```

Here `Num neutral donor groups` is a linear combination of the eight descriptors represented by their labels. The + or – is required and indicates whether the descriptor is to be added to or subtracted from the composite descriptor.

A.2 Molecular Database Structural Analysis

The average structural composition of a given database may be examined using job type 3 (`-j 3` command-line option). `ligparse` determines the number of times each SMARTS pattern is recognized in each molecule and maintains a sum of the number of times each SMARTS pattern has been found in all molecules. After reading the last molecule in a data set, the mean of the number of times each SMARTS pattern has been found is written to the output file. Standard deviations are also given for some measure of the width of the distribution.

More detailed structural information per compound is available with the `-pcsv` command-line option. With this option, a comma-separated file is written that contains the counts of descriptors and composite descriptors per molecule, with one molecule per line. The file name for the comma-separated file can be adjusted with the `-pcsv` *filename* command-line argument. The comma-separated file can be imported into common spreadsheet programs for further processing. In addition, the structures in the input file are copied to a Maestro-formatted output file with the descriptors added as properties.

A.3 Molecular Database Subset Selection: “Culling”

`ligparse` can use four methods for selecting a subset of molecules, which is called *culling*:

- Culling to reproduce the mean value of composite descriptors within the original data set (`-j 2` command-line option)
- Culling to reproduce user criteria for any descriptors (`-j 1` command-line option)
- Culling to retain only compounds that match user criteria (`-j 4` command-line option)
- Culling to remove only compounds that match user criteria (`-j 5` command-line option)

The first two techniques involve optimization of an error function through random molecule substitution. The third and fourth methods allow the use of `ligparse` as a structural filter, removing or retaining molecules that match the user-defined criteria. For job types 4 and 5, if more compounds match the user criteria than is desired to have in the culled subset, molecules are randomly selected from the subset that matched the user criteria.

For a listing of each descriptor and composite descriptor available with the current `str_keys.type` and `composite.types` files, use the `-d` command-line option.

For job types 1, 4, and 5, the user criteria are passed to `ligparse` with the `-i` command-line option. The file containing the user criteria is the `cull.info` file. Any descriptor or composite descriptor may be used in a `cull.info` file. The file is in free format with each line listing the label and target value for a single descriptor or composite descriptor. Any line beginning with “##” is treated as a comment and ignored. Tabs are not recognized and should not be used. For

job type 1, all descriptor and composite descriptor target values must be either integer or real positive values.

For filtering with job type 4, only molecules that match all criteria listed in the `cull.info` file are *retained*. For filtering with job type 5, only molecules that match all criteria listed in the `cull.info` file are *removed*. These behaviors may be modified with the `-any` option, which changes the selection behavior from removing or retaining only compounds that match *all* criteria to removing or retaining only compounds that match *any* of the criteria.

To further enhance the filtering of molecules with job types 4 and 5, you can retain compounds with descriptor values greater than a given value (>*X*), less than a given value (<*X*), or within some range (*X*-*Y*). Ranges are defined by $X < value < Y$, thus a molecular weight range of `300.0-400.0` retains only molecules with molecular weights greater than 300 and less than 400. These conditions are valid only when running `-j 4` or `-j 5` jobs.

Two example `cull.info` files are presented below.

```
## Distribution A for job type 1
Num rings                          2.600
Num charged donor groups           0.000
Num charged acceptor groups        0.000
Num reactive groups                0.000
Molecular weight                 352.247
Num atoms                         41.800


## Filtering for job types 4 or 5
Num rings            0.99-3.01
Neutral Carbonyls    <1.0
```

A.4 The ligparse Command

`ligparse` can only be run from the command line.

Syntax:

`ligparse` {`-mae` *input-file*|`-sdf` *input-file*} [*options*]

You must specify an input file with either the `-mae` or the `-sdf` flag. The output file will be of the same type as the input file. Examples of the `ligparse` command are given below. The order of the specification of the options and the input file is irrelevant. For a description of the options, run the command with the -h option.

For information on how `ligparse` is called by `ligprep`, see Section 4.9 on page 47.

Examples:

To analyze structural components in a data set:

```
ligparse -j 3 {-mae|-sdf} input-file [ -s str-keys -c composite-types
-o output-file -pcsv -csv csv-file -sort -bofix]
```

To select a subset of structures that approximately reproduce user criteria:

```
ligparse -j 1 {-mae|-sdf} input-file -i criteria-file {-k|-n} what-to-keep
[-s str-keys -c composite-types -o output-file -pcsv -csv csv-file -sort
–bofix -l label]
```

To select a subset of structures to retain original property distributions in the data set:

```
ligparse -j 2 {-mae|-sdf} input-file {-k|-n} what-to-keep [-s str-keys
-c composite-types -o output-file -pcsv -csv csv-file -sort -bofix -l label]
```

To select a subset of structures that match user criteria:

```
ligparse -j 4 {-mae|-sdf} input-file -i criteria-file {-k|-n} what-to-keep
[–s str-keys -c composite-types -o output-file -pcsv -csv csv-file -sort
–l label -any -all -bofix]
```

To remove structures that match user criteria:

```
ligparse -j 5 {-mae|-sdf} input-file -i criteria-file {-k|-n} what-to-keep
[-s str-keys -c composite-types -o output-file -pcsv -csv csv-file -sort
–l label -any -all -bofix]
```

To select a random subset of structures:

```
ligparse -j 6 {-mae|-sdf} input-file {-k|-n} what-to-keep [ -o output-file
-pcsv -csv csv-file -l label -any -all -bofix -seed random-seed]
```

A.5 Input and Output Files

Valid input file names for `ligparse` are *jobname*`.mae` for Maestro files, and *jobname*`.sdf` or *jobname*`.sd` for SD files. The stem of the input file name, *jobname*, is used to construct the default output file name. The averages of descriptor values in the input file are written to *jobname*`.out`. The name of this file can be set with the `-o` option.

For job type 3, ligparse copies the structures from the input file to a Maestro output file, and adds the descriptors as properties to this file. The file is named *jobname*`-out.mae`.

For job types 1, 2, 4, and 5, `ligparse` writes a structure file named *jobname*`-`*label*`.mae` or *jobname*`-`*label*`.sdf`. Here, *label* has the default value of `cull` and can be set with the `-l`

option. The input format dictates the file format of the culled structure file. For example, if the input file is a Maestro file, the output contains a subset of molecules in Maestro format. The properties calculated by `ligparse` are added to the output structure files for job types 4 and 5.

If the `-pcsv` option is given, an additional file is written, *jobname*`.csv`. This comma-separated file contains the value of each descriptor per molecule. The file name can be set with the `-csv` option.

Examples of `ligparse` commands and the files generated are given below.

Example 1:

```
ligparse -sdf database.sdf -j 1 -k 10 -i cull.info -l set1
```

Keeps 10 percent of `database.sdf` structures selected to maintain average values of descriptors defined in `cull.info`. Output includes 2 files:

`database.out`	File listing the results of culling and structural information for the starting set of molecules and the culled set of molecules for the descriptors of interest.
`database-set1.sdf`	MDL file of 10 percent of retained structures.

Example 2:

```
ligparse -mae database.mae -j 1 -k 20 -i ../cull.info    -o db1.stat -
pcsv
```

Keeps 20 percent of `database.mae` structures selected to maintain average values of the descriptors listed in the file, `../cull.info`. Output includes 3 files:

`database-cull.mae`	Maestro file of 20 percent of retained structures.
`db1.stat`	File listing the results of culling and structural information for the starting set of molecules and the culled set of molecules for the descriptors of interest.
`database.prop`	File listing the number of times each descriptor was identified per molecule.

Example 3:

```
ligparse -mae database.mae -j 2 -n 500 -l small
```

Keeps 500 `database.mae` structures selected to maintain average values of descriptors found in the original dataset. Output includes 2 files:

`database.out`	File listing the results of culling and structural information for the starting set of molecules and the culled set of molecules for the descriptors of interest.
`database-small.mae`	Maestro file of 50 percent of the retained structures.

Example 4:

```
ligparse -sdf database.sdf -j 3 -c my_composite.types
   -s my_str_keys.types
```

Determines the average values of a number of descriptors for structure in the SD format file, `database.sdf`. The descriptors are read from the `my_str_keys.types` file, with composite descriptor read from the `my_composite.types` file. Output includes 1 file:

`database.out`	File listing average number of various structural keys for compounds in `database.sdf`.

Getting Help

Information about Schrödinger software is available in two main places:

- The `docs` folder (directory) of your software installation, which contains HTML and PDF documentation. Index pages are available in this folder.
- The Schrödinger web site, http://www.schrodinger.com/, In particular, you can use the Knowledge Base, http://www.schrodinger.com/kb, to find current information on a range of topics, and the Known Issues page, http://www.schrodinger.com/knownissues, to find information on software issues.

Finding Information in Maestro

Maestro provides access to nearly all the information available on Schrödinger software.

To get information:

- Pause the pointer over a GUI feature (button, menu item, menu, ...). In the main window, information is displayed in the Auto-Help text box, which is located at the foot of the main window, or in a tooltip. In other panels, information is displayed in a tooltip.

  If the tooltip does not appear within a second, check that Show tooltips is selected under General → Appearance in the Preferences panel, which you can open with CTRL+, (⌘,). Not all features have tooltips.

- Click the Help button in the lower right corner of a panel or press F1, for information about a panel or the tab that is displayed in a panel. The help topic is displayed in the Help panel. The button may have text or an icon:

  

- Choose Help → Online Help or press CTRL+H (⌘H) to open the default help topic.
- When help is displayed in the Help panel, use the navigation links in the help topic or search the help.
- Choose Help → Documentation Index, to open a page that has links to all the documents. Click a link to open the document.

- Choose Help → Search Manuals to search the manuals. The search tab in Adobe Reader opens, and you can search across all the PDF documents. You must have Adobe Reader installed to use this feature.

For information on:

- Problems and solutions: choose Help → Knowledge Base or Help → Known Issues → *product*.
- New software features: choose Help → New Features.
- Python scripting: choose Help → Python Module Overview.
- Utility programs: choose Help → About Utilities.
- Keyboard shortcuts: choose Help → Keyboard Shortcuts.
- Installation and licensing: see the *Installation Guide*.
- Running and managing jobs: see the *Job Control Guide*.
- Using Maestro: see the *Maestro User Manual*.
- Maestro commands: see the *Maestro Command Reference Manual*.

Contacting Technical Support

If you have questions that are not answered from any of the above sources, contact Schrödinger using the information below.

Web:	http://www.schrodinger.com/supportcenter
E-mail:	help@schrodinger.com
Mail:	Schrödinger, 101 SW Main Street, Suite 1300, Portland, OR 97204
Phone:	+1 888 891-4701 (USA, 8am – 8pm Eastern Time)
	+49 621 438-55173 (Europe, 9am – 5pm Central European Time)
Fax:	+1 503 299-4532 (USA, Portland office)
FTP:	ftp://ftp.schrodinger.com

Generally, using the web form is best because you can add machine output and upload files, if necessary. You will need to include the following information:

- All relevant user input and machine output
- LigPrep purchaser (company, research institution, or individual)
- Primary LigPrep user
- Installation, licensing, and machine information as described below.

Gathering Information for Technical Support

The instructions below describe how to gather the required machine, licensing, and installation information, and any other job-related or failure-related information, to send to technical support. Where the instructions depend on the profile used for Maestro, the profile is indicated.

For general enquiries or problems:

1. Open the Diagnostics panel.
   - **Maestro:** Help → Diagnostics
   - **Windows:** Start → All Programs → Schrodinger-2015-2 → Diagnostics
   - **Mac:** Applications → Schrodinger2015-2 → Diagnostics
   - **Command line:** `$SCHRODINGER/diagnostics`
2. When the diagnostics have run, click Technical Support.

   A dialog box opens, with instructions. You can highlight and copy the name of the file.
3. Upload the file specified in the dialog box to the support web form.

   If you have already submitted a support request, use the upload link in the email response from Schrödinger to upload the file. If you need to submit a new request, you can upload the file when you fill in the form.

If your job failed:

1. Open the Monitor panel, using the instructions for your profile as given below:
   - **Maestro/Jaguar/Elements:** Tasks → Monitor Jobs
   - **BioLuminate/MaterialsScience:** Tasks → Job Monitor
2. Select the failed job in the table, and click Postmortem.

   The Postmortem panel opens.
3. If your data is not sensitive and you can send it, select Include structures and deselect Automatically obfuscate path names.
4. Click Create.

   An archive file is created, and an information dialog box with the name and location of the file opens. You can highlight and copy the name of the file.
5. Upload the file specified in the dialog box to the support web form.

   If you have already submitted a support request, use the upload link in the email response from Schrödinger to upload the file. If you need to submit a new request, you can upload the file when you fill in the form.

6. Copy and paste any log messages from the window used to start the interface or the job into the web form (or an e-mail message), or attach them as a file.
   - **Windows:** Right-click in the window and choose Select All, then press ENTER to copy the text.
   - **Mac:** Start the Console application (Applications → Utilities), filter on the application that you used to start the job (Maestro, BioLuminate, Elements), copy the text.

If Maestro failed:

1. Open the Diagnostics panel.
   - **Windows:** Start → All Programs → Schrodinger-2015-2 → Diagnostics
   - **Mac:** Applications → SchrodingerSuite2015-2 → Diagnostics
   - **Linux/command line:** `$SCHRODINGER/diagnostics`
2. When the diagnostics have run, click Technical Support.

   A dialog box opens, with instructions. You can highlight and copy the name of the file.
3. Upload the file specified in the dialog box to the support web form.

   If you have already submitted a support request, use the upload link in the email response from Schrödinger to upload the file. If you need to submit a new request, you can upload the file when you fill in the form.
4. Upload the error files to the support web form.

   The files should be in the following location:

- **Windows:** `%LOCALAPPDATA%\Schrodinger\appcrash`
  (Choose Start → Run and paste this location into the Open text box.)
  Attach `maestro_error_`*`pid`*`.txt` and `maestro.exe_`*`pid_timestamp`*`.dmp`.
- **Mac:** `$HOME/Library/Logs/CrashReporter`
  (Go → Home → Library → Logs → CrashReporter)
  Attach `maestro_error_`*`pid`*`.txt` and `maestro_`*`timestamp_machinename`*`.crash`.
- **Linux:** `$HOME/.schrodinger/appcrash`
  Attach `maestro_error_`*`pid`*`.txt` and `crash_report_`*`timestamp_pid`*`.txt`.

If a Maestro panel failed to open:

1. Copy the text in the dialog box that opens.
2. Paste the text into the support web form.

Index

H

I

J

L

M

N

O

P

R

S

T

U

120 West 45th Street
17th Floor
New York, NY 10036

101 SW Main Street
Suite 1300
Portland, OR 97204

245 First Street
Riverview II, 18th Floor
Cambridge, MA 02142

8910 University Center Lane
Suite 270
San Diego, CA 92122

155 Gibbs St
Suite 430
Rockville, MD 20850-0353

Dynamostraße 13
D-68165 Mannheim
Germany

Zeppelinstraße 73
D-81669 München
Germany

Potsdamer Platz 11
D-10785 Berlin
Germany

Quatro House
Frimley Road
Camberley GU16 7ER
United Kingdom

8F Pacific Century Place
1-11-1 Marunouchi
Chiyoda-ku, Tokyo 100-6208
Japan

No. 102, 4th Block
3rd Main Road, 3rd Stage
Sharada Colony
Basaveshwaranagar
Bangalore 560079, India

SCHRÖDINGER®