AFITT User Manual

version 1.3.1

OpenEye Scientific Software, Inc.

January 23, 2008

9 Bisbee Ct, Suite D
Santa Fe, NM 87508
www.eyesopen.com
support@eyesopen.com

CONTENTS

CHAPTER

ONE

Introduction

AFITT is a crystallographic model building program built on top of the VIDA Visualization Framework, which provides the core of OpenEye's main visualization application. The VIDA Framework is designed to intuitively and effortlessly handle very large data sets while still generating extremely high quality images. It provides multiple modes of display including 1D, 2D, and 3D. Furthermore, the VIDA Framework contains an excellent interface for data analysis as it includes a chemically-aware spreadsheet and a powerful list-based architecture.

AFITT includes capabilities from both OpenEye's Shape and MMFF toolkits as well as powerful new functionality of its own to enable model building, shape fitting, map calculation and display, and real-space refinement using shape and MMFF. AFITT also provides interactive Ramachandran plots and an interface to external refinement programs.

This manual is divided into several chapters, *Installation* should be read by users and administrators that need to install the software on various platforms. The *Changes* chapter describes the changes since the last version. Otherwise, the *Getting Started* chapter provides a good overview of AFITT and it's capabilities and can be used as a quick start for the anxious crystallographer.

The *Ligand Modeling* and *Manual Modeling* sections in the *Modeling* chapter go into more depth on the techniques used to fit small molecules and proteins into density.

The final chapters describes detailed usage of the various widgets available to AFITT.

CHAPTER

TWO

Installation

This chapter provides the necessary details required to install AFITT 1.3. Currently supported platforms include: Microsoft Windows, Linux, SGI Irix, and Mac OS X.

2.1 Microsoft Windows

A simple installation wizard is provided for Microsoft Windows. Double-clicking on the file (e.g. afitt-1.3-microsoft-win32-i686-setup.exe) will start the wizard which will direct the installation process. It is recommended that the installer to be allowed to install AFITT 1.3 in the C:\OpenEye directory if possible. Once the installation process is complete, there will be a AFITT icon on the desktop as well as in the start menu.

To uninstall AFITT, simply click on the *Uninstall AFITT* icon in the C:\OpenEye directory or in the start menu in the *OpenEye\AFITT* group. AFITT can also be uninstalled using the *Add/Remove Programs* option in the Control Panel.

2.2 UNIX/Linux

The UNIX/Linux installation is provided as a gzipped tarball (e.g. afitt-1.3-redhat-9.0-i586.tar.gz). The first step is to create the directory where AFITT will be installed (e.g. /usr/local/openeye). Next, move the installation file to that directory and then gunzip and untar the file using the following commands:

```
tar zxvf afitt-1.3-redhat-9.0-i586.tar.gz
```

This will create a AFITT directory with all of the appropriate files in it. To run AFITT, enter this directory and call:

```
./afitt.
```

This directory can also be added to the `PATH` environment variable so that `afitt` can be typed anywhere to launch the application.

To uninstall AFITT, simply remove the directory in which the application is installed. Nothing else is required.

2.3 Mac OS X

The Mac OS X version comes pre-bundled. All that is required is to drag the AFITT icon to the desired location on the computer such as the Applications folder. Double-clicking on the icon will launch AFITT. The first time AFITT is run, AFITT may prompt for the location of the license file. If this occurs, there will be two AFITT icons in the Dock until AFITT is restarted.

To uninstall AFITT, simply drag the AFITT icon to the trash from the folder in which it was installed. Nothing else is required.

When AFITT is run, it obtains all of its environment variables from the users `~/.profile` file. Users of shells besides the *bash* shell must still alter the `~/.profile` file in order to pass environment variables through to AFITT.

2.3.1 Platform differences

Microsoft Windows and Linux frequently use *Ctrl* keyboard combinations. For all menu operations such as *copy*, *paste* and *undo*, Mac OS X uses the *Apple/Command* button instead.

Similarly, to display information on mouse over in the 3D window, the *Alt/Option* button must be held down instead of the *Ctrl* button.

2.3.2 Launching from the command line

In a terminal window, AFITT can be launched from the command line as follows:

```
open -a afitt
```

as long as AFITT is in the search path. If AFITT is installed into the Applications directory, it will automatically be in the search path.

2.4 Licenses

In order to run AFITT, a valid license provided by OpenEye must be present on the installed computer. OpenEye license files are typically named `oe_license.txt` and most OpenEye software products (including AFITT) expect to find that license file in the location specified by the `OE_LICENSE` environment variable or in the directory specified by the `OE_DIR` environment variable.

If neither of the above environment variables are defined on the computer, AFITT will prompt the user at run time to locate a valid license file. If a valid license file is specified, AFITT will remember the location of that file and will not prompt again until that license expires or is removed.

CHAPTER

THREE

Getting Started

This chapter is designed to help familiarize the user with the AFITT application and interface in general. While AFITT looks very similar on all platforms, there are subtle differences between platforms, such as how each platform displays certain specific dialogs (the color selection dialog for instance). All screenshots provided in this manual were taken using the Microsoft Windows version of AFITT and therefore the actual appearance on other platforms may vary slightly from the images seen here.

3.1 Why Another Modelling Program?

Current methods of ligand fitting such as topological analysis of electron density [1], global optimization of position and conformation of a ligand in a density blob [2], interatomic distance matrices [3, 4, 5, 6, 7] or by varying torsion dihedral angles of shape-matched ligand conformations [8] are unable to prevent creation of high-energy, sometimes even chemically unrealistic, ligand models.

Many such density fitting artifacts have been analyzed resulting in about 10% of ligands deposited up through 2004 including high amounts ($> 10kcal$) of strain[9].

AFITT uses a combination of OpenEye's shape fitting technology and a force field to maintain proper chemistry while fitting the ligand to density[10]. AFITT also automatically enumerates stereo-chemistry and finds the best stereo variant that fits the supplied data.

These techniques have been extended to include residue and side-chain fitting facilitating protein modeling.

Additionally, AFITT generates quality refinement dictionaries[11] (including detecting and preparing covalent bonds) for input to *refmac5*[12] or *cns/x-plor*[13] refinement programs.

In summary, AFITT provides:

- Fast, accurate ligand fitting to density.
- Protein residue and side-chain fiting to density.
- Generation of high-quality refinement dictionaries.

3.2 General Concepts

There are a number of general concepts that are important to discuss early in order to explain how user interaction with AFITT works.

AFITT can real-space fit both ligands and proteins using a variety of techniques such as *best-rotamer search*, *rigid body* and *MMFF/Shape fitting*. These operations can be performed at any time and are fully undoable using AFITT's undo operations available in the *Edit* menu.

AFITT divides crystallographic workflow into multiple *Tasks*. Each task has a seperate set of user-interface widgets and components designed to facilitate workflow. The current tasks are *Ligand*, *Protein* and *Refinement*.

Each *Task* has a different default layout that may be changed by the user. For each task, however, there are commonalities that will be discussed first.

3.3 Object States

AFITT is a stateful application and employs many special states (or properties) in its interaction with the user. The most important of these are: *Focused*, *Visible*, *Locked*, *Marked*, and *Selected*.

Each object displayed on the screen can be in one these states. By *object* we mean anything loaded into AFITT including molecules, density maps and surfaces (sometimes refered to as *blobs*). Some of these states are persistent such as the *Locked* state and some may be transient, for example a *Focused* object will be replaced when a new object becomes *Focused*.

The main reason that states are important is that many operations performed in AFITT operate on a subset of objects and this subset is determined by a given objects state. The user can choose a default *scope* of operation. For example, if the user wants to make a molecular surface for every loaded molecule, the *All* scope should be chosen. For more information on *scope* see section 3.4

3.3.1 Focused State

The *Focused* state is a property of a single object in the application; there can be only one *Focused* object at any given time. Being *Focused* indicates that that particular object is the current object of interest and is therefore the primary focus of all relevant windows. If the *Focused* object is not already *Visible* (see Section 3.3.2), it will behave as if it were *Visible* for as long as it remains *Focused*.

In addition, when a molecule is the *Focused* object, its SMILES representation is displayed in the application's status bar (which can be found at the bottom of the application window). If any of the molecule's atoms are selected, that selection will be indicated in the SMILES display by the coloring of relevant atoms. Selected atoms are colored using the current *Selection Color* (the default is orange) and are also in a bolded font while all of the unselected atoms will be displayed in their usual normal black font.

For large molecules ($>$100 atoms in this case), no SMILES representation is displayed as it would not fit in the status bar.

3.3.2 Visible State

The *Visible* state is an indicator of whether or not the associated object is drawn in the 3D display (see Chapter 5). Multiple objects can be *Visible* simultaneously. If AFITT is operating in tiled mode, each *Visible* object will be displayed in its own individual pane.

If the 2D display (see Chapter 10) is also shown and AFITT is in tiled mode, *Visible* molecules will be drawn in the 2D display as well as the 3D display.

3.3.3 Locked State

The *Locked* state is essentially the same as the *Visible* state (see Section 3.3.2) except in the situation where the application is in tiled mode, in which case each *Locked* object will be displayed in every pane of the 3D display. However, a *Locked* object can also be simultaneously *Visible* and/or *Focused* (see section 3.3.1) in which case it will be displayed in its own individual pane as well as every other pane in the 3D display.

3.3.4 Marked State

The *Marked* state is simply an indicator of user interest in the associated object. The *Marked* state can be used to help filter data as well as to specify the desired input to application functions. For instance, the *Marked* state can be used to specify which molecules out of a large list are saved. Finally, when certain operations generate too many results to be displayed at one time, the *Marked* state can be used to indicate the results of those operations.

3.3.5 Selected State

The *Selected* state is a special property of the *Visible* state (see Section 3.3.2). Only objects that are *Visible*, *Focused* (see Section 3.3.1), or alternatively *Locked* (see section 3.3.3) can be *Selected*. Much like the *Marked* state (see Section 3.3.4), the *Selected* state is a indicator of interest and provides an input set to application functions. Unlike, the *Marked* state, the *Selected* state is more transient and is easily cleared. The act of *selection* is usually performed in either the 3D (see Chapter 5) or 2D (see Chapter 10) display. Selected objects are usually indicated by a change of color (the default is orange) in these displays. More details on the actual process of selection can be found in the 3D and 2D display chapters.

3.4 Scope

Scope is not a specific state or property like those discussed in the previous section but is rather an indicator as to which objects the application should operate on. The default application scope is *Focused* which means that all of the application functions will operate on the *Focused* object (see Section 3.3.1).

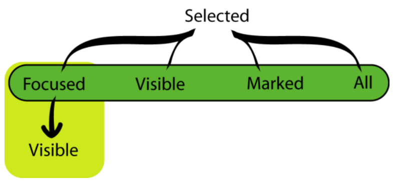


Figure 3.1: Available Scopes

However, as seen in figure 3.1, if there is anything currently *Selected* (see Section 3.3.5), the selected set will take precedence over the current scope. Furthermore, in *Focused* scope, if there is no *Selected* set and no *Focused* object, the *Focused* scope defaults to behaving as if it were the *Visible* scope.

The *Visible* scope operates on all objects that are currently *Visible* (see Section 3.3.2), *Locked* (see Section 3.3.3), and/or *Focused*.

The *Marked* scope operates on all objects that are currently *Marked* (see Section 3.3.4).

The *All* scope operates on every object currently loaded in the application. Operating on the *All* scope can be a very lengthy process, and as such, its progress is displayed in the progress bar in lower right hand corner of the application. Fortunately, operations in the *All* scope can be halted if desired by clicking on the *Stop* button immediately to the left of the progress bar.

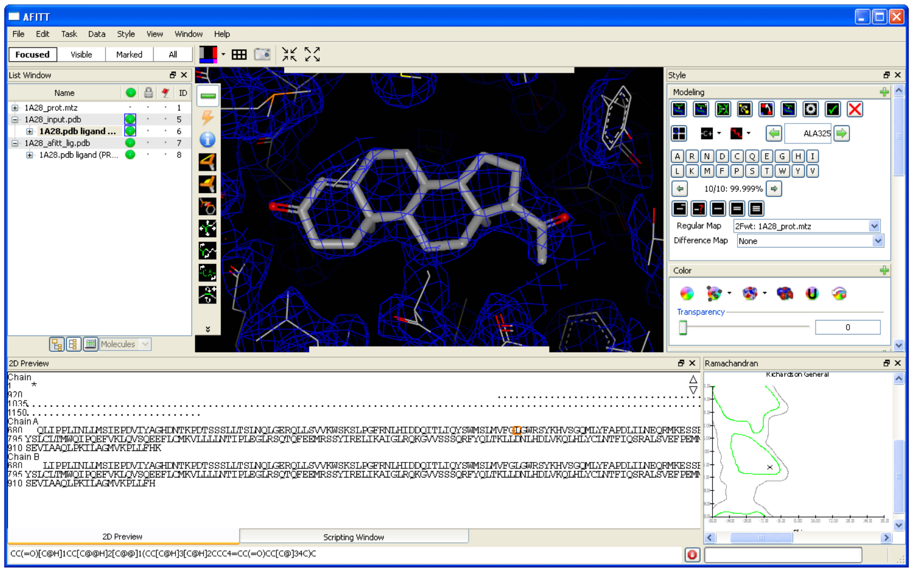
The current scope is displayed in and can be modified from the main application toolbar.

3.4.1 Annoyances with Scope

The largest problem that users have with scope is the difference between a *focused* object and a *visible* object. A *focused* object is always visible but because another molecule may be *focused* next, say by selecting an atom, the previous *focused* object may disappear since it is no longer *focused*. To prevent this, simply make the molecule *visible* then focusing on another molecule will not make it disappear. To make an object permanetly visible click on the dot next to the object in the *List Window* or the *Ligand modeling window*. See the *Layout* section next for more details.

3.5 Layout

The layout of the application should be relatively familiar to most users. AFITT provides a menu bar with many standard as well as specific menus, a toolbar for common operations, a central main window, and a number of peripheral windows. An example of the layout can be see in Figure 3.2.



```
Chain
1   *
920
1035..........................................................
1150...............................
Chain A
680   QLIPPLINLLMSIEPDVIYAGHDNTKPDTSSSLLTSLNQLGERQLLSVVKWSKSLPGFRNLHIDDQITLIQYSWMSLMVFGLGWRSYKHVSGQMLYFAPDLIINEQRMKESSF
795 YSLCLTMWQIPQEFVKLQVSQEEFLCMKVLLLLNTIPLEGLRSQTQFEEMRSSYIRELIKAIGLRQKGVVSSSQRFYQLTKLLDNLHDLVKQLHLYCLNTFIQSRALSVEFPEMM
910 SEVIAAQLPKILAGMVKPLLFHK
Chain B
680    LIPPLINLLMSIEPDVIYAGHDNTKPDTSSSLLTSLNQLGERQLLSVVKWSKSLPGFRNLHIDDQITLIQYSWMSLMVFGLGWRSYKHVSGQMLYFAPDLIINEQRMKESSF
795 YSLCLTMWQIPQEFVKLQVSQEEFLCMKVLLLLNTIPLEGLRSQTQFEEMRSSYIRELIKAIGLRQKGVVSSSQRFYQLTKLLDNLHDLVKQLHLYCLNTFIQSRALSVEFPEMM
910 SEVIAAQLPKILAGMVKPLLFH
```

```
CC(=O)[C@H]1CC[C@@H]2[C@@]1(CC[C@H]3[C@H]2CCC4=CC(=O)CC[C@]34C)C
```

Figure 3.2: Docked window layout

The layout including the position and visibility of the main and peripheral windows is saved on exit and will be restored the next time the application is run. The layout of the windows can be changed by clicking on the title bar of the individual window and dragging it to the desired location. A placeholder window will appear when the mouse is over an area where the dragged window can be placed as seen in Figure 3.3.

Furthermore, multiple windows can be tiled on top of each other into a tab controlled area as can be seen in the bottom window in both Figures 3.2 and 3.3.

In addition to manually changing the layout, there are a number of predefined layouts which are accessible via a drop down button on the toolbar as can be seen in Figure 3.4. The colored icon associated with each option indicates which peripheral windows will be shown and where. Hovering the mouse over any given option for about one second will generate a preview image of the resulting layout. The "Save" option allows for the creation and storage of custom layouts. The "Organize" option enables the reorganization of the ordering of the layouts.

3.5.1 Main Windows

The main window is the window that occupies the central area of the application on the screen. At any given time, there can be only one main window; however, there are multiple windows which are capable of being the main window if desired. The choice of main window can be controlled by selecting the desired option from the "Main window" submenu in the top-level *Windows* menu. In Figure 3.2, the 3D display (see Chapter 5) is serving as the main window.

Currently, there are three different main window options: the 3D display, the 2D display (see Chapter 10), and the spreadsheet (see Chapter 11). More details about the individual windows can be found in their relevant chapters.

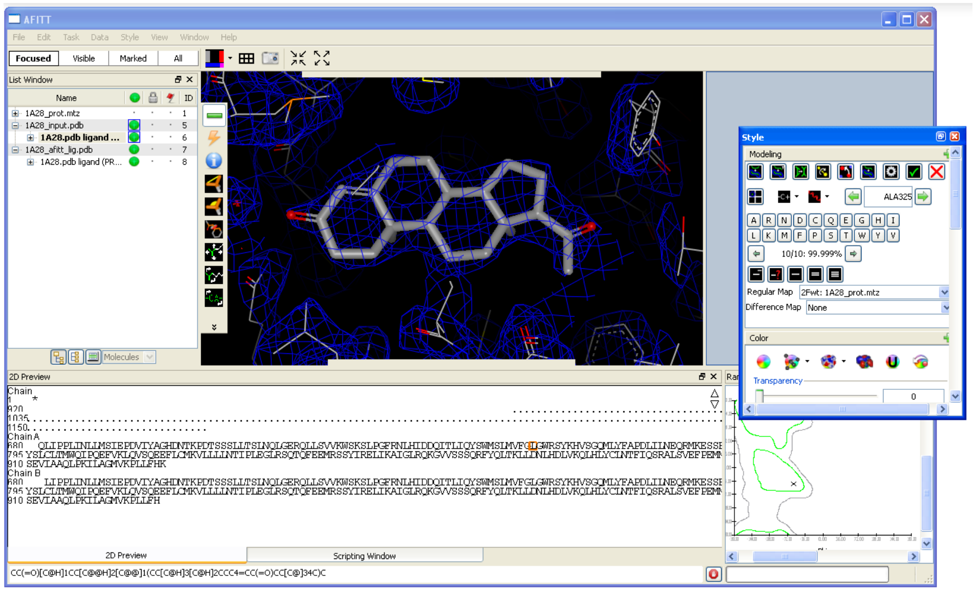


Figure 3.3: Changing the layout by dragging

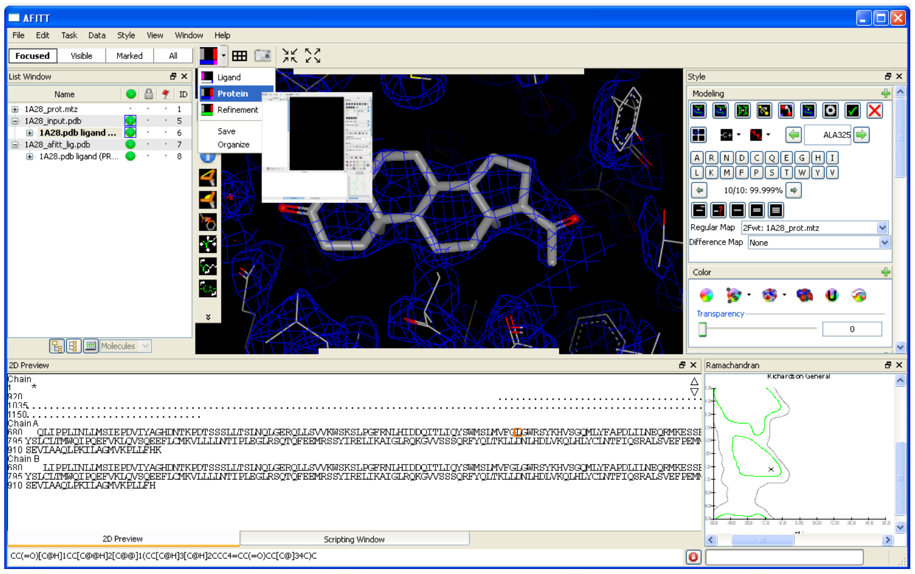


Figure 3.4: Changing the layout from the toolbar

Fullscreen

In addition to the standard layout described above, the main window can optionally be displayed in fullscreen mode, by selecting the “Fullscreen” option in the top-level *View* menu. Pressing the “Esc” key will exit fullscreen mode (please note that clicking the mouse on the screen may be required before hitting the “Esc” key in order to ensure that the key press is registered).

When in fullscreen mode, the main menu bar and status bar are hidden by default. However, they can be toggled on or off by hitting the “F1” and “F2” keys respectively.

The 3D display normally contains popup toolbars around the edge of the window. These toolbars remain when in fullscreen mode to allow for easy access to display functionality. For more details on these toolbars see Chapter 5.

3.5.2 Peripheral Windows

In addition to the main window, there can be multiple peripheral windows docked around the edges of the main window or as separate floating top level windows. Furthermore, multiple windows can be placed on top of each other in a tabbed region.

Currently, multiple peripheral windows (including the potential main windows) are available:

- 2D Display - see Chapter 10
- 2D Preview - see Chapter 10
- 3D Viewer - see Chapter 5
- List Window - see Chapter 12
- Process Manager - see Chapter 13
- Scripting Window - see Chapter 14
- Spreadsheet - see Chapter 11
- Style Control - she Chapter 5

The display of peripheral windows is controlled by toggling the desired window options in the *Windows* menu. Currently visible windows are indicated by the presence of a check mark next to their names. Peripheral can also be hidden by clicking on the ‘X’ in the upper right hand corner of the window. As mentioned above, the layout of the peripheral windows can be controlled by dragging the individual windows to their desired locations (see Figure 3.3).

3.6 Undo/Redo

AFITT provides multi-step undo and redo functionality of almost all operations. Operations can be undone by selecting the “Undo” option in the *Edit* menu or by hitting *Ctrl+Z* on the keyboard. Either of these will undo the last operation performed. This can be done repeatedly or the undo history can be viewed in the “Undo History” option the *Edit* menu which allows selecting how far back into the undo history to go. Please note that while this menu only displays the 10 most recent operations, the undo history can be much greater, so feel free to revisit the “Undo History” option to go back further.

Operations that were just undone can be redone by selecting the “Redo” option in *Edit* menu or by hitting *Ctrl+Y* on the keyboard. Either of these will redo the last operation undone. This can be done repeatedly or the redo history can be viewed in the “Redo History” option in the *Edit* menu which allows selecting how far into the redo history to go. Like the undo history, this menu only displays the 10 most recent undone operations, but the redo history may be much greater and can be revisited to go further.

3.7 3D Window Usage

Most of the time spent in AFITT will be spent working in the 3D viewing window which is the central area of the application. This window shows the currently visible objects, such as density grids, molecules, proteins and blobs. Left-clicking on most items will select them. For instance, left-clicking on a blob will select that blob and focus it. The selected blob will then be used to extract the density region that will be used to fit the small molecule.

Mouse Map - Mouse operations can be found in chapter 20.

3.8 Spreadsheet Usage

The spreadsheet shows more details about the objects that are loaded in AFITT. It is seperated into five tabs: *Molecules*, *Proteins*, *Maps*, *Blobs* and *Results*. The *Molecules* tab displays all of the small molecules that are currently loaded in AFITT. The *Proteins* tab displays all of the proteins currently loaded in AFITT. The *Maps* tab contains the currently loaded densities and grids. The *Blobs* tab shows the auto-detected blobs and surfaces. Finally, the *Results* tab contains the small molecules that have been fit to density. For more information about using the spreadsheet, see Chapter 11.

3.9 2D Preview Usage

The 2D Preview window shows either a 2D depiction of the small molecule or the sequence of the protein depending on which one is currently focused. When deaeling with small moleculees, this view may be unecessary as the current ligand’s 2D structure is also visible in the 3D Window.

3.10 How to Center

The default method of centering is by focusing or selecting a molecule and then clicking the *Center* button in the upper toolbar. This button is next to the *Screenshot* button that looks like a camera.

If an atom or set of atoms are selected, the *Center* button will center on the seleted atoms.

Alternatively, right clicking on a molecule in the *3D Window*, *List Window* or spreadsheet will popup a menu that includes an option to center. Selecting center from this menu will center on the object.

3.11 Tasks

AFITT contains several workflows to assist in various aspects of protein and ligand modeling. These workflows are called *Tasks* and can be selected from the *Task* menu or from the *Layout* widget in the upper toolbar. Each *Task* has a default layout designed for the respective workflow. If the layout is changed, AFITT remembers the changes allowing the user to tailor the workflow to suit their needs.

The basic tasks are *Ligand* and *Manual*. The *Ligand* task focuses the workflow on real-space refinement of small molecules in the corresponding density. The *Manual* task uses OpenEye's shape fitting and molecular forcefield techniques to assist the manual fitting of proteins to density.

Each workflow is described in detail in Chapter 6, however, the basic usage for each task is as follows:

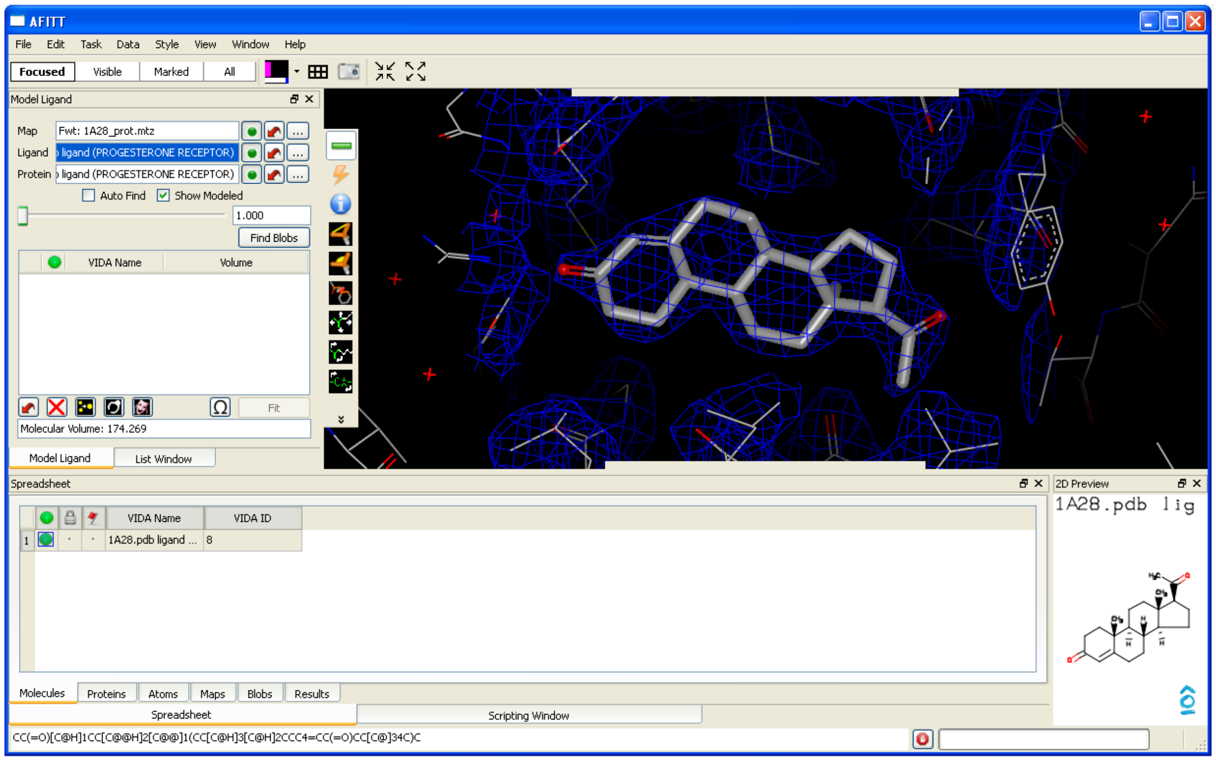
3.12 Ligand Task

The initial task displayed when opening AFITT for the first time is the *Ligand* task. The basic workflow for ligand fitting consists of:

1. Loading the density, ligand and protein
2. Finding the regions of density (a.k.a. blobs) that are indicative of the ligand density
3. Fitting the ligand to these regions

In most cases, the three steps described above are all that are necessary for ligand fitting.

AFITT's initial screen is the *Ligand* task (seen in Figure 3.5). This workflow is divided into four sections described below.



Map: Fwt: 1A28_prot.mtz

Ligand: ligand (PROGESTERONE RECEPTOR)

Protein: ligand (PROGESTERONE RECEPTOR)

[ ] Auto Find [✓ Show Modeled]

1.000

Molecular Volume: 174.269

	VIDA Name	VIDA ID
1	1A28.pdb ligand ...	8

1A28.pdb lig

CC(=O)[C@H]1CC[C@@H]2[C@@]1(CC[C@H]3[C@H]2CCC4=CC(=O)CC[C@]34C)C

Figure 3.5: Ligand mode layout

3.12.1 Ligand Modeling Controls

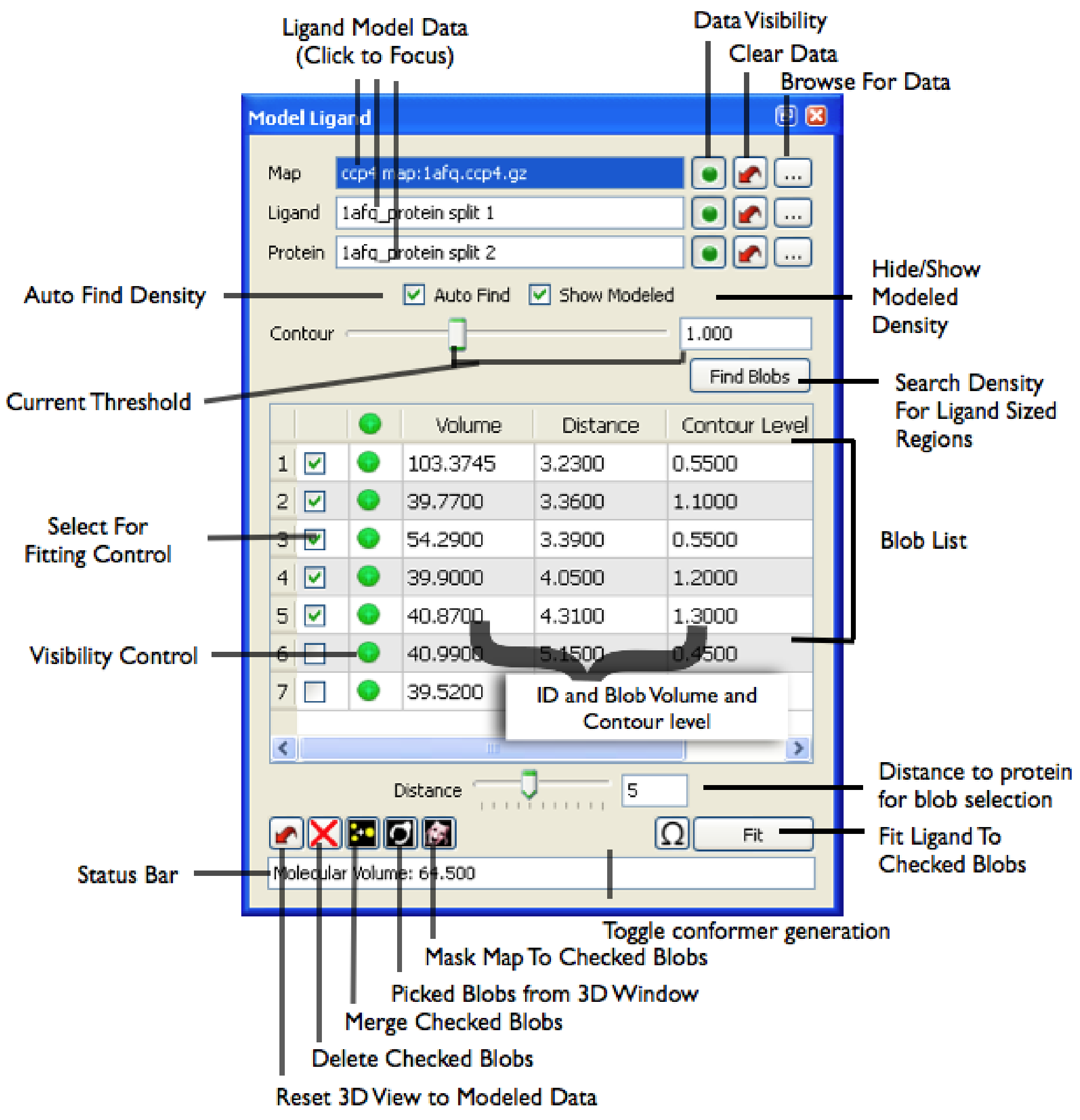
The *Model Ligand Window*, located on the left side of the application (shown in detail in figure 3.6) shows the state and basic operations for fitting small molecules to density. To fit a small molecule to density you will need *density*, a *small molecule* or connection table, and (optionally) a *protein*.

The protein is used to mask the density where the ligand should not be placed. Using the protein is highly recommended as this greatly facilitates the automated blob finding.

Each modeled unit (density, ligand, protein) has a set of controls that can focus the object or toggle its visibility. Clicking on the name of the object will focus it, clicking on the small box to the right of the object will toggle visiblility. Clicking on the arrow will remove the item from the current model and clicking on the button with three dots "..." will browse the current repository for a new object.

Once a model has been loaded, clicking on the *Find Blobs* button will attempt to locate regions of density suitable for the ligand. The *Find Blobs* algorithm uses the current sigma value for the density. If the *Auto* checkbox is selected, then the searching algorithm will examine all density sigmas in an attempt to find a blob of the appropriate size. It is recommended to use the *Auto* option as this has been tuned to searching density for standard ligand sized molecules. The *Auto* checkbox is enabled by default, however AFITT will remember the current state of the checkbox from session to session.

By default, AFITT automatically selects blobs that are within $4\mathring{A}$ to the protein. This distance is computed by an average distance from the blob to the currently modeled protein. By sliding the distance slider seen in Figure 3.6 or by typing in a value in the text entry, the default distance can be altered.


Ligand Model Data
(Click to Focus)
Data Visibility
Clear Data
Browse For Data
Model Ligand
Map
Ligand
Protein
Auto Find Density
Auto Find
Show Modeled
Hide/Show Modeled Density
Contour
Find Blobs
Current Threshold
Search Density For Ligand Sized Regions
Volume
Distance
Contour Level
Blob List
Select For Fitting Control
Visibility Control
ID and Blob Volume and Contour level
Distance
Distance to protein for blob selection
Fit
Fit Ligand To Checked Blobs
Status Bar
Toggle conformer generation
Mask Map To Checked Blobs
Picked Blobs from 3D Window
Merge Checked Blobs
Delete Checked Blobs
Reset 3D View to Modeled Data

Map: ccp4 map:1afq.ccp4.gz
Ligand: 1afq_protein split 1
Protein: 1afq_protein split 2
✓ Auto Find ✓ Show Modeled
Contour: 1.000

		●	Volume	Distance	Contour Level
1	✓	●	103.3745	3.2300	0.5500
2	✓	●	39.7700	3.3600	1.1000
3	✓	●	54.2900	3.3900	0.5500
4	✓	●	39.9000	4.0500	1.2000
5	✓	●	40.8700	4.3100	1.3000
6	☐	●	40.9900	5.1500	0.4500
7	☐	●	39.5200		

Distance: 5

Molecular Volume: 64.500

Figure 3.6: Modeling Ligand Control

Finally, once one or more blobs has been selected, click on the *Fit* button to begin the fitting process. When complete, the fitted small molecules will appear on the screen and in the spreadsheet.

3.12.2 Analyzing Ligand Results

AFITT fits ligands to density using a combined forcefield that forces ligands into density while maintaining appropriate chemistries.

The potential function being used to adapt the ligand is:

$$V = V_{ff} + \lambda V_{shape}$$

where V_{ff} represents the internal energy of the ligand and V_{shape} is the overlap between the ligand and the electron density. λ is a mixing parameter that represents the degree to which the shape of the density to dominate the combined potential[10].

This function is optimized while the strain placed on the ligand is bounded producing high-quality fits with low-strain ligand conformations.

For each ligand, there are columns in the *Results* spreadsheet that should be noted:

1. *Tanimoto Shape* - This is the measure of how much the original conformation matched density.
2. *Tanimoto MMFF/Shape* - This is a measure of how closely the final result of the combined forcefield matches the shape of the density. Note that this will never be an exact match.
3. *Local Strain* - This indicates how much strain was added to the original conformation during the fitting process.
4. *Lambda* - This is the scalar indicating which component of the combined forcefield was used. > 1 implies that more precedance was given to the Shape force (i.e. density) < 1 implies that more weight was given to the *MMFF* force.

In general, the higher the *Tanimoto MMFF/Shape* value, the better the fit to density. There is one important caveat: *Tanimoto MMFF/Shape* can only be used to compare ligands being fit to the same blob. They are not comparable across different density regions. However, when AFITT fits multiple stereovariants to the same blob, sorting the *Tanimoto MMFF/Shape* column will percolate the highest scoring ligand to the top allowing easy identification of the correct stereochemistry.

3.12.3 What To Do With Too Many Blobs

Sometimes, too many blobs will be found when searching density. To filter out undesired blobs when searching, select an atom or residue in the binding pocket and set a distance halo. Any blobs that are found outside this distance will be rejected. See 5.11.3 for more information on setting the distance halo.

3.12.4 Working With Poor Density

Occasionally, the density is too poor to find a blob using the automated process. In this case, there are three options:

1. Split the Protein - Sometimes, the protein has ligands embedded in the structure. You can use the split options of the Edit menu to separate the ligand from the protein.

2. Use a Box - Under the task menu, there is an option labeled "Add Box...". This prompts for a molecule and then for an optional padding. The extents of the molecule and the padding are used to generate a volume of space that constrains the search. For example, if the location of the active binding site is known, the residues around the active site can be selected and used to define the box. When a box is created, the region is visualized in the 3D window.

   At this point the box can be used as a fitting region or clicking on "Find Blobs" will only search the density inside the box.

   Note: multiple boxes can be used simulataneously.

3. Picking Density - Using the "Blob Picking" tool in the modeling widget, individual density regions can be selected from the 3D window.. Simply click on the "Blob Picking" button and then right-click on a piece of density in the 3D window.

4. SCiNi - Generates a shape contour of the active sites. The *Generate Negative Image* option of the *Task* menu attempts to find the active sites of the protein. This generates a Shape Contour Negative Image by searching the protein space with a collection of small molecule fragments. The resulting shape potential can be used to as a type of "density" from which to manually pick blobs.

   This process may take some time, so be prepared to go out to lunch.

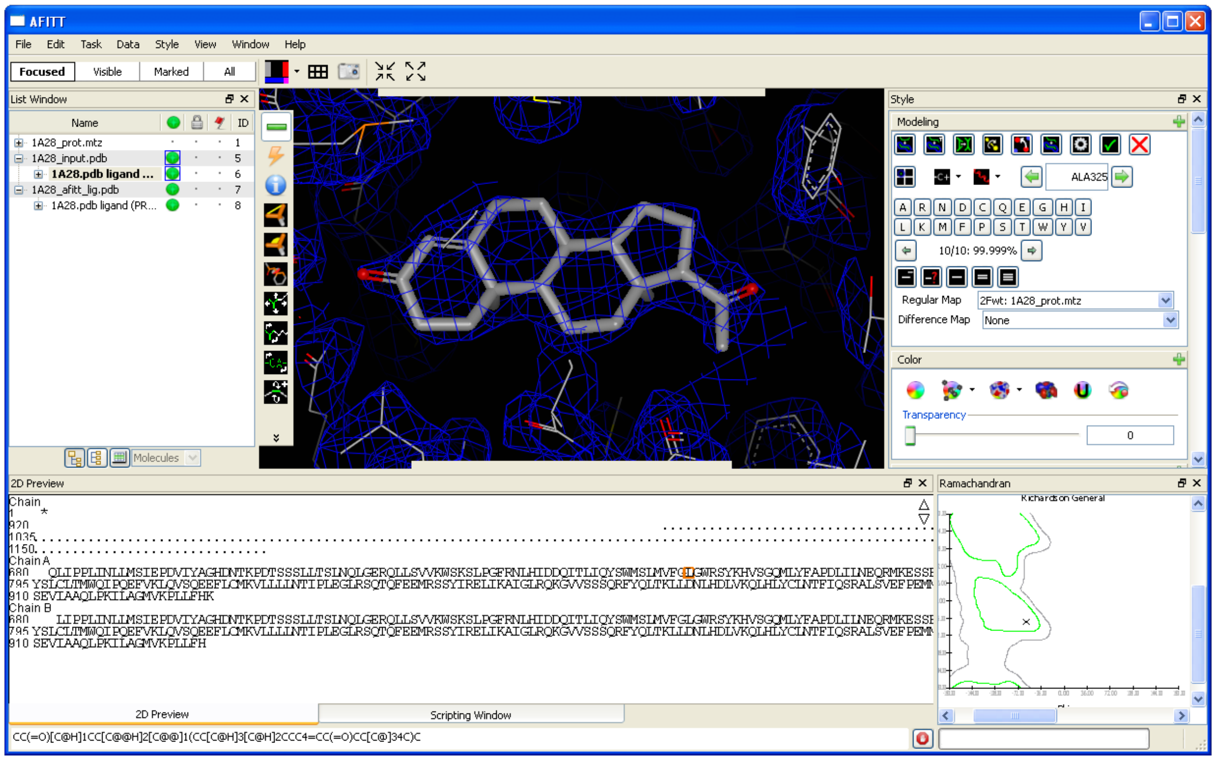
3.13 Protein Task

The *Protein Mode* gives complete control over atom positions in density. The initial view of the manual mode is seen in Figure 3.7.

While the initial view of *Protein Mode* may initially be intimidating, there are only a small handful of operations that are used regularly. These operations also have hot-keys and key stroke combinations to facilitate usage for advanced users. For complete details, see Chapter 6.

To perform modeling operations, one must select the desired structure or residue prior to performing an operation. During some modeling operations, such as rotating Φ and Ψ angles, the behavior of the mouse will alter to perform the requested operation. The current operating mode is indicated at the top right of the *3D Window*. In order to revert to the normal mode of operations, select the *Mouse Performs Selection* option from the *Mouse Toolbar* seen in figure 3.11.

As seen in Figure 3.7, the *Protein Mode* task is divided into sections:



Name	ID
1A28_prot.mtz	1
1A28_input.pdb	5
1A28.pdb ligand ...	6
1A28_afitt_lig.pdb	7
1A28.pdb ligand (PR...	8

Regular Map: 2Fwt: 1A28_prot.mtz
Difference Map: None
Transparency: 0

```
Chain
1      *
920
1035. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
1150. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Chain A
680   QLIPPLINLLMSIEPDVIYAGHDNTKPDTSSSLLTSLNQLGERQLLSVVKWSKSLPGFRNLHIDDQITLIQYSWMSLMVFGLGWRSYKHVSGQMLYFAPDLIILNEQRMKESS
795 YSLCLTMWQIPQEFVKLQVSQEEFLCMKVLLLLNTIPLEGLRSQTQFEEMRSSYIRELIKAIGLRQKGVVSSSQRFYQLTKLLDNLHDLVKQLHLYCLNTFIQSRALSVEFPE
910 SEVIAAQLPKILAGMVKPLLFHK
Chain B
680     LIPPLINLLMSIEPDVIYAGHDNTKPDTSSSLLTSLNQLGERQLLSVVKWSKSLPGFRNLHIDDQITLIQYSWMSLMVFGLGWRSYKHVSGQMLYFAPDLIILNEQRMKESS
795 YSLCLTMWQIPQEFVKLQVSQEEFLCMKVLLLLNTIPLEGLRSQTQFEEMRSSYIRELIKAIGLRQKGVVSSSQRFYQLTKLLDNLHDLVKQLHLYCLNTFIQSRALSVEFPE
910 SEVIAAQLPKILAGMVKPLLFH
```

```
CC(=O)[C@H]1CC[C@@H]2[C@@]1(CC[C@H]3[C@H]2CCC4=CC(=O)CC[C@]34C)C
```

Figure 3.7: Protein mode layout

- *List Window* - On the left side, the *List Window* displays which objects are currently loaded in AFITT along with their visibility and marked states.
- *3D Window* - As in the *Ligand Task* the 3D window takes up a majority of the view and controls most of the action. Changes to protein structure and protein fit to density are indicated here. Atoms, bonds and/or residues may be selected by left-clicking or left-double-clicking.

- *Style Bar and Modeling Widget*

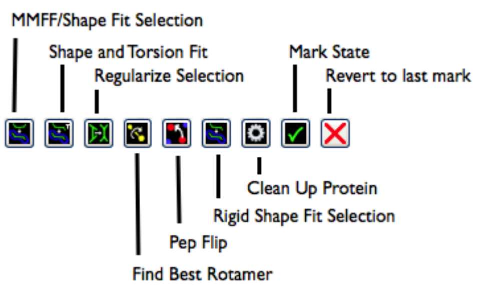


Figure 3.8: Protein Controls

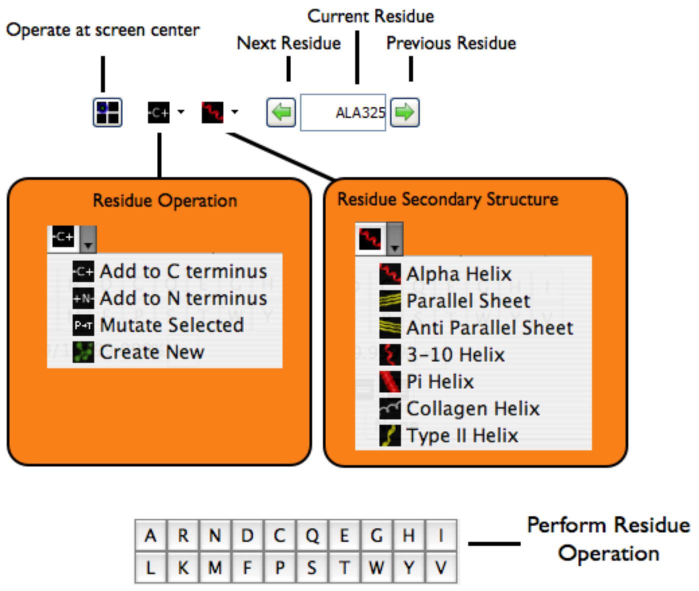


Figure 3.9: Residue Controls

The modeling widget is contained at the top of the *Style Bar*. The *Style Bar* contains many different components that can be used to alter view settings, control density and more. For a complete description of the *Style Bar* please see chapter 5.

The modeling widget is where the remainder of the modeling functionality is located. Modeling functionality includes bond operations (such as breaking and creating bonds), residue operations (such as creating or replacing residues), and rotamer operations (such as locating the highest probabillity rotamers from the various rotamer libraries). A break down of the modeling operations can be seen in Figures 3.8, 3.9 and 3.10.

- *Mouse Toolbar* - this allows control over modifying coordinates, torsions and rotatable bonds as well as inspecting atomic properties such as dihedral angles and residue information. This toolbar is in the right hand side of the 3D window.

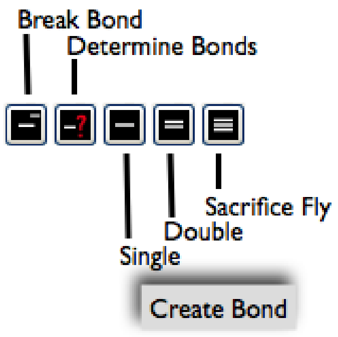


Figure 3.10: Bond Controls

Mouse mode operations change the behavior of the mouse in the 3D window. The default behavior of the mouse controls selection and rotation of the main view. Other mouse modes control altering residue and bond torsions, translating the modeled atoms in 3D and modifying and placing the c-alpha backbones.

Holding the mouse over each button reveals a tooltip describing the operation that will be performed.

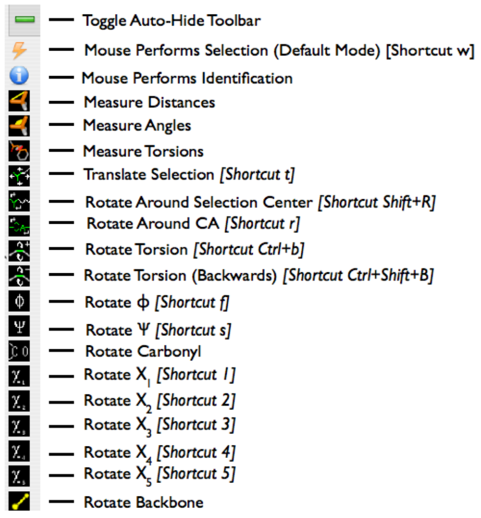
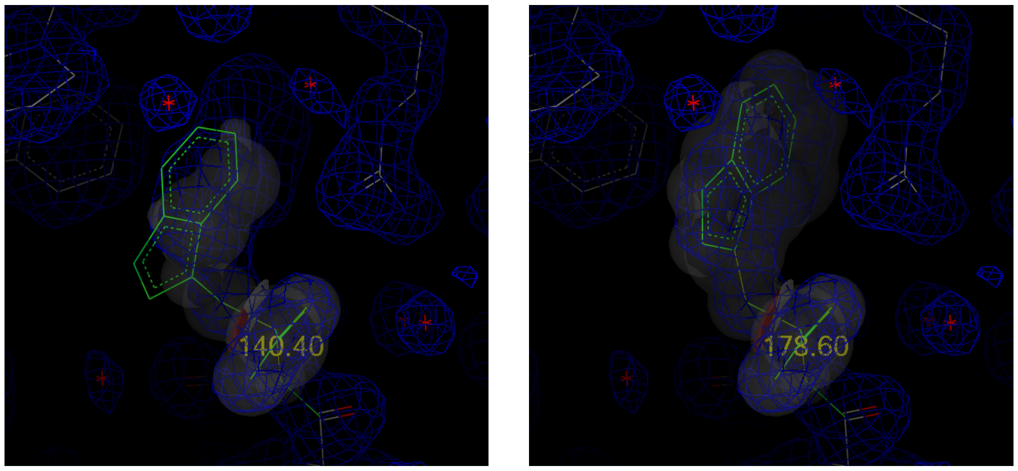

Toggle Auto-Hide Toolbar
Mouse Performs Selection (Default Mode) [Shortcut w]
Mouse Performs Identification
Measure Distances
Measure Angles
Measure Torsions
Translate Selection [Shortcut t]
Rotate Around Selection Center [Shortcut Shift+R]
Rotate Around CA [Shortcut r]
Rotate Torsion [Shortcut Ctrl+b]
Rotate Torsion (Backwards) [Shortcut Ctrl+Shift+B]
Rotate ϕ [Shortcut f]
Rotate Ψ [Shortcut s]
Rotate Carbonyl
Rotate X_1 [Shortcut 1]
Rotate X_2 [Shortcut 2]
Rotate X_3 [Shortcut 3]
Rotate X_4 [Shortcut 4]
Rotate X_5 [Shortcut 5]
Rotate Backbone

Figure 3.11: Mouse Toolbar

Most mouse modes have automatic monitors that show the torsion or rotations angle are being set. Also, if a density grid is available (i.e. visible) the rotated atoms displays a spherical halo the size of which corresponds to its placement in density. Smaller halos indicate that the atom is not surrounded by much density while larger halo's indicate that an atom is surrounded by more density.

These monitors can be turned off from the *Preferences menu*.



(a) Poor density fit (b) Good fensity fit

Figure 3.12: Examples of visualization of poor and good density fits. Noticed that in the poor fitting example, the fitting halo in some of the ring atoms have disappeared

- *Residue Toolbar* - The residue toolbar is located at the bottom of the *3D Window*. This toolbar is designed to modify or append residues to the current protein. If one or more residues are selected, clicking on a residue in the toolbar will replace the currently selected residues. If no residue is selected, then a new residue is append to the end of the protein.

  For example, to append a glycine residue to the end of the current protein, simply click on the “G” button in the residue toolbar as seen in Figure 3.13.

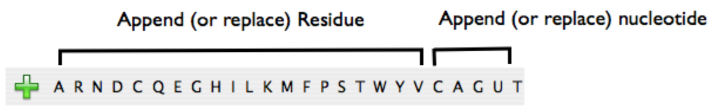

Append (or replace) Residue
Append (or replace) nucleotide
A R N D C Q E G H I L K M F P S T W Y V C A G U T

Figure 3.13: Residue Toolbar

- *Ramachandran Plot* - The ramachandran plot shows the Φ and Ψ status of the protein resdiues. Clicking on a residue in the plot will select it in the *3D Window*. Once a residue is selected the *Mouse Toolbar* can be used to manipulate the Φ, Ψ angles.

- 2D Sequence View The 2D sequence view displays the protein sequence of the currently visible molecule. (If a ligand is *Focused* then it shows the 2D structure of the ligand).

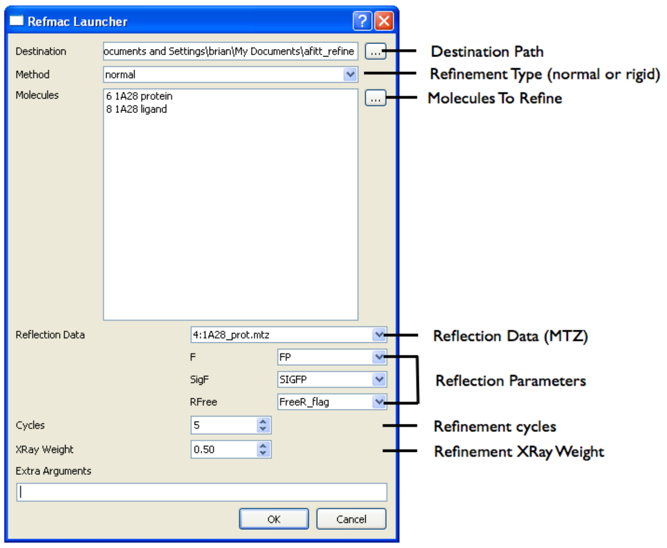
  Residues can be selected by clicking on the appropriate sequence. Right clicking on a selected residue will displays a popup menu that includes various modeling operations that can be used as short cuts.

3.14 Refinement

AFITT automatically prepares protein and ligand input for use with *CCP4's refmac5* protein refinement tool and *X-PLOR/CNX*.

AFITT generates refinement dictionaries using the *MMFF94* force-field including bond-lengths, periods and chiral volumes. AFITT also identifies and includes proper *LINK* records for covalent bonds.

refmac5 can be launched from the *Task* menu by selecting the *Launch Refinement...* option. As long as *refmac5* can be run from the command line, AFITT will be able to launch and refine proteins. The Refmac Launcher widget is seen in Figure 3.14.


Refmac Launcher

Destination	ocuments and Settings\brian\My Documents\afitt_refine
Method	normal
Molecules	6 1A28 protein 8 1A28 ligand
Reflection Data	4:1A28_prot.mtz
F	FP
SigF	SIGFP
RFree	FreeR_flag
Cycles	5
XRay Weight	0.50
Extra Arguments	

Figure 3.14: Refmac Launcher

To view the status of the refinement, simply switch to the *Refinement* task. This workflow shows the current refinements that are being (or have been) run and their status. When a refinement is complete, a prompt will appear and the results can be loaded into AFITT or ignored as desired.

Note: even if *refmac5* is not installed, then AFITT writes the appropriate data (dictionary and pdb file) and the *refmac5* scripts to the destination for future use.

CHAPTER

FOUR

File I/O

4.1 Opening Files

There are multiple mechanisms by which a file can be loaded in AFITT. The simplest method is to select the “Open” option in the *File* menu. This will launch a file selection dialog which prompts the user for the desired files(s) which will subsequently be loaded. A short list of the most recently opened files can also be found in the *File* menu under the “Recents” (or “Open Recents” on Mac OS X) submenu. Selecting an option in this submenu will load the associated file. There is an “Open All” option at the bottom of the recents menu which will load all of the files listed in the recents submenu. Files can also be loaded by specifying them on the command line when starting AFITT or by dragging the file of interest onto the AFITT desktop icon. By default, there can only be one instance of AFITT running at a time, so if an instance of AFITT is running and a new file is specified on the command line or if a file is dragged onto the desktop icon, that file will be loaded in the currently running instance (instead of loading a new instance). If desired, multiple instances can be enabled by selecting the “Allow multiple instances” option in the application preferences (see Chapter 16).

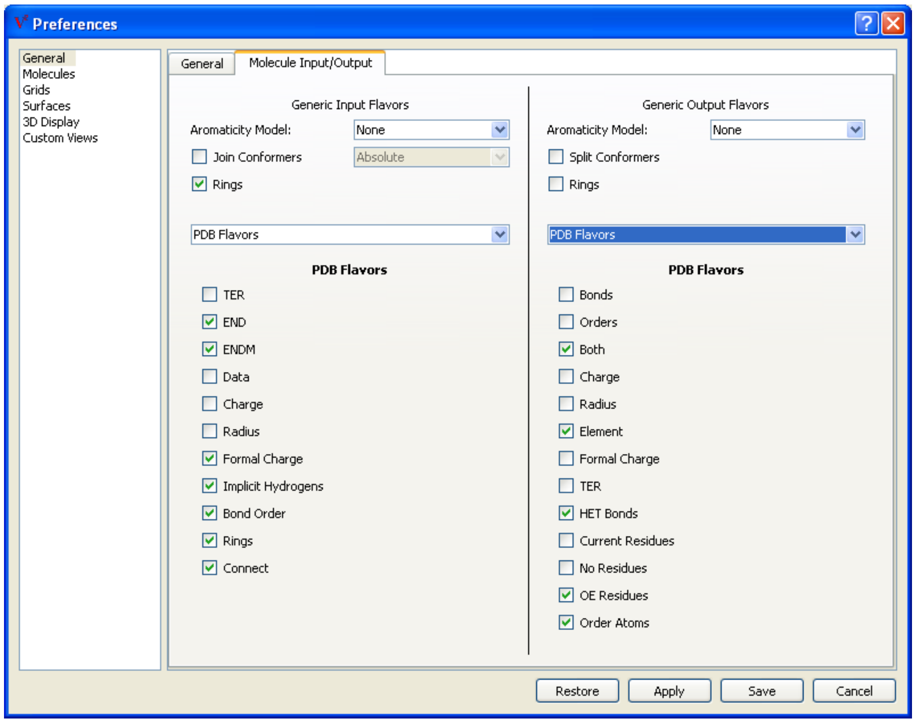
In addition to the “Open” menu item, there is a separate “Open State” item which filters out all files except AFITT state files. State files can be opened using any of the mechanisms described above, but their special nature warranted their own menu item. For more details on state files see Section 4.1.6.

There is also an “Open Special” submenu which contains a “Script” option which like the “Open State” option filters out all files except for Python scripts. The “Open Special” submenu also serves as an excellent target area for adding custom open operations via the scripting interface (see Chapter 14).

4.1.1 Molecules

Multiple molecular file formats are supported for reading and include:

- OpenEye binary format v1 and v2 (.bin and .oeb)
- MDL, RDF, and SDF



Generic Input Flavors

Aromaticity Model: None

[ ] Join Conformers — Absolute

[✓] Rings

PDB Flavors

PDB Flavors

- [ ] TER
- [✓] END
- [✓] ENDM
- [ ] Data
- [ ] Charge
- [ ] Radius
- [✓] Formal Charge
- [✓] Implicit Hydrogens
- [✓] Bond Order
- [✓] Rings
- [✓] Connect

Generic Output Flavors

Aromaticity Model: None

[ ] Split Conformers

[ ] Rings

PDB Flavors

PDB Flavors

- [ ] Bonds
- [ ] Orders
- [✓] Both
- [ ] Charge
- [ ] Radius
- [✓] Element
- [ ] Formal Charge
- [ ] TER
- [✓] HET Bonds
- [ ] Current Residues
- [ ] No Residues
- [✓] OE Residues
- [✓] Order Atoms

Figure 4.1: Advanced molecule input and output options

- Tripos MOL2, MOL2H
- Daylight SMILES, canonical SMILES, and isomeric SMILES
- ChemDraw CDX
- ISIS Sketch
- MacroModel
- MOPAC
- PDB
- XYZ

As is often the case with file formats, the meaning and use of certain fields within a format may change over time which can potentially lead to problems interpreting those files. VIDA, using OpenEye's OEChem toolkit, makes its best effort to interpret all files correctly; however, it does provide a mechanism to allow the user to override the handling of certain formats (SMILES, PDB, Mol2, XYZ, and MacroModel). The ability to change the "flavor" of a specific format can be done in the application preferences as seen in Figure 4.1.

In addition to changing the flavor of a format, there are a few other advanced options available when reading molecules including aromaticity model specification and conformer joining. The desired aromaticity model to be applied can be specified in the pulldown menu next to the "Aromaticity Model" label. The “Join Conformers” checkbox controls whether or not adjacent molecules in an input file will be tested on reading to determine whether they are unique compounds or simply different confomers of the same molecule. The specific test to be performed can be specified in the pulldown menu next to the checkbox. For more specific details on the available aromaticity models and conformer tests, please see the OEChem documentation.

It is important to note that because these advanced options are available through the application preferences, they will be remembered by the application and automatically applied in future “Open” operations unless they are subsequently restored.

4.1.2 Grids

Multiple grid and map formats are supported for reading and include:

- OpenEye grids
- ASCII grids
- GRASP/Delphi grids
- CCP4 maps
- XPLOR maps

4.1.3 MTZ Files

AFITT can open standard *mtz* files output by programs such as *refmac5* and other crystallography applications.

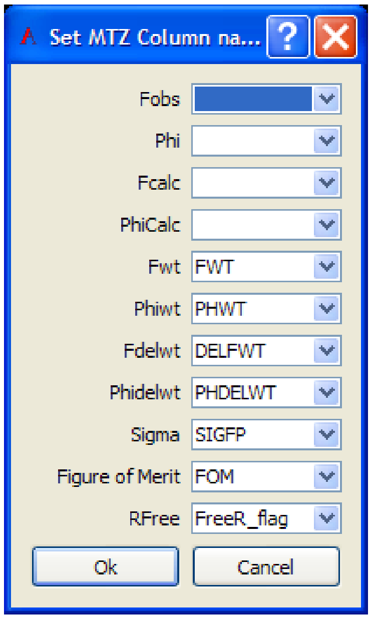
The first time an *mtz* file is opened, AFITT will prompt for the columns to use when loading. (A description of these columns can be seen in Chapter 19.2.5.) This prompt, with the default columns for *refmac5*, can be seen in figure 4.2.

Once the columns are chosen, the type of maps to generate need to be decided. AFITT only displays map choices that can be created from the columns. This dialog can be seen in figure 4.3

AFITT remembers the last type of file opened and will attempt to choose the same columns for the next mtz file. If this succeeds, the file type prompt will not be shown and the maps will automatically be created.

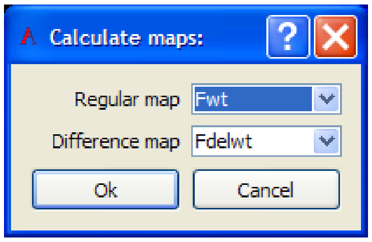
4.1.4 Surfaces

Multiple surface formats are supported for reading and include:



Fobs	
Phi	
Fcalc	
PhiCalc	
Fwt	FWT
Phiwt	PHWT
Fdelwt	DELFWT
Phidelwt	PHDELWT
Sigma	SIGFP
Figure of Merit	FOM
RFree	FreeR_flag

Figure 4.2: Column choosing for mtz files.



Regular map	Fwt
Difference map	Fdelwt

Figure 4.3: Map type chooser for mtz files.

- OpenEye surfaces
- GRASP surfaces

4.1.5 Python Scripts

AFITT supports reading of Python scripts compatible with Python version 2.3. AFITT supports two additional file extensions for Python scripts (.pyv and .vpy) to enable users to associate those file extensions with AFITT without adversely affecting the association of unrelated Python scripts.

For more details about Python scripting, please see Chapter 14 as well as the associated Python scripting API documentation.

4.1.6 State

The entire state of a session can be stored in a single file called a "State File" (.oes). A state file contains all of the molecules, grids, and surfaces that were loaded in AFITT at the time the state file was created. In addition, the state file preserves the actual view, layout, and properties of that session. State files provide an extremely convenient way to save sessions for later work or to share with collaborators. Furthermore, state files are the fastest method of reading and writing large data sets.

To load a state file, choose the "Open State" option in the *File* menu and select the desired state file. State files can also be loaded from the commandline. It is important to note that loading a state file during a run will clear the current state before loading the new one. Therefore, be sure to save the current state before loading a new state if keeping the current state is desired. Furthermore, since state files contain the entire state of the application that generated them, loading a state file will overwrite local preferences; however, they will not be saved on exit.

4.2 Closing Files

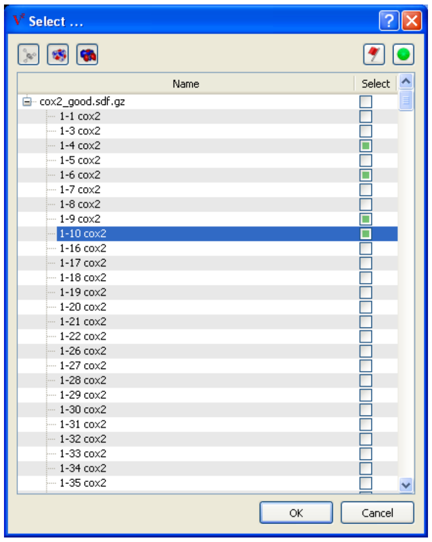
When a file is opened in AFITT, its contents are loaded into its own individual list. To close a given file, simply right-click on the associated list in the List Window (see Chapter 12) and select the "Delete" option. This will close the file and remove the contents from the application. However, if an object is present in multiple lists, deleting one of those lists will not delete that object from the other lists. It is important to note that deleting a list, does not affect the actual file that was read.

To close all of the currently loaded files, select the "Clear All" option in the *File* menu. In addition to clearing everything that is loaded, this option effectively restores the state of the application to what it was in when the application was started.

4.3 Saving Files

AFITT is capable of saving multiple types of files including molecules, grids, surfaces, and state files. The details of the individual formats can be found in the relevant sections below.

A file can be saved by choosing the "Save" option the *File* menu. This will launch a dialog prompting the user to select which objects to save (see Figure 4.4). There are two sets of buttons at the top of this dialog which provide mechanisms for filtering the available options. The three buttons on the top left toggle the display of molecules, surfaces, and grids respectively. The two buttons on the top right provides filters for showing only the *Marked* and/or *Visible* objects respectively. Once one or more objects have been selected and the "OK" button is pressed, a file dialog will appear allowing the user to specify the desired file. The file formats listed in the file dialog filter will reflect which file formats are supported based on the objects selected. Only the OpenEye binary format supports writing multiple object types (molecules, grids, and/or surfaces) to the same file.



Name	Select
cox2_good.sdf.gz	☐
1-1 cox2	☐
1-3 cox2	☐
1-4 cox2	✓
1-5 cox2	☐
1-6 cox2	✓
1-7 cox2	☐
1-8 cox2	☐
1-9 cox2	✓
1-10 cox2	✓
1-16 cox2	☐
1-17 cox2	☐
1-18 cox2	☐
1-19 cox2	☐
1-20 cox2	☐
1-21 cox2	☐
1-22 cox2	☐
1-26 cox2	☐
1-27 cox2	☐
1-28 cox2	☐
1-29 cox2	☐
1-30 cox2	☐
1-31 cox2	☐
1-32 cox2	☐
1-33 cox2	☐
1-34 cox2	☐
1-35 cox2	☐

Figure 4.4: Selecting what to save

This process can be expedited by simplying right-clicking on the desired objects and selecting the "Save" option. This allows the user to bypass the selection dialog and go directly to the file dialog.

4.3.1 Molecules

Multiple molecular file formats are supported for writing and include:

- OpenEye binary format v2 (.oeb)
- MDL, RDF, and SDF
- Tripos MOL2, MOL2H
- Daylight SMILES, canonical SMILES, and isomeric SMILES
- ChemDraw CDX
- MacroModel
- Molecular Formula (MF)
- MOPAC
- FASTA
- PDB
- XYZ

As is often the case with file formats, the meaning and use of certain fields within a format may change over time which can potentially lead to problems interpreting those files. VIDA, using OpenEye's OEChem toolkit, writes all files according to the standard; however, it does provide a mechanism to allow the user to override the handling of certain formats (SMILES, MDL/SDF, PDB, Mol2, MF, MOPAC, and MacroModel). The ability to change the "flavor" of a specific format can be done in the application preferences as seen in Figure 4.1.

In addition to changing the flavor of a format, there are a few other advanced options available when writing molecules including aromaticity model specification and conformer splitting. The desired aromaticity model to be applied can be specified in the pulldown menu next to the "Aromaticity Model" label. For more specific details on the available aromaticity models, please see the OEChem documentation.

It is important to note that because these advanced options are available through the application preferences, they will be remembered by the application and automatically applied in future "Save" operations unless they are subsequently restored.

4.3.2 Grids

Multiple grid formats are supported for writing and include:

- OpenEye grids
- ASCII grids
- GRASP/Delphi grids

4.3.3 Surfaces

Multiple surface formats are supported for writing and include:

- OpenEye surfaces
- GRASP surfaces

4.3.4 State

The entire state of a session can be stored in a single file called a "State File" (.oes). A state file contains all of the molecules, grids, and surfaces that were loaded in AFITT at the time the state file was created. In addition, the state file preserves the actual view, layout, and properties of that session. State files provide an extremely convenient way to save sessions for later work or to share with collaborators. Furthermore, state files are the fastest method of reading and writing large data sets.

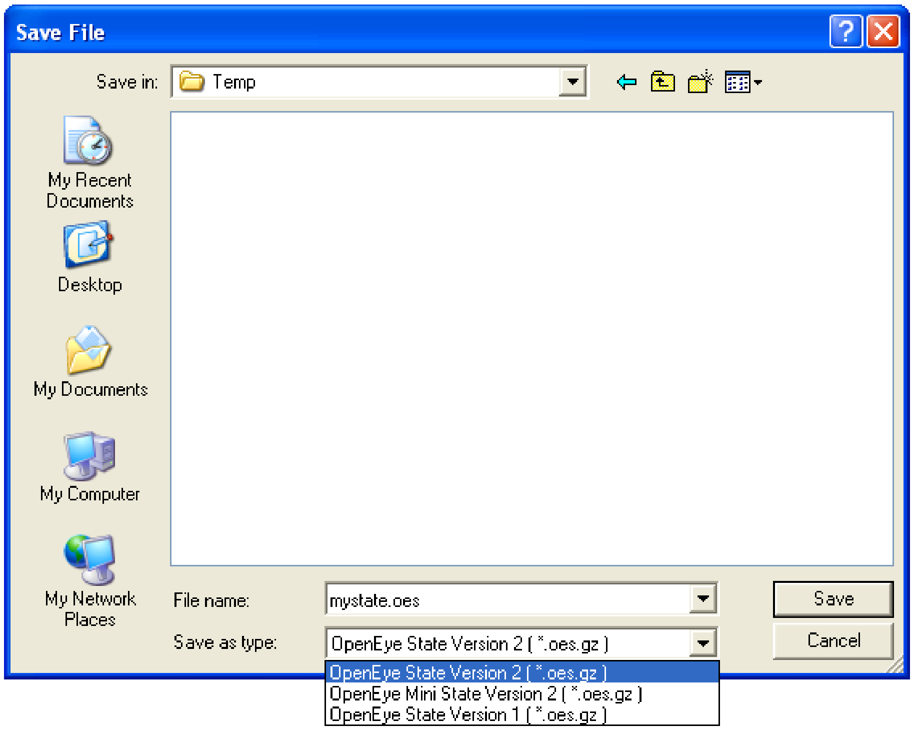

Save File
Save in: Temp
My Recent Documents
Desktop
My Documents
My Computer
My Network Places
File name: mystate.oes
Save as type: OpenEye State Version 2 (*.oes.gz)
OpenEye State Version 2 (*.oes.gz)
OpenEye Mini State Version 2 (*.oes.gz)
OpenEye State Version 1 (*.oes.gz)
Save
Cancel

Figure 4.5: Saving state

With the release of VIDA 3.0, a new state file version was introduced which is significantly smaller in size and faster to load than previous versions. However, this new version, "OpenEye State Version 2" cannot be read or written by VIDA 2.x. To address that issue, AFITT does provide the functionality to save "OpenEye State Version 1" files which can be read and written by all versions of AFITT.

The new state version also supports a "mini" option which only writes out objects that are currently visible or referenced in the display bookmarks. This can provide for an even greater savings in disk size and load time. This mini-state file can be further compressed by selecting the "Compress grid contour information" option in the *General* section of the application preferences. Selecting this option zeros all non-contour related positions in the grid which greatly improves its compressibility. This should only be done if there is no expectation that the user loading the resulting state file will want to look at grid contour levels other than those already defined when the file was generated.

To save a state file, select the "Save State" option in the *File* menu and specify the desired state file. The state file version can be selected using the file format filter as seen in Figure 4.5.

4.4 Importing Files

External spreadsheet data can be imported into AFITT and associated with currently loaded objects. Both comma separated files (.csv) and tab delimited files (.txt) are supported by default. Once a file has been selected, an import dialog will appear which allows the user to customize the column delimiters, column headers, as well as the criteria on which rows are matched to currently loaded objects. For a more specific details on importing spreadsheet data, please see Section 11.9.

4.5 Exporting Files

Multiple special file types can be exported from AFITT including spreadsheet data, images, and scripts. Exact details about these files can be found in the relevant sections below.

4.5.1 Data

Spreadsheet data can be exported in two formats: comma separated files (.csv) and tab delimited files (.txt). Data can be exported by selecting the "Spreadsheet" option in the *Export* submenu in the top-level *File* menu. Selecting this option will launch a file dialog which allows the user to specify the output file. Once the desired file has been chosen, another dialog will appear which allows the user to select which spreadsheet and which columns in that spreadsheet should be exported. For more specific details on exporting spreadsheet data please see Section 11.9.

4.5.2 Images

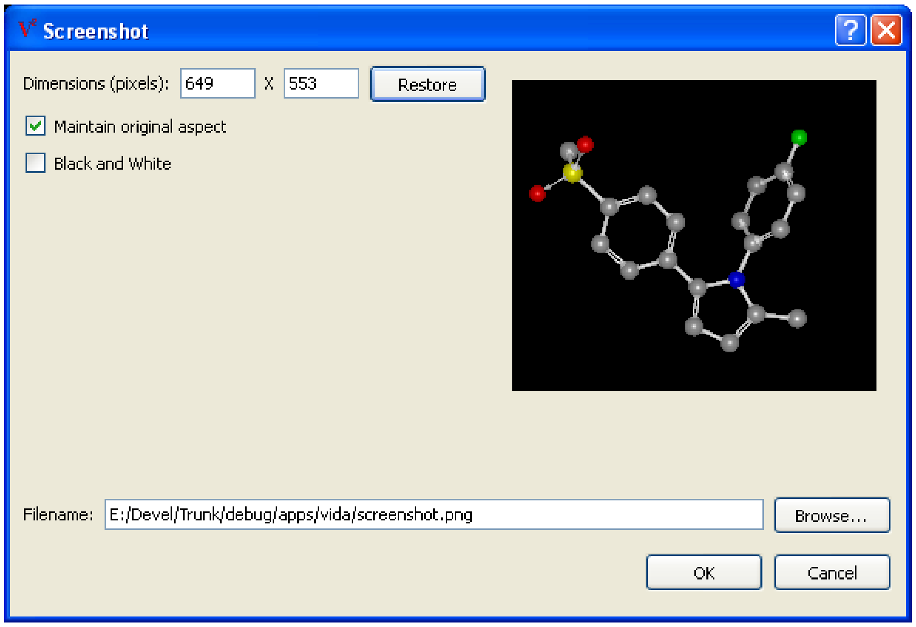
Screenshot

A screenshot of the current main window (see Section 3.5) can be exported in an variety of image formats to a user-specified file by selecting the "Screenshot" option in the *Export* submenu in the *File* menu. This action can also be achieved by clicking on the button with the camera icon in the application toolbar.

Selecting this option will launch a dialog showing a preview of the screenshot to be taken as seen in Figure 4.6. The user can also adjust the desired image resolution as well as specify whether or not to capture the image in black and white (which can be useful for publication purposes). The output file is specified at the bottom of the dialog.

POV-Ray

The scene in the 3D display can be exported to a POV-Ray input file which can be used to generate very high quality and resolution images. To export a POV-Ray file, select the "POV-Ray" option in the *Export* submenu in the *File* menu. The application will then prompt the user to specify a file to which the scene will be saved.



Dimensions (pixels): 649 X 553

[✓] Maintain original aspect

[ ] Black and White

Filename: E:/Devel/Trunk/debug/apps/vida/screenshot.png

Figure 4.6: Capturing a screenshot

4.5.3 Python

A complete history of the application's operations can be written out as a Python script. This script can be exported by selecting the "Script history" option in the *Export* submenu in the *File* menu.

4.6 Drag and Drop

A powerful drag and drop interface is provided for passing molecules, images of molecules, and molecular data between applications. Multiple mechanisms of transfering data via the drag and drop interface are supported.

- Dragging a list of filenames into the List Window (see Chapter 12) will cause those files to be loaded.
- Dragging a list of SMILES into the List Window will cause those SMILES to be parsed into new molecules and stored in a new list called "Pasted". These new molecules will not be assigned 3D coordinates and therefore will not be visible in the 3D display (see Chapter 5).
- Dragging a list of IUPAC or common chemical names into the List Window will cause those names to be parsed into new molecules (using Lexichem) and stored in a new list called "Pasted". Please note, this functionality requires a separate license for the Lexichem toolkit available from OpenEye. These new molecules will not be assigned 3D coordinates and therefore will not be visible in the 3D display (see Chapter 5).

- Dragging molecules from other molecular visualization programs into the List Window will cause those molecules to be loaded into a new list called "Pasted". The ability to load molecules from other applications depends on how the other applications place molecules on the clipboard. Currently supported clipboard formats include SMILES, ChemDraw CDX, ISIS Sketch, MDL SD Files, and OpenEye OEB files. Furthermore, for other molecular visualization programs that support a drag and drop interface, molecules can be dragged from the List Window into those applications.
- Dragging molecules from the List Window into other applications will cause molecules, molecular data, or molecular images to be transfered to that application depending on how that application expects to receive data. For instance, dragging molecules into a Text Editor will result in a list of SMILES corresponding to the dragged molecules to be entered into that application.
- Dragging a 2D depiction from the Spreadsheet (see Chapter 11) will cause that image to be loaded in the associated application if possible.

Along the same vein, a similar copy and paste interface exists for passing molecules, images of molecules, and molecular data between applications. The "Copy" option can be found in the *Edit* menu and can also be performed by pressing *Ctrl+C* on the keyboard. This will copy all of the molecules in the current scope to the clipboard, unless the active window is the Spreadsheet (see Chapter 11) in which case it will copy all of the selected Spreadsheet data to the clipboard instead.

The "Paste" option can be found in the *Edit* menu and can also be performed by pressing *Ctrl+V* on the keyboard. This will paste any molecules found on the clipboard into a new list called "Pasted".

CHAPTER

FIVE

3D Display

The 3D display is the primary visualization interface available to the user (although it is not the only one). The purpose of the 3D display is to allow the user to view and interact with molecules (see Section 5.9), grids (see Section 5.10), and surfaces (see Section 5.11).

5.1 User Interaction

The primary interface mechanism to the 3D display is the mouse which is discussed in detail in Section 5.1.1. However, there are a large number of potential operations that can be performed in the 3D display which necessitates an additional mechanism of interaction besides the simple use of mouse buttons and popup menus. For this reason a separate peripheral window (see Section 3.5.2) is provided to access these operations. This window is called the “Style Control” and is discussed below in Section 5.1.2.

5.1.1 Mouse

As described above, the mouse is considered to be the primary interface mechanism to the 3D display. The details of how to use the mouse to interact with the 3D display are listed here. A three (or two) button mouse is recommended for use; however, support for a single button mouse is available. Holding down the *Ctrl* key while clicking or pressing the mouse button will emulate using the *Right* mouse button on a three button mouse.

For users not comfortable or familiar with the mouse interactions described below, AFITT can be made to emulate the mouse behavior of many other applications including: Coot, Insight, Maestro, MOE, O, Quanta, PyMol, RasMol, and Sybyl. The desired mouse map can be set in the application preferences in the *3D Display* section. Please note that not all the functionality described below is available in the emulated mouse modes, nor is all of the functionality of the emulated applications available in AFITT.

Selection

Objects in the display can be selected by clicking on them using the *Left* mouse button. The selection is cleared between button clicks (or by clicking in the background) unless the *Shift* or *Ctrl* key is also held down when clicking. If a single vertex on a surface has been selected, holding down the *Shift* key and selecting an second vertex will select the shortest path of vertices between the two selected end points.

Double-clicking using the *Left* mouse button expands the current selection to the next logical grouping. For instance, double-clicking on a selected atom in a protein will expand the selection to include all of the atoms in the same residue as the original selected atom. Double-clicking on the selected set again will expand the selection to include either the entire chain if there are multiple chains, otherwise it will include the entire molecule.

Selection can also be performed using a lasso style selection method by holding down the *Right* mouse button and moving the mouse in the window to define a selection rectangle (a dotted rectangle will be displayed on the screen as you do this). Releasing the *Right* mouse button will select all of the atoms and bonds inside of the rectangle. The previous selection will be cleared unless the *Shift* or *Ctrl* key is also held down while moving the mouse. Lasso style selection of surfaces and grids is not currently supported.

Additional methods of selection are available in the *Selection* pane of the Style Control (see Section 5.1.2).

Rotation

The scene can be rotated in three dimensions by holding down the *Left* mouse button and moving the mouse in the window. These movements will emulate rotation using a trackball.

The scene can also be rotated around just the Z axis by holding down the *Alt* key and the *Left* mouse button simultaneously.

Translation

Using the mouse, the scene can be translated in the XY plane by holding down the *Shift* key and the *Left* mouse button simultaneously. Translation along the Z axis can be performed by holding down the *Alt* key, *Shift* key, and the *Left* mouse button simultaneously. The scene can also be translated along the Z axis by holding down the *Alt* key while using the mouse wheel.

Using the keyboard, the scene can be translated horizontally in the display by pressing either the *A* key or *D* key which translate to the left and to the right respectively.

Scale/Zoom

The scene can be scaled (or zoomed) by holding down the *Middle* mouse button, the *Left* and *Right* mouse buttons together, or by using the mouse wheel. Any of these three modes will scale the scene. Multiple modes are provided for convenience and to accommodate the many varied mouse configurations in existence.

The scene can also be scaled using the *W* and *S* keys on the keyboard.

Text Scale

The scale of the text displayed in the scene can be controlled by using the mouse wheel while holding down the *Ctrl* key.

Clipping

The position of the near and far clipping (or slabbing) planes can be controlled using the mouse wheel while holding down both the *Ctrl* and *Shift* keys. Both planes are moved simultaneously and mirror each others' positions. It is important to note that slabbing must already be enabled for these operations to actually be performed.

Contour Level

The contour level of the grids in the default scope can be adjusted using the mouse wheel while simultaneously holding down the *Shift* key. Each incremental turn on the mouse wheel corresponds to an increase or decrease in the contour level by 0.1.

Label

Informative labels can be displayed about atoms and bonds (as well as surfaces and grid contours) underlying the current mouse position if the *Ctrl* key is pressed while moving the mouse (no mouse button need be depressed) on most platforms. On the Mac, this behavior is obtained by holding down the *Alt* key instead of the *Ctrl* key.

Additional methods of labeling atoms and bonds are available in the *Style* pane of the Style Control (see Section 5.1.2).

Menu

A context sensitive popup menu can be generated by clicking in the window while holding the *Right* mouse button down. This menu will contain operations that can be performed on those objects associated with the context when the popup menu was generated.

5.1.2 Style Control

The Style Control is a separate peripheral window (see Section 3.5.2) which is provided to enable additional interactions with the 3D display. The Style Control is a vertical window consisting of many individual panes. Each pane provides its own set of functionality. The display of the individual panes can be toggled by clicking on the title bar of the pane of interest. In addition, individual panes can be "torn off" to act as top-level windows by clicking on the "+" icon in the top right of the title bar. When

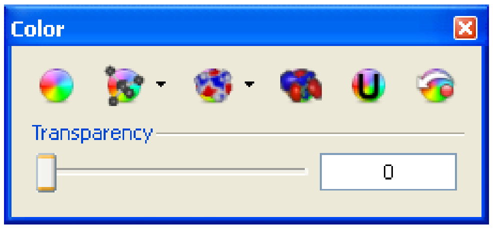


Figure 5.1: Color pane of the Style Control

the pane is torn off the “+” icon becomes a “-” icon which when clicked hides the “torn off” window. The specific details of the individual panes are listed below.

Color

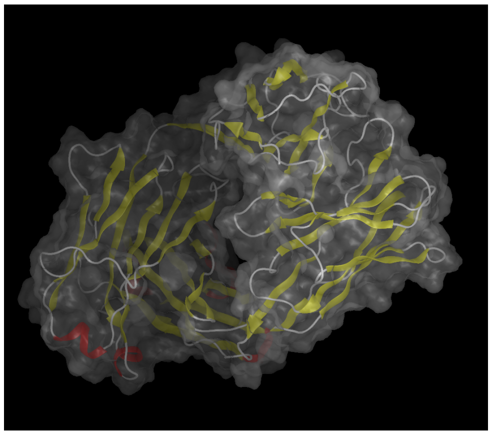
AFITT provides a wide array of coloring options as can be seen in the color pane (see Figure 5.1). The first button (with the color wheel icon) prompts the user to select a specific color which will be applied to all of the objects in the current scope. The second button (with the molecule overlayed on a color wheel) prompts the user to select a specific color that will be applied to all molecules in the current scope. In addition, this button contains a drop down menu of molecule specific coloring schemes which can be applied. Detailed information about the individual color schemes can be found in Section 5.9.2. The third button (with the surface overlayed on a color wheel) prompts the user to select a specific color that will be applied to all surfaces in the current scope. In addition, this button contains a drop down menu of surface specific coloring schemes which can also be applied. Detailed information about the individual color schemes can be found in Section 5.11.2. The fourth button (with the grid overlayed on a color wheel) prompts the user to select a color that will be applied to all grid contours in the current scope. The fifth button (with the “U” icon overlayed on a color wheel) assigns a unique color to every object in the current scope. The last button restores the original color of all the objects in the current scope. More details about the specific coloring schemes can be found below.

Beneath the buttons is a slider which can be used to adjust the transparency of surfaces (see Figure 5.2(a)) and grid contours. The slider values range from 0 (completely opaque) to 100 (completely transparent). The specified transparency value is applied to all surfaces and grid contours in the current scope. However, if a part of a surface is selected, the transparency will only be applied to the selected region as can be seen in Figure 5.2(b).

Selection

The simplest mechanism to perform a selection is to click on the object of interest in the 3D display using the mouse as described in Section 5.11.3. However, a number of addition selection mechanisms are available in the selection pane as can be seen in Figure 5.3.

In the top row of the selection pane, there are seven individual buttons. The first button with the “A” icon selects all of the currently visible molecules. The second button with the partially selected molecule icon performs a selection based on a substructure query. Clicking on this button will launch a query dialog which allows the user to specify a substructure query. The substructure can be entered as a SMARTS


(a) Transparent surface

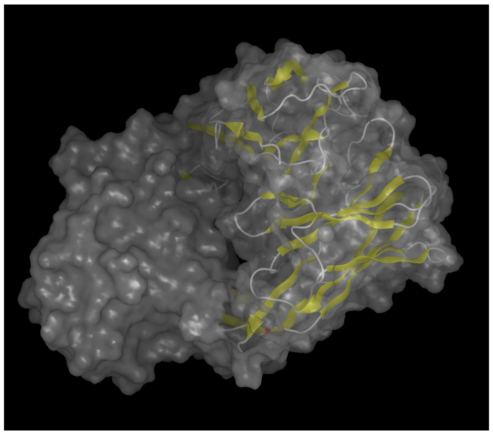

(b) Partially transparent surface

Figure 5.2: Transparent surfaces with underlying ribbon display visible

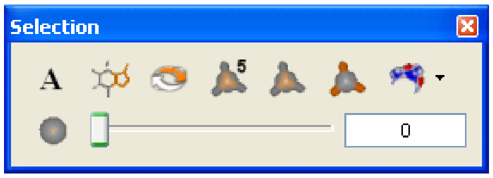


Figure 5.3: Selection pane of the Style Control

pattern, selected from a large number of predefined patterns, or can be specified using a common or IUPAC name which will be converted to a structure using OpenEye's Lexichem toolkit. All atoms in the current scope matching the specified pattern will be selected. If the current scope is set to *All* and more than 50 molecules match the pattern, the user will be prompted as to whether to *Mark* the matching structures instead of selecting them, as being selected would make them visible.

The third button inverts the current selection, unselecting everything that was selected and selecting everything that is currently visible but was not previously selected. The fourth button selects everything within a defined radius of the current selected set (the default is 5 Angstroms). The fifth button selects everything within a user specified radius of the current selected set. This radius is specified by the slider in the bottom row of the pane. The sixth button selects everything outside of the same user specified radius of the current selected set. It is important to note that these distance based selection mechanisms expand their selected sets to include entire residues in the event that only a portion of a residue is included within the radius.

The last button contains a drop down menu which has different options based on the current selection. If a molecule is selected it contains the following options: “Hide Outside”, “Hide Inside”, and “Restore”. Selecting one of these options will hide all of the atoms and bonds outside the current slider specified radius, hide all of the atoms and bonds inside the current slider specified radius, or unhide any previously hidden atoms and bonds respectively. If a surface is the options are: “Crop Outside”, “Crop Inside”, and “Restore”. Selecting one of these options will crop away the unselected portion of the surface, crop away the selected portion of the surface, or restore the surface to its original state respectively. Examples of cropped surfaces can be seen in Figure 5.5.

In the bottom row, there is one button adjacent to a slider and a numeric display. The button is a toggle that controls whether or not unselected atoms and bonds are shown based on their distance to the selected set. The cutoff distance is controlled by the adjacent slider (and the actual value in Angstroms can be seen in the numeric display). This state of this button is ignored when a surface is selected. When a vertex (or a line of vertices) is selected, the slider can be used to flood out from the original selected set to include many more adjacent vertices. The different types of surface selections can be seen in Figure 5.4.

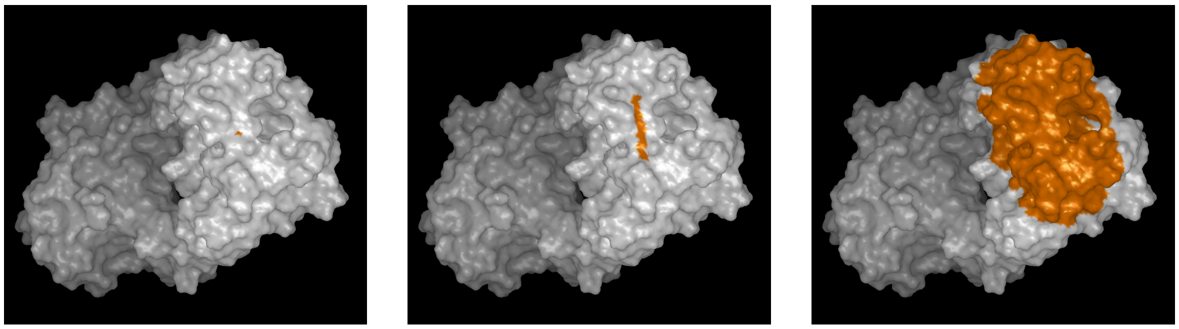

(a) Single vertex selected (b) Line of vertices selected (c) Region of vertices selected

Figure 5.4: Collection of different possible selection states on a surface

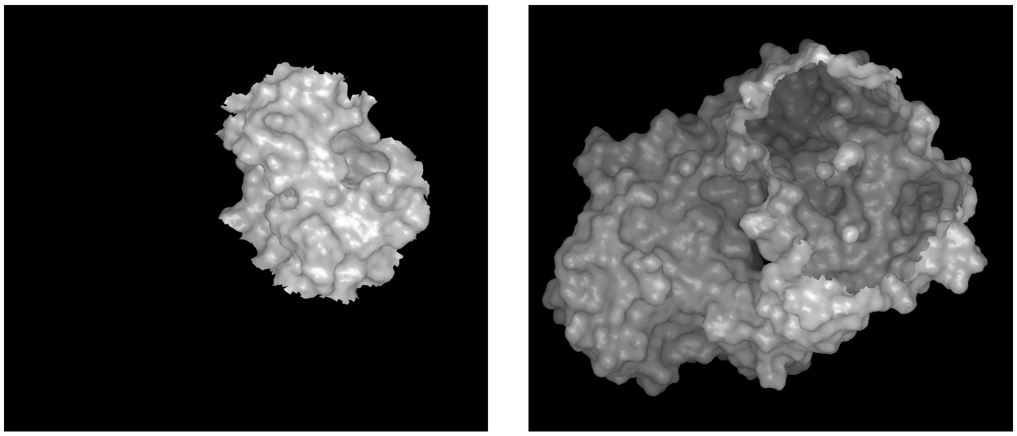

(a) Unselected region removed (b) Selected region removed

Figure 5.5: Different styles of cropping a surface

Style

The style pane of the Style Control is divided into two distinct areas as can be seen in Figure 5.6. The area above the dividing line contains a row of four buttons which control the display style of the various different types of objects that can be visualized in AFITT. The area beneath the dividing line is a collection of buttons whose functionality is specific to just molecules.

In the top area, the first button (with the molecule icon) contains a drop down menu of the available molecule display styles: *ball and stick*, *CPK*, *stars*, *stick*, *wireframe*, and *hidden*. Selecting one of these

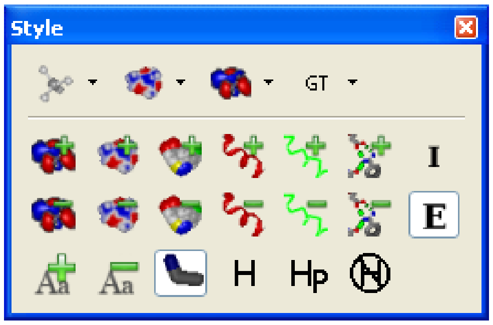


Figure 5.6: Style pane of the Style Control

options will change the display style of all the atoms and bonds in the current scope. For more details on these individual styles, see Section 5.9.1.

The second button (with the surface icon) contains a drop down menu of the available surface display styles: *solid*, *mesh*, and *points*. Selecting one of these options will change the display style of all the surfaces in the current scope. For more details on these individual styles, see section 5.11.1.

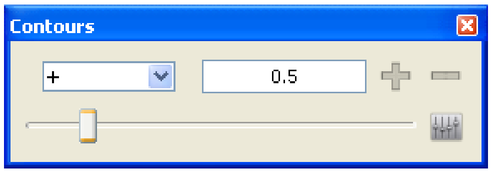
The third button (with the grid icon) contains a drop down menu of the available grid display styles: *solid*, *line*, and *cloud*. Selecting one of these options will change the display style of all the grid contours in the current scope. For more details on these individual styles, see Section 5.10.

The fourth button contains a drop down menu of the supported grid types in AFITT: *Electrostatic*, *ET*, *FRED*, *Generic*, *Difference Map*, and *Regular Map*. Selecting one of these options will change the grid type of all the grids in the current scope. A grid's type determines how it is visualized in AFITT including the default number of contours (and their levels) and the display style as well as the color of those contours. For more details on the individual grid types, see Section 5.10.1.

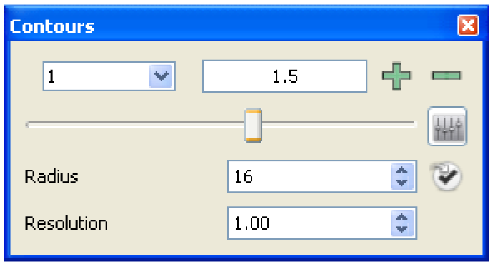
In the bottom area, there are three individual rows of buttons. In the first row, the first three buttons turn on the display of electrostatic grids, molecular surfaces, and accessible surfaces respectively. These specific grids and surfaces are considered display properties of their associated molecules and therefore are not displayed in the List Window (see Chapter 12). Independent (non-property) versions of these can be created from the relevant options in a right-click menu (see Section 5.1.1). The next two buttons turn on the display of protein ribbons and c-alpha traces respectively (for more information about these protein specific displays, see Section 5.9.3). The next button turns on the display of hydrogen bonds by making all of the molecules in the current scope to be hydrogen bond targets. Being a hydrogen bond target means that when it is visible, it displays any hydrogen bonds made between it and any of the other molecules that are also visible. The last button in the row toggles whether or not internal (intramolecular) hydrogen bonds are shown.

In the second row, the first three buttons turn off the display of electrostatic grids, molecular surfaces, and accessible surfaces respectively. The next two buttons turn off the display of protein ribbons and c-alpha traces respectively. The next button makes all of the molecules in the current scope no longer hydrogen bond targets. The last button in the row toggles whether or not external (intermolecular) hydrogen bonds are shown.

In the last row, the first button allows the user to specify and turn on atom and bond labels. The second button turns off all of the atom and bond labels in the current scope. The third button toggles the dis-



(a) Normal View



Radius 16

Resolution 1.00

(b) Advanced View

Figure 5.7: Contours pane of the Style Control

play of non-bonded atoms in the scene. The remaining three buttons control the hydrogen style for the molecules in the current scope. The first of the three turns all hydrogens on, the second buttons shows only polar hydrogens, and the last button hides all hydrogens.

Contours

The contour pane of the Style Control is used to control the number and level of the individual grid contours as can be seen in Figure 5.7(a). When the current application scope is set to *Focused* there will be a pulldown menu in the top left of the window containing a list of all the current grid contours. If the scope is not set to *Focused*, the scope will be displayed in place of the pulldown menu to remind users that all operations in this window apply to more than just the *Focused* grid. Next to this area is a numeric display which displays the current contour level. Next to this display are two buttons, one with a "+" icon and one with a "-" icon. These two buttons allow for the creation and deletion of individual grid contours. Please note that these options are disabled for both *Electrostatic* and *ET* grids as they can have only two contours: positive and negative. Furthermore, the contour levels of the positive and negative contours are coupled together and so changing one will change the other (but with the opposite sign).

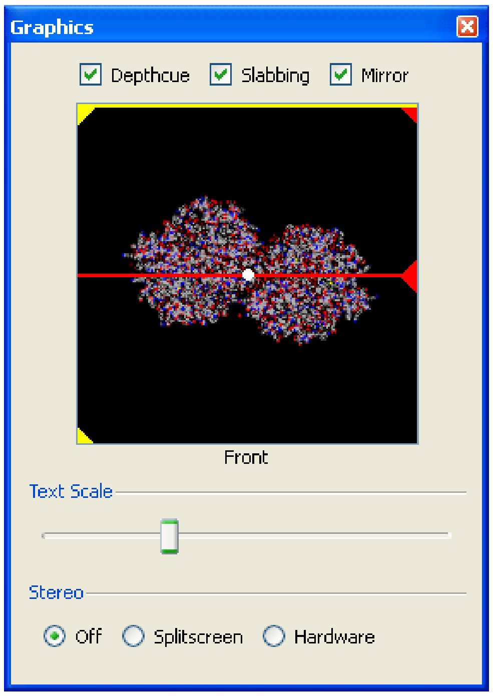
At the bottom of this window is a slider which allows for direct control over the contour level. To the right of the slider is an *Advanced* button (with the equalizer icon) which will expand the pane to show a number of advanced options (see Figure 5.7(b)).

In the advanced section, there are two adjustable parameters (*Radius* and *Resolution*) and one button. The parameters control how contours of *Reentrant* grids are displayed according to their symmetry. The *Radius* parameter specifies how far out from the center of the scene (in Angstroms) AFITT should look in the grid in order to generate of the isocontour. The *Resolution* parameter specifies the resolution (in Angstroms) of the sub-grid used to generate the isocontours.

The single button (with the check icon) to the right of the *Radius* parameter converts the current grid contour into a fixed surface independent of the grid. This can be particularly useful when generating state files as a single surface can take up considerably less disk space than an entire grid.

Graphics

The graphics control pane in the Style Control provides an additional level of control over and interaction with the the 3D display (see Figure 5.8). It is particularly useful for users without mouse wheels.



[✓ Depthcue] [✓ Slabbing] [✓ Mirror]

Stereo: [✓ Off] Splitscreen Hardware

Figure 5.8: Graphics pane of the Style Control

At the top of the graphics pane are three check boxes: “Depthcue”, “Slabbing”, and “Mirror”. The first two check boxes toggle the use of their associated properties respectively. The “Mirror” check box is only enabled when slabbing is on and controls whether the front and back slabbing planes move together (mirrored around the center of the scene) or independently.

Beneath these check boxes is an interactive control which allows the user easy control over both depthcueing and the clipping planes. Displayed in the control is a top-down view of the current scene in the 3D display. The front of the scene is at the bottom as indicated by the label.

When depthcueing is enabled, two red lines will appear in the control which indicate the start and the end points in the depthcueing calculation. These lines can be moved by clicking on the triangular tab at the right of the lines and dragging them to the desired location. The 3D display will update dynamically as these lines are moved.

When clipping (or slabbing) is enabled, two yellow lines will appear in the control which indicate the position of the near and far clipping planes. These can be moved by simply clicking on the triangular tab at the left of the lines and dragging them to the desired location. The 3D display will update dynamically as these lines are moved.

In the center of the control is a small white dot that corresponds to the center of scene that the camera is pointing at. The position of the center can be changed in the Z plane by dragging the dot in the vertical plane. The position of the center can be changed in the X plane by dragging the dot in the horizontal plane.

Beneath this control is a slider which controls the scale of the text drawn in the 3D display. The 3D display will update dynamically as the slider is adjusted.

Beneath the slider are three radio buttons which control the mode of stereoscopic visualization. The default behavior is “Off”. The “Hardware” option is only enabled on machines which are capable of performing 3D hardware stereo-in-a-window. This capability is determined by the computer’s graphics card. For more details on stereo display, please see Section 5.3.

5.2 Rendering

The images shown in the 3D display are rendered using OpenGL, therefore the quality and speed of the rendered scenes in the 3D display is dependent on the graphics card in the computer running the application. Computers with graphics cards that do not support OpenGL acceleration will still be able to display images, but the quality and speed with which the images are displayed may be markedly reduced. To help alleviate situations like this, multiple “levels of detail” are supported ranging from “Fastest” to “Best” which adjust the quality of the rendered scenes accordingly. For most computers purchased within the past year, “Best” level of detail should be appropriate.

In addition to the “level of detail”, the scene can be further adjusted by modifying a few select OpenGL parameters such as the background color as well as the lighting and material models. All of these parameters can be adjusted in the application preferences (see Chapter 16).

5.3 Stereo

Both hardware and split-screen stereo displays are supported in the 3D display. Hardware stereo requires a graphics card that supports “stereo in a window” display as well as the appropriate stereo glasses. The current version of hardware stereo has been successfully tested on Microsoft Windows, Mac OS X, and SGI Irix using CrystalEyes glasses (from StereoGraphics).

In addition to the stereo-capable glasses, a CRT monitor with a high enough refresh rate is required. We recommend a refresh rate of at least 100 Hz (with 120 Hz being preferred) as the effective refresh rate of the monitor is halved due to the fact that the monitor has to swap two different scenes back and forth to create the stereo effect. Unfortunately, CRT monitors (in particular those with high refresh rates) are becoming increasingly difficult to obtain.

For those machines that do not support hardware stereo, split-sreen stereo is still a viable option. Both cross-eyed and wall-eyed views are available. Furthermore, the stereo angle and eye offset parameters can be adjusted in the preferences (see Chapter 16).

5.3.1 Microsoft Windows

We have every expectation that most high-end workstation-class graphics cards (e.g. 3DLabs Oxygen and Wildcat, NVIDIA Quadro, ATI FireGL) will support hardware stereo but we have not been able to test all of them. Furthermore, no testing has been done on Vista and as such, no guarantees can be made about stereo visualization on Vista.

One card that we did test which caused problems was the ATI FireGL V3200. There were no problems observed with the ATI FireGL V5000.

5.3.2 Mac OS X

Mac OS X has supported “stereo in a window” visualization since late 2005 and as such hardware stereo is possible on a Mac, provided that the computer has an appropriate graphics card.

We have successfully test hardward stereo on a Mac Pro configured with the NVIDIA Quadro FX 4500 graphics card (running both Mac OS X and Microsoft Windows via Boot Camp).

5.3.3 UNIX/Linux

Currently, the only UNIX platform (that we know of) that supports hardware stereo is SGI Irix. We have successfully tested hardware stereo on a number of different SGI machines. It is important to note that in order to properly view stereo on an SGI, the screen resolution must be set to a value that is appended with the letter ‘s’. For instance 1200x1024 would likely not behave correctly, whereas 1200x1024s would.

5.4 Viewpoint

When viewing objects in the 3D display, there is always a center to the scene where the virtual “camera” is pointing. When the first molecule, grid, or surface is loaded into the application, the center of that object becomes the center of the scene where the camera is pointing. By default, the center does not automatically change with changes to the scene. The center can be changed explicitly from the main application toolbar, the right-click popup menu, or using the following scripting commands: `ViewerCenterSet` or `ViewerCenterAndRadiusSet`. Furthermore, the center can also be translated using the mouse interactions described in Section 5.1.1.

The center can be made to automatically change when browsing if the “Auto-Center” or “Auto-Fit” preferences are set to true (the default is false). The “Auto-Center” preference directs the window to always recenter the scene after any changes are made to that scene. The “Auto-Fit” preference directs the window to not only recenter the scene after any changes are made but to also ensure that everything in the scene is comletely visible in the window. These preferences can be edited in the applications preferences dialog (see Chapter 16).

5.5 Bookmarks

The 3D display is capable of capturing and storing named bookmarks of the current display scene to enable viewing again at another time without having to set the scene up again by hand. Bookmarks are saved in state files and are particularly useful when trying to communicate multiple pieces of information to other users.

All bookmarks have names and are stored in order. Bookmarks can be created from the top-level *Bookmark* menu by selecting the “Add” option. The order of the bookmarks can be organized by selecting the “Organize” option in the same menu. The *Bookmark* menu also contains an “Animated” option which controls whether AFITT animates the transition between the current scene and the newly selected bookmark. A list of all the currently available bookmarks populate the rest of the menu. Bookmarks can also be created by clicking on the “Add Bookmark” button in the 3D display *Style* toolbar (see Section 5.7.1).

The 3D display also provides a bookmark display widget (see Section 5.8.2) which displays a clickable list of all the available bookmarks directly in the 3D display.

5.6 Tiled Display

The default behavior of the 3D display is to display everything that is *Visible* (see Section 3.3.2) in the same coordinate space; however, there are often occasions when it is desirable to view everything in their own space. This behavior can be obtained in “Tiled” mode, also called “Matrix” mode or “Multi-Pane” mode. “Tiled” mode can be toggled on or off from the main application toolbar.

When the 3D display is in “Tiled” mode every *Visible* and *Focused* object is displayed centered in its own individual pane. However, if desired each object can be shown relative to the same center if the “Auto-Center Panes” option in the *3D Display* section of the application preferences is unchecked. It is important to note that when “Auto-Center Panes” is enabled it is not possible to recenter or translate using mouse interactions in any of the panes.

To indicate which object is *Focused* (see Section 3.3.1), the corresponding pane will contain a blue border just inside the regular pane border. If there are any *Locked* objects (see Section 3.3.3), each of them will be displayed in every pane in addition to the individual *Visible* and *Focused* objects.

5.7 Toolbars

The 3D display provides additional toolbar areas (aside from the main application toolbar) inside the 3D window. There are four possible toolbar locations: at the top of the window, at the bottom of the window, at the left edge of the window and at the right edge of the window. Toolbars in these locations are hidden by default, but popup when the mouse is moved over the toolbar area and disappear when the mouse leaves. These toolbars can be made sticky by clicking on the button with “+” icon in it.

By default, the 3D display provides a "Style" toolbar at the top of the display and a "Mouse" toolbar at the left hand edge. Other toolbars exist in the bottom and right areas ("3D Bottom" and "3D Right" respectively) but are empty and hidden by default. They can be shown and populated via the use of scripting commands (see Chapter 14).

5.7.1 Style Toolbar

The *Style* toolbar contains a single row of buttons which are primarily used to toggle the visibility of the 3D display widgets (see Section 5.8) as well as the torn-off Style Control panes (see section 5.1.2). However, the second button (with the flagged folder icon) creates a new named display bookmark (see Section 5.5) based on the current scene and the last button toggles the use of stereoscopic visualization (see Section 5.3).

The buttons in the middle are organized into two groups. The first group contains four buttons which toggle the display of the annotation widget, the 2D depiction widget, the data display widget, and the bookmark widget respectively. The second group contains five buttons which toggle the display of the torn-off Style Control panes: color, selection, style, contours, and graphics respectively.

This toolbar can be referenced in scripting commands using the name "Style".

5.7.2 Mouse Toolbar

The *Mouse* toolbar contains a single row of buttons which control the behavior of the mouse in the 3D display. Ignoring the "sticky" button, the first button puts the mouse into the default mode which obeys all of the expected interaction mechanisms as described in Section 5.1.1. The second button (with the info icon) changes the behavior of the mouse such that an informative label is displayed on the screen regarding whatever is currently underneath the mouse cursor. The remaining three buttons put the mouse into one of three measurement modes: *Distance*, *Angle*, and *Torsion* respectively. More details about measurement and the monitors generated by the measurements can be found in Section 5.9.4.

This toolbar can be reference in scripting commands using the name "Mouse".

5.7.3 Application Toolbar

In addition to providing its own toolbars, the 3D display is affected by a few of the buttons in the main application toolbar. The application toolbar contains the "Tiled" display toggle button (matrix icon), the main window screenshot button (camera icon), the centering button (four arrows pointing in), and the fitting button (four arrows pointing out).

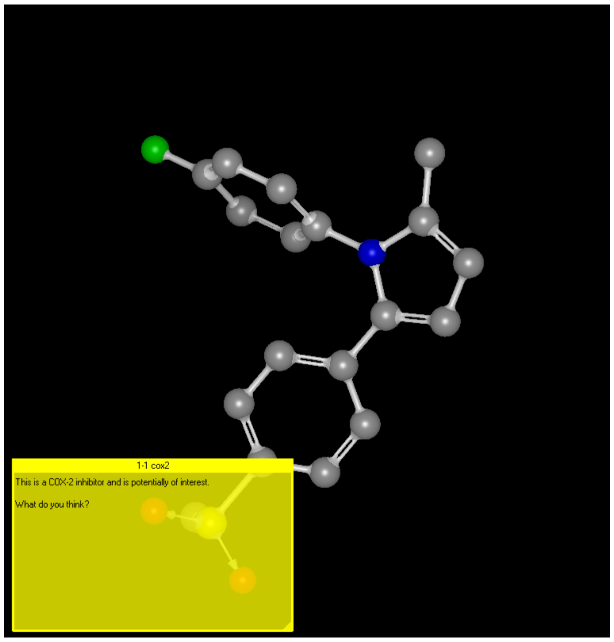
The "Tiled" display button toggles the display between single pane overlay mode and multiple pane tiled mode. For more details see Section 5.6. The screenshot button captures an image of the current main window at a user specified resolution. The centering button centers the display on the currently selected set or whatever molecule(s) are in the current scope. The fit button centers the scene and adjusts the scale factor such that all *Visible* objects can be seen on the screen at the same time.

5.8 Display Widgets

The 3D display provides a number of additional widgets which are drawn directly in 3D display as opposed to on the side or floating above it. These widgets include an annotation widget, a bookmark widget, a data display widget, and a depiction widget.

5.8.1 Annotation

It is often desirable when viewing molecules to make notes or simple annotations about specific molecules. This can be done directly within the 3D display, where the annotation widget appears like a transparent Post-It^{TM} note (see Figure 5.9). The color of the widget as well as the font used can be specified in the application preferences.



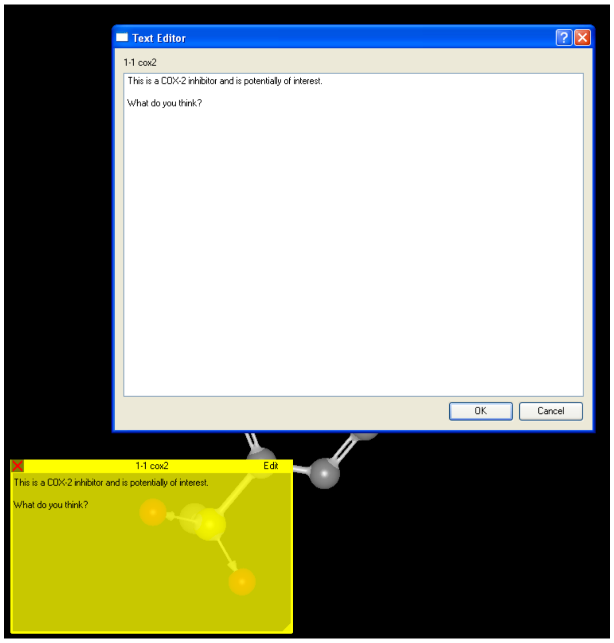
This is a COX-2 inhibitor and is potentially of interest.

What do you think?

Figure 5.9: Molecule with annotation

The widget can be moved by clicking on the title bar and dragging it to the desired location. The widget can be resized by grabbing the small triangular tab in the lower right-hand corner of the widget and dragging until the desired size is achieved.

Clicking in the annotation widget will activate it for editing. Editing can be accomplished by directly typing once the widget has been clicked or by clicking on the "Edit" button in the title bar. Clicking on the “Edit” button will launch a more fully featured text editor which supports copy and paste operations (see Figure 5.10).



This is a COX-2 inhibitor and is potentially of interest.

What do you think?

Figure 5.10: Editing the annotation

Annotations are also stored in the spreadsheet (see Chapter 11) and can be edited there as well as in the 3D display. Annotations are saved as a SD tag when a molecule is written out (in either SD or OEB formats).

5.8.2 Bookmark

The bookmark widget is a simple widget drawn at the bottom of the 3D display containing a clickable list of the currently available bookmarks. Clicking on any individual bookmark will load that bookmark and update the display accordingly. For more information on bookmarks, see Section 5.5.

5.8.3 Data Display

This widget provides a simple table which contains the spreadsheet data of the *Focused* object. The widget can be moved around the 3D display by clicking and dragging it to the desired location. It can also be resized by clicking in the tab in the upper left hand corner. Lastly, if the amount of data associated with the object is too large to be shown all at once, the widget does provide scrolling functionality.

5.8.4 Depiction Widget

The depiction widget draws a 2D depiction of the focused molecule directly into the 3D display. The depiction is drawn in the lower left hand corner of the display (or individual pane if in tiled mode). The depiction style is determined based on the application 2D depiction preferences. The size and other drawing parameters (e.g. anti-aliasing) of the depiction are also specified in the applicaton preferences.

5.9 Molecular Visualization

This section describes the very powerful, versatile, and interactive molecular visualization capabilities of the 3D display. The interactive control of molecular visualization is typically performed using the *Style Control* (see Section 5.1.2) or through scripting commands (for more details on scripting see Chapter 14).

5.9.1 Display Styles

All of the standard display styles are available (*Wireframe*, *Stick*, *Ball and Stick*, *CPK*, and *Stars*) and can be seen in Figure 5.11.

By default, small molecules are drawn in *Stick* mode while large molecules are drawn in *Wireframe* mode. These defaults can be changed in the application preferences (see Chapter 16).

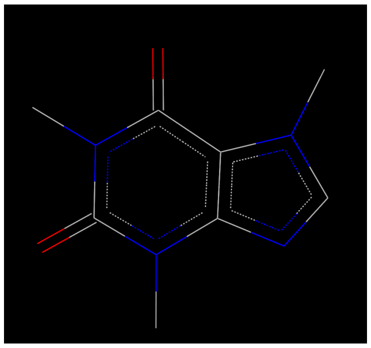
5.9.2 Color

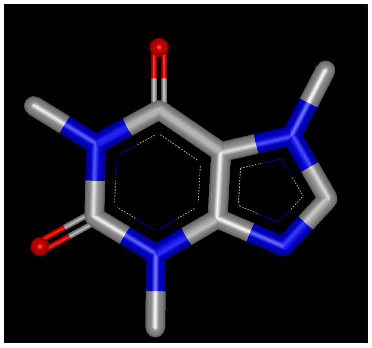
By default, the atoms in a molecule are colored according to their element types and the bonds in a molecule are colored according to the element types of the two atoms defining that bond. There are two color palettes that specify the actual colors used for a given element, one for use with dark-colored backgrounds and the other for use with light-colored backgrounds. Both of these palettes can be viewed and modified in the application preferences (see Chapter 16). The dark-background atom color palette can be seen in Figure 5.12(a).

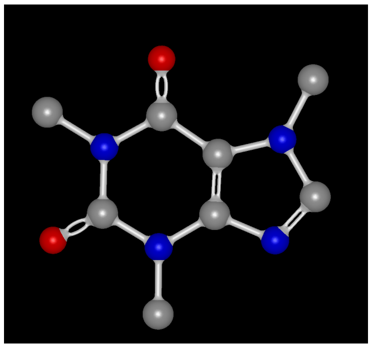
Coloring Schemes

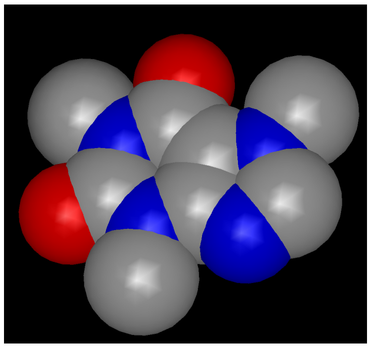
There are a number of molecule specific coloring schemes in addition to the standard element based coloring. The following schemes are discussed below: amino, bfactor, carbon, chain, cpk, cpknew, element, formal charge, group, partial charge, reference, residue, and shapely.

Amino


(a) Wireframe


(b) Stick


(c) Fancy Ball and Stick


(d) CPK

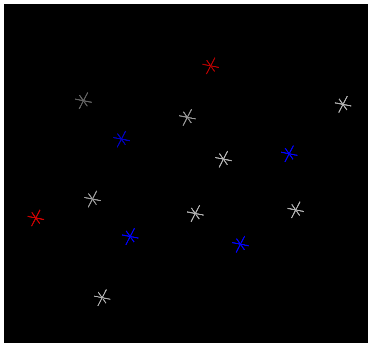
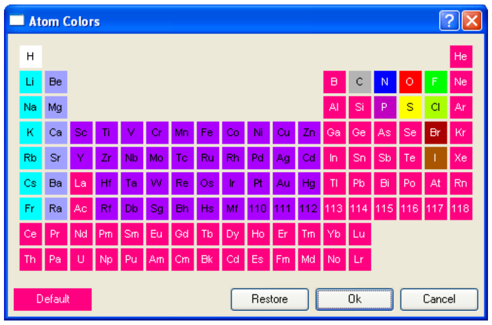

(e) Stars

Figure 5.11: Molecule rendering styles


Atom Colors
H He
Li Be B C N O F Ne
Na Mg Al Si P S Cl Ar
K Ca Sc Ti V Cr Mn Fe Co Ni Cu Zn Ga Ge As Se Br Kr
Rb Sr Y Zr Nb Mo Tc Ru Rh Pd Ag Cd In Sn Sb Te I Xe
Cs Ba La Hf Ta W Re Os Ir Pt Au Hg Tl Pb Bi Po At Rn
Fr Ra Ac Rf Db Sg Bh Hs Mt 110 111 112 113 114 115 116 117 118
Ce Pr Nd Pm Sm Eu Gd Tb Dy Ho Er Tm Yb Lu
Th Pa U Np Pu Am Cm Bk Cd Es Fm Md No Lr
Default
Restore
Ok
Cancel
(a) Atom Color Palette

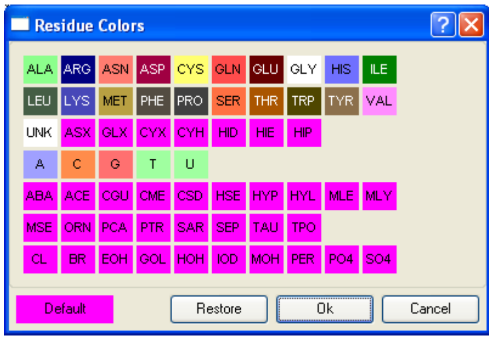

Residue Colors
ALA ARG ASN ASP CYS GLN GLU GLY HIS ILE
LEU LYS MET PHE PRO SER THR TRP TYR VAL
UNK ASX GLX CYX CYH HID HIE HIP
A C G T U
ABA ACE CGU CME CSD HSE HYP HYL MLE MLY
MSE ORN PCA PTR SAR SEP TAU TPO
CL BR EOH GOL HOH IOD MOH PER PO4 SO4
Default
Restore
Ok
Cancel
(b) Residue Color Palette

Figure 5.12: Dark background atom and residue color palettes

The *amino* color scheme is a protein specific scheme which colors atoms according to their individual residues. The residue colors are listed below.

ASP, GLU	Bright Red	[230, 10, 10]
CYS, MET	Yellow	[230, 230, 0]
LYS, ARG	Blue	[20, 90, 255]
SER, THR	Orange	[250, 150, 0]
PHE, TYR	Mid Blue	[50, 50, 170]
ASN, GLN	Cyan	[0, 220, 200]
GLY	Light Grey	[235, 235, 235]
LEU, VAL, ILE	Green	[15, 130, 15]
ALA	Dark Grey	[200, 200, 200]
TRP	Purple	[180, 90, 180]
HIS	Pale Blue	[130, 130, 210]
PRO	Flesh	[220, 150, 130]
Others	Tan	[190, 160, 110]

BFactor

The *bfactor* color scheme is a protein specific scheme which colors atoms according to the bfactor value using a fixed blue-to-red gradient between 0 and 100.

Carbon

The *carbon* color scheme colors all of the carbon atoms in the current scope the same color which is specified by the user.

Chain

The *chain* color scheme is a macromolecule specific scheme which assigns a unique color to each of the macromolecular chains.

CPK, CPKNew

The *CPK* color scheme is an element specific scheme based on the colors used in the popular CPK plastic space filling models. The *CPKnew* color scheme is a variant on the *CPK* color in that it uses a slightly brighter version of some of the colors. The atom colors are listed below with the RGB values for *CPK* in the third column and the RGB values for *CPKNew* in the fourth column (if different).

H	White	[255, 255, 255]	
He	Pink	[255, 192, 203]	
Li	Fire Brick	[178, 34, 34]	[178, 33, 33]
B,Cl	Green	[0, 255, 0]	
C	Light Grey	[200, 200, 200]	[211, 211, 211]
N	Sky Blue	[143, 143, 255]	[135, 206, 235]
O	Red	[255, 0, 0]	
F,Si,Au	Golden Rod	[218, 165, 32]	
Na	Blue	[0, 0, 255]	
Mg	Forest Green	[34, 139, 34]	
Al,Ca,Ti,Cr,Mn,Ag	Dark Grey	[128, 128, 144]	[105, 105, 105]
P,Fe,Ba	Orange	[255, 165, 0]	[255, 170, 0]
S	Yellow	[255, 200, 50]	[255, 255, 0]
Ni,Cu,Zn,Br	Brown	[165, 42, 42]	[128, 40, 40]
I	Purple	[160, 32, 240]	
Unknown	Deep Pink	[255, 20, 147]	[255, 22, 145]

Element

The *element* color scheme is the default color scheme used when coloring molecules. Colors are assigned according to atomic number. The actual colors used for a given element can be edited in the application preferences as seen in Figure 5.12(a).

H	White	[255, 255, 255]
C	Grey	[180, 180, 180]
N	Blue	[0, 0, 255]
O	Red	[255, 0, 255]
F	Green	[0, 255, 0]
P	Magenta	[192, 0, 192]
S	Yellow	[255, 255, 0]
Cl	Lime Green	[170, 255, 0]
Br	Dark Red	[170, 0, 0]
I	Dark Orange	[170, 85, 0]
Group I	Cyan	[0, 255, 255]
Group II	Light Blue	[160, 160, 255]
Transition Metals	Purple	[170, 0, 255]
Others	Pink	[255, 0, 128]

Formal Charge

The *formal charge* color scheme colors atoms according to their formal charge using a fixed red-to-blue gradient between -4 and +4.

Group

The *group* color scheme is a macromolecule specific scheme which colors each atom according to its position in a macromolecular chain. The colors are assigned along a smooth rainbow spectrum from blue to green to yellow to orange to red.

Partial Charge

The *partial charge* color scheme colors atoms according to their partial charge using a red-to-blue gradient between -1.0 and +1.0.

Reference

The *reference* color scheme colors all of the carbon atoms in the current scope using the current *Reference* color which can be set in the application preferences. The default is green.

Residue

The *residue* color scheme is a protein specific scheme which colors atoms according to their individual residues. The actual colors used can be edited in the application preferences as seen in Figure 5.12(b). The default colors correspond to those used in the *shapely* color scheme.

Shapely

The *shapely* color scheme is a protein specific scheme which colors atoms according to their individual residues. This scheme is based upon Bob Fletterick's "Shapely Models" [17]. The residue colors are listed below.

ALA	Medium Green	[140, 255, 140]
GLY	White	[255, 255, 255]
LEU	Olive Green	[69, 94, 69]
SER	Medium Orange	[255, 112, 66]
VAL	Light Purple	[255, 140, 255]
THR	Dark Orange	[184, 76, 0]
LYS	Royal Blue	[71, 71, 184]
ASP	Dark Rose	[160, 0, 66]
ILE	Dark Green	[0, 76, 0]
ASN	Light Salmon	[255, 124, 112]
GLU	Dark Brown	[102, 0, 0]
PRO	Dark Grey	[82, 82, 82]
ARG	Dark Blue	[0, 0, 124]
PHE	Olive Grey	[83, 76, 66]
GLN	Dark Salmon	[255, 76, 76]
TYR	Medium Brown	[140, 112, 76]
HIS	Medium Blue	[112, 112, 255]
CYS	Medium Yellow	[255, 255, 112]
MET	Light Brown	[184, 160, 66]
TRP	Olive Brown	[79, 70, 0]
ASX,GLX,PCA,HYP	Medium Purple	[255, 0, 255]
A	Light Blue	[160, 160, 255]
C	Light Orange	[255, 140, 75]
G	Medium Salmon	[255, 112, 112]
T	Light Green	[160, 255, 160]
Backbone	Light Grey	[184, 184, 184]
Special	Dark Purple	[94, 0, 94]
Default	Medium Purple	[255, 0, 255]

5.9.3 Proteins

In addition to the standard display styles, the two protein specific displays of *C-Alpha Traces* and *Ribbons* are supported. Examples of these display styles can be seen in Figure 5.13.

Given that protein files often inevitably contain many extra components that are not necessarily of interest to the user (waters for example), the ability to hide the display of non-bonded atoms is provided to ease the viewing of proteins without having to manually edit the input file. An example of a protein displayed with and without waters can be seen in Figure 5.14.

Another common scenario when working with proteins is that the input file contains a large multimer which as expected is interpreted as a single molecule. However, sometimes the ability to interact with the individual components is of particular value. The ability to split a molecule into separate components can be performed by selecting the "Components" option in the *From Split* submenu in the *New Molecule* submenu in the top-level *File* menu. This function will then determine what the individual components of the *Focused* molecule are and create a new list containing a new molecule for each component. The

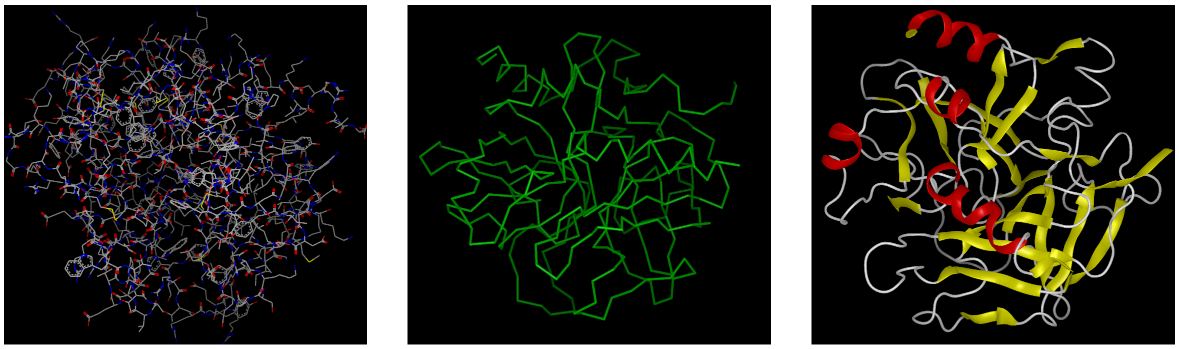

(a) Wireframe (b) C-Alpha Trace (c) Ribbon

Figure 5.13: Different views of the same protein

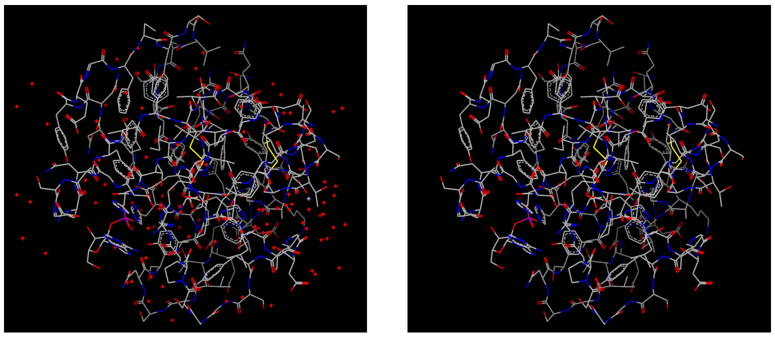

(a) Protein with Waters (b) Protein without Waters

Figure 5.14: Same protein with and without waters

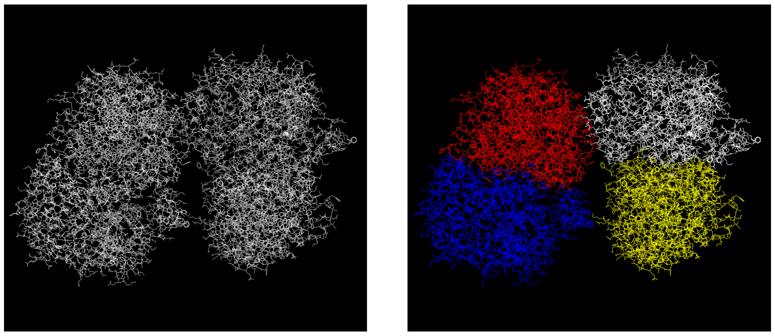

(a) Multimer displayed as a single molecule (b) Multimer displayed as multiple molecules

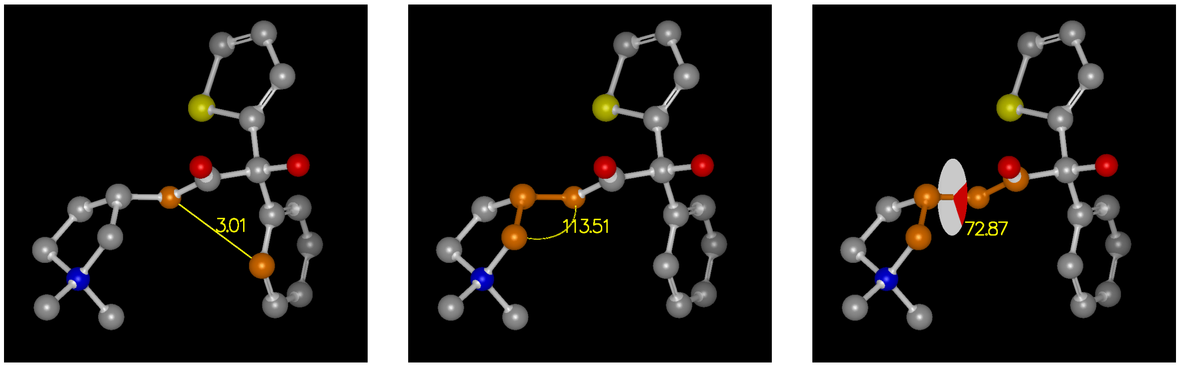
Figure 5.15: Same protein as a single molecule vs as multiple molecules

original molecule is not changed or deleted. An example of this process can be seen in Figure 5.15. The *From Split* submenu also contains “Selected” and “Marked” options which will create new molecules from the original based on the selected or marked set of atoms.

It may also be of interest to put some of these components back together into a single molecule. This can be done by choosing one of the following options from the *From Merge* submenu in the *New Molecule* in the top-level *File* menu: “Selected”, “Marked”, “Visible”. Choosing any one of these options will create a new molecule which is composed of all the molecules that were either “Selected”, “Marked”, or “Visible” at the time of the operation.

5.9.4 Monitors

The ability to display measurement monitors is a useful feature and is well supported. Access to these measurement facilities is available from the *Mouse* popup toolbar on the left hand side of the 3D display. There are three buttons which turn on *Distance*, *Angle*, and *Torsion* measurement respectively.



(a) Distance Monitor (b) Angle Monitor (c) Torsion Monitor

Figure 5.16: Different types of monitors

When the mouse is put into one of these modes, the measurements are made based on the selected set of atoms. In *Distance* mode, the first atom selected is consider the “anchor” atom. Once the anchor atom has been selected, a temporary distance monitor will be displayed to any other atom that the mouse passes over. Selecting another atom will create a permanent monitor and clear the anchor. In *Angle* mode, the first two atoms selected are considered the anchor atoms and a temporary angle monitor will be displayed to any other atom that the mouse passes over. Selecting a third atom will create a permanent monitor and clear the anchors. In *Torsion* mode, the first three atoms selected are considered the anchor atoms and a temporary torsion monitor will be displayed to any other atom that the mouse passes over. Selecting a fourth atom will create a permanent monitor and clear the anchor.

Monitors can be removed by performing a right-click operation in the 3D display and selecting the “Delete visible monitors” option. Monitors are also displayed in the “List Window” and can be deleted individually there (see Chapter 12). Examples of the three types of monitors can be seen in Figure 5.9.4.

5.9.5 Hydrogen Bonds

Both internal (intramolecular) and external (intermolecular) hydrogen bonds can be displayed. Hydrogen bonds are determined by typing all of the atoms as either acceptors, donors, both, or neither. Then the hydrogen bond energy is calculated between all acceptors and donors based on the ChemScore [18] scoring function which takes into account both interatomic distance as well as the coordination geometry. This function provides a relative score between 0 and 1.0 for the energy of the hydrogen bond. For all hydrogen bonds with an energy greater than 0, the bond is drawn as a dotted green line. The thickness of the line as well as the spacing between the dots correlates to the bond energy. Therefore, thicker more solid lines have higher energy than thinner more dotted lines. An example of both internal and external hydrogen bonds can be seen in Figure 5.17.

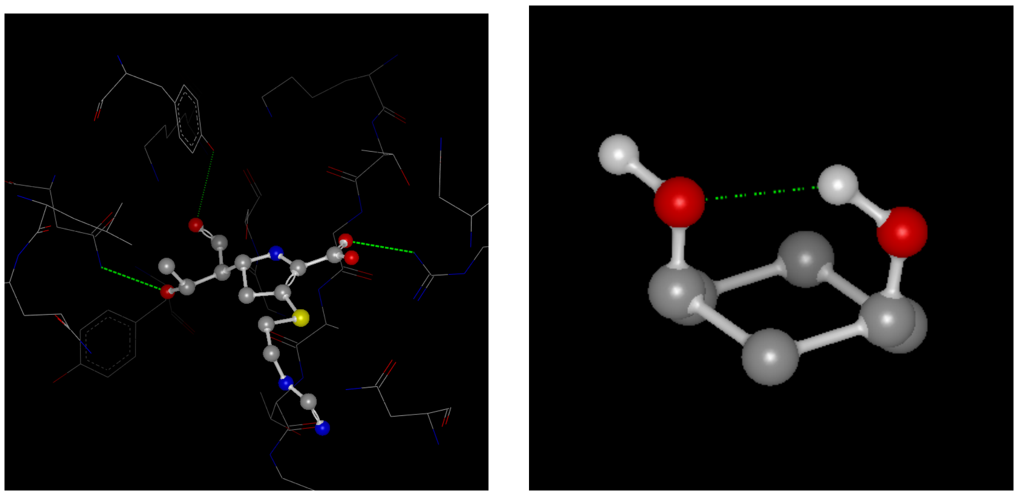

(a) External Hydrogen Bonds (b) Internal Hydrogen Bonds

Figure 5.17: Hydrogen bonds in a protein active site

5.10 Grid Visualization

Grids are spatial arrays of data that are sampled at regular intervals. Grids may contain electrostatic samples of spatial regions, density samples, and other potential fields.

A grid can have a variable number of contours. Each contour has an index starting at 1 for the first contour. Additionally, every contour has its own individual color and a threshold value which can be positive or negative (also known as the "contour level"). There are three different display styles for grids contours: *Solid*, *Mesh* and *Cloud*.

5.10.1 Grid Types

As mentioned in the previous section, there are multiple different predefined grid types that are supported including: *Electrostatic*, *ET*, *FRED*, *Generic*, *Difference Map*, and *Regular Map*.

Electrostatic

Electrostatic grids are the only type of grid that can be actually be generated within AFITT as opposed to simply being visualized. Electrostatic grids are created for individual molecules using OpenEye's Zap toolkit. The creation of the grid requires the presence of partial charges on the input molecule. If partial charges were already present on the molecule, those will be used in the calculation. However, if no partial charges were present (or the user decides to ignore them, which can be done in the application preferences), temporary partial charges will be calculated using MMFF94 or AM1-BCC [19, 20]. If a molecule has greater than 50 atoms, MMFF94 charging will be done even if AM1-BCC was selected. However, for proteins an alternate residue based charging model is available and can be selected in the preferences. This model assigns partial charges based on residue information as is shown below:

ASP	OD1, OD2	-0.5
GLU	OE1, OE2	-0.5
LYS	NZ	+1.0
ARG	NE	+1.0
Other		0.0

The grid resolution is dynamically determined based on the number of atoms in the input molecule. However, it is possible to specify a fixed resolution in the grid preferences. Be warned, however, that specifying very fine resolutions (<2.0 Angstroms) may consume a surprisingly large amount of memory, and in some cases more than may be available on the computer, which can lead to program failure. This should be consider a very advanced feature to be used only when necessary and when all other data has been saved. Finally, it has been observed in research at OpenEye that adding salt to the electrostatic calculation has a good effect on the grids generated. The salt concentration can be specified in the preferences. The default concentration is 0.04 M. The maximum concentration that can be specified is 0.1 M.

Electrostatic grids (as well as ET grids) are special in that they are always generated with two default contours: a positive one and a negative one. These contours cannot be deleted and new contours cannot be added. The positive contour is colored blue by default and the negative contour is colored red. The contour levels of the two contours are coupled together and so changing one will change the other.

ET

ET (or Electrostatic Tanimoto) grids are generated by OpenEye's Brood and EON applications. When viewed in AFITT, they behave and are treated essentially the same as Electrostatic grids (see Section 5.10.1).

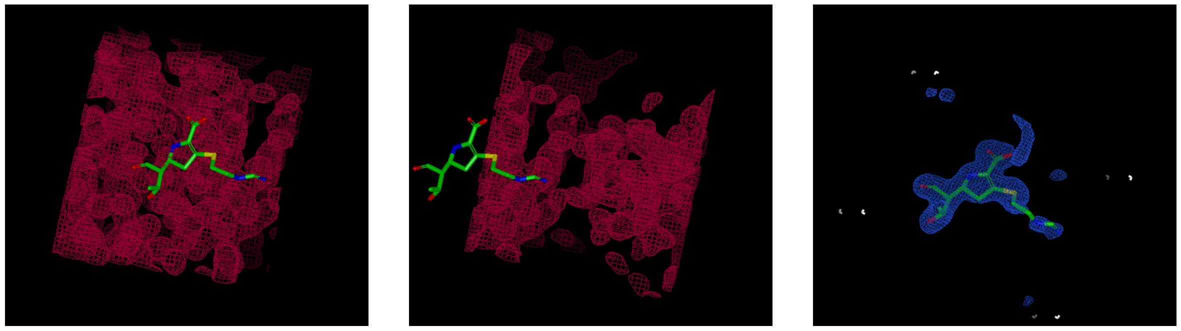
FRED

FRED grids are shape potential grids that are created by OpenEye's FRED application. In AFITT, the behave just like Generic grids (see Section 5.10.1) except that their default contour color is purple instead of blue.

Generic

Generic grids are effectively the most basic grids and act as the primary grid workhorses. They are completely customizable and have bare-bones settings. They default to having just one contour with a default color of blue. Additional contours can be added as well as removed. These default values can be changed in the *Grids* section of the application preferences.

5.10.2 Reentrant Grids



(a) Reentrant (b) Reentrant (c) Non-reentrant

Figure 5.18: Grid contouring examples showing (a) an infinite grid (b) the same grid rendered at a different center, notice that the grid is rendered in the center of the scene, not at the molecule and (c) a finite density grid showing the grid corners or extent of the grid.

Some grids have fixed extents and some grids are infinitely reentrant in space. Reentrant grids obviously cannot be displayed in their entirety, so instead only the region around the center of the current scene is displayed. Grids that are being rendered as reentrant can be distinguished from non-reentrant grids by the absence of grid corners which define the extents of non-reentrant grids.

5.11 Surface Visualization

Surfaces are infinitely thin three dimensional connected regions that represent objects such as molecular or accessible surfaces of molecules. Surfaces can be selected (or scribed), cropped, colored, and made transparent in part or in whole.

5.11.1 Display Styles

Surfaces can be visualized in one of three display styles: Solid, Mesh, and Points. These individual style can be seen in Figure 5.19.

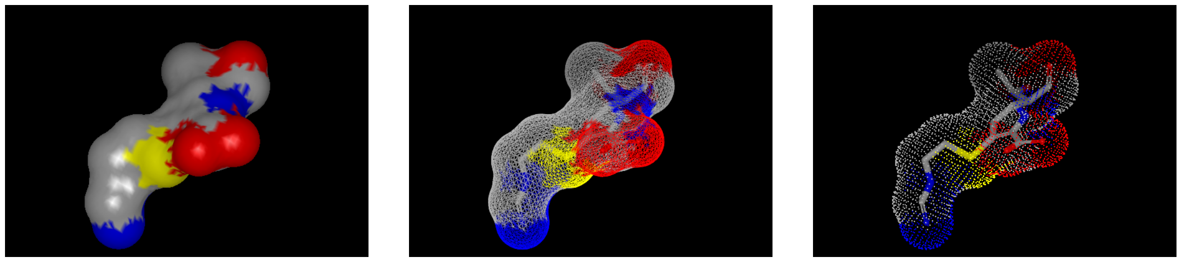

(a) Solid (b) Mesh (c) Points

Figure 5.19: Surface rendering examples showing (a) solid rendering, (b) mesh rendering, and (c) point rendering.

5.11.2 Color

There are a number of surface specific coloring schemes in addition to the standard single color scheme. The following schemes are discussed below: atom color, concavity, curvature, distance, electrostatics, grid, hydrogen bond potential, hydrophobicity, and surface potential.

Atom Color

The *atom color* scheme colors each vertex on the surface using the color of the nearest atom to that vertex in the molecule that the surface was created from (see Figure 5.20(b)). This scheme will not work if the surface was not created from a molecule and also if that molecule is not currently present.

Concavity

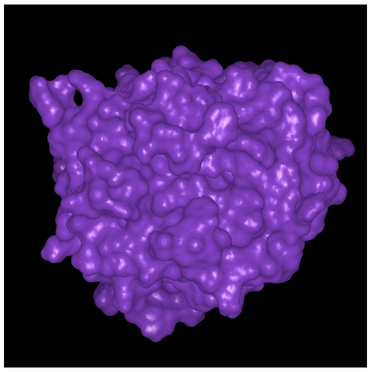
The *concavity* scheme colors each vertex on the surface according to how concave the surface is at that vertex using a red-to-white gradient where red indicates a region of high concavity (see Figure 5.20(c)).

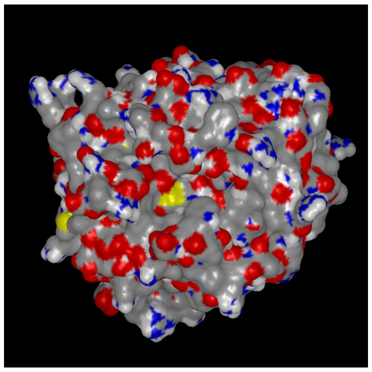
Curvature

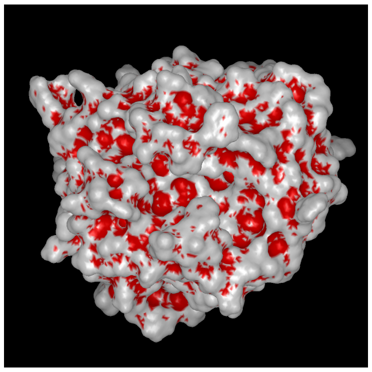
The *curvature* scheme colors each vertex on the surface according to the curvature of the surface at that vertex using a grey-to-green gradient where green indicates a region of high curvature (see Figure 5.20(d)).

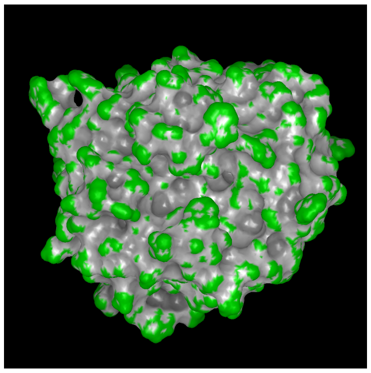
Distance

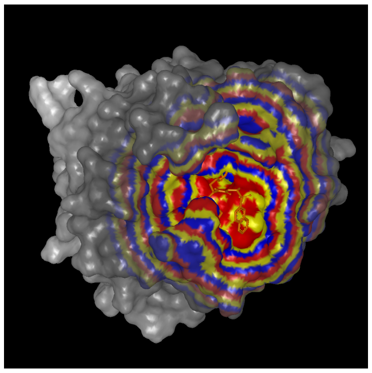
The *distance* scheme colors each vertex by its distance from either the selected set of atoms or the visible set of atoms (if none are selected at the time of coloring). Each band of color represents a distance of one Angstrom from the selected set. The color banding is continuously repeated as the distance increases, but the colors fade to white with each repetition to show the distance effect (see Figure 5.20(e)).

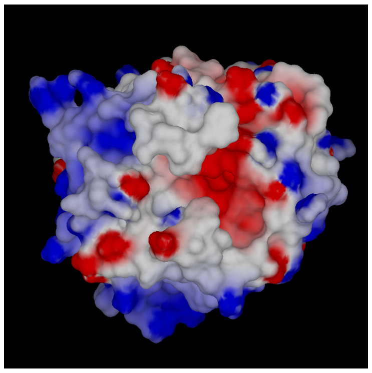

(a) Single color

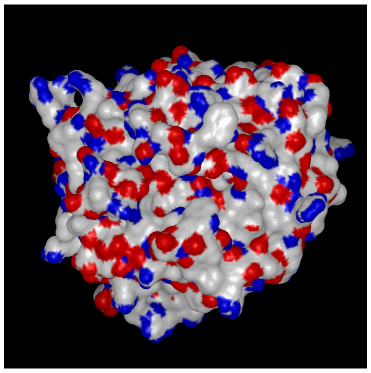

(b) Color by Atom


(c) Color by Concavity


(d) Color by Curvature


(e) Color by Distance


(f) Color by Electrostatics


(g) Color by Hydrogen Bond Potential

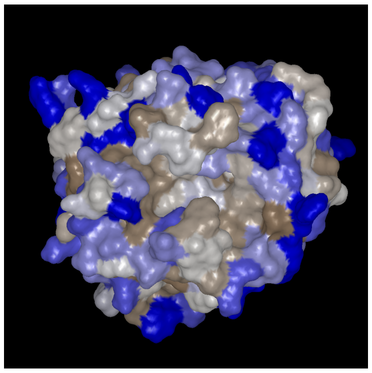

(h) Color by Hydrophobicity

Figure 5.20: Various different surface coloring schemes

Electrostatics

The *electrostatics* scheme colors each vertex on the surface according to the electrostatic potential at that vertex using a red-to-blue gradient from -7.0 to +10.0 (see Figure 5.20(f)). This range can be changed if desired in the application preferences in the *Surfaces* section.

The electrostatic potential at the surface is calculated using OpenEye's Zap toolkit (for more information, please see the Zap toolkit documentation). By default, AFITT will use the molecule's input partial charges in the calculation (if any were specified). However, if no partial charges were specified, AFITT will assign partial charges using either MMFF94 (the default) or AM1-BCC [19, 20]. The choice of charge model can be specified in the application preferences. If a molecule has greater than 50 heavy atoms, MMFF94 will always be used in place of AM1-BCC regardless of the set preferences.

Proteins have an alternative charging option based on residue information. The charge model can be seen in the table below. This model is used by default (if no partial charges were specified), but it can be disabled in the application preferences.

ASP	OD1, OD2	-0.5
GLU	OE1, OE2	-0.5
LYS	NZ	+1.0
ARG	NE	+1.0
Other		0.0

Grid

The *grid* scheme colors each vertex on the surface according to the grid potential at that vertex using a red-to-blue gradient from the minimum grid value to the maximum grid value. The grid potential is determined by mapping the specified grid onto the surface.

Hydrogen Bond Potential

The *hydrogen bond potential* scheme colors each vertex according to the hydrogen bond class of the nearest atom to that vertex in the molecule that the surface was created from (see Figure 5.20(g)). This scheme will not work if the surface was not created from a molecule and also if that molecule is not currently present.

Hydrophobicity

The *hydrophobicity* scheme colors each vertex according to a specified hydrophobic color scale based on residue information associated with a given vertex. This scheme will not work if the surface was not created from a protein and also if that protein is not currently present.

There are four different scales available: Charifson, Eisenberg, Kyte-Dolittle, and White Octanol. The Charifson scale applies defined colors based on residue information. The other three scales each return a specific hydrophobicity value which is then used to return a color based on a brown-to-blue gradient between -2.5 and +2.5 for Eisenberg, -4.5 and 4.5 for Kyte-Dolittle, and -4.0 and 4.0 for White Octanol.

Charifson

All	CA,CB,CG,CD	Yellow	[0, 255, 255]
ASP		Red	[255, 0, 0]
GLU		Red	[255, 0, 0]
ASN		Magenta	[192, 0, 192]
GLN		Magenta	[192, 0, 192]
HIS		Magenta	[192, 0, 192]
LYS		Blue	[0, 0, 255]
ARG		Blue	[0, 0, 255]
ALA		Yellow	[255, 255, 0]
PHE		Yellow	[255, 255, 0]
ILE		Yellow	[255, 255, 0]
GLY		Yellow	[255, 255, 0]
LEU		Yellow	[255, 255, 0]
PRO		Yellow	[255, 255, 0]
VAL		Yellow	[255, 255, 0]
MET		Yellow	[255, 255, 0]
CYS	S	Magenta	[192, 0, 192]
CYS	!S	Yellow	[255, 255, 0]
TRP	N	Magenta	[192, 0, 192]
TRP	!N	Yellow	[255, 255, 0]
SER	O	Magenta	[192, 0, 192]
SER	!O	Yellow	[255, 255, 0]
THR	O	Magenta	[192, 0, 192]
THR	!O	Yellow	[255, 255, 0]
TYR	O	Magenta	[192, 0, 192]
TYR	!O	Yellow	[255, 255, 0]
Other		White	[255, 255, 255]

	Eisenberg	**Kyte-Dolittle**	**White Octanol**
ALA	0.62	1.80	-0.50
ARG	-2.53	-4.50	-1.81
ASN	-0.78	-3.50	-0.85
ASP	-0.90	-3.50	-3.64
CYS	0.29	2.50	0.02
GLN	-0.85	-3.50	-0.77
GLU	-0.74	-3.50	-3.63
GLY	0.48	-0.40	-1.15
HIS	-0.40	-3.20	-2.33
ILE	1.38	4.50	1.12
LEU	1.06	3.80	1.25
LYS	-1.50	-3.90	-2.80
MET	0.64	1.90	0.67
PHE	1.19	2.80	1.71
PRO	0.12	-1.60	-0.14
SER	-0.18	-0.80	-0.46
THR	-0.05	-0.70	-0.25
TRP	0.81	-0.90	2.09
TYR	0.26	-1.30	0.71
VAL	1.08	4.20	0.46
ASX	-0.84	-3.50	-2.25
GLX	-0.80	-3.50	-2.20
Other	0.00	-0.49	-0.52

Surface Potential

The *surface potential* scheme colors each vertex according to the potential value at that vertex using a red-to-blue gradient from the surface potential minimum to the surface potential maximum. This scheme is different than the *electrostatic* scheme in that the coloring is based on the stored potential values as opposed to calculated potential values. Typically, surfaces generated within AFITT will not have any stored potential values. However, surfaces created externally using OpenEye's Spicoli toolkit for example can store values in the potential field and as such can be colored accordingly.

5.11.3 Selection

Vertices can be selected simplying by clicking on the desired location. A line of vertices can be selected by selecting one vertex and then by holding down the *Shift* key when selecting another. This starting selection can be grown (and shrunk) using the Selection pane in the Style Control (see Section 5.11.3). Double-clicking on the surface will select the entire surface.

5.12 Symmetry

AFITT supports symmetry operations. This support includes a menu under the Edit menu (see 5.13) which controls displaying symmetry in the 3D window, and some items in the popup menu of the list window (see 6.4) which allow setting or editing symmetry parameters.

Symmetry information may be read from certain files. When AFITT reads PDB-format files which have valid CRYST1 records, symmetry information is taken from that record for all molecules read from that file. Similarly, when AFITT reads MTZ-format reflection files the symmetry information is read from that file, and of course, AFITT will read symmetry information if it is present in an `.oeb` file.

As described in 5.13, when symmetry display is enabled, all symmetry operators which would result in a molecule within a certain distance (the *symmetry radius*) of the screen center are displayed. This means that as the view is translated, certain symmetry *replicates* will appear and disappear as they move into or out of the symmetry zone. There is, therefore, no need to manually re-calculate the symmmetry after translating the model.

5.13 Symmetry menu

The symmetry menu is located in the *Style* menubar. The items in the symmetry menu are:

- *Edit cell* This item allows editing or setting of the unit cell and space group parameters of the active object, which will generally be either a reflection data set or a molecule. In addition to the interactive symmetry display, information entered here will also be used for the CRYST1 record if the molecule is written as a PDB file.
- *Show unit cell* This toggles the display of the crystallographic unit cell of molecules which have symmetry information specified.
- *Show symmetry* This toggles the display of symmetry-related molecules, hereafter called "replicates". If enabled, *all* visible molecules with symmetry information will display symmetry.
- *Realize symmetry* The symmetry display above is only for display purposes. If the user wishes to actually expand a molecule to produce one with "real coordinates" for the replicates, the symmetry must be "realized". This option will prompt the user to determine if the current molecule(s) should be symmetry-expanded "in place" or if a new copy of the molecule(s) should be created and then symmetry expanded. Only the symmetry operators currently enabled are used in the expansion.
- *Symmetry radius* All symmetry operators which would produce a molecule within the symmetry radius of the screen center are shown. With a large symmetry radius, this may result in more replicates than symmetry operators, as each operator may be used more than once with different translations. As a special case, if the symmetry radius is 0.0, then each operator is used exactly once, with a translation which places the center of mass of the molecule in the default unit cell (i.e. the cell at the origin).

- *Symmetry operators* This submenu lists the symmetry operators for the *focused* molecule. It is possible to toggle individual symmetry operators on and off. This option is very useful for (e.g.) producing the biologically active dimer using crystal symmetry. If there are no operators listed in this submenu, then either a) no object has the focus, b) the focused object is not a molecule, or c) the focused molecule does not have symmetry information. Currently, symmetry information is only read from the CRYST1 record of PDB files.
- *No symmetry color* No special coloring will be used for symmetry replicates.
- *Single symmetry color* A single (distinct) color will be used for all symmetry replicates.
- *Color by operator* Each symmetry operator is assigned a color. All symmetry replicates resulting from the same operator (e.g. X, -Y, Z) have the same color no matter which unit cell they are in
- *Unique color* Each replicate is given a distinct color, so that no two replicates have the same color. Since the number of replicates can change depending on the symmetry radius and/or the current center, this has the somewhat unfortunate side-effect that occasionally translations cause every molecule to be recolored.

Notes: after realizing symmetry, if "Show symmetry" is left on there could be a very large number of molecules on screen as each new molecule is also symmetry expanded.

CHAPTER

SIX

Modeling Basics

AFITT contains modeling specific operations in the various menus (as opposed to generic visualization operations described in Chapter 5) these are described in detail here and in the following chapters.

6.1 Tasks

To facilitate ligand and protein fitting workflows, AFITT divides workflows into *Tasks*. Each *Task* has a basic set of windows related to the current task. For example, the ligand fitting workflow contains the *Ligand Modeling* window, the *List Window*, the *3D Window* and the *Spreadsheet*. These workflows can be selected from the *Task* menu or from the *Layout* pulldown.

It is highly recommended to use the supplied task workflows although they can be modified at will.

6.1.1 Task Menu

The task menu changes the current workflow. The available workflows are

1. Ligand - This workflow is optimized for fitting ligands to crystallographic density.
2. Protein - This workflow is optimized to fitting protein residues to density.
3. Refinement - This workflow helps to view the results of launching external refinement packages, such as *refmac5*.

6.2 File menu

Two modeling specific menus are: "Add from SMILES string" and "Write refinement dictionary".

The “Write refinement dictionary” menu has options to write either a *refmac* or *CNS/X-PLOR* dictionary entry for the currently selected molecule(s). The user will be prompted for the path and filename prefix, which will then be used to produce `.cif` and `.pdf` files or a `.par` and `.top` files, as appropriate. There are three options to the dictionary creation:

- *Include rotors* - By default, constraints for free rotors (torsions) are generated, as they facilitate manual modeling in such crystallography programs such as *Coot*. Note that free rotor torsions are biased toward the current conformation of the molecule. OpenEye recommends leaving this option on, but if desired, setting this parameter off will might be useful in some instances. [Command ModelingDictionaryRotors.]
- *Suppress Hydrogen* - By default, any hydrogen atoms present are *not* included in dictionary parameters, as most commonly the resolution of the data doesn't justify use of hydrogen atoms in the refinement.
- *Aromatic model* - The aromatic model is important because aromatic ring systems are treated as planar and planarity constraints are generated for planar ring systems. The default is the OpenEye (“OE”) aromatic model. Other valid values are “Daylight”, “Tripos”, “MMFF”, and “MDL”. For more information on aromaticity see the OEChem docs[21].

Note that the values set here are remembered as user preferences and will be remembered when restarting AFITT.

6.3 Edit menu

This section describes modeling related editing options located in AFITT's *Edit menu*.

6.3.1 Copying, Cutting and Pasting atoms

- *Copy* - This command copies all currently selected atoms (even atoms from multiple molecules) to an internal clipboard. This clipboard (different from the system clipboard) may then be used for subsequent *Paste* operations.
- *Cut* - This command, similar to Copy, places a copy of the currently modeled atoms on the clipboard. Unlike Copy, however, the atoms are then deleted from the current molecules. If deleting the atoms involves breaking bonds, the implicit hydrogen count is adjusted.
- *Paste* - Add the contents of the clipboard to the each currently modeled molecule. The entire clipboard will be added to each molecule, even if the clipboard consists of parts of multiple molecules. Care should therefore be exercised when calling this function if multiple molecules are being modeled simultaneously.
- *Paste special...* - This menu contains one or more special options for pasting.

The “Paste special...” menu contains:

- *Alternate conformation* This option allows pasting the contents of the cut buffer into a PDB-style alternate conformation (i.e. it sets the OEChem residue’s AlternateLocation property). The user will be prompted for an alternate location identifier (a one-letter code used to distinguish alternate conformers), the occupancy of the new atoms, and to determine if any backbone atoms should be pasted into the molecule. Atoms on the clipboard are pasted into the modeled molecule(s), and the occupancy and alternate location property are set appropriately. At present, the occupancy of atoms already in the model is not changed. Splitting a residue into alternate conformations is currently best accomplished by copying the residue in question and then pasting with this option.

6.3.2 Adding and removing Hydrogens

AFITT includes a “Hydrogens” menu to the edit menu which provides the capability of adding explicit hydrogens to molecules or converting explicit hydrogens to implicit hydrogens. Both of these menu options will prompt for a molecule to operate on.

6.3.3 Changing Residue Names

AFITT allows for the changing of residue names for the *Visible*, *Selected* or *Marked* residues.

This should be used with caution as changing residue names can heavily influence the running of external programs such as *Refmac* and *CNS/Xplor*.

This option is mainly used to allow the changing of ligand residue names when there are two ligands with the same residue name but different structures. If the two ligands have different structures but have the same residue name, refinement dictionaries will most likely not be usable.

To change the residue name for the visible molecule select *Edit→Change Residue Name→Visible*.

At this point you will be prompted with a dialog to enter the new residue name.

Note: This operation is not undoable.

6.4 Popup menus

AFITT includes a “Center on selected” option to the popup menu in the 3D viewer. This option, as the name suggests, centers the view on the selected atoms.

AFITT also includes options to the context-sensitive popup menu in the list widget. Right-clicking on a reflection data set in the list widget shows the “Maps” menu option, while right clicking on a molecule adds the “Set crystal params” option which allows the user to set or modify the unit cell and symmetry information on a molecule.

6.5 Real Space Refinement

AFITT fits ligands and proteins to density in *real-space*. For ligands, this usually means that some portion of a model exists to phase the density in order to reveal the density of the ligand. AFITT supplies some useful facilities to assists in preparing complexes for real space refinement,

6.5.1 Density Versus Blobs

One of the issues that may be encountered is the difference between *density* and *blobs*. A *blob* is not density, rather it is a density boundary. *Blobs* are used to bound the density to use during various procedures. In AFITT 3D surfaces are used to indicate these regions. This allows all surface operations, such as pruning and carving, to be applied to *blobs*. For example, if the *blob* is too large, it can be cropped to remove the undesired regions. For more details on operating on surface, please see Chapter 5.11.

6.5.2 Best Practices

Based on analyzing several hundred protein+ligand combinations, the best way to use AFITT for ligand fitting is described as follows (of course your milage may vary):

- Always use a protein. The protein is used to identify modeled density where the ligand cannot be placed. Without this, AFITT will identify many candidate locations for placing the ligand that will need to be analyzed.
- The reflection data should be refined without a previous ligand conformation. While in many cases the same result will be generated, if the ligand is fit to data refined with a previous ligand conformation, AFITT will be biased towards the pre-existing ligand conformation.
- Sometimes AFITT gets cluttered with too many results and becomes difficult to navigate. Use the *Edit→Hide→All* menu option to clear up the data.

6.5.3 List Window vs Spreadsheet

Many of the same operations can be performed in the spreadsheet and the list window. The spreadsheet has the ability to seperate different object types such as molecules, proteins, maps and blobs into different groups. AFITT uses this cabability to help arrange the results of fitting ligands, for instance and allows for sorting of results. (This is most often used when multiple stereo-variants have been produced and the best fitting variant needs to be identified.)

The list window contains every object loaded into AFITT and is a more general purpose browsing tool. AFITT tries to generate a new list everytime a ligand fitting operation is initiated. All relevant data for the fitting operation will be placed in the newly created list.

CHAPTER

SEVEN

Ligand Modeling

The initial workflow displayed when opening AFITT for the first time is the *Ligand* modeling task. The basic workflow for ligand fitting consists of:

1. Loading the density, ligand and protein,
2. Finding the regions of density (a.k.a. *blobs*) that are indicative of the ligand density
3. Fitting the ligand to these regions. In most cases, these three steps are all that are necessary for ligand fitting.

Advanced usage consists of:

1. Adding constraint boxes for ligand fitting.
2. Automatically identifying the potential binding sites and fitting to these regions.
3. Manually selecting pieces of density to fit against.

The introductory screen for the *Ligand* task is seen in Figure 7.1.

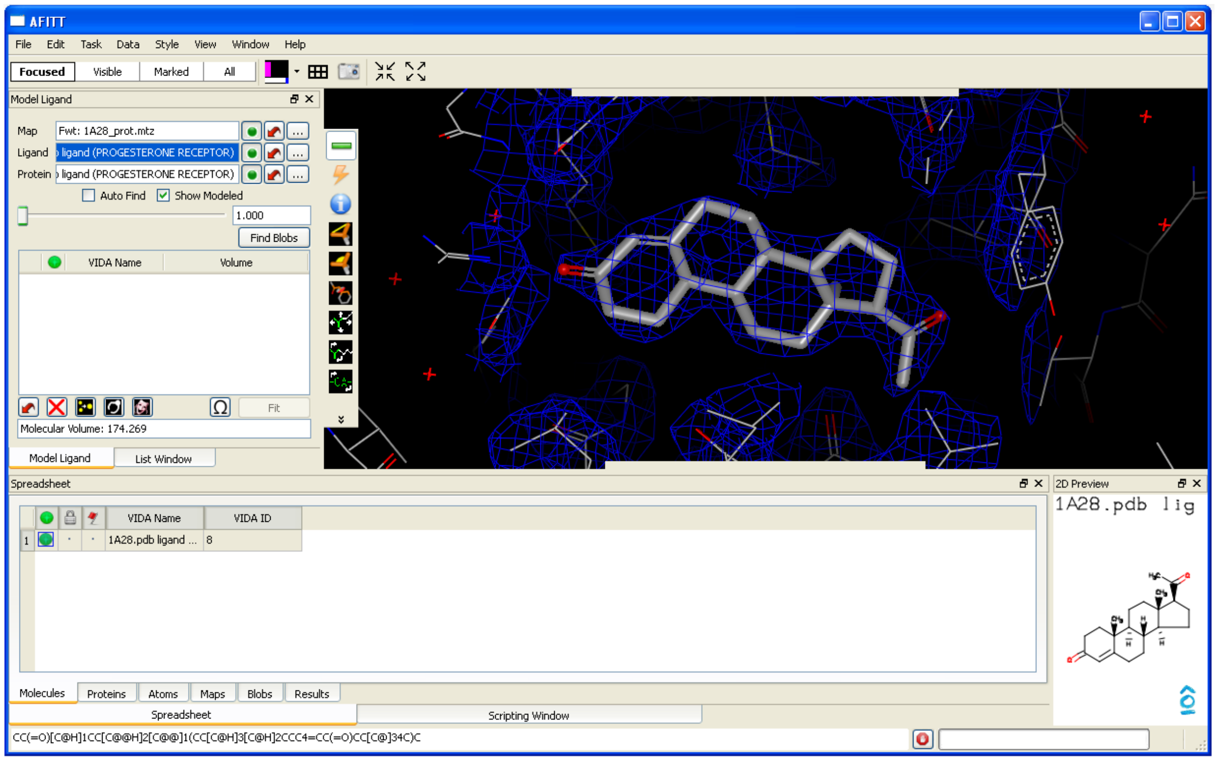
AFITT can automatically locate the appropriate density for the ligand. In most cases, all that is needed is loading the appropriate elements and then clicking on *Find Blobs* and then *Fit*.

The *Model* window organizes the grids and molecules used in ligand fitting and directs the workflow to the final fit.

The main interface elements that implement the ligand fitting task are the *Ligand Modeling Widget*, the *3D Window* and the *Spreadsheet*.

7.1 Ligand Modeling Window

This window, located on the left side of the application as seen in Figure 7.1, shows the state and basic operations available for fitting small molecules to density. To fit a small molecule to density it is nec-



Map: Fwt: 1A28_prot.mtz
Ligand: ligand (PROGESTERONE RECEPTOR)
Protein: ligand (PROGESTERONE RECEPTOR)
☐ Auto Find ☑ Show Modeled
1.000

	●	VIDA Name	Volume

Molecular Volume: 174.269

	●			VIDA Name	VIDA ID
1	●	·	·	1A28.pdb ligand ...	8

```
CC(=O)[C@H]1CC[C@@H]2[C@@]1(CC[C@H]3[C@H]2CCC4=CC(=O)CC[C@]34C)C
```

Figure 7.1: Ligand mode layout

essary to supply ligand density, small molecule. If the density includes an already or partially modeled protein, this greatly assits in locating the appropriate ligand density.

The protein is used to mask the density where the ligand should not be placed. Using the protein is highly recommended as this greatly facilitates the automated blob finding.

Each modeled unit (density, ligand, protein) has a set of controls that can focus the object or toggle the visibility. Clicking on the name of the object will focus it, clicking on the small box to the right of the object will toggle visiblility. Clicking on the arrow will remove the item from the current model and clicking on the button with three dots "..." will browse the current repository for a new object.

Once a model has been loaded, click on the *Find Blobs* button to attempt to find a region of density suitable for the ligand. By default, the *Find Blobs* algorithm uses the current sigma value for the density. Selecting the *Auto* checkbox, forces the searching algorithm to examine all density sigmas in an attempt to find a blob of the appropriate size.

Finally, once a blob has been selected, clicking on the *Fit* button will begin the fitting process. When complete, the fitted small molecules will appear on the screen and in the spreadsheet

7.2 3D window

Most of the time spent in AFITT will be spent in the 3D viewing window which is the largest area of the application. This window shows the currently visible objects, such as density grids, molecules, proteins and blobs. Left-clicking on most items will select them. For instance, left-clicking on a blob will select the blob and make it focused. The selected blob will be used to extract the density region that will be used to fit the small molecule.

7.3 Spreadsheet

The spreadsheet, located at the bottom left as seen in Figure 7.1, shows more details about the objects that are loaded in AFITT. It is seperated into five tabs: *Molecules*, *Proteins*, *Maps*, *Blobs* and *Results*.

The *Molecules* tab holds the small molecules that are loaded into AFITT and the The Proteins tab show the proteins loaded into AFITT. The *Maps* tab contains the currently loaded densities and grids. *Blobs* contain the automatically located blobs and surfaces. Finally, the *Results* tab hold the small molecules that have been fit to the density. For more information about using the spreadsheet, see Chapter 11.

7.4 2D Preview

Located at the bottom right, this shows either a 2D depiction of the small molecule or the sequence of the protein depending on which one is currently focused.

7.5 Locating ligand density

AFITT locates density suitable for ligand placement based on both the type of map being analyzed, and the volume of the ligand to be fit. In the cases where a conformation is given, AFITT computes the volume of the supplied conformation, otherwise the volume of the conformation is estimated through a simple linear approximation based on the number of atoms in the ligand.

Given the estimated volume of the ligand, AFITT starts from the maximum amplitude of the density map and ramps the contour level down until blobs are found within a range of the ligand volume. If the discovered blobs are small relative to the ligand, AFITT includes nearby blobs in an effort to try and match the ligand.

AFITT attempts to include regions within a certain distance from the modeled protein (if available) by default, however, based on the orientation of the protein, the search range may include solvent density or symmetry related blobs that are not near enough to the protein.

Because of this, AFITT includes a distance slider to prune blobs within a certain distance to the protein. The default distance is $4\AA$.

7.6 Filtering Ligand Density Based on Distance to a residue

If the ligand is known to be interacting with certain residues, the selection slider (seen in Chapter 5.11.3) can be used to filter out ligand density that is not close enough to the protein. Simply select the residue by double clicking, then adjust the selection slider or type in a number in the widget. Then, when next clicking on the *Find Blobs* button, any blob that is not within the desired distance will be rejected.

7.7 Working With Poor Density

In some cases, the appropriate ligand density will not be found, usually because the density is too noisy. When this happens, there are several options both automatic and manual:

7.7.1 Split the Protein

Sometimes, the protein has ligands embedded in the structure. Because all components of the protein are used to reject non-ligand dentisy, AFITT can be fooled into rejecting density where the ligand has already been placed. *Note:* Molecules may be split using the *New Molecule From Split* option located in the *File* menu. to separate the ligand, waters and fragments from the protein.

7.7.2 Manual Blob Picking

The blob picking button, located in the *Ligand Model* window, places AFITT in *blob picking mode*. When a piece of density is clicked on in the 3D window, AFITT will attempt to extract the isocontour into a density blob. In many cases, the extracted density can be larger than what is desired, however, the scribing tools located in the *style menu* can be used to crop the surface to the correct size, see Chapter 5.11 for more details.

Note: AFITT always masks visible molecules that are on screen during this procedure. If a blob is picked that has a ligand inside of it, then the blob, in actually, will be masked away. For best results, hide any molecules that are not modeled.

7.7.3 Add a Box Constraint

Constraint regions can be added to limit the areas in which blobs can be detected. A constrain region is a *bounding box* or volume of space where blobs are allowed to be located.

The box itself can also be used as a density blob.

Boxes are defined by molecules or atom selections on a molecule. This is done to simplify box creation. A box can be generated from a ligand or select some atoms on a protein to generate a box.

To add a box, simply select the *Task/Add Box...* menu option from the top menu. This will prompt for a molecule id and then a padding amount. When complete, the bounding region will be displayed in the 3D window. This region will also be added as a blob to the blob window which can immediately be used as a blob for fitting purposes.

The next simplest way is to select some atoms that form the boundary of the desired region. For example, if you have a protein loaded and you can select two atoms that are on the boundary of the pocket. Once the atoms are selected you can choose the *Add Box From Selected...* option from the *Task* menu.

Boxes can also be imported from any molecular file format. For example, an *xyz* formatted file is as follows:

```
2
Box File
C 1 1 1
C 5 10 14
```

The first line indicates the number of atoms. The next line is the name and the remaining lines are the atoms and coordinates. The coordinates are separate by white space (tabs or spaces work just fine) and specified in *Å*ngstroms.

For example, the results of selecting *Add Box...* on the 1ll5 ligand located in the examples directory is displayed in Figure 7.7.3.

There are two things to notice:

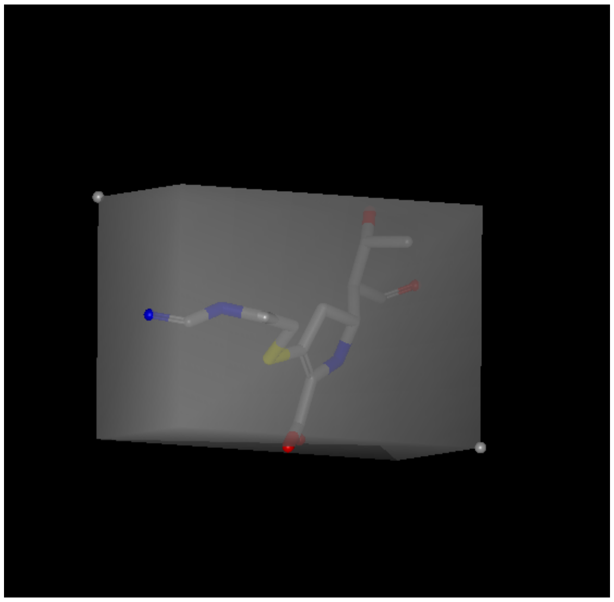
- Associated with the box is a newly created molecule named “Bounding Box”.
- The box has two spheres at either ends (these are actually dummy atoms)

Once a box has been generated, *Mouse Mode Translate* operation can be used to change its size.

7.7.4 Use a Neutral pH Model For The Ligand

In the *Ligand Preferences* page, a neutral pH model can be selected for use with conformer generation.

Otherwise, the *Task/Generate Neutral pH* menu option will apply a neutral pH model to the currently focused or selected ligand.



If the neutral pH model is applied to a protein it most likely will generate undesired results for incomplete chains or residues.

Note: Changing the pH model for a ligand or protein is not undoable. The molecule will have to be reloaded.

7.7.5 SCiNI - Find Active Sites

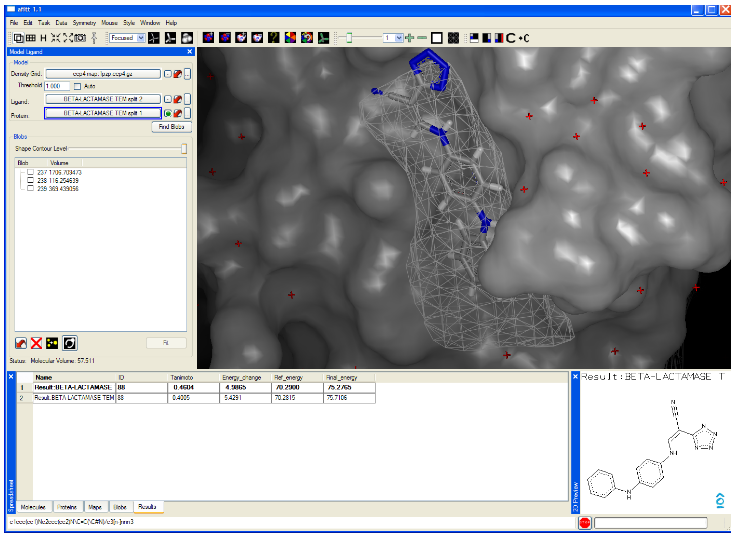
In this case, you can use the *Generate Negative Image* option of the *Task* menu to attempt to find the active sites of the protein. This generates a Shape Contour Negative Image by running a collection of small molecule fragments over the protein. While this can take some time, the resulting potential field can be used as "density" where active site bounding regions can be manually selected as blobs as in section [**?**].

After creating a negative image, the contour sliders can be used to increase or decrease the size of the located active sites. Once a desirable volume has been set, the manual blob picking process can be used to select *blobs* that will be used as bounding regions7.2.

This process is time consuming for large proteins, so be prepared to go out to lunch.

As the figure demonstrates, SCiNI can get remarkably close to the protein's active site, in this case the ligand is the crystallgraphic ligand and the detected region is the active site.

7.8 Writing Refinement Dictionaries



Density Grid: ccp4 map:1pzp.ccp4.gz
Threshold 1.000 Auto
Ligand: BETA-LACTAMASE TEM split 2
Protein: BETA-LACTAMASE TEM split 1

Blob	Volume
237	1706.709473
238	116.254639
239	369.439056

Status: Molecular Volume: 57.511

	Name	ID	Tanimoto	Energy_change	Ref_energy	Final_energy
1	**Result:BETA-LACTAMASE T**	**88**	**0.4604**	**4.9865**	**70.2900**	**75.2765**
2	Result:BETA-LACTAMASE TEM	88	0.4005	5.4291	70.2815	75.7106

```
c1ccc(cc1)Nc2ccc(cc2)N\C=C(\C#N)/c3[n-]nnn3
```

Figure 7.2: Shape Contour Negative Image

AFITT has the ability to writeout refinement dictionaries suitable for REFMAC and CNS/X-PLOR. To write out a refinement dicionary, simply choose "Write Refinement Dictionary" from the file menu, then choose the type of refinement program being used and finally choose the scope of the molecules being written from their respective submenus. At this point a file browser will appear for the location where the dictionary will be written.

For example, to write a *REFMAC* refinement dictionary for all visible molecules, choose:

File→Write refinement dictionary→REFMAC→Visible...

This is the most common procedure for writing out refinement dictionaries.

Dictionary generation uses the *MMFF94* forcefield to form geometric constraints for ligands and proteins under refinement. When writing out REFMAC dictionaries, the following residues will not be written since *REFMAC5* already has geometric constraints (this also means that any ligand that needs to be written out should not have one of the following residue names):

Table 7.1: Residues Names Not Written to Dictionary

ALA	ARG	ASN	ASP	CSH	CYS	GLN	GLU	GLY	HIS
ILE	LEU	LYS	MET	MSE	ORN	PHE	PRO	SER	THR
TRP	TYR	VAL	ACE	FOR	ABA	BOC	BMT	SAR	MLE
MVA	IVA	DFO	NME	AHT	PTR	PCA	HYP	INI	NLE
TYS	CGU	STA	ILG	OCS	KCX	SAH	SAM	SEP	LLP
5HP	CSO	ETA	TFA	ANI	MPR	DAM	ACB	ADD	CXM
DIP	BAL	A	Ad	C	Cd	G	Gd	T	Td
A	Ar	C	Cr	G	Gr	U	Ur	YG	PSU
I	Ir	MAN	MAN-b-D	NAG	NAG-b-D	SIA	FUC	FUC-a-L	GAL
GAL-b-D	GLC	GLC-b-D	XYL	DRB	RIB	FRC	FRU	XYS	XLS
ARB	RIP	ABE	RAM	MAL	LAT	SUC	GCU	GCU-b-D	CEG
CEG-b-D	HOH	5GP	AMP	IMP	UFP	UMP	SGN	1AR	1GL
1GN	1MG	1PA	2AS	2GL	2MA	3DR	3FM	3GA	4SU
5CM	5IT	5IU	5NC	8BR	A23	AAR	ABH	AEI	AET
AF1	AGL	AGM	AH0	AHB	ALM	ALN	ALO	AMG	AMU
APH	APM	ARA	ARM	ASG	ASK	ASL	ASM	ASO	ASQ
ASX	B7G	BCS	BDG	BGL	BGP	BHD	BLG	BMA	BNG
BOG	BRU	BTA	BTC	BTR	BUC	C3X	C4X	C5C	C5P
C5X	C6C	CAN	CAR	CAS	CAY	CBI	CCS	CEA	CH
CHP	CLB	CLD	CME	CPC	CPR	CSB	CSD	CSE	CSP
CSS	CSW	CSX	CTH	CUC	CYG	CYM	D1P	D3	D6G
DAB	DAF	DAG	DAR	DAS	DBY	DCG	DCM	DCY	DDA
DDB	DDL	DFX	DGL	DGN	DGP	DHI	DHN	DHU	DIL
DIV	DLF	DLY	DMT	DNP	DO2	DOH	DOM	DPP	DRI
DSE	DSN	DSP	DSR	DTY	E	EDC	EEB	EFC	EHP
EJT	EMP	EPU	EYS	FAG	FCA	FCB	FCY	FGL	FLA

Table 7.2: Residues Names Not Written to Dictionary (continued)

FME	FMP	FOE	FTR	FTY	G16	G1P	G2F	G4D	G4S
G6P	G7M	GAA	GAM	GAP	GC4	GCM	GCN	GFP	GHP
GL3	GLA	GLB	GLD	GLF	GLM	GLP	GLS	GLX	GLZ
GMA	GMH	GN7	GPM	GS	GSA	GSC	GSR	GSS	GT9
GUP	H2P	H5M	HAR	HGA	HIC	HIP	HMA	HMF	HPH
HSO	HTR	HV5	IAS	IDG	IDS	IDU	IGL	IGU	IIL
IMC	IML	IPT	ITR	IYG	IYR	KAI	KAN	LLY	LNO
LOL	LTA	LTR	LYM	LYZ	M3L	M6P	MA1	MA2	MA3
MA4	MA6	MAT	MBG	MCY	MDA	MDM	MDP	MEN	MF3
MFB	MFU	MGA	MGL	MGN	MGY	MHL	MHO	MHS	MIA
MIS	MLY	MLZ	MNV	MRP	MSA	MTY	MUR	MXY	N5M
NAA	NAM	NBG	NED	NEP	NFA	NGA	NGL	NGS	NMC
NP3	NPH	NVA	OCY	OIC	OIP	OMP	OMT	ONL	OPR
OTG	PA1	PAO	PAQ	PBB	PDU	PEC	PGP	PGY	PHA
PHD	PHI	PHL	PHM	PNA	PNG	POM	PPN	PR3	PRR
PRS	PTH	PYA	PYX	QSI	QUO	R	RHA	RIA	RIN
RON	RPD	RPL	SBD	SBL	SCH	SCR	SCS	SEB	SEC
SEG	SET	SFG	SGA	SGC	SHP	SLZ	SMC	SME	SNC
SOC	STY	SVA	T6A	T6P	TAF	TBG	TBM	TDG	THC
THO	TIH	TMB	TMD	TMP	TMR	TNB	TOA	TOC	TPO
TRF	TRG	TRN	TRO	TS	TYI	TYN	TYQ	TYV	TYY
UAP	VG1	X	X2F	YOF	YYG				

CHAPTER

EIGHT

Protein Modeling

The *Protein Task* gives complete control over atom positions in density. The initial view of the protein mode is seen in 8.1.

While the initial view of *Protein Mode* is intimidating, there are only a small handful of operations that are used regularly. These operations also have hot-keys and key stroke combinations to facilitate usage for advanced users.

Note: For previous users of AFITT, the *Modeling Mode* has been removed. Modeling operations can be performed at any time and can be undone by using the *Edit→Undo* menu or *Ctrl-Z* keystroke.

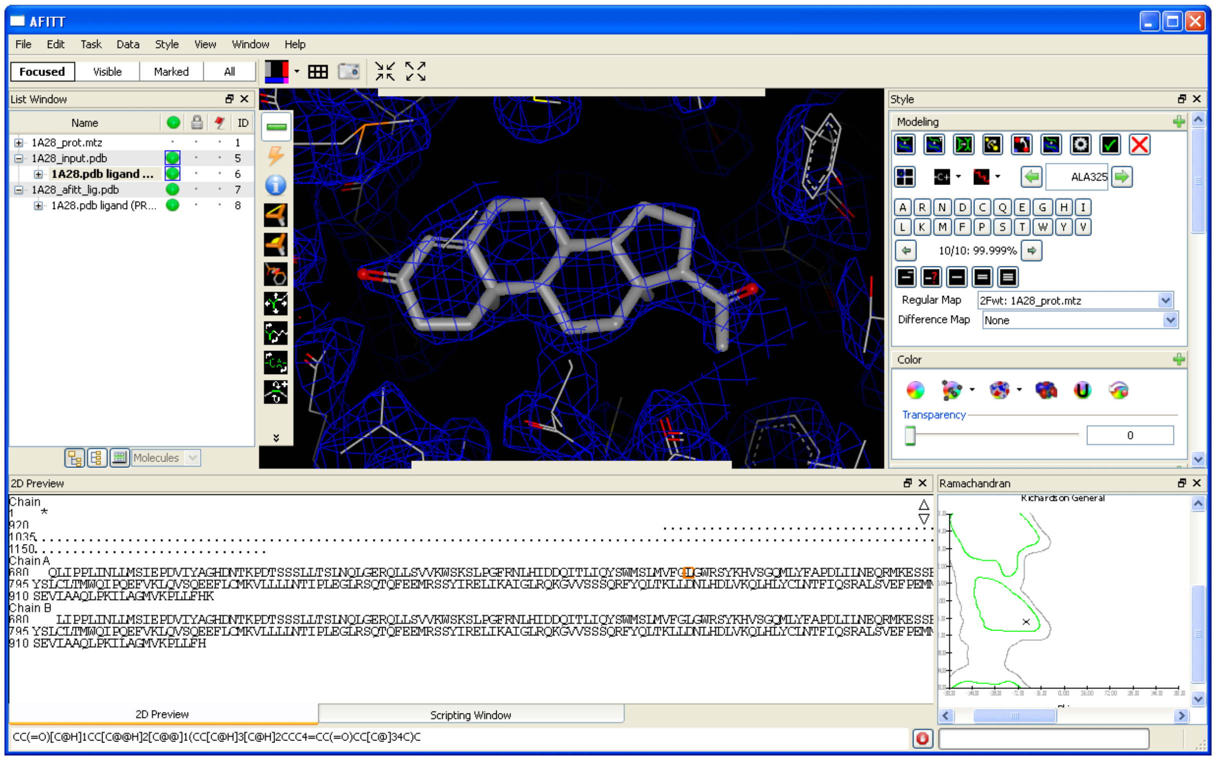
As seen in 8.1, the *Protein Mode* task is divided into several user-interface components.

- *List Window* - controls states of molecules.
- *3D Window* - visualizes results of operations.
- *Style Window* - performs modeling and visualization operations.
- *2D Preview* - Allows easy access to residues.
- *Ramachandran Plot* - Shows overall validation of protein.

These components are described in detail below.

8.1 List Window

On the left side, the *List Window* shows what objects are currently loaded in AFITT and their visibility and marked states.



Name	ID
1A28_prot.mtz	1
1A28_input.pdb	5
1A28.pdb ligand ...	6
1A28_afitt_lig.pdb	7
1A28.pdb ligand (PR...	8

ALA325
10/10: 99.999%
Regular Map 2Fwt: 1A28_prot.mtz
Difference Map None
Transparency 0

```
Chain
1       *
920
1035.....................................................................................
1150.............................
Chain A
680    QLIPPLINLIMSIEPDVIYAGHDNTKPDTSSSLLTSINQLGERQLLSVVKWSKSLPGFRNLHIDDQITLIQYSWMSLMVFGLGWRSYKHVSGQMLYFAPDLIINEQRMKESSF
795 YSLCLTMWQIPQEFVKLQVSQEEFLCMKVLLLLNTIPLEGLRSQTQFEEMRSSYIRELIKAIGLRQKGVVSSSQRFYQLTKLLDNLHDLVKQLHLYCLNTFIQSRALSVEFPEMM
910 SEVIAAQLPKILAGMVKPLLFHK
Chain B
680     LIPPLINLIMSIEPDVIYAGHDNTKPDTSSSLLTSINQLGERQLLSVVKWSKSLPGFRNLHIDDQITLIQYSWMSLMVFGLGWRSYKHVSGQMLYFAPDLIINEQRMKESSF
795 YSLCLTMWQIPQEFVKLQVSQEEFLCMKVLLLLNTIPLEGLRSQTQFEEMRSSYIRELIKAIGLRQKGVVSSSQRFYQLTKLLDNLHDLVKQLHLYCLNTFIQSRALSVEFPEMM
910 SEVIAAQLPKILAGMVKPLLFH
```

```
CC(=O)[C@H]1CC[C@@H]2[C@@]1(CC[C@H]3[C@H]2CCC4=CC(=O)CC[C@]34C)C
```

Figure 8.1: Protein Workflow Layout

8.2 3D Window

As in the *Ligand Task*, the 3D window takes up a majority of the view and controls most of the action. Changes to protein structure and protein fit to density are indicated here and atoms, bonds or residues may be selected by left-clicking or left-double clicking.

8.3 Style Window

The *Style window* provides the primary user interface for performing modeling operations as well as changing visualization aspects such as slabbing control, contour level and molecular style visualization. The Style Control has several different widgets, each of which controls a different aspect of visualization operations. Most of these controls are described in Chapter 5.

In addition to using the buttons in the the modeling window, there are many keyboard shortcuts or hotkeys installed which make the most common modeling operations one-keypress operations. The hotkeys are active when the cursor is in either the 3D window or the modeling window. When applicable, hotkeys are given in the tooltip for the individual command button, and in any menu item which references the given command. The hotkey bindings may be changed by the user if desired. See `WindowRegisterHotkey` in the scripting manual for details on customizing the hotkey bindings.

The modeling widget is where the remainder of the modeling operations are placed. Modeling operations include operations on bonds, such as breaking and creating bonds, residue operations such as creating or replacing residues and operations on rotamers such as locating the highest probabillity rotamers from the various rotamer libraries.

Even though AFITT includes a highly comprehensive *Undo/Redo* facility, sometimes it is useful to backup proteins and ligands during large scale changes, such as automatically cleaning up a protein. AFITT includes a *Mark State* and *Revert to last mark* facility to quickly make a checkpoint of the current session and restore it quickly (seen in Figure 8.2).

The breakdown of the modeling widget is as follows:

8.4 Protein Modeling Operations

These commands attempt to automatically fit the modeled atoms to density using OpenEye's various shape and forcefield tools.

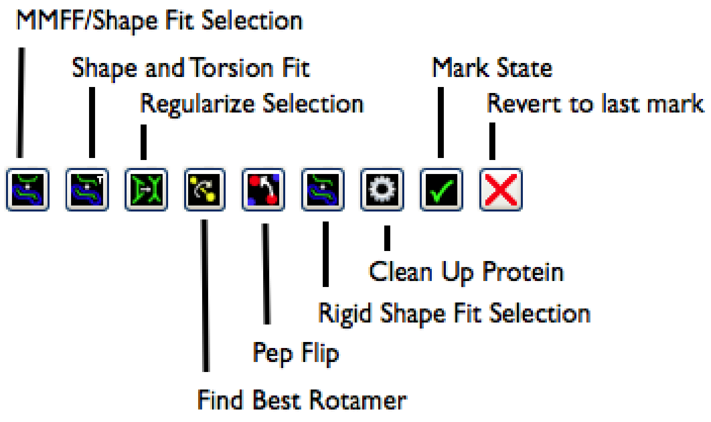


Figure 8.2: Protein Controls

- MMFF Shape Fit [[*Shortcut SHIFT+G*] This takes the current conformation and uses the MMFF force field plus a shape force to fit the selected atoms to the visible density.
- Torsion Shape Fit [*Shortcut SHIFT+D*]

  Uses torsion driving plus the Shape force to drive the residue into density. When this works, it works well though in some cases it generates non-physical structures. A combination of torsion driving and then MMFF Shape Fit seems to work well.
- Regularize [*Shortcut g*]

  This corrects poor chemistry and bond angles between the selected atoms. It is a pure geometric energy minimization and any available density is ignored.

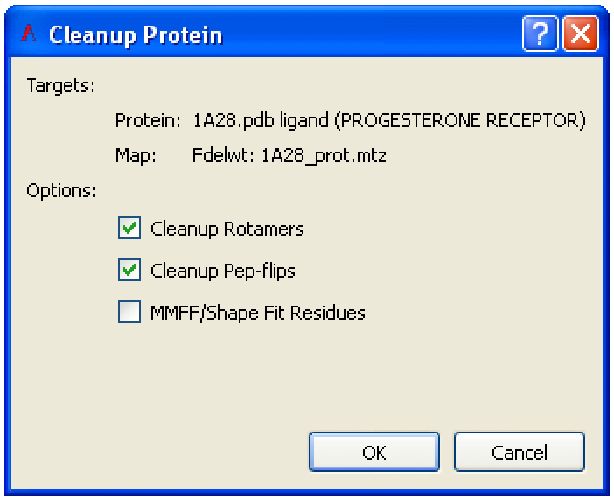
- Best Rotamer Search [*Shortcut o*]

  This searches the Dunbrack or Richardson rotamer libraries to find the highest probability rotamer for the currently modeled residue. If density is currently in scope, this chooses the library rotamer that best matches the density. If no density is being modeled, this chooses the highest probability rotamer.

- Pepflip [*Shortcut v*] Flips the peptide backbone.

- Shape Fit [*Shortcut d*] This performs a rigid body shape fit. Most of the time this produces unacceptable results since the rotamers are not allows to change to match the surrounding density. In some cases, this causes the residue to find a slightly distant portion of density that the current geometry prefers which, in turns, causes ridiculously long bond angles.

- Cleanup Protein Cleans up the entire protein on a residue by residue basis. The following dialog appears when clicked:


Cleanup Protein

Targets:

Protein: 1A28.pdb ligand (PROGESTERONE RECEPTOR)

Map: Fdelwt: 1A28_prot.mtz

Options:

[✓] Cleanup Rotamers

[✓] Cleanup Pep-flips

[ ] MMFF/Shape Fit Residues

Figure 8.3: Clean protein dialog

The current target proteins and map is shown, if these are not the desired targets, simply cancel the dialog and make the desired protein and map (and only the desired protein and map) visible.

This dialog can be used to adjust which aspects of the protein are being cleaned. The available options are

  – Cleanup Rotamers
  – Cleanup pep-flips
  – MMFF/Shape Fit Residues

After the the cleaning process is started, it can be canceled at any time by clicking on the [*Stop button* next to the progress bar.

Similar to the *Best Rotamer Search*, this uses the density in scope.

If rotamers are changed, or peptides are flipped, the changed residues are colored *gold* to indicate that they have been changed and to make them easily visible in both the *3D Window* or the *2D Preview*.

- Fit To Screen Center AFITT's default is to use the density at the center of the screen as the target for optimization and fitting. Normally this works well, however, there are some times where it is desirable to not use the screen center and use the current position of the selected atoms as the density target. This toggle is seen in the panel shown in Figure 8.4.

8.5 Appending, Prepending or Mutating Residues or Rotamers

Residues can be mutated, appended or prepended to the current model in a variety of different conformations. As seen in Figure 8.4, the arrows select the previous and next residues while the buttons simply perform the selcted operation based on the residue type and geometry selected in the pull down menus.

Figure 8.4 shows the break down of the residue operations.

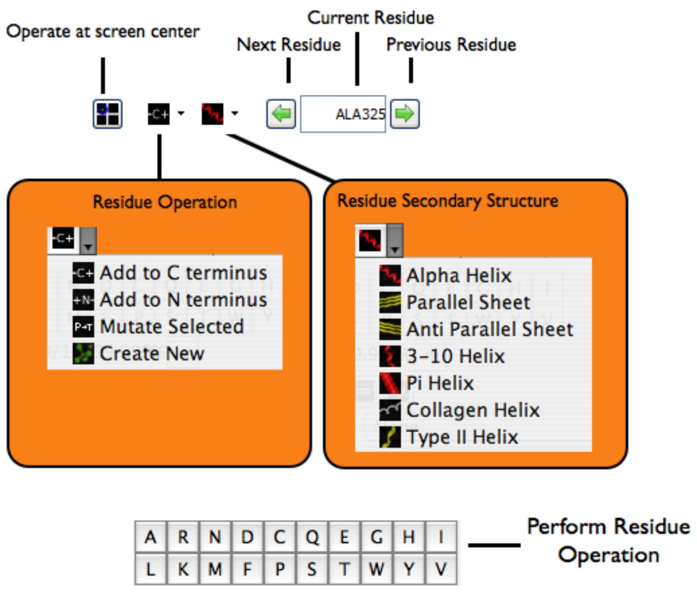


Figure 8.4: Residue Controls

Once the operation (mutate, append, etc.) has been chosen, clicking on the appropriate amino acid code will perform the desired operation.

The rotamer button selects the highest probability rotamers for the selected residue from the current rotamer library. The available rotamer libraries are either the Dunbrack or Richardson rotamer library[14, 16].

When the rotamer search button is pressed, if no density is presently modeled, the highest probability rotamer is selected. If density is present, the rotamer that best fits the surrounding density is selected. The arrows step through the available choices in the rotamer library.

8.6 Bond Specific Operations

Bond operations either operate on the selected bond (when bond order is being changed) or the selected atoms (when bonds are being created).

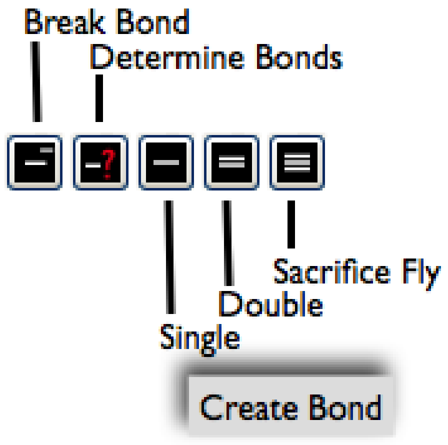


Figure 8.5: Bond Controls

1. Break Bonds - break the modeled bonds.
2. Determine Bonds - determine the bond types for the selected residue (note that if only part of the residue is selected, this determines the types for the whole residue)
3. Single bond - change the selected bond to a single bond or, if no bond exists, generate a single bond between the selected atoms.
4. Double bond - change the selected bond to a double bond or, if no bond exists, generate a double bond between the selected atoms.
5. Triple bond - change the selected bond to a triple bond or, if no bond exists, generate a triple bond between the selected atoms.

For each operation, the number of hydrogens (implicit or otherwise) is correctly determined.

8.6.1 Adding Covalent Bonds

Covalent bonds can be added by selecting an atom from the ligand and an atom from the protein. If the atoms are in different molecules (i.e. did not come from the same PDB file) then a new molecule will be generated by combining the ligand and the protein with the appropriate bond. The old protein and ligand will be preserved. Bound ligands are added to a new chain id that does not exist in the protein. If this cannot be done, then the ligand is assigned a new residue number.

When adding a covalent bond, the ligand atom names are regenerated so that the atom names do not clash with the protein and the atom names themselves are in the proper PDB style. This has a side effect that the atom names (specifically the atom ordering) for the ligand might change. For example, atom "O16" might become atom "O5".

8.7 Mouse Mode Toolbars

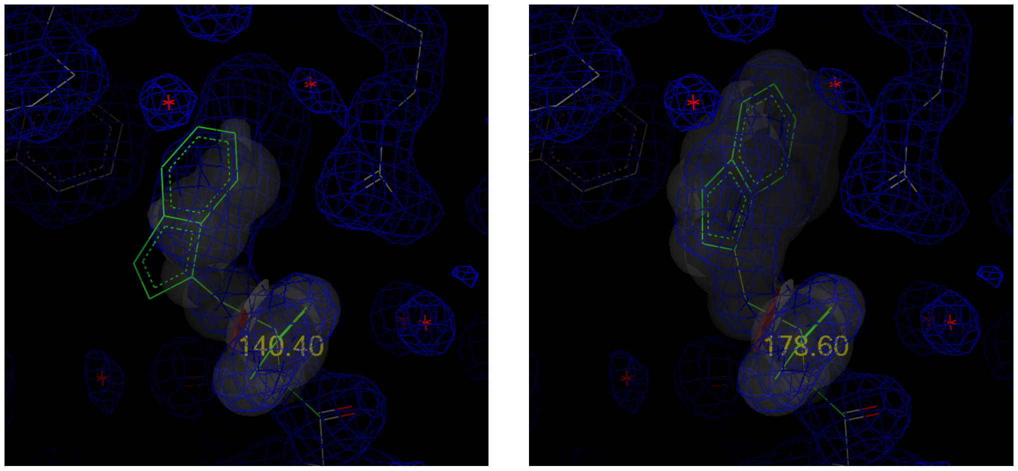
The mouse mode toolbars change the behavior of the mouse in the 3D window. The default behavior of the mouse controls selection and rotation of the main view. Other mouse modes control altering residue and bond torsions, translating the modeled atoms in 3D and modifying and placing the c-alpha backbones.

Hold the mouse over each button to get a tooltip describing the various modes:

1. World Mouse [*Shortcut w*] - Sets the mouse to the default mode.
2. Translation [*Shortcut t*] - translate the selected atoms in space.
3. Rotate Selected (Center of Mass) [*Shortcut Shift+R*]- rotate the selected atoms around their center of mass.
4. Rotate Selected (C-Alpha) [*Shortcut r*] - rotate the selected atoms around the residues C-alpha carbon.
5. Torsion [*Shortcut CTRL+B*] - Rotate around a torsion keeping one side of the torsion fixed.
6. Reverse Torsion [*Shortcut CTRL+SHIFT+B*] - Rotate around a torsion keeping the other side of the torsion fixed.
7. Phi [*Shortcut f*] - Rotate the Phi torsion of a residue.
8. Psi [*Shortcut s*] - Rotate the Psi torsion of a residue.
9. Carbonyl - Rotate the residues carbonyl group.
10. Chi1 [*Shortcut 1*] - Rotate the residues Chi1 torsion.
11. Chi2 [*Shortcut 2*] - Rotate the residues Chi2 torsion.
12. Chi3 [*Shortcut 3*] - Rotate the residues Chi3 torsion.

13. Chi4 [*Shortcut 4*] - Rotate the residues Chi4 torsion.
14. Chi5 [*Shortcut 5*] - Rotate the residues Chi5 torsion.
15. Backbone - Rotate the residue's backbone. This mode rotates the residue's c-alpha carbon around the previous residue's c-alpha carbon and is useful for placing the backbone c-alpha into appropriate density.

Most mouse modes have automatic monitors that show the torsion or rotations angle are being set. Also, if a density grid is available (i.e. visible) the rotated atoms show a spherical halo whose size corresponds to its placement in density. Smaller halos indicate that the atom is not surrounded by much density while larger halo's indicate that an atom is surrounded by more density (as seen in figure 8.6.



(a) Poor density fit (b) Good fensity fit

Figure 8.6: Examples of visualization of poor and good density fits. Noticed that in the poor fitting example, the fitting halo in some of the ring atoms have disappeared

8.8 2D Sequence View

The 2D sequence view shows the protein sequence of the currently visible molecule. (If a ligand is *Focused* then it shows the 2D structure of the ligand).

Residues can be selected by clicking on the appropriate sequence.

8.9 Ramachandran plot

AFITT includes an interactive Ramachandran plot, available from Window→Ramachandran plot in the top menu. The *Focused* molecule is depicted in the Ramachandran plot. There are multiple Ramachandran plot modes available—right click in the plot window to pop up a menu which allows changing the mode.

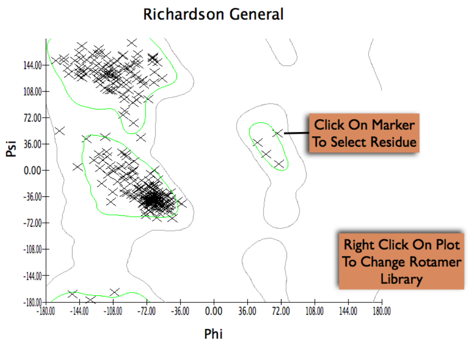


Figure 8.7: Ramachandran Plot

When a specific residue is selected, only that residue and it's immediate neighbors are shown in the *Ramachandran* plot, otherwise, all residues are shown. Clicking on a residue from the plot will select the residue for operations. Additionally, hovering over a residue will show the residue information.

The Richardson Ramachandran plots are, to some degree, specific for given amino acids (e.g. glycine, proline, pre-proline). For this reason, glycine, proline, and residues immediately N-terminal to proline don't show up in the "Richardson general" Ramachandran plot, only in their respective plots.

8.10 Modeling toolbar

The modeling toolbar, shown in Figure 8.8, is a simplified version of the protein modeling widget.

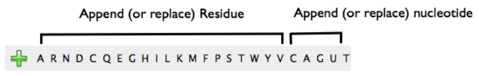

Append (or replace) Residue
Append (or replace) nucleotide
A R N D C Q E G H I L K M F P S T W Y V C A G U T

Figure 8.8: Ramachandran Plot

If a residue is selected, clicking on an amino acid code in the toolbar will mutate the residue. If no residue is selected, the amino acid will be appended to the end of the protein.

The toolbar uses the secondary structure used in the protein modeling controls (see Figure 3.9).

CHAPTER

NINE

Protein Refinement

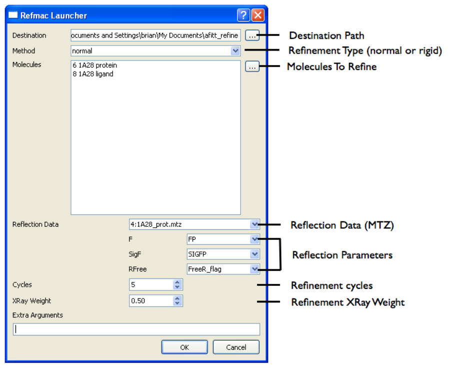
Refmac refinements can be launched by using the *Task→Launch Refinement...* menu option. By default the visible molecules and reflection data is used to launch the refinement. The refinement dialog is seen in Figure 9.1.

9.1 Setting Up Refmac5

For external refinements to work, *refmac5* must be installed and available from the command line.

The *Destination* field contains the directory and initial name for the refinement job. If the path is ...\My Documents\afitt_refine then launching the refinement will produce several output files:

- ...\My Documents\afitt_refine.sh - the shell script that can be used to relaunch the refinement job (on unix)
- ...\My Documents\afitt_refine.bat - the batch script that can be used to relaunch the refinement job (on windows)
- ...\My Documents\afitt_refine.param - the *refmac5* parameter file used for the refinement.



Destination: ocuments and Settings\brian\My Documents\afitt_refine
Method: normal
Molecules:
6 1A28 protein
8 1A28 ligand
Reflection Data: 4:1A28_prot.mtz
F: FP
SigF: SIGFP
RFree: FreeR_flag
Cycles: 5
XRay Weight: 0.50
Extra Arguments:

Figure 9.1: Refinement page of the Modeling Window

- ...\My Documents\afitt_refine.pdb - the input pdb file used for the refinement (automatically generated by AFITT)
- ...\My Documents\afitt_refine.cif - the input refinement dictionary used for the refinement. (automatically generated by AFITT)

Method can be either *Normal* or *Rigid* which speficy a flexible or rigid refinement, respectively.

When the *Refmac Launcher* first appears, the *Molecules* list is populated with the currently visible molecules. If these are not desired, the appropriate molecules can be selected from by clicking on the "..." button to the right of the molecules list.

The "Reflection Data" field allows specification of which reflection object (currently this should be a mtz file) should be passed to the refinement. Note that AFITT does *not* write reflection data, but merely passes the filename of the reflection data which to the script. It is the user's responsibility, therefore, to make sure that the file still exists where it did when AFITT read it.

There are also fields for specifying the number of cycles, the xray weight, and the column names for the given fields. When a data file is chosen, AFITT will make its best guess as to the appropriate column names, but these may need to be changed this before launching the refinement.

Finally there is a space for extra arguments. These extra arguments are passed onto *refmac5* as command line arguments.

Finally, pressing the Launch button will call the given script with the specified parameters. Although not necessary, the progress of the script can be monitored from the *Task-¿Refinement* workflow. Once refinement is finished, AFITT will notify the user that the job is complete and ask if the output file(s) from the job should be read.

9.2 Altering Refmac5 Parameters

AFITT uses the python scripting language to prepare and launch *refmac5*. There are two python variables that hold the default *refmac5* scripts:

- REFMAC_RIGID - holds the parameters for the rigid refinement.
- REFMAC_RIGID_PARAMS - holds the parameters for the rigid refinement.

These parameters can be modified by either changing them in the scripting window (not recommended), loading a python script with the new definitions or placing the new definitions in your "startup.py" file (see Chapter 14.2).

Supplied along with AFITT, is a supplementary file named *refmac_params.py* that contains the original values of these variables that can be used as templates for changing the parameters.

The following is the normal mode of running *refmac5*. When working with python, there are two things to remember; 1) the triple quotes are how python designates a block of text, everything between two sets of triple quotes is captured in the variable, include line breaks. 2) the funny looking:

```
%(F)s
```

indicates that a named variable is passed from AFITT to the *refmac5* parameters. These variables are:

- F - The FC column from the original reflection file.
- SigF - The Sigma F column from the original reflection file.
- RFree - the RFree column from the original reflection file
- weight - The XRay weight to use.
- cycles - The number of refinement cycles to used.

```
REFMAC_PARAMS = """make check NONE
make -
    hydrogen NO -
    hout NO -
    peptide NO -
    cispeptide YES -
    ssbridge YES -
    symmetry YES -
    sugar YES -
    connectivity NO -
    link NO
refi -
    type REST -
    resi MLKF -
    meth CGMAT -
    bref ISOT
ncyc %(cycles)s
scal -
    type SIMP -
    LSSC -
    ANISO
solvent YES
weight -
    MATRIX %(weight)s
monitor MEDIum
labin  %(F)s %(SigF)s %(RFree)s
labout  FC=FC FWT=FWT PHIC=PHIC PHWT=PHWT DELFWT=DELFWT PHDELWT=PHDELWT FOM=FOM
RSIZE 80
END
"""
```

The following are the rigid refinement parameters:

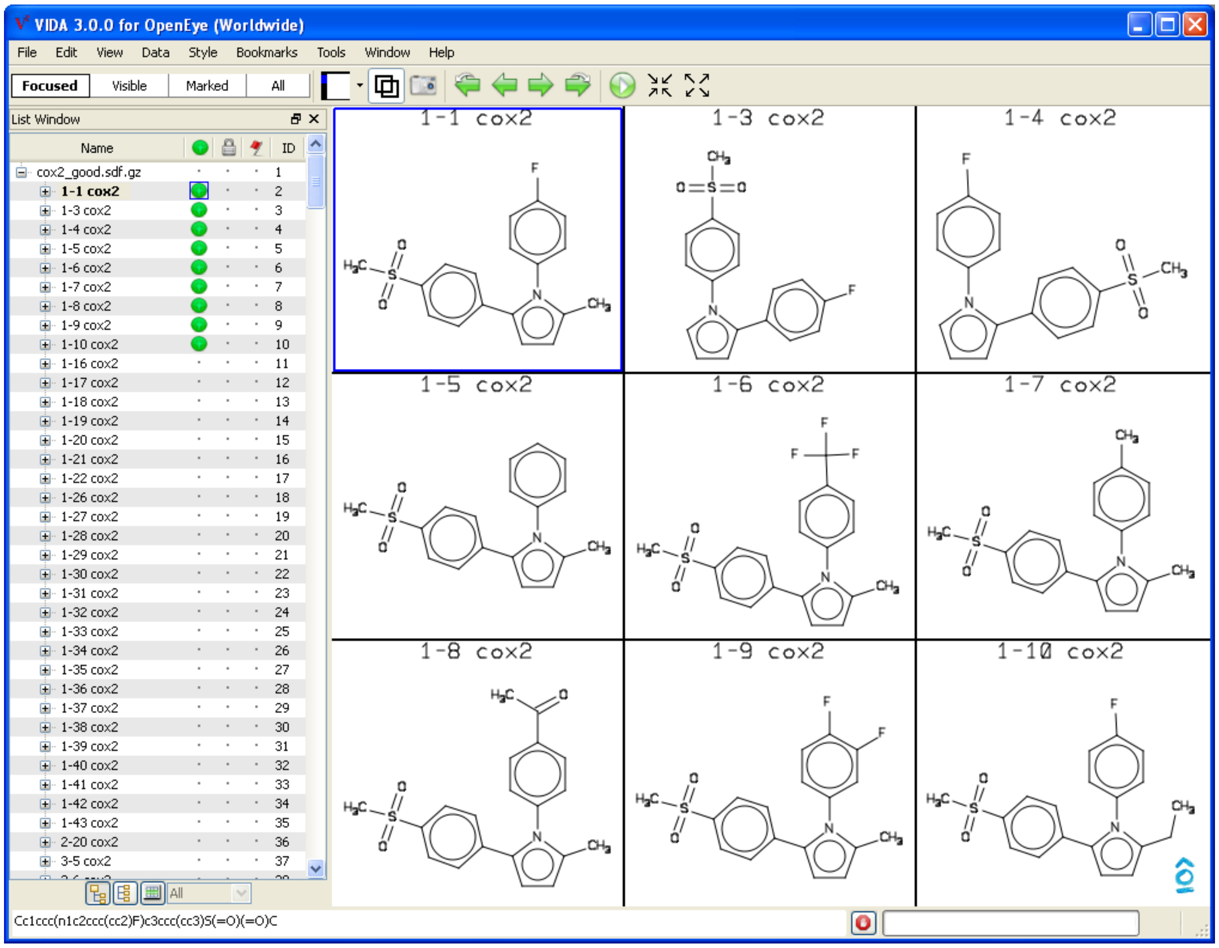
```
REFMAC_RIGID_PARAMS="""make check NONE
make -
    hydrogen NO -
    hout NO -
    peptide NO -
    cispeptide YES -
    ssbridge YES -
    symmetry YES -
    sugar YES -
    connectivity NO -
    link NO
refi -
    type RIGID -
    resi MLKF -
    meth CGMAT
rigid ncycle %(cycles)s
scal -
    type SIMP -
    reso 2.000 100.0 -
    LSSC -
    ANISO
solvent YES
weight -
    MATRIX %(weight)s
monitor  MEDIUM -
    torsion 10.0 -
    distance 10.0 -
    angle 10.0 -
    plane 10.0 -
    chiral 10.0 -
    bfactor 10.0 -
    bsphere 10.0 -
    rbond 10.0 -
    ncsr 10.0
labin  %(F)s %(SigF)s %(RFree)s
labout  FC=FC FWT=FWT PHIC=PHIC PHWT=PHWT DELFWT=DELFWT PHDELWT=PHDELWT FOM=FOM
RSIZE 80
END
"""
```

CHAPTER

TEN

2D Display

2D molecular depictions are provided using OpenEye's Ogham toolkit. A large number of options are available in the applications preferences to customize how the depictions are drawn. Depictions are drawn as black on a white background by default as seen in Figure 10.1, but other color schemes are available and can seen in Figure 10.2.

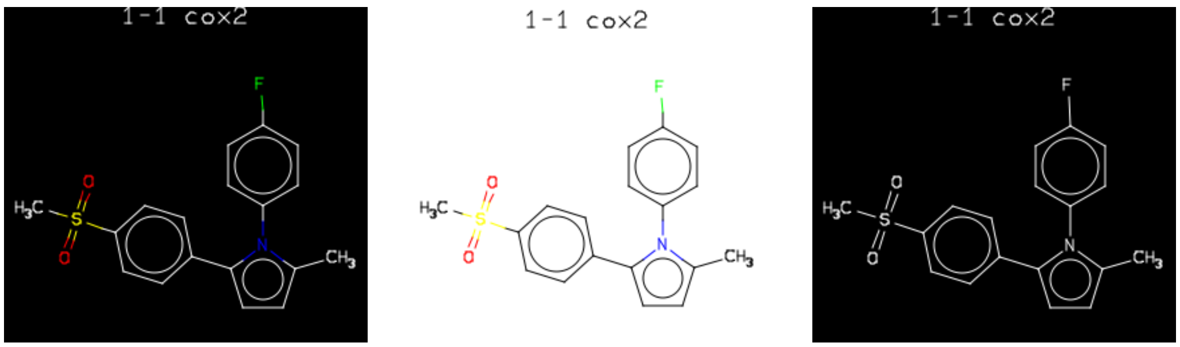


VIDA 3.0.0 for OpenEye (Worldwide)
File Edit View Data Style Bookmarks Tools Window Help
Focused Visible Marked All
List Window
All
1-1 cox2
1-3 cox2
1-4 cox2
1-5 cox2
1-6 cox2
1-7 cox2
1-8 cox2
1-9 cox2
1-10 cox2

Name	●	ID
cox2_good.sdf.gz		1
1-1 cox2	●	2
1-3 cox2	●	3
1-4 cox2	●	4
1-5 cox2	●	5
1-6 cox2	●	6
1-7 cox2	●	7
1-8 cox2	●	8
1-9 cox2	●	9
1-10 cox2	●	10
1-16 cox2		11
1-17 cox2		12
1-18 cox2		13
1-19 cox2		14
1-20 cox2		15
1-21 cox2		16
1-22 cox2		17
1-26 cox2		18
1-27 cox2		19
1-28 cox2		20
1-29 cox2		21
1-30 cox2		22
1-31 cox2		23
1-32 cox2		24
1-33 cox2		25
1-34 cox2		26
1-35 cox2		27
1-36 cox2		28
1-37 cox2		29
1-38 cox2		30
1-39 cox2		31
1-40 cox2		32
1-41 cox2		33
1-42 cox2		34
1-43 cox2		35
2-20 cox2		36
3-5 cox2		37

```
Cc1ccc(n1c2ccc(cc2)F)c3ccc(cc3)S(=O)(=O)C
```

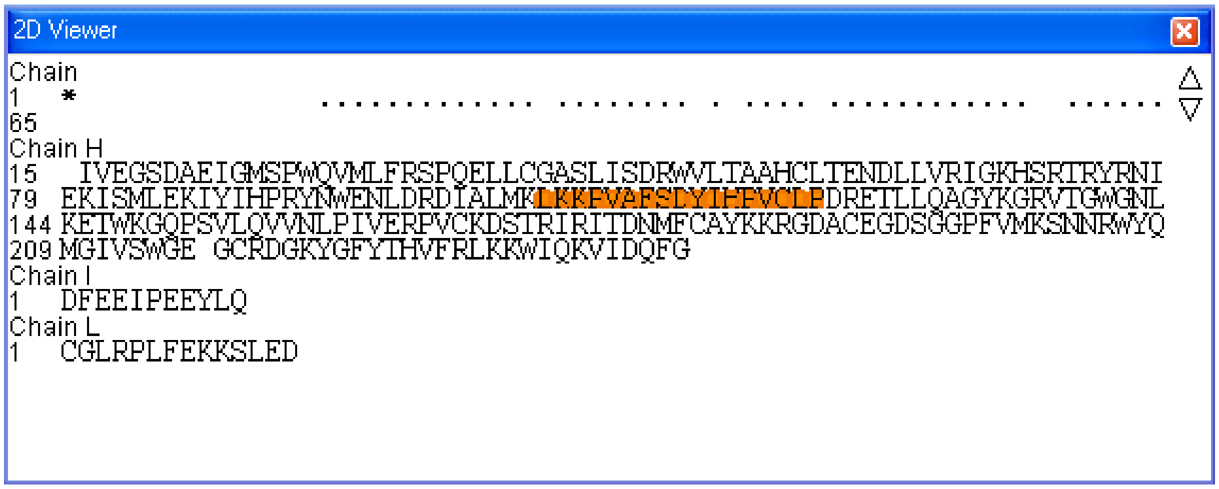
Figure 10.1: 2D Depiction



(a) Color on Black (b) Color on White (c) White on Black

Figure 10.2: Depiction color schemes

As might be expected, 2D depictions of this nature only make sense for small molecules (< 255 atoms in this case). For proteins, an amino acid sequence view is displayed instead (see Figure 10.3).



```
2D Viewer
Chain
1    *                 .............. ......... . .... ............. .......
65
Chain H
15    IVEGSDAEIGMSPWQVMLFRSPQELLCGASLISDRWVLTAAHCLTENDLLVRIGKHSRTRYRNI
79   EKISMLEKIYIHPRYNWENLDRDIALMKLKKPVAFSDYIHPVCLPDRETLLQAGYKGRVTGWGNL
144  KETWKGQPSVLQVVNLPIVERPVCKDSTRIRITDNMFCAYKKRGDACEGDSGGPFVMKSNNRWYQ
209  MGIVSWGE GCRDGKYGFYTHVFRLKKWIQKVIDQFG
Chain I
1    DFEEIPEEYLQ
Chain L
1    CGLRPLFEKKSLED
```

Figure 10.3: Amino Acid Sequence

10.1 Interaction

2D selection behaves much like selection in the 3D display (see Section 5.11.3). Left clicking will select an individual atom, bond, or residue (if in sequence mode). When viewing a depiction, the right mouse button allows lasso style selection, as in the 3D display. When viewing a sequence, holding down the *Shift* key when selecting will allow selection of a continuous range of residues.

The current selection is reflected in the coloring of the 2D depiction. There may be slight variations between the depicted selection in 2D and 3D due to the difference in how hydrogens are drawn.

CHAPTER

ELEVEN

Spreadsheet

The spreadsheet window is used to view, compute, and analyze data associated with molecules and can be seen in Figure 11.1. The spreadsheet window contains three initial spreadsheets by default: *Molecules*, *Proteins* and *Atoms*. The *Atoms* spreadsheet is slightly different in behavior from the other two and is described later.

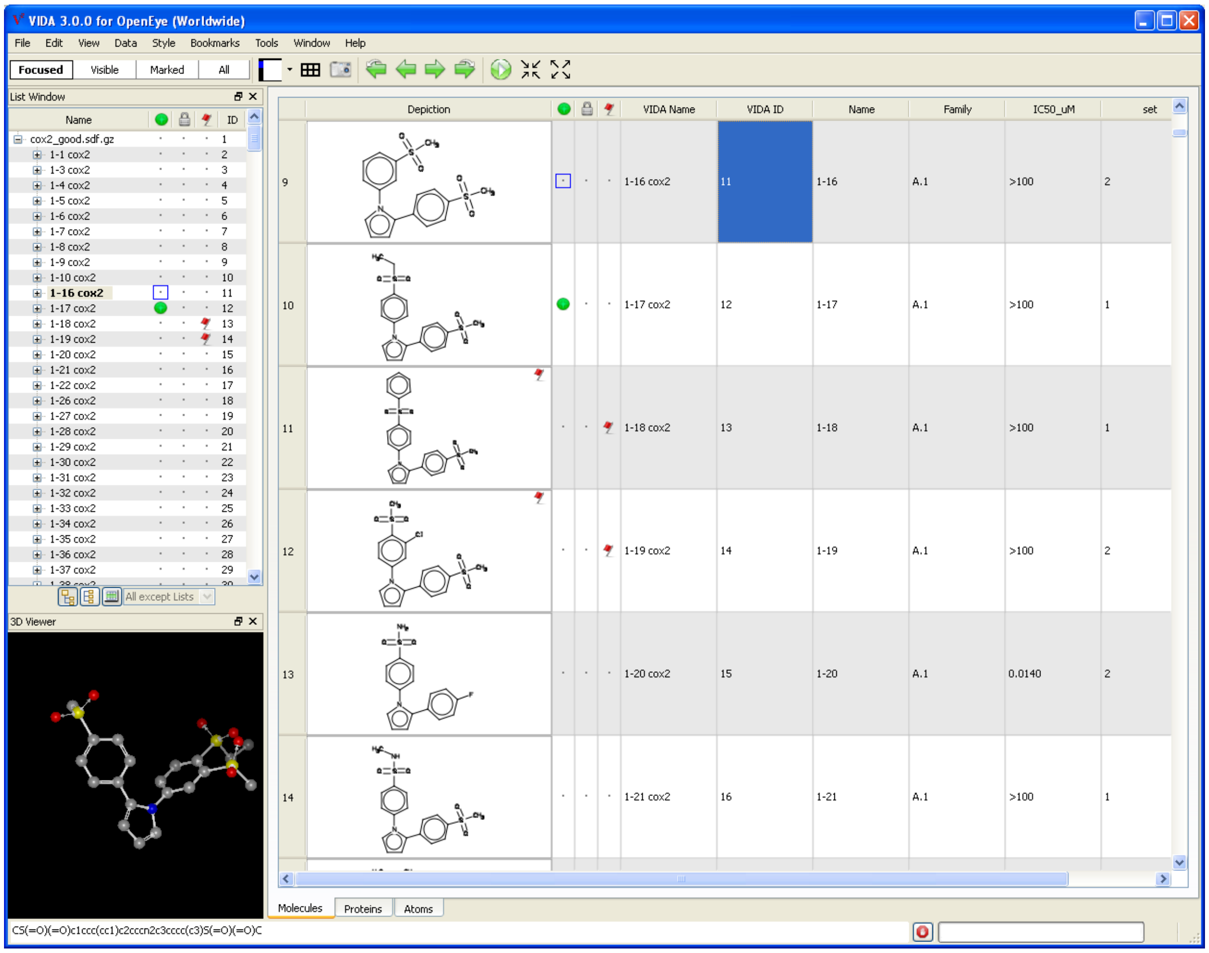
The *Molecules* spreadsheet is populated with all currently loaded small molecules (fewer than 255 atoms). Conversely, the *Proteins* spreadsheet is populated all currenty loaded molecules that have more than 255 atoms.

The columns and their associated values are populated from SD data. SD data can be found in both SDF and OEB files. Additional spreadsheet columns can also be added by computing expressions (see Section 11.5) or by importing external data from a comma-seperated value (.csv) or tab-seperated value file (.txt) (see Section 11.9).

11.1 Basic Usage

Columns are divided into two basic types: data and property. Data columns are those found in SD data and/or imported files. Property columns include:

- Depiction - This column contains an image of 2D structure of the associated molecule.
- VIDA ID - This column contains the unique identifier assigned to that object by VIDA.
- VIDA Name - This column contains the title of the molecule as sepcified in the input file.
- Visible - This column contains an indicator (green dot) showing whether the molecule is *Visible* (see Section 3.3.2).
- Marked - This column contains an indicator (red flag) showing whether the molecule is *Marked* (see Section 3.3.4).
- Locked - This column contains an indicator (padlock) showing whether the molecule is *Locked* (see Section 3.3.3).



Name	●			ID
cox2_good.sdf.gz	·	·	·	1
1-1 cox2	·	·	·	2
1-3 cox2	·	·	·	3
1-4 cox2	·	·	·	4
1-5 cox2	·	·	·	5
1-6 cox2	·	·	·	6
1-7 cox2	·	·	·	7
1-8 cox2	·	·	·	8
1-9 cox2	·	·	·	9
1-10 cox2	·	·	·	10
1-16 cox2	☐	·	·	11
1-17 cox2	●	·	·	12
1-18 cox2	·	·	🚩	13
1-19 cox2	·	·	🚩	14
1-20 cox2	·	·	·	15
1-21 cox2	·	·	·	16
1-22 cox2	·	·	·	17
1-26 cox2	·	·	·	18
1-27 cox2	·	·	·	19
1-28 cox2	·	·	·	20
1-29 cox2	·	·	·	21
1-30 cox2	·	·	·	22
1-31 cox2	·	·	·	23
1-32 cox2	·	·	·	24
1-33 cox2	·	·	·	25
1-34 cox2	·	·	·	26
1-35 cox2	·	·	·	27
1-36 cox2	·	·	·	28
1-37 cox2	·	·	·	29

	Depiction	●			VIDA Name	VIDA ID	Name	Family	IC50_uM	set
9		☐	·	·	1-16 cox2	11	1-16	A.1	>100	2
10		●	·	·	1-17 cox2	12	1-17	A.1	>100	1
11		·	·	🚩	1-18 cox2	13	1-18	A.1	>100	1
12		·	·	🚩	1-19 cox2	14	1-19	A.1	>100	2
13		·	·	·	1-20 cox2	15	1-20	A.1	0.0140	2
14		·	·	·	1-21 cox2	16	1-21	A.1	>100	1

CS(=O)(=O)c1ccc(cc1)c2cccn2c3cccc(c3)S(=O)(=O)C

Figure 11.1: Spreadsheet window

Clicking on any data cell in the spreadsheet will cause the associated molecule to become the *Focused* molecule (see Section 3.3.1). Double-clicking on a cell in the spreadsheet enables modification of the data value in that particular cell. Some columns cannot be changed, such as the “VIDA ID” or “Depiction” columns.

Clicking on a cell and the holding the mouse button down while dragging selects a rectangular region in the spreadsheet. This region can be copied and pasted into other spreadsheets. The depiction column will be automatically converted to a SMILES string when feasible (molecule has fewer than 100 atoms). Similarly, clicking the mouse on a molecule’s depiction and dragging the mouse to another application will transfer that image to the other application. This provides an easy way to transfer molecular images to other documents.

Clicking on a cell in the property columns *Visible*, *Marked* or *Locked* will toggle the respective property. To assign a property to a selected range of cells, simply select the range and then right click on the spreadsheet and select the appropriate action from the popup menu.

11.2 Sorting

A spreadsheet can be sorted by clicking on a column header. That spreadsheet will be sorted according to the data values in the associated column. If the data in the column to be sorted contains both numeric and non-numeric values, each set (numeric and non-numeric) will be sorted independently and then concatenated together.

Multiple columns can be sorted simultaneously by selecting the “Sort Spreadsheet...” option from the top level *Data* menu. This will launch a dialog that can be seen in Figure 11.2. This dialog can also be launched by right-clicking on any column header and selecting the “Sort Spreadsheet...” option from the generated popup menu.

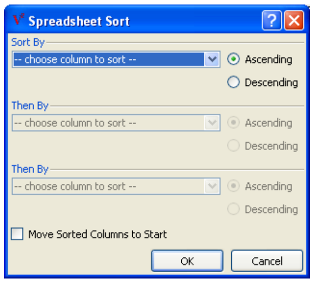

Spreadsheet Sort
Sort By
-- choose column to sort --
[✓ Ascending]
Descending
Then By
-- choose column to sort --
Ascending
Descending
Then By
-- choose column to sort --
Ascending
Descending
Move Sorted Columns to Start
OK
Cancel

Figure 11.2: Sorting the spreadsheet

This sorting dialog allows for the selection of up to three columns. The spreadsheet will be sorted by the first column selected, with ties being broken by the remaining columns. The sorted columns can be moved to the front of the spreadsheet by selecting the “Move Sorted Columns to Start” option in the dialog.

The first time the spreadsheet is sorted, it may take longer than expected as the data values need to be retrieved from the molecule and any expressions must be calculated. The progress of the sort is indicated in the progress bar at the bottom of the application. If desired, the sort can be aborted by clicking on the stop button to the left of the progress bar.

11.3 Displaying Data

The visibility of columns can be toggled by selecting the associated column name from the right-click popup menu generated by clicking on any column header. Data column names can be found in the “Column Visibility” submenu. Hiding the depiction column can allow for more rows to be displayed at the same time.

Certain molecular properties, such as molecular weight, can be added as new columns by selecting the desired option from the “Molecular Properties” submenu in the right-click popup menu. Some of these properties, such as Energy, might not be set on the molecule and as such the associated cell will contain an empty value.

See Section 11.5.2 for a complete listing of the molecule properties available for calculation.

11.4 Organizing Columns

Columns can be organized in the spreadsheet by selecting the “Organize Columns” option in the top level *Data* menu. This will launch a dialog which can be seen in figure 11.3.

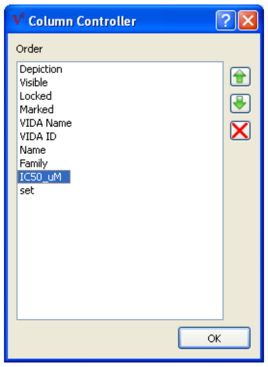
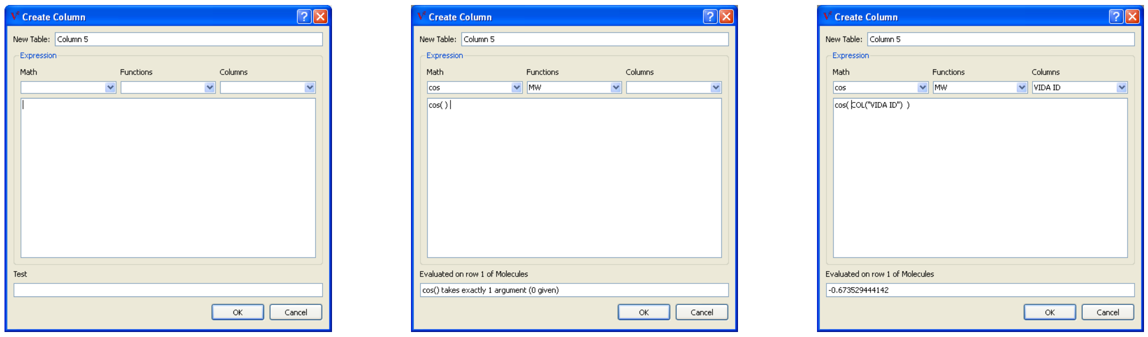


Figure 11.3: Spreadsheet organization dialog

The order of the columns can be controlled by selecting individual columns and then clicking on the up or down arrow buttons to change that columns position in the order. Furthermore, columns can be deleted by clicking on the button with the red X icon. Please be aware that deleting a column will delete the associated data from all currently loaded molecules and spreadsheets.

11.5 Creating New Columns



New Table: Column 5

New Table: Column 5
Math: cos
Functions: MW
cos( )
Evaluated on row 1 of Molecules
cos() takes exactly 1 argument (0 given)

New Table: Column 5
Math: cos
Functions: MW
Columns: VIDA ID
cos(COL("VIDA ID"))
Evaluated on row 1 of Molecules
-0.673529444142

(a) Creating a new column (b) Using a math function in the expression (c) Referencing another column

Figure 11.4: Creating a new column from an expression

New columns can be added to the spreadsheet containing values that are based on or calculated from molecular properties or existing spreadsheet data. To create a new columns, select the "Create Column" option in the top level *Data* menu. This will launch a dialog to guide the process and can be seen in Figure 11.4.

The text entry field at the top contains the name for the new column. A new, empty and editable column can be easily be created by supplying a new column name and then clicking the "Create" button.

Beneath the name is a collection of three drop down boxes containing functions which can be used in building the expression to be evaluated when populating the column. The first drop down box contains a collection of available mathematical functions which can be used in the expression (see Section 11.5.1 for more details). The second drop down box contains a collection of available molecular properties that can be used in the expression (see Section 11.5.2 for more details). The last drop down box contains a list of the other already existing columns which can be used as part of the expression.

Any of the above described functions can be added to the expression simply by selecting them in the appropriate drop down box. If that function requires additional input (such as functions like *cos*), the cursor in the expression editor window will automatically be placed inside the function. Functions can be combined with standard math operators such as *, +, -, and . If a function requires additional input that is not provided, an error will be displayed in the Test section at the bottom of the dialog as seen in Figure 11.4(b). Frequently, the desired input is contained within another column which can be referenced by selecting the desired column from the third drop down box or simply by typing "COL('COLUMN NAME')" as seen in Figure 11.4(c).

Once the desired expression has been assembled, the column can be created and added to the spreadsheet by clicking on the "Create" button.

Name	Description
acos(x)	Return the arc cosine (measured in radians) of x.
asin(x)	Return the arc sine (measured in radians) of x.
atan(x)	Return the arc tangent (measured in radians) of x.
atan2(y,x)	Return the arc tangent (measured in radians) of y/x.
	Unlike atan(y/x), the signs of both x and y are considered.
ceil(x)	Return the ceiling of x as a float. This is the smallest integral value >= x.
cos(x)	Return the cosine of x (measured in radians).
cosh(x)	Return the hyperbolic cosine of x.
degrees(x)	Converts angle x from radians to degrees.
e	The numeric constant "e" or 2.7182818284590451.
exp(x)	Return e^x.
fabs(x)	Return the absolute value of the float x.
floor(x)	Return the floor of x as a float. This is the largest integral value <= x.
fmod (x,y)	Return the floating point remainder of x/y.
hypot(x,y)	Return the Euclidean distance: $\sqrt{x*x+y*y}$.
ldexp(x,y)	Returns $x*2^i$.
log(x[,base])	The logarithm of x to the given base. If the base is not specified, returns ln(x).
log10(x)	The base 10 logarithm of x.
pi	The numeric constant PI or 3.1415926535897931.
pow(x,y)	Return x^y.
radians(x)	Converts angle x from degrees to radians.
sin(x)	Return the sine of x (measured in radians).
sinh(x)	Return the hyperbolic sine of x.
sqrt(x)	Return $\sqrt{x}$.
tan(x)	Return the tangent of x (measured in radians).
tanh(x)	Return the hyperbolic tangent of x.

Table 11.1: Available math functions

11.5.1 Math Functions

A large number of math functions are available for use in spreadsheet expressions as described in the previous section. The currently available math functions can be found in Table 11.1.

11.5.2 OEChem Functions

A large number of molecular properties are available for use in spreadsheet expressions as described above. The currently available molecular properties can be found in Table 11.2.

Name	Property
MW	Molecular weight.
Num Atoms	Number of atoms in the molecule.
Num Bonds	Number of bonds in the molecule.
Carbon-Hetero Ratio	Ratio of carbons to hetero atoms in the molecule.
	Returns 1 if there are no carbons in the molecule.
Energy	Molecular energy of the molecule as specified in the input file.
Actual Charge	Sum of the partial charges on all atoms.
Formal Charge	Sum of the formal charges on all atoms.
Halide Count	Number of halogen atoms in the molecule.
Num Carbons	Number of carbon atoms in the molecule.
Num Formal Charges	Number of atoms with a specified formal charge.
Num Heavy Atoms	Number of heavy atoms (non hydrogen) in the molecule.
Num Hetero Atoms	Number of hetero atoms in the molecule.
Num Hydrogens	Number of hydrogen atoms in the molecule.
Num Rigid Bonds	Number of rigid bonds in the molecule.
Nom Rotatable Bonds	Number of rotatable bonds in the molecule.
Num Chiral Atoms	Number of chiral atoms in the molecule.
Num Chiral Bonds	Number of chiral bonds in the molecule.

Table 11.2: Available molecular properties

11.6 Filtering

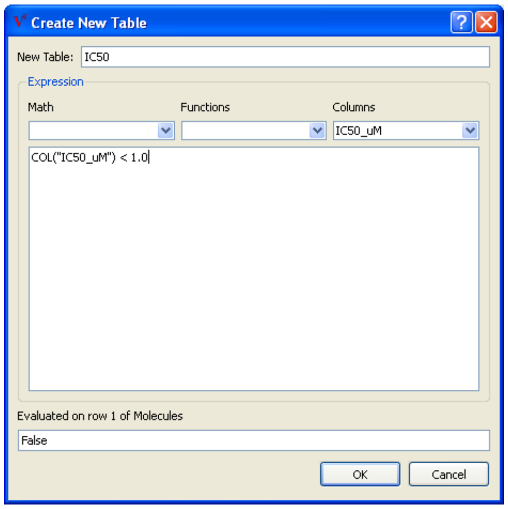
New spreadsheets may be generated from existing one by means of filtering. The mechanism to filter a spreadsheet is very similar to that of create a new column (see Section 11.5). To create a filtered spreadsheet, select the "Filter Spreadsheet..." option from the top level *Data* menu. This will launch a dialog which can be seen in Figure 11.5. This dialog will assist in the creation of a new filtered spreadsheet based on the currently viewed spreadsheet. Please note, that it is possible to create filtered spreadsheets on other filtered spreadsheets. The result of a filtering operation can be seen in Figure 11.6.

Filter expressions are very similar to column generation expressions (see section 11.5), except that the return value of a filter expression determines whether or not that row will be included in the new spreadsheet. Rows for which the expression returns either *False* or *0* will not be included.

11.7 Statistics

Statistics for each numeric column can be viewed in the spreadsheet window. The display of statistics is controlled by toggling the "Show Statistics" option in the top level *Data* menu. Toggling this option controls the display of statistics for just the currently viewed spreadsheet.

The following statistical information is computed for each numeric column:



Create New Table

New Table: IC50

Expression

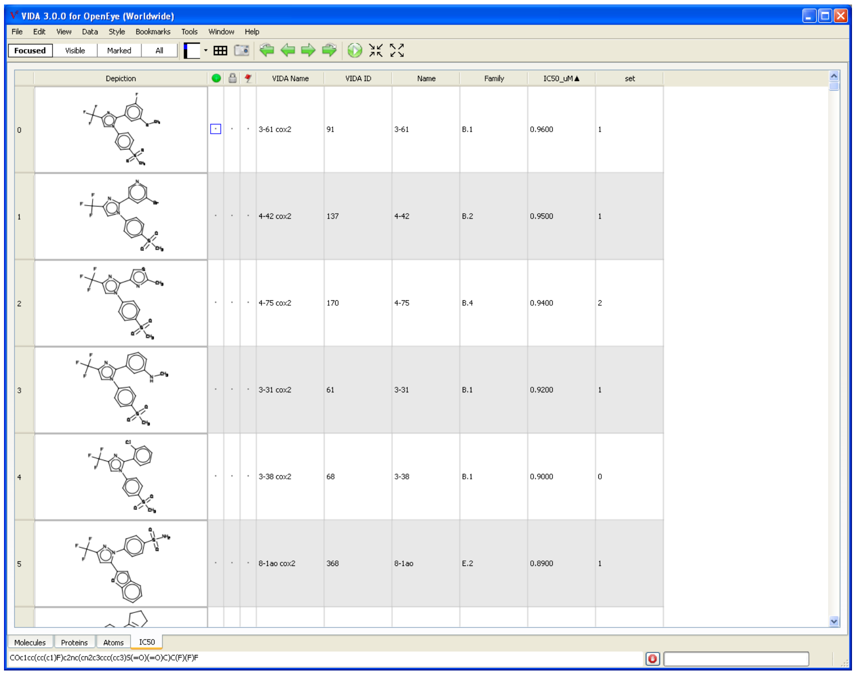
Math | Functions | Columns: IC50_uM

```
COL("IC50_uM") < 1.0
```

Evaluated on row 1 of Molecules

False

Figure 11.5: Filter dialog



VIDA 3.0.0 for OpenEye (Worldwide)

File Edit View Data Style Bookmarks Tools Window Help
Focused Visible Marked All

	Depiction	●			VIDA Name	VIDA ID	Name	Family	IC50_uM▲	set
0		·	·	·	3-61 cox2	91	3-61	B.1	0.9600	1
1		·	·	·	4-42 cox2	137	4-42	B.2	0.9500	1
2		·	·	·	4-75 cox2	170	4-75	B.4	0.9400	2
3		·	·	·	3-31 cox2	61	3-31	B.1	0.9200	1
4		·	·	·	3-38 cox2	68	3-38	B.1	0.9000	0
5		·	·	·	8-1ao cox2	368	8-1ao	E.2	0.8900	1

Molecules
Proteins
Atoms
IC50

COc1cc(cc(c1)F)c2nc(cn2c3ccc(cc3)S(=O)(=O)C)C(F)(F)F

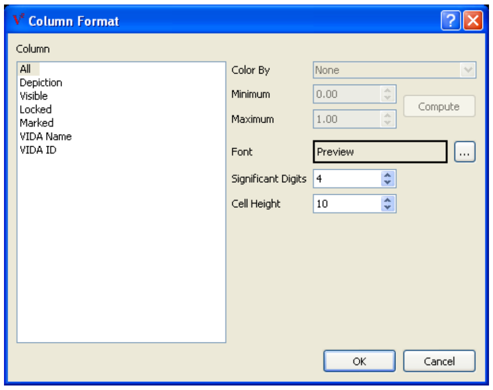
Figure 11.6: A filtered spreadsheet

- *min* - The minimum value of the column.
- *mean* - The maximum value of the column.
- *stddev* - The standard deviation of the column.
- *skewness* - The skewness of the column.
- *kurtosis* - The kurtosis of the column.

The statistics display is a dynamic view of the spreadsheet data and therefore when data is altered in the spreadsheet the statistics view is automatically updated. For large data sets, this may take some time and could cause editing the spreadsheet to be cumbersome. To avoid this problem, it is best to hide the statistics display until the editing process is complete.

11.8 Formatting Columns

There are a variety of different display styles for spreadsheet columns which can be edited on an individual basis. The customizable styles include color, font, significant digits, and cell height. The style for individual columns can be edited by selecting the “Format Columns...” option in the top level *Data* menu. This will launch the dialog seen in Figure 11.7.



Color By: None
Minimum: 0.00
Maximum: 1.00
Font: Preview
Significant Digits: 4
Cell Height: 10

Figure 11.7: Spreadsheet format dialog

On the left hand side of the dialog is a list of all the currently available spreadsheet columns, including an *All* option. Any formatting changes made in this dialog will be applied to the selected columns in this list.

On the right hand side of the dialog is a collection of formatting and display options. At the top is a drop down menu of available coloring schemes which include:

- *Red to Blue*
- *Blue to Red*

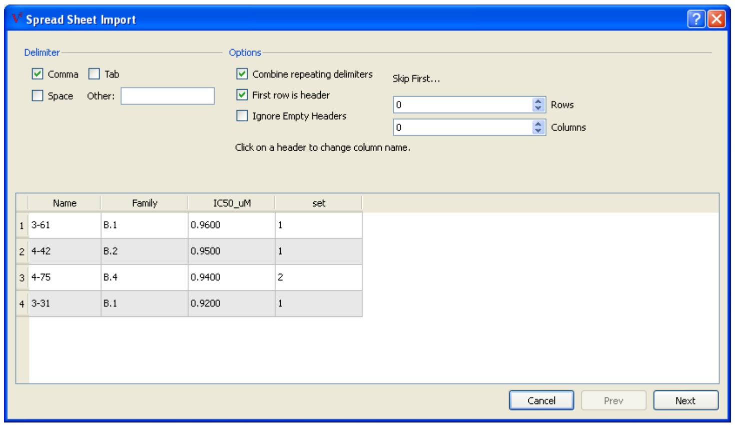
- *Rainbow*
- *Rainbow Backwards*
- *Red Yellow Blue*
- *Grey to Green*

The selected coloring scheme will be applied to the column using the scale specified beneath the drop down menu determined by the minimum and maximum values. The default values are 0.0 and 1.0 respectively, but the current minimum and maximum values for the selected column can be computed by clicking on the “Compute” button to the right of the minimum and maximum fields.

11.9 Importing/Exporting

11.9.1 Importing

Spreadsheet data can be imported from external files in a variety of formats including comma-separated values (.csv) and tab-delimited values (.txt). To import a file, select the “Spreadsheet...” option in the *Import* submenu of the top level *File* menu. This will launch a dialog to assist in the import process which can be seen in Figure 11.8.



Delimiter

[✓ Comma] Tab
Space Other:

Options

[✓ Combine repeating delimiters]
[✓ First row is header]
Ignore Empty Headers

Skip First...
0 Rows
0 Columns

Click on a header to change column name.

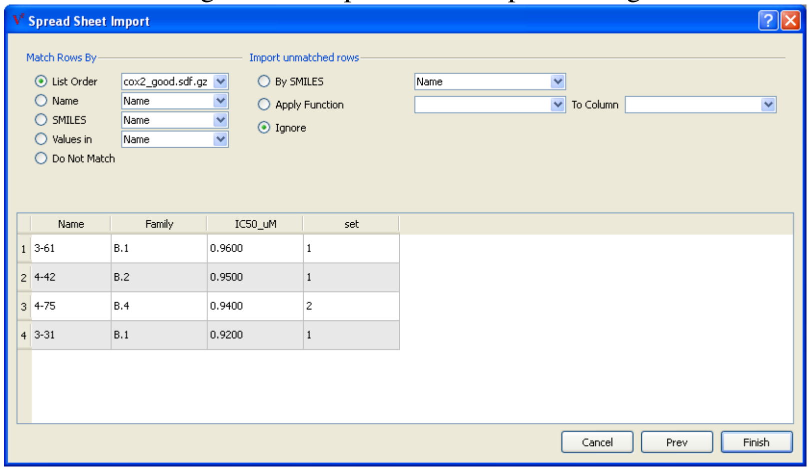
	Name	Family	IC50_uM	set
1	3-61	B.1	0.9600	1
2	4-42	B.2	0.9500	1
3	4-75	B.4	0.9400	2
4	3-31	B.1	0.9200	1

Figure 11.8: Spreadsheet import dialog

At the top of the dialog are a number of options to specify the format of the file to be imported and at the bottom is a quick preview of what AFITT thinks the first five rows are. The preview will update dynamically to match any changes made to the options at the top.

Once the preview of file to be imported is satisfactory, click on the *Next* button to specify how the rows are to be matched to currently loaded molecules and what to do with rows that do not match (see Figure 11.9).

Figure 11.9: Spreadsheet Import Dialog



Match Rows By

[✓ List Order] cox2_good.sdf.gz
Name: Name
SMILES: Name
Values in: Name
Do Not Match

Import unmatched rows

By SMILES: Name
Apply Function: To Column
[✓ Ignore]

	Name	Family	IC50_uM	set
1	3-61	B.1	0.9600	1
2	4-42	B.2	0.9500	1
3	4-75	B.4	0.9400	2
4	3-31	B.1	0.9200	1

There are a number of mechanisms available to match data rows to the appropriate molecules in AFITT:

- List Order - matches the order of rows in the spreadsheet with the order of molecules in the selected list.
- Name - matches the molecule's name with the name in the selected column of the imported data.
- SMILES - matches the molecule's canonical SMILES string with the SMILES string in the selected column of the imported data.
- Values In - matches the value in the selected column between the imported data and the value already in the spreadsheet.
- Do Not Match - assume that you are importing new molecules

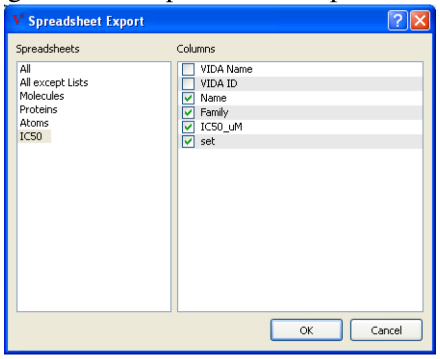
Unmatched rows can also be imported into the spreadsheet by creating a new molecule in AFITT to associated with that row. The following options are available for unmatched rows:

- By SMILES - use the SMILES string in the selected column of the imported data to create a new molecule.
- Apply Function - Apply a Python function to the data in the selected column of the imported data to create a new molecule. This can be particularly useful in cases where one of the columns contains a database ID which can be used to retrieve molecules.
- Ignore - Do not import unmatched data.

11.9.2 Exporting

Spreadsheets may be exported to an external file by selecting the "Spreadsheet..." option in the *Export* submenu of the top level *File* menu. Selecting this option will launch a dialog which allows for specification of which columns to export (see Figure 11.10). Supported formats include either comma separated (.csv) or tab delimited (.txt).

Figure 11.10: Spreadsheet Export Dialog



Spreadsheets: All, All except Lists, Molecules, Proteins, Atoms, IC50

Columns:
- [ ] VIDA Name
- [ ] VIDA ID
- [✓] Name
- [✓] Family
- [✓] IC50_uM
- [✓] set

CHAPTER

TWELVE

List Management

The organization of molecules into lists and the ability to easily browse through these lists is an extremely important and central concept to AFITT. As such a List Window is provided to aid in the navigation and organization of lists. Furthermore, there exist many mechanisms to create, modify, and delete lists which will be discussed in more detail later in this chapter.

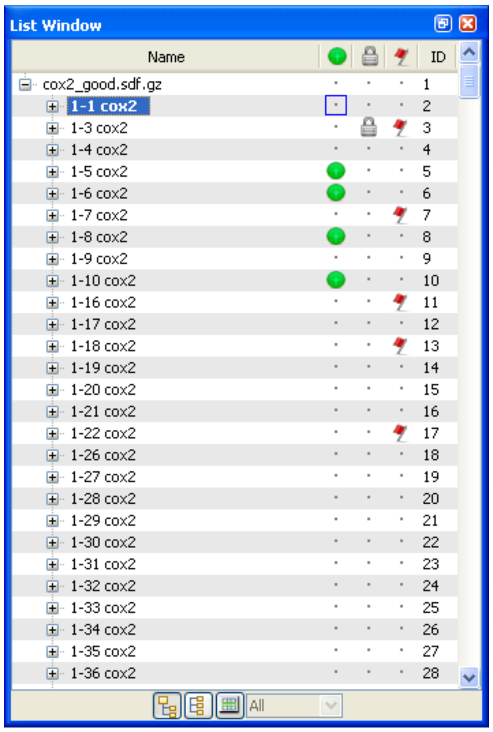
12.1 Introduction

The List Window is one of the primary navigation and organization mechanisms in AFITT. It displays all of the currently loaded files and their contents (molecules, grids, surfaces, etc.) into hierarchical lists for easy browsing and organization.

The List Window can be seen in Figure 12.1 and contains multiple columns for controlling the various states and properties of the associated objects. The first column displays the name. The second column (with the green sphere icon at the top) is the *Visible* state column (see Section 3.3.2). The third column (with the padlock icon at the top) is the *Locked* state column (see Section 3.3.3). The fourth column (with the flag icon at the top) is the *Marked* state column (see Section 3.3.4). The fifth and last column contains the unique identifier assigned to that object.

12.2 Browsing

Every list in the List Window, as well as many of the other objects, can be expanded in the List Window to reveal further detail or the members contained within them. A small box with a "+" sign inside to the left of an object's name indicates that that object can be expanded. Molecules, for example, may be expanded to reveal conformers, molecular structure, or potentially other related children objects such as surfaces or grids. A single mouse click on the box will expand that object to reveal its contents. Double-clicking on the object name will turn on in-place renaming of that object.


List Window

Name	●	🔒	🚩	ID
cox2_good.sdf.gz	·	·	·	1
1-1 cox2	☐	·	·	2
1-3 cox2	·	🔒	🚩	3
1-4 cox2	·	·	·	4
1-5 cox2	●	·	·	5
1-6 cox2	●	·	·	6
1-7 cox2	·	·	🚩	7
1-8 cox2	●	·	·	8
1-9 cox2	·	·	·	9
1-10 cox2	●	·	·	10
1-16 cox2	·	·	🚩	11
1-17 cox2	·	·	·	12
1-18 cox2	·	·	🚩	13
1-19 cox2	·	·	·	14
1-20 cox2	·	·	·	15
1-21 cox2	·	·	·	16
1-22 cox2	·	·	🚩	17
1-26 cox2	·	·	·	18
1-27 cox2	·	·	·	19
1-28 cox2	·	·	·	20
1-29 cox2	·	·	·	21
1-30 cox2	·	·	·	22
1-31 cox2	·	·	·	23
1-32 cox2	·	·	·	24
1-33 cox2	·	·	·	25
1-34 cox2	·	·	·	26
1-35 cox2	·	·	·	27
1-36 cox2	·	·	·	28

All

Figure 12.1: List window

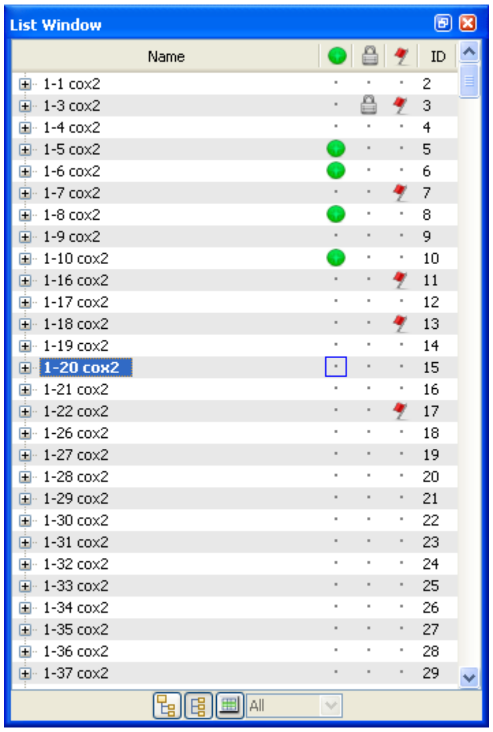
12.3 Views

The List Window provides three different views onto the loaded data. Which view is shown is controlled by the three buttons at the bottom of the List Window. The first view is a traditional hierarchical view, described above, and seen in Figure 12.1. The second view is a flat view (see Figure 12.2(a)) in which the list hierarchy has been removed, but all the contents are shown in the same original order. The third view is a spreadsheet view (see Figure 12.2(b)) which shows only those objects that appear within the associated spreadsheet (as specified by the drop down box next to its button). Furthermore, the order of the objects in this view parallels the current sorting order in the application spreadsheet. For more details about the spreadsheet, please see Chapter 11.

12.4 List Columns

Every object in the List Window may populate one of the several columns described in the introduction. Some columns are simply informative, such as the unique identifier, while others are functional as well as informative.

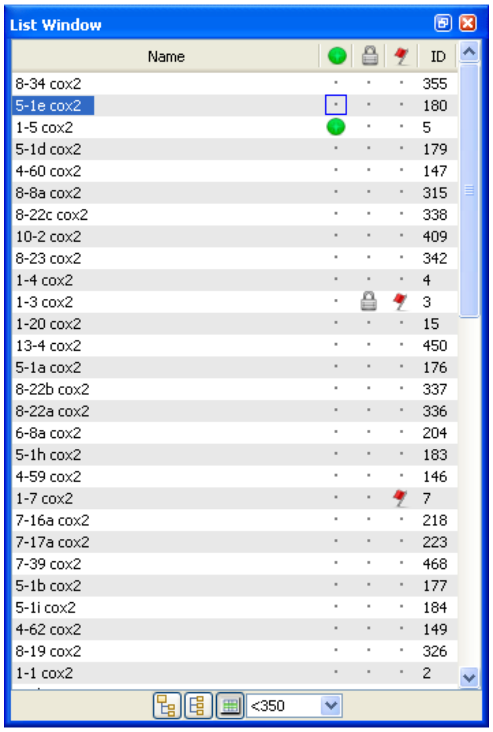
Clicking on an object's name will make that object the *Focused* object (see Section 3.3.1). In the List Window, the *Focused* object's name is displayed in bolded text and there will be a small blue box around the dot in the second column (the *Visible* column). An example of this can be seen in Figure 12.1, where


List Window

Name	●	🔒	🚩	ID
1-1 cox2	·	·	·	2
1-3 cox2	·	🔒	🚩	3
1-4 cox2	·	·	·	4
1-5 cox2	●	·	·	5
1-6 cox2	●	·	·	6
1-7 cox2	·	·	🚩	7
1-8 cox2	●	·	·	8
1-9 cox2	·	·	·	9
1-10 cox2	●	·	·	10
1-16 cox2	·	·	🚩	11
1-17 cox2	·	·	·	12
1-18 cox2	·	·	🚩	13
1-19 cox2	·	·	·	14
1-20 cox2	·	·	·	15
1-21 cox2	·	·	·	16
1-22 cox2	·	·	🚩	17
1-26 cox2	·	·	·	18
1-27 cox2	·	·	·	19
1-28 cox2	·	·	·	20
1-29 cox2	·	·	·	21
1-30 cox2	·	·	·	22
1-31 cox2	·	·	·	23
1-32 cox2	·	·	·	24
1-33 cox2	·	·	·	25
1-34 cox2	·	·	·	26
1-35 cox2	·	·	·	27
1-36 cox2	·	·	·	28
1-37 cox2	·	·	·	29

All

(a) Flat view


List Window

Name	●	🔒	🚩	ID
8-34 cox2	·	·	·	355
5-1e cox2	·	·	·	180
1-5 cox2	●	·	·	5
5-1d cox2	·	·	·	179
4-60 cox2	·	·	·	147
8-8a cox2	·	·	·	315
8-22c cox2	·	·	·	338
10-2 cox2	·	·	·	409
8-23 cox2	·	·	·	342
1-4 cox2	·	·	·	4
1-3 cox2	·	🔒	🚩	3
1-20 cox2	·	·	·	15
13-4 cox2	·	·	·	450
5-1a cox2	·	·	·	176
8-22b cox2	·	·	·	337
8-22a cox2	·	·	·	336
6-8a cox2	·	·	·	204
5-1h cox2	·	·	·	183
4-59 cox2	·	·	·	146
1-7 cox2	·	·	🚩	7
7-16a cox2	·	·	·	218
7-17a cox2	·	·	·	223
7-39 cox2	·	·	·	468
5-1b cox2	·	·	·	177
5-1i cox2	·	·	·	184
4-62 cox2	·	·	·	149
8-19 cox2	·	·	·	326
1-1 cox2	·	·	·	2

<350

(b) Spreadsheet view

Figure 12.2: Alternate views

the molecule "1-1 cox2" is *Focused*.

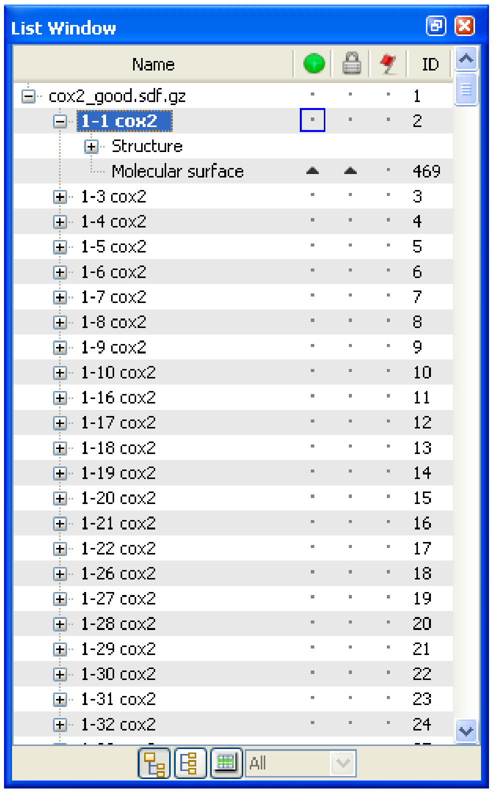
The second column is the *Visible* column. The *Visible* column has a green dot icon in its column header. This column displays and controls the *Visible* property for objects. If an object is *Visible* there will be a green dot identical to the one in the column header present in that object's *Visible* column. If that object is also the *Focused* object, there will be a small blue box around the dot in that column as well.

If a given object has children, the visibility of its children can also be controlled in this column. Child objects have an additional layer of control regarding visibility in that they can be hidden, visible, or can mimic its parent's visibility. If an object is set to mimic's its parents visibility, an upright triangle is shown instead of the standard green dot (as can be seen in Figure 12.3). Clicking in this column cycles through these three states in the following order: hidden, mimic, visible. Please note that molecular structure (atoms, residues, chains, etc) are not considered children objects and as such always mimic the parent molecule's visibility if they are marked visible with a green dot.

The third column is the *Locked* column. The *Locked* column has a small, grey padlock icon in its header. If an object is *Locked* there will be a grey padlock identical to the one in the column header present in that object's *Locked* column.

The fourth column is the *Marked* column. The *Marked* column has a red check mark icon in its header. If an object is *Marked* there will be a red check mark identical to the one in the column header present in that object's *Marked* column.

The fifth and final column is the *ID* column. An object's ID is simply a unique identifier that can be


List Window

Name	●	🔒	📌	ID
cox2_good.sdf.gz	·	·	·	1
1-1 cox2	·	·	·	2
Structure				
Molecular surface	▲	▲	·	469
1-3 cox2	·	·	·	3
1-4 cox2	·	·	·	4
1-5 cox2	·	·	·	5
1-6 cox2	·	·	·	6
1-7 cox2	·	·	·	7
1-8 cox2	·	·	·	8
1-9 cox2	·	·	·	9
1-10 cox2	·	·	·	10
1-16 cox2	·	·	·	11
1-17 cox2	·	·	·	12
1-18 cox2	·	·	·	13
1-19 cox2	·	·	·	14
1-20 cox2	·	·	·	15
1-21 cox2	·	·	·	16
1-22 cox2	·	·	·	17
1-26 cox2	·	·	·	18
1-27 cox2	·	·	·	19
1-28 cox2	·	·	·	20
1-29 cox2	·	·	·	21
1-30 cox2	·	·	·	22
1-31 cox2	·	·	·	23
1-32 cox2	·	·	·	24

All

Figure 12.3: Child object mimicing parent

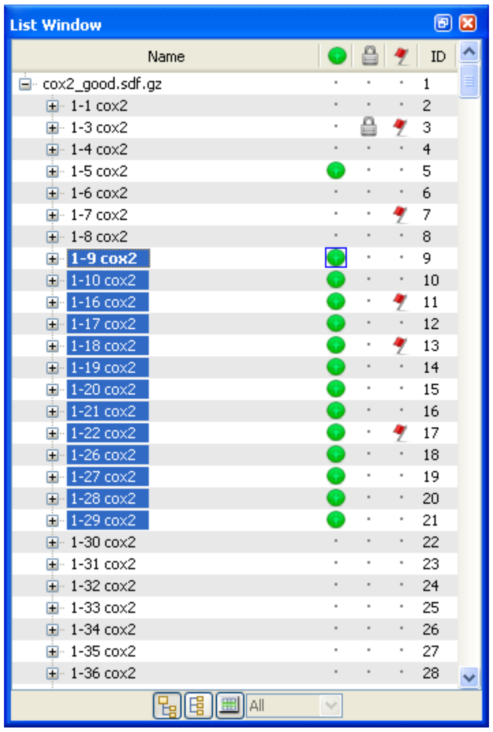
useful to identify specific objects when running various commands. Each top level object has its own unique ID, but child objects like conformers and atoms are all a part of their parent molecules's ID.

Columns can be shown or hidden as desired by right-clicking in the header of the List Window. These changes will be remembered in the Preferences (see Chapter 16) and thus persist between sessions.

The *Visible*, *Locked*, and *Marked* properties can be toggled by clicking on their associated columns in the List Window. If multiple objects are highlighted simultaneously and the click occurs in a column associated with the highlighted region, that change will apply to all of the highlighted objects. Compare Figure 12.4 to Figure 12.1. In Figure 12.4, the highlighted objects have been made *Visible* by clicking in that column. The *Focused* object has changed as a result as well.

12.5 List Manipulation

The List Window operates on the basic concept that every potentially viewable object is placed inside a list. As a result, the root items in the List Window are always lists. These lists can be combined, subsetted, and manipulated into new lists. The properties of all the objects in a list can be set together by clicking in the columns of their containing list. Finally, objects can easily be removed from and inserted into lists.



Name	●	🔒	🚩	ID
cox2_good.sdf.gz	·	·	·	1
1-1 cox2	·	·	·	2
1-3 cox2	·	🔒	🚩	3
1-4 cox2	·	·	·	4
1-5 cox2	●	·	·	5
1-6 cox2	·	·	·	6
1-7 cox2	·	·	🚩	7
1-8 cox2	·	·	·	8
1-9 cox2	●	·	·	9
1-10 cox2	●	·	·	10
1-16 cox2	●	·	🚩	11
1-17 cox2	●	·	·	12
1-18 cox2	●	·	🚩	13
1-19 cox2	●	·	·	14
1-20 cox2	●	·	·	15
1-21 cox2	●	·	·	16
1-22 cox2	●	·	🚩	17
1-26 cox2	●	·	·	18
1-27 cox2	●	·	·	19
1-28 cox2	●	·	·	20
1-29 cox2	●	·	·	21
1-30 cox2	·	·	·	22
1-31 cox2	·	·	·	23
1-32 cox2	·	·	·	24
1-33 cox2	·	·	·	25
1-34 cox2	·	·	·	26
1-35 cox2	·	·	·	27
1-36 cox2	·	·	·	28

Figure 12.4: Changing multiple objects simultaneously

12.5.1 Creating Lists

New lists can be created in several ways. The simplest way is the "New List" submenu in the *File* menu. A number of list creation mechanisms are available and include using the AND of two or more lists (intersection), the OR of two or more lists (union), the XOR of two lists (union - intersection), or from the currently *Marked* set of objects.

The AND list contains only those objects which are in all of the source lists. The OR list contains all of the objects contained in its individual source lists. The XOR list contains the objects which are in only one of the source lists, but not both. Finally, the *Marked* list creates a new list which contains all of the currently *Marked* objects.

The second way to create a new list is by right-clicking on an individual list and selecting the "Subset" submenu from the popup menu. This menu contains four entries – "By Marked", "By Query", "By Not Marked" and "By Not Query". The "By Marked" option creates a new list containing all of the *Marked* objects from within the list. The "By Not Marked" option creates a list with all of the objects from the list which are not *Marked*. "By Query" generates a dialog which prompts for a substructure query that if the objects match, those objects will be placed in the new list. The "By Not Query" behaves the same way, except that it puts the objects that do match the query into the new list.

The last way to create a new list is from the "Find" menu entry in the *Edit* menu. This mechanism searches all of the currently loaded molecules by either Amino Acid Sequence, SMARTS, a regular expression against the molecule title, or by a Query molecule (as a substructure search). After clicking the "Search" button, the matching results will be displayed on the right-hand side of the dialog. The

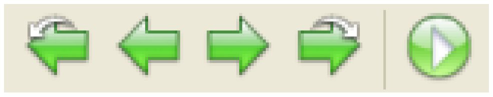


Figure 12.5: List navigation buttons

entire set or just specific individual results can be selected within the dialog to populate the new list.

12.5.2 Deleting from Lists

As previously mentioned, objects can also be removed from lists. There are two ways to remove an object from a list. The first mechanism is to delete the object (by selecting the “Delete” option from the right-click menu). Deleting an object will remove it from ALL of the lists to which it belongs. Choosing the “Remove from List” option from the same right-click menu will remove the object from the specified list only, leaving it perfectly intact in any other lists that it might belong to. When the last object is removed from a list, the list itself will be removed.

12.6 List Toolbar

This List Window installs a collection of five navigations button in the application toolbar which can be see in Figure 12.5. These buttons provide an additional mechanism to browse through lists and their contents in AFITT. In addition, these buttons are aware of the current list view as described in Section 12.3 and as such operate on that view.

The first four buttons in the set browse to the beginning, to the previous object, to the next object, and to the last object respectively. The final button with the “play” icon initiates a slideshow which steps forward to the next object after a certain period of time (this can be set in the application preferences). The slideshow can be paused by clicking on the same button again (which should have changed to have a “pause” icon once the slideshow started). The slideshow will automatically stop when it reaches the end of the objects in the current view of the List Window.

CHAPTER

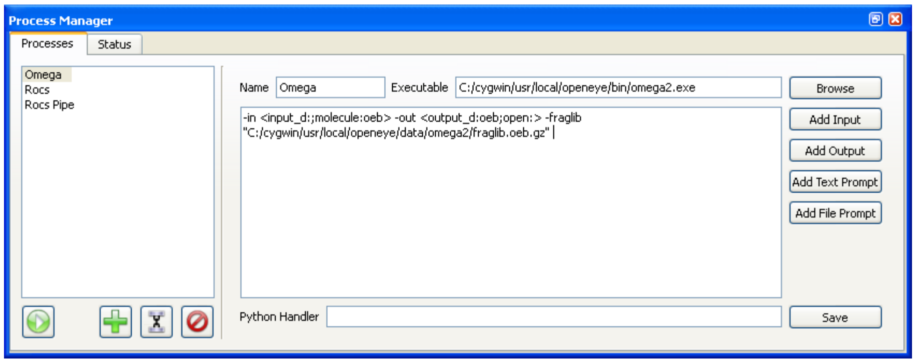
THIRTEEN

Process Management

The Process Manager allows the user to start, monitor, terminate, and load results from processes external to AFITT. As part of this functionality, it is possible to write out currently loaded molecules (or grids) to files which can be used as input to the external processes. Furthermore, it is possible to create “pipes” which chain together the input and output of individual processes into a larger process.

The Process Manager contains two tabbed windows called “Processes” and “Status” respectively as can be seen in Figures 13.1 and 13.2. The “Processes” tab contains the list of available processes and the functionality to create and edit them. The “Status” tab displays all of the currently active and recently completed processes and information about their state. It also displays the commandline output generated by each of the processes.

Beneath the list of processes in the “Processes” tab is a collection of four buttons. The button on the left with the play icon starts the currently selected process. The next button with the plus icon clears the editor on the right hand side to allow for creation of a new process. The next button with the link icon changes the editor on the right hand side to allow for creation of a new pipe (see Figure 13.6). The last button with the delete icon deletes the current selected set of processes from the list of available processes.

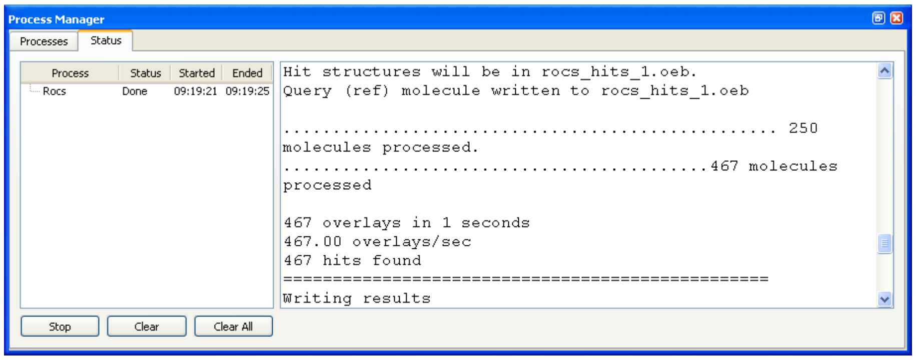


Name: Omega
Executable: C:/cygwin/usr/local/openeye/bin/omega2.exe

```
-in <input_d:;molecule:oeb> -out <output_d:oeb;open:> -fraglib
"C:/cygwin/usr/local/openeye/data/omega2/fraglib.oeb.gz"
```

Python Handler:

Figure 13.1: Process Manager



Process	Status	Started	Ended
Rocs	Done	09:19:21	09:19:25

```
Hit structures will be in rocs_hits_1.oeb.
Query (ref) molecule written to rocs_hits_1.oeb

.................................................. 250
molecules processed.
...........................................467 molecules
processed

467 overlays in 1 seconds
467.00 overlays/sec
467 hits found
=================================================
Writing results
```

Figure 13.2: Output generated by completed ROCS process

13.1 Processes

When creating a new process or editing a current one, there are a number of available options. At the top of the editor there is a field for the process name (which will be displayed in the list of available processes seen on the left as described above) as well as a field for the actual location of the application to be run. The “Browse” button to the right of this field can be used to find the location of the desired application.

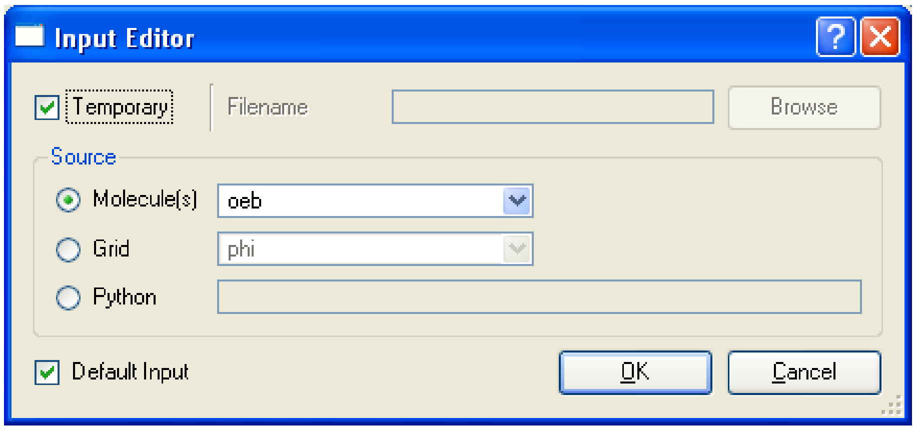
Beneath this is a large text entry box which allows for specification of any desired command line arguments to the application. It is not necessary to enter the application name here.

For most applications, at least one filename will have to be specified as a command line argument. This can be done multiple ways. If the file to be specified is always going to be located in the same location, the filename can be entered directly into the text entry box as a command line argument. If the location of the file is not likely to be constant, a prompt can be specified as opposed to an actual filename. To add a prompt, click on the “Add File Prompt” button to the right of text entry box. More details on adding prompts can be found in Section 13.1.3. If the desired file will need to be created each time, it is possible to create the desired file at start-time by writing out selected molecules (or grids) directly from AFITT. To use this feature, click on the “Add Input” button to the right of the text entry box. At start-time, the user will be prompted to select the desired molecules (or grids), which will then be saved to a file which will be passed as a command line argument to the process. For more details on adding an input file see Section 13.1.1.

In addition to being able to specify specific input files, it is possible to specify an output file which can potentially be read back into AFITT when the process is completed. As discussed above, it is possible to a specify specific filename in the command line or to use a prompt, but if it desired to load the resulting file, the “Add Output” button should be clicked to specify the file. For more details on adding an output file see Section 13.1.2.

Besides being able to specify filenames, it is often desirable to be able to specify other variable arguments at start-time. To do this, click on the “Add Text Prompt” button which will add a prompt argument to the command line arguments. At start-time the user will be asked to specify the desired text to be passed to the process. For more details on text prompts see Section 13.1.3.

Lastly, at the bottom of the Process Editor is a “Python Handler” field which allows for specification



[✓] Temporary | Filename

Source
(●) Molecule(s) oeb
( ) Grid phi
( ) Python

[✓] Default Input

Figure 13.3: Process Input Editor

of a Python function to be run when the process is completed. This allows for handling of output not normally handled by AFITT or for output which is not specified on the command line.

13.1.1 Adding Input

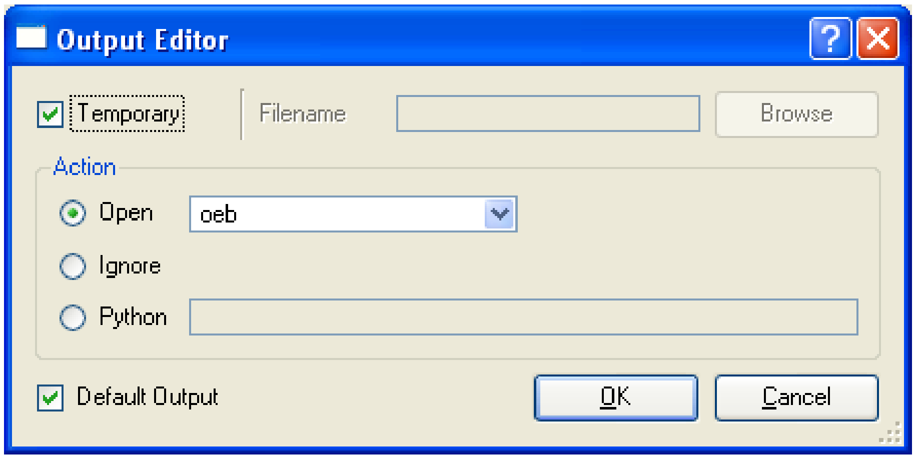
The Input Editor allows the user to specify a mechanism by which an input file is provided to the process to be run. At the top of the editor are two mutually exclusive options, the first is the “Temporary” check box which indicates that the file to be used will be a temporary file which will be removed after the process has completed. The other option is a field which allows specification of a specific filename to be used. In both cases, the user has a choice as to how this file will be populated: saving molecules, saving grids, or running a specified Python function which will handle the population of the file. If molecules or grids are chosen, a file format to save these in must be specified in the pull-down box next to the selected option.

If a Python function is specified to populate the input file, the desired filename can be obtained by using the keyword “@FILE@” in the function specification. This keyword will be substituted with the real filename at start-time before the Python function is called.

Finally, there is a checkbox at the bottom of the dialog to specify whether or not this is a default input. The default input entry is used when constructing pipes and will be discussed in Section 13.2.

13.1.2 Adding Output

The Output Editor allows the user to specify a mechanism by which an output file of the process to be run is handled by AFITT. At the top of the dialog are two mutually exclusive options, the first is the “Temporary” check box which indicates that the output file is a temporary file and will be removed after it has been handled by AFITT. The other option is a field which allows specification of a specific filename to be used for the output file. If this option is selected, the file will not be deleted after it is handled.



[✓] Temporary
Filename
Action: [✓ Open] oeb
Ignore
Python
[✓] Default Output

Figure 13.4: Process Output Editor

Once the above has been specified, the user can specify how AFITT should handle the file when the process is finished. There are three possible options, the first is to open the file, the second is to ignore the file, and the last is to call a Python function to handle the resulting file. If it is desired to open the file, the file format that the process should use to generate the output file must be specified so that AFITT will know how to properly load the results.

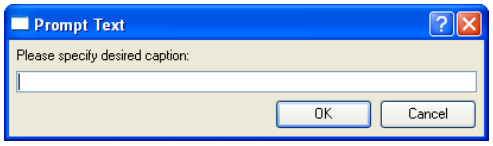
If a Python function is specified to handle the output file, the actual filename can be obtained by using the keyword "@FILE@" in the function specification. This keyword will be substituted with the real filename at process completion time before the Python function is called.

Finally, there is a checkbox at the bottom to specify whether or not this is a default output. The default output entry is used when constructing pipes and will be discussed in Section 13.2.

13.1.3 Adding Prompts

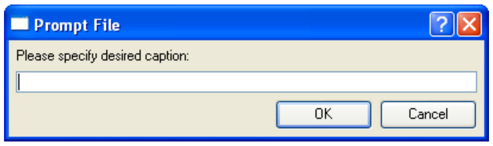
As described in Section 13.1 there are two types of prompts that are available for specification on the command line: a text prompt and a file prompt.

The Text Prompt Editor (see Figure 13.5(a)) allows the user to add a text prompt to the process as well as to specify what the caption of the prompt will be. Specification of an informative caption is helpful in allowing the user to know what information is being requested.



Please specify desired caption:

(a) Text Prompt Editor



Please specify desired caption:

(b) File Prompt Editor

Figure 13.5: Prompt Editors

The File Prompt Editor (see Figure 13.5(b)) allows the user to add a file prompt to the process as well as to specify what the caption of the resulting prompt will be. Specification of an informative caption is helpful in allowing the user to know what specific file or type of is being requested.

The result of both of these prompts are passed as command line arguments to the process in the order in which they appear in the main text entry box in the Process Editor (see Figure 13.1).

13.2 Pipes

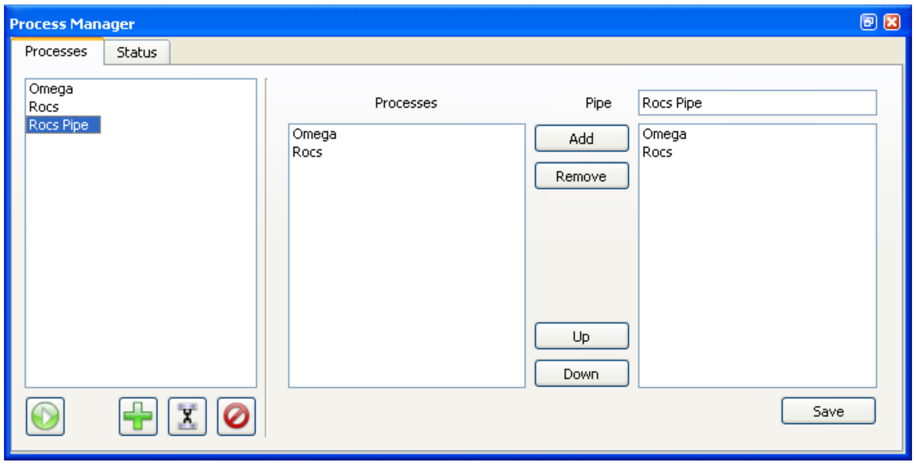


Figure 13.6: Pipe Editor

Once a number of processes have been created, they can be “piped” together using the Pipe Editor (see Figure 13.6). On the left-hand side of the editor is a list of all the currently defined processes. On the right is an ordered list of processes to be run and in which order to run (top to bottom).

When chaining processes together into a pipe, it is important that they have default input and output fields specified (see Sections 13.1.1 and 13.1.2). The default output of a given process is passed into the default input of the following process. When the final process is completed, the user will be notified and the final output file will be handled based on its specification in the associated process.

CHAPTER

FOURTEEN

Scripting

AFITT is in part built on top of the Python scripting language, which is an interactive, interpreted, strongly and dynamically typed language. More information about Python can be obtained from the Python website located at *http://www.python.org/*. The website contains many useful tutorials and other related information.

14.1 Journal File

All user interaction with the application goes through a Python layer even if the user is not aware of it. For instance, every button in the application is actually associated with a Python command that is sent to the internal interpreter when that button is pressed. The same integration is true of all the menu items and other GUI controls in the application. In fact, the journal file written by the application to save its history is itself a Python script which can be run to regenerate the state of the application.

The journal file (`journal.py`) is written to a user-specific local directory on the computer currently running the application. On Microsoft Windows 2000 and XP, the journal can be found in `C:\Documents and Settings\USERNAME\Application Data\OpenEye\AFITT\1.3`. On Microsoft Windows Vista, the journal can be found in `C:\Documents and Settings\USERNAME\AppData\Local\OpenEye\AFITT\1.3`. On all other platforms, the journal can be found in `~USERNAME/.OpenEye/AFITT/1.3`. If the application were to unexpectedly exit, the journal file corresponding to that run will be saved with a separate file containing a time stamp of when the application exited. These journal files are particulary useful for debugging unexpected application behavior and are of extreme benefit to the application developers when tracking down user-reported bugs. The inclusion of journal files (if possible) is greatly encouraged when submitting bug reports.

14.2 Startup File

Sometimes it is desireable to add new functionality or override certain default settings. AFITT provides functionality for users to place their own startup files (writtin in python) for these purposes.

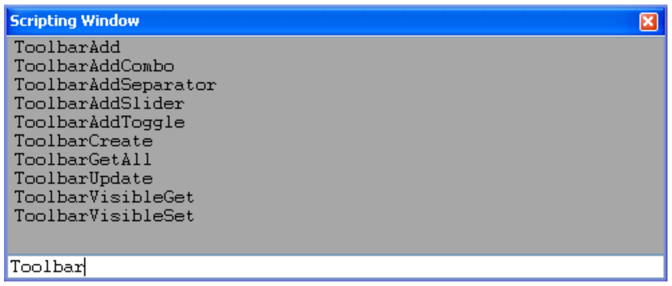
The startup file (`startup.py`) should be placed into a user-specific local directory on the computer currently running the application. On Microsoft Windows 2000 and XP, the startup file should be placed in `C:\Documents and Settings\USERNAME\Application Data\OpenEye\AFITT\1.3`. On Microsoft Windows Vista, the startup file should be placed in `C:\Documents and Settings\USERNAME\AppData\Local\OpenEye\AFITT\1.3`. On all other platforms, the startup file should be in `~USERNAME/.OpenEye/AFITT/1.3`.

14.3 Scripting Manual

The complete details of the Python interface is fully described in a seperate API manual which documents every possible command that can be run within the application.

14.4 Scripting Window

As mentioned above Python is an interactive language and as such a scripting window is provided as an interface to Python. The scripting window can be seen in Figure 14.1.



```
ToolbarAdd
ToolbarAddCombo
ToolbarAddSeparator
ToolbarAddSlider
ToolbarAddToggle
ToolbarCreate
ToolbarGetAll
ToolbarUpdate
ToolbarVisibleGet
ToolbarVisibleSet

Toolbar
```

Figure 14.1: Scripting window with tab completion of Python commands

The scripting window consists of two components – input and output. The large top component of the window is the output area which displays all of the output from commands entered in the bottom input area. It also displays any warnings or errors generated during the run of the application. The input component is the single line text entry area at the bottom of the window. On the input line, the user can directly type in Python commands to be run by AFITT.

The scripting input line is intelligent. Hitting the *Tab* key at the beginning of a line will indent the entry line to the previous line's identation if there was any indentation at all. If there was no indentation, the line will indent five spaces. However, hitting the *Tab* key after typing a portion of a command will cause the scripting line to expand out that command as far as possible without introducing ambiguity. For example, hitting the *Tab* key after typing "Tool" will expand out to "Toolbar." At this point, hitting the

Tab key again will result in a listing (in the output display above) of all the commands that begin with "Toolbar" (see Figure 14.1). Finally, hitting the *Tab* key when a command is fully expanded will display the API documentation associated with that command in the output display.

14.5 OEChemLite

`oechemlite` is a Python module that ships with AFITT which enables access to the currently loaded molecules via the scripting interface. `oechemlite` is distinguished from OpenEye's PyOEChem module in that it `oechemlite` provides only the PyOEChem functions that are not capable of making modifications to the molecules. Future versions of AFITT will enable the import of the full PyOEChem module which will allow modifications, provided that a separate license for the OEChem Python wrappers has been obtained.

Using `oechemlite`, a molecule is accessed by calling the `ScriptableObjectGet` function which takes an ID as a parameter and returns the associated molecule. For more details on the associated API please see the PyOEChem documentation; however, please remember that only the "const" functions (those that do not make changes to their objects) are available in `oechemlite`.

As most scripting routines in AFITT take unique keys as parameters, a `GetKey` function has been added to every object which returns the associated object's unique key identifier. In addition, the following methods have also been added to all `oechemlite` objects for convenience:

1. IsActive()
2. IsMarked()
3. IsLocked()
4. IsSelected()
5. IsVisible()
6. SetLocked(value = True)
7. SetMarked(value = True)
8. SetSelected(value = True)
9. SetVisible(value = True)

Two example scripts which use `oechemlite` appear below:

```
def SelectCarbonAtoms(idOrKey):
  """Select all carbon atoms from the given id or key
  if an id is used, only carbons from the first conformer are selected"""
  PushIgnoreHint()
  try:
    mol = ScriptableObjectGet(idOrKey)
    mol.SetVisible()

    for atom in mol.GetAtoms():
      if atom.IsCarbon():
        atom.SetSelected()

  finally:
    # the try..finally will always call
    # the popignorehint even if there is an error
    # otherwise, the application might not work correctly
    PopIgnoreHint()

def CreateSphereMonitorAtomsByDistance(idOrKey, x, y, z, distance):
  """idOrKey, x, y, z, distance -> generate sphere monitors for all atoms
  belonging to the specified id or key with distance from coordinate x,y,z
  When using an ID only atoms from the first conformer will be examined"""
  distanceSquared = distance * distance

  PushIgnoreHint(True)
  try:
    mol = ScriptableObjectGet(idOrKey)
    mol.SetVisible()

    color = OEColor(255, 255, 0, 100)
    for atom in mol.GetAtoms():
      x1,y1,z1 = mol.GetCoords(atom)
      dx = x-x1
      dy = y-y1
      dz = z-z1
      if dx*dx + dy*dy + dz*dz < distanceSquared:
        CreateSphereMonitor(atom.GetKey(), "Within Distance", color)

  finally:
    # the try..finally will always call
    # the popignorehint even if there is an error
    # otherwise, the application might not work correctly
    PopIgnoreHint()
```

Please note that it is not necessary to `import oechemlite` to access this functionality, that is done automatically at startup.

CHAPTER

FIFTEEN

Query language

15.1 Introduction

AFITT has several built-in scripting commands, such as `Visible`, `Select`, `Lock`, `Mark` and `Subset` which take a string argument (see also Section 15.6), where that string is written in a "query language". This query language is somewhat similar to the command language of the molecular graphics programs GRASP and SPOCK. The AFITT query language provides a powerful query syntax where atoms, bonds, conformers and molecules may be queried to see if they match certain properties. Triangles, vertices and surfaces may be similarly queried.

A fairly simple example is the following command:

```
Select("ch='A' && rn=10")
```

This command will select all atoms in chain A and in residue 10 for every molecule currently in memory.

A more complex query is the following:

```
Select("id=5 && (r=$hydrophobic || rn=(10,50))")
```

which selects every hydrophobic residue or residues with numbers between 10 and 50, but only matches atoms in the molecule with ID 5.

As the examples indicate, the general syntax of the query language based on *expressions*. Each expression consists of a *property*, a mathematical *operator* and a *value*. Expressions may be combined using a syntax similar to the C and/or Python programming languages. Boolean operations are supported via the *and* operator `&&`, as well as the *or* operator `||`. Nesting of parts of the query is possible via parentheses `(` and `)`. The operators can also be spelled out as in Python: `and` and `or` are valid for boolean operators.

15.2 Operators

AFITT's query language supports a full range of mathematical operators for testing a property's value against the value(s) specified in the query string. Supported are $=$, $!=$, $>$, $<$, $>=$, and $<=$.

15.3 Lists and Ranges

In addition to being able to specify a single value for a property, it is also possible to specify a *list* of values, or a *range* of values. Lists and ranges are provided as a convenience, since they allow queries matching groups of atoms in a compact manner.

- *LIST* A list is a series of values enclosed in square brackets, for example: `rn=[4,12,38]`. This is exactly equivalent to `(rn=4 || rn=12 || rn=38)`; each listed residue is selected. It is also possible to negate a list: `rn!=[4,12,38]` selects all residue *except* 4, 12, and 38, and is therefore exactly equivalent to `(rn!=4 && rn!=12 && rn!=38)`.
- *RANGE* A range is a pair of values enclosed in parentheses, e.g. `rn=(1,10)`. All residues between 1 and 10, inclusive are selected, so this command is exactly equivalent to `(rn>=1 && rn<=10)`. Negated ranges are also possible: `rn!=(1,10)` is equivalent to `(rn<1 || rn>10)`, and therefore selects atoms *not* in the given range.

Lists and ranges are only valid with the $=$ operator and the $! =$ operator. It is not legal (and nonsensical) to try to evalate an expression such as `rn>(1,10)`.

15.4 Properties

The following properties are defined and may be used for selection:

- *id* - This unsigned integer property limits the match for the part of the query it appears in to match only object with the given ID as shown in the list window. Example: `id=3`
- *a* - Atom name. This string property matches the atom name. Example: `a=' CA '`. Also may use the pre-defined sets `a=$sch` to match side-chain atoms and `a=$backbone` or `a=$ba` to match backbone atoms.
- *an* - Atom number. This integer property matches the atom number. Example: `an=5`.
- *r* - Residue name. This string property matches the residue name. Example: `r='ALA'`. See also the pre-defined sets at section 15.5.
- *rn* - Residue number. This integer property matches the residue number. Example: `rn=(1,50)`.
- *ch* - Chain. This string property matches the chain. The chain should be a single letter. Example: `ch='A'`.
- *model* - Model number. This integer property matches atoms with the given PDB-style model number (i.e. for different NMR models). Example: `model=2`.
- *altloc* - Alternate Location indicator. This string property matches atoms with the given PDB-style AlternateLocation property. The string should be a single letter. Example: `altloc='A'`.

- *icode* - Insertion code. This string property matches atoms with the the given PDB-style insertion code. Example: `icode='B'`.
- *occ* - Occupancy. This floating point property matches atoms with the the given occupancy property. Example: `occ>=0.5`.
- *b* - B-factor. This floating point property matches atoms with the the given crystallographic temperature (B) factor. Example: `b>=50.0`.
- *q* - Partial charge. This floating point property matches atoms with the the given partial charge. Example: `q>0`.
- *Q* - Formal charge. This floating point property matches atoms with the the given formal charge. Example: `Q>0`.
- *radius* - Radius. This floating point property matches atoms with the the given radius. Example: `radius<1.4`.
- *ac* - Atom color. This unsigned integer property matches atoms with the the given atom color. The color should be specified as a packed integer, e.g. from OEggColor().GetPackedColor(). Example: `ac=Pink.GetPackedColor()`.
- *elem* - Element number. This unsigned integer property matches atoms with the given element number. Example: `elem=6`.
- *weight* - Molecular weight. This floating point property matches molecules with the given molecular weight. Example: `weight>10000`.
- *IsAminoAcid* - This boolean property matches atoms whose residue names is the same as an amino acid residue recognized by OEChem. Example: `IsAminoAcid=1`. Equivalent to `r=$aa`.
- *IsNucleicAcid* - This boolean property matches atoms whose residue names is the same as a nucleic acid recognized by OEChem. A, C, G, T, U. Example: `IsNucleicAcid=1`. Equivalent to `r=$dna`.
- *IsWater* - This boolean property matches atoms which are in a water molecule, as recognized by OEChem. Example: `IsWater=1`. Equivalent to `r=$wat`.
- *IsSubstrate* - This boolean property matches atoms which are not protein, nucleic acid or water, as determined by the above queries. `IsSubstrate=1`. Equivalent to `(IsAminoAcid=0 && IsNucleicAcid=0 && IsWater=0)`.
- *type* - This string property may be one of “mol”, “atom”, “bond”, “tri”, “vert” or “surf”. This limits the matches for the part of the query it appears in to match either molecules, atoms, bonds, triangles, vertices or surfaces, respectively. Example: `type='atom'`
- *index* - This unsigned integer property limits the matches for the part of the query it appears in to items with the given index. Each atom, bond, triangle, etc., has an index assigned when the item is created. Since the indices are generally not exposed to the user, this command is probably of limited utility without OEChem-level access to the molecules in memory. Example: `id=2 && type='atom' && index=10`

- *query* - This string property performs a substructure search where the argument is treated as a SMARTS pattern, and so limits the query to atoms which match the SMARTS pattern. Example: `query='cccn'`
- *subset* - This string property matches all atoms which are in the previously-defined subset with the given name. Subsets may be defined via the `Subset` command, so the `subset` query provides a way of creating shorthand references to other complex queries. Example, first define a subset: `Subset('mysubset','rn=5 && id=2')`. Then the subset may be used as: `subset='mysubset'`.
- *key* - This unsigned integer property limits the match for the part of the query it appears in to match only objects with the given key. Example: `key=100000001`.
- *pkey* - This unsigned integer property limits the match for the part of the query it appears in to match only objects whose parent has the given key. Example: `pkey=100000001`.

15.5 Macros/Pre-defined sets

In addition to explicitly naming residues and atoms, AFITT defines several macros which may be used in query strings. These macros are prefixed with a dollar sign ($). Macros are generally used with the “residue name” (i.e. `r=`) property, although a few are used with the “atom name” (i.e. `a=`) property. The definitions for these sets are largely borrowed from RasMol.

- `r=$aliphatic`: ALA, GLY, LEU, VAL, ILE, PRO
- `r=$hydroxyl`: SER, THR, TYR
- `r=$sulfur`: CYS, MET
- `r=$aromatic`: TYR, HIS, TRP, PHE
- `r=$charged`: ASP, GLU, ARG, LYS
- `r=$amide`: GLN, ASN
- `r=$hydrophobic`: ALA, GLY, LEU, VAL, ILE, PRO, MET, PHE, TRP
- `r=$polar`: SER, THR, CYS, TYR, HIS, ASP, GLU, ASN, GLN, ARG, LYS
- `r=$neutral`: ALA, GLY, LEU, VAL, ILE, PRO, SER, THR, CYS, MET, PHE, TYR, TRP, ASN, GLN
- `r=$acidic`: ASP, GLU
- `r=$basic`: ARG, LYS
- `r=$small`: ALA, GLY, SER

- `r=$medium`: VAL, PRO, THR, CYS, ASP, ASN
- `r=$large`: ILE, MET, PHE, TYR, HIS, TRP, GLU, GLN, ARG, LYS
- `r=$cyclic`: HIS, PRO, TYR, TRP, PHE
- `r=$dna`: ADE, A, GUA, G, CYT, C, THY, T, URA, U
- `r=$aa`: all amino acids
- `r=$at`: ADE, A, THY, T, URA, U
- `r=$cg`: CYT, C, GUA, G
- `r=$purine`: ADE, A, GUA, G
- `r=$pyrimidine`: CYT, C, THY, T, URA, U
- `r=$wat`: WAT, H2O, HOH, TIP, SOL
- `r=$substrate`: this is defined as `r!=$wat && r!=$aa && r!=$dna`.

There are two atom name macros as well, `a=$backbone` (which may be abbreviated as `a=$ba`), which matches specifically protein backbone atoms, and a sidechain macro, `a=$sch`, which matches protein sidechain atoms. Both of these can be negated (`a!=$backbone` or `a!=$sch`).

15.6 Scripting with ScratchScope

In addition to the built-in commands `Visible`, `Select`, `Lock`, `Mark` and `Subset`, it's quite straightforward to use a selection string with any command which operates on a *scope* (see 3.4), by binding the selection string to the *ScratchScope*.

For example the following function defines an atom coloring command "ac":

```
def ac(color,str):
   Subset("scratch",str)  # use string to make named subsett
   ScratchSet("scratch")  # bind it to scratch scope
   AtomColorSetScoped(OEggColor(color),ScratchScope) # use it
```

With two small helper functions, these types of functions can be made even easier to create:

```
def ScratchSubset(str):
    Subset("scratch",str)
    ScratchSet("scratch")
    return ScratchScope

def ScratchSubsetList(str):
    Subset("scratch",str)
    ScratchSet("scratch")
    return ScratchList()
```

With these functions defined, the `ac` function above and functions similar to it can be easily defined. For example:

```
#set atom color
def ac(color,str):
    AtomColorSetScoped(color,ScratchSubset(str))

#set label color
def lc(color,str):
    LabelColorSetScoped(color,ScratchSubset(str))

#set screen center
def center(str):
    ViewerCenterSetScoped(ScratchSubset(str))

#show objects
def show(str):
    OEPySetVisible(ScratchSubsetList(str),True)

#hide objects
def hide(str):
    OEPySetVisible(ScratchSubsetList(str),False)
```

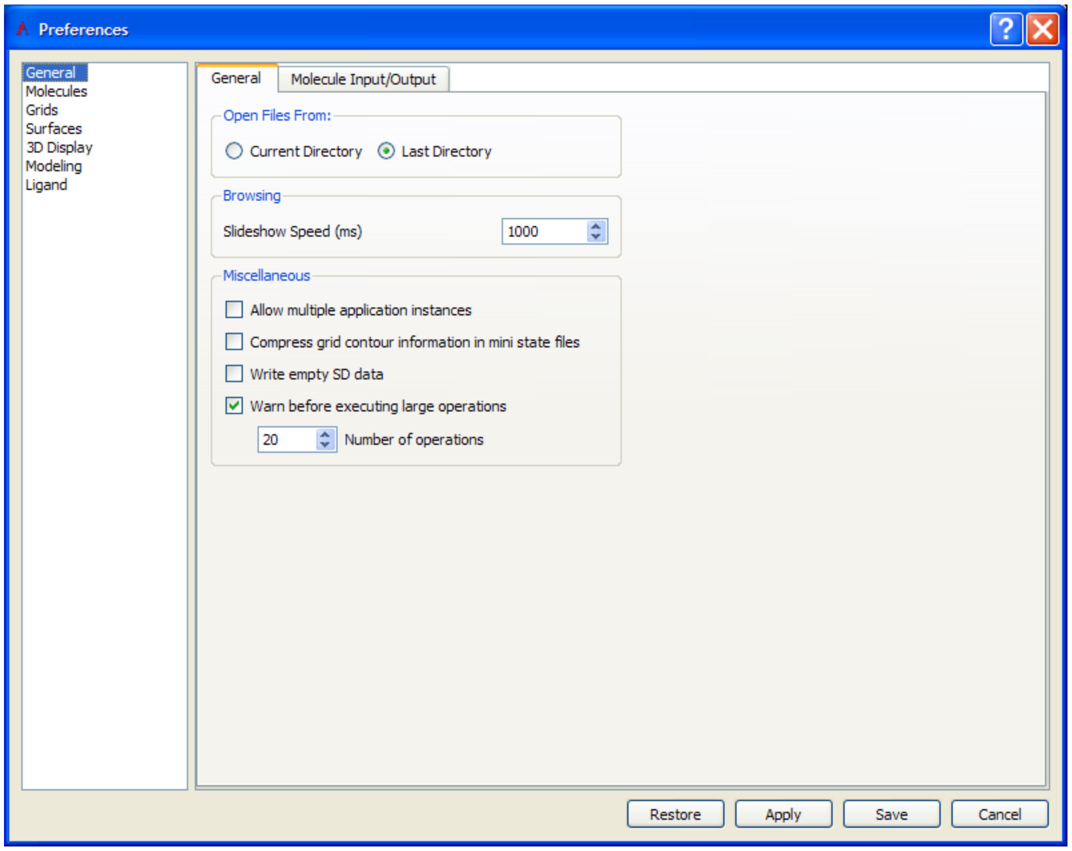
In this manner, any of the AFITT scripting commands which take a scope argument can be easily expanded to take a selection string.

CHAPTER

SIXTEEN

Preferences

Every user has his or her own individual preferences with regards to how molecules, grids, and surfaces should look and how applications should behave. For this reason, a Preferences dialog is available which allows customization of the application to the user's preference. An snapshot of the Preferences dialog can be seen in Figure 16.1.


Preferences

General
Molecules
Grids
Surfaces
3D Display
Modeling
Ligand

General | Molecule Input/Output

Open Files From: Current Directory [✓ Last Directory]

Browsing
Slideshow Speed (ms): 1000

Miscellaneous
- ☐ Allow multiple application instances
- ☐ Compress grid contour information in mini state files
- ☐ Write empty SD data
- ☑ Warn before executing large operations
- 20 Number of operations

Figure 16.1: Preferences

On the left-hand side of the dialog is column containing preference categories. These categories include: *General*, *Molecules*, *Grids*, *Surfaces*, *3D Display*, *Ligand*, and *Modeling*. Clicking on any of these categories will update the right-hand side of the widget to display the options corresponding to the selected category.

There are four buttons at the bottom of the dialog. Clicking on the *Restore* button will restore the current preferences to the default ones. Clicking on the *Apply* button will apply the current preferences to this run only and will not be automatically saved when AFITT is exited. Clicking on the *Save* button will save the current preferences for this run and for future runs of AFITT. Clicking on the *Cancel* button will close this dialog and will not apply any of the changed preferences.

Preferences are stored in a binary file (`preferences.oeb`) in a user-specific local directory on the computer currently running the application. On Microsoft Windows 2000 and XP, the preferences file can be found in `C:\Documents and Settings\USERNAME\Application Data\OpenEye\AFITT\1.3`. On Microsoft Windows Vista, the preferences can be found in `C:\Documents and Settings\USERNAME\AppData\Local\OpenEye\AFITT\1.3`. On all other platforms, the preferences can be found in `~USERNAME/.OpenEye/AFITT/1.3`.

While the preference file shares the same file extension as the OpenEye's binary database file, it cannot be read into AFITT using the "Open" menu item in the *File* menu. The preferences file is loaded automatically when the applications starts and is saved back to disk when the application exits. Deleting this file is equivalent to clicking on the *Restore* button in the dialog box.

There is also a file in this same directory which called AFITT.ini which contains machine specific settings like the list of recents files, prefered layouts, and hardware stereo options. Deleting this file will restore these settings to the defaults as well.

This directory can be opened from within AFITT by selecting the "Open User Directory" menu item in the *Help* menu.

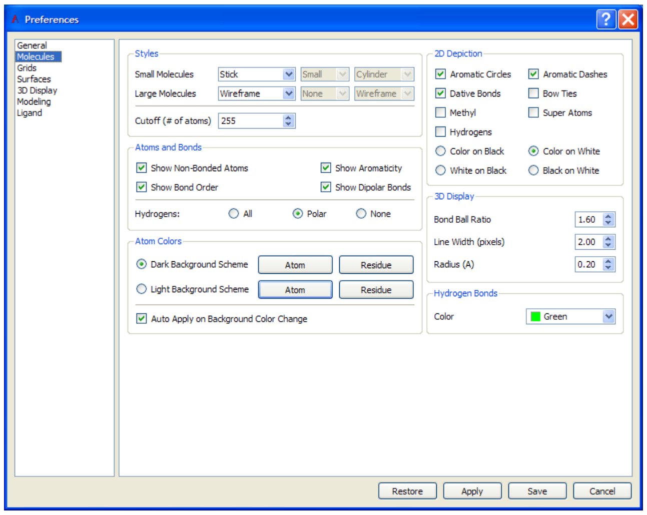
16.1 Useful Preferences

A brief list of useful preferences and their locations is given. These preferences are particularly useful for customizing AFITT for ligand fitting.

16.1.1 Display Hydrogens Preference

By default, only polar hydrogens are shown. While changing the hydrogen display in the style control is possible, this only affects the hydrogens in scope.

By changing the default hydrogen prefence from *Polar* to *All* all explicitly defined hydrogens will be shown when loading new files. This preferences is available in the *Molecules* section as seen in figure 16.2



Small Molecules: Stick, Small, Cylinder
Large Molecules: Wireframe, None, Wireframe
Cutoff (# of atoms): 255

✓ Show Non-Bonded Atoms ✓ Show Aromaticity
✓ Show Bond Order ✓ Show Dipolar Bonds
Hydrogens: All [✓ Polar] None

[✓ Dark Background Scheme] Light Background Scheme
✓ Auto Apply on Background Color Change

✓ Aromatic Circles ✓ Aromatic Dashes
✓ Dative Bonds ☐ Bow Ties
☐ Methyl ☐ Super Atoms
☐ Hydrogens
Color on Black [✓ Color on White]
White on Black Black on White

Bond Ball Ratio: 1.60
Line Width (pixels): 2.00
Radius (A): 0.20

Color: 🟩 Green

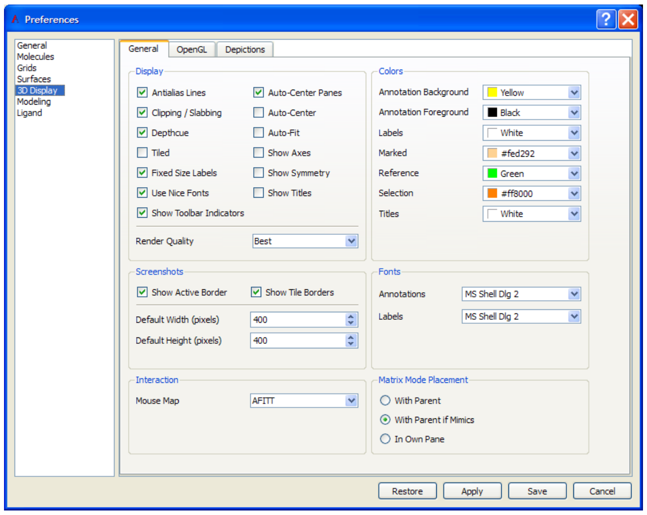
Figure 16.2: Molecule Display Preferences

16.1.2 Changing Mouse Map

The default mouse map is shown in Chapter 20. Additional mouse maps are available from popular crystallography or visualization programs. The default mouse map can be changed in the *Molecules* preference.

The available mousemaps are

- AFITT
- COOT
- Insight
- Maestro
- MOE
- O
- Quanta
- PyMol
- RasMol
- Sybyl
- VIDA



Display

- ✓ Antialias Lines
- ✓ Auto-Center Panes
- ✓ Clipping / Slabbing
- ☐ Auto-Center
- ✓ Depthcue
- ☐ Auto-Fit
- ☐ Tiled
- ☐ Show Axes
- ✓ Fixed Size Labels
- ☐ Show Symmetry
- ✓ Use Nice Fonts
- ☐ Show Titles
- ✓ Show Toolbar Indicators

Render Quality: Best

Colors

Annotation Background	🟨 Yellow
Annotation Foreground	⬛ Black
Labels	⬜ White
Marked	🟧 #fed292
Reference	🟩 Green
Selection	🟧 #ff8000
Titles	⬜ White

Screenshots

- ✓ Show Active Border
- ✓ Show Tile Borders

Default Width (pixels): 400
Default Height (pixels): 400

Fonts

Annotations: MS Shell Dlg 2
Labels: MS Shell Dlg 2

Interaction

Mouse Map: AFITT

Matrix Mode Placement

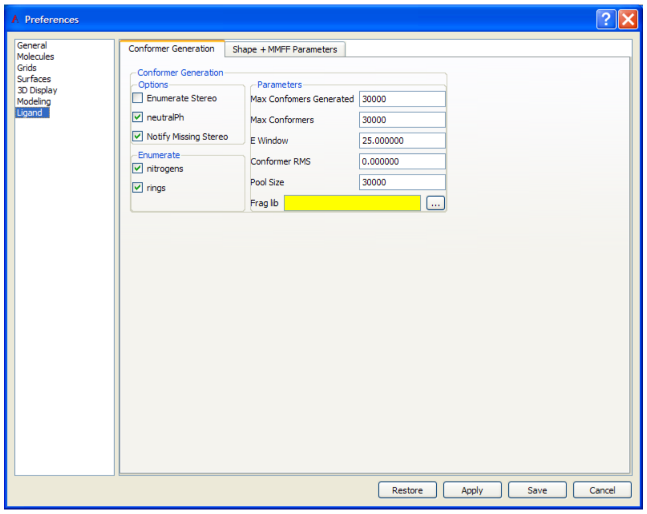
With Parent [✓ With Parent if Mimics] In Own Pane

Figure 16.3: Preferences for 3D Display

16.1.3 Stereo Enumeartion

By default, AFITT only enumerates stereo for unlabeled stereo centers. To force AFITT to enumerate all stereo centers whether they are labelled or not, check the *Enumerate Stereo* option in the *Ligand* preference section as seen in figure 16.4.

Additionally, AFITT displays a text box to notify when stereo centers are missing. To disable this notification, uncheck the *Notify Missing Stereo* option.



Conformer Generation

Options
- ☐ Enumerate Stereo
- ✓ neutralPh
- ✓ Notify Missing Stereo

Enumerate
- ✓ nitrogens
- ✓ rings

Parameters

Max Confomers Generated	30000
Max Conformers	30000
E Window	25.000000
Conformer RMS	0.000000
Pool Size	30000
Frag lib	

Figure 16.4: Preferences for Ligand Fitting

CHAPTER

SEVENTEEN

Example Data

AFITT ships with an example data directory for the user to experiment with in the event that other input files are not immediately available. Included in this directory are:

AFITT includes all of the protein-ligand complexes in the GOLD data set that include structure factors. These data sets are split into different directories, each of which has the following files:

- protein.mtz - the protein only map generated through refmac5.
- protein.pdb = the protein only portion of the coordinates.
- ligand.pdb - the ligand only portion of the coordinates.

These files are suitable both for experimentation and for testing with the supplied automation scripts.

CHAPTER

EIGHTEEN

Example Scripts

AFITT 1.3 ships with an example scripts directory as an example of AFITT's scripting capability. Included in this directory are:

- `pdbopen.py` - This script creates a new menu option in the *File* menu called "Open From PDB" which allows AFITT to open molecule files directly from the online Protein Data Bank. Simply enter the PDB id and if AFITT is able to locate that id in the database, it will download the file and load it into AFITT.
- `cmds.py` - This script defines some commands which use the AFITT query language to do common tasks like centering and coloring atoms or labels.
- `dbsmiles.py` - This script demonstrates how to connect AFITT to a database such as a corporate registry system.
- `refmac_params.py` - This contains the original refmac parameters. The variables in this file can be used as templates to change the default refmac parameters.
- `afitt_automate.py` - This script demonstrates how to automatically fit a ligand to density. The overall process is to read a `.mtz` file, create a map from it, find blobs of unfilled density, generate conformers for a specified ligand to go into that density, fit the conformers to the density. There are many opportunities for customizing this script, as documented in the file.
- do_afitt_gold.sh (.bat) - This is an example shell script that shows the command line that can be used to fit the ligands in the supplied gold data set. The *.bat* version is the appropriate version for *Windows. This script is located in the supplied data directory.*

CHAPTER

NINETEEN

Automation

AFITT was not desigened to be an automation system, but it can serve this purpose. *OpenEye* suggests that, for ligand fitting, the *flynn* command line tools be used instead of *AFITT*. *AFITT's* automation tools are slowly being deprecated in favor of command line automation.

That being said, Automation in afitt can be performed in several ways.

1. Scripting - an example python test script is given in the examples directory. This is a very powerful script that can be used to customize automation.
2. Built-in automation - AFITT has a method of performing automation from the command line as described below. The major downside of this automation is that it is not entirely command line based. The *AFITT* window appears along with the various graphics window. While it is nice to see the process unfold, this does take consume the computer's resources and is quite a bit slower than using the command line tools.

In addition to scripting, AFITT includes powerful automation from the command line. This automation includes the ability to search arbitrary directories on disk and find ligands, proteins and grids and automatically attempt to fit to them. The system can also wait for new data to appear.

The automation is enabled from the command line by sending the "-e automate" flag to afitt. Every other command line parameter is then passed to the automation system.

19.1 Getting Started With Automation

The 1l15 protein and ligand shipped in the DATA directory will be used as an example. That assumption here is that AFITT is being run from the installation directory. The examples given are intended to gently walk you through the automation system and a full list of options are included at the end of this chapter.

By default a filenamed "results.sdf" is output in the working directory. Note that the working directory may change during an automation run, see Section 19.2.1.

If a file with the same name already exists in the working directory, a new one will be created with a number appeneded. For example, if "result.sdf" already exists, a new file "results_1.sdf" will be created.

To stop an automation run, simply click on the “Stop” icon by the progressbar. Note that it may take a minute or two to stop a run.

19.1.1 Fitting a Single Ligand

For example, to fit the ligand *1ll5* to density:

```
bin/afitt -e automate -grid DATA/1ll5.grd -ligand DATA/1ll5.lig.pdb.gz
   -protein DATA/1ll5.nolig.pdb.gz
```

Notice that after the fitting is done, AFITT appears to hang. This is because AFITT is waiting for new data to appear. For this example, it never will, so hit the stop bottom (in the lower left of the screen) to abort the script.

19.1.2 Exiting AFITT When Finished

To exit AFITT when finished processing all the current data, simply add the “-exit” flag to the command line.

```
bin/afitt -e automate -grid DATA/1ll5.grd -ligand DATA/1ll5.lig.pdb.gz
-protein DATA/1ll5.nolig.pdb.gz -exit
```

19.1.3 Writing a Log File

Since AFITT is mainly a GUI application, the logs are not written to the terminal. To keep a log file, use the “-logfile” option.

```
bin/afitt -e automate -grid DATA/1ll5.grd -ligand DATA/1ll5.lig.pdb.gz
-protein DATA/1ll5.nolig.pdb.gz -exit -logfile my.log
```

19.2 Advanced Automation

The automation system can scan multiple directories when looking for data to fit. This is accomplised using the -dir option.

19.2.1 Using WildCards for Directories

Because most shells have “globbing”, that is they expand wildcards in filenames such as “*” AFITT also uses the “@” character to define wildcards on the command line. For example, using

```
-dir DATA/@
```

will tell afitt to search every directory in the subdirectory "DATA". The same wildcards can also be used to specify ligand, protein and grid targets. However, the grid, ligand and protein wildcards are only applied within a directory.

For example:

```
bin/afitt -e automate -dir DATA -grid @.grd -ligand @.lig.pdb.gz
-protein @.nolig.pdb.gz -exit
```

will search the DATA directory for any grid that matches "*.grd", any ligand that matches "*lig.pdb.gz" and any protein that matches "*.nolig.pdb.gz". The matching rules are as follows:

- These wildcards are assumed to be relative to the expanded directory. So if the directories that match the "-dir" option resolve to "a" and "b" the ligands would be "a/*.lig.pdb.gz" and "b/*.lig.pdb.gz"
- If multiple ligands, grids or proteins are found in a given directory, each will be fit in turn. That means that if 2 grids match the wildcard, 2 ligands match the wildcards and 2 proteins match the wildcards, 8 independent fits will be done.

19.2.2 Changing the output filename

By default, the automation system writes a file named "result.sdf" to the directory being analyzed.

The reason that ".sdf" is used and not ".pdb" is so that wildcards searches will not pick up the result files but rather the original ligand ".pdb" files.

To change the result name, use the option

```
-outname this_is_my_new_name
```

In this case, the new output name will become "this_is_my_new_name".

To change the extension, supply one of ".pdb", ".oeb", ".sdf" or ".oes".

Using an extension of ".oes" saves a state file with all of the data used during the fitting processes and all of the results.

To change the output extension, use the option

```
-outext .oes
```

This example will save state files as output.

If a file with the same name already exists in the directory, a new one will be created with a number appeneded. For example, if "result.sdf" already exists, a new file "results_1.sdf" will be created.

19.2.3 Testing the Data Without Fitting

For complicated searches, the data that will be used can be tested without actually performing the fit by using the “-testonly” option. This does everything except the actual fitting process and can be used to see if the command line is set up correctly. For example:

```
bin/afitt -e automate -dir DATA -grid @.grd -ligand @.lig.pdb.gz
-protein @.nolig.pdb.gz -exit -testonly
```

19.2.4 Adding Box Files

Box files can be added by using the “-boxes” switch. This switch is expanded the same way as the ligands are. Each box found identifies a unique region that blobs must intersect in order to be used for fitting. This is a convenient way of defining a receptor site and pruning out regions that should not be used to fit the ligand.

Remeber that box files are really molecules that are used to define the bounding box.

```
bin/afitt -e automate -dir DATA -grid @.grd -ligand @.lig.pdb.gz
-protein @.nolig.pdb.gz -exit -boxes @.lig.pdb.gz
```

If you want to add some padding to the boxes, use the “-boxpadding” flag. For example to add 4.0 *Å*ngstroms to the box:

```
bin/afitt -e automate -dir DATA -grid @.grd -ligand @.lig.pdb.gz
-protein @.nolig.pdb.gz -exit -boxes @.lig.pdb.gz -boxpadding
```

Finally, to fit to the density inside the region (this is useful when there is very poor density in which blobs can’t be found) use the “-boxonly” flag.

```
bin/afitt -e automate -dir DATA -grid @.grd -ligand @.lig.pdb.gz
-protein @.nolig.pdb.gz -exit -boxes @.lig.pdb.gz -boxpadding -boxonly
```

19.2.5 Specifying MTZ Columns and Map Type

Because *MTZ* columns can be arbitrarily labeled, AFITT’s automation system requires that column labels be specified on the command line. The following table gives the command line switches, a description of the column and some usual column names found in *MTZ* files.

A map type must be specified for constructing the density with which to fit.

For example, to load an *MTZ* file with Observed amplitude in column “F”, Calculated Amplitide in column “FC” and Calcualted Phase in column “PHIC” using maptype “3Fo2Fc”

```
bin/afitt -e automate -dir DATA -grid @.mtz -ligand @.lig.pdb.gz
-protein @.nolig.pdb.gz -exit -boxes @.lig.pdb.gz -boxpadding -boxonly
-FO F -FC FC -PHIC PHIC -maptype 3Fo-2Fc
```

Table 19.1: MTZ Column Switches

Switch	Column	Example Column Labels
-FO	Observed Amplitude	(F,FO,FP,FObs)
-PHI	Phase	(PHI, PHIB)
-FC	Calculated Amplitidue	(FC, FB)
-PHIC	Calculated Phase	(PHIC)
-SIGMA	Standard Deviation of Observed Amplitude	
-FOM	Figure of Merit	(FOM)
-RFREE	RFree	
-FWT	Amplitudes for weighted ’2Fo-Fc’ map	
-PHIWT	Phases for weighted ’2Fo-Fc’ map	
-FDELWT	Amplitidues for difference map	
-PHIDELWT	Phases for difference map	

Table 19.2: MTZ Column Switches

Map Types
Fo
Fc
Fo-Fc
2Fo-Fc
3Fo-2Fc
5Fo-3Fc
2Fo-FcSigmaA
Fo-FcSigmaA
FoFom
Fwt
Fdelwt

19.2.6 Filtering on protein distance

By default, blobs farther away than 4 $\mathring{A}$ngstroms from the protein are not selected for fitting. To change this behavior, use the "-distance" flag, or change the default value of the slider in the *Ligand Model* window. The following example sets a protein distance of 10.0$\mathring{A}$.

```
bin/afitt -e automate -dir DATA -grid @.grd -ligand @.lig.pdb.gz
-protein @.nolig.pdb.gz -exit -boxes @.lig.pdb.gz -distance 10.0
```

19.3 Platform Differences

19.3.1 Windows

To run automation from windows, AFITT must be started from the command line using the supplied ".bat" file. If AFITT is installed to the default location, afitt automation scripts will be launched as follows:

```
C:\OpenEye\AFITT\1.3.0\afitt.bat -e automate
```

19.3.2 OS X

Due to the way *OS X* bundles applications, to start automation from the command line, the bundle must be treated as a directory structure. If AFITT is installed into the *Applications* directory, the command line to start automation is as follows:

```
/Applications/afitt.cpp/Contents/MacOS/afitt -e automate
```

Sadly, before the automation begins, the environment details are also output.

CHAPTER

TWENTY

Mouse Maps

20.1 AFITT Mouse Map

Table 20.1: Afitt Mouse Map and Keyboard Shortcuts

Function	Mouse Buttons or Short Cut
Zoom	Wheel
	Y motion while holding *Middle* Button
	Y motion while holding *Left*+*Right* Buttons
Rotate	XY Motion while holding *Left* Button
Translate in XY plane	XY Motion while holding *Left* Button and Shift
Translate in Z plane	Wheel and Alt
Select	Click *Left* Button
Select Rectangle	XY Motion while holding *Right* Button
Select Add	Click *Left* Button while holding Shift
	Click *Left* Button while holding Control
Select Grow	Double Click *Left* Button
Select Add Grow	Double Click *Left* Button while holding Shift
	Double Click *Left* Button while holding Control
Select Rectangle and Add	XY Motion while holding *Right* Button and Shift
Change clipping plane	Wheel and Alt+Control+Shift
Change far clipping plane	Wheel and Alt+Control
Change near clipping plane	Wheel and Alt+Shift
Menu	Click *Right* Button
Adjust contour level	Wheel and Shift
Show Labels	XY Motion and Control

CHAPTER

TWENTYONE

Release Notes

21.1 Version 1.3.1

January 2008

Minor Bug Fixes:

1. Coot mouse mode is more complete.
2. Popup right-click menus now include the option to center in the 3D window and Spreadsheet (this was in the Afitt 1.2 series).
3. CDX (Chemdraw) reader properly assigns hydrogens and generates 2D coordinates when given a 2D sketch. This means that volume calculations are now correct for Chemdraw sketch files.
4. Molecules with only 2D coordinates are not shown in the 3D window anymore.
5. Clearing all data repeatedly from the File menu could result in the result spreadsheet columns being removed.

Major Bug Fixes:

1. Fixed crash bug when finding blobs with no protein.
2. Fixed residue perception when forming covalent bonds.
3. Stereo-enumeration now works from non-connection table molecules. Before it always used the input stereochemistry.

Improvements:

1. Refinement dictionaries now update PDB atom names for molecules that never had them, i.e. smiles strings.
2. Merging molecules now updates PDB atom names for molecules that never had them and attempts to add ligands to new chains, residue numbers are incremented as a last resort.

3. Stereo-enumeration now uses far less memory.
4. Blob finding now can find density at multiple contour levels.
5. Distance to protein slider added to ligand modeling widget. This filters out blobs that are too far from the given protein.
6. Distance to protein flag added to automation system.
7. Molecule splitter dialog now automatically sets up the first split seperating the protein from the ligands.
8. Maps generated from MTZ files are now of higher quality.
9. Added Edit Menu to change residue names.
10. Ligands are added to new chain ids when possible.
11. 3D Toolbar indicators behave more sanely on Linux.
12. Residue names can now be changed. Warning: these names should not be one of the ones listed in the "Writing Refinement Dictionaries" in chapter 7.8 as these may conflict with the standard refinement dictionaries shipped with *refmac* or *cns/x*.

21.2 Version 1.3.0

August 2007

Minor Bug Fixes:

1. Fixed a problem where on some graphics cards only aromatic rings would be drawn when a molecule was in stick mode

Major Bug Fixes:

1. Upgraded to OEChem 1.5 which fixed a number of bugs
2. Upgraded Omega Toolkit to 2.2.1

Improvements:

1. All modeling widgets have been overhauled.
2. Blob finding is heavily improved (in speed and accuracy)
3. Fitting information is now displayed in 3D window.
4. Automated protein clean-up is now available.

5. Modeling mode is now removed.
6. Internal conformation generation is now consistent with Omega 2.2.1
7. Added Coot mouse mode.
8. Anti-aliasing of text and 2D depictions
9. Much better control over placement of dock windows
10. Dock windows can be stacked on top of each other in tabs
11. Single toolbar at the top of the application
12. Additional popup toolbars placed around edge of main window
13. Enabled grid contour transparency
14. Hydrogen style is now a molecule specific property instead of a global property
15. C-Alpha trace display is now a molecule specific property instead of a global property
16. Ribbon display is now a molecule specific property instead of a global property
17. Many miscellaneous 3D rendering improvements and accelerations
18. Many miscellaneous user interface improvements

New Features:

1. Added dynamic help system
2. Added new 2D depiction color schemes: black on white, white on black, color on white and color on black
3. Added new molecule coloring schemes: amino, chain, group, cpk, cpknew, shapely, bfactor, formal charge, and partial charge
4. Added new surface hydrophobicity scales: charifson, eisenberg, kyte-dolittle, and white-octanol
5. Added support for reading attached grids and surfaces in OEB files
6. Added support for reading Spicoli geometric annotations
7. Added support for reading FRED receptor files
8. Added new hydrogen style: polar hydrogens only
9. Added multiple new atom and bond label types (including one for OEChem generic data)
10. Added new 3D display mouse maps for Coot and PyMol

21.3 Version 1.2.2

July 2006

Minor Bug Fixes:

1. Fixed "MolecularEnergy()" calculation method in spreadsheet to correctly report energies on multi-conformer molecules
2. Fixed "SpreadsheetFilter()" command such that it operates on the specified spreadsheet instead of the current spreadsheet
3. Ensured that all current environment variables are properly passed to processes started from the process manager
4. Fixed the icon on the hydrogen display button to accurately reflect what will occur when the button is pressed
5. Fixed a Mac OS X bug where double-clicking on a molecule file started VIDA but did not actually load the file that was clicked on

Major Bug Fixes:

1. Upgraded to OEChem 1.4.1 which fixed a number of bugs associated with handling bad input files
2. Fixed a cross-platform compatibility issue in state files

Improvements:

1. Automation - AFITT now has easy to use automation for fitting ligands to density. Please see Chapter 19 for full details.
2. Box Files - AFITT can now generate boxes to constrain the locations that ligands can be fit to density. A box can be generated from any molecule or selection of atoms. The bounding box of the molecule (plus a specified padding) is used to constrain the density in which to find blobs. Alternatively, the density inside the box can be used for fitting. This is located in the *Data* menu.
3. Mask Density To Blob - The Ligand Model Window has a new button. This button extracts the densities within each blob. These new densities can be saved to disk or used for other operations.
4. Write Refinement Dictionary - If you are writing a refinement dictionary from a non 3D molecule, say a smiles string, a low energy conformer will now be generated for the molecule.
5. Project List - During ligand fitting, all seconday objects like extracted grids and blobs are now put in the same list for easy access int the list window. A new list is created when "Find Blobs" Blobs" is clicked.
6. Blob List - Pressing the up and down arrows while inside the blob list will go to the previous or next blob respectively.

7. Retrieve Density now available on windows - The option to retrieve density from the EDS server (under the Data Menu) is now available.

8. Copy/Paste - Now work on focused/visible molecules when not in modeling mode. When inside modeling mode, they work on modeled items as they did before.

9. Mouse Modes - When mouse modes are being used, the current mouse mode is shown at the top of the 3D window.

10. Updated menu system to be compliant with standard UI design practices (such as '...' usage when menu option launches a dialog)

11. Updated menu system on Mac OS X to follow common Mac paradigms (e.g. "Open Recents" and "Preferences..." options)

12. Reorganized File menu for more logical placement of options

    (a) Collapsed "New > List >" submenu structure to a single "New List >" submenu which appears after the Open, Save block

    (b) Added "New Molecules >" submenu which contains a "From Split" submenu which provides the "Split Molecules" functionality from the old Edit menu. In VIDA 2.2 a "From Merge" will join this option

    (c) Renamed "Clear All" option to "Clear" and moved up in the menu above Save block

    (d) Removed "Abort Script" option as it was redundant (Stop button next to progress bar performs this function) and also could generally not be accessed while a script was running

13. Reorganized Edit menu for more logical placement of options as well as to provide some commonly performed operations in a single place

    (a) Moved "Split Molecules" functionality to File menu

    (b) Added new "Select" submenu which contains the following options: All Visible, Query, Invert, and None

    (c) Added new "Mark" submenu which contains the following options: All, Query, None

    (d) Added new "Hide" submenu which contains the following options: All, Selected, Marked, Not-Selected, Not-Marked, and None

    (e) Removed the following options as they were redundant given the above additions: Hide All, Clear Marked

14. Alphabetized list of available windows in the Window menu

15. Batch shell window no longer appears when running on Windows

16. Redesigned process manager window for easier use

17. Significantly improved the quality of POV-ray exports and reduced the output file size

18. Improved the rendering of electrostatic grids

New Features:

1. Added "Copy Image" function under the Edit menu which takes a screenshot of the current main window and places it on the clipboard
2. Added "Map Index" atom label corresponding to the value returned by the OEChem function OEAtomBase::GetMapIdx().

21.4 Version 1.2.1

March 2006

Minor Bug Fixes:

1. Fixed a problem where extra atom labels could appear on molecules when saved to state
2. Fixed a problem where CTRL-mouse over atom displays were not seen when using non-VIDA mouse maps
3. Fixed a bug in the "Text Scale" slider where its position did not always reflect the current scale being used
4. Fixed a bug where OE_DIR and OE_LICENSE environment variables were not being properly exported to the Process Manager on Mac OS X such that spawned applications could not find a valid oe_license.txt file
5. Fixed a bug where exporting a CSV file on Microsoft Windows attached an extra return character causing a blank line to appear between each row when read into Microsoft Excel
6. Fixed a few cases where the display of depictions in the spreadsheet was not obeying the set preferences
7. Fixed a bug where extra calls to undo cleared the redo stack
8. Turned off the writing of empty data fields when writing molecules to SD or OEB formats. This can be turned back on if truly desired from the Preferences
9. Fixed a number of preferences that were not being properly saved or obeyed by the application
10. Fixed the undo stack such the initial setup of the 3D display cannot be undone by accident
11. Fixed the display of angle monitor measurements from radians to degrees
12. Fixed the display of the text scale widget such that it is not hidden on low resolution monitors
13. Made sure that when saving images and state files, a proper file extension is added to the filename, even if one was not specified

Major Bug Fixes:

1. Fixed a problem with reading spreadsheet information in state files generated by VIDA 2.0.x which caused extra tabs to appear in the spreadsheet
2. Fixed a problem with writing spreadsheet information correctly in VIDA 2.1
3. Fixed a crash when reading a badly formatted state file (potentially generated by the bug fixed above)
4. Fixed a crash when attempting to import a non-standard whitespace delimited spreadsheet (whitespace is not a supported spreadsheet delimiter)
5. Fixed a crash on certain video cards when attempting to display “fixed size text” labels
6. Fixed a crash caused by dragging state files onto the application icon in order to open them
7. Fixed an application hang when reading badly formatted XPLOR maps
8. Fixed an application hang when reading a badly formatted OEB file
9. Fixed a memory leak generated when running a large number of scripting commands
10. Fixed a crash when trying to generate electrostatic contours on extremely small molecules (e.g. water)

Improvements:

1. Filled out Sybyl mouse map in more detail
2. Added “fixed size text” option for labels (see Preferences) such that the label font size remains fixed independent of zooming in the scene, however the font size can be adjusted using the “Text Scale” slider in the Style Bar window
3. Added the ability to specify the font for labels when using “nice fonts” for labels (see Preferences)
4. Added a “custom” option to the atom and bond label dialogs to allow for easier specification of custom labels without having to use scripting commands
5. Improved the drawing of labels such that they always appear in front of the object they are labeling
6. Improved the ability to see the selected state of partially selected residues in the sequence view
7. Added file dialog filter option to allow reading of ASCII formatted grids (*.agd)
8. Added support for native builds on Intel Macs
9. Added interactive text annotation of molecules
10. Added the ability to start, monitor, terminate, and load results from external processes
11. Added a protein sequence viewer as part of the 2D depiction widget
12. Added the `oechemlite` Python module, enabling significantly greater scripting functionality within VIDA

13. Added the ability to create arbitrary sphere monitors via the scripting API (see Python API for documentation of the new method `CreateSphereMonitor(...)`)
14. Added support for fullscreen display
15. Added new *cloud* visualization style for grids
16. Added support for copy and paste of molecules from ISIS/Draw
17. Added support for use of the Maestro mouse map in the 3D display
18. Added support for saving screenshots in JPG format in addition to PNG

21.5 Version 1.2.0

January 2006

Minor Bug Fixes:

1. Fixed a number of preferences that were not being properly saved or obeyed by the application
2. Fixed the undo stack such the initial setup of the 3D display cannot be undone by accident
3. Fixed the display of angle monitor measurements from radians to degrees
4. Fixed the display of the text scale widget such that it is not hidden on low resolution monitors
5. Made sure that when saving images and state files, a proper file extension is added to the filename, even if one was not specified

Major Bug Fixes:

1. Fixed a crash when trying to generate electrostatic contours on extremely small molecules (e.g. water)

Features:

1. Overhauled AFITT's interaction model to be sane.
2. Added interactive text annotation of molecules
3. Added the ability to start, monitor, terminate, and load results from external processes
4. Added a protein sequence viewer as part of the 2D depiction widget
5. Added the `oechemlite` Python module, enabling significantly greater scripting functionality within VIDA

6. Added the ability to create arbitrary sphere monitors via the scripting API (see Python API for documentation of the new method `CreateSphereMonitor(...)`)
7. Added support for fullscreen display
8. Added new *cloud* visualization style for grids
9. Added support for copy and paste of molecules from ISIS/Draw
10. Added support for use of the Maestro mouse map in the 3D display
11. Added support for saving screenshots in JPG format in addition to PNG

Improvements:

1. Cleaned up the appearance of the right-click popup menus
2. Added a “Style” option to the right-click popup menus, when appropriate to the context, which mirrors the main “Style” menu
3. Added significantly greater detail to info labels created when holding down the “ctrl” (or “alt” on Mac) key when moving the mouse in the 3D or 2D views
4. Added two default filtered spreadsheets to the application: “Molecules” and “Proteins”
5. Filled out the scripting API for greater control of the GUI - in particular for menus and toolbar
6. Greatly improved the electrostatic coloring scheme for protein surfaces
7. Added the ability to change the font used by the application

BIBLIOGRAPHY

[1] Menendéz-Velazquez, A., Garciá-Granda, S., “A procedure towards the automatic solution of crystal structures by means of topological analysis of Fourier maps”, *J. Appl. Cryst.* Vol 36, pp. 193-205

[2] Diller, D.J., Verlinde, C., “A critical evaluation of several global optimization algorithms for the purpose of molecular docking.” *J. Comp. Chem*, Vol. 20, 1740-1751, 1999

[3] Koch, M.H., “Automatic interpretation of electron-density maps for organic structures.” *Acta Cryst A*, Vol. 30, pp. 67-70, 1974

[4] Main, P., Hull, S., “The recognition of molecular fragments in E maps and electron density maps”, *Acta Cryst A*, Vol. 34, pp. 353-361, 1978

[5] Cascarano, G., Giacovazzo, C., Camalli, M., Spagna, R., Watkin, D., “Automatic solution and refinement of crystal structures by means of the package UNIQUE”, *Acta. Cryst. A*, Vol. 47, pp. 373-381

[6] Altomare, A., Giacovazzo, C., Ianigro, M., Moliterni, A., Rizzi, R., “Peak labelling in electron density maps from powder data: the use of crystal chemical information”, *J. Appl. Cryst.*, Vol. 35, pp. 21-27, 2002

[7] Zwart, P., Langer G., Lamzin V., “Modelling bound ligands in protein crystal structures”, *Acta Cryst D*, Vol 60, pp. 2230-2239, 2004

[8] Oldfield, T., “Creating structure features by data mining the PDB to use as molecular-replacement models”, *Acta Cryst. D*, Vol 57, pp. 696-705, 2001

[9] Perola E., Charifson P.S., “Conformational analysis of drug-like molecules bound to proteins: an extensive study of ligand reorganization upon binding”, *J. Med. Chem.*, Vol. 47, pp. 2499-510, 2004

[10] Wlodek, S., Skillman A.G., Nicholls A., “Automated ligand placement and refinement with a combined force field and shape potential”, *Acta Cryst. D*, 2005.

[11] Vagin A.A., Steiner R.A., Lebedev A.A., Potterton L., McNicholas S., Long F. and Murshudov G.N., “REFMAC5 dictionary: organization of prior chemical knowledge and guidelines for its use”, *Acta Cryst D.*, Vol. 60, pp. 2184-2195, 2004

[12] Murshudov G.N., Vagin A.A., Dodson E.J., "Refinement of Macromolecular structures by Maximum likelihood method". *Acta Cryst. D* Vol. 53, pp. 240-255, 1997

[13] Brnger A.T., Adams P.D., Clore G.M., DeLano W.L., Gros P., Grosse-Kunstleve R.W., Jiang J.-S., Kuszewski J., Nilges M., Pannu N.S., Read R.J., Rice L.M.,Simonson T. and Warren G.L., "CNS/CNX Crystallography & NMR System: A New Software Suite for Macromolecular Structure Determination" *Acta Cryst.* (1998). D54, 905-921

[14] Lovell, S.C., Word, J.M., Richardson J.S. and Richardson D.C., "The Penultimate Rotamer Library", *Proteins: Structure Function and Genetics*, Vol. 40, pp. 389-408, 2000

[15] McGann, M. R., Almond, H.R., Nicholls, A, Grant, J.A., Brown, F.K., "Gaussian Docking Functions", *Biopolymers*, Vol. 68, pp. 76-90, 2003

[16] Bower, M.J., Cohen, F.E., Dunbrack, R.L., Jr. "Prediction of protein side-chain rotamers from a backbone-dependent rotamer library: a new homology modeling tool", *J. Mol. Biol.*, Vol 267, pp. 1268-1282, 1997.

[17] Robert J. Fletterick, Raymond Matela, "Color-coded α-carbon models of proteins", *Biopolymers*, Vol. 21, pp. 999-1003, 1982.

[18] Matthew D. Eldridge, Christopher W. Murray, Timothy R. Auton, Gaia V. Paolini and Roger P. Mee, "Empirical scoring functions: I. The development of a fast empirical scoring function to estimate the binding affinity of ligands in receptor complexes", *Journal of Computer-Aided Molecular Design*, Vol. 11, pp. 425-445, 1997.

[19] Araz Jakalian, Bruce L. Bush, David B. Jack and Christopher I. Bayly, "Fast, Efficient Generation of High-Quality Atomic Charges. AM1-BCC Model: I: Method", *Journal of Computational Chemistry*, Vol. 21, pp. 132-146, 2000.

[20] Araz Jakalian, David B. Jack and Christopher I. Bayly, "Fast, Efficient Generation of High-Quality Atomic Charges. AM1-BCC Model: II: Parameterization and Validation", *Journal of Computational Chemistry*, Vol. 23, pp. 1623-1641, 2002.

[21] http://www.eyesopen.com/docs/