SLIDE — Screening for Ligands by Induced-fit Docking

User Guide

Version 2.31 – February 2005

Protein Structural Analysis and Design Laboratory
Department of Biochemistry and Molecular Biology
Michigan State University
slide@sol.bch.msu.edu

Contents

Chapter 1

Introduction

SLIDE, “Screening for Ligands by Induced-fit Docking”, is a tool that is designed to search large data sets of small molecule structures for potential ligands that are likely to bind to a given binding site of a target protein. The binding site is represented by surface residues, water molecules, and a set of favorable interaction points above the protein surface. Based on the choice of these template points, SLIDE is able to find potential ligands that mimic known ligand binding modes or to return a large variety of novel potential ligands.

SLIDE docks potential ligands into the binding site in order to evaluate shape and chemical complementarity between the protein and the ligand. During this docking process, induced flexibility of protein side chains is modeled by applying directed minimal rotations to rotatable protein side-chain or ligand bonds, which, in general, are all single bonds. SLIDE uses an approach based on mean-field theory to decide on the optimal, i.e. lowest-cost, conformational changes in both protein and ligand molecules that will generate a complementary interface. Both ligand and protein rotations are utilized.

Each protein-ligand complex is ranked based on the number of intermolecular hydrogen bonds and the hydrophobic complementarity of the contact interface. Ideally, SLIDE, reduces a database of some hundred thousands of compounds down to a ranked list of potential ligands, out of which the, say, top one-hundred compounds should be reasonable candidates for further inspection by either computational docking or binding assays.

This is the user manual for SLIDE, which provides an overview of the methodology and the use of SLIDE. It is not a software documentation, which describes the software, i.e. the data structures and modules used in the implementation of SLIDE. It should rather explain to the user the interface of SLIDE, including all auxiliary scripts and the directory structure of the file system that holds SLIDE’s input and output files.

Chapter 2

Methods

This chapter describes the methods that are used in SLIDE. A more detailed description can be found in the following references:

- V. Schnecke, C. A. Swanson, E. D. Getzoff, J. A. Tainer, and L. A. Kuhn (1998) "Screening a Peptidyl Database for Potential Ligands to Proteins Including Side-Chain Flexibility", *Proteins: Structure, Function, and Genetics* 33, 74–87.
- V. Schnecke and L. A. Kuhn (2000) "Virtual Screening with Solvation and Ligand-Induced Complementarity", *Perspectives in Drug Discovery and Design*, 20, 171–190.
- V. Schnecke and L. A. Kuhn (2002) "Modeling Induced Fit and Controlling Molecular Diversity During Database Screening for Ligands", *Proteins: Structure, Function, and Genetics*, in press.
- M.I. Zavodszky, P.C. Sanschagrin, R.S. Korde, and L.A. Kuhn (2002) Distilling the Essential Features of a Protein Surface for Improving Protein-Ligand Docking, Scoring, and Virtual Screening, *Journal of Computer Aided Molecular Design*, 16, 883-902.

The purpose of this chapter is to review the basic methods and to explain how user-modifiable variables (see Section 3.5) fit in.

2.1 Template Design

Based on the classification of Kuntz et al.[1], SLIDE is a descriptor-matching approach, i.e. it uses a binding site template that defines points for favorable interactions with protein surface atoms. During the search, interaction centers (H-bond donors/acceptors or hydrophobic interaction points) of the screened molecules are mapped onto these template points to provide a basis for docking the potential ligands. Such a template can be rather small and based on known ligand binding modes (Figure 2.1) or larger and automatically generated without any bias towards known ligands.

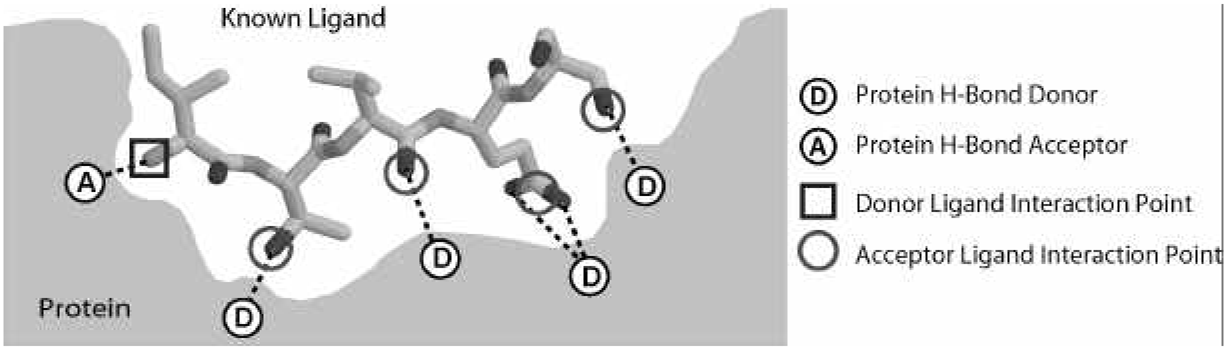

Known Ligand
A
D
D
D
D
D
Protein
D Protein H-Bond Donor
A Protein H-Bond Acceptor
□ Donor Ligand Interaction Point
○ Acceptor Ligand Interaction Point

Figure 2.1: An example of a template based on the binding mode of a known ligand for the target protein. Either the position of a single hydrogen-bond donor and acceptor in this ligand or the average positions of two hydrogen bond donors or acceptors in this ligand, if they form a hydrogen bond to the same protein atom, define a template point.

A template can include four different types of interaction points located in the binding site above the protein surface:

- **Hydrogen-bond donor.** During screening, SLIDE will place a hydrogen-bond donor of the ligand onto this point, which is within favorable hydrogen-bonding distance of a protein hydrogen-bond acceptor.

- **Hydrogen-bond acceptor.** Point within favorable hydrogen-bonding distance to a protein hydrogen-bond donor.

- **Hydrogen-bond donor/acceptor (doneptor).** This point is within hydrogen-bonding distance to both a hydrogen-bond acceptor and a donor of the protein, so either a ligand hydrogen-bond donor or acceptor can be placed here, or a hydroxyl group that can accept and donate at the same time.

- **Hydrophobic interaction center.** This point is placed above a hydrophobic surface patch of the protein, and SLIDE can match a hydrophobic interaction point of a potential ligand (see Section 2.3).

2.1.1 Template Based on Known Binding Modes

The program `ligand_based_template` (formerly `average_template`) can be used to generate a template based on known ligand binding modes. It reads a set of ligand structures provided in the same reference frame as the binding site of the protein and identifies H-bond donors and acceptors and positions of hydrophobic interaction points. Each of these points is a potential template point. Close points of the same type are clustered using complete linkage clustering to reduce the number of template points. If a template based on a single known ligand is being generated, a very low clustering cutoff, e.g. 0.0, should be used.

2.1.2 Unbiased Template with Knowledge-Based Hydrogen-Bonding Point Placement

This is the suggested method for unbiased template generation. The template can be automatically generated using the program `template`. In this case the template is only based on the ligand-free structure of the protein, which reduces bias towards known ligands. When using this method, the limits of the binding site must be defined (see Section 3.6.1 and 5.4). Hydrogen bonding points are assigned based on the protein chemistry in a directed manner. After identifying protein atoms in the binding site which are capable of forming hydrogen bonds, a number of template points, varying from four to 15 depending on the size of the binding site, are placed at and around positions where a ligand atom could form favorable hydrogen bonds with the protein[2, 3]. The points are then labeled as acceptor, donor, or doneptor, based on the protein atom type from which they were derived. Hydrogen bond template points placed closer than 1 Å to each other are clustered together and relabeled, i.e. if a mix of donors and acceptors are combined, the resulting cluster point is labeled as a doneptor. Points which are closer than 2.5 Å to a protein atom are discarded.

To assign hydrophobic template points, the binding site is filled with points that are 2.5 to 5.0 Å from a protein atom. A hydrophobic enhancement score is then calculated for each point as the number of hydrophobic atoms (carbon atoms bonded only to other carbon or hydrogen atoms) located 2.5 - 5.0 Å from the points minus the number of hydrophilic (other, non-carbon) atoms. If this score is greater than 3, the point is classified as a hydrophobic template point. The values for the number of neighboring atoms of each type were chosen after an examination of the neighbors of surface points in a series of structures. Hydrophobic points which are within half of the hydrophobic clustering cutoff from a hydrogen bonding point are eliminated at this stage. To ensure even sampling, points are initially located on a grid, with spacing generally 0.5 Å. After eliminating points either too close or too far from the protein and points not classified as hydrophobic, there are generally 5,000 to 10,000 points remaining. The hydrophobic template points are then clustered using complete linkage clustering to yield a feasible number of total, hydrogen bonding and hydrophobic, template points, generally 100 to 150. The user defined clustering threshold defines a maximal distance of points in a given cluster. Using a lower clustering threshold for hydrophobic points would yield a finer sampling of hydrophobic points, but more of them. The nearest neighbor distance for two adjacent hydrophobic template points will be approximately one-half the clustering threshold. The hydrogen bonding portion of the template can be created as dense, sparse, or minimal. When running the template generation program, SLIDE will not behave properly if atoms with multiple positions, i.e. alternate locations, are included. The user should remove all but one of the alternate locations for each atom as they see fit. A suggestion is to simply include the “A” location.

2.1.3 Template with Key Points

A third way to design a template for the binding site is a mixture of the two approaches above. In this approach the user can assign label specific template points in the set of all template points as key points. In this case, only triangles including at least one key template point will be indexed in the hash tables (Section 2.4), meaning that each matching between the ligand and the protein molecules must contain at least one key point. Key template points are template points that are marked by an ’*’ following the type specifier in each line of the template file (for a description of the template file format see Figure 3.6). In the two other template design approaches described above, all of the points in the template file are marked as key template points by default. Key template points ensure that a least one interaction center of each docked ligand is located in a specific area of the binding site. The use of key template points can significantly speed up the search time, but biases the search towards ligands which can be docked with at least one part in a specific region of the binding site.

2.1.4 Metal Template

Binding-site metal atoms require special processing, because the very short non-covalent to covalent bonds they form with ligands will be interpreted as van der Waals collisions during docking. The metal atom record should be deleted from the `<target>` PDB file and moved into a separate `<target_metal>.pdb` file. The coordiates of the metal are used by the `metal_template` program to generate polar template points which could be matched by ligand atoms. The program creates 32 acceptor template points evenly distributed on the surface of a sphere around each metal. Metal ions included are Ca, Co, Cu, Fe, K, Mg, Mn, Na, and Zn. The sphere radii were determined as the radius corresponding to the first maximum in the metal-ligand distance distributions in the Metalloprotein Database and Browser (The Scripps Research Institute; http://metallo.scripps.edu): Ca, 2.4 Å; Co, 1.9 Å; Cu, 2.1 Å, Fe, 2.2 Å; K, 2.4 Å; Mg, 2.1 Å; Mn 2.2 Å; Na, 2.4 Å; and Zn, 2.1 Å. Those metal template points that are closer than 2.5 Å to any protein or cofactor atom are deleted; similarly, because the protein template was generated in the absence of the metal, any protein template points closer than 2.0 Å to a metal has to be removed using the program `remove_close_points`. The metal template is than merged into the protein template.

2.2 Binding-Site Waters

SLIDE can consider binding-site waters during docking. Typically, these water molecules are taken from the crystal structure of the ligand-free protein and *Consolv* is used to predict if the waters are likely to be conserved upon ligand docking and mediate interactions between the two molecules. *Consolv* is a k-nearest-neighbor classifier system that predicts whether a water is conserved or displaced upon ligand binding based on its local environment[4]. Only those waters that are predicted as being conserved are kept, and there is a penalty based on *Consolv*’s votes, which is coupled to the confidence of the prediction, assessed for displacement of waters that are predicted as conserved. For the format of the water file see Figure 3.7, which also describes how to include the prediction confidence. Information on obtaining and using *Consolv* can be found under “Software” on the Protein Structural Analysis and Design Laboratory web site (http://www.bch.msu.edu/labs/kuhn). Water molecules can also be included as a part of the protein PDB file itself, in which case they are never displaced.

2.3 Ligand Interaction Points

Hydrophobic ligand interaction points are assigned using a rule-based approached summarized in Figure 2.2. These rules strive to place a point at every approximately 1.5 hydrophobic carbon atoms in hydrophobic chains an around the circumference of hydrophobic rings. For this approach, carbon and sulfur atoms bonded only to carbon, sulfur, or hydrogen atoms are considered to be hydrophobic. Other atoms are taken as hydrophilic.

Hydrogen bonding interaction points in the ligand are identified as atoms capable of accepting or donating hydrogen bonds, based on the SYBYL atom types in the mol2 file (described at http://www.tripos.com).

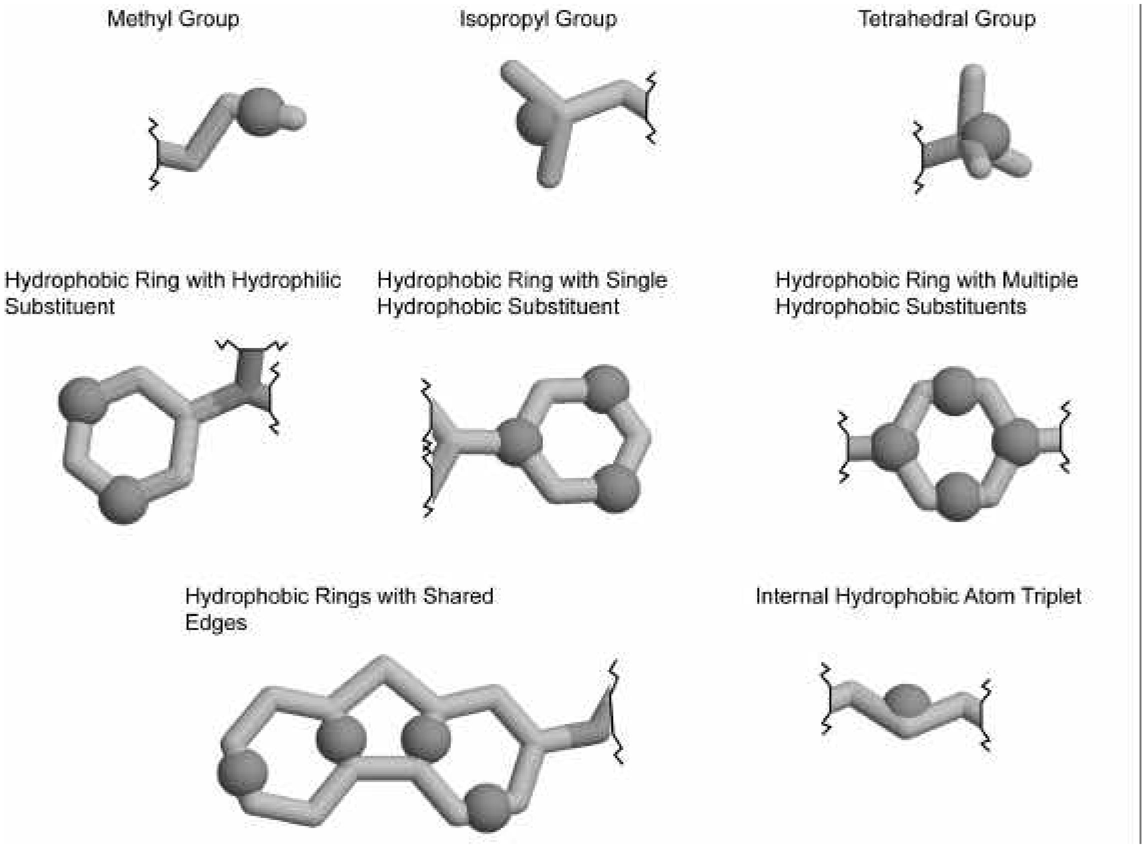

Methyl Group
Isopropyl Group
Tetrahedral Group
Hydrophobic Ring with Hydrophilic Substituent
Hydrophobic Ring with Single Hydrophobic Substituent
Hydrophobic Ring with Multiple Hydrophobic Substituents
Hydrophobic Rings with Shared Edges
Internal Hydrophobic Atom Triplet

Figure 2.2: Summary of rules for hydrophobic interaction point assignment. The overall goal is to place a point at approximately every 1.5 carbon atoms. These rules also place points around the edge of hydrophobic ring structures. Hydrophobic interaction points are denoted as green spheres, carbon atoms in grey (light grey in a black and white print), and nitrogen atoms, representing hydrophilic atoms, in blue (dark grey in a black and white print).

2.4 Multi-Level Hashing

A multi-level hashing approach is used to directly access all feasible template triangles. Before the search, all possible triangles are generated out of the set of template points and are indexed via four levels of hash tables (see Figure 2.3). The parameter `MAX_TEMPLATE_TRIANGLES` defines the maximal number of triangles of template points that can be accessed via the multi-level hashing. This variable can be set in the parameter files (for a description of the use of parameter files see Section 3.5). The first hash table is based on the chemistry of the interaction points of the triangles. The four different types of template points provide 20 indices for the first hash table. The indices in the second hash table are based on the perimeter of the triangles, the third on the length of their

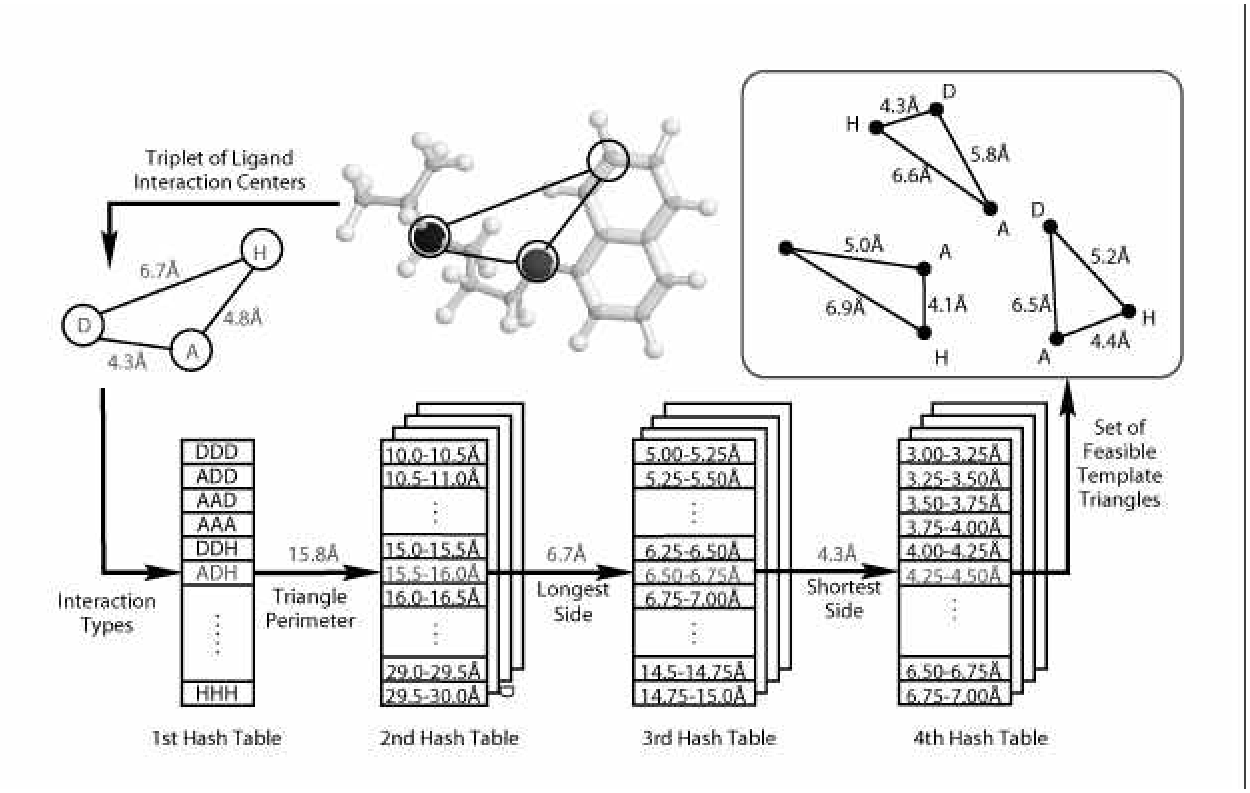


Figure 2.3: For each set of three ligand interaction centers all feasible template triangles are directly accessed via a four-level hashing scheme, which indexes the precomputed template triangles based on their chemical and geometrical properties.

longest side, and the fourth on the length of the shortest side. By using these four properties for a given set of three ligand interaction centers direct access to all template triangles with compatible properties is very efficient. However, triangles must be checked for a feasible one-to-one mapping before the ligand is transformed into the binding site. This is done via a least-squares fit superposition of the three ligand interaction centers onto the three template points. There are two variables that control thresholds for the triangle mapping: `DME_THRESHOLD` and `RMS_THRESHOLD`, both are set in the parameter files. `DME_THRESHOLD` defines the upper bound for the distance matrix error (DME) for the distances between the ligand triangle in comparison to the distances in the template triangle for a given one-to-one mapping. The mapping with lowest DME, if below the threshold, is used to superimpose the triangles onto each other. `RMS_THRESHOLD` defines the maximal accepted value for the root-mean square deviation of this superposition.

For all computations associated with the triangle mapping, those matchings which involve hydrophobic interaction points can be less exact. Weighted DME, RMSD, and least-squares-fit superpositions are computed to take this into account by weighting distances involving hydrophobic points with a factor of 0.3 relative to hydrogen-bond points.

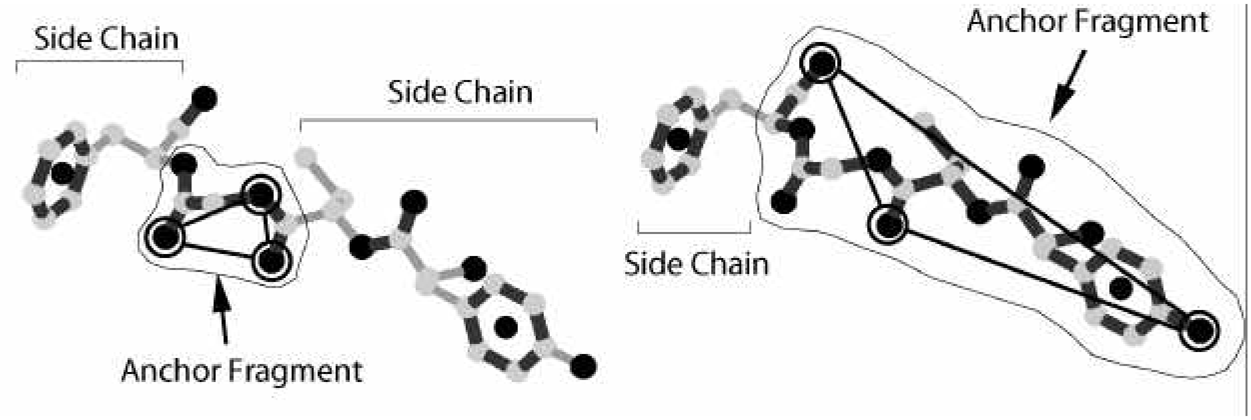


Figure 2.4: Two different triangle mappings are shown for one ligand. The choice of the three ligand interaction centers determines the rigid part of the ligand, its anchor fragment. All rotatable bonds in the remaining parts of the ligand (the side chains) are fully rotatable to resolve collisions after docking.

The matched ligand interaction centers define the anchor fragment, and all chemically and geometrically feasible anchor fragments are tested for each ligand candidate. All flexible bonds within this part of the ligand are rigidified. The remaining parts of the ligand are kept flexible (Figure 2.4), such that all single bonds in these parts can be rotated later if necessary to resolve collisions with protein atoms. Collisions of the anchor fragment with protein main-chain atoms are resolved by iterative translations of the anchor fragment as a rigid body. The allowed overlap between anchor-fragment atoms and main-chain atoms in the target protein (including Cß's) is specified by the parameter `ANCHOR_OVERLAP`, the maximal translation distance is controlled by the parameter `ANCHOR_TRANSLATION`. If all main-chain collisions can be resolved, the side chains are added to the anchor fragment, in the conformation found for the ligand in the database.

2.5 Modeling of Induced Complementarity

Before resolving collisions and computing the chemical complementarity for a protein-ligand complex, all ligand dockings with low shape complementarity are ruled out. In 89 complexes[5], which were used to tune SLIDE's scoring function, on average 88% of the ligand carbon atoms were placed within a distance of 4.0 Å of any protein atom. Based on this observation, in SLIDE all ligand dockings with more than 50% of carbon atoms without any protein atom within 4.0 Å are discarded before doing the computationally expensive modeling of induced complementarity.

Induced fit for the interface between the two molecules is modeled by resolving any collisions of their flexible parts by directed rotations of single bonds either in the ligand or in side chains of the protein. The variable SIDE_CHAIN_OVERLAP in the parameter files defines the tolerated intermolecular overlap, INTRA_OVERLAP is the corresponding variable for the threshold of intramolecular overlaps. The radii used for protein and ligand atoms are those determined by Li and Nussinov[6]. For the protein the united atom model is used, i.e. hydrogen positions are not considered. For the ligand atoms it is assumed that hydrogens are included in the structure of the ligands in the screening database. There are typically several applicable rotations to resolve an intermolecular collision, and an approach based on mean-field theory[7] is used to decide which rotations will improve the shape complementarity in the current conformation.

For each pairwise intermolecular collision, those bonds are identified that can be rotated to resolve it. The extent to which intramolecular collisions within the rotated side chain are tolerated during the optimization is controlled by the variable INTERMEDIATELY_TOLERATED_OVERLAP (see Section 3.5). All rotatable bonds that can be used for resolving a collision are stored in a system together with the corresponding minimum rotation angle and the number of non-hydrogen atoms that will be displaced by the rotation. These values provide the basis for the cost of a rotation. A probability is assigned to each rotation, and all rotations that can be used to resolve one particular collision are initialized with equal probabilities. During the mean-field based optimization, these probabilities converge to assign higher values to those rotations that represent a near-optimal choice to resolve a maximal number of collisions with minimal conformational changes in both molecules, without bias to one or the other.

In each cycle of the mean-field optimization process, a mean cost is computed for each rotation in the system, which is based on the cost associated with this rotation and on correlations with other rotations in the system. Two rotations correlate, when they are not independent of each other, hence, only one of them should be applied at the end of an iteration of the mean-field optimization process. There are both positive and negative correlations, which decrease or increase the mean cost, respectively, and their contributions are weighted by the probabilities of the corresponding rotations. It is especially beneficial when a rotation of a single bond in the system can resolve more than one collision. An example for a negative correlation is a rotation that would displace another bond that is included in the system, so that all corresponding computations regarding that bond would no longer be valid.

The probabilities for all rotations in the system are updated at the end of each cycle, taking into account the mean costs of alternative rotations for the same collision. After up to ten cycles, the probabilities have converged to an approximate optimal set of values, which assigns the highest probabilities to those rotations that solve most of the collisions with the lowest overall cost. These rotations are applied if they do not cause any unacceptable intramolecular collisions. If there are still overlaps, up to ten iterations of the mean-field optimization are done to generate shape-complementary conformations of the two molecules. This comprehensive approach to simultaneously modeling protein and ligand flexibility provides a more realistic representation of induced complementarity than has been achieved for other screening approaches, which focus only on ligand flexibility.

2.6 Scoring

Shape-complementary complexes are then ranked by a scoring function SCORE(P,L) consisting of a term HPHOB(P,L) for the hydrophobic complementarity of the interface between protein P and ligand L and a term for the number of intermolecular hydrogen bonds HBONDS(P,L). For computation of the number of intermolecular hydrogen bonds, SLIDE identifies the favorable position of the shared hydrogen if the distance between donor and acceptor is less than 3.5 ° A. For the protein and for ligands without hydrogen positions provided in the corresponding database structure, the positions of the hydrogens are computed; for all other cases the hydrogen position is taken from the crystal structure, and terminal hydrogens are rotated to optimize hydrogen bonding when applicable[8]. Donation to multiple acceptors is considered if the angular constraints are fulfilled. A distance of 1.0 Å is used between the donor and the hydrogen atom, and a range of 120° to 180° is accepted for the D–H???A angle[9]. All hydrogen bonds are considered as giving equivalent contributions to the overall complementarity.

The hydrophobicity measure is based on residue and atom type for the protein and on atom type the ligand. The corresponding values were taken from a statistical survey of hydration of atomic types in 56 protein structures[10]. The contribution of a single ligand atom is based on the comparison of its hydrophobicity value with the average hydrophobicity of the surrounding protein surface atoms. Given the hydrophobicity *h(a)* of an atom *a*, with *h(a)* [0...635] calculated as the average number of hydrations per 1000 occurrences of that atom type (Table II in ref. 10), a value of 0 represents a maximally hydrophobic atom, 635 is maximally hydrophilic, and 317 is intermediate. The hydrophobic complementarity of the contact surface between protein P and ligand L is computed as:

$$HPHOB(P,L)=\sum_{\substack{l_i\in L\\ \#P_i>0}}\frac{avg\left\{h'(l_i),\bar{h}(P_i)\right\}}{\max\left\{abs\left(h'(l_i)-\bar{h}(P_i)\right),32\right\}}$$

Where

$$h'(l_i)=\max\{317-h(l_i),0\}$$

considers only the hydrophobic contribution of ligand atoms ***l_i***. The hydrophobicity of the protein neighborhood P_i for a single ligand atom is defined as the average hydrophobic contribution of all protein atoms p_j within a distance of 4.0 Å of ***l_i***:

$$\bar{h}\left(P_i\right)=\max\left\{\left(317-\frac{1}{\#P_i}\sum_{p_j\in P_i}h\left(p_j\right)\right),0\right\}$$

The denominator in each term of the sum describing the hydrophobic score (HPHOB(P,L)) is always greater than or equal to 32, which is 10% of the maximum score for a single ligand atom.

This ensures that the overall HPHOB(P,L) score is not dominated by a few contacts with very small differences between protein and ligand hydrophobicity.

The scoring function SCORE(P,L) for a collision-free complex is a linear combination of the two terms:

$$SCORE(P,L) = A * HPHOB(P,L) + B * HBONDS(P,L)$$

The weights A and B have been reevaluated to reflect the affinities of 89 high-resolution complexes[5]. These complexes had an average HPHOB(P,L) value of 28.7 and 7.8 hydrogen bonds. Values of 0.59 and 2.76 for A and B, respectively, give a reasonable approximation to the series of measured affinity values for these complexes (linear correlation coefficient of 0.615), which yields a relative contribution of 1.29:1.0 of hydrogen bonds over the hydrophobicity term. Although the weights were tuned based on experimentally observed affinities, it has to be emphasized that the goal is not to predict a binding affinity for a ligand identified and docked by SLIDE, but to relatively rank the potential ligands based on their complementarities in the given (non-optimized) conformation. Note that this score, unlike an energy measure, is scaled such that a higher the score indicates a more favorable complex.

Chapter 3

Installation & Use

This chapter deals mainly with the directory structure and the format for SLIDE's input and output data files. The last section summarizes the steps required to set up and start a screening run.

During installation and use, SLIDE makes use of two environment variables:

- `$SLIDE_DIR` which points to the root of the global SLIDE software directory (e.g. `/usr/soft/slide`), which contains binaries, parameters files, and auxiliary scripts and

- `$SLIDE_DATA_DIR` which points to the root of the directory structure which holds the SLIDE input and output files and is usually located in the users home directory space (e.g. `/home/username/slide_data`).

`$SLIDE_DIR` must be set, with `setenv`, prior to installation and both must be set prior to running SLIDE. In addition, $SLIDE_DIR/bin has to be added to the path using the command:

```
% setenv PATH ${SLIDE_DIR}/bin:${PATH}
```

3.1 Hardware and Software Requirements

SLIDE has been tested on various Unix Systems, including Sun SPARC running Solaris 2.5–2.7, Intel running Solaris 2.7 and 2.8, SGI Octane running Irix64, and RedHat linux. The particular hardware requirements depend on the application cases. The multi-level hashing is the most memory-demanding part. The minimal process size of any screening run is between 20 and 30 MB, depending on the system configuration (e.g. for Sun Solaris Systems the process size can be about .5 times larger on an Intel computer in comparison to a Sun SPARC machine for the same runs). The total process size depends on the number of indexed template triangles. With large, automatically generated templates with up to 200 points the process size can get close to 100 MB. However, the number of indexed triangles depends on the distribution of the template points, since there are upper and lower bounds for the perimeter and side lengths of those triangles, which are actually indexed in the set of all possible triangles.

In terms of disk space, the installation of SLIDE takes about 4 MB for the source code, the binary, and utility programs and scripts. All ligands in the screening databases have to be available in SYBYL mol2 format and have to be generated and stored in addition to the vendor database that is screened. This can be quite space consuming, but the mol2 format provides all information that is needed to asses flexibility and hydrogen-bond capability of the compounds in the screening database, and. Also, this format is used by several other tools, which allows the generation of SLIDE input data files and the direct use of SLIDE's output for further investigation (e.g. fine docking, rescoring with another scoring function, etc.). The required disk space for SLIDE's output depends on the number of potential ligands that are found and docked. In addition to the conformation of the docked ligand in mol2 format, the final positions of interfacial water molecules and the conformations for the rotated side-chains of the target protein are stored for each hit in the corresponding output directories (see Section 3.7). For a database of small ligands (e.g. CSD) the approximate disk space required for each final ligand conformation is in the order of less than 10 kB.

In terms of software requirements, an ANSI-C compiler is necessary (SLIDE has been tested with the SUN cc and GNU gcc compilers), and for the auxiliary scripts, Perl 5.x is needed. For visualizing the binding modes for the potential ligands, the molecular graphics tool Rasmol can be used.

3.2 Compiling SLIDE and Customizing Perl Scripts

For the examples in this section, we will assume SLIDE is to be installed in `/usr/soft/slide`. Before installing and compiling SLIDE, the `tar` file must be placed in the directory above what will be the root directory of the global SLIDE installation, e.g. placed into `/usr/soft.` The environment variable `$SLIDE_DIR` must be set to the path of the global SLIDE directory, e.g. `/usr/soft/slide` before installing and using slide. The tar file `slide.v2.3.tar.gz` has to be unzipped and untarred by the commands

```
% gunzip slide.v2.3.tar.gz
```

```
% tar xfv slide.v2.3.tar
```

This will create the SLIDE directory `/usr/soft/slidev2.3` and extract all the necessary source code for installation. If there are any existing installations of SLIDE that need to be backed up, save them somewhere else and rename `slidev2.3` directory to `slide`. The `install_slide.pl` script, located in the directory `$SLIDE_DIR`, will do the customization during installation, which mainly consists of setting global variables in the Perl scripts and compiling the source code for SLIDE and auxiliary programs. Installation is accomplished in two steps:

1. Edit the first line of the `install_slide.pl` script to point to the local perl installation. This can be identified by by the shell command `which perl`.

2. Run the install script `./install_slide.pl`, which will start the compilation and installation of the SLIDE package into the `$SLIDE_DIR` directory.

After installation, it is useful to extend the path with the directory which contains the SLIDE binaries and script files (directory `$SLIDE_DIR/bin/`).

3.3 Databases

The screening database for SLIDE is a set of mol2 files and an accompanying interaction point (.pts) file stored on a regular Unix file system. The earlier versions of the software (version 2.0 and earlier) included several Perl scripts to set up a database from a set of structures extracted from the CSD. The latest version of the software (version 2.3) has a much easier simple 4-step installation and running process. It contains Shell scripts that automate the process by calling the Perl scipts internally.

Examples of screening databases that can be used for screening with SLIDE:

- The NCI Open Database (http://cactus.nci.nih.gov/)
- ZINC, a free database of commercially available compounds for virtual screening (http://blaster.docking.org/zinc/)
- The Cambridge Structural Database (http://www.ccdc.cam.ac.uk/products/csd/)

It is advisable to filter the databases used for screening to exclude molecules of unreasonable sizes (containing less than 6 or more than 50 heavy atoms, for example) or those containing unusual atom types. The script check_number_of_atoms_mol2.pl can be used to filter out molecules that are too small or too large. The standard interface for requesting structures from the Cambridge Structural Database, quest, can be used for filtering based on atom types and additional criteria. Similarly, the NCI database has a browser that allows selective retrieval of small molecules according to a variety of criteria.

3.3.1 Assigning Hydrogen Atom Positions and Partial Charges for Ligands.

The 3D structures of the molecules that are screened by SLIDE must be provided in the SYBYL mol2 format (Tripos, Inc.). A description of this file format is available on the WWW at http://www.tripos.com/TechBriefs/mol2_format/mol2.html. In addition to the atomic coordinates and the bond network for the molecule, this data format also provides information about the bond types and atomic orbitals, which is necessary to asses the flexibility and the H-bond donor/acceptor properties of the molecule. Neither ligand hydrogen atom positions nor partial charges are required for docking or for the default scoring function within version 2.3 of SLIDE; however, this processing is recommended for compatibility with other frequently used analysis tools, which require hydrogen atom positions and/or partial charges. In this case, a tool like the molcharge module of QuACPAC (Version 1.1, Open Eye Software, Santa Fe, NM; http://www.eyesopen.com/products/applications/quacpac.html) can be used to assign AM1BCC partial charges to ligands. Though AM1BCC charge assignments in QuACPAC are conformationally dependent, we found it was a reasonable approximation, and much more computationally efficient, to first assign partial charges, then use this file as the input to sample ligand conformations (described below). If hydrogen atom positions are not included in the ligand files, they would be automatically generated by the molcharge module of QuACPAC.

3.3.2 Defining Input Conformations for Ligands and Ligand Candidates.

A critical point is the handling of 2D and 3D conformers as input to SLIDE. If only 2D structure files are available, it is necessary to generate an initial, low-energy 3D conformer using a reliable tool such as Corina (Version 3.0, J. Gasteiger, Erlangen, Germany; http://www2.chemie.uni-erlangen.de/software/corina). Given a 3D ligand conformer, SLIDE will dock it and model limited ligand and protein side-chain flexibility, which works well if the ligand is in a low-energy, near-bioactive conformation. However, SLIDE does not thoroughly search alternative, low-energy ligand conformations. Therefore, for any SLIDE docking involving significantly flexible ligands, especially those not likely to be near their bioactive conformation, it is important to sample and provide the low-energy 3D conformations as input to SLIDE. We recommend using Omega (v. 1.8.b3, Open Eye Software, Santa Fe, NM; http://www.eyesopen.com/products/applications/omega.html). We typically also include the Cambridge Structural Database (http://www.ccdc.cam.ac.uk/products/csd) crystal structure for the ligand in the screening database, when available, as a low-energy structure which may approximate the bioactive conformation.

Commands that can be used for each of these tools:

1. 3D conformer generation, if a low energy 3D conformer for the ligand is not provided:

```
% corina –i t=sdf -o t=mol2 -d wh,stergen,preserve,rc,rs,mc=10,names
     <input_filename>.sdf <output_filename>.mol2
```

2. Assigning partial charges and hydrogen atom positions in ligands, if not already provided:

```
% molcharge -in <input_filename>.mol2 -out <output_filename>.mol2
     –am1bcc
```

AM1BCC charges will not be assigned to molecules with unusual atoms, for example B, Co, etc. In this case AM1 charges can be assigned. The command for doing this is:

```
% molcharge -in <input_filename>.mol2 -out <output_filename>.mol2
     –am1
```

3. Sampling 3D conformers of flexible ligands. The maximum number of conformers to generate depends on the maximum number of rotatable bonds in the ligands being docked. Estimating three states for each single bond in a ligand, the maximum number of conformers will be approximately 3^n, where n is the number of single bonds. The default maximum number of 400 conformers in Omega can be reset using the –maxconfs option. The Omega command line to be used:

```
% omega -in <input_filename> -out <output_filename>.mol2
     -includeinput true -multioutputfiles false -warts true -rms 1.0
     -fixcycle true -finalopt true -finalcut true -finalsheffield
     true -ewindow 7.5
```

3.3.3 Setting up the Screening Database

Several different databases can be used within one SLIDE installation, each database is defined by a database definition file in the directory $SLIDE_DATA_DIR/databases/. The files have the extension `.db` and should have a short base name, since this base name will be used to name the corresponding output directories (see Section 3.7). A database can be spread over several directories, and the database definition file lists all directories that include files belonging to this database (Figure 3.1). The first entry in each line in a database definition file provides the full pathname to the directory which contains the mol2 files of the molecules in the database. The second entry in each line defines the name of the interaction file (`<extension>.pts`), which lists all interaction centers for the structures included in that directory. The interaction files are usually generated automatically when building a database.

First have your protein target PDB file and ligand database ready in Unix directories. SLIDE now supports a mixture of single-molecule mol2 files and multi-mol2 files (each containing a number of molecular entries, up to thousands) in the database directory, `<dbase_loc>`. Each ligand must have a distinct name or errors will result. When the target PDB file and ligand database are ready:

```
setup_dbase <target> <template> <database> <dbase_loc> <target>.pdb
Ex: setup_dbase 7dfr unbiased known_ligands /db/csd /pdb/7dfr.pdb
```

`<dbase_loc>` refers to the directory of the ligand database, and `<target>.pdb` is the filename of the target protein PDB file (path must be included); both of these are input files. The template nametag should correspond to one of two types: "`biased`" or "`unbiased`". Additional template names or identifiers may be also used. Both `<target>` and `<template>` will become the names of subdirectories of `$SLIDE_DATA_DIR` where template files and output will be written. In the example above, `/db/csd` is the directory where the ligand files for the `known_ligands` database are located and `/pdb/7dfr.pdb` is the target PDB file. The command, `setup_dbase,` creates all the subdirectories needed to run SLIDE, then extracts interaction centers (for hydrophobic centers and hydrogen-bond donors and/or acceptors) from molecules in `<dbase_loc>` and stores them in `$SLIDE_DATA_DIR/databases/<database>` directory. The nomenclature of the files which store the interaction centers (`.pts` extension) is different for multi-mol2 and singleton mol2 files, as explained below

A. Multi – mol2 files

The `.pts` files are named as `<multi-mol2_filename>_<n>.pts` (where 'n' is any number starting from 0) in the `$SLIDE_DATA_DIR/databases/<database>` directory. Each `.pts` file can have the interaction centers for 7000 ligands in a multi-mol2 file. Subsequent `.pts` will be created if there are more than 7000 ligands in a multi-mol2 file, e.g. `<multi-mol2_filename>_<n+1>.pts` and so on.

B. Singleton mol2 files

The `.pts` files are named as `singleton_<n>.pts` (where 'n' is any number starting from 0) in `$SLIDE_DATA_DIR/databases/<database>` directory. Each `.pts` file can have interaction centers for 7000 individual singleton mol2 files. Subsequent `.pts` files will be created if there are more than 7000 singleton mol2 files within a single subdirectory/directory in `<dbase_loc>`, e.g. `<singleton>_<n+1>.pts` and so on. However, separate `singleton_<n>.pts` file will be created for each subdirectory in `<dbase_loc>`.

3.4 Protein Target

SLIDE requires the 3D structure of the protein target in PDB format in the root directory for this target: `$SLIDE_DATA_DIR/<target>/<target>.pdb`. This file should include only those atoms of the target protein that are considered as being part of the protein for screening as being part of the protein, i.e., heteroatoms that should be ignored during screening (e.g., ligands that are located in the binding site) must be deleted from the original PDB file. All water molecules to be considered during screening must be included in this file and must have the residue-name HOH. Non-water heteroatoms (for example cofactors) cannot be displaced upon ligand docking. and are handled as rigid parts of the protein. All heteroatoms with names beginning with N or O, for nitrogen and oxygen, respectively, can be considered as H-bond donors, acceptors, or doneptors (donor/acceptor), by changing the second and third letter of the atom name to NDD, NAA, NNN, ODD, OAA, and ONN. If salt bridges should be considered during screening, the charges of the corresponding donors and acceptors must be specified in columns 79 and 80 of the atom definition line by listing the strings `1+` and `1-`, respectively, as defined in the PDB file format description available at the RCSB `http://www.rcsb.org/.`

Hydrogen atoms will be stripped from the target PDB file by the `setup_dbase` command. For terminal H's that can assume different positions relative to the donor and pre-donor atoms (e.g., H of a hydroxyl group), SLIDE uses favored bond lengths and angles to calculate whether an H atom could be placed in a position that allows hydrogen bonding.

Please note, this version of SLIDE was written to work with the PDB format description version 2.2. If the PDB files used do not follow this format, the results may not be correct. Also, SLIDE will not work correctly with PDB files which contain atoms with alternate locations. The user should remove all but one alternate location for each atom as the user sees fit.

3.5 Parameter Files

3.5.1 slide.parameters

The default screening parameters can be found in `$SLIDE_DATA_DIR/params/slide.parameters.` Larger values of the parameters (except for `max_template_triangles`) will result in more docked ligand orientations. To override the defaults, make a copy of the parameter file with the new values (still named `slide.parameters`) in your `$SLIDE_DATA_DIR/<target>/<template>/in/` directory. Starting with the defaults is recommended, followed by adjusting as necessary to dock known ligands (minimize false negatives), then adjusting as needed to minimize the number of false positives when screening. A sample parameter file with default values and recommended ranges is shown in Figure 3.2.

Description of each parameter:

`DME_THRESHOLD`**:** This variable sets the maximum for the DME of a mapping of a triangle of ligand interaction centers onto a template triangle. All triangles with a lower DME will actually be superimposed onto each other by computing their least-squares fit (see Section 2.4). A reasonable value for this variable is 0.3 Å.

RMS_THRESHOLD: The maximal value for the root-mean-squares deviation of superimposed triangles, this value is typically identical to the value of the variable DME_THRESHOLD.

ANCHOR_TRANSLATION: When resolving collisions between the rigid anchor fragment and any main-chain atom of the target protein, this variable limits the maximal contribution for each dimension in the global translation vector which defines the direction of the anchor fragment translation. It is useful to restrict the amount of this translation in order to preserve the interaction described by the triangle mapping. A reasonable value for this variable is 0.3 Å.

ANCHOR_OVERLAP: The value of this variable defines the tolerated overlap between anchor-fragment atoms and any main-chain atoms of the target protein. A reasonable value for this variable is 0.3 Å.

SIDE_CHAIN_OVERLAP: The tolerated van der Waals overlap for a pair of protein-ligand atoms, a reasonable value for this variable is 0.3 Å.

INTRA_OVERLAP: The tolerated overlap for contacts between two atoms within the protein or the ligand.

INTERMEDIATELY_TOLERATED_OVERLAP: The sum of all intramolecular overlaps, involving atoms of a rotated side chain, that is tolerated during the mean-field based optimization when resolving collisions. This value is the overlap in addition to the amount controlled by the parameter INTRA_OVERLAP. The higher this value, the more likely is it that accepted intramolecular collisions cannot be resolved in later interactions of the mean-field optimization. A reasonable value for this parameter is 2.0 Å.

FINALLY_TOLERATED_OVERLAP: The sum of all overlaps (intra and inter) in both molecules that is tolerated in a final docking of a ligand if no collision-free conformation was found for this ligand. A default value of 2.0 Å is reasonable to ensure that after the last iteration of the mean-field optimization no ligand orientation is ruled out because of only minor remaining collisions. Maximally one conformation with remaining collisions is stored for each compound. This will be replaced by a collision-free docking if one is found later during the search and if it has a higher score.

FINALLY_TOLERATED_MAX_BUMP While the FINALLY_TOLERATED_OVERLAP parameter defines the maximum sum of all remaining intermolecular collisions for a potential ligand, this parameter defines a maximum for any single intermolecular collision. If the FINALLY_TOLERATED_OVERLAP parameter is less than the FINALLY_TOLERATED_MAX_BUMP parameter, the former takes precedence as the sum will be less than any single one. Use of this parameter can allow the user to allow for a fair number of minor overlaps while preventing any dockings with a point of drastic overlap from passing. A reasonable value for this parameter is 0.5 Å.

SCORE_CUTOFF: This variable defines the lower score for all ligands that are stored in the output directories. Feasible values depend on the size of the potential ligands that are most likely to be found and on the selectivity of the template. For larger compounds a value of 30 is reasonable. For smaller compounds or very restricted binding sites a value of 20 or 15 may be more suitable. If too many potential ligands are found during the search, the a more stringent score cutoff can be implemented after screening via the Perl-script filter_ligands.pl, which will delete all ligands with a score below a given cutoff value (see Section 5.10.10). As a reminder, the scoring function implemented in SLIDE is scaled such that a higher positive score indicates a more favorable complex.

`MAX_TEMPLATE_TRIANGLES`: This variable defines the maximal number of template triangles that are indexed in the multi-level hashing. The number of indexed triangles mainly determines the memory usage of SLIDE, since each single template triangle needs usually 36 bytes of memory (specifically 3 `float` variables plus 6 `integer` variables).

3.5.2 The Hydrogen-Bond Parameter File

The hydrogen-bond parameter file, hbond.defn, is stored in SLIDE's global parameter-file directory `$SLIDE_DIR/params/`. Figure 3.3 shows two entries in the hydrogen-bond parameter file. This file is read in by the program `get_interaction_centers`, which identifies H-bond donors and acceptors and hydrophobic centers in molecules provided in mol2 file format. The syntax is the same as the one used in DOCK[11-13] parameter files. The first entry in each line defines the interaction type (`donor` or `acceptor`), the second entry the atom type together with an optional orbital specifier, and then a list of atoms which must be bonded with this atom (in (..)) and a list of atoms which cannot be bonded to this atom (in [..]) in order for this atom to donate or accept a hydrogen bond.

3.5.3 The Flexible-Bond Parameter File

The flexible-bond parameter file, `flex.defn`, is also stored in SLIDE's global parameter-file directory `$SLIDE_DIR/params`. It describes all types of flexible bonds by listing all pairs of atoms that can be connected by a flexible bond. Each entry consists of three lines, a name entry with the keyword name as the first entry in the line followed by two lines with the keyword `definition`. Each definition line describes an atom entry in the same manner as in the hydrogen-bond definition file described in the previous section.

3.5.4 desf.h

Depending on the size of the binding site and ligand candidates, it might be necessary to adjust the parameters controlling the size of the template triangles matched by the ligand interaction point triplets. These parameters are used to avoid dockings that match tiny anchor fragments in the ligand to tiny regions in the protein template, since shape and chemical complementarity on a larger size scale is preferable. When the binding site is small (accomodating ligands of max. length of ~10 Å, such as a phenyl ring with small substituents) and important interaction centers such as hydrogen-bonding groups are <2.5 Å apart, it is recommended to reset the defs.h parameters. Recommended values for different cases are shown in Figure 3.4. To change the values of these parameters, edit the $SLIDE_DIR/src/slide/inc/defs.h file, then recompile the program:

```
% cd  $SLIDE_DIR/src/slide
% make
```

3.6 Additional Input Files

3.6.1 borders.xyz

The `borders.xyz` file defines the limits of the binding site for unbiased template generation. `This` file is automatically created in `$SLIDE_DATA_DIR/<target>/<template>/in` when running `temp_gen` in ligand mode (`-l` option) or sphere mode (`-c` option).

The `borders.xyz` file needs to be manually created as described below when running the template generator script `temp_gen` in the box mode (`-b` option).

1. One method to generate this file is via the `make_box.pl` auxiliary script. This will create a box in a PDB formatted file given 6 coordinates which correspond to two opposite corners, i.e. points 1 and 8 in Figure 5.1. A first approximation can be achieved using the minimum and maximum coordinates of a known ligand, if available, or of the binding site atoms. It is best to then read in this box with the target protein into a visualization program and orient it as desired. Further size adjustments can be made by generating a new box with changes in the dimensions as desired and reading this new box into the visualization program referenced to the original box, which alleviates the need to repeat the orientation step, though minor adjustments may be needed. This process is repeated until the box is sized and oriented as desired, at which point it is output in the same reference frame as the target protein and the coordinates for the eight points are extracted into the `borders.xyz` file in `$SLIDE_DATA_DIR/<target>/<template>/in (`e.g. `$SLIDE_DATA_DIR/1vr1/unbiased/in`).
2. An alternative is to simply generate a list of the xyz coordinates triplets of the ligand and/or the binding site as defined by the user. If this is done, SLIDE will generate a box which is orthogonal to the protein's internal axes using the maximum and minimum coordinates in this list. Please note, if eight points are given, SLIDE will assume the first method is used; if the list has eight points, but the second method is desired, the user must add a point which falls within the remaining. It is suggested to use the first method, i.e. the 8-cornered box, as using the simpler method often results in areas outside the binding site being included in the template.

3.6.2 Water Molecules and Metals

If the target binding site contains water molecules to be taken into account during the docking step, they should be included into a separate `waters.pdb` file in the `$SLIDE_DATA_DIR/<target>/<template>/in` directory. During the screening, SLIDE will detect this file and handle the waters as described in section 2.2.

Similarly, if the binding site contains metal ions, they should be removed from the PDB file of the target protein and placed in a separate `<target>_metal.pdb` file in the above mentioned directory. The template points around these metals are generated separately using the `metal_template` command. These points have to merged with the automatically generated `template` (using the `temp_gen` command) by editing the `template` file.

```
% metal_template <target_metal.pdb> <binding_site.pdb>
  <points_on_sphere>
```

Since the original `template` was generated in the absence of the metal, some of its points may be too close to the metal. To remove such template points, the `remove_close_points` command should be used:

```
% remove_close_points <metal_file.pdb> <template_file>
```

3.7 Results Output

The output files are organized in a fixed directory structure, the root of which is specified by the environment variable `$SLIDE_DATA_DIR` (Figure 3.5). The Shell-script `setup_dbase` can be used to generate the directory structure for a new screening experiment. The first level under this main data directory is based on the identifier of the target protein, e.g. on its PDB-code. The target protein should have the same name as the target directory (for instance, the target protein name for target directory `1vr1` should be `1vr1.pdb`). The second level is based on the template. Target protein and template define a screening run, and in the corresponding subdirectory all input and output information is stored.

The input for a screening run is specified in the subdirectory `in/`. In this directory, there is a `borders.xyz` file defining the binding site, a template file (named `template`), which lists the positions of the template points in the format defined in Figure 3.6. Both of these files are created by the script `temp_gen`. There is also a PDB file with the binding-site residues (file `<target>.rad`), which is typically generated by the Shell script `run_slide` (see Reference Manual section). Also, the water file (named `waters.pdb`) lists the conserved binding-site waters including the value for the confidence of the prediction, as described in Figure 3.7.

The output of each screening run is stored in subdirectories that are associated with the screening database. The directories `<db>_ligands, <db>_targets,` and `<db>_waters` store the potential ligands, the rotated target side chains, and the positions of the remaining conserved water molecules, some may be displaced during the docking, respectively. All of these filenames are based on the reference code for the compound in the database. The ligand is in mol2 format and the rotated side-chain and water files are in PDB format. The output of `run_slide` can be redirected to a file, which is automatically stored in the in directory. The following information is shown for each docked ligand orientation:

```
 1  ligand_file_name
 2  score
 3  total_hphobicity
 4  penalty (for breaking intra-protein H-bonds when docking the ligand)
 5  number_of_hbonds
 6  number_of_hbonds - number_of_water_mediated_hbonds
 7  number_of_water_mediated_hbonds
 8  diff_intra_target_hbonds (number of disrupted protein H-bonds)
 9  number_of_salt_bridges
10  number_of_repulsive_charge_contacts
11  buried_carbons
12  total_overlap
```

```
   13  number_of_bumps (van der Waals overlaps)
```

Log files for each screening run are stored in the directory log, again associated with the corresponding database specifier, i.e. `<db>.log`. The log file lists all parameter settings that were used for the screening run, provides statistics for the multi-level hashing, and outlines the efficiency of the various filter steps used in SLIDE to efficiently rule out infeasible ligands. For the latter, the number of all molecules that were ruled out in each of the particular steps are listed. At the end of the log file the host, the date, and the overall screening time are listed.

3.8 Tabulating the Results of a SLIDE Run

To create a table summarizing the results after the run has been completed, the script `results_table.pl` is used:

```
% results_table.pl <target> <template> <database> <number> [-conf]
```

The `–conf` option should be used when multiple conformers of each ligand candidate are used as input to SLIDE. In this case, the docked ligand name is expected to have the format `<ligand_name>_<conformer_no>_<orientation_no>.mol2`. When `–conf` is not specified, the program expects the docked ligand names to have the format `<ligand_name>_<orientation_no>.mol2.`

The output table goes to the `standard out` unless it is redirected into a file. It will contain the `slide.parameters` used for the run and the list of top `<number>` ligands sorted by their score. An example of such table is shown in Figure 3.8.

3.9 Restricting the Output of a SLIDE Run

A large database screening my result in an overwhelmingly large number of docked ligands. There are several ways to restrict the amount of output:

- Use the default option of SLIDE is to output only the best scoring orientation of a docked ligand for database screening. If the best scoring orientation has remaining van der Waals overlaps, and there is at least one overlap-free orientation, than both will be saved. This option requires to have the parameter `OUTPUT_ALL_MATCHES` undefined in the `defs.h` file (more about the `defs.h` in Section 3.5.4 and 5.11). Line 10 of the file `$SLIDE_DIR/src/slide/inc/defs.h` file should like this:

  ```
  #undef OUTPUT_ALL_MATCHES
  ```

- Select a score cutoff in the `slide.parameters` file that is sufficiently high to exclude false positives. It is recommended to dock and score known ligands to the target of interest before screening a database for new ligands. The range of SLIDE scores of known ligands will be a good indicator for selecting a reasonable score cutoff. For more about `slide.parameters` see Section 3.5.1.
- Use the `filter_ligands.pl` script to delete dockings with scores lower then the cutoff specified at the command line (for usage see Section 5.10.10).

- Use the script `results_table_del.pl` which is different from `results_table.pl` (Section 3.8) in that it will delete all dockings not included into the table (whose scores are lower than the score of the last entry in the table).

3.10 Summary of Steps Required for Setting Up and Starting a Screening Experiment

Here is a summary of all necessary steps for setting up a screening experiment and starting SLIDE, in all commands and filenames, the expressions `<target>, <template>` and `<database>` represent the actual target code, the template specifier, and the database specifier, respectively. It is also assumed that the environment variables $SLIDE_DATA_DIR and $SLIDE_DIR have been set correctly prior to running these commands.

1. To set up the directory structure for the input/output files associated with our screening experiment, compute interaction centers and create the database definition file, we first run the script setup dbase:

   ```
   % setup_dbase <target> <template> <database> <dbase_locn>
     <pdb_locn>
   ```

   `dbase_locn` refers to the full pathname of the directory in which all the mol2 files of the ligands are located. `pdb_locn` refers to the location where the pdb file for the target protein is located.

2. Now we have to generate a template for the binding site. This can be done by running the program `temp_gen`, in one of four different ways, or by any other approach that provides the template file `$SLIDE_DATA_DIR/<target>/<template>/in/template`, which lists interaction centers that are located in the binding site above the protein surface.

   A. Ligand–based or "biased" template

   This template is based on clustering hydrophobic and/or hydrogen-bonding interaction centers in a set of known ligands. Essentially, this generates a 3D pharmacophore for docking.

   ```
   % temp_gen –g <target> <template> <clustering_threshold>
     <ligand1>.mol2 [<ligand2>.mol2]
   ```

   where `<ligand1>.mol2 [<ligand2>.mol2]` are known ligands given in their protein-bound orientations.

   B. Unbiased knowledge-based template

   - To specify the binding site as a box encompassing a set of known ligand(s) given in their protein-bound orientations:

     ```
     % temp_gen –l <target> <template> <Hbonding_point_density>
       <grid_spacing> <clustering_threshold> <ligand1>.mol2
       [<ligand2>.mol2]
     ```

   - To specify the binding site using a sphere centered at [x, y, z] with radius r (typically 9-10 Å):

```
% temp_gen  -c <target> <template> <Hbonding_point_density>
    <grid_spacing> <clustering_threshold> <x> <y> <z> <r>
```

- To specify the binding site using a box, which can be irregular in shape (Section 5.4):

```
% temp_gen  -b <target>  <template> <Hbonding_point_density>
    <grid_spacing> <clustering_threshold>
```

3. If solvent should be considered during screening, the following call extracts all water molecules within 5.0 Å from any template point from the protein PDB file in the root directory for this target and stores them in the waters-file `$SLIDE_DATA_DIR/<target>/<template>/in/waters.pdb`:

```
% binding_site_waters <target> <template> 5.0
```

   To correctly consider penalties for replacement of these waters when docking potential ligands, the B-values in the waters file have to be modified as outlined in Figure 3.7.

3. If there are metals in the binding site, place the pdb record of the metal in a separate <target>_metal.pdb file and generate template points around it by:

```
% metal_template <target_metal.pdb> <binding_site.pdb>
    <points_on_sphere> > metal_template
```

   Remove template points from the `template` generated in the absence of the metal and which overlap with the metal, using:

```
% remove_close_points <metal_file.pdb> <template_file> >
    unbumped_template
```

   Merge the remaining template points with those around the metal:

```
% cat unbumped_template metal_template > template
```

4. Finally, the actual screening run has to be started by the command:

```
% run_slide <target> <template> <database> <sphere radius> [<output
    file>]
```

   `sphere_radius` refers to the distance from a template point, within which all the binding-site residues and (non-water) heteroatoms shall be considered. If for this experiment any parameter settings other than the global parameters specified in file `$SLIDE_DIR/params/slide.parameters` are used, settings in a parameter file called `$SLIDE_DATA_DIR/<target>/<template>/in/slide.parameters` can override the global parameter values. The command to start SLIDE screens a full database, which is defined in a database definition file within the data directory `$SLIDE_DATA_DIR`. For screening local databases, or only a list of a few compounds, SLIDE can be called with different parameters, which are outlined in the Reference Section.

5. To tabulate the results, run:

```
% results_table.pl <target> <template> <database> <number> [-conf]
```

```
/psa/db/NCI/A    confs_A_0.pts
/psa/db/NCI/B    confs_B_1.pts
/psa/db/NCI/C    confs_C_2.pts
/psa/db/NCI/D  singleton_3.pts
/psa/db/NCI/E  singleton_4.pts
```

Figure 3.1: An example for a database definition file. Each line lists a directory, which includes a set of 3D molecule structures in mol2 format, and the name of the index file, which lists all interaction centers in the molecules. Both the database definition file (`.db`) and the index files (`.pts`) are located in the `$SLIDE_DATA_DIR/databases/<database>/` directory.

```
                                  Default values  Recommended range

DME_THRESHOLD:                        0.3         0.2 – 0.3
RMS_THRESHOLD:                        0.3         0.2 – 0.3
ANCHOR_TRANSLATION:                   0.3         0.2 – 0.3
ANCHOR_OVERLAP:                       0.3         0.1 – 0.3
SIDE_CHAIN_OVERLAP:                   0.3         0.2 – 0.5
INTRA_OVERLAP:                        0.1         0.1
INTERMEDIATELY_TOLERATED_OVERLAP:     2.0         0.5 – 2.0
FINALLY_TOLERATED_MAX_BUMP:           0.5         0.1 – 0.5
FINALLY_TOLERATED_OVERLAP:            2.0         0.5 – 2.0
SCORE_CUTOFF:                        20            10 - 30
MAX_TEMPLATE_TRIANGLES:         1500000            1500000
```

Figure 3.2: A sample global parameter file for SLIDE.

```
# definition of hydrogen-bond donor and acceptors
donor N. ( H )
acceptor O.3 ( 1 * ) [ N. ]
```

Figure 3.3: An example for entries in the hydrogen-bond parameter file. The first line describes a nitrogen that is bonded to at least one hydrogen, which can be donated to form a H-bond, the second line describes an acceptor, an ***sp3*** oxygen that is bonded to at least one other atom, but not bonded to any nitrogen. [..] describes atoms that are not allowed to be connected to the atom specified in the beginning of the line in order to act as a donor or acceptor, (..) describes atoms that have to bonded to this atom in order to accept or donate a hydrogen bond.

	Small binding site	Large binding site
MIN_PERIMETER	7.0	10.0
MAX_PERIMETER	25.0	30.0
MIN_LONGEST_SIDE	3.0	5.0
MAX_LONGEST_SIDE	11.0	12.0
MIN_SHORTEST_SIDE	2.0	2.5
MAX_SHORTEST_SIDE	8.0	10.0

Figure 3.4: Recommended values of some parameters from the `defs.h` file.

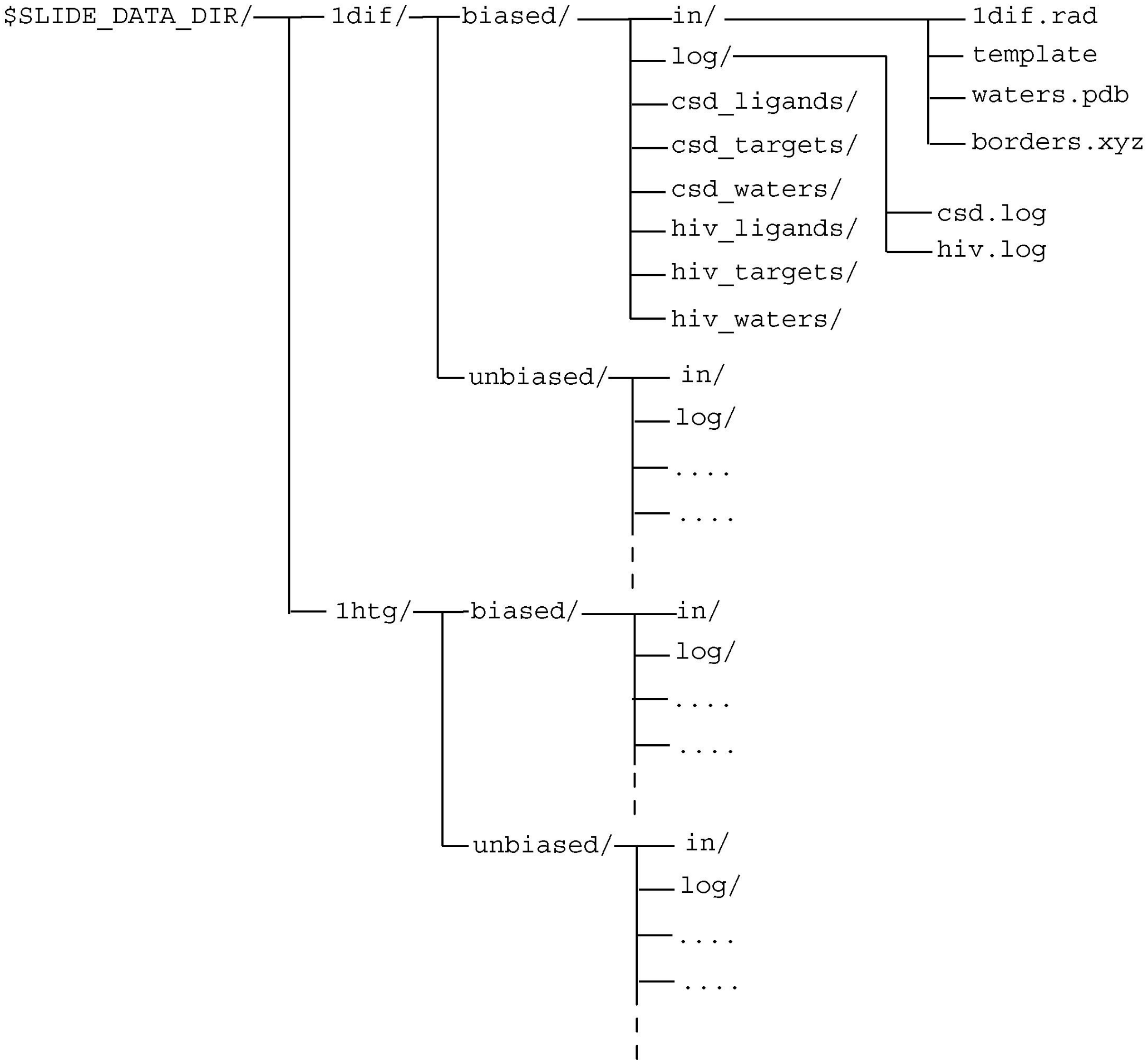


Figure 3.5: A sample directory structure containing Slide’s input and output files for two target proteins, PDB `1dif` and `1htg`. For each of these targets there are two different templates (`biased` and `unbiased`), and two databases are screened (`csd` and `hiv`).

```
# a template file with four key points
H    5.475  70.192  67.458
H    4.337  68.626  62.846
A*   5.604  64.649  62.291
A*   4.337  61.339  68.837
A    0.887  58.801  71.912
A   -0.880  61.099  72.599
D*   6.169  69.585  64.899
D    2.191  69.166  65.940
N*   4.451  63.645  60.536
```

Figure 3.6: A sample template file for a template of nine points: two hydrophobic points (H), four H-bond acceptor points (A), two H-bond donor points (D), and one H-bond doneptor (donor/acceptor) point (N). The points marked with * are key template points, i.e. each template triangle that is indexed in the multi-level hash tables must include at least one of these template points.

```
HETATM  4157  O    HOH   603    0.233  67.584   63.976  0.55 27.16
HETATM  4162  O    HOH   608    0.361  69.609   59.530  0.71 12.40
HETATM  4169  O    HOH   615    3.884  69.138   57.717  0.69 12.05
HETATM  4172  O    HOH   618   -1.808  68.459   66.870  0.62 23.65
```

Figure 3.7: A sample water file which lists four binding-site water molecules in PDB format. The field after the coordinates, which usually lists the occupancy for an atom, lists the prediction for the conservation of this water, based on *Consolv*'s prediction (weakly conserved, 0.50, to strongly conserved, 1.00). Penalty terms for displacement of a water by a non-polar ligand atom are coupled to this value.

```
TOP 20 LIGANDS (OUT OF 70) FOR 1kim (TEMPLATE unbiased)

  slide.parameters

DME_THRESHOLD:                        0.3
RMS_THRESHOLD:                        0.3
ANCHOR_TRANSLATION:                   0.3
ANCHOR_OVERLAP:                       0.3
SIDE_CHAIN_OVERLAP:                   0.3
INTRA_OVERLAP:                        0.1
INTERMEDIATELY_TOLERATED_OVERLAP:     2.0
FINALLY_TOLERATED_MAX_BUMP:           0.5
FINALLY_TOLERATED_OVERLAP:            2.0
SCORE_CUTOFF:                        20
MAX_TEMPLATE_TRIANGLES:         1500000

Fields: 1 rank
        2 ligand name
        3 score
        4 number of protein-ligand H-bonds
        5 hydrophobic complementarity
        6 buried ligand carbons
        7 ligand rotations
        8 protein rotations
        9 sum of remaining van der Waals overlap
       10 top scoring orientation
       11 total number of binding modes

  1       2                 3    4    5     6      7   8    9         10              11

  1 ASI312760             50.3  3   62.4  1.000  0  24  0.732  ASI312760_7_0000     32
  2 CDI356794             49.5  2   70.4  1.000  2  24  1.184  CDI356794_142_0000    6
  3 ASI175234             46.7  5   51.4  1.000  1  39  1.835  ASI175234_22_0000     5
  4 CDI5014               46.2  2   60.1  0.857  0  24  1.069  CDI5014_2_0000        5
  5 ASI49393              46.1  5   41.1  0.929  3  35  0.400  ASI49393_7_0000       4
  6 NET15730              41.4  4   47.2  1.000  0  15  1.412  NET15730_1_0000       3
  7 IBS49650              41.3  0   65.8  0.909  0  21  1.545  IBS49650_3_0000       3
  8 CDI65778              41.1  5   32.5  0.714  0  22  1.776  CDI65778_107_0000    11
  9 SPE16556              40.8  2   46.0  1.000  0  15  1.027  SPE16556_2_0000       4
 10 ASI119475             40.1  2   63.7  1.000  6  20  0.937  ASI119475_4_0000      7
 11 SPE141022             39.9  1   54.0  1.000  4  22  0.633  SPE141022_5_0000      3
 12 CDI165117             39.6  1   48.7  0.941  0  19  1.999  CDI165117_12_0000     1
 13 IBS48581              39.4  1   57.8  0.909  1  19  0.376  IBS48581_2_0000       2
 14 CNR1159               38.9  2   52.3  0.900  3  32  0.775  CNR1159_1_0000        6
 15 NET38058              37.4  5   40.3  1.000 11  26  1.817  NET38058_1_0000      11
 16 ASI194067             36.8  2   53.4  1.000  0  23  0.597  ASI194067_17_0000     7
 17 MAY19089              36.8  2   39.3  0.929  0  15  0.512  MAY19089_9_0000       7
 18 VIT105343             36.6  1   57.8  1.000  0  24  1.890  VIT105343_18_0000     2
 19 ASI174002             36.6  3   38.9  0.923  4  26  1.535  ASI174002_39_0000     3
 20 ASI197014             36.5  4   43.6  0.933  0  28  1.118  ASI197014_4_0000      6
```

Figure 3.8: Example of a table showing the top scoring 20 ligands out of the total 70 docked into the binding site of the target protein is 1kim, using an unbiased template during a SLIDE screening run.

Chapter 4

Examples

In the following examples, screening to identify thrombin ligands is performed. The database consists of a mix of known thrombin ligands, taken from PDB structures, and additional compounds. Screenings will be done using both an unbiased template and a biased (ligand-based) template. For this example, the PDB structure 1vr1, which has an unliganded active site, will be use as the target protein. This example is assumed to be run in `$SLIDE_DATA_DIR` (e.g., `/home/user/slide`). For the command lines in the example, `$SLIDE_DIR` should be replaced with the global SLIDE installation directory, e.g., `/usr/soft/slide`, and `$SLIDE_DATA_DIR` should be replaced with the user's local SLIDE directory, e.g., `/home/user/slide`. All output files can be compared to those in `$SLIDE_DIR/examples/slide`. It is also suggested to include the path to the SLIDE binary files in the user's `PATH` environment variable. This can be done as follows:

```
%setenv PATH ${PATH}:${SLIDE_DIR/bin}
```

4.1 Example 1: Screening Using an Unbiased Template

1. In this example, we would use SLIDE to screen ligands using an unibased template. The first step is to setup a database called `mixed_sample` for the target protein `1vr1` and create the directory structure for `unbiased` template. (\ indicates that the next line is a continuation of the current one. It is used as the command-line line is too long to fit on a single line)

   ```
   % setup_dbase 1vr1 unbiased mixed sample \
        $SLIDE_DIR/examples/slide/databases \
        $SLIDE_DIR/examples/slide/1vr1.pdb
   ```

   Ensure that the entire path is included in giving the location of the directory containing the mol2 files.

2. Copy the customized `slide.parameters` file into the in directory to override the default settings:

```
% cp $SLIDE_DIR/examples/slide/1vr1/unbiased/in/slide.parameters \
     $SLIDE_DATA_DIR/1vr1/unbiased/in
```

3. The next step is to generate the unbiased template file. This is done with the following command:

   ```
   % temp gen -l 1vr1 unbiased sparse 0.6 4.5 \
        $SLIDE_DIR/examples/slide/databases/1a3b ligand.mol2 \
        $SLIDE_DIR/examples/slide/databases/1ad8 ligand.mol2
   ```

   This will generate a template of 80 points, 21 acceptor points, 25 donor points, 8 doneptor points, and 26 hydrophobic points, in the file $SLIDE_DATA_DIR/1vr1/unbiased/in/template. In addition to the template file, several *InsightII* usr files are generated, one each for the different types of template points (acceptor.usr, donor.usr, doneptor.usr, and hydrophobic.usr) and a usr file of the points placed on the grid which are further used for hydrophobic template points generation (points.usr). These files may be useful for visualizing the points. A template.pdb file is also generated, which includes the template points coded as water molecules, residue type HOH, and with the B-values corresponding to point type (acceptor, 0; donor, 50; doneptor, 25; and hydrophobic, 100). Visualization can be done with the included *InsightII* spectrum or by simply coloring by temperature factor, though the automatic coloring scheme may not be intuitive.

   For this example, the idea of key points as described in Section 2.1.5, will be implemented to eliminate the docking orientations which leave key areas of the binding site empty. Here, key points will be defined as those which are within 7.0 Å of the CG carbon of Asp189. The template file needs to be edited to leave only those 9 points marked by * that are also marked by * in the $SLIDE_DIR/examples/1vr1/unbiased/in/key points.pdb. This can be done by simply editing the template file to remove the key marks (*) from all template points except those in the key points.pdb file via comparison of coordinates. Alternatively, the unmark_key_points.pl program can be run on the template file, as described in Section 5.8.29, to remove the key marks for all template points. The file must then be edited to add in key marks back in for the desired atoms.

4. Run SLIDE. This would first extract the shell of binding site residues from the protein for the collision checks during the screening run. Only a shell is used to increase the speed of collision searches since it will not have to check against every atom of the protein, most of which will not collide with the ligand anyway. Then the script would perform the actual screening step.

   ```
   % run_slide 1vr1 unbiased mixed sample 9.0 \
        result_examples_1vr1_unbiased
   ```

   The result output file is $SLIDE_DATA_DIR/1vr1/unbiased/in/result_examples_1vr1_unbiased, the docked ligands are in the $SLIDE_DATA_DIR/1vr1/unbiased/mixed_sample_ligands directory, and the rotated protein side chains are in the $SLIDE_DATA_DIR/1vr1/unbiased/mixed_sample_targets directory. The results file includes information about the run, such as the command-line call and the parameter settings, and output for each molecule screened, including reasons for failure when applicable.

4.2 Example 2: Screening Using a Biased Template

1. The first step is to setup a `mixed_ligand` database and the SLIDE data directory structure for the `biased` template as follows:

```
% setup dbase 1vr1 biased mixed_sample \
     $SLIDE_DIR/examples/slide/databases \
     $SLIDE_DIR/examples/slide/1vr1.pdb
```

   Ensure that the entire path is included in giving the location of the directory containing the `mol2` files.

2. Copy the customized `slide.parameters` file into the in directory to override the default settings:

```
% cp $SLIDE_DIR/examples/slide/1vr1/unbiased/in/slide.parameters \
   $SLIDE_DATA_DIR/1vr1/unbiased/in
```

3. The next step is to generate the unbiased template file.

```
% temp gen -g 1vr1 biased 2.0 \
     $SLIDE_DIR/examples/slide/biased/in/1a46 ligand.mol2 \
     $SLIDE_DIR/examples/slide/biased/in/1a16 ligand.mol2
```

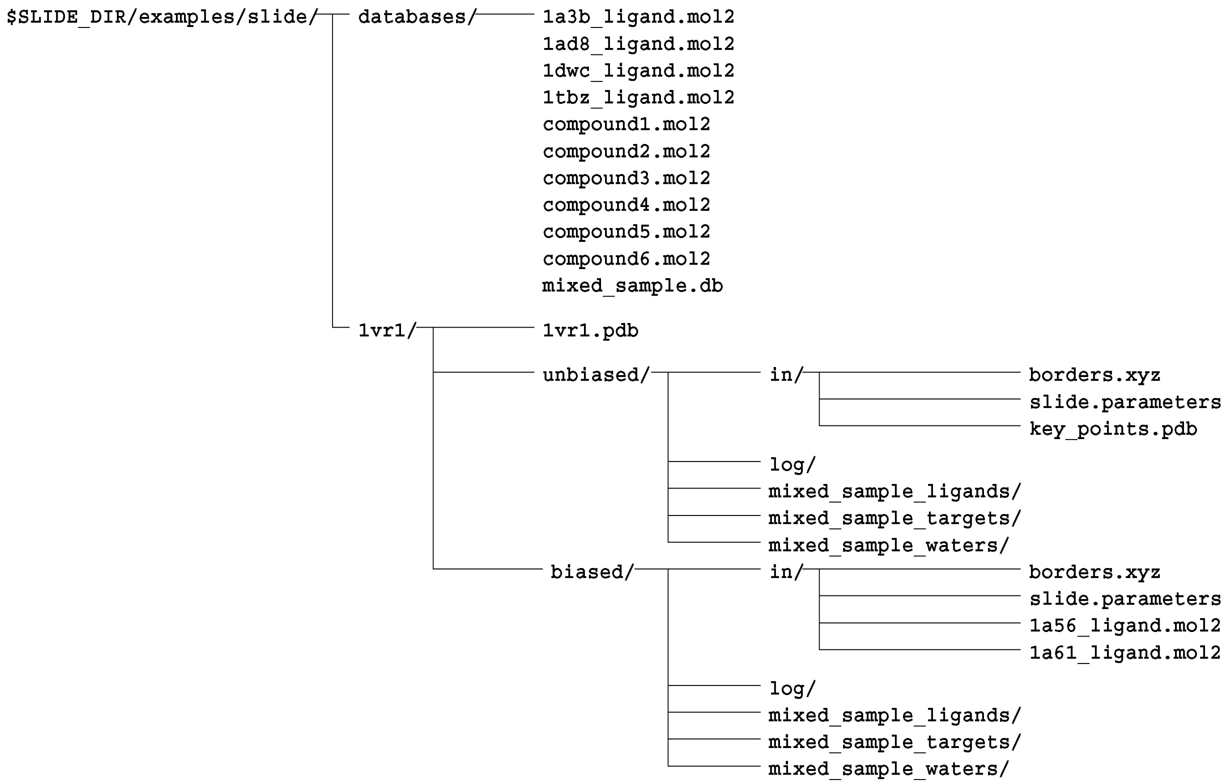
   This will generate a template of 12 points, 4 acceptor points, 6 donor points, and 2 hydrophobic points, in the file `$SLIDE_DATA_DIR/1vr1/biased/in/template`.

4. Run SLIDE.

```
% run slide 1vr1 biased mixed_sample 9.0 result_examples_1vr1_biased
```

   The result output file is
   `$SLIDE_DATA_DIR/1vr1/biased/in/result_examples_1vr1_biased`,
   the docked ligands are in the
   `$SLIDE_DATA_DIR/1vr1/biased/mixed_sample_ligands` directory,
   and the rotated protein side chains are in the
   `$SLIDE_DATA_DIR/1vr1/biased/mixed_sample_targets` directory.

At this point, the directory structure of SLIDE data directory should resemble Figure 4.1.



```
$SLIDE_DIR/examples/slide/
├── databases/
│   ├── 1a3b_ligand.mol2
│   ├── 1ad8_ligand.mol2
│   ├── 1dwc_ligand.mol2
│   ├── 1tbz_ligand.mol2
│   ├── compound1.mol2
│   ├── compound2.mol2
│   ├── compound3.mol2
│   ├── compound4.mol2
│   ├── compound5.mol2
│   ├── compound6.mol2
│   └── mixed_sample.db
└── 1vr1/
    ├── 1vr1.pdb
    ├── unbiased/
    │   ├── in/
    │   │   ├── borders.xyz
    │   │   ├── slide.parameters
    │   │   └── key_points.pdb
    │   ├── log/
    │   ├── mixed_sample_ligands/
    │   ├── mixed_sample_targets/
    │   └── mixed_sample_waters/
    └── biased/
        ├── in/
        │   ├── borders.xyz
        │   ├── slide.parameters
        │   ├── 1a56_ligand.mol2
        │   └── 1a61_ligand.mol2
        ├── log/
        ├── mixed_sample_ligands/
        ├── mixed_sample_targets/
        └── mixed_sample_waters/
```

Figure 4.1: Directory structure of the examples directory used for template generation and screening.

Chapter 5

Reference Manual

Please note, this version of SLIDE was written to work with the PDB format description version 2.2. If PDB files are used which do not follow this format are used, the results may not be correct. Also, SLIDE will not work correctly with PDB files which contain atoms with alternate locations. The user should remove all but one alternate location for each atom as the user sees fit.

5.1 slide

There are four ways to start SLIDE after setting up the databases and the directory structure for the input and output files. SLIDE can be used to screen a globally defined database, a local database, a set of single compounds specified either as command-line arguments, or a set of single compounds listed in a file. SLIDE automatically recognizes, which of the four possible options the user selects:

Usage: `slide <target> <template> <database>`

If only target, template, and database are specified as command-line arguments, the corresponding screening run is started, i.e. database `<database>` is screened for potential ligands for the protein target with binding-site representation template. `<database>` is a globally defined database, i.e. there is a corresponding database definition file in the global installation directory `$SLIDE_DATA_DIR/databases.`

Usage: `slide <target> <template> <database> <mol2-file1> [mol2-file2...]`

This call screens a set of single mol2 files, which are provided as command-line arguments. A selection of one or more mol2 files is screened, using template `<template>`, and stores the potential ligands in the directory `<database>_ligands`. This mode of running SLIDE is useful when checking the binding mode of, for example, single known ligands, before screening a full database to provide a feeling for the docking and score SLIDE assigns to these compounds in order to optimize the template or the parameter settings. Since this is not a full database screen, no log files are written.

Usage: `slide <target> <template> <database> <compound-file>`

This call also screens a set of compounds, but the names of the compounds' mol2 files are listed in a file which lists a single mol2 file in each line rather than passed as command-line arguments. Since this is not a full database screen, no log files are written.

Usage: `slide <target> <template> <database> <local-db-file>`

In this case the fourth command-line argument in the call of SLIDE is the name of a local database definition file, the corresponding file format is outlined in Section 3.3. This option is useful, is a user has set up a database within her/his local environment. In this case log files are written in the log file directory.

5.2 slide_score

Usage: `slide_score protein-file ligand-file [waters-file]`

The program `slide_score` runs the scoring function of SLIDE for a given protein-ligand complex. `protein-file` is a file in PDB format that describes the structure of the target protein and `ligand-file` is a mol2 file of the ligand in its binding conformation in the same reference frame as the protein. The file `waters-file` lists all water molecules that mediate interactions between the two molecules. This file is optional. All water molecules that are included in the PDB file for the protein are ignored.

5.3 ligand_based_template

Usage:
```
ligand_based_template <threshold> <ligand1.mol2>
         [ligand2.mol2 ...]
```

This program generates a template that is based on the chemistry of known ligands, typically based on their binding modes from crystal structures with the target protein. The complexes with these ligands are superimposed onto the structure of the target protein that is used for screening, and the corresponding ligand orientations have to be stored in mol2 format. The program `ligand_based_template` (named `average_template` in previous versions of SLIDE) reads in one or more ligand mol2 files and computes the centers of their carbon rings and the positions of hydrogen-bond donors and acceptors in these ligands. Then complete-linkage clustering is used with the threshold that is defined as the first command-line parameter to cluster points of the same type. The output files are the template-file `template`, a log-file `template.log`, and a PDB-file `template.pdb` that lists the template points as oxygens of water molecules. In these files the temperature factor for the atoms is set to 100, 50, 25, and 0, for hydrophobic, donor, doneptor, and acceptor points, respectively, which gives the user the possibility to automatically color template points differently for the four types using a temperature color spectrum in a molecular graphics program.

5.4 template

Usage:
```
template <target> <template> <grid_spacing>
<dense|sparse|minimal> <hydrophobic-threshold> [ligand.mol2
distance]
```

This program generates an unbiased template for the binding site of the protein that is described by the PDB file `$SLIDE_DATA_DIR/<target>/<target>.pdb`. All heteroatoms but waters (residue name HOH) from this file are considered. The program requires a file `borders.xyz` in the directory `$SLIDE_DATA_DIR/<target>/<template>/in/` that contains a list of points (as x y z coordinates in each line), which define the borders of the binding site. If a set of 8 points is given as the border points, the box, not necessarily a cube, is created using these 8 points as the corners. When giving 8 points, the points must be in a specific order relative to each other as shown in figure 5.1. If other than 8 points are given, the box is defined as the minimal box (edges parallel to base coordinate system) including all listed points. Hydrogen-bonding template points are assigned to positions which can form optimal hydrogen bonds with the protein. Control of the density of hydrogen bonding points can be achieved via the `<dense|sparse|minimal>` parameter setting. The box is filled with points placed on a grid of <`grid_spacing>` . Then, all points are deleted that are either located underneath or are too close to the protein surface, or too far away from any protein atom to serve as template points. If a reference molecule is given, points not within <`distance>` of the molecule are also removed. For each of the remaining points is checked to see if it is located above a hydrophobic surface patch. All hydrophobic points are then clustered using <`hydrophobic_threshold>` . Hydrophobic points which are within one-half the clustering threshold distance of hydrogen bond forming points are eliminated.

?

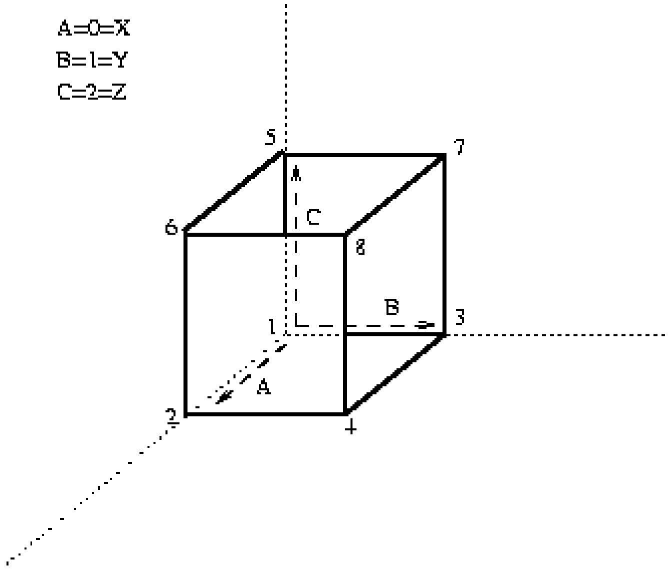


Figure 5.1: Point relationship used for 8 cornered box definition in template program. If using an 8 cornered box in the random template creators, the points must be order in such a way as the program is able to correctly identify the connectivity. Rotations of the box, which move the points in space but retain their connective relationships, are allowed.

?

The program creates several output files, all are stored in the input directory `$SLIDE_DATA_DIR/<target>/<template>/in/`. The file template is the template definition file that is read by SLIDE during screening, the file `template.log` is a log file, and the file template.pdb lists the template points as oxygens of water molecules. In these files the temperature factor for the atoms is set to 100, 50, 25, and 0, for hydrophobic, donor, doneptor, and acceptor points, respectively, which gives the user the possibility to automatically color template points differently for the four types using a temperature color spectrum in a molecular graphics program. There are five files that list points in MSI's `usr` format and can be read by MSI's InsightII to visualize the distribution of random points onto which the template is based. These files are called `points.usr, hphob.usr, donor.usr, acceptor,` and `doneptor`, and include all random points, the hydrophobic points, and all H-bond donor, acceptor, and doneptor points, respectively.

5.5 metal_template

Usage: `metal_template <target_metal.pdb> <binding_site.pdb> <points_on_sphere>`

The program `metal_template` generates acceptor template points around metal ions in the target binding site.

5.6 remove_close_points

Usage: `remove_close_points <metal_file.pdb> <template_file>`

The program `remove_close_points` filters the template and keeps only template points that are further than 2.0 Å from any metal ion in the target binding site.

5.7 check_connectivity

Usage: `check_connectivity <mol2-file>`

The program `check_connectivity` is a reduced version of the program `compute_interaction_centers`, which is described in the previous section. It reads in a mol2 file and checks, if all atoms in the molecule described in this file are connected by bonds. If the mol2 file contains more than one unconnected residue, it outputs an error message and terminates with an error return value.

5.8 generate_rasmol_script

Usage: `generate_rasmol_script  <mol2-file>`

This program reads a mol2 file and outputs a Rasmol script called `<mol2 file>.ras` (with `.ras` having replaced the original `.mol2` file extension). This script can be invoked in

Rasmol with the command `script <mol2 file>.ras`. It then reads the mol2 file for the molecule, colors rotatable bonds in yellow, and shows H-bond donors/acceptors as large spheres (donors in blue, acceptors in red, and doneptors in cyan). This script is helpful to visualize SLIDE's interpretation of flexibility and chemistry for a molecule.

5.9 Shell Scripts

The following Shell scripts have been included in the SLIDE distribution starting with version 2.01 to simplify and automate the process of setting up the database, template generation and screening. They can be used in conjunction with the other Perl scripts provided with the software.

5.9.1 run_slide

Usage: `run_slide <target> <template> <database> <sphere_radius> [<output file>]`

This script checks for all the necessary files for running slide and automates slide's running process. It first runs `binding_site_residues.pl,` which creates the `<target>.rad` file in the template input directory. The `<target>.rad` file includes only those residues of the target protein that have at least one atom within the sphere radius from any template point. A value of 9.0 or 10.0 Å is reasonable for `sphere_radius`. The script then checks if all the files necessary for running slide are present in the correct locations. Then it runs slide and displays the output. If an output file is specified, the result is stored in `$SLIDE_DATA_DIR/<target>/<template>/in/output_file`.

5.9.2 setup_dbase

Usage: `setup_dbase <target> <template> <database> <database_locn> <target_pdbfile_locn>`

This script automates the process of setting up of database and creation of the directory sturcture under `$SLIDE_DATA_DIR` necessary for running SLIDE. The directory structure is created using `slide_setup.pl`. The script then removes the hydrogen atoms from the target protein file at `target_pdbfile_locn` and copies the file into the target directory as `<target>.pdb`. Next it extracts the interaction centers from all the mol2 files present in `database_locn` using `compute_interaction_centers` and stores the result as `<database>.pts` in `$SLIDE_DATA_DIR/databases` directory. It also creates the database definition file `<target>.db` and stores it in `$SLIDE_DATA_DIR/databases`.

5.9.3 temp_gen

This script is used to generate template points for a SLIDE run. It also automates the process of generation of `borders.xyz`. This script can be run in four different modes. In all the modes for generating unbiased templates, `Hbonding_point_density` has to be one of `minimal,` `sparse` or `dense` depending on the number of template points to be generated. Typical values for grid spacing is between 0.5 and 0.6 Å and typical value for clustering threshold is between 3.0 and 4.0 Å. These parameters have been explained in detail under template in Section 5.4.

5.9.3.1 Generating Ligand-Based/Biased Template

Usage:
```
temp_gen –g <target> <template> <clustering_threshold>
  <ligand1>.mol2 [<ligand2>.mol2..]
```

When run in this mode, the scripts calls the program `ligand_based_template` to generate the template points.

5.9.3.2 Generating Unbiased Template Using Ligand(s) to Specify the Binding Site

Usage:
```
temp_gen –l <target> <template> <Hbonding_point_density>
  <grid_spacing> <clustering_threshold> <ligand1>.mol2
  [<ligand2>.mol2..]
```

In this mode, the script first creates `borders.xyz` in `$SLIDE_DATA_DIR/template/target/in`. The box is created around the specified ligands in the frame of reference of target protein with a tolerance of 2.0 Å distance. It then calls the program `template` to generate the template points.

5.9.3.3 Generating Unbiased Template Using a Sphere to Specify the Binding Site

Usage:
```
temp_gen –c <target> <template> <Hbonding_point_density>
  <grid_spacing> <clustering_threshold> <x> <y> <z> <r>
```

This mode is used to specify the binding site using a sphere. In this mode, the script first creates `borders.xyz` in `$SLIDE_DATA_DIR/template/target/in` by creating a box around the sphere located at `[x, y, z]` and radius `r`. It then calls the program `template` to generate the template points.

5.9.3.4 Generating Unbiased Template by Manually Specifying the Binding Site

Usage:
```
temp_gen –b <target> <template> <Hbonding_point_density>
  <grid_spacing> <clustering_threshold>
```

This mode is used when the user has already created the `borders.xyz` file. The details about creating a `borders.xyz` are given in Section 3.6.1. The `borders.xyz` file has to be located in `$SLIDE_DATA_DIR/template/target/in` prior to running this command. In this mode, the script simply calls the program `template`.

5.10 Perl Scripts

5.10.1 add_consolv_occupancy.pl

Usage: `add_consolv_occupancy.pl [-a] <PDB-code>`

This script automates the process of adding the predictions for the conservation of water molecules performed by Consolv into the protein PDB file. It assumes Consolv was run in the current directory and the `.hits, .env,` and `.pred` files must remain in the directory. The PDB file, `PDB-code.pdb`, must also reside in the current directory. Output is a file named `PDB-code.predwats.pdb`. Use of the `-a` option results in all water molecules from the original PDB file being output while the default action is to output only those water molecules which are predicted to be conserved, i.e. those with a confidence of at least 50%.

5.10.2 binding_site_residues.pl

Usage: `binding_site_residues.pl <target> <template> <distance>`

This script reads the PDB-file `target.pdb`, which should be located in the base directory for this target protein, which is `$SLIDE_DATA_DIR/target`, and the `template` file, and outputs a PDB file `target.rad` in the input directory for this template, which includes only those residues of the target protein that have at least one atom within the distance distance from any template point. These residues are the only ones that are considered during collision checks when docking the ligand. A value of 9.0 or 10.0 for the distance (in Å) is reasonable and yields a file with usually around 700 atoms. Smaller values reduce the number of atoms and thus the computation time for all intermolecular and intra-protein bump checks. However, it is actually more than just one shell of binding-side residues necessary to ensure that no rotations of protein side chains are applied, that generate collisions with residues further away from the binding site.

5.10.3 binding_site_waters.pl

Usage: `binding_site_waters.pl <pdb-file> <target> <template> <distance>`

This script extracts all water molecules from the PDB file `pdb-file` which are within `distance` Å from any of the template points. These water molecules are then stored in the file `$SLIDE_DATA_DIR/<target>/<template>/in/waters.pdb` and read by SLIDE during screening.

5.10.4 check_number_of_atoms_mol2.pl

Usage: `check_number_of_atoms_mol2.pl <directory> <min-atoms> <max-atoms>`

This Perl script checks all mol2 files in directory `<directory>` and deletes those that have less than `<min-atom>` or more than `<max-atom>` nonhydrogen/heavy atoms.

5.10.5 count_template_points.pl

Usage: `count_template_points.pl [-s] <template-file>`

This script will give a summary of the number of template points of each type. Default output is in a table form. Simplified output in the form of number of acceptors/number of donor/number of doneptors/number of hydrophobic points/total number of points can be invoked with the -s option. For example, the 18/52/9/56/135 simplified output means 18 H-bond acceptors, 52 H-bond donors, 9 H-bond doneptors, 56 hydrophobic points, and 135 total template points.

5.10.6 create_dbase.pl

Usage: `create_dbase.pl <db_locn> <dbase> <slide_dir> <slide_data_dir> <maxno>`

This script is used for computing ligand interaction points from mol2 files spread over multiple directories. The root directory is `<db_locn>`, in which a directory named `<dbase>` will be created. `<maxno>` is the maximum number of mol2 files in a directory SLIDE can handle and it is defined in `$SLIDE_DIR/src/slide/inc/defs.h.` This script is normally called by `setup_dbase`.

5.10.7 create_pts_files.pl

Usage: `create_pts_files.pl`

This script computes the interaction centers in all mol2 files in all subdirectories in the current directory and lists them in pts-files together with the filenames of the corresponding mol2 files. The `.pts` files are automatically created in the `$SLIDE_DATA_DIR/databases` directory. The program `compute_interaction_centers` is run for each mol2 file. Molecules having less than three interaction centers will be ignored.

5.10.8 cut_mol2.pl

Usage: `cut_mol2.pl mol2-file`

This script will take a mol2 file which consists of multiple molecule entries and seperate it into individual molecules named as MOLNAME ###, each in a mol2 file of the same name. This is useful when generating a set of conformers, which are named identically in a single mol2 file, for a SLIDE screening run.

5.10.9 extract_mol2_residues.pl

Usage: `extract_mol2_residues.pl <directory>`

This Perl script checks for all mol2 files in directory `<directory>`, and if they contain more than one residue, the residues are extracted into separate mol2 files called `<file>_<residue>.mol2` and stored in the same directory. The original mol2 file is deleted.

5.10.10 filter_ligands.pl

Usage: `filter_ligands.pl <target> <template> <db> <cutoff>`

This script checks SLIDE's score for all ligands found in database db when screening for target `target` with template `template`. The command-line parameter `cutoff` defines the lower bound for the score, all ligands with the corresponding target and water files in the directories `<db>_targets` and `<db>_waters`, respectively, that have a lower score than cutoff are deleted.

5.10.11 gen_sph_box.pl

Usage: `gen_sph_box.pl <x> <y> <z> <r>`

This script will create a box around the sphere of radius `<r>` with center in `<x>, <y>, <z>`, and will output the coordinates of the corners of the box. It is normally called by `temp_gen` when the `–c` flag is used.

5.10.12 make_box.pl

Usage: `make_box.pl <x1> <y1> <z1> <x2> <y2> <z2>`

This script will generate a box formatted as a PDB file given the coordinates for the extreme corners, i.e. points 1 and 8 in Figure 5.1. It is used to generate a box which can be read into a molecular visualization program along with the target protein, transformed to the binding site, output to a transformed file, and then have the coordinates extracted into a `borders.xyz` file for use with the template unbiased template generator. For more details on its use, please see Section 3.6.1.

5.10.13 merge_pdb_target.pl

Usage: `merge_pdb_target.pl [-n] <rotation_file> <base-PDB-file>`

This script is used to merge the side chains rotated by SLIDE during a screening run with the original PDB file, i.e. the file given as input to `binding_site_residues.pl`. Output from this script can be used to generate a PDB file appropriate for post-analysis by other programs or scoring methods. The `-n` option may be necessary with older version of SLIDE, specifically the `binding_site_residues.pl` script, which did not output the PDB file's chain IDs, alternate location indicators, or insertion code data.

5.10.14 mol2_dehydrogen.pl

Usage: `mol2_dehydrogen.pl <mol2-file>`

This script will remove the hydrogen atoms from a mol2 file, including removal of the corresponding bonds, correction of the bond numbering, and recalculation of the number of atoms and bonds. It can be useful as there are occasional misnumberings of hydrogen atoms in SLIDE output files which cause problems when reading into another program.

5.10.15 pdb_to_template.pl

Usage: `pdb_to_template.pl <pdb-file>`

This script reads the file `template.pdb` in the current directory and outputs a template-file to standard out, which is used by SLIDE to identify the binding-site template. The input file contains a set of water molecules, which are centered at the position of template points. The type for each template point is assigned based on the temperature factor for each atom. Atoms with an occupancy of 1.00 are marked as key and, otherwise are unmarked. When generating a binding-site template by `temp_gen or template`, both `template` and `template.pdb` files are created automatically. The purpose of the script `pdb_to_template.pl` is to generate a regular template file out of a PDB file `template.pdb`, in which atoms were moved when interactively changing the template, e.g, in a program for crystallographic fitting of structures, or in which atoms have been deleted.

5.10.16 replace_score.pl

Usage: `replace_score.pl <SLIDE-ligand-file> <new-score>`

This script replaces the score in the ligand file output from SLIDE with the supplied score. It can be used after post-screening scoring with another scoring function to generate a set of files with the new score. These files can then be processed using the `filter_ligands.pl` or the `results_table.pl` scripts

5.10.17 results_table.pl

Usage: `results_table.pl <target> <template> <database> <number> [-conf]`

This Perl script outputs a table of the top-scoring `<number>` ligands that SLIDE found for target `target` using binding-site template `template` when screening the database `database`. The `–conf` flag should be used when multiple conformers of each ligand candidate are used as input to SLIDE. In this case, the docked ligand name is expected to have the format `<ligand_name>_<conformer_no>_<orientation_no>.mol2`. When `–conf` is not specified, the program expects the docked ligand names to have the format `<ligand_name>_<orientation_no>.mol2.`

5.10.18 results_table_del.pl

Usage: `results_table_del.pl <target> <template> <database> <number> [-conf]`

This Perl script outputs a table of the top-scoring `<number>` ligands that SLIDE found for target `target` using binding-site template `template` when screening the database `database`. All other ligands will be deleted from the directory together with their corresponding files from the `<database>_targets` and `<database>_waters` directories. The `–conf` flag should be used when multiple conformers of each ligand candidate are used as input to SLIDE. In this case, the docked ligand name is expected to have the format

<ligand_name>_<conformer_no>_<orientation_no>.mol2. When –conf is not specified, the program expects the docked ligand names to have the format <ligand_name>_<orientation_no>.mol2.

5.10.19 rmsd_mol2.pl

Usage: `rmsd_mol2.pl <docked_mol2_file> <reference_mol2_file>`

This script calculates the heavy-atom RMSD between two different orientations of conformations of the same ligand molecule. The script doesn't do superpositioning before the RMSD calculation, and expects to find the same number of atoms in the same order in the two mol2 files.

5.10.20 show_ligands.pl

Usage: `show_ligands.pl <target> <template> <database>`

This script generates a Rasmol script that can be used to browse through the ligands that SLIDE found in the database database for protein target using binding-site template template. The script reads in all corresponding mol2 files based on their ranking, shows the ligands in rasmol, and outputs the score and other relevant information in the terminal window in which Rasmol was started.

5.10.21 slide_setup.pl

Usage: `slide_setup.pl <target> <template> [<database>]`

This script creates all subdirectories in the output directory specified by the environment variable $SLIDE_DATA_DIR, which will hold the input/output data associated with a screening experiment. The database specifier is optional, but the database directories must be created before a SLIDE run is started. This script can also be used to add subdirectories to the directory structure of an existing screening experiment, for example, when screening a new database.

5.10.22 template_to_pdb.pl

Usage: `template_to_pdb.pl <template-file>`

This script reads the file template in the current directory, which lists the interaction centers of the binding-site template for a screening experiment. It outputs a "file" to the standard out which is a PDB file consisting of water molecules at the positions of the template points listed in the input file. The temperature factors (B-values) are set according to the template point's type, 100, 50, 25, and 0 for hydrophobic, H-bond donor, H-bond doneptor, and H-bond acceptor respectively, and the occupancy is set according to the key indicator, 1.00 and 0.50 for key and non-key respectively.

5.10.23 unmark_key_template_points.pl

Usage: `unmark_key_template_points.pl <template-file>`

This script simply removes the key mark (’*’) from all of the points in the template file. It is used when one wants to mark only specific points as key in a automatically generated template. Once the key marks are removed, the user can edit the file and add key marks for only the desired key points.

5.10.24 xyztopdb.pl

Usage: `xyztopdb.pl <coordinate-file>`

This script will convert a file of XYZ coordinates to a PDB formatted file with atoms positioned at these points. The atoms in the resulting PDB formatted file will be of type “O” with residue type of “XXX”.

5.11 Compilation Options

5.11.1 OUTPUT_ALL_MATCHES

When defined, all docked orientations with less then `FINALLY_TOLERATED_OVERLAP` (see Section 3.5.1), including those with remaining overlap, will be output. When undefined, output will consist of all complexes with no remaining overlap and only the best scoring complex found with less than `FINALLY_TOLERATED_OVERLAP` which was found before any overlap-free complex.

5.11.2 OUTPUT_BUMPS

When defined, collisions which occur for each triangle matching trial will be explicitly output to the output stream, i.e. that which goes to standard out or is redirected into a file. Output will be in the form:

```
BUMP no. 10: CB GLU 167 ( 5.058 55.725 41.943) <-->
     40 C.ar ( 5.142 57.146 42.780) : 1.651 ( 0.539).
```

This corresponds to the bump number; the target protein atom's name, residue name, residue number, and coordinates; the ligand atoms' name and coordinates; the atom center to atom center distance; and the resulting overlap. It is advised to only use this option for limited databases as there is a large amount of output generated.

5.11.3 BUMP_FILES

When defined, this will output a ligand mol2 file and a target PDB file for each triangle matching. In order for this parameter to function, the above `OUTPUT_BUMPS` parameter must also be defined. As before, use of this option should be restricted to very small database due to the large volume of output.

5.11.4 OUTPUT_ALL_BUMPS

When defined, this option allows the output of all complexes without regard to the tolerated final overlap criteria. It is advised to use this option with only small databases due to the *very* large volume of output it produces.

Bibliography

1. Kuntz ID, Meng EC, Shoichet BK. Structure-Based Molecular Design. Accounts of Chemical Research 27:117-123, 1994.
2. Ippolito JA, Alexander RS, Christianson DW. Hydrogen bond stereochemistry in protein structure and function. J Mol. Biol. 215:457-471, 1990.
3. McDonald, I. and Thornton, J. M. Atlas of side-chain and main-chain hydrogen bonding. http://www.biochem.ucl.ac.uk/mcdonald/atlas/.
4. Raymer ML, Sanschagrin PC, Punch WF, Venkataraman S, Goodman ED, Kuhn LA. Predicting conserved water-mediated and polar ligand interactions in proteins using a K-nearest-neighbors genetic algorithm. J Mol. Biol. 265:445-464, 1997.
5. Eldridge MD, Murray CW, Auton TR, Paolini GV, Mee RP. Empirical scoring functions: I. The development of a fast empirical scoring function to estimate the binding affinity of ligands in receptor complexes. J Comput. Aided Mol. Des 11:425-445, 1997.
6. Li AJ, Nussinov R. A set of van der Waals and coulombic radii of protein atoms for molecular and solvent-accessible surface calculation, packing evaluation, and docking. Proteins 32:111-127, 1998.
7. Koehl P, Delarue M. Application of a self-consistent mean field theory to predict protein side-chains conformation and estimate their conformational entropy. J Mol. Biol. 239:249-275, 1994.
8. Hooft RW, Sander C, Vriend G. Positioning hydrogen atoms by optimizing hydrogen-bond networks in protein structures. Proteins 26:363-376, 1996.
9. Habermann SM, Murphy KP. Energetics of hydrogen bonding in proteins: a model compound study. Protein Sci. 5:1229-1239, 1996.
10. Kuhn LA, Swanson CA, Pique ME, Tainer JA, Getzoff ED. Atomic and residue hydrophilicity in the context of folded protein structures. Proteins 23:536-547, 1995.
11. Kuntz ID, Blaney JM, Oatley SJ, Langridge R, Ferrin TE. A geometric approach to macromolecule-ligand interactions. J Mol. Biol. 161:269-288, 1982.
12. Leach AR, Kuntz ID. Conformational-Analysis of Flexible Ligands in Macromolecular Receptor-Sites. Journal of Computational Chemistry 13:730-748, 1992.
13. Shoichet BK, Kuntz ID. Matching chemistry and shape in molecular docking. Protein Eng 6:723-732, 1993.

Appendix A

Version History

Version 2.31 – Release Date: February 2005.

Users of SLIDE version 2.3 who wish to use the new scripts and the updated user guide added to this version should contact the authors at slide@.bch.msu.edu .

- Additions to the program:
  - `metal_template`, `points_on_sphere.txt` and `remove_close_points` for template generation around metal ions
  - `results_table.pl` and `results_table_del.pl` to create a tabulated form of the results for an easy overview. The ligands are ranked based on their score with a control over how many should be included in the table.
  - `rmsd_mol2.pl` to calculate RMSD between two conformations or docked orientations of the same molecule in mol2 format.
- Obsolete features related to web interface setup and screening the CSD removed from the program:
  - `binding_site.pl`
  - `create_csd_set_data.pl`
  - `create_global_csd_results_table.pl`
  - `create_global_genericDB_table.pl`
  - `create_html_files_csd.pl`
  - `create_html_files_genericDB.pl`
  - `create_pts_files_csd.pl`
  - `cut_mol2_csd.pl`
  - `known_ligand.pl`
  - `process_csd_set.pl`
  - `renum_pdb_atoms.pl`
- Reformatted the user manual in MS Word. Updated the manual as follows:
  - Added reference Zavodszky et al., JCAMD 2002 to Chapter 2, Methods.

- Corrections to the description of unbiased template generation 2.1.2
- Metal template description added to 2.1.4
- Included a description of how H-bonding ligand interaction points are identified as 2.3.
- Specified Linux as a platform on which SLIDE has been tested and can run in 3.1.
- Added a section on recommended screening databases as 3.3, assigning hydrogen atom positions and partial charges as 3.3.1, and conformer generation for ligands as 3.3.2
- Added section 3.5.4 on parameter settings in the `defs.h` file
- Reorganized Chapters 3 and 4. Chapter 3 includes information on installation and use, while Chapter 4 contains the examples and tutorials.
- Added two entries to Appendix B, Known Problems:
  - no slash in the ligand name is tolerated
  - every conformer name in the multimol2 file has to be different
- Added description of changes for Version 2.3 and Version 2.31 to Appendix A, Version History

- Updated the Quick Guide to have the correct references to the various manual sections. Added metal template description, suggestions to parameter selections and tabulating the result. Removed description of web interface setup.

Version 2.3 – Release Date: April 2004.

- Added Linux-compatibility
- Command line arguments written to `fp_log`
- Added a script called `create_dbase.pl` that is called from `setup_dbase`. This allows the program togo through the database directory recursively and find all interaction centers for all the mol2 files in the directory tree. The maximum number of `mol2` files in every `.pts` file is 7000. The scripts also splits up the directory into chunks of 7000 `mol2` files.
- Updated the quick guide and `README` file.
- Bug fixes
  - to add substructure and residue information to the output mol2 file
  - in the interaction points file (.pts) generation
  - changes in the source code in to generate the same output on different computers structures and platforms.
  - to make the hydrophobic interaction point modeling consistent with the hydrophobic template point modeling when biased template is generated
  - to correct the count of total number of atoms in the docked ligand mol2 file to account for hydrogens added by SLIDE

Version 2.01 – Release Date: May 2002.

- Added `setup_dbase, temp_gen, run_slide` scripts to simplify and automate the process of database setup, template generation and screening for the commonest of cases.
- Added two more ways to specify binding site for generating unbiased template. One method is by specifying known ligands so that the software gets an aprroximate idea of the binding site. The other method is by specifying a sphere. Using these methods, the user does not have to worry about manually creating the `borders.xyz`.
- Changed the default location of `.pts` and `.db` files from `$SLIDE_DIR/databases` to `$SLIDE_DATA_DIR/databases`. This was done to cleanly demarcate user and system space. Under the new release, the users doesn't need to obtain root permissions to make changes to the installation directory and/or database locations.

Version 2.0 – Release Date: November 2001.

- Extraction of van der Waals radii to an easy-to-modify `.h` file in `$SLIDE_DIR/src/slide/inc/vdwrad.h`. Instructions for modifying van derWaals radii are included in this file. If changes are made, SLIDE must be recompiled via make.
- Added missing `ANCHOR_OVERLAP` parameter to .log file output.
- SLIDE now will report an error if it is unable to output a docked ligand instead of resulting in a segmentation fault error.
- Implemented the hydrophobic ligand interaction point method described here, which is significantly different from the method implemented in version 1.1.
- Implemented the hydrogen bond template point method described here, which assigns points to optimal positions based on the protein target chemistry. This is significantly different than the previously used method.
- Added the ability to output all ligands with remaining collisions below a certain threshold, previous versions output only the best scoring of those found before any collision-free complexes were identified. This option is set and controlled in the `$SLIDE_DIR/src/slide/inc/defs.h` file (by defining `OUTPUT_ALL_MATCHES`) and requires recompilation of SLIDE (with make) upon changing. For more details see Section 5.11.
- Added the ability to output the collisions between a ligand and a target protein, including output of associated PDB and mol2 files. Also, there is the option of ignoring the overlap parameters set in the parameter file. These options are useful only for a *very* small database of ligands as they generate a *very* large amount of output. These options are also set in the `$SLIDE_DIR/src/slide/inc/defs.h` via the `OUTPUT_BUMPS`, `BUMP_FILES`, and `OUTPUT_ALL_BUMPS` definitions. For more details, please see Section 5.11.
- Added version number to standard out output, which is normally redirected to a file.
- Added the output of the chain identifier, alternate location identifier, and insertion code to the rotated side chain output files.

- Corrected the output of the ligand mol2 files in regard to listing of the template point matches. In previous versions, the internal numbers, which include the splitting of each doneptor point into an acceptor point and a donor point, were output. Now, the number of the point as taken from the template file is output, allowing easy identification of the triangle mapping used for a particular docked orientation.
- Implemented the ability to control the single maximum tolerated overlap in a docked complex in addition to the total tolerated overlap. This is set in the parameter file via the `FINALLY_TOLERATED_MAX_BUMP` parameter. For more details, please see Section 3.5.1.

Version 1.1 – Release Date: September 2000.

- Bug fixes and changes to source code to facilitate future changes, but which have no functional effect in this release
- Change in hydrophobic point definition from one in an area of average hydrophobic character to one having a required minimum number of hydrophobic atoms and no more than a maximum number of hydrophilic atoms
- Addition of ability to define 8 corners of a box to define a binding-site to place points for template generation (Previous versions only used a maximum and minimum for each of coordinate.)
- Addition of the ability to handle differing clustering thresholds for hydrophobic and hydrogen bond forming template points
- Addition of key point inclusion in template2pdb script and addition of pdb2template script.
- The programs `unbiased_template` and `grid_template` are no longer distributed with SLIDE. They can be obtained upon request from the authors.
- The program `average_template` has been renamed to `ligand_based_template`.

Version 1.0 – Release Date: September 1999

- Initial SLIDE release.

Appendix B

Known Problems

- Some of the output mol2 files may have hydrogens which are not numbered correctly. This may cause some molecular graphics programs to not be able to load the molecule for viewing. A workaround is the use the mol2_dehydrogen.pl script (see Section 5.8.21) to remove the hydrogens and then view the structure containing only heavy atoms.
- Some of the output mol2 files may have formatting problems that cause incorrect additional bonds to be generated in some molecular graphics programs. The correct bonds are being formed and the SLIDE is using the correct bond information, but the visualization may not be correct.
- SLIDE expects the number of atoms listed in a mol2 file header to match the number of actual atoms in the file. If these numbers do not match, an error will occur.
- If the ligand identifier (the line following @<TRIPOS>MOLECULE in the mol2 file) contains the „/" (slash character), the setup_dbase script will result in empty .pts files. Make sure to replace slashes from molecule ID's with other characters.
- When using multi-mol2 files containing different conformers of the same ligand, make sure the conformers have different names, specified in the line after @<TRIPOS>MOLECULE (for example: ABC_001, ABC_002, etc.). Having identical names or identifiers for the conformers will result in segmantation fault.

Appendix C

Error Messages

Error Message/Problem	*Possible Causes*	*Possible Fixes*
SLIDE *Screening Errors*		
General Problems		
`Segmentation Fault`	Ligand directory cannot be found	Ensure that the setup was correctly done, the necessary directories exist, the necessary directories are writeable by the user, and there is sufficient disk space. (This should be detected and reported with a specific error message).
	Some input files not found.	
	To many template points for the available memory.	Decrease the number of template points by increasing the hydrophobic clustering threshold or using a more sparse hydrogen bond point method, or move the run to a machine with more memory.
`No ligand can be docked.`	The `mol2` files of the ligands cannot be found.	Ensure the mol2 files are in the directory pointed at in the `.db` file for the database.
	The template is poor.	Generate a PDB file of the template with `template_to_pdb` and examine it in conjunction with the target (i.e. in InsightII or Rasmol). The template should fill the binding site, especially in any areas known to be important for ligand binding.
	The screening parameters are too stingent.	Choose less stringent screening parameters.
	The ligand has too few interaction points.	SLIDE cannot dock any molecule which has less than three interaction points or which has no triplet of points which meet the triangle parameters, in

Error Message/Problem	*Possible Causes*	*Possible Fixes*
		terms of perimeter and edge lengths, defined in the `defs.h` file.
`FATAL ERROR: SLIDE_DIR environment variable not set`	`SLIDE_DIR` is not set.	Set the `SLIDE_DIR` environment variable to the global SLIDE installation directory (e.g. /usr/soft/slide)
`FATAL ERROR: SLIDE_DATA_DIR environment variable not set`	`SLIDE_DATA_DIR` is not set.	Set the `SLIDE_DATA_DIR` environment variable to the local SLIDE directory (e.g. /home/user/slide)
Problems with General Input Files		
`FATAL ERROR: global parameter file not found`	The parameter file is not in `SLIDE_DIR`/params.	Copy the `slide.parameters` file from the distribution to `SLIDE_DIR/params`. (Even if there is a local parameters file, the global one must exist as it is read first and then the parameters are overwritten by any local ones).
	The parameter file is not readable by the user.	Check that the global parameters file is readable by any group of users that are running SLIDE and correct as needed.
`FATAL ERROR: unknown parameter`	There is a parameter in the `slide.parameters` file which is unknown to SLIDE	Check the `slide.parameters` file to ensure the parameter names are correct and there are no additional parameters.
`FATAL ERROR: unable to open flex.defn`	The `flex.defn` is not in `SLIDE_DIR`/params.	Copy the `flex.defn` file from the SLIDE distribution to `SLIDE_DIR`/params
	The `flex.defn` file is not readable by the user.	Check that the `flex.defn` file is readable by any group of users that are running SLIDE and correct as needed.
`FATAL ERROR: too many flexible-bond rules`	There are more flexible bonds rules in the `flex.defn` file than SLIDE allows.	Change the `MAX_NUMBER_OF_FLEX_BOND_RULES` definition in the `defs.h` file and recompile SLIDE **OR** Rewrite the rules to yield an equivalent set with fewer actual rules.
`FATAL ERROR: only one atom for flexible-bond rule defined`	A flexible bond rule must be defined with two atoms, since it is a bond.	Correct the rule.
`FATAL ERROR: unable to open hyd.defn`	The `hyd.defn` is not in `SLIDE_DIR`/params.	Copy the `hyd.defn` file from the SLIDE distribution to `SLIDE_DIR`/params.
	The `hyd.defn` file is not readable by the user.	Check that the `hyd.defn` file is readable by any group of users that are running SLIDE and correct as needed.

Error Message/Problem	*Possible Causes*	*Possible Fixes*
`FATAL ERROR: too many donor rules`	There are more rules for donors in the `hyd.defn` than SLIDE allows.	Change the `MAX_NUMBER_OF_HYD_BOND_RULES` definition in the `defs.h` file and recompile SLIDE **OR** Rewrite the rules to yield an equivalent set with fewer actual rules.
`FATAL ERROR: too many acceptor rules`	See above.	
`FATAL ERROR:too many prohibitive rules for a single atom`	There are more rules for an atom than are allowed.	Change the `MAX_RULES_PER_BONDED_ATOM` definition in the `defs.h` file. **OR** Rewrite the rules to yield an equivalent set with fewer actual rules.
`FATAL ERROR: too many required rules for a single atom`	See above.	

Problems with Target, Template, and Waters Files

`FATAL ERROR: unable to open template`	The template file does not exist or is in the wrong location.	Ensure a template was generated, that the file is in `SLIDE_DATA_DIR/<target>/<template>/in`, and that the filename is `template`.
	The template file is not readable by the user.	Check that the user has read permissions on the template file and change if necessary.
`FATAL ERROR: Wrong activity definition: ACTIVITY`	A template point exists which is not of acceptor, donor, doneptor, or hydrophobic type.	Examine the template file and correct the point's type.
`FATAL ERROR: internally more than MAX_TEMPLATE_POINTS`	The template is larger than allowed by SLIDE. (The internal representation includes two points for each doneptor.)	Generate a smaller template. **OR** Change the definition in the `defs.h` file and recompile. (It is suggested to use the first fix as a large template is already allowed within the SLIDE program.)
`FATAL ERROR: no template key points`	There are no points marked as key points (by *) in the `template` file.	Mark at least one template point as key. Marking all points as key is equivalent to using no key points.
`FATAL ERROR:more than MAX_PDB_ATOMS atoms`	There are more atoms in the `.rad` file than allowed by SLIDE.	Use a smaller radius with `binding_site_residues.pl` if appropriate. **OR** Change the definition in the `defs.h` file and recompile.
`FATAL ERROR: more Than MAX_PDB_RESIDUES residues`	There are more residues in the `.rad` than allowed by SLIDE	See above.
`ERROR: unknown atom type: NAME RESIDUE RESIDUE_NUM`	The `.rad` file contains an atom type not known to SLIDE.	Add this atom type as instructed at "ADD ATOM" in the `defs.h` file and recompile.

Error Message/Problem	*Possible Causes*	*Possible Fixes*
`ERROR: unknown atom type: RESIDUE RESIDUE_NUM`	The `.rad` file contains an residue type not known to SLIDE.	Change the residue type in the `.rad` file to a standard amino acid type, which must have the standard atoms.
`ERROR: unknown atom level: NAME_RESIDUE_RESIDUE_NUM`	The `.rad` file contains an atom at a nonstandard level (i.e. alpha, beta, gamma, etc.).	Change the atom name for this atom to a standard one.
`WARNING: unable to open waters file`	SLIDE was unable to find or read the `waters.pdb` file.	If you are not including water molecules in the screening run, ignore this error. If water molecules are included in the screening run, ensure the file exists as `SLIDE_DATA_DIR/<target>/<template>/in/waters.pdb` and that it is readable.

Problems with database files

`FATAL ERROR: unable to open database file`	The `.db` file for the database cannot be found.	Ensure the `.db` file is either in the `SLIDE_DIR/databases` directory and named <`database>.db` or is named as called on the command line. Also double-check the spelling of the database name.
	The `.db` file for the database is unreadable	Check that the database `.db` file has read permissions for all users of SLIDE and change if necessary.
`WARNING: unable to open file FILENAME (read_pts_file)`	SLIDE cannot find a `.pts` file listed in the `.db` file.	Check that all of the `.pts` files in the `.db` file exist, have the same name as in the `.db` file, and that they are readable by the user. The `.db` file should include full path names to the `.pts` files
`FATAL ERROR: illegal interaction type`	An interaction point of type other than A, D, N, or H is in the `.pts` file.	Ensure types of only A, D, N, and H are in the `.pts` file and correct as necessary.
`FATAL ERROR: more than MAX_NUMBER_OF_TOTAL_COMPOUND_INTERACTION_POINTS interaction points`	There are more interaction points in the `.pts` file than allowed.	Split this .pts file into separate files, including both files in the appropriate .db file. **OR** Change the definition in the defs.h file and recompile. (The first solution is suggested.)
`FATAL ERROR: more Than MAX_NUMBER_OF_PTS_COMPOUNDS compounds`	There are more compounds in the `.pts` file than allowed.	Split this `.pts` file into separate files, including both files in the appropriate `.db` file. **OR** Change the definition in the `defs.h` file and recompile. (The first solution is suggested.)
`WARNING: ligand NAME has < 3 points`	This ligand has fewer than 3 points and cannot make a triangle match.	Check the interaction point assignment for this molecule. If it is correct, this ligand cannot be docked with the SLIDE

Error Message/Problem	*Possible Causes*	*Possible Fixes*
		algorithm as it requires a triangle of points to match.
`WARNING: unable to open file FILENAME (read_mol2)`	The `.mol2` file for this ligand cannot be found.	Ensure the `.mol2` file for this ligand is in the same directory as the `.pts` file.
	The `.mol2` file for this ligand is unreadable by the user.	Ensure that the SLIDE user has read permissions on the `.mol2` files.
`WARNING: more than MAX_NUMBER_OF_MOL2_ ATOMS (FILENAME)`	There are more atoms in the `.mol2` file than allowed by SLIDE.	Change the definition in the `defs.h` file and recompile. (SLIDE allows for up to 500 atoms, which is likely to be large enough for most ligands. Consider reevaluating this molecule as a potential ligand.)
`WARNING: unknown atom type (assign_type_and_or bit)`	The atom type in the `.mol2` file is not recognized by SLIDE.	Edit the `defs.h` file to add the atom type as described by the instructions in the `defs.h` file at `ADD MOL2 ATOM`
`WARNING: unknown orbit type (assign_type_and_or bit)`	The atom orbital type in the `.mol2` file is not recognized by SLIDE.	Change the orbital type to one known by SLIDE.
`WARNING: unknown radius for ATOM_TYPE (NAME)`	The radius for this type of atom is not known by SLIDE.	Add the radius for this atom to the `vdwrad.h` file (at `ADD_MOL2_ATOM`) and the appropriate assignment code to `read_mol2.c` (also at `ADD_MOL2_ATOM`) and recompile.
`WARNING: more than MAX_NUMBER_OF_MOL2_ BONDS (FILENAME)`	There are more bonds in the `.mol2` file than allowed by SLIDE.	Check that the bond definitions in the `.mol2` file are correct and hange the parameter definition in the `defs.h` file,if necessary, and recompile. SLIDE allows for up to 500 bonds, which is likely to be large enough for most ligands. Consider reevaluating this molecule as a potential ligand.)
`WARNING: unknown bond type: BOND (NAME)`	The bond type for this bond in the `.mol2` file is not recognized by SLIDE.	Change the bond to one which SLIDE recognizes which has equivalent rotatability, i.e. a rotatable bond would be type 1 and a nonrotatable bond would be type 2.
Problems with output files		
`FATAL ERROR: file open failed (write_log_file)`	The directory structure is not properly setup and the `SLIDE_DATA_DIR/<target>/<template>/log` file does not exist.	Run `slide_setup.pl` and check that the log directory is writeable by the user.

Error Message/Problem	*Possible Causes*	*Possible Fixes*
`FATAL ERROR: file open failed (write_ligand_mol2)`	SLIDE directory is not correctly setup and the `SLIDE_DATA_DIR/<target>/<template>/<db>_ligands` directory does not exist.	Run `slide setup.pl` and check that the `<database>_ligands` directory is writeable by the user.
	The disk containing the output directories has no remaining space or the user has reached their hard quota limit (if implemented on the system).	Decrease the template size and/or use more stringent screening parameters to reduce the number of ligands selected. **OR** Use a disk with more available room for the output directories and link into the SLIDE directory structure.
`FATAL ERROR: file open failed (write_target_pdb)`	SLIDE directory is not correctly setup and the `SLIDE_DATA_DIR/<target>/<template>/<db>_targets` directory does not exist.	Run `slide setup.pl` and check that the <`database>_targets` directory is writeable by the user.
	The disk containing the output directories has no remaining space or the user has reached their hard quota limit (if implemented on the system).	Decrease the template size and/or use more stringent screening parameters to reduce the number of ligands selected. **OR** Use a disk with more available room for the output directories and link into the SLIDE directory structure.
`FATAL ERROR: file open failed (write_waters.pdb)`	SLIDE directory is not correctly setup and the `SLIDE_DATA_DIR/<target>/<template>/<db>_waters` directory does not exist.	Run `slide_setup.pl` and check that the `<database>_waters` directory is writeable by the user.
	The disk containing the output directories has no remaining space or the user has reached their hard quota limit (if implemented on the system).	Decrease the template size and/or use more stringent screening parameters to reduce the number of ligands selected. **OR** Use a disk with more available room for the output directories and link into the SLIDE directory structure.

Interaction Point Errors

`FATAL ERROR: SLIDE_DIR environment variable not set`	`SLIDE_DIR` is not set.	Set the `SLIDE_DIR` environment variable to the global SLIDE installation directory (e.g. /usr/soft/slide)
`FATAL ERROR: Unable to open output file`	The user does not have write permission for the current	Check the current directories permissions and change if necessary.

Error Message/Problem	*Possible Causes*	*Possible Fixes*
	directory.	
`WARNING: file open failed`	The user does not have write permission for the current directory.	Check the current directories permissions and change if necessary.
	The disk has no remaining space or the user is over their hard quota limit (if implemented on the system).	Run the program in a directory on a disk with more free space or adjust the quota.
`FATAL ERROR: unable to open flex.defn`	The `flex.defn` is not in `SLIDE_DIR/params`.	Copy the flex.defn file from the SLIDE distribution to `SLIDE_DIR/params`.
	The `flex.defn` file is not readable by the user.	Check that the `flex.defn` file is readable by any group of users that are running SLIDE and correct as needed.
`FATAL ERROR: too many flexible-bond rules`	There are more flexible bonds rules in the `flex.defn` file than SLIDE allows	Change the `MAX_NUMBER_OF_FLEX_ BOND_RULES` definition in the `defs.h` file and recompile SLIDE **OR** Rewrite the rules to yield an equivalent set in fewer actual rules.
`FATAL ERROR: only one atom for flexible-bond rule defined`	A flexible bond rule must be defined with two atoms, since it is a bond.	Correct the rule.
`FATAL ERROR: unable to open hyd.defn`	The `hyd.defn` is not in `SLIDE_DIR/params`.	Copy `hyd.defn` file from the SLIDE distribution to `SLIDE_DIR/params`.
	The `hyd.defn` file is not readable by the user.	Check that the `hyd.defn` file is readable by any group of users that are running SLIDE and correct as needed.
`FATAL ERROR: too many donor rules`	There are more rules for donors in the `hyd.defn` than SLIDE allows.	Change the `MAX_NUMBER_OF_HYD_ BOND_RULES` definition in the `defs.h` file and recompile SLIDE **OR** Rewrite the rules to yield an equivalent set in fewer actual rules.
`FATAL ERROR: too many acceptor rules`	See above.	
`FATAL ERROR:too many prohibitive rules for a single atom`	There are more rules for an atom than are allowed.	Change the `MAX_RULES_PER_BONDED_ATOM` definition in the `defs.h` file. **OR** Rewrite the rules to yield an equivalent set in fewer actual rules.
`FATAL ERROR: too many required rules for a single atom`	See above.	
`FATAL ERROR: unable to open file FILENAME (read_mol2)`	The `.mol2` file for this ligand cannot be found.	Ensure the `.mol2` file for this ligand is in the location specified by the `.db` file.

Error Message/Problem	*Possible Causes*	*Possible Fixes*
	The `.mol2` file of this ligand is unreadable by the user.	Ensure that the SLIDE user has read permissions on the `.mol2` files.
`FATAL ERROR: atom numbers incorrect`	The number of atoms in the `.mol2` file header does not agree with the number of atom records in the file.	Check the `.mol2` file and correct the header and/or the atom record entries as needed.
`FATAL ERROR: more than MAX_NUMBER_OF_MOL2_ ATOMS (FILENAME)`	There are more atoms in the `.mol2` file than allowed by `SLIDE`.	Change the definition in the `defs.h` file and recompile. (SLIDE allows for up to 500 atoms, which is likely to be large enough for most ligands. Consider reevaluating this molecule as a potential ligand.)
`FATAL ERROR: unknown atom type`	The atom type in the `.mol2` file is not recognized by SLIDE.	Edit the `defs.h` file to add the atom type as described by the instructions in the `defs.h` file at `ADD_MOL2_ATOM`
`FATAL ERROR: unknown orbit type`	The atom orbital type in the `.mol2` file is not recognized by SLIDE.	Change the orbital type to one known by SLIDE.
`FATAL ERROR: unknown radius for ATOM_TYPE (NAME)`	The radius for this type of atom is not known by SLIDE.	Add the radius for this atom to the `vdwrad.h` file (at `ADD_MOL2_ATOM`) and the appropriate assignment code to `read_mol2.c` (also at `ADD_MOL2_ATOM`) and recompile.
`FATAL ERROR: more than MAX_NUMBER_OF_MOL2_ BONDS (FILENAME)`	There are more bonds in the `.mol2` file than allowed by SLIDE.	Check that the bond definitions in the `.mol2` file are correct and change the parameter definition in the `defs.h` file, if necessary, and recompile. (SLIDE allows for up to 500 bonds, which is likely to be large enough for most ligands. Consider reevaluating this molecule as a potential ligand.)
`FATAL ERROR: unknown bond type: BOND (NAME)`	The bond type for this bond in the `.mol2` file is not recognized by SLIDE.	Change the bond to one which SLIDE recognizes which has equivalent rotatability, i.e. a rotatable bond would be type 1 and a non-rotatable bond would be type 2.

Template Errors

`FATAL ERROR: SLIDE_DIR environment variable not set`	`SLIDE_DIR` is not set.	Set the `SLIDE_DIR` environment variable to the global SLIDE installation directory (e.g. /usr/soft/slide)
`FATAL ERROR: SLIDE_DATA_DIR environment variable not set`	`SLIDE_DATA_DIR` is not set.	Set the `SLIDE_DATA_DIR` environment variable to the local SLIDE directory (e.g. /home/user/slide)
`FATAL ERROR: unable to open log file`	SLIDE directory structure not set up correctly.	Run `slide_setup.pl`

Error Message/Problem	*Possible Causes*	*Possible Fixes*
`FATAL ERROR: unable to open usr file`	See above.	
`FATAL ERROR: unable to open template.pdb file`	See above.	
`FATAL ERROR: unable to open template file`	See above.	
`FATAL ERROR: unable to open file FILENAME (read_borders_file)`	Binding site borders file, `borders.xyz` does not exist.	Create a `borders.xyz` file as described in Section 3.6.1
`FATAL ERROR: more than MAX_NUMBER_OF_CENTER_POINTS points`	There are too many points in the `borders.xyz` file.	Reduce the number of points in the `borders.xyz` file by removing those enclosed by the outermost shell of points.
`FATAL ERROR: unable to open flex.defn`	The `flex.defn` is not in `SLIDE_DIR/params`.	Copy the `flex.defn` file from the SLIDE distribution to `SLIDE_DIR/params`.
	The `flex.defn` file is not readable by the user.	Check that the `flex.defn` file is readable by any group of users that are running SLIDE and correct as needed.
`FATAL ERROR: too many flexible-bond rules`	There are more flexible bonds rules in the `flex.defn` file than SLIDE allows.	Change the `MAX_NUMBER_OF_FLEX_BOND_RULES` definition in the `defs.h` file and recompile SLIDE **OR** Rewrite the rules to yield an equivalent set in fewer actual rules.
`FATAL ERROR: only one atom for flexible-bond rule defined`	A flexible bond rule must be defined with two atoms, since it is a bond.	Correct the rule.
`FATAL ERROR: unable to open hyd.defn`	The `hyd.defn` is not in `SLIDE_DIR/params`.	Copy the `hyd.defn` file from the SLIDE distribution to `SLIDE_DIR`/params.
	The hyd.defn file is not readable by the user.	Check that the hyd.defn file is readable by any group of users that are running SLIDE and correct as needed.
`FATAL ERROR: too many donor rules`	There are more rules for donors in the `hyd.defn` than SLIDE allows.	Change the `MAX_NUMBER_OF_HYD_BOND_RULES` definition in the `defs.h` file and recompile SLIDE **OR** Rewrite the rules to yield an equivalent set with fewer actual rules.
`FATAL ERROR: too many acceptor rules`	See above.	
`FATAL ERROR:too many prohibitive rules for a single atom`	There are more rules for an atom than are allowed.	Change the `MAX_RULES_PER_BONDED_ATOM` definition in the `defs.h` file. **OR** Rewrite the rules to yield an

Error Message/Problem	*Possible Causes*	*Possible Fixes*
		equivalent set in fewer actual rules.
`FATAL ERROR: too many required rules for a single atom`	See above.	
`FATAL ERROR: unable to open file FILENAME (read_mol2)`	The `.mol2` file for this ligand cannot be found.	Ensure the `.mol2` file for this ligand is in the location specified by the `.db` file.
	The `.mol2` file for this ligand is unreadable by the user.	Ensure that the SLIDE user has read permissions on the `.mol2` files.
`FATAL ERROR: atom numbers incorrect`	The number of atoms in the `.mol2` file header does not agree with the number of atom records in the file.	Check the `.mol2` file and correct the header and/or the atom record entries as needed.
`FATAL ERROR: more than MAX_NUMBER_OF_MOL2_ ATOMS (FILENAME)`	There are more atoms in the `.mol2` file than allowed by SLIDE.	Change the definition in the `defs.h` file and recompile. (SLIDE allows for up to 500 atoms, which is likely to be large enough for most ligands. Consider reevaluating this molecule as a potential ligand.)
`FATAL ERROR: unknown atom type`	The atom type in the `.mol2` file is not recognized by SLIDE.	Edit the `defs.h` file to add the atom type as described by the instructions in the `defs.h` file at `ADD_MOL2_ATOM`
`FATAL ERROR: unknown orbit type`	The atom orbital type in the `.mol2` file is not recognized by SLIDE.	Change the orbital type to one known by SLIDE.
`FATAL ERROR: unknown radius for ATOM_TYPE (NAME)`	The radius for this type of atom is not known by SLIDE.	Add the radius for this atom to the `vdwrad.h` file (at `ADD_MOL2_ATOM`) and the appropriate assignment code to `read_mol2.c` (also at `ADD_MOL2_ATOM`) and recompile.
`FATAL ERROR: more than MAX_NUMBER_OF_MOL2_ BONDS (FILENAME)`	There are more bonds in the `.mol2` file than allowed by SLIDE.	Check that the bond definitions in the `.mol2` file are correct and change the parameter definition in the `defs.h` file, if necessary, and recompile. (SLIDE allows for up to 500 bonds, which is likely to be large enough for most ligands. Consider reevaluating this molecule as a potential ligand.)
`FATAL ERROR: unknown bond type: BOND (NAME)`	The bond type for this bond in the `.mol2` file is not recognized by SLIDE.	Change the bond to one which SLIDE recognizes which has equivalent rotatability, i.e. a rotatable bond would be type 1 and a non-rotatable bond would be type 2.
`FATAL ERROR: unable to open file FILENAME (read_pdb_file)`	The input PDB file does not exist or is unreadable.	Check that the pdb file is in `SLIDE_DIR/<target>/ <target>.pdb`

Error Message/Problem	*Possible Causes*	*Possible Fixes*
`FATAL ERROR: more than MAX_PDB_ATOMS atoms`	There are more atoms in the `.rad` file than allowed by SLIDE.	Use a smaller radius with `binding_site_residues.pl` if appropriate **OR** change the definition in the `defs.h` file and recompile.
`FATAL ERROR: overflow cluster-point array`	There are too many template points generated after clustering.	Generate a template with fewer points by using sparser parameter settings.

Index