User Guide

AutoDock Version 4.2

Automated Docking of Flexible Ligands to Flexible Receptors

Garrett M. Morris, David S. Goodsell, Michael E. Pique, William “Lindy” Lindstrom, Ruth Huey, Stefano Forli, William E. Hart, Scott Halliday, Rik Belew and Arthur J. Olson

Modification date: February 25, 2010 03:08 PM

Contents

Automated Docking

Introduction

AutoDock is an automated procedure for predicting the interaction of ligands with biomacromolecular targets. The motivation for this work arises from problems in the design of bioactive compounds, and in particular the field of computer-aided drug design. Progress in biomolecular x-ray crystallography continues to provide important protein and nucleic acid structures. These structures could be targets for bioactive agents in the control of animal and plant diseases, or simply key to the understanding of fundamental aspects of biology. The precise interaction of such agents or candidate molecules with their targets is important in the development process. Our goal has been to provide a computational tool to assist researchers in the determination of biomolecular complexes.

In any docking scheme, two conflicting requirements must be balanced: the desire for a robust and accurate procedure, and the desire to keep the computational demands at a reasonable level. The ideal procedure would find the global minimum in the interaction energy between the substrate and the target protein, exploring all available degrees of freedom (DOF) for the system. However, it must also run on a laboratory workstation within an amount of time comparable to other computations that a structural researcher may undertake, such as a crystallographic refinement. In order to meet these demands a number of docking techniques simplify the docking procedure. AutoDock combines two methods to achieve these goals: rapid grid-based energy evaluation and efficient search of torsional freedom.

This guide includes information on the methods and files used by AutoDock and information on use of AutoDockTools to generate these files and to analyze results.

Getting Started with AutoDock

AutoDock and **AutoDockTools**, the graphical user interface for AutoDock are available on the WWW at:

http://autodock.scripps.edu/

The WWW site also includes many resources for use of AutoDock, including detailed **Tutorials** that guide users through worked of basic AutoDock usage, docking with flexible rings, and virtual screening with AutoDock. Tutorials may be found at:

http://autodock.scripps.edu/faqs-help/tutorial

AutoDock calculations are performed in several steps: 1) preparation of coordinate files using **AutoDockTools**, 2) precalculation of atomic affinities using **AutoGrid**, 3) docking of ligands using **AutoDock**, and 4) analysis of results using **AutoDockTools.**

Step 1—Coordinate File Preparation. AutoDock4.2 is parameterized to use a model of the protein and ligand that includes polar hydrogen atoms, but not hydrogen atoms bonded to carbon atoms. An extended PDB format, termed **PDBQT**, is used for coordinate files, which includes atomic partial charges and atom types. The current AutoDock force field uses several atom types for the most common atoms, including separate types for aliphatic and aromatic carbon atoms, and separate types for polar atoms that form hydrogen bonds and those that do not. PDBQT files also include information on the torsional degrees of freedom. In cases where specific sidechains in the protein are treated as flexible, a separate PDBQT file is also created for the sidechain coordinates. In most cases, AutoDockTools will be used for creating PDBQT files from traditional PDB files.

Step2—AutoGrid Calculation. Rapid energy evaluation is achieved by precalculating atomic affinity potentials for each atom type in the ligand molecule being docked. In the AutoGrid procedure the protein is embedded in a three-dimensional grid and a probe atom is placed at each grid point. The energy of interaction of this single atom with the protein is assigned to the grid point. AutoGrid affinity grids are calculated for each type of atom in the ligand, typically carbon, oxygen, nitrogen and hydrogen, as well as grids of electrostatic and desolvation potentials. Then, during the AutoDock calculation, the energetics of a particular ligand configuration is evaluated using the values from the grids.

Step 3—Docking using AutoDock. Docking is carried out using one of several search methods. The most efficient method is a Lamarckian genetic algorithm (LGA), but traditional genetic algorithms and simulated annealing are also available. For typical systems, AutoDock is run several times to give several docked conformations, and analysis of the predicted energy and the consistency of results is combined to identify the best solution.

Step 4—Analysis using AutoDockTools. AutoDockTools includes a number of methods for analyzing the results of docking simulations, including tools for clustering results by conformational similarity, visualizing conformations, visualizing interactions between ligands and proteins, and visualizing the affinity potentials created by AutoGrid.

What's New

AutoDock 4.2 includes several enhancements over the methods available in AutoDock 3.0.

Sidechain Flexibility. AutoDock 4.2 allows incorporation of limited sidechain flexibility into the receptor. This is achieved by separating the receptor into two files, and treating the rigid portion with the AutoGrid energy evaluation and treating the flexible portion with the same methods as the flexible ligand.

Force Field. The AutoDock 4.2 force field is designed to estimate the free energy of binding of ligands to receptors. It includes an updated charge-based desolvation term, improvements in the directionality of hydrogen bonds, and several improved models of the unbound state.

Expanded Atom Types. Parameters have been generated for an expanded set of atom types including halogens and common metal ions.

Desolvation Model. The desolvation model is now parameterized for all supported atom types instead of just carbon. Because of this, the `constant` function in AutoGrid is no longer used, since desolvation of polar atoms is treated explicitly. The new model requires calculation of a new map in AutoGrid which holds the charge-based desolvation information.

Unbound State. Several models are available for estimating the energetics of the unbound state, including an extended model and a model where the unbound state is assumed to be identical with the protein-bound state.

For users of AutoDock 4.0, there are several changes in AutoDock 4.2:

Default Unbound State. The default model for the unbound state has been changed from "extended" to "bound=unbound". This is in response to persistent problems sterically-crowded ligands. The "extended" unbound state model is available in AutoDock 4.2 through use of the "unbound extended" keyword.

Backwards Compatibility. We have made every attempt to ensure that docking parameter files generated for use in AutoDock 4.0 should be correctly run by AutoDock 4.2.

Support

AutoDock is distributed *free of charge*. There are some caveats, however. Firstly, since we receive limited funding to support the academic community of users, we cannot guarantee rapid (or even slow) response to queries on installation and use. While there is documentation, it may require at least some basic Unix abilities to install. If you still need help:

(1) Ask your local system administrator or programming guru for help about compiling, using Unix/Linux, *etc*.

(2) Consult the AutoDock web site, where you will find a wealth of information and a FAQ (Frequently Asked Questions) page with answers on AutoDock:
`http://autodock.scripps.edu/faqs-help`

(3) If you can't find the answer to your problem, send your question to the AutoDock List (ADL) or the AutoDock Forum. There are many seasoned users of computational chemistry software and some AutoDock users who may already know the answer to your question. You can find out more about the ADL on the WWW at:
`http://mgldev.scripps.edu/mailman/listinfo/autodock`
The Forum is available on the WWW at:
`http://mgl.scripps.edu/forum`

(4) If you have tried (1), (2) and (3), and you still cannot find an answer, send email to goodsell@scripps.edu for questions about AutoGrid or AutoDock; or to rhuey@scripps.edu, for questions about AutoDockTools.

Thanks for your understanding!

E-mail addresses

Arthur J. Olson, Ph.D. olson@scripps.edu
David S. Goodsell, Ph.D. goodsell@scripps.edu
Ruth Huey, Ph.D. rhuey@scripps.edu

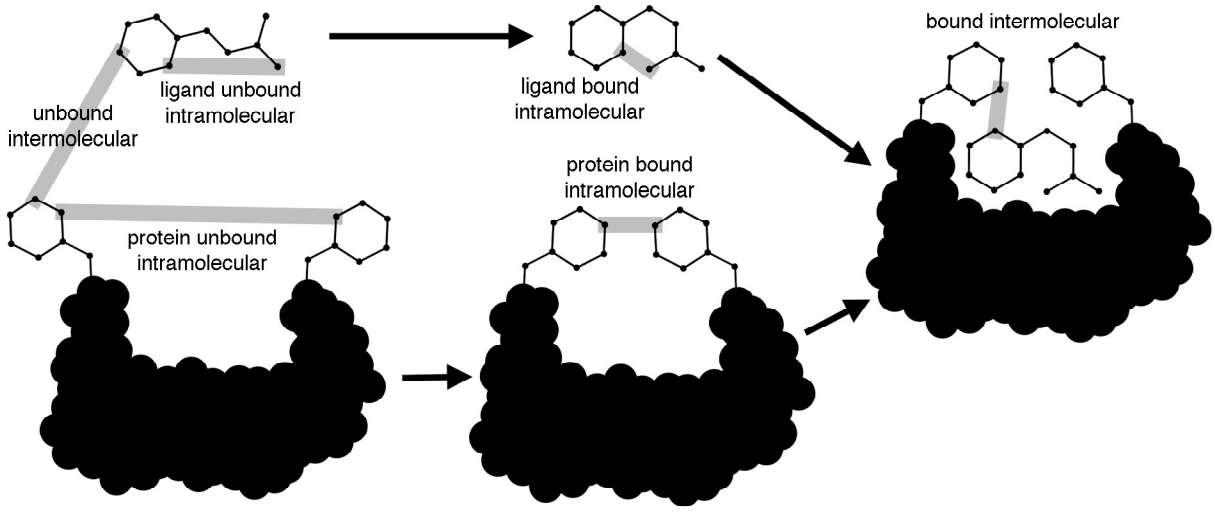
Fax: *+ (858) 784-2860*

The Scripps Research Institute,
Molecular Graphics Laboratory,
Department of Molecular Biology, Mail Drop MB-5,
10550 North Torrey Pines Road,
La Jolla, CA 92037-1000, U.S.A.

Theory

Overview of the Free Energy Function

AutoDock 4.2 uses a semiempirical free energy force field to evaluate conformations during docking simulations. The force field was parameterized using a large number of protein-inhibitor complexes for which both structure and inhibition constants, or K_i, are known.



The force field evaluates binding in two steps. The ligand and protein start in an unbound conformation. In the first step, the intramolecular energetics are estimated for the transition from these unbound states to the conformation of the ligand and protein in the bound state. The second step then evaluates the intermolecular energetics of combining the ligand and protein in their bound conformation.

The force field includes six pair-wise evaluations (V) and an estimate of the conformational entropy lost upon binding (ΔS_{conf}):

$$\Delta G = (V_{bound}^{L-L} - V_{unbound}^{L-L}) + (V_{bound}^{P-P} - V_{unbound}^{P-P}) + (V_{bound}^{P-L} - V_{unbound}^{P-L} + \Delta S_{conf})$$

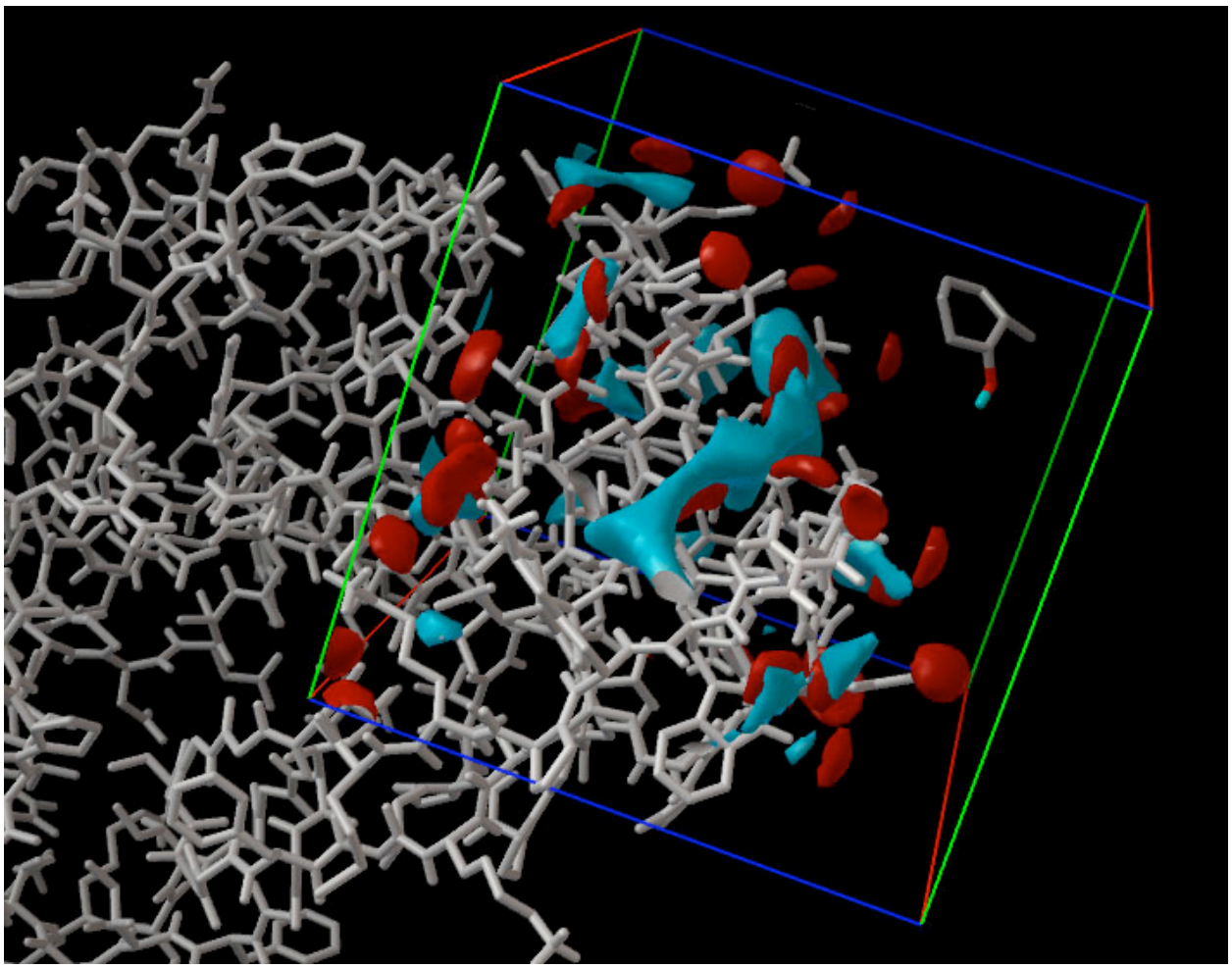
where L refers to the "ligand" and P refers to the "protein" in a ligand-protein docking calculation.

Each of the pair-wise energetic terms includes evaluations for dispersion/repulsion, hydrogen bonding, electrostatics, and desolvation:

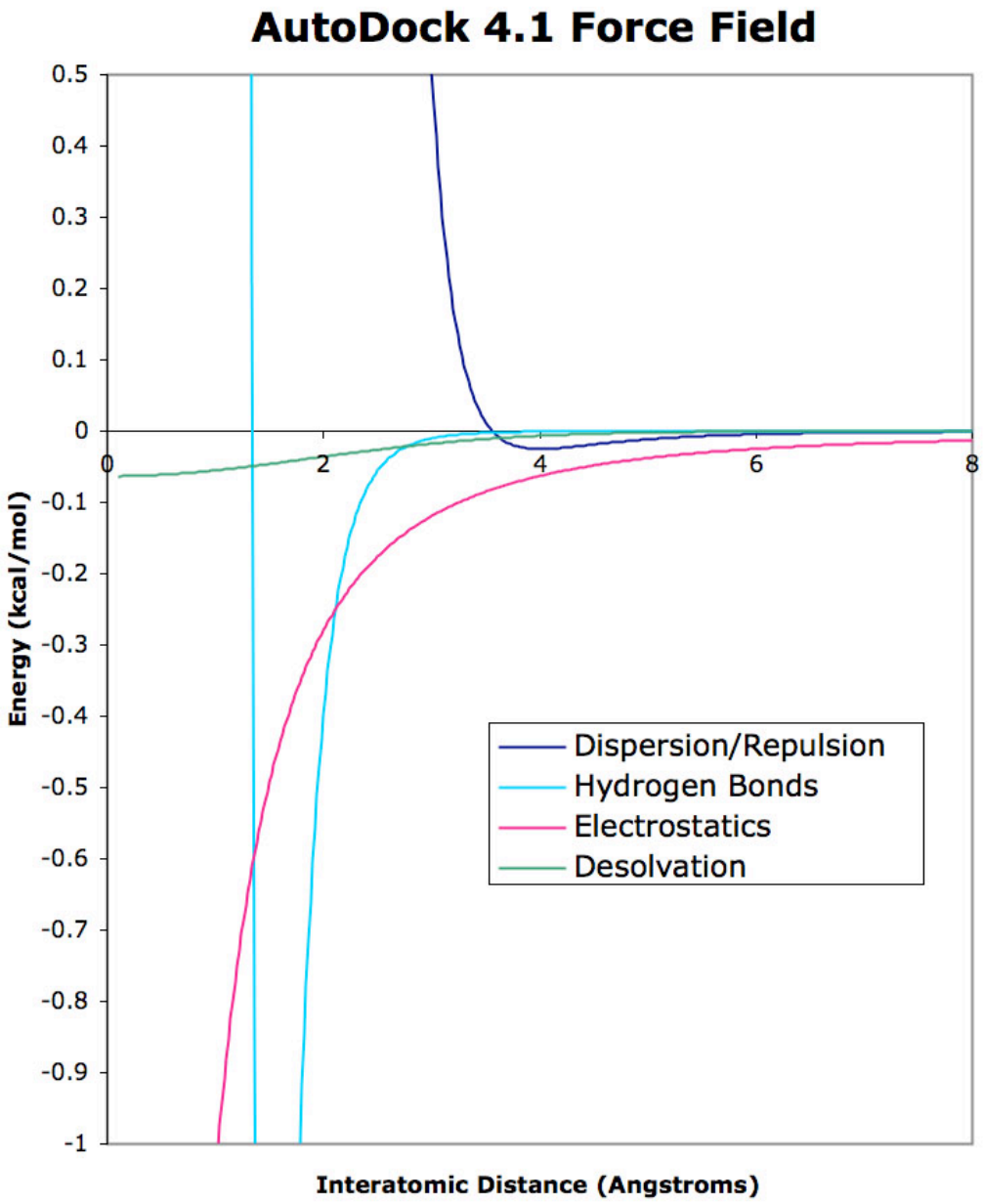
$$V = W_{vdw} \sum_{i,j} \left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} \right) + W_{hbond} \sum_{i,j} E(t) \left(\frac{C_{ij}}{r_{ij}^{12}} - \frac{D_{ij}}{r_{ij}^{10}} \right) + W_{elec} \sum_{i,j} \frac{q_i q_j}{\varepsilon(r_{ij}) r_{ij}} + W_{sol} \sum_{i,j} \left(S_i V_j + S_j V_i \right) e^{(-r_{ij}^2 / 2\sigma^2)}$$

The weighting constants W have been optimized to calibrate the empirical free energy based on a set of experimentally-determined binding constants. The first term is a typical 6/12 potential for dispersion/repulsion interactions. The parameters are based on the Amber force field. The second term is a directional H-bond term based on a 10/12 potential. The parameters C and D are assigned to give a maximal well depth of 5 kcal/mol at 1.9Å for hydrogen bonds with oxygen and nitrogen, and a well depth of 1 kcal/mol at 2.5Å for hydrogen bonds with sulfur. The function E(t) provides directionality based on the angle t from ideal h-bonding geometry. The third term is a screened Coulomb potential for electrostatics. The final term is a desolvation potential based on the volume of atoms (V) that surround a given atom and shelter it from solvent, weighted by a solvation parameter (S) and and exponential term with distance-weighting factor σ=3.5Å. For a detailed presentation of these functions, please see our published reports, included in Appendix II.

By default, AutoGrid and AutoDock use a standard set of parameters and weights for the force field. The `parameter_file` keyword may be used, however, to use custom parameter files. The format of the parameter file is described in the Appendix I.



Viewing Grids in AutoDockTools. *The protein is shown on the left in white bonds, and the grid box is shown on the right side. The blue contours surround areas in the box that are most favorable for binding of carbon atoms, and the red contours show areas that favor oxygen atoms. A ligand is shown inside the box at upper right.*



AutoDock Potentials. *Examples of the four contributions to the AutoDock force field are shown in this graph. The dispersion/repulsion potential is for interaction between two carbon atoms. The hydrogen bond potential, which extends down to a minimum of about –2 kcal/mol, is shown for an oxygen-hydrogen interaction. The electrostatic potential is shown for interaction of two oppositely-charged atoms with a full atomic charge. The desolvation potential is shown for a carbon atom, with approximately 10 atoms displacing water at each distance.*

Using AutoDock

STEP 1: Preparing Coordinates

The first step is to prepare the ligand and receptor coordinate files to include the information needed by AutoGrid and AutoDock. These coordinate files are created in an AutoDock-specific coordinate file format, termed PDBQT, which includes:

1) Polar hydrogen atoms;

2) Partial charges;

3) Atom types;

4) Information on the articulation of flexible molecules.

For a typical docking calculation, you will create a file of coordinates for the receptor, and a separate file of coordinates for the ligand. In dockings where selected amino acids in the receptor are treated as flexible, you will create a third file that includes the coordinates of the atoms in the flexible portions of the receptor.

In a typical study, the user prepares coordinate files in several steps using AutoDockTools. The first two steps may be performed using the tools in the `Edit` menu of AutoDockTools, or with other molecular modeling programs:

1) Add hydrogen atoms to the molecule.

2) Add partial charges.

Then, read the molecule into AutoDockTools using the `Ligand` (for the ligand) or `Grid` (for the receptor) menus, and create the PDBQT file:

3) Delete non-polar hydrogens and merge their charges with the carbon atoms.

4) Assign atom types, defining hydrogen bond acceptors and donors and aromatic and aliphatic carbon atoms.

5) Choose a root atom that will act as the root for the torsion tree description of flexibility.

6) Define rotatable bonds and build the torsion tree.

There are a few things to keep in mind during this process:

AutoDockTools and PMV currently use Babel to add hydrogen atoms and assign charges. Unfortunately Babel has trouble with some molecules. In those cases, hydrogen positions and charges may be assigned by the user's preferred method, *e.g.* using Reduce, InsightII, Quanta, Sybyl, AMBER or CHARMm.

In addition, most modeling systems add polar hydrogens in a default orientation, typically assuming each new torsion angle is 0° or 180°. Without some form of refinement, this can lead to spurious locations for hydrogen bonds. One option is to relax the hydrogens and perform a molecular mechanics minimization on the structure. Another is to use a program like "pol_h" which takes as input the default-added polar hydrogen structure, samples favorable locations for each movable proton, and selects the best position for each. This "intelligent" placement of movable polar hydrogens can be particularly important for tyrosines, serines and threonines.

Care should be taken when the PDB file contains disordered residues, where alternate location indicators (column 17) have been assigned. For each such atom, the user must select only one of the possible alternate locations, making sure that a locally consistent set is chosen.

Please note: coordinate preparation is the most important step in the docking simulation. The quality and accuracy of the docked results will only be as good as the quality of the starting coordinates. Be critical and carefully examine hydrogen positions, atom type assignments, partial charges, and articulation of the molecules to ensure that they make sense chemically. If you are using the Babel method within AutoDockTools to add charges and hydrogens, carefully check the results and make corrections if necessary—it often has trouble with molecules such as nucleotides.

Creating PDBQT files in AutoDockTools

Overview of AutoDockTools

AutoDockTools is a set of commands implemented within the Python Molecular Viewer (PMV). It is available at: http://autodock.scripps.edu/resources/adt.

The AutoDockTools window has several parts:

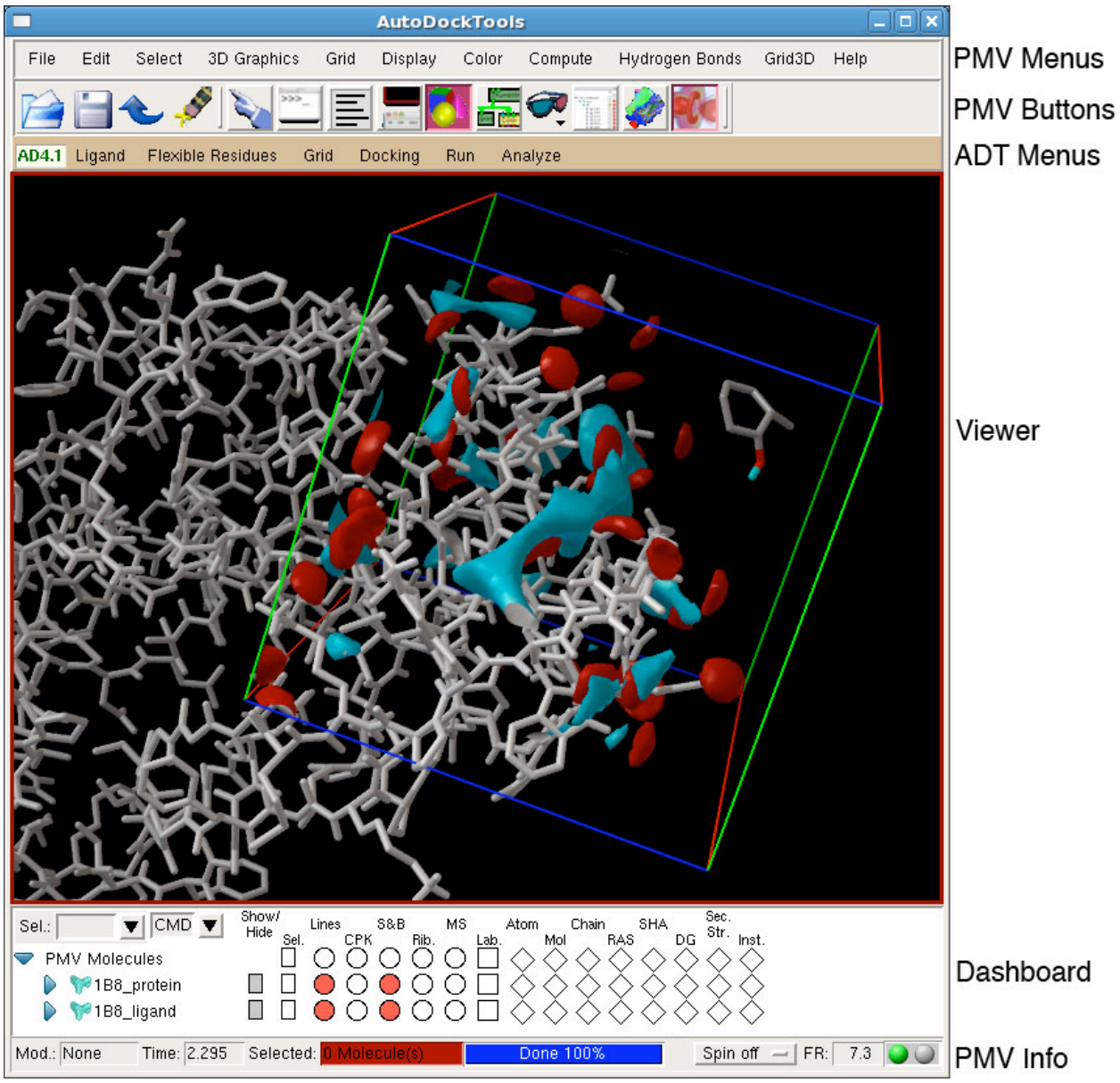
1) at the top are menus that access the general methods available in PMV. These include tools for reading and writing coordinates and images, for modifying coordinates, for selection, and for visualization.

2) a row of buttons allows quick access to the most popular tools of PMV

3) below the buttons, there are a series of menus that access the AutoDock-specific tools of AutoDockTools.

4) the 3-D molecular viewer is at the center.

5) the Dashboard, located below the viewer, allows quick selection, visualization, and coloring of molecules currently displayed in the viewer.



Hydrogen Atoms and Charges

The tools available in PMV are used to read coordinates in PDB and other formats, to add hydrogens, to select portions of the molecule, and to add partial charges. These functions are all accessed through menus at the top of the PMV window. A few useful commands will be described here—for more information on the many other functions of PMV, please see the PMV documentation.

`File>ReadMolecule:` opens a browser that allows reading of PDB coordinate files

Edit>Delete: several options for deleting entire molecules, selected sets of atoms, or hydrogen atoms.

Edit>Hydrogens>Add: options for adding all hydrogens or polar hydrogens using Babel.

Edit>Charges: options for computing Gasteiger charges for arbitrary molecules using Babel.

Ligand PDBQT Files – the "Ligand" Menu

Once ligand coordinates are created with hydrogen atoms and charges, they can be processed in the "Ligand" menu to create the ligand PDBQT file.

Ligand>Input>QuickSetup: uses defaults to create the PDBQT file. PDB files can be read from the PMV viewer or from a file, and written directly to a new PDBQT file. Please note that hydrogen atoms will not be added.

Ligand>Input>Open: reads coordinates from a file.

Ligand>Input>Choose: chooses a molecule already read into PMV.

Ligand>Input>OpenAsRigid: reads an existing PDBQT file and writes a new file with NO active torsions.

TorsionTree>ChooseRoot: manual selection of the root atom.

TorsionTree>DetectRoot: automatic detection of the root that provides the smallest largest subtree.

TorsionTree>ShowRootExpansion: for molecules with several atoms in the root, displays small spheres to show all atoms in the root, including atoms connected to each root atom by rigid bonds.

TorsionTree>ShowRootMarker: displays a sphere on the root atom.

TorsionTree>ChooseTorsions: launches an interactive browser for choosing rotatable bonds. Rotatable bonds are shown in green, and non-rotatable bonds are shown in red. Bonds that are potentially rotatable but treated as rigid, such at amide bonds and bonds that are made rigid by the user, are shown in magenta. Rotation of rotatable bonds may be switched on and off by clicking on the bonds.

TorsionTree>SetNumberOfTorsions: sets the number of rotatable bonds in the ligand by leaving the specified number of bonds as rotatable. The two options will choose the torsions that rotate either the fewest atoms in the ligand or the most atoms in the ligand.

AromaticCarbon>SetNames: clicking on atom positions will switch carbon atoms between aromatic and aliphatic. Aromatic carbons are shown in green. Click on the "Stop" button when finished.

`AromaticCarbon>AromaticityCriterion:` Sets the angular deviation from planarity that AutoDockTools uses to identify aromatic rings.

`Ligand>Output:`opens a browser to write the formatted PDBQT file.

Rigid Receptor PDBQT Files – the "Grid" Menu

For docking calculations using rigid receptor coordinates, add the hydrogen atoms and charges in PMV, then read the coordinates into AutoDockTools using the "Grid" menu.

Grid>Macromolecule>Open: launches a browser to open an existing PDBQT file.

Grid>Macromolecule>Choose: chooses a molecule that has been previously read into PMV. It will merge non-polar hydrogen atoms and charges, assign aromatic carbons, and prompt the user to write a PDBQT file.

Flexible Receptor PDBQT Files – the "FlexibleResidues" Menu

For docking calculations with selected flexibility in the receptor, add the hydrogen atoms and charges in PMV, then create two PDBQT files in AutoDockTools, one for the rigid portion of the receptor and one for the flexible atoms.

FlexibleResidues>Input>OpenMacromolecule: launches a browser to open an existing PDBQT file.

FlexibleResidues>Input>ChooseMacromolecule: chooses a molecule that has been previously read into PMV. It will merge non-polar hydrogen atoms and charges, assign aromatic carbons, and prompt the user to write a PDBQT file.

FlexibleResidues>ChooseTorsionsInCurrentlySelectedResidues: flexible residues are chosen using the tools in the PMV "Select" menu, then this option is used to assign these residues as flexible. As with the ligand, you can choose which bonds to keep rotatable by clicking on the bonds.

FlexibleResidues>Output>SaveRigidPDBQT:
FlexibleResidues>Output>SaveFlexiblePDBQT: these two commands launch a browser to write PDBQT files for the rigid portion of the receptor and the flexible portion of the receptor.

STEP 2: Running AutoGrid

AutoDock requires pre-calculated *grid maps*, one for each atom type present in the ligand being docked. This helps to make the docking calculations fast. These maps are calculated by **AutoGrid**. A grid map consists of a three dimensional lattice of regularly spaced points, surrounding (either entirely or partly) and centered on some region of interest of the macromolecule under study. This could be a protein, enzyme, antibody, DNA, RNA or even a polymer or ionic crystal. Typical grid point spacing varies from 0.2Å to 1.0Å, and the default is 0.375Å (roughly a quarter of the length of a carbon-carbon single bond). Each point within the grid map stores the potential energy of a '*probe*' atom or functional group that is due to all the atoms in the macromolecule.

AutoGrid requires an grid parameter file to specify the files and parameters used in the calculation. The grid parameter file usually has the extension ".gpf". As described below, AutoDockTools may be used to create the grid parameter file. A full description of the grid parameter file is included in Appendix I.

To run AutoGrid, the command is issued as follows:

```
% autogrid4 -p macro.gpf [-l macro.glg]
```

where '-p macro.gpf' specifies the grid parameter file, and '-l macro.glg' specifies the log file written during the grid calculation. If no log file is specified, the output is written to the terminal.

AutoGrid writes out the grid maps in ASCII form, for readability and portability; AutoDock expects ASCII format grid maps. For a description of the format of the grid map files, see Appendix I. Check the minimum and maximum energies in each grid map: these are reported at the end of the AutoGrid log file (here, it is "`macro.glg`"). Minimum van der Waals' energies and hydrogen bonding energies are typically -10 to -1 kcal/mol, while maximum van der Waals' energies are clamped at $+10^5$ kcal/mol. Electrostatic potentials tend to range from around -10^3 to $+10^3$ kcal/mol/e: if these are both 0, check to make sure that partial charges have been assigned on the macromolecule.

As well as the grid maps, AutoGrid creates two files, with the extensions '.fld', and '.xyz'. The former is a *field file* summarizing the grid maps, and the latter describes the spatial extent of the grids in Cartesian space.

Creating grid parameter files in AutoDockTools

The tools available in "grid" menu of AutoDockTools may be used to create grid parameter files.

`Grid>OpenGPF:` gets parameters from an existing grid parameter file.

Grid>Macromolecule: has options for opening an existing PDBQT file or choosing a molecule that has been read using PMV.

Grid>SetMapTypes: tools to define the atom types for the grids that will be calculated. Grids must be calculated for each type of atom in the ligand, and if flexible sidechains are used in the receptor, their atom types must also be included. The option "Directly" allows the user to input the list of atom types directly. Other options allow the user to define the atom types based on a ligand or flexible residue that has been read by PMV, or to open ligand or flexible residue PDBQT and use the atom types in these files.

Grid>SetMapTypes>SetUp CovalentMap: specifies parameters for creation of a covalent map, which may be used in specialized applications to favor binding of a given ligand atom in a single position. This is particularly useful for docking of covalent complexes between ligands and proteins. This will calculate a separate grid with atom type "Z" with a favorable Gaussian well at the coordinates given. The potential will have zero energy at the site, rising to the energy barrier height in surrounding areas.

Grid>GridBox: launches interactive commands for setting the grid dimensions and center. To enter numbers on the thumbwheel, place the cursor over the thumbwheel and type in the new value. Right clicking on the thumbwheel gives more options. IMPORTANT: when finished, use the "close saving current" option in the "File" menu on the Grid Options Panel. Options in the "Center" menu on the browser provide different methods to choose the center of the grid box.

Grid>OtherOptions: allows specification and editing of an existing parameter file.

Grid>Output: writes a new grid parameter file.

Grid>EditGPF: interactive editor for grid parameter files, which allows viewing of the lastest grid parameter file written by AutoDockTools.

STEP 3: Docking with AutoDock

AutoDock uses one of several conformational search algorithms to explore the conformational states of a flexible ligand, using the maps generated by AutoGrid to evaluate the ligand-protein interaction at each point in the docking simulation. In a typical docking, the user will dock a ligand several times, to obtain multiple docked conformations. The results may be clustered to identify similar conformations—this is described in more detail in the section on Analysis (Step 4, below).

AutoDock requires: 1) grip maps for each atom type in the ligand, calculated by AutoGrid, 2) a PDBQT file for the ligand, and 3) a docking parameter file that specifies the files and parameters for the docking calculation. AutoDockTools may be used to generate the docking parameter file, as described below, which typically has the extension ".dpf". A full description of the docking parameter file is included in Appendix I. The final docked coordinates are written into the docking log file. As described in Step 4 below, these docked conformations may be viewed using AutoDockTools, they may be written as PDBQT files using AutoDockTools, or they may be taken directly from the docking log file using a text editor.

An AutoDock calculation is started from the command line using the following command:

```
% autodock4 [-k][-i][-u][-t] -p lig.dpf [-l lig.dlg]
```

Input parameters are specified by "`-p lig.dpf`", and the log file containing the output and results from the docking is defined by "`-l lig.dlg`". This is the normal usage of AutoDock, and performs a standard docking calculation.

`-p dpf_filename`

Specifies the docking parameter file.

`-l dlg_filename`

Specifies the docking log file. If this is omitted, output will be written to the terminal and the results of the docking will not be saved.

`-k`

keep the original residue number of the input ligand PDBQT file. Normally AutoDock re-numbers the starting position to residue-number 0, and any cluster-representatives are numbered incrementally from 1, according to their rank (rank 1 is the lowest energy cluster).

`-i`

This is used to *ignore* any grid map header errors that may arise due to conflicting filenames. This overrides the header checking that is normally performed to ensure compatible grid maps are being used.

`-u, -h`

This returns a helpful message describing the command line *usage* of AutoDock.

`-t`

This instructs AutoDock to parse the PDBQ file to check the *torsion* definitions, and then stop.

`-version`

This returns a message describing the version of AutoDock being used.

Choosing a protocol for your application

AutoDock provides a number of different methods for doing the docking simulation, and different methods might be useful for different applications. This section includes some guidelines for choosing the best approach.

1) Conformation Search. AutoDock provides several methods for doing the conformation search. Currently, the Lamarckian Genetic Algorithm provides the most efficient search for general applications, and in most cases will be the technique used. It it typically effective for systems with about 10 rotatable bonds in the ligand. The Genetic Algorithm may also be run without the local search, but this is typically less efficient than the Lamarckian GA-LS combination. Simulated Annealing is also less efficient that the Lamarckian Genetic Algorithm, but it can be useful in applications where search starting from a given point is desired. Local Search may be used to optimize a molecule in its local environment.

2) Number of Evaluations. Each of the search methods include parameters for determining the amount of computational effort that will be used in the search. In the GA methods, this parameter is `ga_num_evals`, and in simulated annealing, this is `nacc` and `nrej`. The defaults given for these parameters are typically sufficient for docking systems with 10 or fewer rotatable bonds, and shorter simulations may often be used for systems with very few rotatable bonds. For complex systems with many more rotatable bonds that this, it is not generally effective to simply increase the number of evaluations. Rather, it is best to look for simpler formulations of the system, such as breaking a large ligand into two pieces and docking them separately, or freezing some rotatable bonds in likely conformations.

3) Model for the Unbound Ligand. In order to estimate a free energy of binding, AutoDock needs to estimate an energy for the unbound state of the ligand and protein. Several options are available for this. By default, AutoDock4.2 uses the assumption that the conformation of the unbound ligand and protein are the same as the conformation of the ligand and protein in the complex. Because these two conformations are the same, the total contribution of the internal energy (the interaction of atoms within the ligand or the interaction of atoms within the protein) will be zero, and reported in line 4 of the energy breakdown in the docking log file.

AutoDock4.0 used a different model, where the ligand was assumed to be in an extended state in solution, and an energy was calculated for this extended state before the docking simulation was performed. This model may be used in AutoDock4.2 by using the key word

"`unbound_model_extended`." This keyword will launch the calculation of the extended model and then will report the difference between the internal energy of the unbound model and the internal energy of the ligand when it is bound to the protein. In studies where many separate dockings are performed with the same ligand, this energy for the extended ligand may be precalculated and then used in the free energy calculation by using the key word "`unbound_model_extended_energy VALUE`."

The user may also use other methods to calculate the energy of the unbound ligand outside of AutoDock. In this case, the keyword "`unbound_energy VALUE`" may be used to set the internal energy of the unbound state to a desired value. This value will then be used in the difference between the bound and unbound states to estimate the free energy.

4) Special Cases. AutoDock4.2 includes a number of optional methods for use in specialized applications. For instance, the keyword intnbp_r_eps may be used to override the standard parameters for the internal energy calculation. This has been used to model flexible cyclic molecules, but creating a special set of atom types to close rings during a docking simulation (this method is described in more detail in a tutorial on the AutoDock WWW site). Other optional features include methods for adding torsional constraints, and options for modifying the force field and analysis.

Creating docking parameter files in AutoDockTools

The tools available in "Docking" menu of AutoDockTools may be used to create docking parameter files.

`Docking>OpenDPF:` gets parameters from an existing docking parameter file.

`Docking>Macromolecule>SetRigidFilename:`
`Docking>Macromolecule>SetFlexibleResiduesFilename:` these two commands specify the PDBQT file name that will be used for the rigid receptor, and if flexible receptor residues are used, specifies the PDBQT file name for the flexible portion of the receptor.

`Docking>Ligand>Choose`
`Docking>Ligand>Open:` These two commands allow the user to choose a ligand that is already read into ADT, or open an existing ligand PDBQT file.

`Docking>Ligand>Ligand_Parameters:` opens a panel for setting various ligand parameters, including the starting values for the translation, rotation, and torsion angles. For details, see the full description of the docking parameter file in the Appendix I.

`Docking>SearchParameters>GeneticAlgorithmParameters:`
`Docking>SearchParameters>SimulatedAnnealingParameters:`
`Docking>SearchParameters>LocalSearchParameters:` these three commands open a panel for setting the parameters used by each of the search algorithms, such as temperature

schedules in simulated annealing and mutation/crossover rates in genetic algorithms. For details of each parameter, see the full description in the Appendix.

Docking>DockingParameters: opens a panel for setting the parameters used during the docking calculation, including options for the random number generator, options for the force field, step sizes taken when generating new conformations, and output options. For details of each parameter, see the full description in the Appendix I.

Docking>OtherOptions: specifies the name of an external atomic parameter file, if used.

Docking>OutPut>LamarckianGA:
Docking>OutPut>GeneticAlgorithm:
Docking>OutPut>SimulatedAnnealing:
Docking>Output>LocalSearch: These four commands write the docking parameter file using one of the four available search methods.

Docking>Edit: interactive editor for docking parameter files, which allows viewing of the lastest docking parameter file written by AutoDockTools.

STEP 4: Evaluating the Results of a Docking

At the end of a docking simulation, AutoDock writes the coordinates for each docked conformation to the docking log file, along with information on clustering and interaction energies. AutoDockTools provides options for analyzing the information stored in the docking log file.

Information in the Docking Log

The **analysis** command in the docking parameter file causes AutoDock to perform a cluster analysis of the different docked conformations. The results of this analysis are reported as a histogram, which may be found by searching for the word "HISTOGRAM" (all in capital letters) in the docking log file. This is followed by a table of RMSD values within each cluster.

AutoDock then writes coordinates for the conformation of best predicted energy in each cluster (to write coordinates for all conformations, include the keyword **write_all** in the docking parameter file). A header for each conformation includes information on the predicted energy of binding, broken down into several components, along with information on the state variables of the conformation. The coordinates are written in a modified PDB format, with four real values appended after the x,y,z coordinates: the vdW+hbond energy of interaction of the atom, the electrostatic interaction of the atom, the partial charge, and the RMSD from the reference conformation.

Analyzing Docking Results with AutoDockTools

Options in the "Analyze" menu of AutoDockTools may be used to process and analyze the results from a docking simulation.

Analyze>Dockings>Open: opens a docking log file.

Analyze>Dockings>OpenAll: opens a set of docking log files in a directory.

Analyze>Dockings>Select: selects from a set of log files previously read into AutoDockTools.

Analyze>Dockings>Clear: clears log files that have been read into AutoDockTools.

Analyze>Dockings>ShowAsSpheres: creates a sphere at the center of mass of each docked conformation, which may be colored according to the predicted energy of interaction.

Analyze>Dockings>ShowInteractions: creates a specialized visualization to highlight interactions between the docked conformation of the ligand and the receptor. By default, the ligand is shown as ball-and-stick, surrounded by a molecular surface. The surface is colored with atomic colors in regions that contact the receptor, and gray in regions that are not in contact. Portions of the receptor that are in contact with the ligand are shown with ball-and-stick and spacefilling spheres. Hydrogen bonds are shown as a string of small spheres. A dialogue box is also launched that provides many other options for visualization.

Analyze>Macromolecule: options to open a macromolecule PDBQT file or choose a macromolecule that is already read into PMV.

Grids>Open

Grids>OpenOther: Opens a grid map file and launches the AutoDockTools grid visualizer. A dialogue box allows specification of the contour level and several rendering options. The contour level slider and input box limits the range to favorable energies. The "sampling" value is used to create coarse representations of complex maps—set to 1, it uses the actual grid spacing, set to higher values, it decimates the map to coarser grid spacing. The "Grid3D" tool is also available in the PMV menu for more advanced representation methods for grid visualization. The "OpenOther" command allows opening of grid map files that are not specified in the current docking log that is being displayed.

Conformations>Play

Conformations>PlayRankedByEnergy: Opens a window with controls for stepping through conformations as a movie. "Play" will use the order of conformations as they were found in the docking calculations, and "PlayRankedByEnergy" will order the conformations from lowest energy to highest energy. The "&" button opens a window with additional options:

`ShowInfo` opens a panel that displays information on the predicted energy of interaction, RMSD, etc.

`BuildHbonds` and `ColorbyATOM/vdW/elect/total` allow visualization of hbonds and interaction energies.

`PlayMode` and `PlayParameters` modify the parameters of the player.

`BuildCurrent` will build a new set of coordinates in the viewer for the conformation currently specified in the player. This is useful for displaying multiple conformations in the same view. `BuildAll` will build coordinates for all conformations in the player.

`WriteCurrent` will write a PDBQT file for the current conformation in the player. `WriteAll` will write separate PDBQT files for all conformations in the player. `WriteComplex` will write a PDBQT file for the current conformation of the ligand and the receptor.

Conformations>Load: Launches an interactive browser that allows selection of clustered docked conformations. Information on the predicted interaction energy is shown at the top, and individual conformations may be chosen in the bottom panel. The "rank" value gives the cluster_rank—for instance, "1_3" is the third most favorable conformation in the best cluster. Buttons at the bottom, which may be revealed by enlarging the the window, will write the current coordinates and dismiss the window.

Clusterings>Show: Tools to show an interactive histogram of clustered conformations.

Clusterings>Recluster: Reclusters docked conformations based on new tolerances. Several values may be input in the dialogue window for use in reclustering. The results may be analyzed using **Clusterings>Show**.

Clusterings>ReclusterOnSubset: Reclusters docked conformations using only a selected set of atoms. The selection is performed using the tools in the "select" menu of PMV, and then using the "save current selection as a set" option.

Appendix I: AutoDock File Formats

PDBQT Format for Coordinate Files

Extension: **`.pdbqt`**

```
"ATOM  %5d %-4s%1s%-3s %1s%4d%1s   %8.3f%8.3f%8.3f%6.2f%6.2f%4s%6.3f %2s \n",
atom_serial_num, atom_name, alt_loc, res_name, chain_id, res_num, ins_code, x, y, z,
occupancy, temp_factor, footnote, partial_charge, atom_type
(The " " symbol is used to indicate one space.)
```

The PDBQT format adds four things to standard formatted PDB files:

1) partial charges are added to each ATOM or HETATM record after the temperature factor (columns 67-76).

2) atom types (which may be one or two letters) are added to each ATOM or HETATM record after the partial charge (columns 78-79).

3) To allow flexibility in the ligand, it is necessary to assign the rotatable bonds. AutoDock can handle up to `MAX_TORS` rotatable bonds: this parameter is defined in "autodock.h", and is ordinarily set to 32. If this value is changed, AutoDock must be recompiled. *Please note that AutoDock4.2 is currently effective for systems with roughly 10 torsional degrees of freedom, and systems with more torsional flexibility may not give consistent results*. Torsions are defined in the PDBQT file using the following keywords*:*

`ROOT / ENDROOT`
`BRANCH / ENDBRANCH`

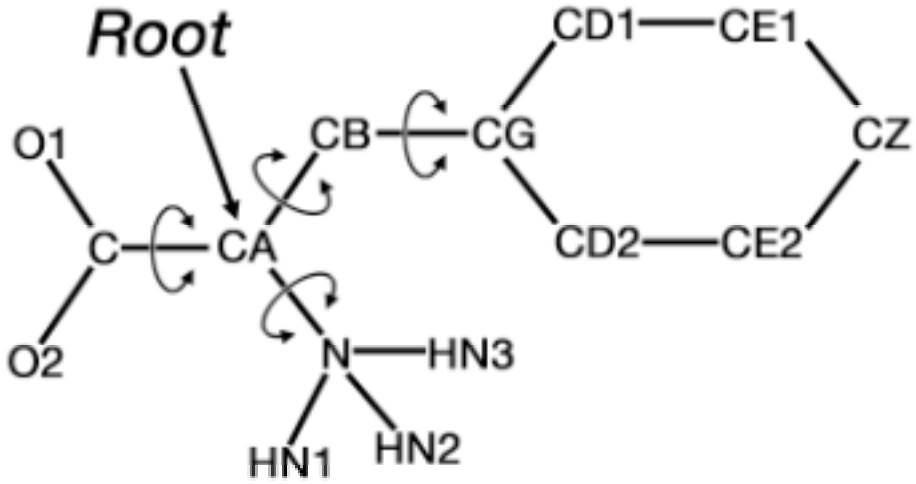
These keywords use the metaphor of a tree. See the diagram below for an example. The "root" is defined as the central portion of the ligand, from which rotatable 'branches' sprout. Branches within branches are possible. Nested rotatable bonds are rotated in order from the "leaves" to the "root". The PDBQT keywords must be carefully placed, and the order of the ATOM or HETATM records often need to be changed in order to fit into the correct branches. AutoDockTools is designed to assist the user in placing these keywords correctly, and in re-ordering the ATOM or HETATM records in the ligand PDBQT file.

4) The number of torsional degrees of freedom, which will be used to evaluate the conformational entropy, is specified using the `TORSDOF` keyword followed by the integer number of rotatable bonds. In the current AutoDock 4.2 force field, this is the total number of rotatable bonds in the ligand, including rotatable bonds in hydroxyls and other groups where only hydrogen atoms are moved, but excluding bonds that are within cycles.

Note: AutoDockTools, AutoGrid and AutoDock do not recognize PDB "CONECT" records, neither do they output them.

Sample PDBQT file

```
REMARK  4 active torsions:
REMARK  status: ('A' for Active; 'I' for Inactive)
REMARK    1  A    between atoms: N_1  and  CA_5
REMARK    2  A    between atoms: CA_5  and  CB_6
REMARK    3  A    between atoms: CA_5  and  C_13
REMARK    4  A    between atoms: CB_6  and  CG_7
ROOT
ATOM      1  CA  PHE A   1      25.412  19.595  12.578  1.00 12.96     0.287 C
ENDROOT
BRANCH   1   2
ATOM      2  N   PHE A   1      25.225  18.394  13.381  1.00 13.04    -0.065 N
ATOM      3  HN3 PHE A   1      25.856  17.643  13.100  1.00  0.00     0.275 HD
ATOM      4  HN2 PHE A   1      25.558  18.517  14.337  1.00  0.00     0.275 HD
ATOM      5  HN1 PHE A   1      24.247  18.105  13.350  1.00  0.00     0.275 HD
ENDBRANCH   1   2
BRANCH   1   6
ATOM      6  CB  PHE A   1      26.873  20.027  12.625  1.00 12.45     0.082 C
BRANCH   6   7
ATOM      7  CG  PHE A   1      27.286  20.629  13.923  1.00 12.96    -0.056 A
ATOM      8  CD2 PHE A   1      27.470  22.001  14.050  1.00 12.47     0.007 A
ATOM      9  CE2 PHE A   1      27.877  22.571  15.265  1.00 13.98     0.001 A
ATOM     10  CZ  PHE A   1      28.108  21.754  16.360  1.00 13.84     0.000 A
ATOM     11  CE1 PHE A   1      27.919  20.380  16.242  1.00 13.77     0.001 A
ATOM     12  CD1 PHE A   1      27.525  19.821  15.027  1.00 11.32     0.007 A
ENDBRANCH   6   7
ENDBRANCH   1   6
BRANCH   1  13
ATOM     13  C   PHE A   1      25.015  19.417  11.141  1.00 13.31     0.204 C
ATOM     14  O2  PHE A   1      24.659  20.534  10.507  1.00 12.12    -0.646 OA
ATOM     15  O1  PHE A   1      25.024  18.283  10.608  1.00 13.49    -0.646 OA
ENDBRANCH   1  13
TORSDOF 4
```



PDBQT Format for Flexible Receptor Sidechains

Flexible sidechains in the receptor are treated explicitly during AutoDock simulation. AutoDock requres a separate PDBQT file with atomic coordinates of the sidechains that will be treated as flexible. Atomic coordinates and branching keywords for each amino acid is placed between BEGIN_RES and END_RES records. The atom linking the amino acid to the protein, which will remain in fixed position during the simulation, is included as the root. The atoms included in the flexible residue PDBQT must be omitted from the PDBQT for the rigid portions of the receptor. For instance, in the example below, the CA atom of the PHE residue is used as the root of the flexible residue. It is included in the flexible sidechain PDBQT file, and it will be omitted from the rigid protein PDBQT file.

Sample flexible residue file, with two flexible amino acids

```
BEGIN_RES PHE A  53
REMARK  2 active torsions:
REMARK  status: ('A' for Active; 'I' for Inactive)
REMARK    1  A    between atoms: CA   and  CB
REMARK    2  A    between atoms: CB   and  CG
ROOT
ATOM      1  CA  PHE A  53      25.412  19.595  12.578  1.00 12.96     0.180 C
ENDROOT
BRANCH   1   2
ATOM      2  CB  PHE A  53      26.873  20.027  12.625  1.00 12.45     0.073 C
BRANCH   2   3
ATOM      3  CG  PHE A  53      27.286  20.629  13.923  1.00 12.96    -0.056 A
ATOM      4  CD1 PHE A  53      27.525  19.821  15.027  1.00 11.32     0.007 A
ATOM      5  CE1 PHE A  53      27.919  20.380  16.242  1.00 13.77     0.001 A
ATOM      6  CZ  PHE A  53      28.108  21.754  16.360  1.00 13.84     0.000 A
ATOM      7  CE2 PHE A  53      27.877  22.571  15.265  1.00 13.98     0.001 A
ATOM      8  CD2 PHE A  53      27.470  22.001  14.050  1.00 12.47     0.007 A
ENDBRANCH   2   3
ENDBRANCH   1   2
END_RES PHE A  53
BEGIN_RES ILE A  54
REMARK  2 active torsions:
REMARK  status: ('A' for Active; 'I' for Inactive)
REMARK    3  A    between atoms: CA   and  CB
REMARK    4  A    between atoms: CB   and  CG1
ROOT
ATOM      9  CA  ILE A  54      24.457  20.591   9.052  1.00 12.30     0.180 C
ENDROOT
BRANCH   9  10
ATOM     10  CB  ILE A  54      22.958  20.662   8.641  1.00 11.82     0.013 C
ATOM     11  CG2 ILE A  54      22.250  19.367   9.046  1.00 12.63     0.012 C
BRANCH  10  12
ATOM     12  CG1 ILE A  54      22.266  21.867   9.298  1.00 13.03     0.002 C
ATOM     13  CD1 ILE A  54      20.931  22.246   8.670  1.00 14.42     0.005 C
ENDBRANCH  10  12
ENDBRANCH   9  10
END_RES ILE A  54
```

AutoGrid Grid Parameter File: GPF

Extension: **`.gpf`**

The grid parameter file specifies an AutoGrid calculation, including the size and location of the grid, the atom types that will be used, the coordinate file for the rigid receptor, and other parameters for calculation of the grids. Unlike previous versions of AutoGrid, the pairwise atomic parameters are now read from a separate file (described below) or taken from defaults in AutoGrid.

All delimiters where needed are white spaces. Default values assigned by AutoDockTools, where applicable, are given here in square brackets [thus]. A comment must be prefixed by the "#" symbol, and can be placed at the end of a parameter line, or on a line of its own. Although ideally it should be possible to give these keywords in any order, not every possible combination has been tested, so it would be wise to stick to the following order.

AutoGrid Keywords and Commands

`parameter_file <string>`

(optional) User defined atomic parameter file (format described in the next section). By default, AutoGrid uses internal parameters.

`npts <integer> <integer> <integer>`

[40, 40, 40]

Number of *x*-, *y*- and *z*-grid points. Each *must* be an even integer number. When added to the central grid point, there will be an odd number of points in each dimension. The number of *x*-, *y*- and *z*-grid points need not be equal.

`gridfld <string>`

The grid field filename, which will be written in a format readable by AutoDock. The filename extension is '.fld'.

`spacing <float>`

[0.375 Å]

The grid point spacing, in Å. Grid points are orthogonal and uniformly spaced in AutoDock: this value is used in each dimension.

`receptor_types <string>`

[A C HD N OA SA]

Atom types present in the receptor, separated by spaces; *e.g.* for a typical protein, this will be, "A C HD N OA SA". Atom types are one or two letters, and several specialized types are used in the AutoDock4.2 forcefield, including: C (aliphatic carbon), A (aromatic carbon), HD (hydrogen that donates hydrogen bond), OA (oxygen that accepts hydrogen bond), N (nitrogen that doesn't accept hydrogen bonds), SA (sulfur that accepts hydrogen bonds).

`ligand_types <string>`

[A C HD N NA OA SA]

Atom types present in the ligand, separated by spaces, such as, "A C HD N NA OA SA".

`receptor  <string>`

Macromolecule filename, in PDBQT format.

`gridcenter  <float>  <float>  <float>`
`gridcenter  auto`

[auto]

The user can explicitly define the center of the grid maps, respectively the *x*, *y* and *z* coordinates of the center of the grid maps (units: Å, Å, Å.) Or the keyword "auto" can be given, in which case AutoGrid will center the grid maps on the center of the macromolecule.

`smooth  <float>`

[0.5 Å]

Smoothing parameter for the pairwise atomic affinity potentials (both van der Waals and hydrogen bonds). For AutoDock4, the force field has been optimized for a value of 0.5 Å.

`map  <string>`

Filename of the grid map for each ligand atom type; the extension is usually ".X.map", where "X" is the atom type. One line is included for each atom type in the **`ligand_types`** command, in the order given in that command.

`elecmap  <string>`

Filename for the electrostatic potential energy grid map to be created; filename extension '.e.map'.

`dsolvmap  <string>`

Filename for the desolvation potential energy grid map to be created; filename extension '.d.map'.

`dielectric  <float>`

[-0.1465]

Dielectric function flag: if negative, AutoGrid will use *distance-dependent* dielectric of Mehler and Solmajer; if the float is positive, AutoGrid will use this value as the dielectric constant. AutoDock4 has been calibrated to use a value of –0.1465.

Atomic Parameter File

Filename: **`AD4.2_bound.dat`**

Atomic parameters are assigned by default from values internal to AutoGrid and AutoDock, but custom parameters may be read from a file. The file includes weighting parameters for each term in the free energy function, and parameters for each atom type.

```
FE_coeff-vdW  <float>
FE_coeff-hbond  <float>
FE_coeff-estat  <float>
FE_coeff-desolv  <float>
FE_coeff-tors  <float>
```

Weighting parameters for each term in the empirical free energy force field.

```
atom_par  <string>  6*<float>  4*<integer>
```

Pairwise atomic parameters for each type of atom. Each record includes:

1. Atom type.
2. Rii = sum of the vdW radii of two like atoms (Å).
3. epsii = vdW well depth (kcal/mol)
4. vol = atomic solvation volume (Å^3)
5. Rij_hb = H-bond distance between heteroatom and hydrogen (Å)
   value is included in the heteroatom record and set to zero for hydrogens
6. epsij_hb = well depth for hydrogen bonds (kcal/mol)
7. hbond = integer indicating the type of hbond
   0, no hbond
   1, spherical H donor
   2, directional H donor
   3, spherical acceptor
   4, directional N acceptor
   5, directional O/S acceptor
8. rec_index = initialized to –1, used to hold number of atom types
9. map_index = initialized to –1, used to hold the index of the AutoGrid map
10. bond_index = used to detect bonds of different lengths, see “mdist.h” for information

Default Atomic Parameter File

```
#
#               Free Energy Coefficient
#               ------
FE_coeff_vdW    0.1662
FE_coeff_hbond  0.1209
FE_coeff_estat  0.1406
FE_coeff_desolv 0.1322
FE_coeff_tors   0.2983

#
#         Atom    Rii                             Rij_hb       rec_index
#         Type          epsii             solpar          epsij_hb     map_index
#                               vol                             hbond      bond_index
#         --      ----  -----  -------   --------  ---  ---  -  --   --  --
atom_par H      2.00  0.020   0.0000    0.00051  0.0  0.0  0  -1  -1  3  # Non H-bonding Hydrogen
atom_par HD     2.00  0.020   0.0000    0.00051  0.0  0.0  2  -1  -1  3  # Donor 1 H-bond Hydrogen
atom_par HS     2.00  0.020   0.0000    0.00051  0.0  0.0  1  -1  -1  3  # Donor S Spherical Hydrogen
atom_par C      4.00  0.150  33.5103   -0.00143  0.0  0.0  0  -1  -1  0  # Non H-bonding Aliphatic
Carbon
atom_par A      4.00  0.150  33.5103   -0.00052  0.0  0.0  0  -1  -1  0  # Non H-bonding Aromatic
Carbon
atom_par N      3.50  0.160  22.4493   -0.00162  0.0  0.0  0  -1  -1  1  # Non H-bonding Nitrogen
atom_par NA     3.50  0.160  22.4493   -0.00162  1.9  5.0  4  -1  -1  1  # Acceptor 1 H-bond Nitrogen
atom_par NS     3.50  0.160  22.4493   -0.00162  1.9  5.0  3  -1  -1  1  # Acceptor S Spherical
Nitrogen
atom_par OA     3.20  0.200  17.1573   -0.00251  1.9  5.0  5  -1  -1  2  # Acceptor 2 H-bonds Oxygen
atom_par OS     3.20  0.200  17.1573   -0.00251  1.9  5.0  3  -1  -1  2  # Acceptor S Spherical
Oxygen
atom_par F      3.09  0.080  15.4480   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Fluorine
atom_par Mg     1.30  0.875   1.5600   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Magnesium
atom_par MG     1.30  0.875   1.5600   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Magnesium
atom_par P      4.20  0.200  38.7924   -0.00110  0.0  0.0  0  -1  -1  5  # Non H-bonding Phosphorus
atom_par SA     4.00  0.200  33.5103   -0.00214  2.5  1.0  5  -1  -1  6  # Acceptor 2 H-bonds Sulphur
atom_par S      4.00  0.200  33.5103   -0.00214  0.0  0.0  0  -1  -1  6  # Non H-bonding Sulphur
atom_par Cl     4.09  0.276  35.8235   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Chlorine
atom_par CL     4.09  0.276  35.8235   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding
Chlorineatom_par Ca     1.98  0.550   2.7700  -0.00110  0.0  0.0  0  -1  -1  4 # Non H-bonding
Calcium
atom_par CA     1.98  0.550   2.7700   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Calcium
atom_par Mn     1.30  0.875   2.1400   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Manganese
atom_par MN     1.30  0.875   2.1400   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Manganese
atom_par Fe     1.30  0.010   1.8400   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Iron
atom_par FE     1.30  0.010   1.8400   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Iron
atom_par Zn     1.48  0.550   1.7000   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Zinc
atom_par ZN     1.48  0.550   1.7000   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Zinc
atom_par Br     4.33  0.389  42.5661   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Bromine
atom_par BR     4.33  0.389  42.5661   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Bromine
atom_par I      4.72  0.550  55.0585   -0.00110  0.0  0.0  0  -1  -1  4  # Non H-bonding Iodine
atom_par Z      4.00  0.150  33.5103   -0.00143  0.0  0.0  0  -1  -1  0  # Non H-bonding covalent map
atom_par G      4.00  0.150  33.5103   -0.00143  0.0  0.0  0  -1  -1  0  # Ring closure Glue
Aliphatic Carbon  # SF
atom_par GA     4.00  0.150  33.5103   -0.00052  0.0  0.0  0  -1  -1  0  # Ring closure Glue Aromatic
Carbon   # SF
atom_par J      4.00  0.150  33.5103   -0.00143  0.0  0.0  0  -1  -1  0  # Ring closure Glue
Aliphatic Carbon  # SF
atom_par Q      4.00  0.150  33.5103   -0.00143  0.0  0.0  0  -1  -1  0  # Ring closure Glue
Aliphatic Carbon  # SF
```

Grid Map File

Extension: **`.map`**

The first six lines of each grid map hold header information which describe the spatial features of the maps and the files used or created. These headers are checked by AutoDock to ensure that they are appropriate for the requested docking. The remainder of the file contains grid point energies, written as floating point numbers, one per line. They are ordered according to the nested loops: z(y(x)), so x is changing fastest.

Sample Grid Map File

```
GRID_PARAMETER_FILE vac1.nbc.gpf
GRID_DATA_FILE 4phv.nbc_maps.fld
MACROMOLECULE 4phv.new.pdbq
SPACING 0.375
NELEMENTS 50 50 80
CENTER -0.026 4.353 -0.038
125.095596
123.634560
116.724602
108.233879
:
```

Grid Map Field File

Extension: **`.maps.fld`**

This is essentially two files in one. It is both an AVS field file, which may be read by a number of scientific visualization programs, and and AutoDock input file with AutoDock-specific information in the comments at the head of the file. AutoDock uses this file to check that all the maps it reads in are compatible. For example, in this file, the grid spacing is 0.375 Angstroms, there are 60 intervals in each dimension (and 61 actual grid points), the grid is centered near (16., 39., 1.), it was calculated around the macromolecule '`protein.pdbqt`', and the AutoGrid parameter file used to create this and the maps was '`protein.gpf`'. This file also points to a second file, '`protein.maps.xyz`', which contains the minimum and maximum extents of the grid box in each dimension, *x, y,* and *z*. Finally, it lists the grid map files that were calculated by AutoGrid, here '`protein.A.map`', '`protein.C.map`', etc.

Sample Grid Map Field File

```
# AVS field file
#
# AutoDock Atomic Affinity and Electrostatic Grids
#
# Created by autogrid4.
#
#SPACING 0.375
#NELEMENTS 60 60 60
#CENTER 16.000 39.000 1.000
#MACROMOLECULE protein.pdbqt
#GRID_PARAMETER_FILE protein.gpf
#
ndim=3                # number of dimensions in the field
dim1=61               # number of x-elements
dim2=61               # number of y-elements
dim3=61               # number of z-elements
nspace=3              # number of physical coordinates per point
veclen=8              # number of affinity values at each point
data=float            # data type (byte, integer, float, double)
field=uniform         # field type (uniform, rectilinear, irregular)
coord 1 file=protein.maps.xyz filetype=ascii offset=0
coord 2 file=protein.maps.xyz filetype=ascii offset=2
coord 3 file=protein.maps.xyz filetype=ascii offset=4
label=A-affinity      # component label for variable 1
label=C-affinity      # component label for variable 2
label=HD-affinity     # component label for variable 3
label=N-affinity      # component label for variable 4
label=OA-affinity     # component label for variable 5
label=SA-affinity     # component label for variable 6
label=Electrostatics  # component label for variable 7
label=Desolvation     # component label for variable 8
#
# location of affinity grid files and how to read them
#
variable 1 file=protein.A.map filetype=ascii skip=6
variable 2 file=protein.C.map filetype=ascii skip=6
variable 3 file=protein.HD.map filetype=ascii skip=6
variable 4 file=protein.N.map filetype=ascii skip=6
variable 5 file=protein.OA.map filetype=ascii skip=6
variable 6 file=protein.SA.map filetype=ascii skip=6
variable 7 file=protein.e.map filetype=ascii skip=6
variable 8 file=protein.d.map filetype=ascii skip=6
```

AutoDock Docking Parameter File: DPF

Extension: **`.dpf`**

The docking parameter file specifies the files and parameters for an AutoDock calculation, including the map files that will be used for the docking, the ligand coordinate files, and parameters for the search. Unlike previous versions of AutoDock, the pairwise atomic parameters used for the internal energy calculation may now read from a separate file (described above), or taken from defaults in AutoDock.

All delimiters where needed are white spaces. Default values assigned by AutoDockTools, where applicable, are given here in square brackets [thus]. A comment must be prefixed by the "#" symbol, and can be placed at the end of a parameter line, or on a line of its own. Although ideally it should be possible to give these keywords in any order, not every possible combination has been tested, so it would be wise to stick to the following order.

Parameter to set the amount of output

`outlev <integer>`

[1]
Diagnostic output level. For SA (simulated annealing): 0 = no output, 1 = minimal output, 2 = full state output at end of each cycle; 3 = detailed output for each step. For GA and GA-LS (genetic algorithm-local search): 0 = minimal output, 1 = write minimum, mean, and maximum of each state variable at the end of every generation, 2 = full output at every generation.

Atomic parameters for pairwise energy evaluation

`parameter_file <string>`

(Optional) Atomic parameter file used for pairwise energy evaluation in internal energies and interactions between ligand and flexible sidechains. If this is not given, AutoDock uses default parameters identical to values in the file AD4.2_bound.dat.

`intelec`

(Optional) If this keyword is included, internal ligand electrostatic energies will be calculated; the products of the partial charges in each non-bonded atom pair are pre-calculated, and output. Note that this is only relevant for flexible ligands.

`intnbp_r_eps <float><float><integer><integer><string><string>`

(Optional) This optional keyword allows the user to manually override the internal energy potential for a given class of interactions. The parameters are: r_{eq}, ε, n, m, and the two atom types, where r_{eq} is the equilibribium distance for the bottom of the energetic well, ε is the depth of the well, n and m are the coefficients. For instance, the command "intnbp_r_eps 1.5 10. 12 6 OA FE" will set up a potential with well depth of 10 kcal/mol at a distance of 1.5 Å for interaction between oxygen and iron atoms. The potential V(r) is calculated with the expression:

$$V(r)=C_n/r^n - C_m/r^m$$

$$C_n = m/(n-m) * \varepsilon * r_{eq}^{\ n}$$

$C_m = n/(n-m) * \varepsilon * r_{eq}^{m}$

A special atom type "G" has been created for using this feature for ring closure simulations. Please see the tutorial for more information.

Command to set the seed for the random number generator

seed <long_integer>
seed time
seed pid
seed <long_integer> <long_integer>
seed time <long_integer>
seed <long_integer> time
seed time pid
seed pid <long_integer>
seed <long_integer> pid
seed pid time

AutoDock includes two random number generator libraries, one uses the intrinsic function available in C, and the second is the portable library from the University of Texas Biomedical School. If the user gives just one argument to "`seed`", then AutoDock will use the system's implementation of the random number generator and corresponding system seed call. On most platforms, these are "drand48" and "srand48". The UTBS library, however, requires two seed values. Giving two arguments to "seed" tells AutoDock to use the platform-independent library for random number generation. The UTBS library is required for the genetic algorithm and the Solis and Wets routines, so for these, include two seed values. It cannot be used with the simulated annealing routine, so for simulated annealing, use just one seed parameter.

The random-number generator for each docking job can be 'seeded' with either a user-defined, a time-dependent, or process-ID-dependent seed. If using two two seeds, they can be any combination of explicit long integers, the keyword "`time`" or the keyword "`pid`". The keyword, "`time`" sets the seed based on the current time, and"pid" sets the seed based on the UNIX process ID of the currently executing AutoDock process.

Parameters defining the grid maps to be used

ligand_types <string>

Atom names for all atom types present in ligand using the same blank-separated, one or two letter atom types used in AutoGrid.

fld <string>

Grid data field file created by AutoGrid (must have the extension ".fld").

map <file name>

Filename for the **AutoGrid** affinity grid maps. This keyword plus filename must be repeated for all atom types in the order specifed by the "`ligand_types`" command. In all map files a 6-line header is required, and energies must be ordered according to the nested loops z(y(x)).

elecmap <file name>

Filename for the electrostatics grid map. 6-line header required, and energies must be ordered according to the nested loops z(y(x)).

desolvmap <file name>

Filename for the desolvation grid map. 6-line header required, and energies must be ordered according to the nested loops z(y(x)).

Parameters defining the state of the unbound ligand

(optional) **unbound_energy <float>**

Sets the internal energy of of the unbound state to the value.

(optional) **unbound_model_extended**

Launches a calculation to find an extended conformation of the ligand, then uses this conformation to calculate the internal energy of of the unbound state. The AutoDock4.0 keyword "`compute_unbound_extended`" will perform the same process.

(optional) **unbound_model_extended_energy <float>**

Sets the internal energy of of the unbound state to the value. This also sets the default atomic parameters used for pairwise energy evaluate to be appropriate for the extended unbound model.

Parameters defining the ligand and its initial state

move <file name>

Filename for the PDBQT coordinate file of the ligand to be docked.

about <float> <float> <float>

[0.0 0.0 0.0]

Use this keyword to specify the center of the ligand, *about* which rotations will be made. (The coordinate frame of reference is that of the ligand PDBQT file.) Usually the rotation center of the ligand is the mean *x,y,z*-coordinates of the molecule. Units: Å, Å, Å.

tran0 <float> <float> <float>
tran0 random

[random]

Initial coordinates for the center of the ligand, in the same frame of reference as the receptor grid maps. Every docking simulation specified in the docking parameter file starts the ligand from this location.

Alternatively, the user can just give the keyword "random" and AutoDock will pick random initial coordinates instead.

The user *must* specify the absolute *starting* coordinates for the ligand, used to start each run. The user should ensure that the ligand, when translated to these coordinates, still fits within the volume of the grid maps. If there are some atoms which lie outside the grid volume, then AutoDock will automatically correct this, until the ligand is pulled completely within the volume of the grids. (This is necessary in order to obtain complete information about the energy of the initial state of the

system.) The user will be notified of any such changes to the initial translation by AutoDock. (Units: Å, Å, Å.)

quat0 <float> <float> <float> <float>
quat0 random

[random]
Respectively: Q_x, Q_y, Q_z, Q_Θ. Initial axis-angle for orientation (applied to ligand) - Q_x, Q_y, Q_z define the unit vector of the direction of rigid body rotation, and Q_Θ defines the angle of rotation about this unit vector, in ° . (Units: none,none,none, °.)

Alternatively, the user can just give the keyword "random" and AutoDock will pick a random unit vector and a random rotation (between 0° and 360°) about this unit vector. Each docking simulation specified in the docking parameter file will begin at this same random rigid body rotation.

dihe0 <float> ...
dihe0 random

[random]

Initial **relative** dihedral angles; there must be a floating point number specified on this line for each rotatable bond in the PDBQT file. Each value specified here will be added to the corresponding torsion angle in the input PDBQT file, at the start of each run. Torsion angles are only specified by two atoms, so the definition of rotations is relative to the input conformation of the ligand, not an absolute conformation. Units: °.

Parameters defining ligand step sizes for simulated annealing calculations

tstep <float>
tstep <float> <float>

[2.0 Å]
The first form, with one argument, defines the maximum translation jump for the first cycle that the ligand may make in one simulated annealing step. When "trnrf" is less than 1, the reduction factor is multiplied with the tstep at the end of each cycle, to give the new value for the next cycle. The second form allows the user to specify the value for the first cycle and the last cycle: AutoDock then calculates the reduction factor that satisfies these constraints. Units: Å.

qstep <float>

[50.0°]
Maximum angular step size for the orientational component. Units: °.

dstep <float>
[50.0°]

Maximum dihedral (torsion) step size. Units: °.

Parameters defining optional ligand torsion constraints

barrier <float>

[10000.0]
(Optional) This defines the energy-barrier height applied to constrained torsions. When the torsion is at a preferred angle, there is no torsion penalty: this torsion's energy is zero. If the torsion angle

falls within a disallowed zone, however, it can contribute up to the full barrier energy. Since the torsion-energy profiles are stored internally as arrays of type 'unsigned short', only positive integers between 0 and 65535 are allowed.

`gausstorcon <integer> <float> <float>`

(Optional) Adds a constraint to a torsion. The torsion number is identified by an integer. This identifier comes from the list at the top of the AutoDockTools-generated input ligand PDBQT file (on the REMARK lines). An energy profile will be calculated for this torsion. An inverted Gaussian is added for each new constraint. To completely specify each Gaussian, two floating point numbers are needed: the *preferred angle* and the *half-width* respectively (both in degrees). Note that the preferred angle should be specified in the range -180° to +180°; numbers outside this range will be wrapped back into this range. This angle, χ, is ***relative*** to the original torsion angle in the input structure. The *half-width* is the difference between the two angles at which the energy is half the barrier. The smaller the half-width, the tighter the constraint.

If you wish to constrain to absolute-valued torsion angles, it will be necessary to zero the initial torsion angles in the ligand. The problem arises from the ambiguous 2-atom definition of the rotatable bond *B-C*. To identify a torsion angle unambiguously, 4 atoms must be specified: *A-B-C-D*:

The sign convention for torsion angles which we use is anti-clockwise (counter-clockwise) are positive angles, clockwise negative. In the above diagram, looking down the bond *B-C*, the dihedral angle *A-B-C-D* would be positive.

There is no limit to the number of constraints that can be added to a given torsion. Each new torsion-constraint energy profile is combined with the pre-existing one by selecting the minimum energy of either the new or the existing profiles.

Please note that in our tests, torsion constrains are highly inefficient, and are only effective when used in systems with few degrees of freedom in the ligand, and only a few torsion constraints.

`showtorpen`

(Optional) (Use only with "gausstorcon") This switches on the storage and subsequent output of torsion energies. During each energy evaluation, the penalty energy for each constrained torsion, as specified by the "gausstorcon" command, will be stored in an array. At the end of each run, the final docked conformation's state variables are output, but with this command, the penalty energy for each torsion will be printed alongside its torsion angle.

Parameters for cluster analysis of docked conformations

`rmstol <float>`

[2.0Å]
When more than one run is carried out in a given job, cluster analysis or 'structure binning' will be performed, based on all-atom root mean square deviation (RMSD), ranking the resulting families of docked conformations in order of increasing energy. The lowest energy representative from each cluster is written in PDBQT format to the log file. To keep the ligand's residue number in the input PDBQT file, use the '**`-k`**' flag; otherwise the clustered conformations are numbered incrementally from 1. (Units: Å).

rmsref <filename>

(Optional) If included, the RMSD of the docked conformations will be calculated with respect to the coordinates in the PDB or PDBQT file specified here. This is useful when the experimentally determined complex conformation of the ligand is known. The order of the atoms in this file must match that in the input PDBQT file given by the **move** command. These values of RMSD will be output in the last column of the final PDBQT records, after the clustering has been performed. If this keyword is not included, the RMSD is calculated based on the starting position of the ligand.

rmsnosym

(Optional) The default method for structure binning allows for atom similarity, as in a tertiary-butyl which can be rotated by +/-120°, but in other cases it may be desirable to bypass this similar atom type checking and calculate the RMSD on a one-for-one basis. The symmetry checking algorithm scans all atoms in the reference structure, and selects the nearest atom of identical atom type to be added to the sum of squares of distances. This works well when the two conformations are very similar, but this assumption breaks down when the two conformations are translated significantly. Symmetry checking can be turned off using the **rmsnosym** command; omit this command if you still want symmetry checking.

rmsatoms all

(Optional) If this keyword is included, RMSD calculation will be performed using both ligand and flexible receptor sidechain atoms. If an "rmsref" file is specified, it must include both ligand and flexible receptor atom coordinates.

Parameter for energies of atoms outside the grid

extnrg <float>

[1000.]
External grid energy assigned to any atoms that stray outside the volume of the grid during a docking. Units: kcal/mol.

Parameter for calculating energy of a ligand

epdb

This keyword will report the energy of the ligand included in the "move" record. This command may be used to calculate the energy of a particular ligand conformation without performing a docking calculation.

Parameters for simulated annealing searches

e0max <float> <positive_integer>

[0., 10000]
This keyword stipulates that the ligand's initial state cannot have an energy greater than the first value, nor can there be more than the second value's number of retries. Typical energy values range from 0 to 1000 kcal/mol. If the initial energy exceeds this value, a new random state is generated and tested. This process is iterated until the condition is satisfied. This can be particularly useful in preventing runs starting in exceptionally high energy regions. In such cases, the ligand can get trapped because it is unable to take a long enough translational jump. In those grids were the ligand

is small enough to fit into the low energy regions with ease, there will not be many iterations before a favorable location is found. But in highly constrained grids, with large ligands, this initialization loop may run almost indefinitely.

rt0 <float>

[500. cal/mol].
Initial "annealing temperature"; this is actually the absolute temperature multiplied by the gas constant R. R = 8.314 J mol^{-1} K^{-1} = 1.987 cal mol^{-1} K^{-1}. (Units: cal mol^{-1}.)

rtrf <float>

Annealing temperature reduction factor, g [0.95 $cycle^{-1}$]. At the end of each cycle, the annealing temperature is multiplied by this factor, to give that of the next cycle. This must be positive but < 1 in order to cool the system. Gradual cooling is recommended, so as to avoid "*simulated quenching*", which tends to trap systems into local minima.

linear_schedule
schedule_linear
linsched
schedlin

These keywords are all synonymous, and instruct AutoDock to use a linear or *arithmetic* temperature reduction schedule during *Monte Carlo* simulated annealing. Unless this keyword is given, a *geometric* reduction schedule is used, according to the **rtrf** parameter just described. If the linear schedule is requested, then any **rtrf** parameters will be ignored. The first simulated annealing cycle is carried out at the annealing temperature **rt0**. At the end of each cycle, the temperature is reduced by (**rt0/cycles**). The advantage of the linear schedule is that the system samples evenly across the temperature axis, which is vital in entropic calculations. Geometric temperature reduction schedules on the other hand, under-sample high temperatures and over- sample low temperatures.

runs <integer>

[10]
Number of automated docking runs.

cycles <integer>

[50]
Number of temperature reduction cycles.

accs <integer>

[100]
Maximum number of accepted steps per cycle.

rejs <integer>

[100]
Maximum number of rejected steps per cycle.

select <character>

[m]
State selection flag. This character can be either **m** for the *minimum* state, or **l** for the *last* state found during each cycle, to begin the following cycle.

trnrf <float>

[1.0]
Per-cycle reduction factor for translation steps.

quarf <float>

[1.0]
Per-cycle reduction factor for orientation steps.

dihrf <float>

Per-cycle reduction factor for torsional dihedral steps [1.].

Parameters for genetic algorithm, Lamarckian GA and evolutionary programming searches

ga_pop_size <positive_integer>

[150]
This is the number of individuals in the population. Each individual is a coupling of a genotype and its associated phenotype. Typical values range from 50 to 200.

ga_num_evals <positive_integer>

[2500000]
This is the maximum number of energy evaluations performed during each GA calculation.

ga_num_generations <positive_integer>

[27000]
This is the maximum number of generations simulated during each GA or LGA calculation.

ga_elitism <integer>

[1]
This is used in the selection mechanism of the GA. This is the number of top individuals that are guaranteed to survive into the next generation.

ga_mutation_rate <float>

[0.02]
This is a floating point number from 0 to 1, representing the probability that a particular gene is mutated. This parameter is typically small.

ga_crossover_rate <float>

[0.80]
This is a floating point number from 0 to 1 denoting the crossover rate. Crossover rate is the expected number of pairs in the population that will exchange genetic material. Setting this value to 0 turns the GA into the evolutionary programming (EP) method, but EP would probably require a concomitant increase in the ga_mutation_rate in order to be effective.

ga_window_size <positive_integer>

[10]
This is the number of preceding generations to take into consideration when deciding the threshold for the worst individual in the current population.

ga_cauchy_alpha <float>

[0.0]

ga_cauchy_beta <float>

[1.0]
These are floating point parameters used in the mutation of real number genes. They correspond to the alpha and beta parameters in a Cauchy distribution. Alpha roughly corresponds to the mean, and beta to something like the variance of the distribution. It should be noted, though, that the Cauchy distribution doesn't have finite variance. For the mutation of a real valued gene, a Cauchy deviate is generated and then added to the original value.

Genetic algorithm parameters

set_ga

This command sets the global optimizer to be a genetic algorithm [GA]. This is required to perform a GA search. This passes any 'ga_' parameters specified **before** this line to the global optimizer object. If this command is omitted, or it is given before the 'ga_' parameters, your choices will not take effect, and the default values for the optimizer will be used.

To use the traditional (non-Lamarckian) genetic algorithm, do not specify the local search parameters, and do not use the "set_sw1" or "set_psw1" commands.

To use the **Lamarckian genetic algorithm**, you must also specify the parameters for local search, and then issue either the 'set_sw1' or 'set_psw1' command. The 'set_sw1' command uses the strict Solis and Wets local search algorithm, while 'set_psw1'uses the pseudo-Solis and Wets algorithm (see below).

Parameters for local search

sw_max_its <positive_integer>

[300]
This is the maximum number of iterations that the local search procedure apply to the phenotype of any given individual.

sw_max_succ <positive_integer>

[4]
This is the number of successes in a row before a change is made to the "rho" parameter in Solis & Wets algorithms. This is an unsigned integer and is typically around four.

sw_max_fail <positive_integer>

[4]
This is the number of failures in a row before Solis & Wets algorithms adjust "rho." This is an unsigned integer and is usually around four.

sw_rho <float>

[1.0]
This is a parameter of the Solis & Wets algorithms. It defines the initial variance, and specifies the size of the local space to sample.

sw_lb_rho <float>

[0.01]
This is the lower bound on rho, the variance for making changes to genes (*i.e.* translations, orientation and torsions). rho can never be modified to a value smaller than "sw_lb_rho".

ls_search_freq <float>

[0.06]

This is the probability of any particular phenotype being subjected to local search.

Commands to choose and set the local search method

Both of these commands, 'set_sw1' and 'set_psw1', pass any 'sw_' parameters set before this line to the local searcher. If you forget to use this command, or give it before the 'sw_' keywords, your choices will not take effect, and the default values for the optimizer will be used. Currently, the psw1 method has shown the best performance and is used as the default.

set_sw1

Instructs AutoDock to use the classical Solis and Wets local searcher, using the method of uniform variances for changes in translations, orientations, and torsions.

set_psw1

Instructs AutoDock to use the pseudo-Solis and Wets local searcher. This method maintains the relative proportions of variances for the translations in Å and the rotations in radians. These are typically 0.2 Å and 0.087 radians to start with, so the variance for translations will always be about 2.3 times larger than that for the rotations (*i.e.* orientation and torsions).

Commands to specify the search method

simanneal

This command instructs AutoDock to do the specifed number of docking runs using the simulated annealing (SA) search engine. This uses the value set by the "runs" keyword as the number of SA docking runs to carry out. All relevant parameters for the simulated annealing job must be set first. These are indicated above by **[SA]** in each keyword description.

do_local_only <integer>

[50]

This keyword instructs AutoDock to carry out only the local search of a global-local search; the genetic algorithm parameters are ignored, with the exception of the population size. This is an ideal way of carrying out a minimization using the same force field as is used during a docking calculation. The "ga_run" keyword should not be given. The number after the keyword determines how many local search simulations will be performed.

do_global_only <integer>

[50]

This keyword instructs AutoDock to carry out dockings using only a global search, i.e. the traditional genetic algorithm. The local search parameters are ignored. The "ga_run" keyword should not be given. The number after the keyword determines how many dockings will be performed.

ga_run <integer>

[10]

This command invokes the Lamarckian genetic algorithm search engine, and performs the requested number of dockings. All appropriate parameters must be set first: these are listed above by "ga_".

Command to perform clustering of docked conformations

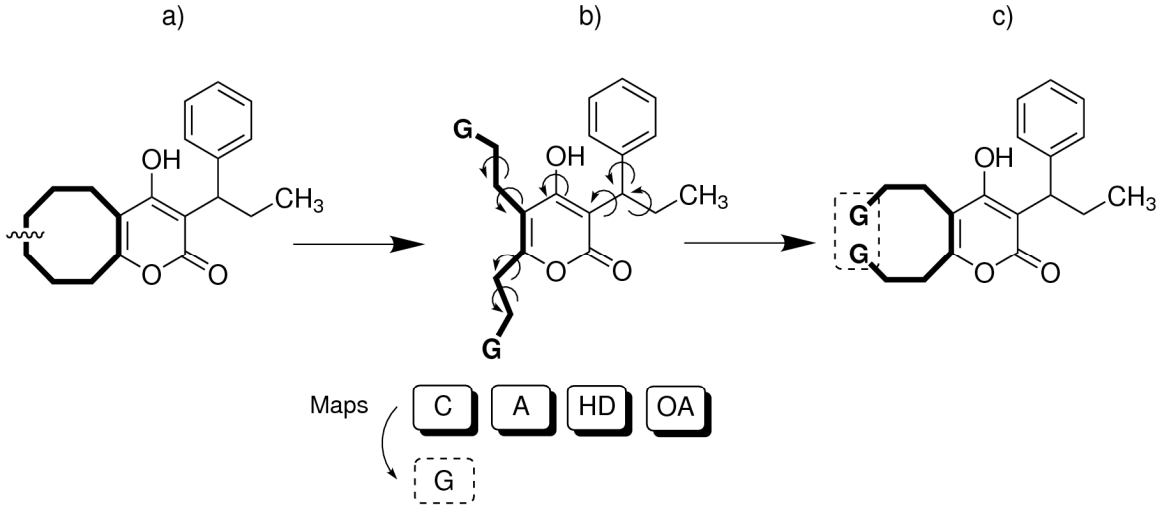
analysis

This performs a cluster analysis on results of a docking, and writes the results to the log file. The docked conformations are sorted in order of increasing energy, then compared by root mean square deviation. A histogram is printed showing the number in each cluster, and if more than one member, the cluster's mean energy. Furthermore, a table is printed to the docking log file of cluster rmsd and reference rmsd values.

Appendix II: Docking Flexible Rings with AutoDock

1. Introduction

AutoDock is not able to manage directly the flexibility associated with bonds in cyclic molecules, which leads to cyclic portions of the ligands to be considered as rigid. Different approaches can be used to dock macrocyclic molecules, like identifying one or more low energy conformations and docking them as different ligands, but generating them and docking them separately can be a time-consuming task. As an alternative, an indirect method may be used to manage the ring as a fully flexible entity and use the AutoDock GA to explore its flexibility. The method was initially developed for version 3.05, and now is implemented in version 4.2. The protocol converts the cyclic ligand into its corresponding acyclic form by removing a bond, and then docks the fully flexible molecule in the open form. A special atom type definition allows AutoDock to restore the original cycle structure during the calculation while exploring the cycle conformations with the GA. The protocol can be subdivided in three main steps:

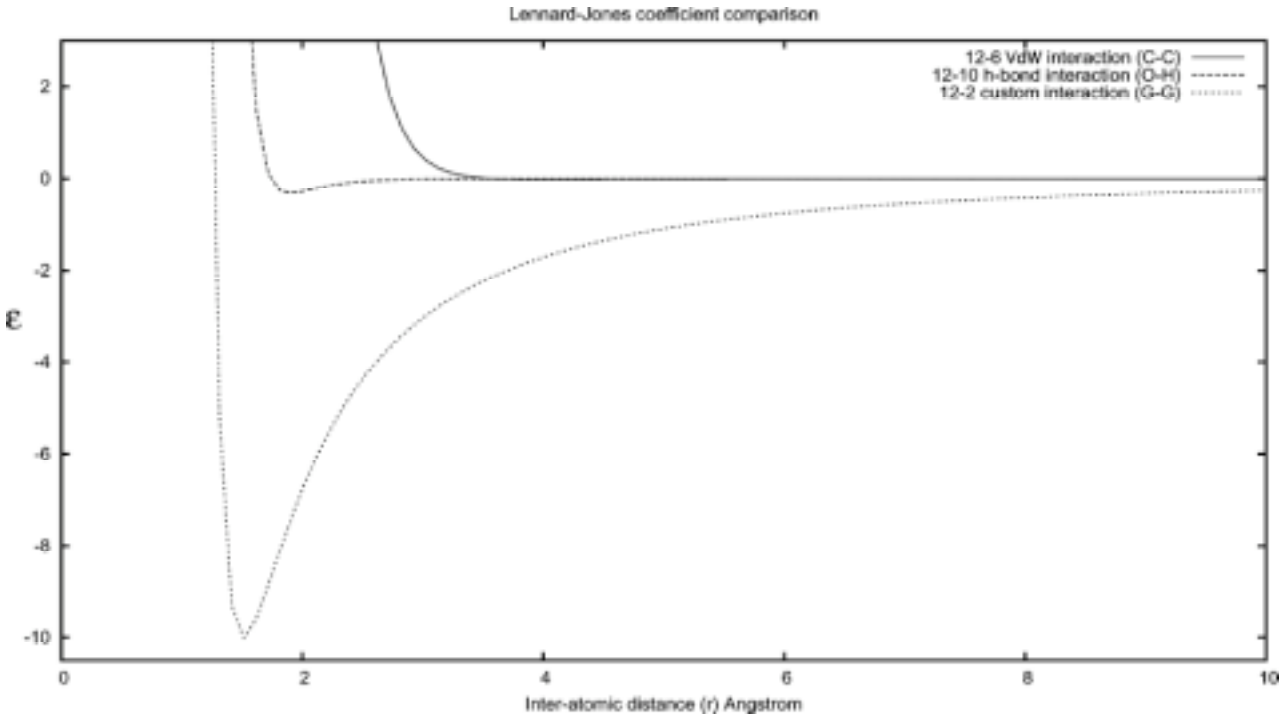


RING OPENING (a): by removing a bond, the ring is opened and the ligand is transformed to an acyclic form.

LIGAND PRE-PROCESSING (b): the ligand is processed following the standard AutoDockTools protocol, but the edge atoms are replaced with G atoms.

DOCKING AND RING CLOSURE (c): the ligand is docked applying a 12-2 pseudo-Lennard-Jones potential to the G-atoms that restore the cyclic structure.

To restore the closed ring geometry a custom long range pseudo-Lennard-Jones 12-2 potential is applied to these atoms during the docking calculation. This potential is effective at long range distances and guarantees the ring closure even with large cycles.



Ring closure parameters. *Comparison between standard 12-6 van der Waals, 12-10 hydrogen bond and 12-2 pseudo-Lennard-Jones potentials, before the AutoDock smoothing function is applied.*

No extra maps are calculated for the G atoms because, for sake of evaluation of ligand-protein interaction, they are considered as normal carbon atoms. Then, C maps are used in place. During the docking process, the potential guides the edge atoms next to each other resulting in an effective ring closure, while allowing the GA algorithm to explore the ring conformations.

2. Flexible rings

Opening the ring

To convert the molecule into the acyclic form, the bond to be disrupted must be identified. The way the acyclic form is obtained influences the subsequent the ring closure. The following guidelines may help to choose which bond to remove while keeping the calculation simple and improving the quality of the final results:

Keep number rotatable bonds low

The ring opening can dramatically increase the total number of rotatable bonds, requiring longer calculation times. Therefore, when less flexible or partially rigid regions are present they should not be broken. Bonds resulting in shorter chains should be preferred.

Break carbon-carbon bonds

For sake of calculation consistency, the bond to be broken should be between two identical atom types. AutoDock supports aliphatic and aromatic carbon atoms. Atoms different than carbon can be used but they will require a special

parameterization (see 4.Extension and Limits).

Avoid chiral atoms (...whenever possible)

Due to the lack of directionality and the united atom description of hydrogens, original chirality is *not guaranteed* if a bond between one or more chiral carbon atoms is broken. When all ring carbons are chiral (e.g. natural compounds, antibiotics) any bond can be suitable, while chirality in docking results should be inspected and manually corrected if necessary.

Once the ring is disrupted, the previously connected atoms must be renamed as "G" in the atom type column of the PDBQT file:

```
[...]
HETATM   21  CD4 UIN B 100      -2.919  22.061  19.604  1.00 19.90     0.005 G
[...]
HETATM   24  CD3 UIN B 100      -3.821  22.402  20.791  1.00 19.60     0.005 G
[...]
```

The 12-2 potential is defined in the DPF using the `intnb_r_eps` keyword to override the AutoDock internal interaction parameter table, using the following syntax :

```
intnbp_r_eps  1.51 10.000000 12 2  G G
```

AutoDock will acknowledge the new parameterization in the DLG:

```
Ring closure distance potential found for atom type G :
    Equilibrium distance   = 1.51 Angstroms
    Equilibrium potential  = 10.000042 Kcal/mol
    Pseudo-LJ coefficients = 12 - 2


Calculating internal non-bonded interaction energies for docking calculation;

Non-bonded parameters for G-G interactions, used in internal energy calculations:

                  281.0           27.4
    E       =  ----------  -  ----------
      G,G          12              2
                  r               r
```

More than one ring

Multiple flexible rings can be docked by disrupting a bond for each ring and using a different atom type for each edge atom pair. AutoDock includes four ring closure carbon atom types: `G`, `J`, `Q` (aliphatic) and `GA` (aromatic), then up to four flexible rings can be docked simultaneously. For example, if a second ring is opened then the next two edge atoms are renamed as J, and the DPF will include an extra `intnb_r_eps` keyword and another C map reference. While there is no actual limitation to the number of cycles that can be opened in the same molecule, there is the

implicit limit of the docking complexity, as well as the maximum number of rotatable bonds allowed. If further atom types need to be defined (e.g., -S-S- disulfide bond), a customized atomic parameter file must be generated and included in the DPF with the `parameter_file` keyword.

Limitations

Using this approach for docking flexible rings can save a lot of time compared to rigid ring docking of different conformations, but there are some limitations associated with the protocol implementation.

> *Chirality*. Hydrogen atoms bound to chiral edge atoms will be merged in the *united-atom* model used in AutoDock, then chirality information is lost. In the docking process G-atoms can eventually approach each other from directions different than the original geometry, leading to potentially wrong chirality.
>
> *Bond distance.* The pseudo-Lennard-Jones potential parameters describes the ideal equilibrium distance of the two G-atoms, corresponding to the equilibrium C-C bonding distance (~1.5 Å). The final distance although can be slightly bigger, because of the van der Waals repulsion between the two atoms preventing atomic volume overlaps.
>
> *Energy calculation.* During the calculation the pseudo-Lennard-Jones potential provides an extra energy contribution to the total energy sum to induce the ring closure. This can result in a overall shifting of the final energy to lower values. While not being an actual limitation, it should be considered to avoid comparisons between scores obtained with and without flexible rings.

For these reasons, the final docking result should be refined by inspecting the chirality and performing a geometry refinement to correct bonding angles and distances.

This is the DPF corresponding to the example structure in the introduction:

```
autodock_parameter_version 4.1        # used by autodock to validate parameter set
outlev 1                               # diagnostic output level
intelec                                # calculate internal electrostatics
seed pid time                          # seeds for random generator
ligand_types A C G HD OA               # atoms types in ligand
fld protein.maps.fld                   # grid_data_file
map protein.A.map                      # atom-specific affinity map
map protein.C.map                      # atom-specific affinity map
map protein.C.map                      # C map fir G atoms
map protein.HD.map                     # atom-specific affinity map
map protein.OA.map                     # atom-specific affinity map
elecmap protein.e.map                  # electrostatics map
desolvmap protein.d.map                # desolvation map
intnbp_r_eps 1.51 10.000000 12 2 G G # pseudo-LJ potential
move ligandG.pdbqt                     # small molecule
about -0.8665 18.5882 20.1623          # small molecule center
tran0 random                           # initial coordinates/A or random
quat0 random                           # initial quaternion
dihe0 random                           # initial dihedrals (relative) or random
tstep 2.0                              # translation step/A
qstep 50.0                             # quaternion step/deg
dstep 50.0                             # torsion step/deg
torsdof 8                              # torsional degrees of freedom
rmstol 2.0                             # cluster_tolerance/A
extnrg 1000.0                          # external grid energy
e0max 0.0 10000                        # max initial energy; max number of retries
ga_pop_size 350                        # number of individuals in population
ga_num_evals 2500000                   # maximum number of energy evaluations
ga_num_generations 27000               # maximum number of generations
ga_elitism 1                           # number of top individuals to survive to next
generation
ga_mutation_rate 0.02                  # rate of gene mutation
ga_crossover_rate 0.8                  # rate of crossover
ga_window_size 10                      #
ga_cauchy_alpha 0.0                    # Alpha parameter of Cauchy distribution
ga_cauchy_beta 1.0                     # Beta parameter Cauchy distribution
set_ga                                 # set the above parameters for GA or LGA
sw_max_its 300                         # iterations of Solis & Wets local search
sw_max_succ 4                          # consecutive successes before changing rho
sw_max_fail 4                          # consecutive failures before changing rho
sw_rho 1.0                             # size of local search space to sample
sw_lb_rho 0.01                         # lower bound on rho
ls_search_freq 0.26                    # probability of performing local search on individual
set_sw1                                # set the above Solis & Wets parameters
unbound_model bound                    # state of unbound ligand
ga_run 100                             # do this many hybrid GA-LS runs
analysis                               # perform a ranked cluster analysis
```

3. Reference

Forli, S., et al. *J. Chem. Inf. Model.*, 2007, 47, 1481–1492

4. Tutorial

http://autodock.scripps.edu/faqs-help/tutorial/flexible-rings-docking

Appendix III: AutoDock References

AutoDock 4.2

Morris, G. M., Huey, R., Lindstrom, W., Sanner, M. F., Belew, R. K., Goodsell, D. S. and Olson, A. J. (2009) J. Comput. Chem. in press. “Autodock4 and AutoDockTools4: automated docking with selective receptor flexiblity.”

AutoDock 4.0

Huey, R., Morris, G. M., Olson, A. J. and Goodsell, D. S. (2007) J. Comput. Chem. 28, 1145-1152. “A semiempirical free energy force field with charge-based desolvation.”

Huey, R., Goodsell, D. S., Morris, G. M. and Olson, A. J. (2004) Letters in Drug Design and Discovery 1, 178-183. “Grid-based hydrogen bond potentials with improved directionality”.

AutoDock 3.0

Morris, G. M., Goodsell, D. S., Halliday, R.S., Huey, R., Hart, W. E., Belew, R. K. and Olson, A. J. (1998), *J. Computational Chemistry*, **19**: 1639-1662. "Automated Docking Using a Lamarckian Genetic Algorithm and and Empirical Binding Free Energy Function".

AutoDock 2.4

Morris, G. M., Goodsell, D. S., Huey, R. and Olson, A. J. (1996), *J. Computer-Aided Molecular Design*, **10**: 293-304. "Distributed automated docking of flexible ligands to proteins: Parallel applications of AutoDock 2.4".

Goodsell, D. S., Morris, G. M. and Olson, A. J. (1996), *J. Mol. Recognition,* **9**: 1-5. "Docking of Flexible Ligands: Applications of AutoDock".

AutoDock 1.0

Goodsell, D. S. and Olson, A. J. (1990), *Proteins: Str. Func. and Genet.*, **8**: 195-202. "Automated Docking of Substrates to Proteins by Simulated Annealing".