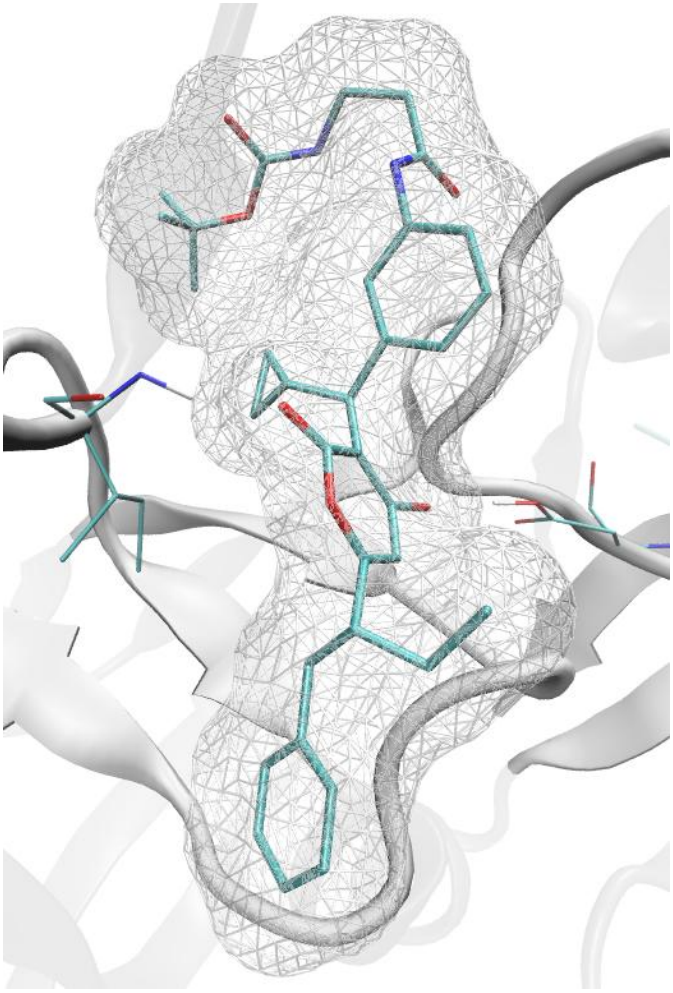

Build_model

v.2.0.1

User Guide

MolTech Build_model User Guide

www.moltech.ru

Please send your comments and suggestions to contact@moltech.ru.

Table of Contents

Introduction

"Build_model" is a tool for creating protein models and their preparation for docking. With this tool you can refine raw protein structure, add missing sidechains, assign protonation states of side chains at given pH, add missing hydrogen atoms, reconstruct crystallographically-related protein subunits and extract a reference ligand from the structure.

"Build_model" can be useful for computational and medicinal chemists working on drug discovery; pharmacologists and toxicologists for ADMET in silico modeling and biochemists conducting their research in the field of enzyme's enantioselectivity and rational enzyme design.

Input and output data

"Build_model" takes input protein structure, and an optional file with settings. The output of "Build_model" is prepared full atomic model of the protein structure and a log file of model building. "Build_model" does not have any specific limitations on input structure except for the compliance with file format.

It is well known that quality of 3D structure of the protein is of utmost importance for molecular modeling of biological activity. Low-quality protein structures inevitably lead to low-quality predictions of activity. "Build_model" software allows to perform the following steps in preparation of high-quality protein models:

- Protein structures especially those derived from open sources, e.g. PDB usually are missing hydrogen atoms. "Build_model" adds non-polar hydrogen atoms to protein and cofactors.

- "Build_model" adds missing polar hydrogens taking into account
    a) the dependence of protein ionization state on pH;
    b) alternative ways of attaching hydrogens to equivalent atoms (e.g. in the case of His, Asp or Glu);
    c) positions of polar hydrogens that are optimized to saturate their hydrogen-bonding interactions and minimize steric strain.

- Sometimes crystallographic structures contain residues with unresolved sidechains. "Build_model" software automatically detects and reconstructs them.

- "Build_model" can take into account crystallographic subunits in vicinity of input protein structure. When needed, these subunits can be assembled and saved to output file.

- "Build_model" can automatically detect and extract the most suitable reference ligand for docking. This is useful for automatic preparation of large amounts of protein models for virtual ligand screening and docking. Reference ligand can be specified manually (see "Description Language" section).

"Build_model" takes input structure of the protein in the PDB file format. To convert your structure in PDB format, you may use free program Open Babel http://sourceforge.net/projects/openbabel/.

Since the time of model building depends on the number of ionizeable residues, we recommend to delete all protein fragments that are not really needed. For example, many structures from PDB have crystallographic subunits, which don't interact with ligand. Deletion of such subunits could significantly accelerate model building procedure.

Installation

Installation in Windows.

Program «Build_model» doesn't require special installation. You can copy all input files in the folder that contains executable file (*build_model.exe*) and all essential libraries and then run the command:

build_model.exe –f protein.pdb –omm results_protein.pdb –olog results_model.log

You can add folder with an executable "Build_model" file to system paths. Since "Build_model" is a command-line application, adding folder "Build_model" in system paths will allow you to run "Build_model" from any directory via simple command "*build_model.exe*". Remember that changes in system paths are applied only after reboot.

Installation on Linux

Program «Build_model» doesn't require special installation. You can copy all input files in the folder that contains executable file ("*build_model.exe"*) and all essential libraries and then run the command:

./build_model –f protein.pdb –omm results_protein.pdb –olog results_model.log

For convenient usage you can copy executable file and all essential libraries into the folder /usr/local/bin or any other folder or add it to system paths in ~/.bashrc or create symbolic link in /usr/local/bin. Remember that changes in ~/.bashrc are applied after next boot of terminal program.

Configuration settings

Configuration parameters for program "Build_model" can be set in the command prompt. The order of parameters is of no object.

Program "Build_model" takes input structure of the protein and a file with additional settings. The output of "Build_model" is a prepared full-atom protein model and log-file of model building (*model.log*). *model.log* file contains information on reconstructed residues, calculated residues' pKa and information on selected ligand.

-f <filename>

Mandatory parameter. Specifies the initial protein structure for which the model building will be performed.

-set <filename>

Optional parameter. Specifies the program settings file which contains instructions on residue protonation, water removal etc. For the complete description of parameters defined in the set-file see the «Set-file format» section.

-oref <filename>

Output file with the ligand structure in the pdb format.

-olig <filename>

Output file with the ligand structure in the mol format.

-omm <filename>

Output file with the prepared protein structure.

-olog <filename>

Model building log-file.

-h

Print help on configuration settings.

-v

In the verbose mode program displays a detailed information about current actions on the display.

-mode [normal | fast | superfast]

Program precision mode (affects the protonation). Three modes are available: normal, fast and superfast. In the normal mode sampling operators are applied to the whole system. In the fast mode sampling operators are applied only to residues which have contacts with the

ligand. In superfast mode conformational sampling is not performed. Fast mode is used by default.

-cryst <X>

Generate crystallographic subunits at a specified distance from the crystallographic selection. Way of defining the crystallographic selection is described in the “Set-file format”. Ligand is used as a crystallographic selection by default.

-pH <X>

Specify the pH to determine ionization states of protein residues and ligand.

Set-file format

General settings

Input text-file with description of tasks to perform. Data are read line by line. Lines starting with semicolon (;) are treated as comments. Each line contains one of the following instructions.

complete_sidechains <yes|no>

Whether to reconstruct side chains of amino acids

accuracy_mode <mode>

Program precision mode (affects the protonation). Three modes are available: normal, fast and superfast. In the normal mode sampling operators are applied to the whole system. In the fast mode sampling operators are applied only to residues which have contacts with ligand. Fast mode is used by default.

extract_water <yes|no>

Whether to remove water from the structure.

protonate_intact <selection>

Do not add protons on the specified selection. Number of protons and their position on these atoms should not change. See description of the selection format in «Description language» section.

protonate_yes <selection>

Specified selection will be protonated. If there is no univocal way of protonation for the given selection (e.g. if the selection contains Asp side chain any oxygen can be protonated), the choice of protonated states will be carried using the protonation algorithm and deprotonated states will not be considered. You can explicitly specify which atoms should be protonated and thus choose specifically oxygen atom of Asp to put the proton. See description of the selection format in «Description language» section.

protonate_no <selection>

Specified selection will be deprotonated. If there is no univocal way of deprotonation for the given selection (e.g. if the selection contains His side chain which has two deprotonated forms HisE and HisD), the choice of deprotonated states will be carried using the protonation algorithm and protonated states will not be considered. You can explicitly specify which atoms should be protonated and thus choose specifically nitrogen atom of His which should and nitrogen atom which shouldn’t have a proton. See description of the selection format in «Description language» section.

cryst_distance <X>

Distance for crystallographic subunits reconstruction. Subunits which are closer than X from the central selection (cryst_core) will be reconstructed.

cryst_core <selection>

Crystallographic selection center from which distance to reconstruct subunits is measured. Ligand is used as selection by default. See description of the selection format in «Description language» section.

cryst_subunit <selection>

Selection which should be reconstructed. Reconstruction of the whole system is carried out by default. See description of the selection format in «Description language» section.

ligand <selection>

Use the selection as a ligand. If selection is not specified or it's empty ligand is determined automatically. See description of the selection format in «Description language» section.

ligand_distance <X>

Use specified distance from ligand in fast and superfast protonation modes. 6Å is used by default.

save_ligand <yes|no>

Whether to save ligand separately from the protein. If ligand won't be saved the whole protein is saved.

mutate <task>

Perform amino acid replacement (mutation). Section «amino acid replacement settings» contains amino acid replacement format description. Several replacements are possible.

Amino acid replacement settings

Each string should contain description of mutation on some position:

[<chain>] <residue> [<name>]

<chain>

Protein chain identifiers for given position (for example, A). If chain identifier is not specified, all mutants are generated for specific position in all chains of the protein.

<residue>

Number of the residue to mutate/reconstruct

<name>

Mutant which should be constructed on the current position. One-letter and three-letter codes of aminoacids are allowed. If the name is omitted, side-chain of the residue on the given position will be reconstructed.

Examples of the tasks:

A 145 ARG

Replace residue 145 of chain A by arginine.

A 146 F

Replace residue 146 of chain A by phenylalanine.

Sidechains libraries format

Sidechains library is a text file with #include directives and comments after semicolons. By default "Build_model" searches for sidechains library in directory, given by environment variable $MUTATE_LIB, then in current directory, in subdirectory "libraries" of current directory, in program directory and in subdirectory "libraries" of the program directory. File "sidechains.lib" should contain set of commands, with each command located on a separate line. Available commands are:

sdf <filename>

Load sdf-file with sidechains from <filename>. Every record in sdf-file should contain header, that will be used to identify residue in the library. Records should contain atoms with names N,C,CA that will be used to identify sidechains and anchor points of residues.

resname <name> <synonym1> [...]

Set up synonym for residue <name> (e.g., this is the way to declare one-letter codes of aminoacids)

subset <name> <names>

Set up name for subset of aminoacids

pKa <name> <X> <acidic|basic>

Set up pKa <X> for group <name>. Besides value group type should be defined: acidic – group population decreases with increase of pH, basic – group population increases with increase of pH. Library should contain pKa for all residue forms.

energy <name> <X>

Set up additional energy for group <name>. This parameter could be applied for instance if it's necessary to change energy of one cysteine form during mutation of an arbitrary residue to CYS and in analogous cases.

group <names>

Join list of the residues into one group. If a group name is specified in residues replacement list, the best mutant from the group will be selected.

operator <new|old> <name|default> <list_of_operators>

Set up geometry operators for residue with name <name>. If keyword *new* is specified, geometry operators are applied when the residue is reconstructed, or mutation to this residue is performed. If keyword *old* is specified, geometry operators are applied for residues, which are neighbors of mutated/reconstructed positions. When instead of the name of the residue keyword *default* is specified, operators are applied for all residues, for which geometry operators were not specified explicitly.

List of geometry operators

Geometry operators generate ensemble of group conformations based on the initial group state and several rules. For each operator except operator *randomize* the initial conformation is presented in the generated ensemble. During residue reconstruction its initial conformation corresponds to the conformation of this residue in the wild-type protein. During mutations (including residue mutation by itself) initial conformation is taken from the sidechains library.

amide
: Amide group flip.

imidazole
: Imidazole group flip (e.g. in histidines)

rotate-H
: Rotation of hydrogens with the default step

rotate-H-symmetric
: Rotation of hydrogens taking into account symmetry of the group (e.g. for NH_3^+ group rotation period is 120°). Rotation is performed with the default step

rotate-H-60
: Full rotation of hydrogen with step 60°.

rotate-H-90
: Full rotation of hydrogen with step 90°.

water
: Specific sampling of water molecules.

LP90
: Rotation of electron pairs of acceptor by 90°.

shake_N_X
shake_N
shake
: Local sampling with N conformations and maximal RMS from original structure X Å. If X is not specified, N conformations will be generated with RMS 1.5 Å. If number of conformations is not specified, 10 conformations with RMS 1.5 Å will be generated.

shake_N_X_Y

shake_N_X_Y_Z

If operator shake is applied to the fragment which have no anchor atoms (is not attached covalently to other system) it is possible to specify distribution of RMS over rotational, translational and internal degrees of freedom. When only X parameter is specified, RMS is distributed evenly over translation, rotation and internal degrees of freedom. When X and Y parameters are specified, internal degrees of freedom will have RMS X Å, rotation and translations - ½Y Å each. When all three values (X, Y, Z), then X sets RMS of internal rotations, Y sets RMS of rotation of the whole fragment, and Z sets RMS of translations.

fullshake

fullshake_N

fullshake_N_X

Operator of stochastic global sampling generates N random uniformly distributed conformations of the fragment. If the fragment is not attached to the rest of the system by covalent bonds, uniform translations at X Å (1 Å by default) and rotations are generated as well. If the number of conformations is not specified, 4100 conformations will be generated.

fullscan

fullscan_N

Scan of all internal rotations with step 360/N degrees. If N is not specified, default step 120° (N=3) is used. If number of generated conformations exceeds 4100, then operator fullshake with 4100 conformations is used instead of fullscan.

scan_R1_S1_R2_S2...RN_SN

Local scanning of first N dihedrals of the residue. For each dihedral radius of scan R and step S (integer amount of degrees) should be specified. R1 and S1 correspond to χ_1, R2, S2 – to χ_2 etc.

auto

Automatic detection of suitable operator of hydrogens rotations.

randomize

Operator of randomization of original conformation. Can be used to remove initial conformation from the generated conformational ensemble.

rotamers

Generate rotamers by sidechain rotamers library (rotamers.lib)

Description language

Various operations (e.g. protonation) can be performed with distinct parts of molecules which could be defined in selections.

Simple selections

1. String «atom name CA N C O» selects atoms with names "CA", "N", "C", "O".
2. String «atom num 1 2 5» selects atoms with numbers "1", "2" и "5"
3. String «residue name ARG LYS » selects residues with name "ARG", "LYS.
4. String «residue num 10 12» select residues with numbers "10" and "12".
5. String «element H» selects all hydrogens in the molecule.

Note: Different residues (or atoms) may have the same number (e.g. residues in each pdb file chain can be enumerated separately). Therefore before applying selections check the numbering in your file.

Extended selection syntax

If you often use selections, you need a more flexible and more powerful selections language. Consider specific examples.

Selection types

1. Select atoms with name "CA", "N", "C", "O":
   a. «**atom name** CA N C O»
   b. «**a name** CA N C O» Keyword «**atom**» is replaced by «**a**».
2. Select residues with numbers "1", "25":
   a. «**residue num** 1 25»
   b. «**r num** 1 25» Keyword «**residue**» is replaced by «**r**».
3. Select nitrogen and chlorine atoms in the molecule
   a. «**element** N Cl».
   b. «**el** N Cl».

Logical operators

4. Select atoms with names "CA", "N", "C", "O" in residues ARG and ASP:
   a. «(*residue name* ARG ASP) **and** (*atom name* CA N C O)».
   b. «(*r name* ARG ASP) **and** (*a name* CA N C O)».
5. Select atoms with names "CA", "N", "C", "O" and residues ARG and ASP:
   a. «(*residue name* ARG ASP) **or** (*atom name* CA N C O)».
   b. «(*r name* ARG ASP) **|** (*a name* CA N C O)»
6. Select all except residues ARG and ASP:
   a. «**not** (*residue name* ARG ASP)».

b. «**!** (*r name* ARG ASP)»

7. In residues ARG select all atoms except "CA", "N", "C", "O"
   a. «(*r name* ARG ASP) **and not**(*a name* CA N C O)».
   b. «(*r name* ARG ASP) **& !**(*a name* CA N C O)»

Examples

Installation package of "Build_model" software include three examples, illustrating different ways of usage of the software.

Example 1. Preparation of the protein model with default settings.

First example demonstrates how to prepare protein model with default settings. Input structure is proto-oncogene tyrosine-protein kinase Src (PDB: 1a1b). To prepare model, run command:

build_model.exe -f 1a1b.pdb -olog model.log

The program will produce file *protein.pdb* with prepared model. File *ligand_reference.pdb* will contain ligand in pdb-format, and file *ligand.mol* will contain structure in mol-format. Prepared models of protein, reference and ligand can be used in docking. File *model.log* contains information of 14 reconstructed side chains, calculated pKa values and information about the selected ligand.

Example 2. Usage of descriptions to specify ionization state.

This example demonstrates how to use descriptions language to assign correct ionization states to specific protein residues. Input file contains structure of prothrombin (PDB: 1gj4), file *1gj4.set* is used to specify the protonation state of atom "O6'" of the ligand. To prepare model, run command:

build_model.exe -f 1gj4.pdb -set 1gj4.set -olog model.log

The program will produce file *protein.pdb* with prepared model of the protein. File *ligand_reference.pdb* will contain ligand in pdb-format, and file *ligand.mol* will contain structure in mol-format. Note the protonation state of atom "O6'" of the ligand and the position of the "HG" proton of Ser195.

Example 3. Usage of descriptions to select ligand.

It is well known, that often input protein structure apart from ligand could contain buffer components and cofactors. Besides, sometimes an input structure contains several ligands in different binding sites. "Build_model" software has internal rules for discriminating buffer components and cofactors. However, if the input structure contains several ligands, one may want to use different ligand from what the program selected automatically. This example demonstrates how to use description language to select specific ligand of two ligands, contained in the structure of cell division control protein 42 homolog (PDB: 1a4r). Ligand is defined in file *1a4r.set*. To prepare model, run command:

build_model.exe -f 1a4r.pdb -set 1a4r.set -olog model.log

The program will produce file *protein.pdb* 6with prepared model of the protein. File *ligand_reference.pdb* will contain ligand in pdb-format, and file *ligand.mol* will contain structure in mol-format.