Table of Contents

autoBUSTER Documentation : Introduction

Contact buster-develop@GlobalPhasing.com

Contents

Scope and intended use

autoBUSTER is a framework and collection of tools, intended for refining structures with the latest version of the BUSTER-TNT refinement program in conjunction with the geometry module **GELLY**, the **grade** tool for generating restraint dictionaries, and the **MakeTNT** collection of tools for manipulating restraint dictionaries.

autoBUSTER handles a variety of cases typical for macromolecular refinement:

- protein structures with or without ligands or co-factors
- DNA and RNA structures
- high- and low-resolution structures
- presence of non-crystallographic symmetry
- TLS parameterisation
- already well-refined structures or structures near the beginning of the refinement process

The main source of documentation for **autoBUSTER** is the BUSTER wiki.

If you have questions or problems, please contact buster-develop@globalphasing.com.

How to cite use of BUSTER

- Please refer to the BUSTER wiki page: How to cite use of BUSTER.

Authors and contributions

- **autoBUSTER**: The tools, scripts and programs that make up the autoBUSTER framework are developed by: G. Bricogne, C. Flensburg, P. Keller, W. Paciorek, A. Sharff, O. Smart, C. Vonrhein, T. Womack with contributions from former group members E. Blanc M. Brandl and P. Roversi.

- **BUSTER-TNT**: The writing of BUSTER was started in 1989 by Gerard Bricogne. Subsequent contributions, including the interfacing of BUSTER to TNT, have come from John Irwin (1994 - 1998), Pietro Roversi (1995 - 2003), Clemens Vonrhein (since 1998), Eric Blanc (1998 - 2003), Maria Brandl (2004 - 2009), Wlodek Paciorek

(since 2004), Claus Flensburg (since 2000), Oliver Smart (since 2004), Thomas Womack (since 2007) and Andrew Sharff (since 2009).

- **GELLY**: The writing of GELLY was started by Oliver Smart in May 2004, with further contributions from Claus Flensburg (since December 2004) and Thomas Womack (since 2007).
- **MakeTNT**: The writing of the MakeTNT toolkit was developed by Maria Brandl between 2004 to 2009.
- **grade**: grade was written by Oliver Smart and Thomas Womack, starting in April 2010.

Last modification: 04.02.11

autoBUSTER Documentation : Installation

Contact buster-develop@GlobalPhasing.com

Contents

Prerequisites & remarks

For users of **SHARP/autoSHARP** or users of very old versions of **BUSTER-TNT** it might be of interest, that no running httpd or configured user is required (although some messages might appear during the installation, stating the contrary: the http daemon is stopped immediately and should not be running after installation).

Every machine that you want to run BUSTER refine and other tools on needs an individual valid licence key (unless you are a Consortium user with a "magic" licence that works on all machines). For information on licencing please visit http://www.globalphasing.com/buster/. All the separate licence keys should be included in the file `$BDG_home/.licence`.

Installing

The installation, configuration of BUSTER and related tools is now dealt with in a separate set of detailed instructions..

Last modification: 28 August 2013

autoBUSTER Documentation : File formats

Contact buster-develop@GlobalPhasing.com

Contents

PDB

A great deal of checking is done by the program pdbchk, which is used as part of each **autoBUSTER** job. Some of the problems and inconsistencies found in the starting PDB file can also be corrected at that step. These include e.g.

1. each atom should have a chain identifier (e.g A, B and C for protein chains and W for water)
2. a correct CRYST1 card is required (see PDB format guide), especially the space group symbol.
3. although not enforced by the PDB standard, it seems sensible to use letters (A, B, C etc) in column 17 of ATOM/HETATM records to denote alternate conformations and numbers (1, 2, 3 etc) in column 27 of ATOM/HETATM records to denote insertion code.

autoBUSTER internally uses atom/residue nomenclature PDB v2.

MTZ

Reflection data is given in CCP4 MTZ format (binary file format):

1. normal MTZ file with F/SIGF columns (any column name is possible, but the column types have to be F/Q - which they nearly always are anyway, unless something went really wrong)
2. the cell parameters for the refinement are taken from the MTZ file header (please note that it does **not** yet handle *different* cell entries for different datasets as e.g. in CCP4 5.0.X/6.0.X/6.1.X!). The assumption is that the MTZ file usually contains only a single dataset.
3. if the MTZ file contains a set of columns with Hendrickson-Lattmann coefficients (usually named HLA, HLB, HLC and HLD) these can be used as additional, external phase information (unless the MTZ file is actually the output of a previous **BUSTER** run - which would not be a good idea). The user needs to set the parameter autoBUSTER_hls to the four column names, e.g. with '`refine autoBUSTER_hls="HLA HLB HLC HLD" ...` '.

Rigid-body description

- The rigid-body description file is used with the `-RB` command line argument, e.g.

```
% refine -p some.pdb -m other.mtz -RB rigid.dat
```

- These files describe the rigid bodies to be used for the initial big cycle(s) of rigid-body refinement that is done if `-RB rigid.dat` is specified. After this first big cycle of rigid-body refinement, normal (xyz and B) refinement is done for all subsequent cycles.

  Note that `-RB` without a file being specified will define a single rigid body for every chain in the input pdb file. This is often a sensible initial approach.

- The rigid-body file uses gelly combine syntax. E.g.:

```
NOTE BUSTER_COMBINE XYZ { A|5 - A|73  A|150 - A|170 }
NOTE BUSTER_COMBINE XYZ { A|74 }
NOTE BUSTER_COMBINE XYZ { A|75 }
NOTE BUSTER_COMBINE XYZ { A|76 }
NOTE BUSTER_COMBINE XYZ { A|77 - A|120 }
```

  This sets up two large rigid bodies for two domains. The first domain contains residues 5 to 73 and 150 to 170. The second domain goes from residue 77 to 120. The three residues in between (the linker) are treated as individual rigid bodies. This can be sensible because bonded interactions remain fully active throughout rigid-body refinement using **BUSTER** - only non-bonded contacts are being zero weighted in rigid-body refinement cycles. So to allow the domains to move more freely, the linker residues are kept individually rigid. A good alternative would be to simply delete a single residue in the linker to remove any bonded connection between the domains.

- You can use several `-RB` arguments as in

```
% refine -RB rigid1.dat -RB rigid2.dat ...
```

  Here the first two big cycles will be rigid-body refinement cycles - with the rigid-body parameters ***rigid1.dat*** for the first big cycle and ***rigid2.dat*** for the next big cycle. From big cycle 3 onwards, no rigid-body restraints will be used.

- If you want to restrict the resolution range in a particular rigid-body refinement cycle, then this can be done by adding a special RESOLUTION card to a rigid-body definition file. Just add a line (starting with a hash) to the beginning of the file:

```
# RESOLUTION <low res> <high res>
```

  In this case, only reflections within the specified resolution range will be used during that particular rigid-body refinement cycle. As an example: to use only data to 4 Å in a two chain rigid-body refinement step:

```
# RESOLUTION 50.0 4.0
NOTE BUSTER_COMBINE XYZ { A|* }
NOTE BUSTER_COMBINE XYZ { B|* }
```

  Using only low resolution data during a rigid-body refinement cycle can help increasing the radius of convergence.

- For further discussion as to the use of rigid-body refinement, see the Rigid-body usage section.

NCS

There are several ways of specifying NCS:

1. By far the easiest option is to just use the **`-autoncs`** command-line flag (for LSSR-type NCS restraints)
2. The second way is to use **GELLY** syntax for NCS-specification.
3. The old option for superposition-based NCS restraints:

- uses normal TNT style syntax for describing NCS restraints.
- a simple example would look like this:

```
CLUSTER N1 RESIDUE   1 -  20 \
           RESIDUE  22 -  79 CHAINS A B
CLUSTER N2 RESIDUE  80 - 101 CHAINS A B
```

This describes a two-domain protein (`N1` and `N2`) which crystallises with 2 molecules (chains `A` and `B`) in the asymmetric unit. Residue 21 in the first domain (`N1`) has been taken out of the NCS relation (maybe due to a different crystal contact).

TLS description

The default behaviour of the TLS refinement options in **autoBUSTER** is to read existing TLS group definitions from the PDB file header, if present. Failing that, a single TLS group will be defined per macro-molecular chain. Please refer to TLS refinement for more information on how to set up simple TLS refinement.

For more complex TLS parameterisation, it is possible to specify custom TLS group definitions in a **GELLY** syntax file given as an argument to the -TLS command.

There are several cards that describe a TLS group. They fall into three groups listed below. All of them use a unique `tag` to specify a particular TLS group.

1. Specification of the content of a group:

```
NOTE BUSTER_TLS_SET <tag> <spec>
```

This card is mandatory for TLS-refinement.

The specification `<spec>` can be either a single selection using 'curly-braces', eg.

```
NOTE BUSTER_TLS_SET tls1 { A|1 - A|150 A|201 - A|360 }
```

or a single set specified using the **NOTE BUSTER_SET** syntax, eg.

```
NOTE BUSTER_BUSTER_SET group1 = { A|1 - A|150 }
NOTE BUSTER_BUSTER_SET group2 = { A|201 - A|360 }
NOTE BUSTER_BUSTER_SET tls1 = group1 + group2
NOTE BUSTER_TLS_SET tls1 tls1
```

2. The values of any known parameters of a given TLS group, either the origin or the unique values of the T-, L-, and, S-tensors, are specified as follows:

```
NOTE BUSTER_TLS_O <tag> <X> <Y> <Z>
NOTE BUSTER_TLS_T <tag> <T11> <T22> <T33> <T12> <T13> <T23>
NOTE BUSTER_TLS_L <tag> <L11> <L22> <L33> <L12> <L13> <L23>
NOTE BUSTER_TLS_S <tag> <S2211> <S1133> <S12> <S13> <S23> <S21> SS31> S32>
```

NOTE: `tag` must be the same as in the NOTE BUSTER_TLS_SET card

These cards are not mandatory. If no origin has been specified, the centroid of the atoms in the group is used. Similarly, if the T, L, and, S parameters are unspecified the values are set to zero. The element `<S2211>` is `<S22> - <S11>`, while `<S1133>` is `<S11> - <S33>`.

The values must be given in the TNT-Cartesian system and the units are Å, $Å^2$, $°^2$, and, Å°, respectively.

3. The following card will determine whether to keep the TLS parameters fixed or to refine them:

```
NOTE BUSTER_TLS_FIX <tag> (RB|ALL)
```

A value of `RB` specifies that the parameters associated with the Rigid-Body part of a TLS group are kept fixed, ie. the location and the relative orientation (this is the default). A value of `ALL` completely fixes the TLS group.

Switching the refinement of TLS-parameters on or off at different big cycles of an **autoBUSTER** run, is controlled by the variables: TLSfixcycRB and TLSfixcycALL.

Example: these cards would specify two TLS groups that are to be refined with fixed translational/rotational parts:

```
NOTE BUSTER_TLS_SET tlsA  { A|* }
NOTE BUSTER_TLS_T   tlsA  -0.05 -0.11 -0.15 -0.01  0.03  0.02
NOTE BUSTER_TLS_L   tlsA   2.88  1.70  1.17 -0.41  0.32 -0.35
NOTE BUSTER_TLS_S   tlsA  -0.11  0.02 -0.10 -0.09  0.04  0.01  0.01 -0.01
NOTE BUSTER_TLS_O   tlsA   6.42  3.54 15.71
NOTE BUSTER_TLS_FIX tlsA  RB

NOTE BUSTER_TLS_SET tlsB  { B|* }
NOTE BUSTER_TLS_T   tlsB  -0.01 -0.03 -0.03  0.00  0.00  0.02
NOTE BUSTER_TLS_L   tlsB   0.38  0.45  0.58  0.04 -0.04 -0.02
NOTE BUSTER_TLS_S   tlsb  -0.02 -0.02 -0.01  0.01 -0.02 -0.02  0.02 -0.01
NOTE BUSTER_TLS_O   tlsB  -4.40 28.29 43.24
NOTE BUSTER_TLS_FIX tlsB  RB
```

The pdb2tls tool provided, can be used as an easy way of generating a TLS definition file - especially when applied to a PDB file already refined with TLS (which then should contain a REMARK 3 section with TLS details). The resulting file should be a good example to understand the format used within **BUSTER**.

TNT sequence file

The TNT sequence file describes the connectivity between residues and atoms in the PDB file. Every residue in the PDB file must be described in the TNT sequence file, though it is permitted for the TNT sequence file to describe residues or atoms which are missing in the coordinate file – you can keep the residue type the same even if a sidechain is truncated. If you have very large missing sections in your input model, you can generate a sequence file from a FASTA or PIR ASCII sequence using `seq2seq`.

By default, the sequence file is generated automatically from the input model using the MakeLINK utility. MakeLINK is aware of a number of common covalently-bound cofactors and glycosylation patterns; if you have more complicated linkages in your protein, you have two choices.

1. You can produce the sequence file manually, edit it by hand, and submit it with the `-Seq` option
2. You can incorporate `MakeLINK` directives in a TNT-format dictionary passed to `refine` with the `-l` option, and autoBUSTER will arrange to pass these to MakeLINK. See GradeCovalentTutorial on the BUSTER wiki for an example.

If your input model contains accidental contacts between protein regions from different parts of the sequence (this is something we have seen for output from Buccaneer or from mediocre molecular-replacement output) then MakeLINK may introduce incorrect cross-links, which will tend to be reported as sanity-check failures from autoBUSTER. In these cases you can run with `SequenceFileGeneration=pdb2seq` to use a different sequence file generation method; note that this method is unaware of covalent linkages other than that present in protein (peptides) or DNA/RNA.

Geometry restraints and standard libraries

autoBUSTER needs to be given information about the geometry of the ligands in your file. This should be made available as a refmac-compatible .cif file, as produced by many dictionary-generation programs, including grade which is part of the autoBUSTER distribution.

If you do not give a dictionary and **autoBUSTER** does not have one available internally, you will get an error message from `refine` telling you for which three-letter codes dictionaries are needed.

Dictionaries for ligands which are known to the PDB can be made very easily using the **grade_PDB_ligand** tool; you

need to have `babel` on your path, and you will get very much better results if you have the CSD tool `mogul` on your path. You must use the `-nomogul` option to `grade_PDB_ligand` if you don't have mogul.

autoBUSTER contains a library of restraint dictionaries for fifty or so of the most common residues, mostly generated with the grade_PDB_ligand tool mentioned above, but with some tweaks applied by hand. Giving a dictionary for the residue using the `-l` option will override the one in the library, though we would appreciate reports if you have ever had to do this because the dictionary in the library does not work correctly.

It is at present still possible to use the legacy TNT format for dictionaries, and indeed the protein and sugar restraint libraries are currently distributed in this format. We would not recommend that this format be used for any new work, though it is still necessary for accessing certain features.

Last modification: 25.04.2014

autoBUSTER Documentation : Usage

Contact buster-develop@GlobalPhasing.com

Contents

Running the "refine" command

- Please note that the installation, configuration of BUSTER and related tools is now dealt with in a separate set of detailed instructions.. These instructions describe how to get 'refine' working.
- in its simplest form, the 'refine' binary just needs the name of a starting PDB file and MTZ file:

```
% refine -p some.pdb -m other.mtz
```

- to have all results in a separate sub-directory (instead of the current directory - always a good idea):

```
% refine -p some.pdb -m other.mtz -d results.dir
```

- it is also recommended to save the standard output (and stderr) in a file:

```
% refine -p some.pdb -m other.mtz -
d results.dir > results.lis 2>&1   # for sh/bash/ksh/zsh
   - or -
% refine -p some.pdb -m other.mtz -d results.dir >& results.lis       # for csh/tcsh
```

- if there is non-crystallographic symmetry (NCS) present:

```
% refine -p some.pdb -m other.mtz -autoncs -d results
```

- if a ligand is (probably) present but the location isn't known:

```
% refine -p some.pdb -m other.mtz -L -d results
```

- if a ligand is (probably) present and the location can be described by a PDB file:

```
% refine -p some.pdb -m other.mtz -Lpdb site.pdb -d results
```

- The most important options can be seen by running:

```
% refine -h
```

- All options can be seen by running:

```
% refine -hhh
```

Command line arguments for the "refine" command

The most important command line flags are summarised below:

Flag	Arguments	Explanation	Remark
-h		Basic help message	Special option to print help message and exit. Most important options shown.
-hh		Longer help message	Special option to print help message and exit. More options shown. To show all options use -hhh.
-p	<PDB file>	PDB file with complete macro-molecule to be refined	PDB file requirements for autobuster.
-m	<MTZ file>	reflection file in MTZ format with correct space-group and cell parameters	MTZ file requirements for autobuster.
-d	<subdir>	all files will be created in sub-directory	it is a good idea to use some systematic numbering, otherwise the current working directory might get cluttered with output. If I/O over the network is slowing down calculations, this sub-directory should be located on a fast, local file-system.
-l	<library>	user supplied geometric restraints dictionary	several -l switches may be given (as many as may be necessary); these restraint dictionary files can be REFMAC-style CIF restraint dictionaries or in TNT-format. Conversion of REFMAC-style restraint dictionaries is done with refmacdict2tnt (since March 2010 release).
-WAT	[<ncyc>]	switches on water updating (optionally only after cycle <ncyc>)	default = don't do water updating.
-M		use a predefined macro	each macro combines a set of related parameters to perform a specific task. To see a list of available macros: *refine -M list*.
-nbig	<no BIG cyc>	Number of BIG cycles (refinement/water building/bulk solvent mask update/weight adjustment) to perform	default = 5. Note that this number may be automatically increased if water updating is selected and there is a significant change in the overall water-structure.

-nsmall	<no SMALL cyc>	Number of SMALL cycles of refinement to perform during each BIG cycle	default = 100.
-R	<reslow> <reshigh>	low- and high-resolution limits for refinement	default = use all data present in MTZ file.
-r	<rms(bond) target>	target value for rms(bond) deviation	used for automatically adjusting X-ray weight; default = 0.010 Å.
-w	<X-ray weight>	Starting X-ray weight	default = take the recorded value from the header of the input PDB file (if it was previously refined with **BUSTER** - otherwise it will start with a value of 4.0). Note that the weight will still be adjusted throughout the run to achieve the desired rms(bond) deviation, as set by the -r flag (or at least get reasonable close to this value). To use a constant X-ray weight, set the desired weight with the -w flag and the parameter AdjustXrayWeightAutomatically to "no".
-Seq	<TNT seq>	TNT sequence file	default = generate automatically from input PDB file using the **pdb2seq** tool. For more complex connectivity, such as covalently bound cofactors, see TNT sequence file section.
-RB	[<rigid.dat>]	Perform rigid-body refinement for one BIG cycle	The default behaviour of -RB is to assign a single rigid body per chain. Specific rigid-body descriptions can be supplied in the optional file. Please see Rigid-body description format for more details. Several -RB flags may be defined (in which case rigid-body refinement will be performed for one BIG cycle for each of the specified rigid-body descriptions in the order given); see Rigid-body usage for more details.
-L		Turns on water updating and uses it to enhance difference density to aid in identification of potential ligand sites with unknown location.	If potential locations are found, they will be described in form of PDB files cluster-<i>.pdb. These are also used to generate thumb-nail pictures of those regions (see file ***analyse.html***). For further information please see ligand chasing procedure (unknown position).
-Lpdb	<PDB file>	Turns on water updating and uses it to enhance difference density to aid in the identification of unmodelled ligands whose location is known.	the location is described by a PDB file which contains "atoms" describing the space of the binding site. Any water atoms placed around the positions defined in this PDB file will be removed prior to the last BIG cycle. For further information please see ligand chasing procedure (known position).

-noWAT	[<ncyc>]	switches off water updating for the first <ncyc> cycles. The default is to switch it off for all cycles.	Since the default is NOT to update waters (see -WAT), this argument only has an effect if -L or -Lpdb is set PREVIOUSLY.
-autoncs		use automatic setup of LSSR-type NCS restraints	Please see NCS restraints section for more details.
-autoncs_noprune		switch off automatic pruning of NCS outliers	Please see NCS restraints section for more details.
-target	<target PDB>	target structure refinement against known, high-quality and/or high-resolution structure using LSSR restraints	See Target restraints.
-sim_swap_equiv		improve the NCS relationship of symmetrical side-chains Asp, Glu, Tyr, Phe and Arg by swapping equivalent atoms.	
-sim_swap_equiv_plus		as -sim_swap_equiv, but also includes Asn, Gln and His.	
-nthreads	<no. of threads>	how many threads to use on multi-CPU/multi-core machines	default is to use a limited number of available threads. See Controlling the number of threads for details. If given a negative parameter, then a fraction of the available threads is used (eg `-2` means to use half the threads and `-4` means to use a quarter of the threads)
-report		run `buster-report` at the end of `refine`	It is important to ensure `buster-report` is correctly setup before using this option. See `buster-report` chapter for details.
-qm	< ligand name and charge> (eg `<LIG+1>`)	Residue type for which to use the quantum energy. Can be given more than once to handle multiple types	BUSTER from the October 2010 release onwards can compute the quantum-mechanical energy of a ligand conformation directly, and use this as part of the objective function in refinement. See AutobusterLigandQM on the wiki for details
Less frequently used command-line arguments:			
-TLS	[<tls.dat>]	do TLS refinement (with optional TLS description)	We would recommend the use of the *-M TLSbasic* macro in the first instance. Please see TLS refinement for more details on the use of TLS refinement.

-Gelly	<file>	file with GELLY-style commands	Use of the -Gelly flag allows expert setting of more complex NCS restraints, target restraints, B-factor groupings and occupancy refinement. Please see the GELLY Manual for more details.
-x	<PDB file>	waters will not be placed around any atoms in this PDB file at any step during the refinement	This has a slightly different effect from the -Lpdb flag! For further information please see ligand chasing procedure (known position: variation).
-autoncs_weight	<number>	weight to use for -autoncs LSSR restraints	default = 2/(no. of ncs chains in the set); see the LIST.html file (with the **BUSTER** run details) for actual value. It is not normally necessary to change the default. However, if -autoncs worsens Rfree, try reducing this weight.
-target_weight	<number>	weight to use for -target LSSR restraints	default = 1.0. It is not normally necessary to change the default. However, if applying target restraint worsens Rfree, try reducing the target weight.
-dlim	<number>	set the convergence limit within each BIG cycle: maximum rmsd distance to starting structure.	default = not set.
-glim	<number>	set the convergence limit within each BIG cycle: maximum value of gradient.	default = 4.0
-special_dist	<number>	Distance in Angstroms used to identify atoms and ions at special positions.	
-B	<B-ref type>	type of B-factor refinement you want to do - one of "individual", "None" or "user".	default = determined automatically by resolution. At higher than 3.5 Å resolution, individual B-factors are refined. Below 3.5 Å, no B-factor refinement is performed. *-B user* must be used in conjunction with any -Gelly command that describes a user-defined B-factor refinement scheme. Please see B-factor refinement for details.
-reportrm		run `buster-report` at the end of `refine` **and remove the original -d directory**	It is important to ensure `buster-report` is correctly setup before using this option. See `buster-report` chapter for details. **Use this option with caution**

Controlling the number of threads used by BUSTER

BUSTER can take advantage of multiprocessor machines, as it includes OpenMP multiprocessing code. By default, the "refine" command will obtain the number of CPU's as reported by the operating system on the machine on which it is run (see below), and will use the number of threads shown in the Table below, unless the environment variable **OMP_NUM_THREADS** is set or the refine argument **-nthreads** is used.

Number of CPU's reported	Default number of threads used by BUSTER
1	1
2	2
3	3
4-23	4
24-63	6
64-	8

The number of CPU's reported by the operating system is determined by running:

```
Linux : % grep -c '^processor' /proc/cpuinfo
Darwin: % /usr/sbin/sysctl hw.ncpu
```

If you want to override this default behaviour, this can be done by setting the environment variable **OMP_NUM_THREADS**, in which case its value will be used in preference to the default. **It should be noted that other applications using OpenMP can be affected by the OMP_NUM_THREADS environment variable so care needs to be taken as conflicts could arise**.

Another way to control the number of threads used by a "refine" job is the nthreads="8" refine parameter. This could be included in a `.autoBUSTER` file but this would seldom be useful.

Finally, use of the "refine" command-line argument **-nthreads** will take precedence over both the default behaviour and the environment variable **OMP_NUM_THREADS**. A positive value <N> is used directly, while a negative value makes **BUSTER** use the fraction: (all available)/<|N|>.

Some information as to how **BUSTER** "refine" scales with number of threads on a 24 CPU machine is available on the BUSTER wiki page BusterShortRefineTest2.

Picture generation with Pymol

To get some final thumbnails (and larger pictures) of the (potential) binding site with various types of density displayed, the graphics program Pymol needs to be installed (and in your path as "pymol"). ImageMagick programs are only used to convert the final pictures into JPEG format.

This is only relevant, if the **-L** or **-Lpdb** flag is used, i.e. **autoBUSTER** tries to detect ligand binding sites. The file ***analyse.html*** will then contain pictures of the (potential) binding site(s).

Automatic restraints generation

If a residue is encountered for which no standard dictionary is found in the Engh & Huber parameter file for proteins (***$BDG_home/tnt/data/protgeo_eh99.dat***) or the distributed DNA/RNA parameter file (***$BDG_home/tnt/data/nuclgeo.dat***), the following logic is used:

1. check the other well-defined dictionary files for co-factors (***$BDG_home/tnt/data/cofactor_geo.dat***), sugars (***$BDG_home/tnt/data/sugar.dat***) and other frequent compounds (***$BDG_home/tnt/data/othergeo.dat***).
2. If the `NeverGenerateDictionary` option is set to `no`, PDB2TNT is used to generate a dictionary based on the current set of coordinates as found in the PDB file. This does not work if the current coordinates for the ligand include hydrogen atoms.

We would strongly recommend that you **do not** turn on the automatic restraints generation, and instead use grade to generate dictionaries. A set of sample coordinates, particularly without hydrogens, is a very bad description of a ligand's chemistry, and there are serious problems with hysteresis over repeated refinements. It is also possible to use quantum-mechanical restraints for a ligand, with the `-qm LIG` option, but a ligand dictionary in CIF format is still required in order to get the atom typing right.

Last modification: 25.04.2014

autoBUSTER Documentation : Examples

Contact buster-develop@GlobalPhasing.com

Contents

Normal refinement

To do a normal refinement only a PDB and MTZ file are needed:

```
% refine -p some.pdb -m other.mtz -d Results.1
```

Results available

The results of an **autoBUSTER** refinement (in the current directory or in the subdirectory pointed to with the "-d" flag) include:

- **refine.pdb**: the final, refined PDB file (including a header section with additional information)
- **refine.mtz**: MTZ file with columns to calculate electron density maps. Use
  - `2FOFCWT/PH2FOFCWT` (2Fo-Fc map)
  - `FOFCWT/PHFOFCWT` (Fo-Fc map)

  It is easy to load these two files e.g. into Coot using

  ```
  % coot --pdb refine.pdb --auto refine.mtz
  ```

- **refine.corr**: tabulated values for real-space correlation of refined model against 2Fo-Fc map
- **analyse.html**: small HTML document with tabulated statistics for each BIG cycle (and thumbnails of potential ligand-binding sites - if -L/-Lpdb options was used).
- **refine_CC-mc_Chain-<ID>.mtv** and **refine_CC-sc_Chain-<ID>.mtv**: graphical plots of main-chain (mc) and side-chain (sc) real-space correlation for each chain <ID>. These can be viewed using `plotmtv` (e.g. in `$BDG_home/helpers/linux/plotmtv`).

Handling of waters

By default, the water structure will not be updated. This might be a good idea at a stage when the protein model has been built and refined and is very close to the final structure. At early stages of refinement (when the macro-molecule is still requiring major manual or automatic rebuilding), the placement of water molecules might not be ideal. On the other hand: if larger parts of the model are still missing, placing these so-called "waters" might indicate to the bulk solvent correction a much better and more realistic envelope. Similarly, towards the end of refinement - when water molecules have been checked manually - this feature should probably be left switched off.

Rigid-body refinement

When the starting model is poor or the cell parameters have changed (e.g. between an apo structure and a compound soak) it is a good idea to first start with some rigid-body refinement. This allows for collective motions that would otherwise take a lot of time or be impossible to achieve within a normal refinement.

- To perform rigid-body refinement use the -RB command line argument. This will set up a single rigid body for each chain and start refinement with a single big cycle of rigid-body refinement (after which it will switch to normal, positional refinement for the subsequent big cycles).
- It is possible to produce custom rigid-body definitions and use them with the `-RB <rigid.dat>` command line argument. See Rigid-body description file format section for their syntax and how to do this.
- We recommend using rigid-body refinement when starting from any molecular replacement structure or where there is a reasonable degree of non-isomorphism between the data and input model.
- During a rigid-body refinement big cycle non-bonded contacts are weighted to zero but bonded contacts continue to be active. This is a good idea as it allows e.g. misplaced loops on the outside of the protein to have short contacts with other chains or to adjacent symmetry copies. Such contacts may be relieved by normal refinement after the initial rigid-body step(s), but there can be problems: particularly for loops that are in close contact to symmetry-related copies of themselves. It is important to check for bad contacts in the screen type output or using the **visualise-geometry-coot** tool after doing a rigid-body refinement.
- Temperature factors are held constant during rigid-body refinement big cycles.
- It is sometimes a good idea to use only low resolution data during the rigid-body refinement cycles. See the Rigid-body description section for details how to do this.

NCS restraints

The recommended way of defining NCS is to start from the initial hypotheses that all copies of the macro-molecule within the asymmetric unit are identical. Only if there are clear indications that parts of one monomer differ from the rest (side-chains in crystal contacts, domain and loop movements, etc) should these parts be taken out of the NCS restraints. Therefore, the procedure to define NCS restraints should start from a completely restrained description that changes during the course of refinement and rebuilding to leave parts of the the molecules out. However, the final NCS restraints should probably still cover between 80-90 % of the atoms in each monomer.

The easiest way to define NCS restraints is using the -autoncs command-line flag. This will apply LSSR-type NCS restraints between all matching chains. It will automatically take care of real differences by removing those from the NCS-relation (so-called "pruning"). If the NCS-relation within the starting structure has been allowed to diverge too much (by over-eager model building into noisy maps or too agressive refinements), it might be a good idea to try and re-instate the NCS-relation. For that the pruning option can be switched off with -autoncs_noprune. This might also be necessary for situations where the X-ray data is rather weak, e.g. at lower resolution. But it depends a lot on the particular problem and especially the modeling history (NCS restraints are not something happening only during refinement, the manual model building also needs to be done under NCS restraints).

Another useful tool is the -sim_swap_equiv flag: this will try and correct problems where NCS-related atoms are chemically identical but have been given different atom names in the PDB files.

B-factor refinement

Under normal circumstances, the mode of B-factor refinement is determined automatically, depending on the resolution. At lower than 3.5 Å resolution the default is to turn off any B-factor refinement, whereas individual atomic B-factors are refined at higher than 3.5 Å.

Previous versions of autobuster used grouped B-factor models at moderate resolution (2.8 - 3.0 Å). However, we have found that with the use of tight BCORREL restraints (as implemented as default in **BUSTER**), use of individual B-factors gives superior results.

Individual B-factor refinement at lower than 3.5 Å resolution, or turning off B-factor refinement at higher than 3.5 Å, can be enforced by use of -B individual or -B None.

The resolution cutoff between these two schemes can be set with the parameter UseBrefNoneFrom.

More complex B-factor refinement modes can be set by use of the -B user option, in conjunction with -Gelly <gelly.file>. As an example, the following command may be used to refine a structure, defining a single B-factor per protein chain.

```
% refine -p some.pdb -m other.mtz -B user -Gelly gelly.dat
```

The gelly.dat file uses gelly combine syntax.

```
NOTE BUSTER_COMBINE B { A|* }
NOTE BUSTER_COMBINE B { B|* }
```

TLS refinement

To enable the use of TLS parametrisation, use the -TLS option of the refine command.

In its simplest invocation use:

```
% refine -p some.pdb -m other.mtz -TLS -d Results.1
```

This will perform TLS refinement for the first big cycle and do regular refinement for subsequent big cycles. If TLS definitions are present in the input pdb file header (both group definitions AND tensors), they will be used. Otherwise, it will define a single TLS group per macro-molecular chain.

Alternatively, use of:

```
% refine -p some.pdb -m other.mtz -TLS tls.dat -d Results.1
```

will similarly do TLS refinement for the first big cycle, but using TLS domain definitions specified in ***tls.dat***.

For convenience two different macros can be used.

- **TLSbasic**

  ```
  % refine -p some.pdb -m other.mtz -M TLSbasic -d Results.1
  ```

  This will switch on TLS refinement for the first and third big-cycle and do regular refinement on the other big-cycles. If TLS definitions are present in the input pdb file header, they will be used (group definitions ONLY). Otherwise, it will define a single TLS group per macro-molecular chain. We would recommend use of *-M TLSbasic* in the first instance.

- **TLSalternate**

  ```
  % refine -p some.pdb -m other.mtz -M TLSalternate -TLS tls.dat -d Results.1
  ```

  Similar to use of -TLS or -TLS tls.dat alone, but will perform (up to 10) alternating cycles of TLS and restrained refinement (starting with TLS). Note that the -TLS option *must* be specified with this macro. Furthermore, this option does not increase the number of big cycles (default is 5). To carry out the full 10 cycles (if wanted) specify -nbig 10.

  This can be especially useful when carrying out additional refinement cycles after small model alterations. The current set of TLS parameters can always be extracted using the pdb2tls tool and that output used as argument to the -TLS flag.

NOTE: Any atoms that are not included in a TLS domain definition will undergo normal restrained refinement.

For a more detailed description of the use of these TLS options please see the TLS tutorial WIKI.

Some ligand is (possibly) present, but location is not well known

The **-L** flag tells the program to remove water atoms around residual difference density at the last cycle. This should make the difference density in these (potentially) 'interesting' regions clearer. The starting PDB file should obviously **not** contain any atoms for the unknown ligand.

```
% refine -p some.pdb -m other.mtz -L -d Results.2
```

The file ***Results.2/analyse.html*** can be used to look at pictures of the found (possible) binding sites.

A ligand is (possibly) present, and the location is well known

If the location of the binding site of a new ligand is known (e.g. from previously solved structures, biochemical data or docking experiments), a PDB file with a model of this (or a similar) ligand can be given with the **-Lpdb** flag. This PDB file should **not** contain the putative ligand as present in the crystal or even a similar structure (the risk of introducing bad model bias would be unacceptably high), but just a collection of atoms that cover the space likely to be occupied by the unknown ligand structure, without highlighting its shape.

This option tells the program to remove waters atoms around this PDB file at the last cycle. This should make the difference density in these 'interesting' regions clearer.

Note : Be careful, when using dummy atoms to describe a large area in space: these atoms are also used to describe the region **not** covered by bulk solvent. So if these dummy atoms are within the bulk solvent region, some artificial difference density will appear (corresponding to the bulk solvent).

```
% refine -p some.pdb -m other.mtz -Lpdb lig-model.pdb -d Results.3
```

The file ***Results.3/analyse.html*** can be used to look at pictures of densities within the user-defined binding sites.

A ligand is (possibly present) in a known location. A variation: excluding regions from bulk solvent during refinement

Use the -x flag to exclude a region described by the provided PDB file from both water addition and bulk solvent region throughout the refinement. This should make the difference density in this region clearer.

However, there is always the danger of creating a biased imprint of the used PDB file in cases where nothing has bound in that site. Under those circumstances, the difference density visible is due to unmodelled bulk solvent (since the region is left out of the bulk-solvent mask). Be careful when decreasing the density level while looking at maps, especially Fo-Fc difference density maps: if one has to go to a level at which there is a lot of difference density all over the remainder of the model, it is unlikely to be significant.

Some settings that might need adjustment

Here are some flags that might need changing:

- **-l <library>**

  If a good-quality geometry dictionary is already available for ligands/compounds that are present in the input PDB file, it is recommended that these are given on the command line (to prevent the automatic generation of geometric restraints based on the current coordinates). Make sure that the residue name is correct and that all atom names match (some modelling programs rename atom names sequentially, so that the coordinates and the dictionary might be out-of-sync).

  In the March 2010 and subsequent releases, -l ligand.cif is the preferred way to present ligand restraints from external generators to **autoBUSTER**. This uses internally a new tool (refmacdict2tnt) that does a good job in converting those restraint dictionaries to the internally used TNT format. In particular, atom-type information from the dictionary will be used when computing the ideal-contact term.

  While developing grade, we found that some other ligand dictionary generators produce torsion terms which do not make chemical sense as restraints. So the internal conversion routine in **autoBUSTER** will generally increase the sigma on torsion terms to deactivate them. Dictionaries produced by **grade** contain a special keyword to indicate to the routine that the torsions are to be believed. If you are completely confident of the torsion terms in your ****.cif*** file, add a line

  ```
  # BUSTER-KEYWORD TRUSTTORS
  ```

  to the file and the torsions will be used as-is.

- **-Gelly <NCS file>**

  If there is more than one copy of a macro-molecule in the asymmetric unit, NCS restraints should be used. In general it seems a good initial assumption that the various copies of a monomer are identical to each other. Only if the density or crystal-contact analysis give clear indications might it be necessary to leave some residues and/or loops out of the NCS restraints. Also, if different domain-orientations can be seen, some fine-tuning in the description of the NCS-relations might be necessary.

  However, completely removing NCS-restraints in case of several monomer-copies per asymmetric unit seems a bad idea and will most likely lead to over-fitting.

  This is now mostly automated by the -autoncs (and related) command-line flags.

- **-WAT [<ncyc>]**

  If the solvent structure of the input PDB file is already very complete, it might be a good idea to leave the automatic update of the water structure switched off. Also, if the input structure is just at the beginning of the refinement (and rebuilding) process, the addition of waters too early in the process might prevent larger parts of the structure from moving. On the other hand, if the structure is fairly incomplete, the interpretation of so-far unexplained density by adding waters might be better than to leave large regions of additional density unmodelled.

  It is difficult to give an easy recipe how to deal with waters (present in the input PDB as well as visible through difference (Fo-Fc) maps). Some experimentation based on the characteristics of each structure/dataset/project is necessary.

  There are several methods available for updating the solvent structure: PKMAPS, PKMAPS with restraints on hydrogen-bonding partners, Coot's findwaters program as well as the possibility of a completely user-defined plugin.

- **-r <rms(bond) target>**

  The value given here is probably a rather complicated way of actually weighting the X-ray and geometric terms relative to each other. Effectively, the X-ray weight will be adjusted so that the rms(bond) value comes out roughly with a value of 0.010. Using only a single criterion for judging the relative weight between X-ray and geometric term is probably not sufficient. Also, the value of 0.010 is most likely not to be correct in a lot of cases (the only reason we came up with this value is that an analysis of the whole PDB gives something very close to this as the mean value in nearly all resolution ranges).

  Note: the whole area of weighting X-ray and geometric term as well as the weighting of the various geometric terms relative to each other will be revisited for the next releases.

- **-RB [<rigid.dat>]**

  If large movements are to be expected (e.g. when refining an apo-structure against a new dataset containing a compound) and the most-likely movements are already well known (active-site loop motion, domain closure, etc ...), it will be good to give one or several rigid-body describing files to **autoBUSTER** containing these rigid-body movements. The command **pdb2rig** can be used to generate (fairly complete) templates for rigid-body descriptions (in GELLY syntax).

- **-B <B-ref type>**

  Sometimes it is a good idea to switch off the default B-factor refinement scheme (-B None), especially at lower resolution and/or at early stages of refinement. In case of very high non-crystalligraphic symmetry it could still be useful to do B-factor refinement even at resolutions lower than the current 3.5 Å cutoff (-B individual).

- **-nbig <no BIG cycle>**

  If one wants to calculate a map very quickly, the following command-line flags could be used:

  ```
  refine -nbig 1 -noWAT ...
  ```

- **-nsmall <no SMALL cycle>**

  The current set of defaults for a refinement using **BUSTER** seem a good compromise for a whole range of refinements. However, for rigid-body refinement of large rigid-bodies, a smaller number of cycles could be used. Also, a larger number of cycles (several hundred) might be able to move much more side-chains into the correct place, even when large rotations/movements are required.

  Note: we're working on better convergence criteria to make these decisions automatically.

Last modification: 23.03.10

autoBUSTER Documentation : Advanced features

Contact buster-develop@GlobalPhasing.com

Contents

The *.autoBUSTER* system of files and advanced command-line syntax

Some advanced features that are not available through command line switches (see also 'refine -h' for a complete list of those) can be set using two mechanisms:

1. (preferred) a command line argument of the form

   ```
   parameter="value"
   ```

2. To change some installation-wide defaults, a file `.autoBUSTER` can be placed into the same directory where the 'refine' binary is placed after installation, e.g. `$BDG_home/autoBUSTER/bin/linux/.autoBUSTER`. A file `$HOME/.autoBUSTER` (to set user-specific options) or `./.autoBUSTER` (to set project-specific options) can also be used. The syntax of these files is:

   - any line starting with hash (`#`) is a comment (and ignored)

- each line has the format:

```
parameter="value"
```

or

```
parameter="value-1 value-2 ... value-N"
```

Additionally, the environment variable `$MyDotAutobuster` can point to a file that will be used on top of the above hard-coded files.

Some of these options are described below: if a refinement doesn't behave as expected, or some additional control is required, please let us know: it is possible that some parameters are already available to do what you need.

The macro feature

To group related sets of parameters and to give easier access to refinement strategies for specific situations, a macro feature has been introduced. This uses simple ASCII text-files of the format

```
# Comment line(s) explaining the purpose
# of this macro
__args="-adding -command -line -arguments"
param1=val1
# other comment (ignored)
param2="valA valB"
```

Notes:

- user-created macros should be placed into (readable) directories, and should have filenames containing only letters and numbers
- those directories need to be set in the environment variable `$autoBUSTER_MacroDirs` (in a form similar to the colon-separate dlist in PATH)
- a list of available macros is printed with the `-M list` command-line flag
- macros are processed at the time they appear in the command-line: so later arguments might override settings from a macro
- it is possible to chain macros, i.e. refering to a macro with a "-M" argument within another macro
- the top comment section is printed as part of the "-M list" output
- the special parameter __args will prepend the given list of arguments to the remaining list of arguments when the macro is processed.

Grouped list of parameters

The list of parameters (sorted alphabetical) is given in Appendix 1.

- Parameters affecting the generation of the TNT sequence file:

  AddMissingSsbondRecords, AdjustBasedOnLinkRecords, AdjustBasedOnLinkRecordsAllowAltloc, AdjustBasedOnLinkRecordsAngleSigma, AdjustBasedOnLinkRecordsBcorrelSigma, AdjustBasedOnLinkRecordsBondCutOffMax, AdjustBasedOnLinkRecordsBondCutOffMin, AdjustBasedOnLinkRecordsBondSigma, AdjustBasedOnLinkRecordsImproperSigma, AdjustBasedOnLinkRecordsMetalsKeep, AdjustBasedOnLinkRecordsMethod, AdjustBasedOnLinkRecordsPlaneSigma, AdjustBasedOnLinkRecordsTrigonalSigma, AdjustFivePrimeEnd, AdjustModifiedAminoAcids, AdjustModifiedNucleotides, AnalyseFivePrimeEnd, AnalyseForModifiedResidues, AnalyseLinkRecords, ExcludeResiduesFromSequence, MaxAllowedCNDistanceInSeq, MaxAllowedOPDistanceInSeq, MinAllowedCNDistanceInSeq, MinAllowedOPDistanceInSeq, SsbondSgDistanceMax, SsbondSgDistanceMin, StandardDictionaries, StandardDictionariesAll, TntDictionary_connect, UseGapAsBreakInSeq.

- Parameters affecting the various checks performed:

  PdbChk_AdditionalChecksToDo, PdbChk_AtomNameUnsupportedCharacters1, PdbChk_AtomNameUnsupportedCharacters2, PdbChk_AtomNamesAgainstStandardRestraintsExclude, PdbChk_AtomNamesAgainstStandardRestraintsWarning, PdbChk_ChecksNotToDo, PdbChk_FixAtomNamesOfResidues, PdbChk_MaxNumToPrint, PdbChk_MaximumCellAngle, PdbChk_MaximumCellEdge, PdbChk_MaximumCellVolume, PdbChk_MinimumCellAngle,

PdbChk_MinimumCellEdge, PdbChk_MinimumCellVolume, PdbChk_PossibleChainIds, PdbChk_RecordFormats, PdbChk_TooShortRecordsList, PdbChk_WrongReferenceToCoordinateRecordError, PdbStandardResidues, RemoveScaleCardsFromPdb, RenumberIfBelow, ReuseSequenceFile, RmAnisou, RmLink, RmModres, RunGellySanityCheck, RunGellyScreen, StandardDictionaries, StandardDictionariesAll, UseMtzchk, UsePdbchk, WaterChainId, WaterNamingAtom, WaterNamingResidue, WaterResidueNames.

- Parameters affecting the refinement strategy:

  AdjustXrayWeightAutomatically, AllowBrefInRigidBody, AutomaticRestrictLowres, AutomaticRestrictLowresBinCut, AutomaticRestrictLowresCcCut, AutomaticRestrictLowresFromCycle, BusterCrdMlscalKeyword, BusterExe, BusterExtraArgs, BusterGellyKwd, BusterRigidBodyBimpfFrgLowResCut, BusterRigidBodyBimpfFrgNeverRefine, DoRigidIfCellDiffer, FixXyz, FormfactorCorrection, KeepCurrentRmsBond, KeepHydrogens, KeepZeroOcc, LastCycleBsolv2Bmiss, LastCycleKsolv2Kmiss, LastCycleRefineBmiss, LastCycleRefineKmiss, MxlcycCutBuster, NoOverallBanisoRefinement, PassThroughArgs PassThroughArgsUser ReuseSequenceFile, ScreenNumBuster, ScreenSigmaBuster, ScreenSigmaInitial, StopOnGellySanityCheckError, StopOnMissingContactDistance, TntBfacMax, TntBfacMin, TntWeightGeomRes, UseBrefGroupFrom, UseBrefMcScFrom, UseBrefNoneFrom, UseCrdScaleAfterRigid, UseHighResInRigid, UseLlgradAsFoFc, UseLowResInRigid, UseMapAsNup, UseMapAsSlv, UseMaxEntLastCycle, UseMaxEntThroughout, UseNmissThroughout, blkblr, blkrad, frgrad, mskblr, mskisl, mskrad, mxlcyc_start, nmiss, refocc, refscl_rfr, refscl_rfs, refscl_rif, refscl_rir, refscl_ris, refscl_rkim, refscl_rkis, solc, weight_max, weight_min, weight_start.

- Parameters affecting the analysis for potential, bound ligands:

  AnaPdbmapsCut1, AnaPdbmapsCut2, AnaPdbmapsMinVol, AnaPdbmapsPadding, AnalyseBusterFoFc, AnalyseClusterMethod, LigandDescribingPdbMethod, UseEpdbLastCycle, UseLlgradAsFoFc, UseLpdbLastCycle, UseMxlcycLastCycle, UseNmissLastCycle.

- Parameters affecting solvent structure (water) updating:

  DoWaterRemoveDeleted, KeepAddingWatersAfterN, UpdateWaters, UseSortwater, WaterChainId, WaterFindSigma, WaterFindSigmaLlg, WaterMinDistance, WaterNamingAtom, WaterNamingResidue, WaterPickingOptimise, WaterRemoveDeleted, WaterRemoveDistFac, WaterRemoveMerge, WaterRemoveSigma, WaterResidueNames, WaterUpdateFftResMin, WaterUpdateProgram.

- Parameters affecting handling of geometric restraints:

  AddModifiedAminoAcidToBusterSet, AdjustBasedOnLinkRecords, AdjustBasedOnLinkRecordsAllowAltloc, AdjustBasedOnLinkRecordsAngleSigma, AdjustBasedOnLinkRecordsBcorrelSigma, AdjustBasedOnLinkRecordsBondCutOffMax, AdjustBasedOnLinkRecordsBondCutOffMin, AdjustBasedOnLinkRecordsBondSigma, AdjustBasedOnLinkRecordsImproperSigma, AdjustBasedOnLinkRecordsMetalsKeep, AdjustBasedOnLinkRecordsMethod, AdjustBasedOnLinkRecordsPlaneSigma, AdjustBasedOnLinkRecordsTrigonalSigma, AdjustXrayWeightAutomatically, AnalyseGellySanityCheckForDuplicateBonds, DicFromPdbAllAtomsInBond, ExcludeBadContacts, KeepCurrentRmsBond, MaxAllowedCNDistanceInSeq, MaxAllowedOPDistanceInSeq, MinAllowedCNDistanceInSeq, MinAllowedOPDistanceInSeq, RunGellySanityCheck, StandardDictionaries, StandardDictionariesAll, TntDictionary_assume, TntDictionary_bcorrel, TntDictionary_cofactor, TntDictionary_connect, TntDictionary_contact, TntDictionary_csdx, TntDictionary_nuclgeo, TntDictionary_othergeo, TntDictionary_pdbfixup, TntDictionary_sugar, TransferExoticAAFromSeqToGelly, UseAutomaticDicts, UseAutomaticDictsCcp4, UseAutomaticDictsMsd, UseDictionaryOrder, GeometryWeight_angle, GeometryWeight_bcorrel, GeometryWeight_bond, GeometryWeight_chiral, GeometryWeight_contact, GeometryWeight_improper, GeometryWeight_ncs, GeometryWeight_plane, GeometryWeight_pseudo, GeometryWeight_torsion, GeometryWeight_trigonal. GeometryWeight_ideal.

- Parameters affecting handling of internal cavities (voids):

  AnaVoids_dist_and, AnaVoids_dist_and_fac, AnaVoids_dist_not, AnaVoids_dist_not_fac, AnaVoids_rmss, AnalyseVoids, AnalyseVoidsAlways, AnalyseVoidsLast.

- Parameters affecting final analysis:

  AnalyseExtraEpdbs, AnalysePictureCarve, AnalysePictureLarge, AnalysePictureLevel_2FoFc, AnalysePictureLevel_FoFc, AnalysePictureSmall, do_analyse, do_maps, FinalMapsCoverPdb, FinalMapsNormalized.

- Parameters affecting creation of final PDB file:

AssumePdbFromRefinerCorrect, RunCor2Pdb, RunHarvest, RunSfcheck.

Some parameters most likely to be of interest

This list is probably not complete (see above for a nearly complete list): if you feel the need for doing something slightly different, please get in contact with us (we might be already able to show you how this could be done).

1. There are two programs available for adding/removing/updating waters: arp_waters (from the CCP4 suite - especially written for updating of the water/solvent structure) or pkmaps (from **SHARP/autoSHARP**). The default (for the moment) is to use pkmaps (which might be better when the goal is to explain positive difference density through 'waters').

   There is also the possibility to use an external program/script for this task: in that case the absolute path to this program/script should be given. This script will be given the arguments

   1. current PDB file
   2. MTZ file for 2Fo-Fc map (columns 2FOFCWT/PH2FOFCWT)
   3. MTZ file for Fo-Fc map (columns FOFCWT/PHFOFCWT)
   4. (sub-)directory for this run (e.g. for writing additional or temporary output to)
   5. name of output PDB file (should be a copy of the input PDB file as much as possible, with only the waters updated)
   6. (optional) PDB file with coordinates of waters that have already been removed in previous steps

   The parameter WaterUpdateProgram can be either set to "PKMAPS", "ARP_WATERS" or the absolute path of a user-supplied program/script.

   The levels for adding removing waters can be controlled using the parameters WaterFindSigma, WaterRemoveSigma and WaterRemoveMerge

   To completely switch off the update of solvent structure (or delay if for a few initial BIG cycles) see the -noWAT command line switch.

2. The resolution limits where different B-factor refinement schemes are used can be set with the parameters UseBrefNoneFrom, UseBrefGroupFrom and UseBrefMcScFrom. To enforce a specific B-factor refinement scheme, the command line flag -B can be used.

   The parameters to -B are

individual	Refine one B-factor per atom. This is almost always the right option to use
group-mcsc	Refine one B-factor shift for each main chain (N, C, CA, O) and another for the side chain
group-res	Refine one B-factor shift for each residue
None	Do not change the B-factors from the input file

   Note that BUSTER refines B-factor **shifts** within groups; it is therefore important to use the InitialiseBiso option when using grouped B refinement. Either pick an initial B-factor that you think appropriate, or use `InitialiseBiso=wilson`.

   If a grouped B-factor refinement scheme is selected, the restraint in temperature factors of bonded atoms (BCORREL) could be switched off using the GeometryWeight_bcorrel parameter.

3. The various (relative) weights on geometric restraints can be set with the parameters GeometryWeight_bond, GeometryWeight_angle, GeometryWeight_torsion, GeometryWeight_plane, GeometryWeight_trigonal, GeometryWeight_chiral, GeometryWeight_contact, GeometryWeight_bcorrel, GeometryWeight_ncs, GeometryWeight_improper, GeometryWeight_pseudo and GeometryWeight_ideal.

4. To change the starting X-ray weight (and associated minimum and maximum allowed values), use the XrayWeight_start, XrayWeight_min and XrayWeight_max parameters. **autoBUSTER** will automatically adjust the X-ray weight, unless told not to do so.

5. GELLY has its own set of additional command-line arguments that can be added using the BusterExtraArgs parameter.

6. controlling output formatting can be done by setting the environment variable `$autoBUSTER_NO_HIGHLIGHT` to a non-empty value. This will supress the use of certain escape sequences to create bold, underline or italic characters. Please

note that this needs to be set as an environment variable, rather than an option on the `refine` command line!

Last modification: 24.01.11

autoBUSTER Documentation : Additional tools

Contact buster-develop@GlobalPhasing.com

Contents

---

checkdeps check that all 3rd party tools needed work properly.

This is a utility that will check programs in the BUSTER suite in turn. `checkdeps` makes sure that all the required 3rd party tools are installed, available and function properly. Problems are indicated on lines starting "ERROR". If no problems are found then this is shown by "SUCCESS". The utility prints out a summary of results found at the end. The script's exit status will be 0 for success but 1 if any problem is found

currently `checkdeps` runs:

- `refine -h` to check licence is OK.
- `grade -checkdeps`
- `grade_PDB_ligand -checkdeps`
- `hydrogenate -checkdeps`
- `buster-report -checkdeps`

For help in configuring the software including advice on how to use `checkdeps` see the detailed installation instructions..

`checkdeps` command line option:

Parameter	Options	Explanation	Remark
-n		turn off the prompt for user to hit the `Enter` key before running each check	

---

corr - calculate real-space correlation

This tool allows the easy calculation of real-space correlation between a model (PDB file) and a map (usually a 2Fo-Fc map). The normal use is e.g.:

```
% corr -p refine.pdb -m refine.mtz -F 2FOFCWT -P PH2FOFCWT
```

which will produce overall and per-residue correlation coefficients on standard output as well as some PLOTMTV-formatted files of main-chain and side-chain correlation plots (e.g. named ***refine_CC-mc_Chain-A.mtv***).

Flag	Arguments	Explanation	Remark
-p	<PDB file>	PDB file with standard CRYST1 card	
-m	<MTZ|MAP file>	MTZ or MAP file	MTZ file with columns for F, PHI and (optionally) WEIGHT or MAP file in CCP4 format
-F	<F>	amplitude	
-P	<PHI>	phase	
-Fc	<Fcalc>	(optional) amplitude of model	default is to calculate structure factors of model from input PDB file (which will then not contain bulk-solvent correction or anisotropic scaling)
-Pc	<PHIcalc>	phase of model	
-a	<atom name>	rename atoms to this name	done before the CC calculation
-d	<subdir>	directory name	results are expected in this sub-directory and all files will be created there too
-R	<resl> <resh>	low- and high-resolution limits	MTZ file: default is to use full resolution range from this file
-W	<WEIGHT>	(optional) weight	usual coefficients (2FOFCWT, PH2FOFCWT) are already correct map coefficients, so this doesn't need to be given

---

gelly_refine - interface to GELLY (geometric refinement)

This is a simple interface to the stand-alone version of GELLY, which will do purely geometric refinement (i.e. no X-ray term involved). Therefore, this command can be used to

- idealise/regularise a structure, e.g. the PDB file after some manual model building, against the Engh & Huber set of parameters.
- check the correctness of a geometric restraints dictionary

Flag	Arguments	Explanation	Remark
-f		force overwriting of files	default= stop if a file would be overwritten
-p	<PDB file>	PDB file to be refined	
-o	<output file>	output PDB file	
-d	<subdir>	all (temporary) o/p will be written to directory	default = current directory
-l	<TNT dictionary>	additional TNT dictionary files	several -l flags can be given; default is to use the standard dictionaries distributed with BUSTER-GELLY-TNT/**autoBUSTER**
-s	<space-group>	space-group name	default = pick from CRYST1 card of PDB file
-c	<cell parameters>	cell parameters a, b, c, alpha, beta, gamma	default = pick from CRYST1 card of PDB file

-Seq	<TNT sequence file>	TNT sequence file	default = create on-the-fly from input PDB file
-I	<Identifier>	automatically generated files will start with the string <Identifier>	default = "gelly"
-jiggle_xyz	<rms>	adds a random perturbation (jiggle) to all input atoms before starting refinement	The size of this perturbation is given as a mean rms deviation (default is to not jiggle)

Any command-line options not in the above list will be passed directly to the `gelly` binary; see GELLY for a list of useful options, and a couple of usage examples.

Additionally, the following parameters are defined (which can be overwritten on the command line, using the `parameter=value` syntax):

Parameter	Default	Explanation	Remark
weight_bond	2.0	bond distances	
weight_angle	2.0	bond angles	
weight_improper	0.0	improper angles	
	2.0		if all residues in input PDB file are described by user-supplied dictionary files (via the -l flag)
weight_torsion	0.0	torsion angles	
	2.0		if all residues in input PDB file are described by user-supplied dictionary files (via the -l flag)
weight_pseudo	0.0		
	2.0		if all residues in input PDB file are described by user-supplied dictionary files (via the -l flag)
weight_trigonal	2.0		
weight_plane	5.0	planarity	
weight_contact	5.0	contact distances	
weight_bcorrel	0.0	B-factor correlation of bonded atoms	
weight_chiral	5.0	chirality	

---

graph_autobuster_recipCC view the reciprocal-space correlation coefficient plot

This is a utility that locates the last reciprocal-space correlation coefficient plot produced by autoBUSTER during a refinement and launches plotmtv to view it. For help on its use see BUSTER Output Interpretation page on the BUSTER wiki. For help with the command options use:

Parameter	Options	Explanation	Remark
-h		Print brief help message	
-man		Print man page for full description	

---

graph_autobuster_R produce a graph that shows how Rwork and Rfree change during a refinement

This is a utility that allows the production of a graph that shows how Rwork and Rfree change during a refinement. For help on its use see BUSTER Output Interpretation page on the BUSTER wiki. For help with the command options use:

Parameter	Options	Explanation	Remark
-h		Print brief help message	
-man		Print man page for full description	

---

graph_autobuster_QM produce a graph that shows how the QM energy for a ligand changes during a refinement

This is a utility to be used with `-qm` option of BUSTER. For help on its use see Direct use of weighted Quantum Chemical Energy for ligands page on the BUSTER wiki. For help with the command options use:

Parameter	Options	Explanation	Remark

-h		Print brief help message	
-man		Print man page for full description	

---

hydrogenate - add hydrogen atoms to protein and/or ligands

This is a tool for adding hydrogen atoms to proteins and/or ligands; it requires 'reduce' program (distributed as part of the MolProbity suite) to be on the PATH or to be defined using the `$BDG_TOOL_MOLPROBITY_ROOT` environment variable.

Parameter	Options	Explanation	Remark
-checkdeps		Check that all the dependencies are present	Special option that checks that the external tools required (`reduce`) have been setup properly. This option is one of the tests run by the `checkdeps` script.
-p	<input filename>	Protein to hydrogenate	
-o	<output filename>	Name for the output file	
-l	<dictionary1.cif> <dictionary2.cif> ...	List of CIF-format dictionaries for the ligands	`hydrogenate` writes out a list of the residue IDs it was unable to hydrogenate; you will want to provide dictionaries for most of them (though obviously not metals); grade_PDB_ligand will be helpful for this.
-ligonly		Only hydrogenate the ligands	
-zero		Insert hydrogens with zero occupancy	
-f		Overwrite the output if it already exists	

---

mk_coot_macros.sh - generate macros to use with Coot

This is a simple script to be run in the **autoBUSTER** output directory (i.e. where the refine.pdb file is). It will create a file ***Coot.scr*** that can be used in Coot:

```
% mk_coot_macros.sh
% coot --script Coot.scr
```

See also visualise-geometry-coot - launch coot to see BUSTER refinement result

---

mk_pymol_macros.sh - generate macros to use with Pymol

For Pymol, this script will generate a file ***pymol.pml*** to be used like this:

```
% mk_pymol_macros.sh
% pymol pymol.pml
```

---

pdb2seq - generate TNT sequence from PDB

If a TNT sequence file is needed (e.g. when running gelly_refine), this command will generate it for you.

Please note that you can't use standard output (captured in a file) directly as a TNT sequence file. If you want to create a file please use the -o command line argument.

Flag	Arguments	Explanation	Remark
-p	<PDB file>	PDB file following the recommendations	
-o	<output file>	(optional) output file for TNT sequence	default is standard output

By default chain breaks in the input PDB file will be converted into `BREAK` statements in the resulting sequence file. If the parameter **UseGapAsBreakInSeq** is set to yes (on the command line: `UseGapAsBreakInSeq=yes`), then a so-called GAP-residue is used instead. The effect is that a range-definition (e.g. for defining a rigid-body) can 'step over' a GAP-residue but not over a `BREAK`.

---

pdbchk - check (and optionally fix) PDB files

This tool can be used to make sure a PDB file conforms to most of the PDB format standards as well as some slightly more stringent requirements for **BUSTER** and **autoBUSTER**.

Flag	Arguments	Explanation	Remark
-p	input file	PDB formatted coordinate file	
-o	output file	(optional) PDB formatted coordinate file	the presence of this optional argument triggers functionality within "pdbchk" that will try and fix any encountered problems of the input file

The list of tests performed (in this order) is:

Test (name)	Explanation	Fixing
NoCryst1	checking if we're missing CRYST1 record	-
Cell	checking for cell parameters on CRYST1 record	-
NoSpacegroup	checking if CRYST1 doesn't contain a spacegroup	-
Spgr	checking for spacegroup name on CRYST1 record	-
EmptyLines	checking for empty records	-
HaveCoordinateRecords	checking if we have any coordinate records	-
RecordsStartingWithSpace	checking if we have any records starting with a space	-
SeveralModels	checking if PDB file contains several models	-
WeirdCellParameters	checking if cell parameters on CRYST1 are weird	-
WeirdCellVolume	checking if cell volume (from CRYST1 record) is weird	-
BarSpacegroup	checking if spacegroup symbols has 'bar' (e.g. P -1/P 1-)	change spacegroup symbol (e.g. from "P 1-" to "P -1"
R3H3	checking if R3/R32/R3m/R3c is meant to be H3/H32/H3m/H3c	change spacegroup symbol (e.g. from "R 3" to "H 3"
UnknownSpacegroup	checking if spacegroup name is unknown	-
CellSpacegroupInconsistency	check if cell and spacegroup are consistent	-
UnknownTntSpacegroup	checking if for given spacegroup we have a TNT equivalent	-
RecordsStandardOrder	checking if records are in standard order	records will be reordered according to PDB Format (up to CRYST1 record)

RecordFormat	checking if some crucial records have correct format	-
SsbondIsCys	checking if SSBOND records contain only CYS residues	-
ResidueNumbersOnRecordsAreInteger	check if residue numbers on records are Integer	re-write residue numbers as integers on records SEQADV, MODRES, HET, SSBOND, CISPEP, LINK, SLTBRG, HYDBND, SITE, ATOM and HETATM
ResidueNumberInsertionCodeFive	checking if residue number > 999 and insertion code present (TNT limitation)	-
EmptyAtomNameOnLinkRecord	check if LINK records contain empty atom names (in both positions)	remove those LINK records
WrongReferenceToCoordinateRecord	checking for wrong references to coordinate records	-
NoChainId	checking for ATOM/HETATM records without chain identifier	add new chain ID to records without one (this includes the following records: DBREF, SEQADV, SEQRES, MODRES, HET, SSBOND, LINK, HYDBND, SLTBRG, CISPEP, SITE, ATOM, SIGATM, ANISOU, SIGUIJ, TER and HETATM)
OxyResidueName	checking if there are residues called "OXY" (special treatment in TNT)	residues will be renamed from "OXY" to " O2" (if the "OXY" residue contains atoms " O1 " and " O2 ")
DuplicateChainRes	checking for ATOM/HETATM records where the same chainID+resSeq+iCode is used for different resName	if possible, adding chain ID "W" to water residues (residue name "HOH")
StandardResiduesHetatm	checking if standard residues have (wrong) HETATM record	change record from HETATM to ATOM
NonStandardResiduesAtom	checking if non-standard residues have (wrong) ATOM record	change record from ATOM to HETATM
BfactorNegative	checking if ATOM/HETATM records have negative B-factors	set B-factor to zero
OccRange	checking if ATOM/HETATM records have occupancy in range 0.0 ... 1.0	limit occupancy to range zero to one
AlternateConformationsOccSum	checking if alternate conformations of ATOM/HETATM records have an occupancy sum in range 0.0 ... 1.0	-
AtomNamesWithSpaces	checking if atom names have space in them	replaces spaces by underscore "_"
ElementType	checking if element type is present and consistent with atom name	guesstimate element from atom name

---

seq2seq - generate TNT sequence from ASCII file

To convert simple ASCII files with sequence information (FASTA, PIR etc), this tool can be used. It recognised all 20 amino-acids (so Se-MET containing proteins need editing of the resulting TNT sequence file).

Flag	Arguments	Explanation	Remark
-s	ASCII sequence file	file with (upper-case) protein sequence	
-i	ResNumStart	starting residue number	default = 1
-c	ChainId	1-character chain identifier	default = " "

---

pdb2dpi - calculate various versions of the "diffraction-component precision index"

Using the information recorded in the REMARK section of a PDB file, this tool will calculate various versions (based on R or Rfree) of the diffraction-component precision index as dedfined by Cruickshank and Blow.

Flag	Arguments	Explanation	Remark
-p	PDB file		

---

pdb2occ - generate template for refining occupancy from PDB file

Simple script to generate some Gelly-syntax statements for occupancy refinement from a given PDB file. It analyses residues with alternate conformation indicators (column 17) as well as residues with occupancies lower than one. Some assumptions about a sensible PDB format are made.

Consecutive residues with alternate conformations and same occupancy will be grouped together. If only two alternate conformations are given for a residue, then their summed occupancy will be restrained to 1.0.

For further details on how to use pdb2occ and how to perform occupancy refinement see the occupancy refinement tutorials on the BUSTER wiki.

Flag	Arguments	Explanation	Remark
-p	PDB file		
-o	output file	optional	

---

pdb2tls - extract TLS information from PDB file

Flag	Arguments	Explanation	Remark
-p	PDB file		
-o	output TLS file	optional	
-t	format type	type of format. Either 'BUSTER' or 'REFMAC' (default is 'BUSTER')	
-a	autotype	use automatic definition for BUSTER. The automatic definition type can be one of "EachMacroMolChain" or "OnePerChain". Default is "EachMacroMolChain".	

See TLS refinement section for further information.

---

refmacdict2tnt - convert REFMAC dictionary to TNT format

This program converts a REFMAC-style cif restraint dictionary to TNT format, preserving atom-type information which is used by the Gelly ideal contact term.

The typical usage would be:

```
% refmacdict2tnt <REFMAC restraint file> <TNT output file> [<PDB output file]
```

Note that autoBUSTER can usually handle cif restraint dictionaries directly if you pass them using the `-l` flag; if you find yourself routinely converting them manually, please contact `buster-develop@globalphasing.com` and we will try to make your work-flow easier.

Note that the flags for refmacdict2tnt must go before the filenames

Flag	Explanation	Remark
-nopdb	Don't extract atom-position information from the input .cif file	If you don't use this option, you need to specify a filename for the PDB output

-believetorsions	Preserve sigma values when translating torsion cards in the input	
-notorsions	Ignore all torsion cards in the input	
-oneplane	Do not output an extra, dehydrogenated version of any plane containing hydrogens	**If *any* atom in a plane is missing then BUSTER will not apply that plane restraint at all** - so if your input dictionary has large planes containing hydrogens, and you are refining a model lacking hydrogens, you must use `-oneplane`
-fixplanesigma	Tweak sigma values for planes so that the TNT and REFMAC geometry functions give identical values	
-tlc XXX	Set three-letter code to use for the single ligand in the CIF file	
-model abc.pdb	Convert only ligands which appear in `abc.pdb` with a HETSYN card containing a synonym of the form `+id`; use the three-letter code that appears in that HETSYN card.	This option (introduced in early 2012) is intended to make it easier to work with compound libraries without having to worry about unique three-letter codes for each ligand

---

visualise-geometry-coot - launch coot to see BUSTER refinement result

this is a useful way of quickly launching coot to the view the results of a BUSTER refinement. It should launch coot (that must be on your path) and load the final refine.pdb structure together with maps from the mtz file. In addition a listing of the worst geometry violations is displayed. Click on this to jump to the atoms in question.

For help on its use see BUSTER Output Interpretation page on the BUSTER wiki.

The procedure is run

```
%  visualise-geometry-coot <autoBUSTER refinement directory>
```

---

diff_fourier - calculate (and analyse) various types of difference Fourier maps

Introduction

We will described a tool to calculate different types of difference Fourier maps. We will not be dealing here with the normal difference ("Fo-Fc") or "2Fo-Fc" map that is used in model refinement and building, but rather with maps that use differences between measured amplitudes.

Running the tool

Running

```
%  diff_fourier -h
```

should bring up a help message.

Upon successful running, the script will create several output files - the prefix of which can be set with the `-o` flag. Other potentially useful flags (for full details see output of `-h`):

- `-keepmap`: to keep the calculated map file (CCP4 format, which can be loaded into Coot directly)
- `-R <resl> <resh>`: to set resolution limits (eg. restricting to only data with anomalous signal)
- `-negative`: to also look for negative peaks (but then the `-pdb` option has no effect)

Anomalous difference Fourier map

Running

```
%  diff_fourier -m truncate.mtz -p refine.mtz -P PH2FOFCWT FOM -o AnoFourier
```

will

- use anomalous differences in file `truncate.mtz` (default: first D/Q column pair - e.g. DANO/SIGDANO)
- use phases `PH2FOFCWT` and weight `FOM` from file `refine.mtz`
- calculate an anomalous Fourier map and produce output files with the prefix "AnoFourier"

If a PDB file (consistent with the phases) is also given with

```
%  diff_fourier -m truncate.mtz -p refine.mtz -P PH2FOFCWT FOM -o AnoFourier -pdb refine.pdb
```

then

- the found peaks will be placed close to the PDB model
- the found peaks will be compared to atoms present in the PDB file

An example output looks like this:

```
================================================================================

 mtz ..................................... truncate.mtz
 F ....................................... F
 SIGF .................................... SIGF
 DANO .................................... DANO
 SANO .................................... SIGDANO

 pmtz .................................... refine.mtz
 PHI ..................................... PH2FOFCWT
 FOM ..................................... FOM

 pdb ..................................... refine.pdb

...
   7 peaks above 20 sigma
   9 peaks above 15 sigma
  11 peaks above 10 sigma
  11 peaks above  8 sigma
  12 peaks above  6 sigma
  12 peaks above  5 sigma
  37 peaks above  4 sigma

-rw-r--r-- 1 vonrhein vonrhein  2132 Oct 10 15:29 AnoFourier.ANO.compare
-rw-r--r-- 1 vonrhein vonrhein 13940 Oct 10 15:29 AnoFourier.ANO.hatom
-rw-r--r-- 1 vonrhein vonrhein 24715 Oct 10 15:29 AnoFourier.ANO.pdb
```

AnoFourier.ANO.compare:

```
Peak         Closest atom in refine.pdb
[rms]                                                 Distance (<= 1.0 )
--------------------------------------------------------------------------
 31.23  <=>  SE   MSE F   7  (  0.84 40.87)  :       0.07
 30.91  <=>  SE   MSE A   7  (  0.84 45.76)  :       0.04
 30.22  <=>  SE   MSE A 126  (  0.66 45.08)  :       0.08
 29.08  <=>  SE   MSE F 126  (  0.66 40.55)  :       0.13
 23.72  <=>  SE   MSE F 137  (  0.73 42.17)  :       0.06
 22.10  <=>  SE   MSE A 137  (  0.73 45.81)  :       0.13
 21.10  <=>  SE   MSE F 293  (  0.88 70.46)  :       0.27
 18.64  <=>  SE   MSE F 139  (  0.58 47.16)  :       0.32
 16.20  <=>  SE   MSE A 293  (  0.88 93.55)  :       0.43
 14.81  <=>  SE   MSE A 139  (  0.58 53.66)  :       0.26
 11.24  <=>  SE   MSE F   1  (  0.56 72.94)  :       0.19
  7.26  <=>  SE   MSE A   1  (  0.56 92.71)  :       0.49
  4.10  <=>   O   THR A 161  (  1.00 43.14)  :       0.92
  3.81  <=>   CB  THR F 261  (  1.00 65.06)  :       0.58
...
```

AnoFourier.ANO.hatom:

```
ATOM Se -0.0623 -0.0435  0.3244    31.23
ATOM Se  0.0630 -0.0264 -0.2195    30.91
ATOM Se -0.0761  0.0141 -0.0840    30.22
ATOM Se  0.0776  0.0031  0.1880    29.08
ATOM Se -0.0028 -0.1375  0.2705    23.72
ATOM Se  0.0042 -0.1241 -0.1671    22.10
ATOM Se -0.0712  0.2201  0.1354    21.10
ATOM Se -0.0261 -0.1020  0.3066    18.64
```

```
ATOM Se  0.0787  0.2230 -0.0277    16.20
ATOM Se  0.0204 -0.0827 -0.2023    14.81
ATOM Se -0.3329 -0.1845  0.3639    11.24
ATOM Se  0.3373 -0.1699 -0.2602     7.26
ATOM Se  0.0752 -0.1320  0.2399     4.56
...
```

AnoFourier.ANO.pdb:

```
CRYST1   62.827   90.075  191.529  90.00  90.00  90.00 P 21 21 21
SCALE1      0.015917  0.000000  0.000000        0.00000
SCALE2      0.000000  0.011102  0.000000        0.00000
SCALE3      0.000000  0.000000  0.005221        0.00000
ATOM    182  C   DUM     1      -3.916  -3.917  62.136  1.00 31.23   11
ATOM    136  C   DUM     2       3.955  -2.381 -42.043  1.00 30.91   11
ATOM    313  C   DUM     3      -4.783   1.274 -16.088  1.00 30.22   11
ATOM     24  C   DUM     4       4.875   0.282  36.013  1.00 29.08   11
ATOM    172  C   DUM     5      -0.178 -12.385  51.807  1.00 23.72   11
ATOM    170  C   DUM     6       0.264 -11.178 -32.014  1.00 22.10   11
ATOM    319  C   DUM     7      -4.476  19.827  25.928  1.00 21.10   11
ATOM    173  C   DUM     8      -1.639  -9.191  58.728  1.00 18.64   11
ATOM     33  C   DUM     9       4.943  20.085  -5.303  1.00 16.20   11
ATOM    154  C   DUM    10       1.282  -7.447 -38.744  1.00 14.81   11
ATOM    281  C   DUM    11     -20.916 -16.621  69.699  1.00 11.24   11
ATOM     62  C   DUM    12      21.190 -15.308 -49.836  1.00  7.26   11
ATOM    133  C   DUM    13       4.726 -11.886  45.946  1.00  4.56   11
...
```

So we have

- a list of fractional coordinates for anomalous peaks eg. for input into experimental phasing with SHARP/autoSHARP
- a PDB file with those anomalous peaks eg. for visualisation in Coot
- a comparison of those peaks with a PDB file (to check that all strong anomalous peaks are correctly explained in the model)

Fo-Fo Difference map

If two sets of amplitudes are available, a difference Fourier map can be calculated with something like

```
%  diff_fourier -m apo.mtz -p apo_refine.mtz -P PH2FOFCWT FOM -m2 inhibitor.mtz -o IsoFourier -pdb
apo_refine.pdb -noANO -compare_cut 10.0
```

which

- uses the first amplitude/sigma (F/Q) pair from `apo.mtz`
- and the first amplitude/sigma (F/Q) pair from `inhibitor.mtz`
- plus the phases from the refined apo-model (in MTZ file `apo_refine.mtz`)
- to calculate a `F_inhibitor-F_apo` map
- compare the found peaks within 10A of existing model atoms

```
================================================================================

 mtz ..................................... apo.mtz
 F ....................................... FP
 SIGF .................................... SIGFP
 DANO .....................................
 SANO .....................................

 pmtz .................................... apo_refine.mtz
 PHI ..................................... PH2FOFCWT
 FOM ..................................... FOM

 pdb ..................................... apo_refine.pdb

 mtz2..................................... inhibitor.mtz
 F2 ...................................... FP
 SIGF2 ................................... SIGFP

...
   0 peaks above 20 sigma
   0 peaks above 15 sigma
   0 peaks above 10 sigma
   2 peaks above  8 sigma
```

```
   3 peaks above  6 sigma
   5 peaks above  5 sigma
  20 peaks above  4 sigma

-rw-r--r-- 1 vonrhein vonrhein  1846 Oct 10 15:56 IsoFourier.ISO.compare
-rw-r--r-- 1 vonrhein vonrhein  6068 Oct 10 15:56 IsoFourier.ISO.hatom
-rw-r--r-- 1 vonrhein vonrhein 10891 Oct 10 15:56 IsoFourier.ISO.pdb
```

This will show positive peaks where data in `inhibitor.mtz` predicts density that is absent in `apo.mtz`, eg. for an inhibitor:

IsoFourier.ISO.compare:

```
 Peak         Closest atom in apo_refine.pdb
 [rms]                                                Distance (<= 10.0 )
-----------------------------------------------------------------------
   9.37  <=>   O   HOH A 501  (  1.00 27.89)  :       1.97
   8.72  <=>   NZ  LYS A  89  (  1.00 43.51)  :       0.87
   6.85  <=>   O   HOH A 505  (  1.00 44.68)  :       2.09
   5.99  <=>   O   HOH A 505  (  1.00 44.68)  :       1.68
   5.48  <=>   O   HOH A 508  (  1.00 41.07)  :       2.34
   4.85  <=>   CB  LYS A  89  (  1.00 30.25)  :       2.54
   4.47  <=>   CG2 ILE A 186  (  1.00 12.12)  :       1.45
...
```

If we had already a model of the inhibitor and used that PDB file instead:

```
%  diff_fourier -m apo.mtz -p apo_refine.mtz -P PH2FOFCWT FOM -m2 inhibitor.mtz -o IsoFourier -pdb
inhibitor.pdb -noANO
```

we would get IsoFourier.ISO.compare:

```
Peak         Closest atom in inhibitor.pdb
 [rms]                                                Distance (<= 1.0 )
-----------------------------------------------------------------------
   9.37  <=>   C10 DT4 A1299  (  1.00 38.54)  :       0.32
   8.72  <=>   S1  DT4 A1299  (  1.00 54.82)  :       0.31
   6.85  <=>   N5  DT4 A1299  (  1.00 43.09)  :       0.54
   5.99  <=>   C15 DT4 A1299  (  1.00 47.69)  :       0.81
   5.48  <=>   N7  DT4 A1299  (  1.00 43.13)  :       0.68
   4.85  <=>   CD  LYS A  89  (  1.00 43.87)  :       0.56
   4.32  <=>   NZ  LYS A  33  (  1.00 41.01)  :       0.68
   4.26  <=>   C   PRO A 171  (  1.00 30.06)  :       0.84
   4.02  <=>   C4  DT4 A1299  (  0.75 45.81)  :       0.85
...
```

showing us the peaks being very close to the inhibitor.

---

ana_diffmap_residue - analyse difference map around specific residues

This little tool analyses difference maps around residues in a model. The residues can be either given by the user (as residue name or specified through chain and residue number) or the program will use all non-standard residues within the PDB file.

The output could be useful to get a quick and automatic idea about the amount of difference density features around specific residues (like co-factors, active-site residues or ligands).

A typical usage could be (see also help messages with the "-h" flag):

```
% ana_diffmap_residue -p refine.pdb -m refine.mtz
```

---

fetch_PDB - fetch coordinates and reflection data from local or online PDB archive (and convert reflection data to MTZ format)

This script will fetch the deposited atomic coordinates and reflection data from a local or online PDB archive. The reflection data will be converted into MTZ format (using the CCP4 program http://www.ccp4.ac.uk/dist/html/cif2mtz.html after appropriate checks and clean-ups on the deposited mmCIF file).

A large number of additional checks and analysis are carried out - eg to inform the user about inconsistencies between

- REMARK 3 (refinement) and REMARK 200 (data collection) items
- REMARK 200 items and the deposited reflection data

If a local copy of the PDB archive is available, the environmental variable `BDG_TOOL_LOCALPDBDIR` can be set to the full path of this directory (it expects to then find `$BDG_TOOL_LOCALPDBDIR/data/structures/all/`).

The typical usage for PDB identifier "1ABC" would be:

```
% fetch_PDB 1ABC
```

which will create an output directory (`1ABC`) and report basic statistics for the deposited structure and the resulting MTZ reflection file.

References

- Blow, D. (2002). Rearrangement of Cruickshank's formulae for the diffraction-component precision index. Acta Cryst. D58, 792-797
- Cruickshank, D. W. J. (1999). Remarks about protein structure precision. Acta Cryst. D55, 583-601.

Last modification: 22.01.2014

autoBUSTER Documentation : integration with coot

Contact buster-develop@GlobalPhasing.com

Contents

---

visualise-geometry-coot

The `visualise-geometry-coot` tool is intended as the principal way of visualising the results of a BUSTER refinement; it provides several lists of different kinds of geometry outliers, which in our experience tend to correspond to places where the model can easily be tweaked to fit better into the density.

You can invoke it either by `visualise-geometry-coot` when you're in a directory containing a BUSTER refinement, or `visualise-geometry-coot <directory name>`.

If you close the geometry-outliers window, select 'Geometry issues ...' from the 'BUSTER' menu to reopen it.

The BUSTER button

When running visualise-geometry-coot, a 'BUSTER' menu is added to the coot menu bar.

Using the BUSTER button

When you select 'Launch a BUSTER job' from the menu, you get a window allowing you to fill in

- Which molecule you want BUSTER to refine
- The MTZ file to refine against (this is filled in automatically in most cases). Note that this should be the output of the data-processing stage, rather than the `refine.mtz` file from an earlier refinement.
- Any extra dictionaries to use (again, this is filled in automatically in most cases)
- The refinement protocol:
  - Do you just generate a map, or refine the geometry somewhat?
  - 'void correction': if refining geometry, should you do a second pass which mitigates the issue of getting negative difference density in very hydrophobic parts of the molecule? (default is yes)
  - If refining geometry, should you use automatic NCS? (default is yes)

Click the 'Start BUSTER' button to start the job. The command-line output from BUSTER will appear in the terminal window from which you started coot, and a progress window will appear which indicates how far BUSTER has got with the refinement.

At the end of the refinement, assuming it's successful, the refined molecule and map are loaded into the coot window

Installing the BUSTER button in your usual coot

Some users will have coot set up to load extension modules from a particular directory; if you add a link from that directory to `$BDG_home/scripts/buster-button.scheme` then coot will start by default with the BUSTER menu present.

When is it appropriate to use the BUSTER button?

The present form of the BUSTER button allows you to invoke one of three standard macros: `MapOnly`, `ShortRun` and `ShortRunVoid`. The first of these does not do any refinement, and is appropriate to use on any structure.

But the `ShortRun` macros turn off a number of the features of BUSTER in order to run more quickly; they are appropriate for doing a small amount of geometry optimisation (if, for example, you have sorted out some misplaced side-chains in a structure, or a dubious conformation of a ligand), but it is not sensible to use the `ShortRun` macros on structures which have not already been through `BUSTER`.

Last modification: 21.04.11

autoBUSTER Documentation : buster-report

Contact buster-develop@GlobalPhasing.com

Contents

Using `buster-report`

`buster-report` is a tool for providing clear reports about the progress of and the results from a run of BUSTER. It includes `mogul`-based reporting of the geometric properties of the ligands in the output file, and a `molprobity` analysis of the protein geometry including unusual-rotamer information.

The report is intended to contain enough information about the refinement that it would be possible to delete the refinement directory and keep the report; for example, it contains the output PDB and MTZ files, the full contents of the CIF dictionaries given on the command line, and enough information about geometry outliers that you can run `visualise-geometry-coot` on an output directory from `buster-report`.

`buster-report` command-line options

Note that options can be abbreviated provided there is no ambiguity created.

Option	Arguments	Explanation	Remarks
`-h`		Display usage information	Special option to print help message and exit
`-checkdeps`		Check that all the dependencies are present	Special option that checks that the external tools buster-report needs are accessible and work properly. Useful for setting up buster-report and testing that the program works on a particular host. This option is one of the tests run by the `checkdeps` script.
`-d`	`<BUSTER refinement directory>`	The `refine -d` directory on which to produce a report.	This is the only option that must be specified.
`-dr`	`<output directory>`	The place to put the report	optional, by default report directory name will be based on the BUSTER refinement directory with `-report` added to it.
`-ligand`	`XXX,XXY,XXZ`	A comma-separated list of only the ligands you want to see reports on	If you use both `-ligand` and `-boring` then the `-boring` request will be ignored
`-boring`	`NAD,FAD`	Specify a comma-separated list of the three-letter codes of ligands not to report on in addition to the defaults. For instance use `-boring NAD,FAD` to add NAD and FAD to list of "boring" ligands.	default list of three-letter codes regarded as boring: HOH, MSE, PO4, SO4, EDO, EOH, GOL, FMT, ACT, ACE, CIT, BOG, MPD, TAM, BTB, EPE, MES, PIN, DMS, DTT, 15P, PG4, PE5, DA, DC, DG, DT
`-interesting`	`MSE,GOL`	Specify a comma-separated list of ligands that should be removed from boring list.	For instance to report on MSE and GOL use `-interesting MSE,GOL`

-title	<title>	The title to display on the report.	This title will be used at the top of the report as well as appearing in both the html and pdf browser toolbars. The default title is Report on BUSTER refinement run in directory followed by the directory as specified in -d
-dname	<name>	The name to use for the .pdb and .mtz files in the report	Default name is the directory as specified in -d. This option is useful to give files recognisable names in the coot display manager window, particularly when using visualise-geometry-coot.
-f		Overwrite the output directory if it already exists	If you do not specify -f then buster-report will not overwrite an existing directory but instead will terminate with an error.
-delete		Delete the input directory if buster-report runs without error	If you find yourself entirely happy with buster-report output then you might want to use this option to save some disc space. Please note that buster-report is being improved.
-nopdf		Do not produce a PDF version report as well as the HTML.	The same thing can be achieved by setting environment variable $BDG_TOOL_PDFLATEX to none
-pdf		Produce a PDF version of the report	This option is now redundant as this is the new default.
-nomogul		Do not do any Mogul analysis of the final ligand geometry.	The same thing can be achieved by setting environment variable $BDG_TOOL_MOGUL to none
-nopic		Do not draw pictures of ligands	Do not draw any pictures of ligand density or ligand outliers. The same thing can be achieved by setting environment variable $BDG_TOOL_PYMOL to none
-nolig		Do not do any ligand analysis.	No ligand analysis will be done. This is a more drastic option than -nomogul or -nopic.
-nomp		Do not do the MolProbity analysis	The same thing can be achieved by setting environment variable $BDG_TOOL_MOLPROBITY_ROOT to none.
-png		Use PNG format rather than SVG for graphs	SVG graphs look considerably better in Firefox but do not display correctly in some versions of Internet Explorer
-pyray		Run correctly with certain older versions of pymol	If the -pyray option is needed, buster-report will display an warning message advising you to use it

External tools used by buster-report

buster-report uses a number of programs (tools) to produce its report, some are optional but others must be installed for buster-report to run. buster-report will first then check whether a tools location is defined by the relevant environment variable has been defined. If the environment variable is not defined then the tool will be found from the user's $PATH. Tools provided by the operating system will automatically be added to the user's $PATH and are best provided in this way. It is recommend that other tools are defined using environment variables are these are set in the files $BDG_home/setup_local.sh and $BDG_home/setup_local.csh as explained in the detailed installation instructions..

To check whether the external tools used by buster-report are properly setup then use:

```
% buster-report -checkdeps
```

The following table describes each of the external tools used by buster-report

. . .

Program	required or optional?	Environment Variable	Remarks
ImageMagick convert	required	$BDG_TOOL_CONVERT can be set to the full path for the convert executable.	Normally provided by an operating system supplied package and so convert will normally be found from the user's $PATH.

ImageMagick `identify`	required	`$BDG_TOOL_IDENTIFY` can be set to the full path for the `identify` executable.	Normally provided by an operating system supplied package and so `identify` will normally be found from the user's `$PATH`.
Ghostscript `ps2pdf`	required	`$BDG_TOOL_PS2PDF` can be set to the full path for the `ps2pdf` executable.	Normally provided by an operating system supplied package and so `ps2pdf` will normally be found from the user's `$PATH`.
`xmgrace` `gracebat`	required	`$BDG_TOOL_GRACEBAT` can be set to the full path for the `gracebat` executable.	Normally provided by an OS supplied package and so `gracebat` will normally be found from the user's `$PATH`. Some recent Ubuntu versions have `gracebat` that produce mangled xml `buster-report -checkdeps` will detect these. See URL for details.
CCDC mogul	optional: turn off with argument `-nomogul` or by setting `$BDG_TOOL_MOGUL` to `none`	`$BDG_TOOL_MOGUL` should be set to the full path for the mogul executable or `none`. Also used by `grade` (optional) and `grade_PDB_ligand` (optional). `$BDG_MOGUL_LOCAL_DATABASE_FILE` can be set to provide support for Mogul with additional in-house libraries	`mogul` is used to check ligand geometry against CSD small molecule structures. `buster-report -checkdeps` should be used to check that the mogul licence works. To get the licence working run mogul interactively and fill in the licence information. There have been some reports of issues using mogul from initial 2014 release of Cambridge Structural Database System (CSDS) if CSDS is installed on a NFS-mounted file system, see https://www.globalphasing.com/buster/wiki/index.cgi?SoftwareMogulRelease2014NFSissues. See below for support for Mogul with additional in-house libraries
Open Babel obabel	optional: turn off by setting `$BDG_TOOL_OBABEL` to `none`	`$BDG_TOOL_OBABEL` should be set to the full path for the obabel executable or `none`. Also used by `grade` (optional) and `grade_PDB_ligand` (optional).	`obabel` is used to generate 2D coordinates for ligands used in 2D schematic pictures. Versions 2.3.0 and 2.3.1 work. obabel is supplied by some operating systems.
MolProbity	optional: turn off with argument `-nomp` or by setting `$BDG_TOOL_MOLPROBITY_ROOT` to `none`	`$BDG_TOOL_MOLPROBITY_ROOT` should be set to the full path of the root directory of the MolProbity installation or `none`. The root directory of the MolProbity installation must contain the files: `cmdline/reduce-nobuild`, `cmdline/multichart` and `lib/hless.jar` Also used by `hydrogenate` (required).	`buster-report` will run MolProbity in a batch mode. Note that there is no need to install apache as `buster-report` does not use the web interface. Tested with both the new 4.02b version of MolProbity available from http://molprobity.biochem.duke.edu/ and the old 3.19 http://helix.research.duhs.duke.edu/. Make sure that you have configured MolProbity with the `setup.sh` script supplied with MolProbity as per the instructions.
java	optional: but needed by MolProbity, turn off with argument `-nomp` or by setting `$BDG_TOOL_MOLPROBITY_ROOT` to `none`	`$BDG_TOOL_BUSTERREPORT_JAVA` can be set to the full path for the java executable (unless it is on the `$PATH`).	Normally java will be on the users `$PATH`. Note that `gij` cannot be used (`buster-report` will check for this and stop with an ERROR message if `gij` is used)
pymol	optional: turn off with argument `-nopic` or by setting `$BDG_TOOL_PYMOL` to `none`	`$BDG_TOOL_PYMOL` can be set to the full path for the `pymol` executable (unless it is on the `$PATH`).	Most OS's provide a `pymol` package. `buster-report` works will all versions tested.
latex	optional: turn off with argument `-nopdf` or by setting `$BDG_TOOL_PDFLATEX` to `none`	`$BDG_TOOL_PDFLATEX` can be set to the full path for the `latex` executable.	Most OS's provide a `latex` package and if this is used then `pdflatex` will be on the users `$PATH`.

Support for Mogul with additional in-house libraries

In late 2014 CCDC provided to selected corporate users a facility to prepare additional libraries for Mogul containing information taken from in-house databases of small molecule structures. If you have this facility then these libraries can be used by `buster-report`, `grade` and `grade_PDB_ligand`. To do this prepare a file containing `Mogul` instructions to use the libraries following this template:

```
# YourCompanyName private Mogul database DD-MMM-YYYY using NNNNN structures.
MOGUL DATA LIBRARY /path/to/library/
MOGUL DATA DATABASE /path/to/database/file
MOGUL DATA CSD ON
```

Please include an informative comment as the first line of the file as this will be included in `buster-report`, `grade` and `grade_PDB_ligand`

output. Once you have prepared the file specify its location (including the full file path) by environment variable `$BDG_MOGUL_LOCAL_DATABASE_FILE` as described above.

Last modification: 29.04.2014

autoBUSTER Documentation : References

Contact buster-develop@GlobalPhasing.com

References

- Branden C. and Jones A. (1990). *Nature* **343** 687-689.
- Bricogne, G. (1993). Direct Phase Determination by Entropy Maximisation and Likelihood Ranking: Status Report and Perspectives. *Acta Cryst.* D**49**, 37-60.
- Bricogne, G. (1997). The Bayesian Statistical Viewpoint on Structure Determination: Basic Concepts and Examples, in *Methods in Enzymology*, **276**A, 361-423. C.W. Carter & R.M. Sweet, eds.
- Brünger, A. T. (1992). The Free R value: a Novel Statistical Quantity for Assessing the Accuracy of Crystal Structures. *Nature* **355**, 472-474.
- Collaborative Computational Project, Number 4 (1994). The CCP4 Suite: Programs for Protein Crystallography. *Acta Cryst.* **D50**, 760-763.
- DeLano, W.L. (2002). The PyMOL User's Manual, DeLano Scientific, San Carlos, CA, USA.
- Kabsch W. (1976). *Acta. Cryst.* **A32** 922-923.
- Murshudov,G. N., Vagin, A. A. and Dodson, E. J. (1997). Refinement of Macromolecular Structures by the Maximum-Likelihood Method. *Acta Cryst.* **D53**, 240-255.
- Roversi, P., Blanc, E., Vonrhein, C., Evans, G. and Bricogne, G. (2000). Modelling prior distributions of atoms for Macromolecular Refinement and Completion. *Acta Cryst.*, D**56**, 1313-1323.
- Ten Eyck, L. F. (1973). Crystallographic fast Fourier transforms. *Acta Cryst.* A**29**, 183-192.
- Ten Eyck, L. F. (1977). *Acta Cryst.* A**33**, 486.
- Tronrud, D. E., Ten Eyck, L. F., & Matthews, B. W. (1987). An Efficient General-Purpose Least-Squares Refinement Program for Macromolecular Structures. *Acta Crystallogr A*, **43**, 489-501.
- Tronrud, D. E. (1992). Conjugate-Direction Minimization - An Improved Method for the Refinement of Macromolecules. *Acta Crystallogr A*, **48** (November), 912-916.
- Tronrud, D. E. (1996). Knowledge-Based B-Factor Restraints for the Refinement of Proteins. *J App Cryst*, **29** (2), 100-104.
- Tronrud, D. E. (1997). The TNT Refinement Package. in Macromolecular Crystallography, Part B, Eds Charlie Carter, and Robert Sweet, Volume 277 in Methods in Enzymology, pp 306-319.
- Tronrud, D. E. (1999). The Efficient Calculation of the Normal Matrix in Least-Squares Refinement of Macromolecular Structures. *Acta Crystallogr A*, **55**, 700-703.

- Vagin, A. A. and Isupov, M. N. (2001). Spherically averaged phased translation function and its application to the search for molecules and fragments in electron-density maps.*Acta Cryst.* **D57**, 1451-1456.

Last modification: 11.06.09

autoBUSTER Documentation : Appendix 1

Contact buster-develop@GlobalPhasing.com

Alphabetical list of parameters

Most parameters are given in the table below. However, for some tools/steps there could be additional parameters available: if you want to change a specific behaviour please contact us.

Parameter [program] step affected	default	
AddMissingSsbondRecords [pdb2seq] generation of TNT sequence file	"no"	When automatica specified): add a
AddModifiedAminoAcidToBusterSet [pdb2seq] generation of TNT sequence file	"yes"	When automatica distributed set of these residues.
AdditionalAnalysisAfterBuster [run_buster] running BUSTER	"yes"	Report more info
AdjustBasedOnLinkRecords [pdb2seq] generation of TNT sequence file	"yes"	Make use of LIN are not covered b
AdjustBasedOnLinkRecordsAllowAltloc [pdb2seq] generation of TNT sequence file	"yes"	When adding ad allow atom defin
AdjustBasedOnLinkRecordsAngleSigma [pdb2seq] generation of TNT sequence file	"5.0"	Default sigma va records of input l
AdjustBasedOnLinkRecordsBcorrelSigma [pdb2seq] generation of TNT sequence file	"20.0"	Default sigma va records of input l
AdjustBasedOnLinkRecordsBondCutOffMax [pdb2seq] generation of TNT sequence file	"2.5"	Only LINK reco used to generate
AdjustBasedOnLinkRecordsBondCutOffMin [pdb2seq] generation of TNT sequence file	"1.2"	Only LINK reco used to generate
AdjustBasedOnLinkRecordsBondSigma [pdb2seq] generation of TNT sequence file	"0.04"	Default sigma va records of input l
AdjustBasedOnLinkRecordsIgnoreResidues [pdb2seq] generation of TNT sequence file	"MSE"	List of residues t for those anyway
AdjustBasedOnLinkRecordsImproperSigma [pdb2seq] generation of TNT sequence file	"5.0"	Default sigma va records of input l
AdjustBasedOnLinkRecordsMetalsKeep [pdb2seq] generation of TNT sequence file	""	list of (space-sur when a metal ato PDB file). By de the BOND and A is not recommen
AdjustBasedOnLinkRecordsMethod [pdb2seq] generation of TNT sequence file	"PDB2TNT"	When defined as two atoms refere (recommended), description is do
AdjustBasedOnLinkRecordsPlaneSigma [pdb2seq] generation of TNT sequence file	"0.02"	Default sigma va records of input l
AdjustBasedOnLinkRecordsTrigonalSigma [pdb2seq] generation of TNT sequence file	"0.020"	Default sigma va LINK records of
AdjustBoundaryResiduesInRigidBodyDefinition [refinetools] creating a rigid-body definition file	"yes"	For a residue ran are actually pres
AdjustFivePrimeEnd [pdb2seq] generation of TNT sequence file	"yes"	Take extra steps
AdjustModifiedAminoAcids [pdb2seq] generation of TNT sequence file	"YES"	When encounteri normal peptide g those are amino-
AdjustModifiedNucleotides [pdb2seq] generation of TNT sequence file	"yes"	When encounteri normal sugar-ph assume those are

AdjustXrayWeightAutomatically [refine] overall refinement (BIG cycles)	"yes"	The weight betw (restraints) will b KeepCurrentRms value for that we
AdjustXrayWeightPrecision [refine] overall refinement (BIG cycles)	"4"	When adjusting t what precision (i value?.
AdjustXrayWeightSignificantChange [refine] overall refinement (BIG cycles)	"33"	What change (gi constitutes a sign BIG cycles.
AllowBrefInRigidBody [run_buster] refinement during each BIG cycle	"no"	Should we do B-
AnaBusterIterSkip [ana_buster] after each BIG cycle	"1"	Report statistics iteration will alw
AnaBusterVerbosity [ana_buster] after each BIG cycle	"0"	Verbosity level f information.
AnaPdbmapsCut1 [various] detection of potentially bound ligands	"3.0"	minimum density potential bound l
AnaPdbmapsCut2 [various] detection of potentially bound ligands	"1.0"	minimum density a potential bound
AnaPdbmapsMinVol [various] detection of potentially bound ligands	"50.0"	minimum volum ligand
AnaPdbmapsPadding [various] detection of potentially bound ligands	"5.0"	safety border [Å]
AnaVoids_ClusterSize [anavoids]	"5.0 10.0"	If using the "ana of cluster sizes to
AnaVoids_dist_and [anavoids] handling of voids	"3.00"	during detection selected peaks
AnaVoids_dist_and_fac [anavoids] handling of voids	"1.0"	during detection during looping o
AnaVoids_dist_not [anavoids] handling of voids	"1.50"	detection of void
AnaVoids_dist_not_fac [anavoids] handling of voids	"0.9"	detection of void looping over Ana
AnaVoids_method [anavoids]	"orig"	one of "orig" (ba density regions)
AnaVoids_rmss [anavoids] handling of voids	"3.5 3.0 2.5 2.0"	detection of void
AnalyseBusterFoFc [analyse] final analysis	"FOFCWT"	column name in
AnalyseClusterMethod [analyse] final analysis	"new"	Which method to highly recommer
AnalyseExtraEpdbs [analyse] final analysis	""	List of PDB files the automatically
AnalyseFivePrimeEnd [pdb2seq] generation of TNT sequence file	"yes"	Check if residues - which would re
AnalyseForModifiedResidues [pdb2seq] generation of TNT sequence file	"yes"	Should we analy
AnalyseGellySanityCheckForDuplicateBonds [run_buster] refinement during each BIG cycle	"yes"	After running the if Gelly encounte
AnalyseLinkRecords [pdb2seq] generation of TNT sequence file	"yes"	Should we analy the TNT sequenc
AnalysePictureCarve [analyse] final analysis	"3.0"	When generating describing PDB
AnalysePictureLarge [analyse] final analysis	"800,800"	Size of large pict
AnalysePictureLevel_2FoFc [analyse] final analysis	"1.0"	Density level [rm
AnalysePictureLevel_FoFc [analyse] final analysis	"3.0"	Density level [rm
AnalysePictureSmall [analyse] final analysis	"100,100"	Size of small pic

AnalyseVoids [refine] overall refinement (BIG cycles)	"yes"	Should we try an are probably not empty)?
AnalyseVoidsAlways [refine] overall refinement (BIG cycles)	"no"	Usually, the anal every BIG cycle.
AnalyseVoidsLast [refine] overall refinement (BIG cycles)	"yes"	Should the analy very last BIG cyc
AnalyseVoidsStopOnError [refine]	"no"	Should we stop v stop - better to gi void-correction).
AssumePdbFromRefinerCorrect [refine] overall refinement (BIG cycles)	"yes"	Is the PDB file c processing)?
AutomaticFormfactorCorrection [run_buster]	"no"	When encounteri wavelenght: adju Usually, the wav if data was proce
AutomaticFormfactorCorrectionAnalyse [run_buster]	"yes"	Analyse element AutomaticFormf
AutomaticRestrictLowres [refine] overall refinement (BIG cycles)	"yes"	Should we try an correlation CC(F
AutomaticRestrictLowresBinCut [refine] overall refinement (BIG cycles)	"0.5"	During analysis bin should be exc
AutomaticRestrictLowresCcCut [refine] overall refinement (BIG cycles)	"0.0"	During analysis restriction of the
AutomaticRestrictLowresFromCycle [refine] overall refinement (BIG cycles)	"2"	During analysis analysis be starte
BusterCrdMlscalKeyword [run_buster] refinement during each BIG cycle	"MLSCAL"	Used to fine-tune
BusterExe [run_buster] refinement during each BIG cycle	"$BDG_bin/buster"	Full path of BUS
BusterExtraArgs [run_buster] refinement during each BIG cycle	""	Extra command- one would want
BusterFreeFlagValue [run_buster] refinement during each BIG cycle	"0"	Set value of Free
BusterGellyKwd [run_buster] refinement during each BIG cycle	"GELLY=1"	Used to fine-tune
BusterReportCmd [refine]	"buster-report"	if BusterReportR (see also BusterR $subdir -drep
BusterReportDir [refine]	""	Into which direct default will be "r argument). The u end will be with end.
BusterReportRun [refine]	"no"	Determines if "bu flag.
BusterRigidBodyBimpfFrgLowResCut [run_buster] refinement during each BIG cycle	"4.0"	Resolution cut-o refined when run
BusterRigidBodyBimpfFrgNeverRefine [run_buster] refinement during each BIG cycle	"no"	Should the imper refinement?
ColumnName_FreeR_flag [refine] Start of refinement	"FreeR_flag"	Column name in
ColumnName_FreeR_flag_allowed [refine] Start of refinement	" I FreeR_flag| I FreeRflag| I FREE| I R-free-flags"	List (|-separated)
Cor2Pdb_FixAtomNamesOfResidues [cor2pdb] converting TNT-formatted cor file to PDB	"FAD|NAP|NAI|COA|NDP|NAD|AP5|CAA|NAH|ACO"	for which residue
Cor2Pdb_FixHydrogenAtomNames [cor2pdb] converting TNT-formatted cor file to PDB	"yes"	Should we try an
Cor2Pdb_FixResidueNameRightJustified [cor2pdb] converting TNT-formatted cor file to PDB	"yes"	When generating
Cor2Pdb_RenameWat [cor2pdb] converting TNT-formatted cor file to PDB	"yes"	Should we renam
CorrEnforceSfcalc [corr] calculation of real-space correlations	"no"	by default, "corr" this parameter se file instead.

CorrHighResScale [corr] calculation of real-space correlations	"1.0"	To get a finer gri value greater tha
CorrMainChainAtoms [corr]	" N | CA | C | O "	bar-separated list against columns of residues in the CorrMainChainR
CorrMainChainReset [corr]	"no"	Should we ensur CorrMainChainR classified as mai classified as side
CorrMainChainResidues [corr]	"ALA|CYS|ASP|GLN|PHE|GLY|HIS|ILE|LYS|LEU|MET|ASN|PRO|GLU|ARG|SER|THR|VAL|TRP|TYR|MSE"	bar-delimited list CorrMainChainA ATOM/HETAT
CorrMainChainResiduesAdd [corr]	""	bar-delimited list CorrMainChainR
CorrMainChainResiduesDel [corr]	""	bar-delimited list CorrMainChainR
CorrMtvPrefix [corr] calculation of real-space correlations	"CC"	Prefix for files cr
CorrRemapRes [corr]	"yes"	remap residues t existing chain id
CorrRunSeparateChains [corr] calculation of real-space correlations	"no"	When calculating each chain? This bookkeeping of g
CuKa [run_buster]	"1.54180"	default waveleng
DicFromPdbAllAtomsInBond [ab_pdb2tnt] generation of TNT restraints	"yes"	Should we check coordinates has e
DiffFourier_CompareCut [diff_fourier]	"1.0"	distance (includi map and (optiona
DoCleanupAfterBuster [run_buster] refinement during each BIG cycle	"yes"	Should autoBUS BIG cycle?
DoCleanupAfterBusterCleanHtml [run_buster] refinement during each BIG cycle	"no"	When removing corresponding H
DoRigidIfCellDiffer [refine] overall refinement (BIG cycles)	"yes"	If the cell parame autoBUSTER pe
DoWaterRemoveDeleted [maptools] various steps involving maps	"yes"	When updating w waters that have adding/deleting t
ExcludeBadContacts [run_buster] refinement during each BIG cycle	""	Shortcut to exclu http://www.uoxr separated list of
ExcludeResiduesFromFetching [ab_pdb2tnt]	""	(|-separated) list are already dealt
ExcludeResiduesFromSequence [pdb2seq] generation of TNT sequence file	"HOH|OXY"	Residue names (when creating th
ExcludeWatersCloseDist [pkmapstools] water updating	"2.5"	Minimum distan
ExcludeWatersCloseMapDiss [pkmapstools] water updating	"3.0 2.5 2.0 5.0"	A list of distance
ExcludeWatersCloseMapRmss [pkmapstools] water updating	"6.0 5.0 4.0 0.0"	A list of cut-off l
ExcludeWatersClusterCloseDist [refine]	"2.5"	After having defi clusters for poter regions.
FftMapMinHighResLimit [prep_rhofit/refine]	"1.5"	maps will be calc go that far): this
FinalMapsCoverPdb [refine] presentation of results	"no"	Should the final (refine.pdb)?
FinalMapsNormalized [refine] presentation of results	"yes"	Should the final of zero and a rms
FinalResultsLinkOnly_mtz [refine] presentation of results	"no"	Should we create copying it?
FixXyz [various] refinement	"no"	should all atom p
FormfactorCorrection [run_buster] refinement during each BIG cycle	""	Space separated (<Fprime>) of a formfactor calcul

FormfactorCorrectionMinDiff [run_buster]	"0.1"	minimum differe wavelength) befo applied.
FixPotentiallyWrongEndTargetsForOxt [pdb2seq] generation of TNT sequence file from PDB file	"no"	Should we try an oxygens (OXT)?
FixedSolventScales [run_buster] during each BIG cycle	"no"	Should we keep resolution-depen deposited PDB s
FixedSolventScales_K_SOLV [run_buster] during each BIG cycle	(calculated with formulae below)	Solvent scale fac
FixedSolventScales_K_SOLV_m [run_buster] during each BIG cycle	"0.163"	For calculating K
FixedSolventScales_K_SOLV_b [run_buster] during each BIG cycle	"0.456"	For calculating K
FixedSolventScales_B_SOLV [run_buster] during each BIG cycle	(calculated with formulae below)	Solvent scaling B
FixedSolventScales_B_SOLV_m [run_buster] during each BIG cycle	"32.56"	For calculating B
FixedSolventScales_B_SOLV_b [run_buster] during each BIG cycle	"-31.91"	For calculating B
FixedSolventScales_B_IMPF_SOLV [run_buster] during each BIG cycle	(calculated with formulae below)	Solvent scaling i
FixedSolventScales_B_IMPF_SOLV_m [run_buster] during each BIG cycle	"-34.34"	For calculating B
FixedSolventScales_B_IMPF_SOLV_b [run_buster] during each BIG cycle	"114.68"	For calculating B
FixedSolventScales_K_IMPF_SOLV [run_buster] during each BIG cycle	1.0	Solvent imperfec
GeometryWeight_angle [run_buster] refinement during each BIG cycle	"2.0"	Weight for geom
GeometryWeight_bcorrel [run_buster] refinement during each BIG cycle	"20.0"	Weight for geom default is set to z
GeometryWeight_bond [run_buster] refinement during each BIG cycle	"2.0"	Weight for geom
GeometryWeight_chiral [run_buster] refinement during each BIG cycle	"5.0"	Weight for geom
GeometryWeight_contact [run_buster] refinement during each BIG cycle	"5.0"	Weight for non-b
GeometryWeight_defaults [run_buster.sh] refinement during each BIG cycle	"2.0 2.0 0.0 2.0 0.0 5.0 2.0 5.0 5.0 20.0 50.0 4.0"	list of defaults fo
GeometryWeight_ideal [run_buster] refinement during each BIG cycle	"2.0"	Weight for geom
GeometryWeight_improper [run_buster] refinement during each BIG cycle	"0.0"	Weight for geom
GeometryWeight_names [run_buster] refinement during each BIG cycle	"bond angle improper torsion pseudo plane trigonal chiral contact bcorrel ncs ideal"	identifier for vari
GeometryWeight_ncs [run_buster] refinement during each BIG cycle	"50.0"	Weight for NCS
GeometryWeight_plane [run_buster] refinement during each BIG cycle	"5.0"	Weight for geom
GeometryWeight_pseudo [run_buster] refinement during each BIG cycle	"0.0"	Weight for geom
GeometryWeight_torsion [run_buster] refinement during each BIG cycle	"2.0"	Weight for geom
GeometryWeight_trigonal [run_buster] refinement during each BIG cycle	"2.0"	Weight for geom
HarvestCalculateDpi [harvest] creation of REMARK 3 section	"all"	Which "diffracti
IncludeNonStandardInitialRemarkSectionInFinalPdb [refine] creation of final PDB file	"yes"	Should we includ final PDB file?

InitialiseBiso [refine] overall refinement (BIG cycles)	"no"	Should we initial "yes", "no", "wil
InitialiseBisoFallback [refine] overall refinement (BIG cycles)	"20.0"	If the original int failed: what valu
KeepAddingWatersAfterN [refine] overall refinement (BIG cycles)	"yes"	After the initial r should we keep g
KeepAdjustingXrayWeightAfterN [refine] after each BIG cycle	"no"	Should we increa significantly?
KeepCurrentRmsBond [refine] overall refinement (BIG cycles)	"no"	Try to adjust the value of the inpu
KeepHydrogens [various] handling of user-supplied PDB file	"yes"	should hydrogen
KeepHydrogensBabslv [run_buster] handling of PDB file for bulk solvent masking	"no"	should hydrogen
KeepHydrogensNup [run_buster] handling of PDB file for non-uniform prior	"no"	should hydrogen
KeepZeroOcc [various] handling of user-supplied PDB file	"yes"	should atoms wit
KeepZeroOccBabslv [run_buster] handling of PDB file for bulk solvent masking	"no"	should atoms wit mask?
KeepZeroOccNup [run_buster] handling of PDB file for non-uniform prior	"no"	should atoms wit prior mask?
LastCycleBsolv2Bmiss [run_buster] refinement during each BIG cycle	"no"	When using miss be initialized to t
LastCycleKsolv2Kmiss [run_buster] refinement during each BIG cycle	"no"	When using miss contribution be i
LastCycleRefineBmiss [run_buster] refinement during each BIG cycle	"no"	Should the B-fac atoms channel is
LastCycleRefineKmiss [run_buster] refinement during each BIG cycle	"no"	Should the scale atoms channel is
LigandDescribingPdbMethod [refine] overall refinement (BIG cycles)	"new"	Which method to recommended ov
MacroReport [refine]	"no"	Should we report useful for debugg
MaxAllowedCNDistanceInSeq [pdb2seq] generation of TNT sequence file	"2.0"	If C-N distances PEPTIDE) are ab
MaxAllowedOPDistanceInSeq [pdb2seq] generation of TNT sequence file	"2.5"	If O3*-P distanc dSUGPHOS/SU
MinAllowedCNDistanceInSeq [pdb2seq] generation of TNT sequence file	"1.65"	If C-N distances BREAK) are bel
MinAllowedOPDistanceInSeq [pdb2seq] generation of TNT sequence file	"2.0"	If O3*-P distanc a BREAK) are b
MissingAtomsBfac [run_buster] running BUSTER	"150.0" or Wilson-B times factor	Which B-factor s channel is being MissingAtomsB instead.
MissingAtomsBfacFactor [run_buster] running BUSTER	"1.25"	Factor to apply t have a higher B-
MissingAtomsBfacSigma [run_buster] running BUSTER	"15.0" or Wilson-B times factor	Which sigma sho atom channel is MissingAtomsB instead.
MissingAtomsBfacSigmaFactor [run_buster] running BUSTER	"0.125"	Factor to apply t
Mtv2PngGeom [mtv2png]	"1200x900"	
MtzChk_MaxNumToPrint [mtzchk] Initial check of input MTZ file	"10"	How many reflec
Mtztools_FreeRflag_FrcMax [mtztools] Handling MTZ file	"0.10"	Maximum fracti needs to be creat
Mtztools_FreeRflag_FrcMin [mtztools] Handling MTZ file	"0.05"	Minimum fracti to be created).

Mtztools_FreeRflag_NumAim [mtztools] Handling MTZ file	"1000"	Ideally, this num required).
MxlcycCutBuster [refine] overall refinement (BIG cycles)	"0.75"	When reducing t TNT minimizer - reduce (or increa
NeverGenerateDictionary [run_buster] refinement within each BIG cycle	"yes"	Disable the (stro on the current co
NmissMinimumFrac [run_buster] refinement within each BIG cycle	"0.05"	Minimum fractio missing atom cha
NmissMinimumFracWarnOnly [run_buster] refinement within each BIG cycle	"no"	Should we only atom channel dro
NoOverallBanisoRefinement [run_buster] refinement during each BIG cycle	"no"	If set to yes (and BUSTER
NonStandardInitialRemarkSectionCard [refine] Generation of final PDB file	"REMARK"	string to be used top of refine.pdb
NonStandardInitialRemarkSectionDelimiter [refine] Generation of final PDB file	"yes"	should we create standard PDB he
PassThroughArgs [refine] overall refinement (BIG cycles)	"-autoncs -autoncs_weight:n -autoncs_noprune -target:f -target_weight:n -dlim:n -glim:n -sim_swap_equiv -sim_swap_equiv_plus -screen:i -screen_sigma:n -verbose:i -verbose_set -type:s -max:i -gelly_fn -tnt_fn -tnt_range_match -torsharm -updatedist:n -special_tnt -special_dist:n -keeppdb:i"	List of command BUSTER
PassThroughArgsUser [refine] overall refinement (BIG cycles)	""	List of additional directly to the ge
Pdb2Dpi_NparPerAtom [pdb2dpi]	"4"	default number o used in DPI calcu
Pdb2OccLim [pdb2occ]	"1.0"	consider atoms w
Pdb2OccRes [pdb2occ]	""	space-delimited l occupancy-refine single compound
Pdb2Tls_AdjustBorderResidueNumbers [pdb2tls] creating TLS description from PDB file	"yes"	If there is a resid check and adjust
Pdb2Tls_AutomaticDefinition_EachMacroMolChain_usecurly [pdb2tls] creating TLS description from PDB file	"yes"	When creating an
Pdb2Tls_AutomaticDefinition_OnePerChain_SkipBreaks [pdb2tls] creating TLS description from PDB file	"no"	When creating an sequence?
Pdb2Tls_AutomaticDefinition_OnePerChain_minatm [pdb2tls] creating TLS description from PDB file	"100"	When creating an chain need to hav
Pdb2Tls_AutomaticDefinition_type [pdb2tls] creating TLS description from PDB file	"EachMacroMolChain" or "OnePerChain"	When creating an EachMacroMolC
Pdb2Tls_FormatType [pdb2tls]	"BUSTER"	
Pdb2Tls_HeaderDefinition_useval [pdb2tls] creating TLS description from PDB file	"yes"	If a REMARK 3 values reported i
PdbChk_AdditionalChecksToDo [pdbchk] initial checking of PDB file	""	Additional tests t provided progra
PdbChk_AtomNameUnsupportedCharacters1 [pdbchk] initial checking of PDB file	"#!"	Which characters
PdbChk_AtomNameUnsupportedCharacters2 [pdbchk] initial checking of PDB file	"AB"	Substitute charac characters (see al
PdbChk_AtomNamesAgainstStandardRestraintsExclude [pdbchk] initial checking of PDB file	"^OXT$\|^H$|^H[HZGABDE0-9]|^[0-9HA]H|^HT[0-9]$"	When testing ato atoms.
PdbChk_AtomNamesAgainstStandardRestraintsWarning [pdbchk] initial checking of PDB file	"^O3P$"	When testing ato warning for thos
PdbChk_ChecksNotToDo [pdbchk] initial checking of PDB file	""	Selected tests no
PdbChk_FixAtomNamesOfResidues [pdbchk] initial checking of PDB file	"FAD|NAP|NAI|COA|NDP|NAD|\ AP5|CAA|NAH|ACO"	Adjust the atom columns 13-16).
PdbChk_InitialChecksToDo [pdbchk]	""	space-separated l user-provided pr
PdbChk_MaxNumToPrint [pdbchk] initial checking of PDB file	"10"	Maximum numb
PdbChk_MaximumCellAngle [pdbchk] initial checking of PDB file	"155.000"	Maximum allow

PdbChk_MaximumCellEdge [pdbchk] initial checking of PDB file	"9999.999"	Maximum allow
PdbChk_MaximumCellVolume [pdbchk] initial checking of PDB file	"10000000000.00"	Maximum allow
PdbChk_MinimumCellAngle [pdbchk] initial checking of PDB file	"25.000"	Minimum allowe
PdbChk_MinimumCellEdge [pdbchk] initial checking of PDB file	"1.010"	Minimum allowe
PdbChk_MinimumCellVolume [pdbchk] initial checking of PDB file	"10.00"	Minimum allowe
PdbChk_PossibleChainIds [pdbchk] initial checking of PDB file	"ABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890"	String containing
PdbChk_RecordFormats [pdbchk] initial checking of PDB file	"CRYST1 ATOM HETATM"	List of PDB reco
PdbChk_TooShortRecordsList [pdbchk] initial checking of PDB file	"HEADER;66: REVDAT;66: DBREF ;68: HELIX ;76: SSBOND;72: LINK ;72: HYDBND;72: SLTBRG;72: CISPEP;59: SITE ;61: ORIGX1;55: ORIGX2;55: ORIGX3;55: SCALE1;55: SCALE2;55: SCALE3;55: MTRIX1;60: MTRIX2;60: MTRIX3;60: MODEL ;14: ATOM ;80: SIGATM;80: ANISOU;80: SIGUIJ;80: HETATM;80: TER ;27: ENDMDL;6: CONECT;61: END ;6"	List of records (a minimum length.
PdbChk_WrongReferenceToCoordinateRecordError [pdbchk] initial checking of PDB file	"SSBOND|LINK "	List of (bar-delin record will trigge
PdbStandardResidues [pdbchk] initial checking of PDB file	"ALA|ARG|ASN|ASP|ASX|CYS|GLN|GLU|GLX|GLY|\ HIS|ILE|LEU|LYS|MET|PHE|PRO|SER|THR|TRP|\ TYR|UNK|VAL| A| +A| C| +C| G| +G| I|\ +I| T| +T| U| +U"	List of standard r
Pdbtools_OccMax [pdbtools]	"1.00"	maximum allowe
Pdbtools_OccMin [pdbtools]	"0.00"	minimum allowe
RemoveOldNonStandardInitialRemarkSection [refine] creation of final PDB file	"yes"	Should we remov records from the
RemoveScaleCardsFromPdb [various] conversion from PDB to ATOMC format and back	"yes"	To make sure tha done with the co the PDB file befo
RenumberIfBelow [run_buster] refinement during each BIG cycle	"-999"	Residues in the i
ReportHostname [refine]	"yes"	Should we report standard output?
ReuseFormfactorFile [refine]	"yes"	From BIG cycle generated before
ReuseGeometryFile [refine] overall refinement (BIG cycles)	"yes"	From BIG cycle generated before
ReuseSequenceFile [refine] overall refinement (BIG cycles)	"yes"	From BIG cycle generated before
RhoMacroMol [run_buster] refinement during each BIG cycle	0.42	Mean electron de 0.60 for a nucleic
RhoSolvent [run_buster] refinement during each BIG cycle	0.33	Mean electron de
RmAnisou [refine] overall refinement (BIG cycles)	"yes"	Should we remov
RmLink [refine] overall refinement (BIG cycles)	"no"	Should we remov

RmModres [refine] overall refinement (BIG cycles)	"no"	Should we remov
RunBusterPrintUsefulInfo [run_buster] refinement during each BIG cycle	"no"	Should there be s
RunCor2Pdb [run_buster] refinement during each BIG cycle	"yes"	Should we run "c refinement into a
RunGellyNcsScreen [run_buster] refinement during each BIG cycle	"no"	Should we create current set of NC cycle??
RunGellySanityCheck [run_buster] refinement during each BIG cycle	"yes"	Should we run an refinement in eac
RunGellyScreen [run_buster] refinement during each BIG cycle	"yes"	Should we create the actual refiner
RunHarvest [run_buster] refinement during each BIG cycle	"all"	Should we run th header)? One of
RunPdb2Cor [run_buster] refinement during each BIG cycle	"yes"	Should we run th format?
RunSfcheck [run_buster] refinement during each BIG cycle	"yes"	Should we run S
RunWilsonTwice [run_buster] refinement during each BIG cycle	"no"	Do we need to ru
ScreenNumBuster [run_buster] refinement during each BIG cycle	"100"	Maximum numb BUSTER for the
ScreenSigmaBuster [run_buster] refinement during each BIG cycle	"3.0"	All outliers abov screen_*.txt files
ScreenSigmaInitial [run_buster] refinement during each BIG cycle	"5.0"	All outliers abov initial geometry.
SequenceFileGeneration [ab_pdb2tnt] generation of TNT sequence file	"MakeLINK"	Which tool to us "MakeLINK").
SsbondSgDistanceMax [pdb2seq] generation of TNT sequence file	"2.5"	When automatica use this as longes
SsbondSgDistanceMin [pdb2seq] generation of TNT sequence file	"1.5"	When automatica use this as shorte
StandardDictionaries [various] standard dictionaries/databases	"protgeo_eh99 exoticaa nuclgeo bcorrel contact idealdist_contact"	list of files (in $E dictionaries
StandardDictionariesAll [various] standard dictionaries/databases	"protgeo_eh99 exoticaa nuclgeo sugar cofactor_geo othergeo bcorrel contact idealdist_contact assume connect"	list of files (in $E dictionaries
StartFromPreviousWeightInPdb [refine] overall refinement (BIG cycles)	"yes"	If the X-ray weig file: should it be
StopAfterBuster [refine] overall refinement (BIG cycles)	"no"	Should we stop r
StopAfterCmdServer [run_buster] refinement during each BIG cycle	"no"	Should we stop r
StopAfterGellySanityCheck [run_buster] refinement during each BIG cycle	"no"	Should we stop r
StopAfterRunBuster [run_buster] refinement during each BIG cycle	"no"	Should we stop r
StopBeforeCmdServer [run_buster] refinement during each BIG cycle	"no"	Should we stop r
StopOnCreatingEmptyLinkDictionary [pdb2seq] generation of TNT sequence file	"no"	Should we stop w connectivity is de dictionary?
StopOnDifferentSpacegroup [refine] overall refinement (BIG cycles)	"yes"	Should we stop w
StopOnInitialXrayWeightFromPdbOutsideLimits [refine] overall refinement (BIG cycles)	"yes"	Should we stop w limits?
StopOnInitialXrayWeightOutsideLimits [refine] overall refinement (BIG cycles)	"yes"	Should we stop w
StopOnGellySanityCheckError [run_buster] refinement during each BIG cycle	"yes"	Should we stop w

StopOnMissingContactDistance [run_buster] refinement during each BIG cycle	"no"	Should we stop v
TLSAutoFuseMultiCurly [refine]	"yes"	detect and autom single BUSTER_
TLSfixcycALL [refine] overall refinement (BIG cycles)	"2"	from which BIG
TLSfixcycRB [refine] overall refinement (BIG cycles)	"1"	from which BIG
TlsUseFromPdbRemark3 [refine] overall refinement (BIG cycles)	"yes"	If TLS refinemer we try and extrac of the input PDB
TntBfacMax [run_buster] refinement during each BIG cycle	"300"	Upper limit for a
TntBfacMin [run_buster] refinement during each BIG cycle	"3"	Lower limit for a
TntDictionary_assume [various] standard dictionary	"$BDG_home/tnt/data/assume.dat"	file that enables
TntDictionary_bcorrel [various] standard dictionary	"$BDG_home/tnt/data/bcorrel.dat"	standard dictiona
TntDictionary_cofactor [various] standard dictionary	"$BDG_home/tnt/data/cofactor_geo.dat"	standard dictiona
TntDictionary_connect [various] standard dictionary	"$BDG_home/tnt/data/connect.dat"	file describing co and C-terminal n
TntDictionary_contact [various] standard dictionary	"$BDG_home/tnt/data/contact.dat"	database with mi
TntDictionary_csdx [various] standard dictionary	"$BDG_home/tnt/data/csdx_protgeo.dat"	old standard dict
TntDictionary_exoticaa [various] standard dictionary	"$BDG_home/tnt/data/exoticaa.dat"	standard dictiona
TntDictionary_formfactor [various] standard dictionary	"$BDG_home/tnt/data/formfactor.dat"	database with for
TntDictionary_idealdist [various] standard dictionary	"$BDG_home/tnt/data/idealdist_contact.dat"	standard dictiona
TntDictionary_nuclgeo [various] standard dictionary	"$BDG_home/tnt/data/nuclgeo.dat"	standard dictiona
TntDictionary_othergeo [various] standard dictionary	"$BDG_home/tnt/data/othergeo.dat"	standard dictiona
TntDictionary_pdbfixup [various] standard dictionary	"$BDG_home/tnt/data/pdb_fixup.dat"	database to trans
TntDictionary_protgeo [various] standard dictionary	"$BDG_home/tnt/data/protgeo_eh99.dat"	standard dictiona
TntDictionary_sugar [various] standard dictionary	"$BDG_home/tnt/data/sugar.dat"	standard dictiona
TntWeightGeomRes [refine] overall refinement (BIG cycles)	"no"	Should we get th (default is to adj
TransferExoticAAFromSeqToGelly [refine] overall refinement (BIG cycles)	"no"	Should we add a sequence file to t
UpdateWaters [refine] overall refinement (BIG cycles)	"no"	Should we updat
UpdateWatersEpdbOnly [refine] overall refinement (BIG cycles)	"no"	Should the updat solvent mask (an
UseBrefGroupFrom [refine] overall refinement (BIG cycles)	"999.0"	Resolution limit
UseBrefMcScFrom [refine] overall refinement (BIG cycles)	"999.0"	Resolution limit chain and one fo
UseBrefNoneFrom [refine] overall refinement (BIG cycles)	"3.5"	Resolution limit overall B-factor)
UseCrdScaleAfterCycle [refine] overall refinement (BIG cycles)	"yes"	Should we use th

UseCrdScaleAfterRigid [refine] overall refinement (BIG cycles)	"no"	Should we use th
UseDictionaryOrder [ab_pdb2tnt] generation of TNT restraints	"msd ccp4 maketnt"	What is the prefe residues? "msd" $BDG_home/tnt "maketnt" will cr
UseEpdbLastCycle [refine] overall refinement (BIG cycles)	"yes"	Should the bulk- regions (given e.
UseGapAsBreakInSeq [pdb2seq] generation of TNT sequence file	"no"	Instead of the de
UseGellyPdb [run_buster] refinement during each BIG cycle	"yes"	Should we use th
UseHighResInRigid [refine] overall refinement (BIG cycles)	"yes"	Should we use al body refinement
UseLinkFromGellyPdb [run_buster] refinement during each BIG cycle	"yes"	Should we use L coming from Gel
UseLlgradAsFoFc [refine] overall refinement (BIG cycles)	"no"	Instead of using
UseLlzThroughout [refine] overall refinement (BIG cycles)	"no"	Should we keep calculate log-like starting values at
UseLowResInRigid [refine] overall refinement (BIG cycles)	"yes"	Should we use al 6 Å)?
UseLpdbLastCycle [refine] overall refinement (BIG cycles)	"yes"	Should the bulk- -Lpdb flag into a
UseMakeTntAuto [dic2tnt] converting restraint dictionaries on the fly	"no"	Should all user-s
UseMapAsNup [refine] overall refinement (BIG cycles)	"no"	Should we use a
UseMapAsSlv [refine] overall refinement (BIG cycles)	"no"	Should we use a
UseMaxEntLastCycle [refine] overall refinement (BIG cycles)	"no"	Should we run a
UseMaxEntThroughout [refine] overall refinement (BIG cycles)	"no"	Should we run a
UseMtzchk [refine] overall refinement (BIG cycles)	"yes"	Should we use th
UseMxlcycLastCycle [refine] overall refinement (BIG cycles)	"yes"	Should we run re
UseNmissLastCycle [refine] overall refinement (BIG cycles)	"no"	Should we use th
UseNmissThroughout [refine] overall refinement (BIG cycles)	"no"	Should we use th
UsePdbchk [refine] overall refinement (BIG cycles)	"yes"	Should we use th
UsePdbcmb [run_buster] refinement during each BIG cycle	"yes"	Should we use th information (fro
UseRefmacdict2tnt [dic2tnt] converting restraint dictionaries on the fly	"yes"	Should we use th dictionaries (*.ci
UseSortwater [maptools] various steps involving maps	"no"	Should we use th closer to PDB fil
WaterChainId [various] water updating	"W"	Use this chain id ATOM/HETAT
WaterFindSigma [maptools] various steps involving maps	"3.2"	Cut-off level [rm
WaterFindSigmaLlg [maptools] various steps involving maps	"6.0"	Cut-off level [rm
WaterMinDistance [maptools] various steps involving maps	"2.5"	Minimum distan

WaterNamingAtom [various] water updating	" O "	atom name of wa records of PDB f
WaterNamingResidue [various] water updating	"HOH"	residue name of records of PDB f
WaterPickingOptimise [maptools] various steps involving maps	"no"	When picking pe optimized?
WaterPickingHydrogenPartner [maptools] various steps involving maps	"no"	When picking pe distance of a hyd
WaterPickingHydrogenPartnerAll [maptools] various steps involving maps	"no"	Do we accept all protein residues)
WaterRemoveDeleted [maptools] various steps involving maps	"0.5"	Any waters close added (again).
WaterRemoveDistFac [maptools] various steps involving maps	"0.16666"	Used in conjunct existing water at it will be remove
WaterRemoveHpartner [maptools] various steps involving maps	"3.5"	If a hydrogen bo
WaterRemoveMerge [maptools] various steps involving maps	"2.2"	Water molecules atom (when usin
WaterRemoveSigma [maptools] various steps involving maps	"0.8"	Water molecules removed.
WaterResidueNames [various] user-supplied PDB file	"HOH WAT"	list of residue na file
WaterUpdateFftResMin [maptools] various steps involving maps	"1.5"	Minimum high r
WaterUpdateProgram [refine] overall refinement (BIG cycles)	"PKMAPS"	Program to use f "PKMAPS", "AF program/script). water_updater.sh
XrayWeight_max [refine] overall refinement (BIG cycles)	"50.0"	Maximum value
XrayWeight_min [refine] overall refinement (BIG cycles)	"1.0"	Minimum value
XrayWeight_start [refine] overall refinement (BIG cycles)	"4.0"	Starting value of and parameter K **Note:** the unit of instead of an inte (typically betwe
autoBUSTER_MacroDirs [general]	""	colon-separated l
autoBUSTER_fss [refine] overall refinement (BIG cycles)	""	List of (space-se "FP,SIGFP Fnat, column type "Q"
autoBUSTER_hls [refine] overall refinement (BIG cycles)	"no"	List of (space-se refined against e automatic usage
blkblr [run_buster] refinement during each BIG cycle	"50"	Bulk-solvent ma
blkrad [run_buster] refinement during each BIG cycle	"215"	Bulk-solvent ma
do_analyse [refine] presentation of results	"yes"	Should we run fi
do_maps [refine] presentation of results	"yes"	Should we create
frgrad [run_buster] refinement during each BIG cycle	"215"	Fragment mask r
mskblr [run_buster] refinement during each BIG cycle	"50"	Non-uniform pri
mskisl [run_buster] refinement during each BIG cycle	"0"	Should we try an
mskrad [run_buster] refinement during each BIG cycle	"400"	Non-uniform pri
mxlcyc_start [refine] overall refinement (BIG cycles)	"100"	Starting value fo

nmiss [refine] overall refinement (BIG cycles)	"0"	Number of atom
nthreads [run_buster] refinement during each BIG cycle	fraction of available	How many threa the "-nthreads" c threads for detail
refmacdict2tnt_args [dic2tnt] converting restraint dictionaries on the fly	""	extra arguments
refocc [run_buster] refinement during each BIG cycle	"off"	Should we refine
refscl_rfr [run_buster] refinement during each BIG cycle	"0"	Should we refine
refscl_rfs [run_buster] refinement during each BIG cycle	"1"	Should we refine
refscl_rif [run_buster] refinement during each BIG cycle	"1"	Should we refine
refscl_rir [run_buster] refinement during each BIG cycle	"0"	Should we refine
refscl_ris [run_buster] refinement during each BIG cycle	"1"	Should we refine
refscl_rkim [run_buster] refinement during each BIG cycle	"0"	Should we refine
refscl_rkis [run_buster] refinement during each BIG cycle	"0"	Should we refine
rmsBOND_target [refine] overall refinement (BIG cycles)	"0.010"	Target value rms
solc [run_buster] refinement during each BIG cycle	""	solvent content [PDB file
wavelength [run_buster] refinement during each BIG cycle	"1.54180"	Wavelength of re

Last modification: 08.05.2013