Supporting information

Description of the *processPIXEL* program

The supplied executable file should be copied onto a *Windows* PC, probably into the same directory as the *CLP* executables (usually in the *batch* subdirectory), with the path properly configured. This allows *processPIXEL* to be run from the Command Prompt immediately after completion of the *runpixelc* module.

S1. Command line arguments

`processPIXEL`	Run the program in the current working directory with default options
`processPIXEL help`	Display a list of command line arguments to the console
`-d %1`	Run the program in the working directory specified by the argument *%1*. Either a full path or sub-path can be specified. Default is "."
`-h %1`	Turn on or off the energy-vector ("hedgehog") output. Allowed values for argument *%1*: 0 (off) or 1 (on: default).
`-s %1`	Sort the interactions in the output file **_mlc.txt*. Allowed values for argument *%1*: 0 (not sort), 1 (distance), 2 (coulomb), 3 (polarisation), 4 (dispersion), 5 (repulsion), 6 (total: default).
`-t %1`	Set the minimum length for energy vectors output in the PDB file. Argument *%1* must be parsable as a double. If no value is specified, the program outputs all vectors in the *.mlc* file.
`-v %1`	Turn on or off verbose output. Allowed values for argument *%1*: 0 (off: default) or 1 (on).

S2. Program description

The program processes sequentially all files *%1*.oeh in the specified working directory. Files with name *%1*ort.oeh are read, but they are skipped if they are found to contain an orthonormal metric (1, 1, 1, 90, 90, 90) and there are no symmetry operators listed. The program should parse successfully *.oeh* files from the current version of the *CLP* package, but it will fail for *.oeh* files from older versions of the *OPIX/PIXEL* package on account of some deprecated atom type codes, and other changes to the file format. Back-compatibility was not attempted because it would increase the chances of problems in the file parsers, and on the premise that newer versions of the *CLP* package should offer improvements over the older versions.

S2.1. Reading the *.oeh* file

It is assumed that the *.oeh* file is written out by the *runretcor* module and is not subsequently modified. Therefore, only limited formatting checks are applied.

- The metric information is assumed to be on line 3. The first 6 tokens (strings separated by white space) on this line must be parsable as doubles (a,b,c,α,β,γ). The cell volume is calculated. In verbose mode, a confirmation line is issued to the console immediately after the metric information is read successfully.
- The first token on line 5 defines the number of atoms to be read (nAt). It must be parsable as int.
- The next nAt lines are read. Each line must have 7 tokens, and the first 6 tokens must be parsable as int, double, double, double, int, int. The last token on each line (atomic charge) is not read. If any of the nAt lines cannot be parsed as an atom, the program terminates. Each atom has a residue number (1 or 2). If any atom is found to have residue no. 2, the structure is flagged as zPrime = 2. After reading nAt lines, the next line is checked: if this can be successfully parsed as an atom, the program notes the apparent inconsistency with the value of nAt and terminates. In verbose mode, a confirmation line is issued to the console immediately after all atoms are read successfully, and the identified value of zPrime is noted.
- The first token on the next line (two after the end of the atom list) defines the number of symmetry operators to be read (nSymm). It must be parsable as int.
- The next nSymm pairs of lines are read. The lines must alternately have 9 and 3 tokens, all parsable as doubles. If any of the nSymm lines cannot be parsed as an operator, the program terminates. After reading nSymm pairs of lines, the next line is checked: if this has 9 tokens, it is assumed that this must be another symmetry operator. The program notes the apparent inconsistency with the value of nSymm and terminates. In verbose mode, a confirmation line is issued to the console immediately after the symmetry operators have been read successfully.

S2.2. Skipping of files **.ort.oeh*

Files **ort.oeh* are read in full as outlined above. After reading, files with names **ort.oeh* are checked for an orthonormal metric and an empty list of symmetry operators. If all three conditions are met (name & metric & symmetry), the file is not processed further.

S2.3. Output of the file *%1_OEH.cif*

Centres of mass are calculated for the input molecule(s) (assembled by fragment number). The stored relative atomic masses are taken from the *CLP* manual, and look-up is based on the *CLP* species number. Invalid *CLP* species numbers will lead to program failure at this stage. In verbose mode, the program outputs the centre(s) of mass to the console immediately after calculation.

The output file *%1_OEH.cif* is then written. It contains the metric information read from the *.oeh* file (plus the calculated cell volume), the symmetry operators read from the *.oeh* file (in the sequence that they are read) and the atoms from the *.oeh* file (labelled by element type and sequence no. in the *.oeh* file). The centres of mass are added as element type "X", which is conveniently represented in *Mercury* as a non-connected point. The centre-centre distances are consistent with the distances in the *PIXEL* output.

S2.4. Reading the *.mlc* file

For each file *%1.oeh*, the program reads the corresponding file *%1.mlc*. If this file is not present, the program moves on to process the next *.oeh* file. It is assumed that the *.mlc* file is exactly as written by the *runpixelc* module, so only limited formatting checks are applied. Interactions can be removed from the lists by deleting entire lines. The list of symmetry operators should not be edited to preserve correspondence with the operator indices in the atom list.

- The second token on line 1 defines the number of operators to be read (nOp).
- The next nOp lines are read. Each line must have 13 tokens, and be parsable as int followed by 12 doubles. If any of the nOp lines cannot be parsed as an operator, the program terminates. In verbose mode, a confirmation line is issued to the console after all operators have been read.
- The file read continues line-by-line, looking for "a…a", "a…b", "b…a" or "b…b" as the first token on a line. When such a line is found, the following lines are processed as interactions until a line is reached where the number of tokens does not equal 10. For each interaction line, only the first 8 tokens are read (AA and QQ values are ignored). These must be parsable as int, int, 6 × double. Any deviation from this format will cause the program to terminate. Each interaction is stored with a note of the molecular fragment number (1 or 2, derived from the title lines "a…a", "a…b", *etc.*), the index number of the applied operators in the list, the centre-centre distance and the 5 energy terms. When a line is encountered with no. of tokens not equal to 10, the program stops reading interactions and continues to check for other headings ("a…b", *etc.*) to be processed until the end of the file is reached.
- In verbose mode, the numbers of read A…A, A…B, B…A and B…B interactions are written to console immediately after **all** interactions have been read.

Note: for structures with two independent molecules, the A…B list contains the interaction without any applied symmetry operators. This is not duplicated in the B…A list. Internally, this interaction is also added as a B…A interaction so that it also appears in the vectors emanating from B. The numbers of interactions quoted in verbose mode do not include this extra interaction – they are quoted exactly as read from the *.mlc* file for checking purposes.

S2.5. Output of the file *%1_MLC.txt*

An output file is produced in .txt format, with table columns separated by ";". This file can be easily imported into *MS Excel* or converted into a table in *MS Word* (select all text, Insert – Table – Convert Text To Table). It contains all interactions in *the .mlc* file, with applied symmetry operators included as conventional representations, rather than the indexed matrix representation output by *PIXEL*. It also includes the labels that are applied to the energy vectors in the output PDB files, so that each interaction line can be matched to the models. The interactions in the table can be sorted according to any of the columns (distance, coulomb, polarisation, dispersion, repulsion, total) using the command-line switch –s. The default is to sort by total *PIXEL* energy. For interactions with equal energy, a secondary sort on distance is applied (which is often the same since these cases are usually symmetry equivalent!).

S2.6. Generation of the energy-vector models

Unless deactivated by the switch –h 0, the program next calculates the energy vector models. This involves the following steps:

- Scan through all stored interactions to identify the maximum absolute value for each of the 5 stored energies.
- For each interaction, calculate the centre-centre vector and scale it to the requried length, based on whichever energy type is being considered (as described in the main text of the paper).
- Calculate coordinates for the end point of the energy vector in the crystallographic frame.
- Convert all coordinates to a Cartesian frame, with storage of the transformation matrix. The matrix required for the SCALE lines of the PDB file is the transpose of the orthonormalisation matrix for the metric tensor (putting x_{ort} parallel to the crystal *a* axis, y_{ort} in the crystal *ab* plane and z_{ort} perpendicular to these two). Crystal coordinates are generated by: SCALE matrix (3×3) * Cartesian coordinates (as a 3×1 matrix).
- Write the PDB file. Add only the energy vectors with length greater than the specified threshold. Specify connectivity for all included interactions using individual CONECT lines. Unless

overridden by the –t switch, the threshold is set to 0, meaning that all non-negative energy vectors are output.

Note: after calculating the scaled energy vector lengths, all vectors are further scaled by a factor of 0.999. This overcomes an issue in *Mercury*, which arises when the structure is packed: overlapping atoms at the ends of the longest energy vectors can be calculated to be connected to themselves, which throws an exception and prevents packing. Rescaling by 0.999 moves the ends of the vectors sufficiently to overcome this problem (in all tested cases). It is not perceptible to the eye.

S2.7. Visualisation of the results

The output files are targeted specifically at the *Mercury* program, and it is advisable to place the *processPIXEL* executable in the same directory as the *CLP* executables. After completion of the *runpixelc* module, type *processPIXEL*. For each file *%1.oeh* in the directory (often there is only one), the program creates a subdirectory with the name *%1*. In that directory, open the file *%1_OEH.cif* using *Mercury*. Tick on "Multiple Structures" in *Mercury*, then open *%1_Total.pdb* (or one of the other files). Turn on "Packing" in *Mercury*, and view along one of the cell axes to line up the two models. The energy-vector model will appear as magenta lines. It is best viewed in "Wireframe" or "Capped Stick" modes. It is usually advisable to pack 2×2×2 to be sure that the longest energy vectors are properly joined. The labels in the energy-vector diagrams are "A0" (and "B0" when zPrime = 2) for the molecular centroids, then "Axx" for fragment A after application of symmetry operator xx, *etc.* The labels used in the PDB files are also indicated in the output file *%1_MLC.txt*.

The five output files *%1_Coul.pdb*, *%1_Disp.pdb*, *%1_Pol.pdb*, *%1_Rep.pdb*, *%1_Total.pdb* refer to the various components scaled against the total *PIXEL* energy. The four output files *%1_Individual_Coul.pdb*, *%1_Individual_Disp.pdb*, *%1_Individual_Pol.pdb* and *%1_Individual_Rep.pdb* refer to the various components scaled against the maximum value of that specific energy term. For the representations of the repulsion terms, longer vectors represent more *destabilising* interactions.