Journal of
Applied Crystallography
ISSN 0021-8898

Received 18 March 2004
Accepted 16 April 2004

CIF applications. XVI. *CIF2CRY*: for CIF input into the *CRYSTALS* program

Richard Ian Cooper,[a]* Bruce M. Foxman[b] and Lin Yang[b]

[a]University of Oxford, Chemical Crystallography Laboratory, Chemistry Research Building, Mansfield Road, Oxford OX1 3TA, UK, and [b]Department of Chemistry, Brandeis University, Waltham, MA 02454, USA. Correspondence e-mail: richard.cooper@chemical-crystallography.oxford.ac.uk

CIF2CRY is a program for converting structural data and reflection data contained in a CIF into instructions to be read by the *CRYSTALS* program [Betteridge, Carruthers, Cooper, Prout & Watkin (2003). *J. Appl. Cryst.* **36**, 1487]. The program can be used, for example, to read in CIFs that are archived by crystallographic journals, allowing users to study and re-refine published structures. Multiple data blocks in one CIF can be converted into a continuous file of *CRYSTALS* instructions, including an instruction for running user-specified calculations after each structure has been input. The program uses the CIFtbx Fortran library.

1. Introduction

The CIF format (Hall *et al.*, 1991) has become a *de facto* standard for storing and sharing crystallographic results. While many authors have produced programs that can write a CIF, it is a much more challenging task to read the format, requiring a detailed knowledge of CIF structure and syntax, and this fact has limited the use of CIF as a format for sharing data between programs. A few programs have implemented the required CIF reading routines, notably *PLATON* (Spek, 1998), which will read a CIF and can output *SHELX* format files (Sheldrick, 1998) for subsequent refinement. Fortunately for authors of other programs, CIF reading libraries, such as the CIFtbx Fortran library (Hall, 1993), provide some help in transferring data between program memory and CIF data items.

CIF2CRY is a format conversion program, written in Fortran, which exploits the access to CIF data items provided by CIFtbx. The program aims to extract all relevant information present in a CIF and write a file of instructions to input that data into *CRYSTALS* (Betteridge *et al.*, 2003). The program runs with a minimum of user interaction and, as shown below, can be hidden from the user behind a point-and-click dialog interface in *CRYSTALS*.

2. Usage

CIF2CRY can be run interactively in a terminal window or with options specified as command-line arguments, or a combination of the two. The program can be accessed from within *CRYSTALS via* a simple dialog box.

2.1. Command-line arguments

Specifying incorrect or illegal combinations of command-line arguments causes *CIF2CRY* to display usage information as follows:

cif2cry [−f | − a] [[−n extension] | [−o outputfile]] [inputfile].

−f causes only the first data block to be extracted from the CIF.
−a causes all data blocks to be extracted from the CIF.

The options −f and −a are mutually exclusive. If neither option is specified, *CIF2CRY* will ask interactively whether each data block should be output.

The −n extension option causes output to be written to a file named by prefixing the string 'extension' with the data block name from the CIF.

−o outputfile causes output to be written to a file named outputfile.

The options −n and −o are mutually exclusive. Note that if multiple data blocks are to be extracted from the CIF, then −n implies that each block will be written to a separate file, while −o will cause all output to be written to a single file. If neither option is specified, *CIF2CRY* will ask interactively for an output filename for each data block.

inputfile is the name of the CIF from which data will be extracted. If no CIF filename is specified on the command line, then *CIF2CRY* will interactively ask for one when it is executed.

2.2. *CRYSTALS* dialog interface

A simple user interface providing access to some common modes of use of *CIF2CRY* is accessible from the *CRYSTALS* menus or by typing '#script cif2cry' at the *CRYSTALS* command line (Fig. 1).

There are three operations available. Firstly, *Extract first or only data block*, which is equivalent to the command-line options −f −o; secondly, *Extract all data blocks to a single file*, which is equivalent to the command-line options −a −o; and finally, *Extract all data blocks to multiple files*, which is equivalent to the command-line options −a −n. For the first two operations, an output filename must be specified, and this file can be automatically input into *CRYSTALS* if desired. The third operation produces multiple files and requires just a filename extension. There is no option to process multiple instruction files automatically.

3. Algorithm

The program extracts as much data about the structure as it can from the CIF. Firstly, the data block name itself is stored as a *CRYSTALS*

'Title' instruction, the cell and space-group data are retrieved, and appropriate *CRYSTALS* instructions are written to the output file. Next the _cell_formula_unit_Z and _chemical_formula_sum data are used to construct a unit-cell composition instruction in the output file.

The atom-site data are processed next. *CRYSTALS* uses element type and a serial number to identify atoms uniquely, while the CIF specification uses an _atom_type_site_symbol to specify the element and a character label, _atom_site_label, to identify the atoms. Converting from the *CRYSTALS* format to CIF is not a problem, but the reverse is more complicated since the labels are not always composed simply of concatenated element symbol and number but may be of the form C1*a*, H3* or H3′, or simply Fe. Since there is no simple scheme that can be devised to convert such labels to serial numbers in a reversible way, we simply replace non-numeric symbols with integer numbers or, in the case of lone element symbols, insert a 0. The original label is stored safely on the end of a *CRYSTALS* atom instruction for future retrieval. Labels such as H3* and H3′ that occur in the same structure will both end up with serial number 30, so an instruction is included after the list of atoms to make *CRYSTALS* find and rename any duplicate atom identifiers.

Information about disorder can be contained in a CIF atom list using the _atom_site_disorder_assembly data name to identify clusters of atoms that are positionally disordered within a structure and, within each assembly, using _atom_site_disorder_group to identify uniquely those atoms that are simultaneously occupied. This information can be input into *CRYSTALS* and, as well as being useful in defining the constraint matrix for refinement of occupancies, it is of most use in restricting the output of distance and angle calculations to those that are chemically relevant.

Following the atom list, any experimental details that are known to *CRYSTALS* are found from the CIF and written to the output file.

Finally, *CIF2CRY* searches for a list of *hkl* indices containing either structure-factor amplitudes or amplitudes squared and associated sigma values. If present, the *hkl* data are stored in a separate file and instructions are inserted into the *CRYSTALS* input to read data from this file.

Each set of instructions for *CRYSTALS* finishes with a call to a *CRYSTALS* script, '#script cifproc', which will automatically execute the contents of a file 'cifproc.dat' if present in the working directory.

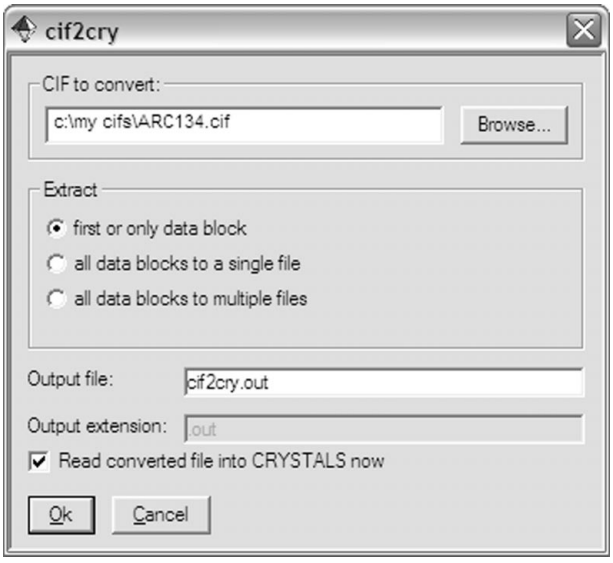

Figure 1
The *CIF2CRY* user interface provided within *CRYSTALS*.

This file may contain any *CRYSTALS* instructions, *e.g.* distance and angle calculations, refinement instructions, residual analysis, output of structural diagrams or even output of a new CIF. This facility is intended for batch processing of a large number of structures that have all been output to a single file of instructions (see example below).

4. Examples

Two examples follow: firstly, the one-off extraction of data from a CIF downloaded from the IUCr Journals Online section, and secondly, the batch processing of a large CIF containing many structures resulting from a CSD search.

4.1. Examine a structure from a recent *Acta Crystallographica* journal

The IUCr Crystallography Journals Online section contains CIF data for structures that appear in IUCr journals. Locate the CIF and the associated structure factors that you are interested in and save them in a new folder with an appropriate name, *e.g.* qa0401.cif and qa0401Isup2.fcf.

1. To extract a single structure from the CIF, run 'cif2cry −f −o structure.dat qa0401.cif'. To extract the structure factors run 'cif2cry −f −o hkl.dat qa0401Isup2.fcf '.

2. Start *CRYSTALS* in the new folder and type '#use structure.dat', and then '#use hkl.dat'. Alternatively, the *CRYSTALS* dialog can be used to carry out steps 1 and 2.

3. The structure and data can now be analysed and re-refined in *CRYSTALS*, and also viewed in *CAMERON* (Watkin *et al.*, 1996), a package for plotting anisotropic displacement parameters and crystal packing.

4.2. Batch processing CSD data

Batch processing of a large number of structures may be useful if there is a simple calculation that *CRYSTALS* can carry out on each structure. This example generates an anisotropic shape tensor for each molecule, which will enclose all of the atoms present (Fig. 2) and thus provide a simple representation of molecular shape and packing.

1. To begin the procedure, the program *CONQUEST* (Bruno *et al.*, 2002) is used to search for and obtain a list of interesting structures.

2. From the *CONQUEST* File menu, choose the option 'Export Entries as . . . '. Choose CIF as the export format. Choose a filename (*e.g.* search1.cif) and click the Save button.

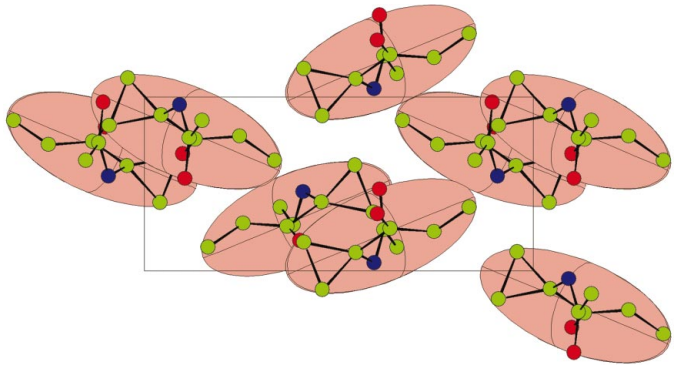

Figure 2
A plot of the anisotropic shape tensor, generated to fit the molecular shape.

3. All structures are to be extracted to a single file for *CRYSTALS* to process, so run 'cif2cry −a −o crysin.dat search1.cif'.

4. Create a text file called cifproc.dat containing a set of instructions to be executed after each set of data has been input. For example, if we were interested in the overall shapes of molecules in a related series, the following *CRYSTALS* instructions replace a molecule with a single atom whose anisotropic displacement parameters are replaced with a shape tensor that is calculated to enclose the molecule, thus representing some properties of the molecular shape:

```
# Replace whole molecule by a single
# atom at the centroid with a large
# enveloping ADP.
\EDIT
CENTROID 1 ALL
SELECT TYPE eq QC
END
# Output data for analysis.
# First the title :
\SCRIPT punchtitle
# Then the new atom with its adp parameters
# (this command outputs atom list):
\PUNCH 5
END
```

5. To process all the data in one go, start *CRYSTALS* in the folder containing crysin.dat and cifproc.dat and type '#use crysin.dat'. The output will be appended to a *CRYSTALS* output file (bfile.pch by default).

5. Programs and availability

The *CIF2CRY* executable for Win32 platforms can be obtained as part of the distribution of the *CRYSTALS* software from http://www.xtl.ox.ac.uk/crystals.html. The source code is also included as required under the CIFtbx licence. *CIF2CRY* is distributed free of charge.

The authors thank the National Science Foundation (grant No. DMR-0089257) for partial support of the work carried out at Brandeis University.

References

Betteridge, P. W., Carruthers, J. R., Cooper, R. I., Prout, K. & Watkin, D. J. (2003). *J. Appl. Cryst.* **36**, 1487.
Bruno, I. J., Cole, J. C., Edgington, P. R., Kessler, M., Macrae, C. F., McCabe, P., Pearson, J. & Taylor, R. (2002). *Acta Cryst.* B**58**, 389–397.
Hall, S. R. (1993). *J. Appl. Cryst.* **26**, 482–494.
Hall, S. R., Allen, F. H. & Brown, I. D. (1991). *Acta Cryst.* A**47**, 655–685.
Sheldrick, G. M. (1998). *SHELX*97-2. Institut für Anorganische Chemie der Universität, Tammannstrasse 4, D-37007 Göttingen, Germany.
Spek, A. L. (1998). *PLATON*. Utrecht University, The Netherlands.
Watkin, D. J., Prout, C. K. & Pearce, L. J. (1996). *CAMERON*. Chemical Crystallography Laboratory, University of Oxford, UK.