ACTA ABP BIOCHIMICA POLONICA

Vol. 60, No 4/2013
591–593
on-line at: www.actabp.pl

Regular paper

Personalization of structural PDB files

Tomasz Woźniak[1] and Ryszard W. Adamiak[1,2]✉

[1]Institute of Bioorganic Chemistry, Polish Academy of Sciences, Poznań, Poland, [2]European Center for Bioinformatics and Genomics, Institute of Computing Science, Poznan University of Technology, Poznań, Poland

PDB format is most commonly applied by various programs to define three-dimensional structure of biomolecules. However, the programs often use different versions of the format. Thus far, no comprehensive solution for unifying the PDB formats has been developed. Here we present an open-source, Python-based tool called PDB*inout* for processing and conversion of various versions of PDB file format for biostructural applications. Moreover, PDB*inout* allows to create one's own PDB versions. PDB*inout* is freely available under the LGPL licence at http://pdbinout.ibch.poznan.pl

Key words: structural bioinformatics, PDB file, format, conversion

Received: 04 June, 2013; revised: 12 November, 2013; accepted: 03 December, 2013; available on-line: 17 december, 2013

INTRODUCTION

Structural studies based on NMR and X-Ray crystallography have become an important tool for studying the structure of molecules. Development of computational methods like molecular dynamics methods, structure prediction, docking and structure analysis tools resulted in the development of numerous programs that work on structural data. Atom coordinates of molecules, which are represented in different file formats. Conversion between these formats is being assured by the programs like Open Babel (O'Boyle *et al.,* 2011) or fconv (Neudert & Klebe, 2011). However, most popular PDB file format has multiple variants since some programs do not adhere to PDB format guidelines (Berman *et al.,* 2000). This concerns especially coordinate section.

Although one may manually introduce these changes in PDB file, changing the ATOM section of the PDB file appears a serious problem if there are hundreds of files to convert. Recently, we have encountered the problem when developing a new method and a server for an automatic prediction of RNA 3D structures (Popenda *et al*., 2012), and a novel approach for quality analysis of RNA structural models (Lukasiak *et al.* 2013). Some tools like MMTSB (Feig *et al.,* 2004), VEGA (Pedretti *et al.,* 2002) have been offered to change some characteristics of PDB format between two versions. However, there is no conversion tool that can process most of the known PDB format versions.

Here, we present PDB*inout* program that allows to transform a complete ATOM section of any defined version of PDB file into recommended PDB format, or some more predefined formats used in popular programs like Amber (Case *et al.,* 2012), CHARMM (Brooks *et al.,* 2009), XPLOR (Schwieters *et al.,* 2006) or CYANA (Herrmann *et al.,* 2002). One can also generate new personalized versions of PDB formats by using semi-automatic detection routine that may help to detect atom and residue names, residue atoms order, and other PDB format characteristics.

MATERIALS AND METHODS

PDB*inout* program was implemented using Python programming language, version 2.7 (http://www.python.org/). Module PyYaml (http://pyyaml.org/) was used to apply clear and consistent data format for program parameters. For documentation purposes, TiddlyWiki (http://www.tiddlywiki.com/) was utilized. PDB*inout* is also fully documented with docstrings. Although the program is easy to use, short tutorial is available. For the unit tests, Logbook 0.3 (http://pypi.python.org/pypi/Logbook/) and setuptools (http://pypi.python.org/pypi/setuptools/) were used. PDB*inout* program is offered for Linux, Windows, MacOs and Solaris operating systems.

Usage example: python -m PDB*inout* -i myinput.pdb -f myformat -o myoutput.pdb

Comprehensive unit tests were created to avoid errors. PDB*inout* program was evaluated by conversion of 1000 randomly selected PDB files from default PDB format to randomly selected one. Additional PDB files containing different molecules and formatted in various formats were added to the test set to check format recognition abilities. The test set was also enlarged to include the files considered earlier (Hamelryck & Manderick, 2003) as generating difficulties in processing. Conversion of 1000 randomly selected structures took 29 minutes and 54 seconds on 3.3 GHz PC, on average about 1.8 seconds per structure. Selected output files were tested by importing them into a specific program. All tests were run in Linux Ubuntu environment. Development packages containing unit tests and tests for various formats are given for download.

RESULTS AND DISCUSSION

For conversion purposes, coordinate section of PDB file is represented using a hierarchic data structure (pdb/model/molecule/ter/chain_break/residue/atom). Unlike in biopython, conversion demands clear definition of the chain fragments terminated by TER mark or defined by missing residues. These fragments are represented by ter and chain_break sections.

The data defining each format are grouped in the dictionary data structure and written in Yaml format (http://www.yaml.org/). Each format definition has

✉e-mail: adamiakr@ibch.poznan.pl

Abbreviations: PDB*inout,* Python programming language

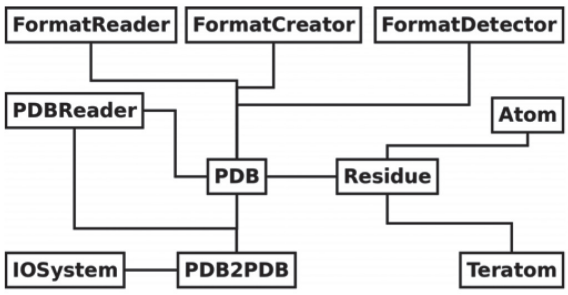

Figure 1. UML class diagram.

many parameters describing, for example, if multiple models are allowed, or should TER mark be introduced in the target format. The most important part of the format description are atoms and residues sections. Those sections are needed to convert atom and residue names to standardized ones, and later, to the names specified in the output format. PDB*inout* can also process protonation states of the amino acids residues from the input PDB file and use them in the output format. Declaration of the input format is not necessary — the program will recognize it automatically, and search for lacking data in other formats. In case of modified residues, both residue and atom names are kept unchanged. In order to change this default option, a user can introduce a naming scheme for these residues in the format file. To allow multi file conversions, command line user interface was implemented.

PDB*inout* allows to create one's own PDB versions. In order to generate new PDB formats, semi-automatic procedure is offered. This procedure may be applied to include formats used by newly created programs or to match specific user's needs. A user has also a possibility to create or modify format definition manually (as described in the documentation). In case of difficulties with processing input, the PDB file or data format errors as well as warnings are generated and written in the log file.

Figure 1 shows a class diagram of the PDB*inout* program structure. PDB class is a main class which both organizes data structure and generates the output file. PDBReader class processes the input file. Both FormatReader and FormatDetector classes are used to read the known format definitions, and detect the format most similar to the already imported PDB file. In case of a new format generation, both FormatDetector and FormatCreator are used. Residue and Atom classes are responsible for proper generation of the atom coordinate lines. Teratom is used to create the lines containing TER mark. User parameters are processed by PDB2PDB class, while IOSystem manages an access to the PDB file.

The PDB*inout* conversion program is a powerful and easy-to-use tool that allows to avoid manual and repeatable tasks of adjusting PDB format to the user's needs. It can be utilized for changing atom names in large set of PDB files generated, for example, in the molecular dynamics simulations. In addition, employing text mode, PDB*inout* converter may be used for interconnecting the software working on PDB files. Program code may be also incorporated in more complex programs. This solution has advantage for the users interested in automatisation of structural software pipeline (Popenda *et al.,* 2012). Moreover, unique ability of PDB*inout* to process various PDB files without prior user knowledge of the input format version makes it useful for usage in the programs or webservers processing PDB files.

PDB conversion lies in performing a set of various, sometimes tedious operations. Some PDB conversion tasks to be encountered are quite complex, and our experience shows that they cannot be executed using general purpose grep and sed programs, or are troublesome using one's own Python/Perl/AWK scripts. Here, we would like to point to several such more complex tasks, which can be conveniently solved with PDB*inout*. Moreover, two of them are given in more details in the Supplementary Materials (at www.actabp.pl).

Case 1. Concerns a situation where the alternate location indicator place (column 17 within PDB file) is occupied by the name of an atom or a residue. PDB*inout* is able to detect and correct that case automatically; thus, all atoms and residue names are always properly parsed and generated.

Case 2. Files used by Amber program (Case *et al.,* 2012) need to contain information about strand orientation. This information is assured by adding numbers "3" and "5" at the end of the residue name of 3' and 5' terminal residues of RNA or DNA, and letters "C" and "N" in case of terminal protein residues. PDB*inout* conversion tool can properly parse and generate strand orientation data, taking into account models, molecules and continuous fragments of chains (see Supplementary Materials, Case 2).

Case 3. PDB files are often lacking or exceeding ATOM line elements (strand identificator, atom type, and specific SeqID). However, some programs require presence or removal of these elements. PDB*inout* allows to correct this according to the output format definition requested.

Case 4. Atom names in various formats are sometimes misguiding, especially in case of hydrogen atoms. For example, when working with MacroMoleculeBuilder (MMBUILDER) (Flores *et al.,* 2011), atom names like HB2 and HB3 have to be converted into 1HB and 2HB accordingly (see Supplementary Materials, Case 4). PDB*inout* assures that atom and residue names are always properly changed into corresponding names in the target format.

Case 5. While working in CHARMM program (Brooks *et al.,* 2009) with the use of trajectory data formerly generated in Amber (Case *et al.,* 2012), assuring the right order of ATOM lines is of importance. PDB*inout* allows either to keep this order intact or to change it. Therefore, in the newly created file ATOM lines are given respective to the atoms order in the target format, and if necessary, subsequently renumbered.

Obviously, PDB*inout* deals with simpler but important tasks, like: covering residues in different protonation states, or forming disulfide bridges, selective deleting hydrogens and pseudoatoms, dealing with the right choice of structural models and placement of END and TER marks.

PDB*inout* is freely available under the LGPL licence at http://pdbinout.ibch.poznan.pl

Acknowledgements

The authors would like to thank Mariusz Popenda and Joanna Sarzyńska for their contribution in testing PDB*inout*. The project has been supported by the National Science Centre Poland, grant 2012/06/A/ST6/00384.

REFERENCES

Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE (2000) The Protein Data Bank. *Nucleic Acids Res* **28**: 235–242.

Brooks BR, Brooks III CL, Mackerell AD, Nilsson L, Petrella RJ, Roux B, Won Y, Archontis G, Bartels C, Boresch S, Caflisch A, Caves L, Cui Q, Dinner AR, Feig M, Fischer S, Gao J, Hodoscek M, Im W, Kuczera K, Lazaridis T, Ma J, Ovchinnikov V, Paci E, Pastor RW, Post CB, Pu JZ, Schaefer M, Tidor B, Venable RM, Woodcock HL, Wu X, Yang W, York DM, Karplus M (2009) CHARMM: The Biomolecular simulation Program. *J Comp. Chem.* **30**: 1545–1615.

Case DA , Darden TA , Cheatham III TE, Simmerling CL, Wang J, Duke RE, Luo R, Walker RC, Zhang W, Merz KM, Roberts B, Hayik S, Roitberg A, Seabra G, Swails J, Goetz AW, Kolossváry I, Wong KF, Paesani F, Vanicek J, Wolf RM, Liu J, Wu X, Brozell SR, Steinbrecher T, Gohlke H, Cai Q, Ye X, Wang J, Hsieh MJ, Cui G, Roe DR, Mathews DH, Seetin MG, Salomon-Ferrer R, Sagui C, Babin V, Luchko T, Gusarov S, Kovalenko A, Kollman PA (2012), AMBER 12. *University of California, San Francisco.*

Feig M, Karanicolas J and Brooks CL III (2004) MMTSB Tool Set: enhanced sampling and multiscale modeling methods for applications in structural biology. *J Mol. Graph. Model* **22**: 377–395.

Flores S, Sherman M, Bruns C, Eastman P, Altman R (2011) Fast flexible modeling of RNA structure using internal coordinates. *Transactions in Computational Biology and Bioinformatics* **8**: 1247–1257.

Hamelryck T, Manderick B (2003) BDB file parser and structure class implemented in Python. *Bioinformatics* **19**: 2308–2310.

Herrmann T, Güntert P, Wüthrich K (2002) Protein NMR structure determination with automated NOE assignment using the new software CANDID and the torsion angle dynamics algorithm DYANA. *J Mol. Biol.* **319**: 209–227.

Lukasiak P, Antczak M, Ratajczak T, Bujnicki JM, Szachniuk M, Adamiak RW, Popenda M, Blazewicz J (2013) RNAlyzer--novel approach for quality analysis of RNA structural models. *Nucleic Acids Res* (2013) [Epub ahead of print] doi: 10.1093/nar/gkt318.

Neudert G, Klebe G (2011) fconv: format conversion, manipulation and feature computation of molecular data. *Bioinformatics* **27**: 1021–1022.

O'Boyle NM, Banck M, James CA, Morley C, Vandermeersch T, Hutchison GR (2011) Open Babel: An open chemical toolbox. *J Cheminform* **3**: 3–33.

Pedretti A, Villa L, Vistoli G (2002) VEGA: a versatile program to convert, handle and visualize molecular structure on Windows-based PCs. *J Mol. Graph. Model* **21**: 47–49.

Popenda M, Szachniuk M, Antczak M, Purzycka KJ, Lukasiak P, Bartol N, Blazewicz J, Adamiak RW (2012) Automated 3D structure composition for large RNAs. *Nucleic Acids Res* **40**: e112.

Schwieters CD, Kuszewski JJ, Clore GM (2006) Using Xplor-NIH for NMR molecular structure determination. *Progr. NMR Spectroscopy* **48**: 47–62.