JOURNAL OF
APPLIED
CRYSTALLOGRAPHY

ISSN 1600-5767

AutoFP: a GUI for highly automated Rietveld refinement using an expert system algorithm based on *FullProf*

Xiaopeng Cui,[a] Zhenjie Feng,[a,b]* Yuan Jin,[a] Yiming Cao,[a] Dongmei Deng,[a] Hao Chu,[c] Shixun Cao,[a,b] Cheng Dong[d] and Jincang Zhang[a,e]

[a]Department of Physics, Shanghai University, Shanghai 200444, People's Republic of China, [b]Shanghai Key Laboratory of High-Temperature Superconductors, Shanghai 200444, People's Republic of China, [c]Department of Physics, California Institute of Technology, Pasadena, CA 91125, USA, [d]National Laboratory for Superconductivity, Institute of Physics, Chinese Academy of Sciences, Beijing 100080, People's Republic of China, and [e]Materials Genome Institute, Shanghai University, Shanghai 200444, People's Republic of China. *Correspondence e-mail: fengzhenjie@shu.edu.cn

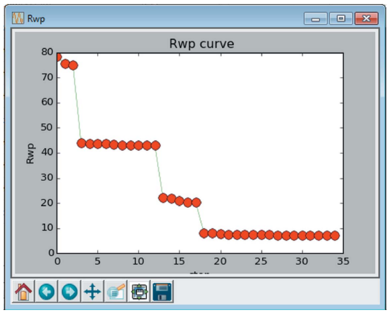
Received 2 May 2015
Accepted 13 August 2015

Edited by K. Chapman, Argonne National Laboratory, USA

Keywords: *AutoFP*; Rietveld refinement; *FullProf*; computer programs.

Supporting information: this article has supporting information at journals.iucr.org/j

AutoFP, a highly automated software toolkit, has been developed to improve the extent of automation of the widely used Rietveld refinement program *FullProf* [Rodríguez-Carvajal (1993). *Physica B*, **192**, 55–69]. An expert system algorithm is used as the control layer to simulate the manual process when *FullProf* is used to perform Rietveld refinement. This enables the program to complete the Rietveld refinement highly automatically. It is shown that the expert system algorithm is a good choice for automating Rietveld refinement. The programming interface is available for advanced users to implement their own acquired experience of refinement or add new Rietveld refinement engines to *AutoFP*. *AutoFP* can be also used as an automated Rietveld refinement engine by other programs. *AutoFP* is an open-source software package developed in Python, and it is user friendly, easy to learn and easy to use.



1. Introduction

Atomic arrangement or crystalline structure is extremely important in determining the properties of solids. Single-crystal growth is not easy and is sometimes impossible for certain compounds. As a result, the powder diffraction method is a powerful tool for crystal structure determination. In powder diffraction crystallography it is difficult to obtain complete information on a crystal structure without using Rietveld refinement, because there are strongly overlapping reflections (Le Bail *et al.*, 1988). Rietveld refinement is a method of analysing powder diffraction data in which the crystal structure is refined by fitting the entire profile of the diffraction pattern to a calculated profile using a least-squares approach (Rietveld, 1969). Rietveld refinement does not involve the intermediate step of extracting structure factors, so patterns containing many overlapping Bragg peaks can be analysed (Young, 1993). In addition, Rietveld refinement uses all the powder diffraction data instead of only the peaks, so it can obtain more information from the powder diffraction data (Rietveld, 1967). Many computer programs have been developed for crystal structure determination and Rietveld refinement (Larson & Von Dreele, 2004; Rodríguez-Carvajal, 1993; Toby, 2001; Akselrud & Grin, 2014; Petříček *et al.*, 2014; Murray *et al.*, 1990; Howard & Hunter, 1998; Izumi, 1989; Bergmann *et al.*, 1998; Coelho, 2004). However, most of these Rietveld programs require intensive user intervention, and there is also a sharp learning curve for new users (Tian *et al.*, 2013). *SrRietveld* (Tian *et al.*, 2013; Tian & Billinge, 2011),

http://dx.doi.org/10.1107/S1600576715015125

which was developed as a part of the *DANSE* project, is a fairly good start for a highly automated software toolkit for Rietveld refinement.

AutoFP uses an expert system algorithm to improve the extent of automation of the widely used Rietveld refinement program *FullProf* (Rodríguez-Carvajal, 1993). An expert system is a computer system that simulates the decision-making ability of a human expert (Jackson, 1998; Liao, 2005). The first expert systems were developed in the 1970s and they boomed in the 1980s (Leondes, 2002; Russell & Norvig, 2009). The manual process of using *FullProf* can be considered as a decision-making process, and an expert system algorithm can be used to simulate that process.

The current Rietveld refinement programs pose three major difficulties for beginners. First, it is difficult to decide the refinement order of parameters (McCusker *et al.*, 1999). Second, divergence often occurs, which destroys the whole outcome of the refinement. Third, the repeated operation of checking and unchecking parameters is inefficient and slow. The expert system algorithm is used to help decide the refinement order of parameters automatically, which solves the first and third difficulties. In order to solve the second difficulty, *AutoFP* saves every best result for reuse when divergence occurs. The use of *FullProf* as the Rietveld refinement engine guarantees the correctness of the Rietveld refinement process. *AutoFP* provides an automation layer controlled by the expert system algorithm and a graphical user interface (GUI). The layer is implemented in Python packages that can manipulate and communicate with the refinement engines. In order to save development time, *AutoFP*'s interface with *FullProf* is based on the *SrRietveld* program (Tian *et al.*, 2013). The expert system algorithm takes $R_{\rm wp}$ as the target function to be minimized and takes full advantage of the existing knowledge of refinement to adjust the order and values of the parameters. Therefore, the manual refinement process can be simulated automatically, and the results can be displayed and analysed by *AutoFP*.

It should be pointed out that the expert system algorithm is a classic simulation decision system, and in this paper it is shown to be successful at automating *FullProf*. Other appropriate machine-learning artificial intelligence algorithms, such as the Q-learning algorithm (Watkins & Dayan, 1992), might also be good strategies to improve the extent of automation of Rietveld refinement programs.

Although *AutoFP* has automated the *FullProf* Rietveld refinement, it is important to note that even a stable refinement with a low R factor does not guarantee a correct result, and that there is no replacement for human expertise in evaluating the correctness of a structure based on Rietveld refinement. In particular, an incorrect initial choice of space group, incorrectly assigned atoms *etc.* can never be corrected by this system, and their effects can prove extremely subtle. Furthermore, *AutoFP* is designed to automate the refinement from the correct `.pcr` format file (the input parameters for *FullProf*). Users should be sure of the validity of the `.pcr` file before using *AutoFP*, or no meaningful result can be obtained from *AutoFP*.

2. *AutoFP*

2.1. Design principles

AutoFP has been developed in the Python language (Sanner, 1999; http://www.python.org) using object-oriented programming (OOP) concepts. The OOP design enables *AutoFP* to be easily maintained and extended. Python is used extensively in scientific software development. It is a cross-platform language and suitable for fast development. *NumPy* (http://www.numpy.org) and *Matplotlib* (http://matplotlib.org) are well designed packages for scientific programming and visualization. In *AutoFP*, the *Matplotlib* package is used to plot the $R_{\rm wp}$ curve, while the *NumPy* package is used to handle the arrays. The user interface (UI) of *AutoFP* was developed using the *PyQt* package (http://sourceforge.net/projects/pyqt). Qt is a cross-platform application and UI framework, and *PyQt* is the Python bindings for Qt. The features of *Python* and *PyQt* can help *AutoFP* to run on more operating system platforms.

AutoFP is based on the following design principles:

(1) Simulate the decision-making ability of a human expert to complete the Rietveld refinement.

(2) Make the refinement process highly automated.

(3) Be open source and extensible.

To address principle (1), the control layer of *AutoFP* uses an expert system. Expert systems are designed to solve complex problems by reasoning based on knowledge, represented primarily as if–then rules. The knowledge base represents facts and rules. The inference engine applies the rules to the known facts to deduce new facts. Inference engines can also include explanation and debugging capabilities (Ryan, 2014).

In the *AutoFP* program, the refinement engine is *FullProf*, the inference engine is a control layer based on the expert system algorithm, and the knowledge base is the existing acquired experience of refinement and strategies. We collect the refinement experiences and strategies, then simplify them as if–then rules, which form the refinement knowledge base. When *AutoFP* starts a refinement, the control layer automatically arranges the refined parameters in order, automatically checks and unchecks the refined parameters, monitors the status of the *FullProf* refinement, and deals with all kinds of refinement problems through the refinement knowledge base, such as refinement divergence, inner errors *etc.*, to complete complex refinement tasks highly automatically.

For principle (2), to simplify and automate the refinement process, *AutoFP* is designed to be user friendly and reduce the user's input as much as possible. In order to reduce the steep learning curve of Rietveld refinement for new users, the if–then rules of the refinement knowledge base are built into the *AutoFP* program code. Therefore, users can skip inputting their own refinement knowledge. The control layer saves every intermediate result of *FullProf*. If the refinement detects errors or divergences, the control layer can automatically revive the refinement process from the best saved intermediate result. The control layer tries to find the minimum $R_{\rm wp}$ through this strategy.

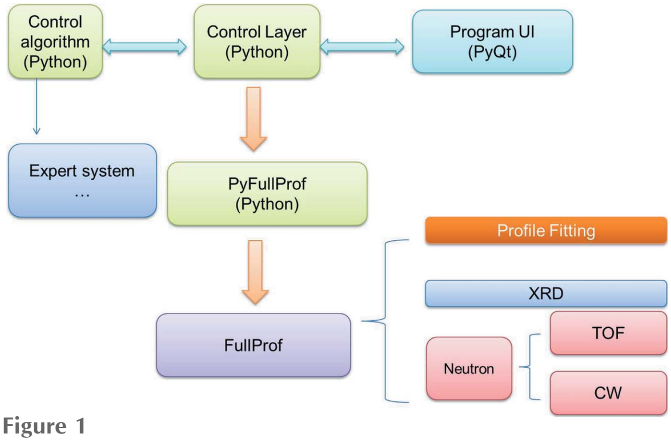


Figure 1
The architecture of *AutoFP*. The control layer, expert system algorithm and *PyFullProf* are Python packages in the *AutoFP* program. *FullProf* is the underlying refinement engine which can handle profile matching tasks for X-ray diffraction (XRD), and neutron continuous wave (CW) or time-of-flight (TOF) tasks. See text for further details.

For principle (3), it is possible for advanced users to develop their own refinement strategies and contribute to the development of *AutoFP*.

2.2. Implementation

The architecture of *AutoFP* and its several constituent programming units is shown in Fig. 1. The *AutoFP* interface is a fully featured GUI that enables the user to load a `.pcr` file, configure the automated refinement strategy and output the resulting files. It can also display the refinement results. The control layer is the core of *AutoFP*, and it controls the UI and uses the expert system algorithm to control the *FullProf* refinement engine. *PyFullProf* is a Python program that reads and writes the files of *FullProf*. *PyFullProf* is based on the *SrRietveld* project (Tian *et al.*, 2013).

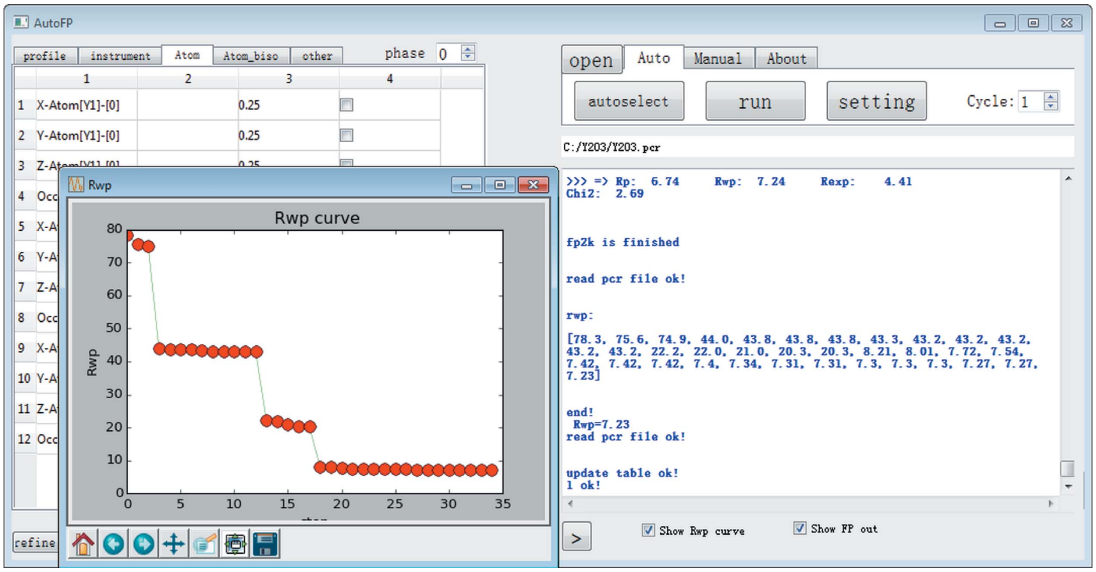
Fig. 2 shows a screenshot of one of the main interfaces of *AutoFP* during an automatic refinement of Y_2O_3 powder diffraction data, as described in detail in the figure caption.

AutoFP incorporates most of the functionality supported by the current versions of *FullProf* and is developed mainly for Rietveld analysis (structure profile refinement) of neutron [constant wavelength (CW) or time-of-flight (TOF)] or X-ray powder diffraction data collected at constant or variable steps in the scattering angle 2θ. The program uses the expert system algorithm to select a good refinement strategy (McCusker *et al.*, 1999), and it captures errors and attempts to recover automatically when a refinement divergence or a *FullProf* error occurs. This increases the robustness of the automatic refinement process. The program takes $R_{\rm wp}$ as the target function to minimize and tries to obtain the best refinement result. In addition, the refinement speed of *AutoFP* is very fast; commonly, it only takes a few minutes to perform a classical refinement (dual core i5-4300U 1.90 GHz, 8 GB DDR3 RAM).

In order to improve its ease of use, *AutoFP* provides flexibility and extensibility to advanced users. *AutoFP* is modularly designed and focuses on extensibility, so the default behaviour of the software can be set individually. *AutoFP* owns the command mode and knowledge interface. Therefore, other programs can use it as an automatic Rietveld refinement engine, and advanced users can implement their acquired refinement experience in it.

2.3. The program flow chart

There are many parameters in Rietveld refinement. One kind of parameter order is called a refinement strategy.



phase 0

Cycle: 1

	1	2	3	4
1	X-Atom[Y1]-[0]		0.25	□
2	Y-Atom[Y1]-[0]		0.25	□
3	Z-Atom[Y1]-[0]		0.25	□
4	Occ			
5	X-A			
6	Y-A			
7	Z-A			
8	Occ			
9	X-A			
10	Y-A			
11	Z-A			
12	Occ			

C:/Y203/Y203.pcr

```
>>> => Rp:  6.74     Rwp:  7.24     Rexp:    4.41
Chi2:  2.69



fp2k is finished


read pcr file ok!


rwp:

[78.3, 75.6, 74.9, 44.0, 43.8, 43.8, 43.8, 43.3, 43.2, 43.2, 43.2,
43.2, 43.2, 22.2, 22.0, 21.0, 20.3, 20.3, 8.21, 8.01, 7.72, 7.54,
7.42, 7.42, 7.42, 7.4, 7.34, 7.31, 7.31, 7.3, 7.3, 7.3, 7.27, 7.27,
7.23]


end!
 Rwp=7.23
read pcr file ok!


update table ok!
1 ok!
```

☑ Show Rwp curve ☑ Show FP out

Figure 2
A screenshot of *AutoFP* during refinement of Y_2O_3 powder diffraction data. In the ‘Auto’ tab in the right-hand panel, the ‘autoselect’ button selects and checks the refined parameters automatically. The ‘setting’ button sets the strategy of refinement and other options, while the ‘run’ button runs the automated refinement. The ‘open’ button loads the `.pcr` file into *AutoFP*. The bottom right-hand textbox displays messages about the refinement from both *FullProf* and *AutoFP*. The ‘Rwp curve’ window shows the $R_{\rm wp}$ values from the automatic refinement process.

Different refinement strategies will generate very different refinement results, some of which will be **near perfect** while others will be considerably worse. *AutoFP* is designed to automate the refinement process with a specific refinement strategy, and this can be modified by users according to their needs. To assist users, *AutoFP* has a default refinement strategy which is based on the work of McCusker *et al.* (1999), Rietveld (1969) and Young (1993).

AutoFP will refine the profile parameters first, then the structural parameters and preferred orientation. The order in which the parameters are turned on in the default strategy is as follows: 2θ zero correction, unit-cell parameters, sample displacement, simple background correction, scale factor, peak width, peak shape function, peak asymmetry, complex background correction, atomic positions, atomic displacement parameters, occupancy parameters, preferred orientation.

Although *AutoFP* provides a default strategy for general Rietveld refinement, it cannot satisfy all cases. So, sometimes writing one's own strategy is more effective. Since *AutoFP* is open source, modifying the default strategy to the user's own strategy is very easy.

The program flow chart is displayed in the following scheme.

```
1. Read the PCR file and decide the type of Rietveld refinement (XRD, Neutron-CW,
   Neutron-TOF)

2. Select the refinement parameters and configure the automatic options

3. Start automatic Rietveld refinement

   Loop begins:

       Read the refinement strategy from knowledge library, save the PCR file, the
       parameters of Rietveld refinement consist of three main classes: profile parameters,
       structural parameters and preferred orientation.
      Sub-loop begins:

      Use expert system to check next parameter and run FullProf
        a. Profile parameters
             peak shape: scale, peak shape function, peak width as a function of $2\theta$, peak
             width as a function of hkl (anisotropic line broadening), peak asymmetry as a
             function of $2\theta$ (axial divergence);
             peak position: zero point error, sample displacement, unit cell parameters
             background
        b. Structural parameters
             atomic positions, atomic displacement factors, atomic occupancy parameters
             Occ
        c. Sample
             preferred orientation

          if: $R_{wp}$ ascends or FullProf meets error
               then: backup the PCR file
               else: save the new PCR file
      End of the sub-loop

   End of the loop (when $R_{wp}$ convergence or the specified number of cycles)

4. End the program
```

2.4. How to use *AutoFP*

Once the .pcr format file and diffraction data files have been prepared correctly, there are two ways to run *AutoFP*.

(1) UI mode. The UI mode needs just three steps: click the 'open' button to load the .pcr file, click the 'autoselect' button to select the parameters needed to refine automatically, and then click the 'run' button to start the automatic refinement process.

Table 1
Rietveld refinement results for Y_2O_3 using *AutoFP*.

Sample	Space group	Unit cell (Å)	*R* factors
Y_2O_3	$Ia\bar{3}$	$a = b = c = 10.60397$ (1)	$R_p = 0.0674$ $R_{wp} = 0.0723$ $\chi^2 = 2.69$

Table 2
Rietveld refinement results for $Dy_{0.5}Sr_{0.5}MnO_3$ using *AuoFP*.

Sample	Space group	Unit cell (Å)	*R* factors
$Dy_{0.5}Sr_{0.5}MnO_3$	*Pbnm*	$a = 5.40196$ (4) $b = 5.41494$ (0) $c = 7.62853$ (2)	$R_p = 0.105$ $R_{wp} = 0.128$ $\chi^2 = 3.56$

(2) Command mode. The automated refinement command line is `autofp -a file.pcr`. The `-a` option makes *AutoFP* automatically select the parameters needed to perform the refinement.

3. Example

The following examples are chosen to show the features of *AutoFP*.

3.1. Y_2O_3

This example is the subject of the article by Santos *et al.* (2005). It shows how to perform a simple Rietveld refinement of a high-symmetry single-phase oxide material using laboratory X-ray diffraction data with *AutoFP*. A high-quality diffraction pattern was recorded on a bulk sample of Y_2O_3 using a Siemens D5000 diffractometer equipped with a Cu tube, graphite-diffracted beam monochromator and scintillation counter (FPSchool 2013; http://www.ill.eu/en/press-and-news/past-events/2013/fpschool-2013/documentation/.

The refinement results are shown in Fig. 3 and Table 1. The curve of R_{wp} is shown in Fig. 4. The space group is $Ia\bar{3}$, $a = b = c = 10.60397$ (1) Å. Fig. 4 shows that *AutoFP* can complete the

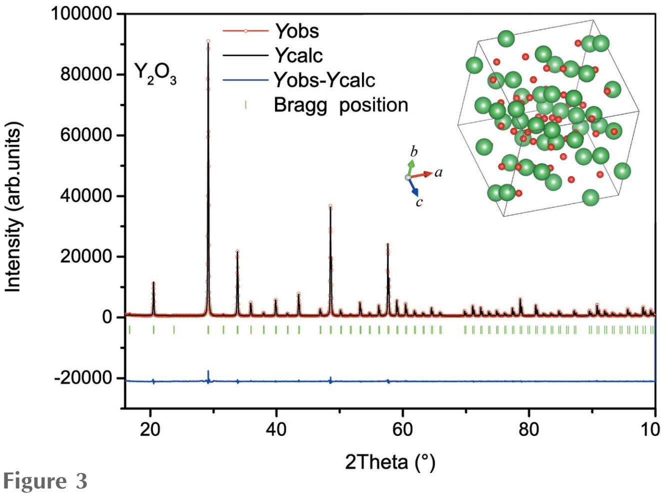


Figure 3
The refinement result for Y_2O_3 using *AutoFP*. The *R* factors are $R_p = 0.0674$ and $R_{wp} = 0.0723$, and $\chi^2 = 2.69$. The space group is $Ia\bar{3}$, with $a = b = c = 10.60397$ (1) Å.

refinement automatically, solve the divergence cases or *FullProf* errors, and take R_{wp} as the target function to minimize.

As is shown in Fig. 3 and Table 2, the refinement result with *AutoFP* is **excellent**.

3.2. $Dy_{0.5}Sr_{0.5}MnO_3$

To demonstrate a typical application, automatic refinement was carried out on X-ray powder diffraction patterns measured from $Dy_{0.5}Sr_{0.5}MnO_3$ (DSMO) at room temperature using a Rigaku 18 kW D/MAX 2550 powder diffractometer with Cu $K\alpha$ radiation. The $Dy_{0.5}Sr_{0.5}MnO_3$ powder was synthesized by the standard solid-state reaction method using $SrCO_3$ (99.99%), Dy_2O_3 (99.99%) and MnO_2 (99.95%). The diffractometer was operated at 40 kV and 200 mA, the 2θ scan range was from 10 to 120° with a step size of 0.02°, and the count time was 4 s per step.

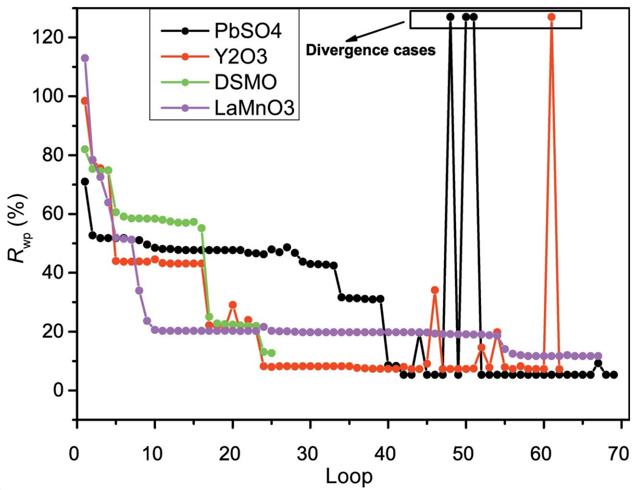


Figure 4
The R_{wp} curves from *AutoFP* refinement for the examples of $PbSO_4$, Y_2O_3, DSMO and $LaMnO_3$. The plot shows that *AutoFP* can complete the refinement tasks automatically and solve the divergence problem to obtain minimum R_{wp} values. It should be pointed out that the divergence values are not shown in the 'Rwp curve' window when *AutoFP* is running, because a very large divergence value will make the R_{wp} curve into a single horizontal line. The whole set of R_{wp} data can be found in the output file.

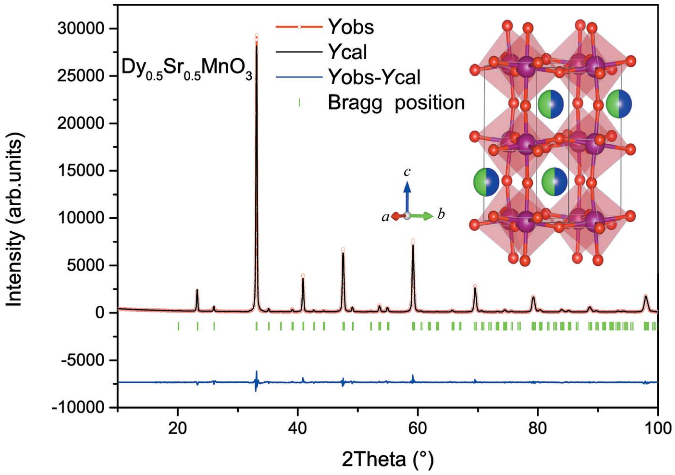


Figure 5
The refinement result for $Dy_{0.5}Sr_{0.5}MnO_3$ using *AutoFP*. The *R* factors are R_p = 0.105 and R_{wp} = 0.128, and χ^2 = 3.56. The space group is *Pbnm*, with *a* = 5.40196 (4), *b* = 5.41494 (0) and *c* = 7.62853 (2) Å.

Table 3
Riteveld refinement results for examples using *AutoFP*.

Sample	R_p	R_{wp}	χ^2	Diffraction mode
Y_2O_3	0.0727	0.0674	2.69	X-ray
$PbSO_4$	0.0499	0.0578	3.63	Neutron-CW
DSMO	0.127	0.105	3.56	X-ray
$LaMnO_3$	0.117	0.150	1.40	Neutron-CW
$YBaCuCoO_7$	0.0764	0.0731	1.59	Neutron-CW

The refinement result is shown in Fig. 5 and the curve of R_{wp} is shown in Fig. 4. Because DSMO is a new compound, the $La_xSr_{1-x}MO_3$ structure (Pinsard *et al.*, 1997) was used as the initial structure to refine DSMO. The space group is *Pbnm*, with *a* = 5.40196 (4), *b* = 5.41494 (0) and *c* = 7.62853 (2) Å. Fig. 4 shows that *AutoFP* can complete the refinement correctly and automatically, and can accomplish the task of new structure refinement.

3.3. Other tests

$PbSO_4$. This example is the subject of the IUCr Round Robin (Hill, 1992), and was used to prove that *AutoFP* can make an automatic Rietveld refinement of neutron-CW diffraction data.

$LaMnO_3$. The neutron-CW powder diffraction pattern of $LaMnO_3$ (Rodriguez-Carvajal *et al.*, 1998) was measured at the LLB diffractometer G4.2 with λ = 2.59 Å.

$YBaCuCoO_7$. The neutron powder diffraction pattern was measured at ILL with the instrument D1A on a sample of the compound $YBa_2Cu_{3-x}Co_xO_{7+y}$, with x = 0.33 (FPschool 2013; http://www.ill.eu/press-and-news/past-events/2013/fpschool-2013/documentation).

The refinement results and the drop-down curves of R_{wp} are shown in Table 3 and Fig. 4.

4. Software distribution

AutoFP is open-source software distributed under the GPLv3 licence. It is free to use, subject to the copyright restrictions and disclaimer, though we ask that papers **reporting results obtained** using *AutoFP* cite this paper, as well as the paper describing the particular refinement engine used (*FullProf*). The *AutoFP* program and more information on it can be obtained from the project web pages (http://pmedia.shu.edu.cn/autofp) or by contacting Xiaopeng Cui (xpclove@126.com) or Zhenjie Feng (fengzhenjie@shu.edu.cn).

Acknowledgements

We thank the authors of *FullProf* and *SrRietveld* for their enormous efforts in developing those programs. We acknowledge funding provided by the Shanghai Pujiang Program (grant No. 13PJD015), the Science and Technology Commission of Shanghai Municipality (grant Nos.

13ZR1415200, 13JC1402400 and 11DZ1100305) and the National Natural Science Foundation of China (NSFC, grant Nos. 51372149, 51371111, 51302249 and 11204171).

References

Akselrud, L. & Grin, Y. (2014). *J. Appl. Cryst.* **47**, 803–805.

Bergmann, J., Friedel, P. & Kleeberg, R. (1998). *IUCr Commission on Powder Diffraction Newsletter*, No. 20, pp. 5–8.

Coelho, A. A. (2004). *TOPAS*. Bruker AXS GmbH, Karlsruhe, Germany.

Hill, R. J. (1992). *J. Appl. Cryst.* **25**, 589–610.

Howard, C. J. & Hunter, B. A. (1998). *A Computer Program for Rietveld Analysis of X-ray and Neutron Powder Diffraction Patterns*, pp. 1–27. Lucas Heights Research Laboratories, New South Wales, Australia.

Izumi, F. (1989). *Rigaku J.* **6**, 10–20.

Jackson, P. (1998). *Introduction to Expert Systems*, 3rd ed., p. 2. Harlow: Addison Wesley.

Larson, A. C. & Von Dreele, R. B. (2004). *GSAS*. Report LAUR 86-748. Los Alamos National Laboratory, New Mexico, USA.

Le Bail, A., Duroy, H. & Fourquet, J. L. (1988). *Mater. Res. Bull.* **23**, 447–452.

Leondes, C. T. (2002). *Expert Systems: The Technology of Knowledge Management and Decision Making for the 21st Century*. Waltham, Massachusetts: Academic Press.

Liao, S. (2005). *Expert Syst. Appl.* **28**, 93–103.

McCusker, L. B., Von Dreele, R. B., Cox, D. E., Louër, D. & Scardi, P. (1999). *J. Appl. Cryst.* **32**, 36–50.

Murray, A. D., Cockcroft, J. K. & Fitch, A. N. (1990). *Powder Diffraction Program Library (PDPL)*. University College London, London, England.

Petříček, V., Dušek, M. & Palatinus, L. (2014). *Z. Kristallogr.* **229**, 345–352.

Pinsard, L., Rodríguez-Carvajal, J. & Revcolevschi, A. (1997). *J. Alloys Compd.* **262–263**, 152–156.

Rietveld, H. M. (1967). *Acta Cryst.* **22**, 151–152.

Rietveld, H. M. (1969). *J. Appl. Cryst.* **2**, 65–71.

Rodríguez-Carvajal, J. (1993). *Physica B*, **192**, 55–69.

Rodríguez-Carvajal, J., Hennion, M., Moussa, F., Moudden, A. H., Pinsard, L. & Revcolevschi, A. (1998). *Phys. Rev. B*, **57**, R3189–R3192.

Russell, S. & Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River: Prentice Hall.

Ryan, M. (2014). *The Digital Mind: An Exploration of Artificial Intelligence.* pp. 27–29. Createspace Independent Publishing.

Sanner, M. (1999). *J. Mol. Graph. Model.* **17**, 57–61.

Santos, C., Strecker, K., Suzuki, P., Kycia, S., Silva, O. & Silva, C. (2005). *Mater. Res. Bull.* **40**, 1094–1103.

Tian, P. & Billinge, S. J. L. (2011). *Z. Kristallogr.* **226**, 898–904.

Tian, P., Zhou, W., Liu, J., Shang, Y., Farrow, C. L., Juhás, P. & Billinge, S. J. L. (2013). *J. Appl. Cryst.* **46**, 255–258.

Toby, B. H. (2001). *J. Appl. Cryst.* **34**, 210–213.

Watkins, C. J. C. H. & Dayan, P. (1992). *Mach. Learn.* **8**, 279–92.

Young, R. A. (1993). *The Rietveld Method*, Vol. 5. Oxford University Press.