Available online at www.sciencedirect.com
SCIENCE DIRECT®
Ultramicroscopy 102 (2005) 233–237
ultramicroscopy
www.elsevier.com/locate/ultramic

EDM 1.0: Electron direct methods

R. Kilaas[a], L.D. Marks[b,*], C.S. Own[b]

[a]National Center for Electron Microscopy, Lawrence Berkeley National Labs, Berkeley, CA, USA
[b]Department of Materials Science and Engineering, Northwestern University, 2225 N Campus Drive, Evanston, IL 60201, USA

Received 21 July 2004; received in revised form 30 September 2004; accepted 12 October 2004

Abstract

A computer program designed to provide a number of quantitative analysis tools for high-resolution imaging and electron diffraction data is described. The program includes basic image manipulation, both real space and reciprocal space image processing, Wiener-filtering, symmetry averaging, methods for quantification of electron diffraction patterns and two-dimensional direct methods. The program consists of a number of sub-programs written in a combination of C++, C and Fortran. It can be downloaded either as GNU source code or as binaries and has been compiled and verified on a wide range of platforms, both Unix based and PC's. Elements of the design philosophy as well as future possible extensions are described.

1. Introduction

It is now very well established that numerous image processing strategies can improve the amount of information that can be extracted from high-resolution electron microscopy (HREM), e.g. [1–5]. It is also well established that one can use direct methods with electron diffraction data to both extend the effective resolution of the microscope when combined with HREM [6–10] or to solve structures directly just from the diffraction intensity data [11–19]. However, all these computer analysis strategies have to date relied on either commercial codes such as Semper [2,4] or Crisp [20] or specialized research software which have not been generally available, are not well integrated and in many cases the research software is not portable, poorly (if at all) documented and only available at a few locations. (An important exception is the VEC package [8] which is freely available, user-friendly and operates in a Windows environment albeit without source code being available.).

To try and make these higher-level analysis methods available to a larger number of users, we embarked on a project to try and create a user-friendly free suite of programs that incorporate many of the more advanced HREM analysis and direct methods approaches. The program, called

*Corresponding author. Tel.: +1 84 7491 3996; fax: +1 84 7491 7820.
E-mail address: L-marks@northwestern.edu (L.D. Marks).

0304-3991/$ - see front matter © 2004 Published by Elsevier B.V.
doi:10.1016/j.ultramic.2004.10.004

electron direct methods (EDM), was beta-tested in the fall of 2003 and the first version (1.0) was released in February 2004. At the time of writing there have been more than 300 unique downloads of the software, most for PC's with smaller numbers for Macintosh, Linux and Unix systems. The intention of this note is to provide some general information about the design philosophy, structure of the code as well as future possible extensions. More information is available at the main site, http://www.numis.nortwestern.edu/edm.

2. Design strategy and implementation

Any software designed to be useable on a large number of different computers, and able to survive changes in operating systems with time has to satisfy a number of constraints.

1. The software should be written in a high-level machine independent language which is normally provided on all systems in a standard configuration.
2. The software must be fairly easily adaptable to different systems with minimal work either by the user or developer.
3. The software must have help and other information in a portable format.
4. The software must include a reasonable number of examples.

The approach we decided to implement was to use X-windows as the common graphical environment. The conventional GNU autoconfigure structure [21] serves to maintain Makefiles for different systems and help information is formatted in HTML. Implementation for the Microsoft™ Windows™ uses cygwin, a Unix emulator for Windows running on 32-bit Intel architectures [22]. While this does require the user to have an X-window server running and leads to a rather different look to what one commonly finds on a windows machine, it means that any code changes can be seamlessly ported to all environments as none of the source code is machine dependent. Packaging into a user-friendly form for the windows environment was done with the NSIS software [23], which creates a standard self-installing executable, and using PackageManager [24] for the Macintosh environment. Under Windows, the installed package will appear to the user in the conventional Windows "Start" menu. Crossover from C++ code to Fortran code is done by running shell scripts, which (with a little care since even some very simple commands change with different flavors of Unix) are highly portable. This avoids complications with how different operating systems handle C++/C/Fortran interoperability, for which no standard currently seems to exist. To assist in maintenance and user support a listserver (see http://www.numis.northwestern.edu/edm/#list) has been established.

The system dependencies, such as CPU and X-server byte-ordering are handled by the autoconfigure structure. Thus, most compilations and installations will succeed without any user intervention. In a few cases, where specific locations of libraries need to be defined, flags can specifically be set to point to these locations. It should be mentioned that this assumes a fully functional computer operating system; in testing on a number of computers it became clear that quite often there are inconsistencies in some of the system software on many machines.

We can state with 99.99% confidence that the code will compile and run on any Unix system with a gcc/g++/g77 compiler. It will also run on most systems using native compilers, although there are a surprisingly large number of differences between how C++ is implemented which could lead to compiler errors. In some cases, it can also be compiled on Linux systems using the icc/ifc compiler, although this combination appears to be somewhat unstable in some cases.

3. Functionality

The full functionality of the software is rather large, and described in the documentation which is available on-line at http://www.numis.northwestern.edu/edm/documentation/edm.html. The main functionality includes:

1. User friendly mouse-driven interface.

2. Crystallographic operations (e.g. symmetry averaging) on HREM images with good numerical accuracy (ppm in many cases).
3. A number of image processing options, including Wiener-filtering (e.g. [3,25]), masked/windowed Fourier Transforms and a Hanning Window Fourier Transform.
4. Accurate cross-correlation based methods of measuring spot diffraction intensities [26] and user-friendly symmetry averaging with or without Friedel symmetry. (less accurate background subtraction methods are also included).
5. Accurate methods of extracting phases from HREM images [27].
6. Includes an improved version of direct methods code fs98 to solve structures, which can also be used with other types of data (e.g. surface X-ray diffraction) as well as TED intensity data. This uses a feasible set approach [28] with a sharing protected genetic algorithm search and a Kullback–Liebler metric; see [29] and references therein for further details. Analysis of the use of direct methods with dynamical electron diffraction data has been published previously [15,30–33] to which the interested reader is referred.
7. HRTEM Image Simulation. The multislice code NCEMSS [34] (National Center for Electron Microscopy Simulation Software) which is a separate package is installed by default for quantitative analysis of HREM images.

We deliberately decided to have NCEMSS [34] installed by default since for completely rigorous analysis of the data dynamical diffraction effects must always be considered, even though with a fairly thin sample a kinematical approximation (implicit to some extent in direct methods) is still statistically accurate in many cases [31–33].

As much as possible the code has been designed to try and follow typical workflow in terms of analysis of images and diffraction data. For instance, with diffraction data regions of the main window for quantization are only available to the user when the axes of the diffraction pattern are defined. Similarly, the direct methods analysis is only available when diffraction data intensities have either been measured by the code or imported from a text file. The overall flow of the programs is shown in Fig. 1.

4. Future plans

The longer range intent is to move from the original code which was developed at Northwestern University and Lawrence Berkeley National Labs to a wider base with code from a number of different sources all integrated into a common package. One part of this will be to move from code which currently resides at Northwestern University to a cvs distribution so that the maintainer base can be broadened. The intent is also to expand the functionality, for instance to include:

1. Code for simulation of CBED patterns, work currently in progress.
2. More functionality for analysis of diffraction patterns, e.g. indexing and analysis of ring patterns.
3. Include code for structure completion. For technical reasons this is not so simple because automated structure completion for two-dimensional electron diffraction data is not trivial.
4. Enable use of standard libraries such as libtiff (if available) so more image formats can be supported; this is easy to automatically recognize using the autoconfigure structure.
5. Include code for multislice structure refinement in a user-friendly format.
6. Implement code for merging multidatasets to extend the dynamic range of diffraction measurements.

It is hoped that as the code develops it will become a widely used general resource, maintained in part by the general electron microscopy community.

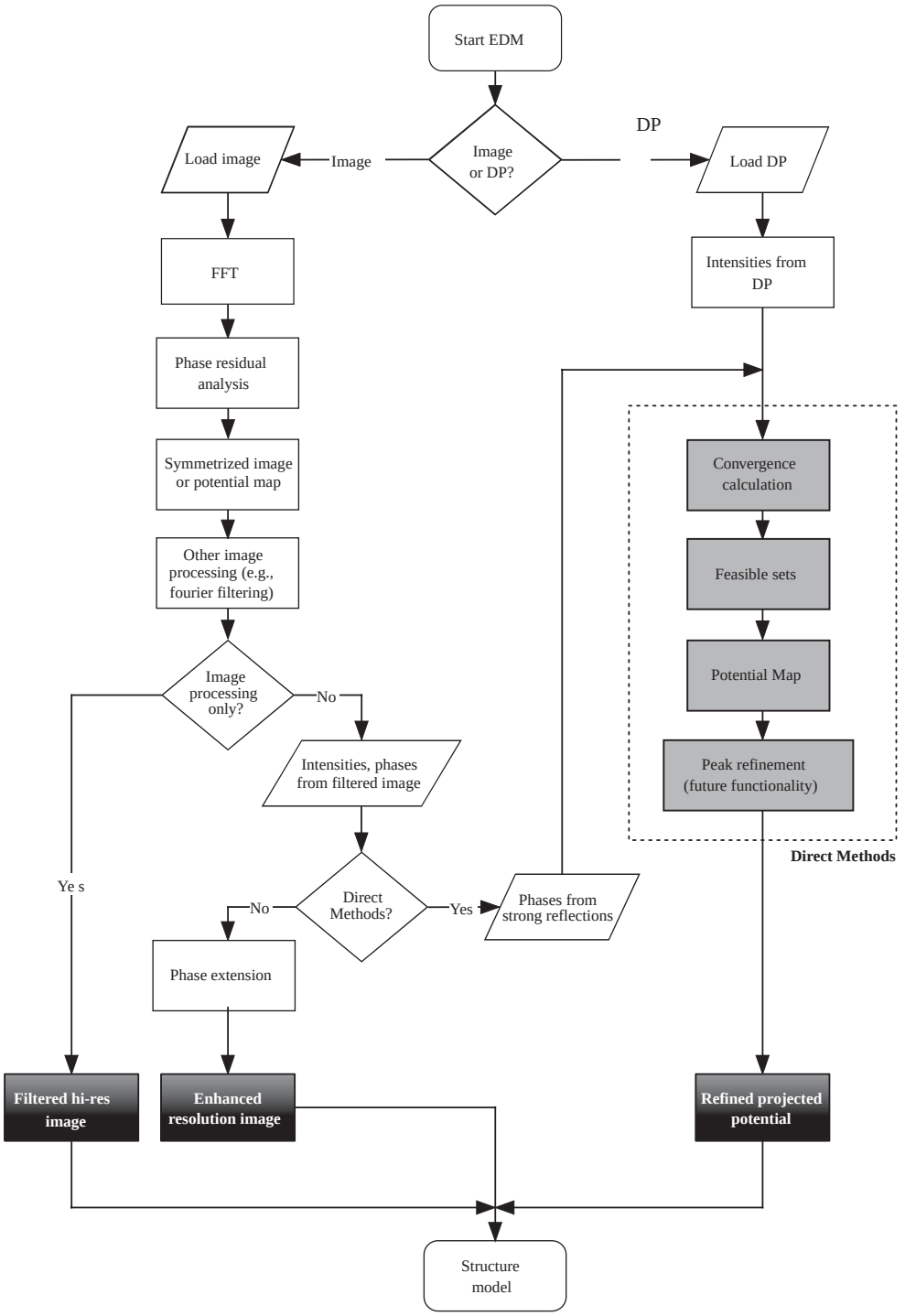

Fig. 1. Overall flow diagram of the current version of EDM (1.0).

Acknowledgements

This development would have been impossible without input from both a large number of students, beta-testers and users. We would like to acknowledge support from the Department of Energy on grant number DE-FG02-01ER45945 (LDM) the Hertz Foundation (CSO) and support by the director, Office of Energy Research, Office of Basic Energy Sciences, Materials Science Division, of the US Department of Energy under contract No. DE-AC03-76SF00098.00098 (RK).

References

[1] W.O. Saxton, Computer Techniques for Image Processing in Electron Microscopy, University of Cambridge, Cambridge, 1975.
[2] W.O. Saxton, T.J. Pitt, M. Horner, Ultramicroscopy 4 (3) (1979) 343–353.
[3] L.D. Marks, Ultramicroscopy 62 (1–2) (1996) 43–52.
[4] W.O. Saxton, J. Struct. Biol. 116 (1) (1996) 230–236.
[5] Z.H. Wan, Y.D. Liu, Z.Q. Fu, Y. Li, T.Z. Cheng, F.H. Li, H.F. Fan, Z. Kristallogr. 218 (4) (2003) 308–315.
[6] W. Dong, T. Baird, J.R. Fryer, C.J. Gilmore, D.D. Macnicol, G. Bricogne, D.J. Smith, M.A. Okeefe, S. Hovmoller, Nature 355 (6361) (1992) 605–609.
[7] S. Hovmoller, X.D. Zou, T.E. Weirich, Adv. Imag. Electron Phys. 123 (2002) 257–289.
[8] F.H. Li, Rare Met. Mater. Eng. 31 (3) (2002) 161–166.
[9] T.E. Weirich, Z. Kristallogr. 218 (4) (2003) 269–278.
[10] M.W. Anderson, T. Ohsuna, Y. Sakamoto, Z. Liu, A. Carlsson, O. Terasaki, Chem. Commun. 8 (2004) 907–916.
[11] D.L. Dorset, Acta Crystallogr. Sec. B. 52 (1996) 753–769.
[12] J. Gjonnes, V. Hansen, B.S. Berg, P. Runde, Y.F. Cheng, K. Gjonnes, D.L. Dorset, C.J. Gilmore, Acta Crystallogr. Sect A 54 (1998) 306–319.
[13] W. Sinkler, L.D. Marks, D.D. Edwards, T.O. Mason, K.R. Poeppelmeier, Z. Hu, J.D. Jorgensen, J. Solid State Chem. 136 (1) (1998) 145–149.
[14] C.J. Gilmore, Microsc. Res. Tech. 46 (2) (1999) 117–129.
[15] W. Sinkler, L.D. Marks, Ultramicroscopy 75 (4) (1999) 251–268.
[16] T.E. Weirich, Acta Crystallogr. Sect. A 57 (2001) 183–191.
[17] N. Erdman, K.R. Poeppelmeier, M. Asta, O. Warschkow, D.E. Ellis, L.D. Marks, Nature 419 (6902) (2002) 55–58.
[18] D.L. Dorset, Z. Kristallogr. 218 (4) (2003) 237–246.
[19] J. Gjonnes, V. Hansen, S.J. Andersen, C.D. Marioara, X.Z. Li, Z. Kristallogr 218 (4) (2003) 293–307.
[20] S. Hovmoller, Ultramicroscopy 41 (1–3) (1992) 121–135.
[21] Free Software Foundation, I., Autoconf. 2004.
[22] Red Hat, I., Cygwin. 2004.
[23] Nullsoft Scriptable Install System, 2004.
[24] Apple Computer, I., Package Manger, 2004.
[25] A. Subramanian, L.D. Marks, Ultramicroscopy 98 (2–4) (2004) 151–157.
[26] P. Xu, G. Jayaram, L.D. Marks, Ultramicroscopy 53 (1) (1994) 15–18.
[27] W.J. Ruijter, J. Comput. Assist. Microsc. 6 (4) (1994) 1–20.
[28] L.D. Marks, W. Sinkler, E. Landree, Acta Crystallogr. Sect. A. 55 (1999) 601–612.
[29] L.D. Marks, N. Erdman, A. Subramanian, J. Phys.-Condensed Matter 13 (47) (2001) 10,677–10,687.
[30] W. Sinkler, L.D. Marks, Mater. Character. 42 (4–5) (1999) 283–295.
[31] J.J. Hu, F.N. Chukhovskii, L.D. Marks, Acta Crystallogr. Sect. A. 56 (2000) 458–469.
[32] F.N. Chukhovskii, J.J. Hu, L.D. Marks, Acta Crystallogr. Sect. A. 57 (2001) 231–239.
[33] L.D. Marks, W. Sinkler, Microsc. Microanal. 9 (5) (2003) 399–410.
[34] R. Kilaas, Interactive software for simulation of high resoluton TEM images, in: Proceedings of 22nd Annual Conference of the Microbeam Analysis Society, 1987.