Acta Crystallographica Section A
Foundations of Crystallography
ISSN 0108-7673

Received 5 July 2007
Accepted 7 September 2007

© 2008 International Union of Crystallography
Printed in Singapore – all rights reserved

A short history of *SHELX*

George M. Sheldrick

Department of Structural Chemistry, University of Goettingen, Tammannstrasse 4, D-37077 Goettingen, Germany. Correspondence e-mail: gsheldr@shelx.uni-ac.gwdg.de

An account is given of the development of the *SHELX* system of computer programs from *SHELX*-76 to the present day. In addition to identifying useful innovations that have come into general use through their implementation in *SHELX*, a critical analysis is presented of the less-successful features, missed opportunities and desirable improvements for future releases of the software. An attempt is made to understand how a program originally designed for photographic intensity data, punched cards and computers over 10000 times slower than an average modern personal computer has managed to survive for so long. *SHELXL* is the most widely used program for small-molecule refinement and *SHELXS* and *SHELXD* are often employed for structure solution despite the availability of objectively superior programs. *SHELXL* also finds a niche for the refinement of macromolecules against high-resolution or twinned data; *SHELXPRO* acts as an interface for macromolecular applications. *SHELXC*, *SHELXD* and *SHELXE* are proving useful for the experimental phasing of macromolecules, especially because they are fast and robust and so are often employed in pipelines for *high-throughput* phasing. This paper could serve as a general literature citation when one or more of the open-source *SHELX* programs (and the Bruker AXS version *SHELXTL*) are employed in the course of a crystal-structure determination.

1. Introduction

It was thought that a general overview of the *SHELX* system of computer programs would be timely, because most of the earlier accounts of *SHELX* are scattered in the excellent series of books describing the IUCr Computing Schools (Sheldrick, 1982, 1985*a*, 1991, 1993, 1996; Robinson & Sheldrick, 1988) that are mostly not available in computer-readable form, and later papers primarily emphasized specific macromolecular applications. Recently, a book on *SHELXL* refinement has been published in the IUCr series of *Monographs on Crystallography* (Müller *et al.*, 2006). However, a general citable reference on *SHELX* was still missing, so this paper, which will be made available as open access, is intended to fill the gap.

The first version of *SHELX* was written around 1970, primarily to replace the author's first rather primitive attempts at crystallographic programs that were written in Titan Autocode (a sort of user-friendly assembler language) with a Fortran program in preparation for the IBM 370/165 that replaced the University of Cambridge ICL Titan computer and started operation there in March 1972. The program was written in a small subset of Fortran IV, which made it relatively easy to port it from Titan to the IBM 370 and later to other computers. This subset showed a curious resemblance to Titan Autocode, which had the useful side effect that, since it was close to a machine language, it compiled to rather efficient code. However, the biggest breakthrough on replacing Titan with the IBM 370 was without doubt the *introduction* of punched cards (Titan could only read paper tapes)! Although they have long since disappeared, these punched cards still have a dominant influence on the *SHELX* input formats. In fact, the programming style of *SHELX* has changed little since that time, though a few particularly useful features of later Fortran compilers (*e.g.* character variables in Fortran 77 and run-time array allocation in Fortran 90) were exploited when they became available.

2. Discussion

2.1. *SHELX*-76

After *SHELX* had been used locally in the Cambridge University Chemical Laboratory for several years and so was well tested and debugged, it became clear that it would be useful to have one official definitive final 'export' version that would not need any further changes; this version was *SHELX*-76. The design criteria were that it should perform the calculations necessary for inorganic, organic and mineral crystal structure solution and refinement, and fit, together with suitable test data, into one box of punched cards to facilitate distribution, *e.g.* by post. Since only 2000 cards fitted into the box, this was a tight constraint.

Despite efficient programming and rather sparing use of comment statements, the code for this – for the time – comprehensive crystallographic system amounted to 5000 Fortran statements. Although it might be possible with a modern computer language such as Python, by making extensive use of C++ libraries and higher-level compiler options, to write such a program in even fewer statements, these statements would be much longer and it should be remembered that *SHELX*-76 only used a small Fortran subset and was entirely self-contained, calling no library routines, even for the least-squares matrix algebra and Fourier syntheses. This *zero-dependency* philosophy also applies to all subsequent *SHELX* programs, which makes them easy to port to different computer systems, and is probably one reason why *SHELX* has survived for so long.

The problem of the number of cards was solved by 'compressing' the Fortran source to use all 80 columns on each card, providing a little Fortran program (on uncompressed cards of course) to uncompress the source again to the full 5000 cards for local use. This compression was made more efficient by only using single (or at the most two) letter names for the Fortran variables and arrays. For the five test structures, the reflection data (but not the instruction files) were compressed to about nine reflections per card; this 'condensed data' format involved rounding insignificant digits but, by taking advantage of the known characteristics of a sorted reflections list, a compression ratio was achieved that was almost as high as *bzip2* would now achieve with the same data. No decompression program was required because *SHELX*-76 (and other later programs) could read the condensed data. So everything fitted into the card box. Condensed data proved very useful as long as reading punched cards was the normal form of job input, and enjoyed a brief renaissance in the early days of BITNET (the predecessor of the Internet).

One major innovation in *SHELX*-76 was the use of a free-format instruction file containing the instructions, crystal data and possibly the current atom coordinates. Since this predated the introduction of free-format input in Fortran, the instructions were interpreted character by character using Fortran (although character variables and functions had also not yet been introduced into the Fortran language, it was possible to use Hollerith strings for this purpose). This, coupled with the extensive use of sensible default values, made the input card deck easy to read and modify; very often only a couple of cards would need to be replaced to run the next job. At about the same time, free-format input was independently implemented in Fortran in the molecular graphics program *PLUTO* (Motherwell, 1978).

A further innovation that now seems self-evident was that all calculations were valid for all space groups, in conventional settings or otherwise. For this reason, the symmetry information was communicated to the program by means of the coordinates of the general position, not by the name or number of the space group.

At the time *SHELX*-76 was introduced, most X-ray intensities were still estimated by eye from diffraction patterns recorded on photographic film, so *SHELX*-76 included facilities for processing Weissenberg camera data, for example elimination of systematic absences, Lp and face-indexed absorption corrections, scaling different films, Weissenberg layers and crystals together by the method of Rae & Blake (1966), and sorting and merging the resulting data. To solve the structure, one could either calculate the Patterson and interpret the peak list or use one of the two direct-methods routines provided. The first of these, the EEES instruction for centrosymmetric structures, represented phases as 0 or 1 to save computer time and memory. It was very efficient for straightforward small structures. The second, the TANG instruction for non-centrosymmetric structures, required an experienced user to select the origin and enantiomorph-fixing reflections by hand; it was inspired by *MULTAN* (Germain *et al.*, 1970), which was the direct-methods program of choice at the time.

The least-squares refinement in *SHELX*-76 included a number of innovations that have stood the test of time. The use of *free variables* enabled a number of problems to be handled routinely that would previously have required the user to write some complicated Fortran code, *e.g.* the application of constraints to the coordinates and anisotropic displacement parameters of atoms on special positions or the coupling of occupancies of atoms in disordered groups. Some widely used macromolecular refinement programs still lack such facilities! The simple way in which rigid groups such as phenyl rings could be set up and refined and the automatic generation and riding model for H atoms were also most appreciated by users. Distance restraints, suggested by Waser (1963), also proved very useful. The least-squares algebra of *SHELX*-76 and in particular the treatment of complex scattering factors within the least-squares refinement framework was based closely on algorithms developed by Durward Cruickshank, who kindly provided the author with his notes in advance of publication (Cruickshank, 1970). In addition to blocked refinement, there was also an option to sum and invert the least-squares matrix in the first cycle only; in subsequent cycles, the stored inverse matrix was simply rescaled, saving appreciable computer time (in those days one cycle of full-matrix least-squares refinement of a small molecule could take several hours).

Since most computer systems in the 1970's were mainframes with little or no facilities for graphical display, molecular diagrams and pictures of crystals (for the absorption corrections) were printed by *SHELX*-76 on line-printer paper, leaving the user to join up the numbers by hand; these outputs often found their way to the local kindergarten. Similarly, Fourier maps were output in numerical form for hand-contouring on perspex sheets.

2.2. Minicomputer implementations of *SHELX*

Soon after the release of *SHELX*-76, it was ported to the Data General Nova and Eclipse minicomputers in collaboration with Syntex Analytical Instruments (now Bruker AXS), making it available for in-house use in the form of the *SHELXTL* system, which later migrated to the Vax computers

and then to PCs. Relative to a modern desktop PC with say four CPUs and 4 Gbyte memory (the computer currently used by the author for *SHELX* development), the Nova was about 200000 times slower. The program and operating system had to fit into 32 K 16-bit words, a factor of about 62500! Despite this, as a result of many hours spent hand-optimizing the code, it was entirely possible to solve and refine small-molecule structures in a few hours or days on such a machine. Although 'overlay' techniques meant that only the code immediately needed was stored in memory, with the rest on the disk, the memory allocation was critical for least-squares structure refinement. To avoid having to store a large least-squares normal matrix on the disk, a *blocked cascade algorithm* was developed. In this algorithm, the structure was divided into small overlapping full-matrix blocks, each corresponding to a few atoms and some global parameters. The structure factors were only recalculated for atoms that changed in the current or previous block, and the blocks were chosen differently on each pass through the atom list so that parameters that were correlated with each other were refined more often in the same block. This algorithm minimized the amount of disk access and made very efficient use of the limited computing resources. In fact, the final version of this program still left one 16-bit word of memory not used. Fortunately, in those days there was no danger of the operating system updating itself automatically *via* the Internet and thus requiring more memory! The *SHELXTL* system of Bruker AXS (Madison, WI 53711, USA) currently runs under Windows, Linux and MacOSX and includes proprietary software for space-group determination and data processing (*XPREP*) and molecular graphics (*XP*) in addition to the programs corresponding to the open-source version of *SHELX*.

2.3. Direct and Patterson methods for small molecules: *SHELXS* and *SHELXD*

Direct methods for solving small-molecule structures made considerable progress in the two decades that followed the introduction of *SHELX*-76, so it was necessary to replace the rather primitive direct methods in *SHELX*-76 by the more sophisticated procedures in *SHELXS*-86, -90, -93 and -97 (Sheldrick, 1985*a*,*b*, 1990, 1993; Robinson & Sheldrick, 1988). These programs made extensive use of the negative quartets that had been discovered by Schenk (1974) to eliminate false solutions. *SHELXS*-90 introduced a novel simulated-annealing approach that proved particularly efficient in the solution of small-molecule structures with up to about 100 unique non-H atoms. Although several other powerful and user-friendly direct-methods programs are now available, for example *SnB* (Miller *et al.*, 1993, 1994), *SIR* (Burla *et al.*, 2005) and *SHELXD* (Usón & Sheldrick, 1999), that can solve much larger structures and also obtain more complete solutions, *SHELXS* is still widely used.

SHELXS-86 also introduced a method to solve a heavy-atom Patterson automatically that was basically a computer implementation of a classical method of hand interpretation (Sheldrick, 1985*b*). The Patterson was usually sharpened in *SHELXS*-86 and *SHELXS*-97 by replacing the coefficients F^2 by $\sqrt(E^3F)$, where E is a normalized structure factor; *SHELX*-76 used coefficients EF for a sharpened Patterson. In *SHELXS*-86, possible choices for the first heavy atom were sites for which all vectors to symmetry equivalents were present in the Patterson peak list; the remaining heavy atoms were found by testing the Patterson peaks as potential cross vectors involving the starting atom. In *SHELXS*-90, this was replaced by a more powerful algorithm that made extensive use of the Patterson superposition minimum function (Buerger, 1959; Richardson & Jacobson, 1987); a detailed description of the algorithm was given by Sheldrick (1991, 1997). To save computer memory, the Patterson superposition minimum function was calculated and peak-searched on the fly. This program was widely used for finding the heavy-atom sites for SAD and SIR phasing of protein structures and is still useful for tackling problem inorganic and mineral structures today. An important feature of the output of both the *SHELXS*-86 and *SHELXS*-97 Patterson interpretation algorithms was a *crossword table* (Sheldrick, 1991; Schneider & Sheldrick, 2002) that tabulated, in the form of a triangular matrix, the minimum Patterson density at any vector connecting two atoms and also the minimum distance between them, in both cases taking all symmetry equivalents into account. This considerably simplified the interpretation, especially in high-symmetry space groups.

SHELXS-86 and later versions provided an iterative E-map recycling facility (Sheldrick, 1982), shown in Fig. 1, to expand from incomplete direct methods or Patterson solutions to the full structure. After this approach had been tested successfully on the known small-protein structures rubredoxin, finding the FeS_4 unit, and crambin, finding the six S atoms by automatic Patterson interpretation, and then in both cases expanding to the full structure (Sheldrick *et al.*, 1993), it was applied to solve

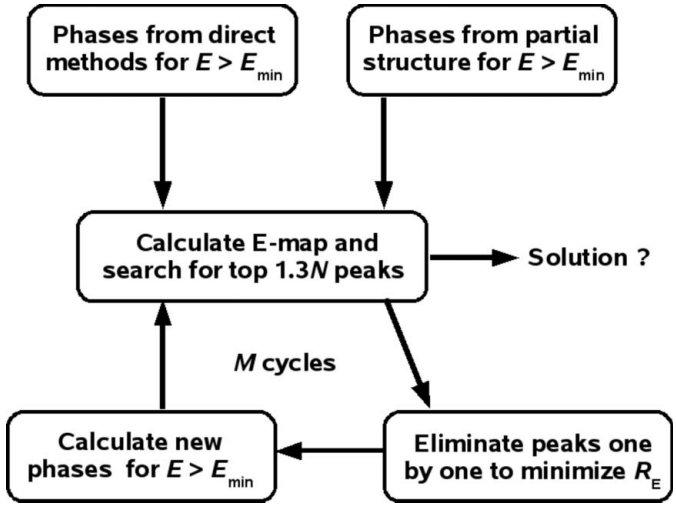

Figure 1
E-Fourier recycling as used in *SHELXS*-86 and later versions to improve phases from direct methods or partial structures. E are normalized structure factors, N is the expected number of atoms, M is the number of iterations and $R_E = \sum |E_o - E_c| / \sum E_c$.

Table 1
Large unknown native structures solved using *SHELXD*.

N is the number of unique non-*N* atoms.

Compound	Space group	*N* (no solvent)	*N* (incl. solvent)	Heavy atoms	Resolution (Å)
Hirustasin	$P4_32_12$	402	467	10 S	1.20 and 1.40
Cyclodextrin	$P2_1$	448	467	None	0.88
Decaplanin	$P2_1$	448	635	4 Cl	1.00
Cyclodextrin	$P1$	483	562	None	1.00
Buchandin	$C2$	516	634	10 S	1.05
Amylose-CA26	$P1$	624	771	None	1.10
Viscotoxin B2	$P2_12_12_1$	722	818	12 S	1.05
Mersacidin	$P3_2$	750	826	24 S	1.04
PNA	$P2_1$	750	900	None	1.05
Feglymycin	$P6_5$	828	1026	None	1.10
Tsuchimycin	$P1$	1069	1283	24 Ca	1.00
HiPIP Cv	$P2_12_12_1$	1264	1599	8 Fe	1.20
Cytochrome c_3	$P3_1$	2024	2208	8 Fe	1.20

the unknown structure of the protein cytochrome c_6 *via* one Fe and three S atoms (Frazão *et al.*, 1995). Because the *E*-Fourier recycling involved stepwise elimination of atoms to optimize the correlation coefficient between the observed and calculated *E* values, it was 'bistable' and could be used to turn a centrosymmetric double image (arising *e.g.* from a single heavy atom in the space group $P2_1$) into a complete but single image of the structure. It would have been an obvious extension to include the direct-methods phase refinement into this recycling, but this innovation, which substantially increased the size of structures that could be solved by direct methods, had to wait until the Buffalo group introduced *dual-space* recycling seven years later in 1993!

The direct-methods program *SHELXD* (Sheldrick, 1998; Usón & Sheldrick, 1999; Sheldrick *et al.*, 2001) was based on the dual-space (*Shake and Bake*) strategy introduced in the program *SnB* (Miller *et al.*, 1993, 1994; Sheldrick *et al.*, 2001); for this reason, the first beta-test version of *SHELXD* was referred to as *half-baked*. A major difference between *SHELXD* and *SnB* was the extensive use made of the Patterson function in *SHELXD*, both to provide better than random starting phases for the dual-space recycling and as a figure of merit. The first version (Sheldrick & Gould, 1995), which was never officially released, expanded the data to space group *P*1. Then, from the three atoms of a vector triangle located in the sharpened Patterson, the rest of the structure was found as in Fig. 1, but optimizing the correlation coefficient between E_o and E_c instead of the corresponding *R* value. For structures containing heavy atoms (even S or Cl in some cases), many starting triangles led to structure solution, but it was necessary to find the origin shifts so that the solutions were consistent with the symmetry of the space group.

The distributed version of *SHELXD* optionally uses the *probabilistic sampling* algorithm to obtain a large number of sets of possible starting atoms that are consistent with the (sharpened) Patterson function (Sheldrick *et al.*, 2001). Each unique general Patterson vector of suitable length is a potential heavy-atom to heavy-atom vector, and may be employed as a two-atom search fragment in a translational search based on the Patterson minimum function for all the independent vectors involving the two atoms and their symmetry equivalents. Alternatively, a vector of known length – *e.g.* an S—S bond (2.03 Å) – but random orientation can be used. For each position of the two atoms in the cell, the Patterson height P_j is found for all vectors between them and their equivalents, and the sum (PSUM) of the lowest (say) 35% of P_j calculated. It would be easy to find the global maximum of PSUM using a fine three-dimensional grid, but in borderline cases this sometimes does not lead to the solution of the structure! An effective approach is to generate many different starting positions by simply taking the best of a finite number of random trials each time. The full-symmetry Patterson superposition minimum function is used to expand from the two atoms to a much larger number before entering the dual-space recycling. The resulting phases are refined by the dual-space recycling, using a tangent expansion from the best determined phases in the reciprocal-space stage and picking the strongest *N* independent peaks that are sufficiently far apart from each other in the real-space stage.

A further *SHELXD* innovation in the dual space applies to the real-space part of the cycle. Instead of taking the top *N* unique peaks, 1.3*N* can be chosen and 30% of them left out at random. It was discovered by accident that this can improve the success rate by about an order of magnitude; by analogy with the use of omit maps in protein crystallography, it is called the *random omit algorithm* (Sheldrick *et al.*, 2001). In fact, given enough iterations, this method is able to solve large structures even when no phase refinement is performed in the reciprocal-space part of the cycle! Although both the Patterson probabilistic sampling and the random omit algorithm can each improve the chances of success by about an order of magnitude, combining them is unfortunately not much more effective than either alone.

Table 1 shows some of the largest structures that have been solved by *ab initio* direct methods using *SHELXD*. Similar tables can be made for the programs *SnB* (Miller *et al.*, 1993, 1994) and *SIR* (Burla *et al.*, 2005) that are at least as effective. The strongest restraint is still the requirement of atomic resolution (*ca* 1.2 Å). All three programs are able to solve much larger structures if heavier atoms (even S or Cl) are present (especially when *SHELXD* or *SIR* make use of the Patterson), and for such structures the resolution requirement is much less rigid. For example, the largest unknown equal-atom (*i.e.* no atom heavier than O) structure solved using *SHELXD* so far is probably feglymycin (Bunkóczi *et al.*, 2005) with 1026 unique non-H atoms. Hirustasin, with 10 S atoms amongst the 467 unique atoms, could be solved using both the 1.2 Å low-temperature data and the 1.4 Å room-temperature data (Usón, Sheldrick *et al.*, 1999).

2.4. Experimental phasing for macromolecules: *SHELXC*, *SHELXD* and *SHELXE*

The location of heavy-atom sites from SAD, SIR, SIRAS, MAD or RIP data from macromolecules follows the scheme shown in Fig. 2, but without the part in the shaded box.

Instead, the dual-space recycling is followed by two cycles of conjugate-gradient least-squares refinement of the occupancies of the heavy atoms. The expected number of heavy atoms *N* is treated only as a guide, for the best results it should be within 20% of the true value. Heavy-atom derivatives prepared by soaking the crystals in a solution containing a heavy-atom salt naturally exhibit a full range of occupancies; in other cases, the occupancy refinement is able to compensate, at least in part, for different elements present, for different displacement parameters (*B* values) and for disorder. Fig. 3 shows typical occupancy distributions, plotted using the *hkl2map* graphical user interface (GUI) (Pape & Schneider, 2004). The largest substructure solved so far with *SHELXD* is probably PDB code 2pnk, solved by Qingping Xu of the Joint Center for Structural Genomics (JCSG), with 197 correct and no incorrect sites out of 205. About 1.6 million *SHELXD* trials were needed to obtain one correct solution when Patterson seeding was employed but, with the Patterson seeding switched off, many good solutions were obtained.

There is clearly more scope for improving the chances of success and quality of heavy-atom location in the case of weak anomalous signals, *e.g.* in sulfur SAD experiments (Debreczeni, Bunkóczi, Girmann & Sheldrick, 2003; Debreczeni, Bunkóczi, Ma *et al.*, 2003). This is shown by the success of the option available in *SHELXD* to search for disulfide units directly in the peak-search routine (Debreczeni, Girmann *et al.*, 2003). In addition to searching for other heavy-atom clusters, an efficient algorithm for recognizing and exploiting non-crystallographic symmetry inside the dual-space iteration would be particularly useful. The most critical parameter affecting the success of *SHELXD* with SAD or MAD data from poorly diffracting crystals is the resolution to which the data should be truncated. The correlation coefficient between the signed anomalous differences between different crystals or wavelengths gives a good estimate (Schneider & Sheldrick, 2002), but it may be best simply to run *SHELXD* several times with different cut-offs. If a multiple CPU system is available, this parameter can be varied in parallel *SHELXD* runs.

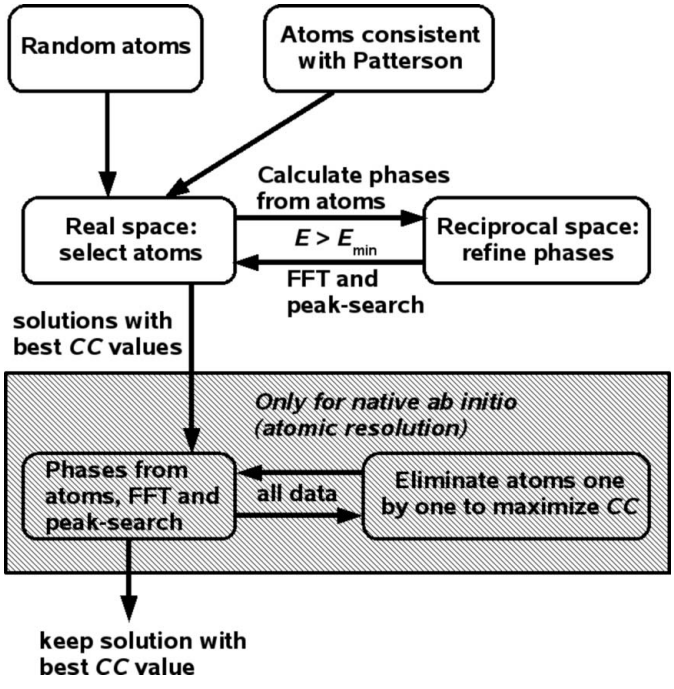

Figure 2
Dual-space recycling as used in *SHELXD* for integrated Patterson and direct-methods phasing. The *E*-Fourier recycling to improve the model depicted in the shaded box is only performed for *ab initio* location of all atoms at atomic resolution, the remaining operations are the same for this and for the location of heavy atoms from SAD, SIR, MAD *etc.* data for experimental phasing of macromolecules. *CC* is the correlation coefficient between E_o and E_c.

The program *SHELXE* (Sheldrick, 2002) was designed to provide a quick and robust method for experimental phasing of macromolecules using heavy atoms found by *SHELXD*. The heavy-atom phases are used to obtain starting native phases that are improved by density modification. This is based on the *sphere of influence algorithm*, which exploits the fact that 1,3 distances in macromolecules are often close to 2.42 Å. A sphere of radius 2.42 Å is constructed around each pixel in the map. If the density in this spherical surface has a high variance, *i.e.* probably contains a few atoms, the pixel at the centre of the sphere is also a potential atomic position. The density at such potential atomic positions is truncated if it is negative and optionally sharpened; for other pixels, the density is inverted (*i.e.* its sign is changed). With some precautions to prevent model bias building up, this provides a simple and effective method of improving the phases; as with all density-modification methods, it works best if the solvent content is high. The variance of this variance (referred to as *contrast* in the output of the program) is a good indication of whether the map really looks like a macromolecule, *i.e.* has connected regions of high fluctuation (protein *etc.*) and also connected regions of low fluctuation (solvent); it may be employed to decide whether it is necessary to invert the heavy-atom enantiomorph or not.

If the resolution of the native data is about 2.0 Å or better, it appears to be possible to improve the density and phases by extrapolating to a higher resolution than was actually measured. This idea apears to have been first successful implemented in the program *ACORN* in about 2001. After its incorporation in *ACORN* (Yao *et al.*, 2005) and independently in *SIR200X* (Caliandro *et al.*, 2005), it was also successfully added to *SHELXE*, where for obvious reasons it is called the *free lunch algorithm*. In this algorithm, the phases of the reflections that had not been measured are calculated by density modification using the sphere-of-influence algorithm (Sheldrick, 2002). The extrapolated amplitudes, including those of missing low-order reflections, are obtained by Fourier transformation of the density-modified map and are then normalized to fit an extrapolated Wilson plot (Usón *et al.*, 2007). Phase improvements of up to 30° have been observed, although usually the improvement is more modest (*e.g.* 5°).

The program *SHELXC* is designed to prepare the three files necessary for running *SHELXD* and *SHELXE*. These contain filtered anomalous or isomorphous differences, or estimated heavy-atom structure factors $|F_A|$ from SIRAS or MAD experiments, and the phase shifts that should be added

to the heavy-atom reference phases to obtain starting values for the native phases. In addition, *SHELXC* produces useful statistics, *e.g.* to indicate at which resolution the data should be truncated for heavy-atom location. The three programs *SHELXC*/*D*/*E* may be called from the command line or from a script, but most users prefer to call them *via* a GUI such as *hkl2map* (Pape & Schneider, 2004); this has the advantage of nice graphical displays such as those shown in Fig. 3. *SHELXC* and *SHELXE* also contain facilities for radiation-damage-induced phasing (RIP) (Nanao *et al.*, 2005) and for calculating anomalous Fourier maps for element identification (Cuesta-Seijo *et al.*, 2006).

2.5. Small-molecule refinement with *SHELXL*

For many years, the large majority of small-molecule and inorganic structures that have been determined from single-crystal X-ray data have been refined using *SHELX*-76 or *SHELXL*-97. By the late 1980's, the *SHELX*-76 restriction to 160 atoms (including H atoms) for least-squares refinement, later extended to 400 with the help of Dobi Rabinovitch, proved to be too restrictive. It was necessary to produce a new version, which after about 10 years of careful in-house alpha-testing and debugging was released as a beta-test *SHELXL*-93 and then as a final version *SHELXL*-97. This fundamental rewrite provided the opportunity to change from refinement against F to refinement against F^2 and make many other changes, including CIF output that played a central role in the automated validation and publication of small-molecule structures in IUCr journals.

SHELXL-97 requires only two input files, a .hkl reflection file with one reflection per line and a .ins instruction file that contains the crystal data, current atom coordinates and instructions. The basic format of the .hkl file has not changed for over 30 years, but with hindsight it might have been better if it had included the unit cell and wavelength as the first line. It was however extended to include the free-R flag (Brunger, 1992) and data from non-merohedral twins. The .ins file has also remained more or less upwards compatible despite the addition of many new features in *SHELXL*-97. This free-format file is very compact and makes extensive use of sensible default values, and is designed to be easily understood and updated by the user; it is essentially the *user interface* to *SHELXL*. Each refinement run writes a .res results file in the same format as the .ins file; this may be edited if required and renamed as the .ins file for the next refinement. *SHELXL* is started by typing 'shelxl name' at the command line, which automatically generates the names of the name.ins and name.hkl input files and the output files name.res, name.lst (listing) and, if required, name.cif (CIF output for structure

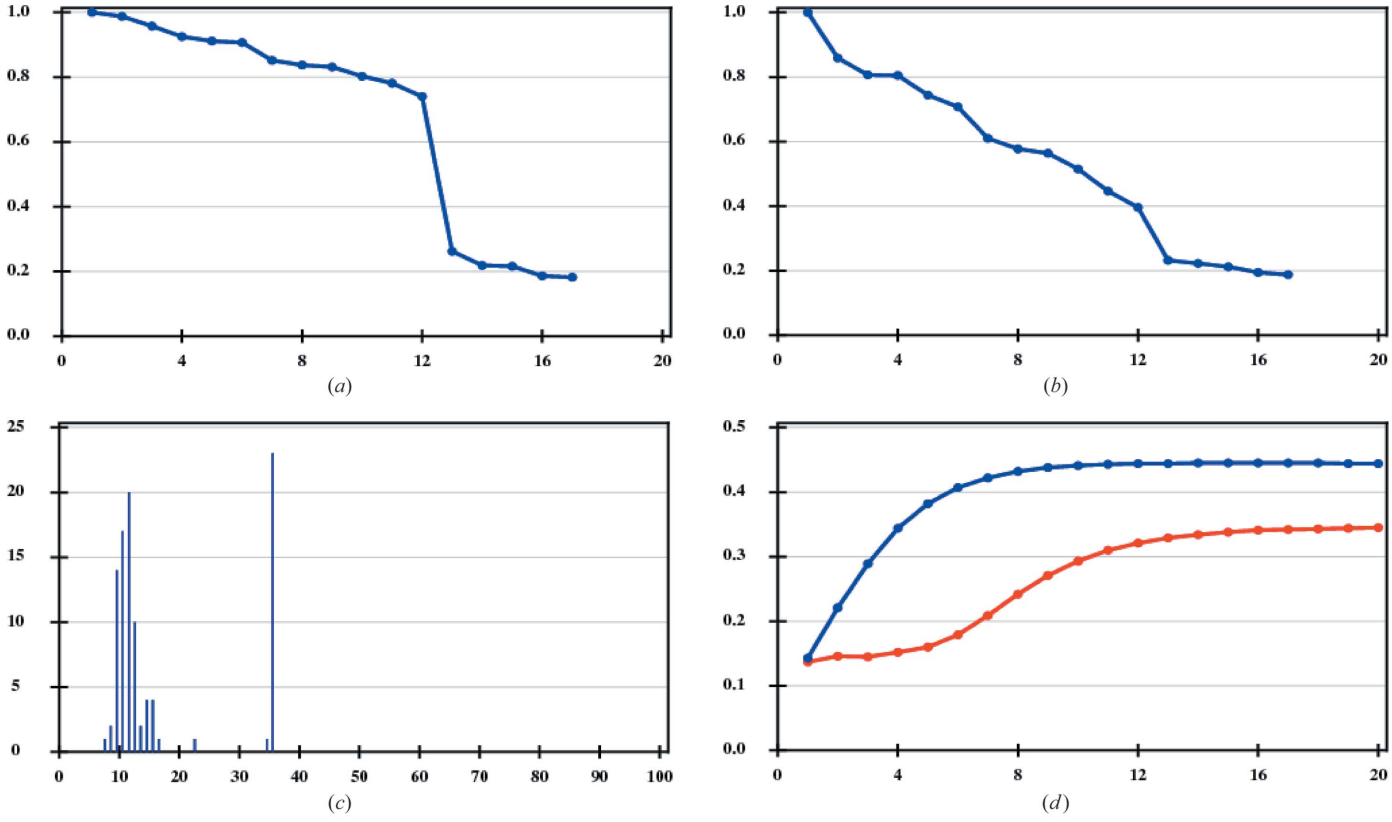

Figure 3
(*a*) Refined occupancy against peak number for *SHELXD* location of the 12 S atoms from SAD data for elastase. (*b*) The same for SIRAS data from an iodide soak of elastase; the peaks with occupancies less than 0.2 are probably noise. (*c*) *CC* histogram for the sulfur-SAD phasing showing 23 correct solutions with *CC* = 36% out of 100 trials. (*d*) Variation of the contrast (see §2.3) during *SHELXE* density modification starting from substructure (*a*); the inverted heavy-atom substructure (blue points) is correct. The diagrams were drawn with *hkl2map* (Pape & Schneider, 2004).

deposition), name.fcf [a CIF format file containing observed and calculated structure factors, suitable for direct input into a graphics program such as *Coot* (Emsley & Cowtan, 2004)] and name.pdb (PDB output).

SHELXL performs full-matrix, blocked full-matrix or conjugate-gradient least-squares refinement using a conventional structure-factor summation (much slower but more precise than a FFT) with complex scattering factors. This possesses the flexibility necessary for handling merohedral and non-merohedral twins and enables a racemic twinning parameter to be refined to establish the correct absolute configuration (Flack, 1983). *SHELX*-76 and *SHELXL*-97 have also been used for neutron data, though the riding H-atom model employed in *SHELXL* is less appropriate for neutron data. To handle Laue data with significant anomalous scattering, a different wavelength may be assigned to each reflection. The standard uncertainties are calculated using the full correlation matrix for all derived geometric parameters such as bond lengths, angles, torsion angles, least-squares planes *etc.* Even if the structure is refined to convergence using the conjugate-gradient solution of the least-squares normal equations (Konnert, 1976), which is faster and more robust for large structures, a final full-matrix cycle is required to obtain these standard uncertainties. Cowtan & Ten Eyck (2000) analysed the conditioning and parameter correlations by finding the eigenvectors of the least-squares matrix from *SHELXL*. *SHELXL* has been adapted by Diederichs (2000) to run in multi-threaded mode on multiple CPU systems with a high degree of parallelization.

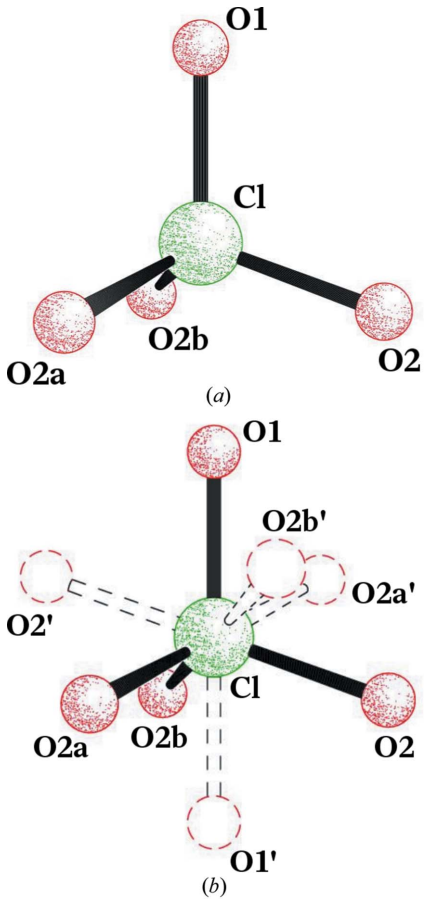

Figure 4
(*a*) A (rare) undisordered perchlorate anion on a threefold axis. Atoms O1 and Cl lie on this axis, O2*a* and O2*b* are symmetry equivalents of O2. (*b*) A (more common) disordered perchlorate anion. It still lies on a threefold axis, but there is a second orientation with atoms O1′, O2′ *etc.*

SHELXL enables extensive use to be made of *constraints* (*e.g.* rigid groups and common occupancies), which leads to a reduction in the number of least-squares parameters, and *restraints* (*e.g.* bond lengths, planarity), which are treated as additional observational equations. Coordinate and displacement parameter constraints are generated automatically for atoms on special positions, and floating-origin restraints are applied automatically by the method of Flack & Schwarzenbach (1988). The thermal displacement parameters can be restrained using the rigid-bond restraint (Rollett, 1970) as well as similarity restraints and approximately isotropic restraints. Detailed examples of the application of restraints may be found in Van der Maelen & Sheldrick (1996) and of course in Peter Müller's book (Müller *et al.*, 2006); the refinement of the twinned structure of mersacidin (Schneider *et al.*, 2000) made extensive use of similar distance restraints, taking advantage of the six independent molecules in the asymmetric unit.

In view of space constraints (restraints?), we will only be able to illustrate the handling of disorder using constraints and restraints here with two common examples. They make use of *free variables*, a cryptic but powerful way of defining constraints and restraints introduced in *SHELX*-76. Atom and various other parameters q are specified as a single number that is interpreted as $10m + p$, where m is an integer and $-5 < p < 5$. If m is zero, then q is refined normally starting at the value p. If m is 1, q is fixed at the value p (so an occupancy given as 11 is fixed at 1.0). If $m > 1$, then q is p times $f(m)$, where $f(m)$ is free variable number m. Finally, if $m < -1$, q is p times $[f(-m) - 1]$. For example, if a disordered solvent molecule has two orientations occupying the same site, the occupancies of one component can be given as 31 [*i.e.* $f(3)$] and the occupancies of the other as -31 {*i.e.* $(-1)[f(3) - 1] = 1 - f(3)$}, where $f(3)$ (free variable number 3) refines starting at a value specified on the FVAR instruction. This *constraint* requires only one least-squares parameter and ensures that the occupancies add up to unity; for three components, it is more convenient to use three free variables and to *restrain* their sum to unity.

For H-atom and restraint generation, *SHELXL* sets up a *connectivity array* automatically; the user may fine tune it if required. A shell of symmetry-equivalent atoms that are bonded to the unique atoms is included in the connectivity array. In the case of disorder, PART instructions may need to be specified so that only genuine bonds are included. Bonds are generated (if the atoms are close enough) between atoms in PART 0 (usually most of the atoms in the structure) and atoms with any PART number, and between two atoms with the same PART number, but not between atoms with different non-zero PART numbers. If a PART number is negative, no bonds are made to symmetry equivalents of the unique atoms.

Fig. 4(*a*) shows a perchlorate anion in which atoms Cl and O1 lie on a threefold axis; O2*a* and O2*b* are symmetry equivalents of O2. *SHELXL* will automatically generate the constraints imposed by the threefold axis on the coordinates and anisotropic displacement parameters of O1 and Cl. The occupancies (as understood by the *SHELX* programs) of these two atoms are fixed at 1/3, *i.e.* given as 10.33333. It would probably be possible to refine this anion without extra restraints but, if we wish to restrain it to be a regular tetrahedron, there are two good ways. The first is to use similar distance restraints to restrain the two Cl—O and two O···O distances to be respectively equal:

```
SADI Cl O1 Cl O2
EQIV $1 -y, x-y, z
SADI O1 O2 O2 O2_$1
```

which requires defining a symmetry equivalent of O2 (atom O2*a* in Fig. 4*a*). The remaining Cl—O and O···O distances become equal by symmetry, producing a regular tetrahedron. Note that the .ins input file is in free format and not case sensitive. The second method involves the use of free variable number 2, which should be set to a suitable starting value for a Cl—O distance (*e.g.* 1.5) on the FVAR instruction. We take advantage of the fact that the O···O distance in a regular tetrahedron is 1.6330 times the Cl—O distance:

```
DFIX 21 Cl O1 Cl O2
DFIX 21.6330 O1 O2
```

and again symmetry will ensure a regular tetrahedron. In Fig. 4(*b*), the perchlorate is disordered over two sites, both lying on the threefold axis. We can easily extend the geometrical restraints so that both components are restrained to be regular tetrahedra with the same dimensions, *e.g.*

```
DFIX 21 Cl O1 Cl O2 Cl O1' Cl O2'
DFIX 21.6330 O1 O2 O1' O2'
```

We introduce PART instructions for the two components and free variable number 3 for the fraction of the first component (the starting value of 0.7 is given on the FVAR instruction), giving the codes shown below for the occupancies:

```
Cl ... ... ... ... 10.33333 ...
PART 1
O1 ... ... ... ... 30.33333 ...
O2 ... ... ... ... 31 ...
PART 2
O1' ... ... ... ... -30.33333 ...
O2' ... ... ... ... -31 ...
PART 0
```

The extension to allow two different Cl-atom sites is straightforward. If these sites are very close to each other, it would be necessary to constrain (EADP) or restrain (SIMU) their isotropic or anisotropic displacement parameters to be equal.

The second disorder example (Fig. 5) has a toluene molecule on an inversion centre, a common misadventure. We need to set all seven occupancies to 10.5 (*i.e.* fixed at 0.5) and put a PART −1 instruction before the first atom and PART 0 after the last. We could use SADI for chemically equivalent distances and restrain the seven atoms to lie in a plane:

```
SADI C7 C2 C7 C6
SADI C1 C2 C2 C3 C3 C4 C4 C5 C5 C6 C6 C1
SADI C1 C4 C2 C5 C3 C6
FLAT C1 > C7
```

or alternatively make C1 to C6 a rigid regular hexagon with variable C—C bond length by putting AFIX 69 before C1 and AFIX 0 after C6, and enforce planarity of C1 by restraining its chiral volume to zero (CHIV C1); the first SADI instruction is still required. This has the advantage that the hexagon can be fitted to any three of its six atoms, so starting coordinates are not required for all six; the coordinates of the other atoms should be set to zero for this purpose. Because of the strong overlap, restraints are needed for the anisotropic displacement parameters, and it is best to allow the program to add the H atoms so that the correct occupancies are generated:

```
SIMU C1 > C7
DELU C1 > C7
HFIX 43 C2 > C6
HFIX 123 C7
```

The last HFIX instruction generates two methyl groups for C7 rotated by 60° from each other, appropriate for a methyl attached to an sp^2-hybridized C atom; these six H atoms would each be assigned occupancies of 10.25 (fixed at 0.25) automatically in this example.

In the refinement of twinned structures, reflections from several twin components may contribute to one observed intensity. In the case of merohedral or pseudo-merohedral twinning, the other contributing reflections may be generated from the first by application of a transformation matrix, possibly more than once. For non-merohedral twins and other difficult cases, it is necessary to prepare a special 'HKLF 5' format .hkl file. The twin factors are refined subject to the constraint that they add up to unity. Detailed descriptions of twinned refinement strategy have been given by Herbst-Irmer & Sheldrick (1998, 2002) and Schneider *et al.* (2000). In general, merohedral and pseudo-merohedral twinning reduces the information content of the diffraction data, so it is usually necessary to apply geometric and displacement parameter restraints to ensure that a chemically sensible model is obtained. This is less of a problem for non-merohedral twins

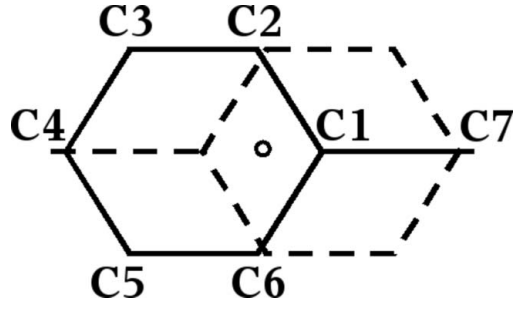

Figure 5
A typical disordered toluene molecule on an inversion centre.

when the reflection overlap is not too severe, such twins actually have the advantage that more reflections can be measured in the same time, so the resulting structures may be as good as could be obtained from untwinned crystals.

The program *CIFTAB* was distributed as part of *SHELX*-97 in order to merge CIF files and prepare tables (*e.g.* in rich text format, RTF) for padding out publications and PhD theses, and because at the time few programs were available that could read CIF files. These days such padding is less desirable and there are excellent programs such as *enCIFer* (Allen *et al.*, 2004) for working with CIF files, so *CIFTAB* is now effectively redundant. It is however essential to archive the final .res file from a *SHELXL* refinement as well as the .cif and .fcf files, because CIF format is still some way from being able to recreate a *SHELXL* refinement job including all the necessary restraints and constraints.

2.6. Macromolecular refinement with *SHELXL*

Although *SHELXL* was originally intended for refining inorganic and small-molecule structures, it has proved useful for the refinement of macromolecular structures against high-resolution data (better than 2 Å), because it provides a number of useful facilities such as the least-squares estimation of individual standard uncertainties (Cruickshank, 1999; Parisini *et al.*, 1999), flexible treatment of disorder including the constrained refinement of occupancies, automatic constraints for atoms on special positions, inclusion of anomalous scattering and refinement against twinned data that are not always available or so convenient to use in programs designed purely for macromolecular refinement. The application of *SHELXL* to macromolecular refinement has been reviewed in detail by Sheldrick & Schneider (1997).

Although some facilities such as conjugate-gradient refinement (Konnert, 1976), the free *R* factor (Brunger, 1992) and non-crystallographic symmetry (NCS) restraints (Usón, Pohl *et al.*, 1999) were added specially for macromolecular refinement, there are also features missing that would be desirable for macromolecules, such as a way of defining peptide and DNA chains, FFT-based structure-factor calculation, maximum-likelihood refinement, torsion-angle restraints, **TLS** restraints or constraints for anisotropic refinement and a more sophisticated solvent model. Torsion-angle restraints were deliberately left out so that backbone and side-chain torsion angles could be used for verification purposes, *e.g.* using the molprobity (Lovell *et al.*, 2003) server at http://molprobity.biochem.duke.edu/, but this verification shows, especially for refinements against twinned data at resolutions that normally would be regarded as low for *SHELXL* refinement, that side-chain torsion-angle restraints and better anti-bumping restraints would be desirable. Peptide planarity has to be imposed by a weak planarity restraint rather than a torsion-angle restraint. This has the advantage that it is possible for a *trans*-peptide to refine to a (correct) *cis*-peptide (Stenkamp, 2005), illustrating a drawback of torsion-angle restraints: they tend to lock the structure into false local minima.

The program *SHELXPRO* is distributed with *SHELXL* to facilitate protein refinement. Although many of the options in *SHELXPRO* have been made obsolete by better verification tools and *Coot* (Emsley & Cowtan, 2004), it is still recommended for interconverting PDB and *SHELX* format files and preparing the first *SHELXL* refinement job. Another facility that is probably better performed with the help of molecular graphics is the *automated water divining* in the program *SHELXWAT*, which iteratively calls *SHELXL* for refinement followed by a difference-electron-density map, renaming suitable peaks as waters for the next *SHELXL* iteration.

One should remember that a set of X-ray data corresponds to electron density that has been averaged over millions of unit cells and over the time taken to collect the data. We usually try to interpret this as a single structure that may contain alternative conformations for the side chains and possibly parts of the main chain. This is clearly a barely adequate approximation for a macromolecule and explains why the free *R* factor very rarely goes under 10%, however high the resolution of the data is. Verification of the geometry with *e.g. molprobity* (Lovell *et al.*, 2003) plus warning signs from a *SHELXL* refinement indicate where the model may need changing or alternative conformations included. Typical warning signs are restraint violations, high displacement parameters (U or U_{ij} in *SHELX*), deviations from NCS and, for anisotropic refinements, non-positive-definite (NPD) and 'may be split' atoms. Often it is a good strategy to reduce the occupancies of offending parts of the structure and run several cycles of refinement before examining the sigma-A weighted difference density to locate alternative sites or conformations. *Coot* (Emsley & Cowtan, 2004) can create this map directly from the *SHELXL* .fcf file (created using the LIST 6 instruction) and can also read and write the *SHELXL* .res and .ins files. Alternative conformations and solvent or cryoprotectant molecules can then be added using PART numbers, geometric and displacement parameter restraints, and coupled occupancies as in the small-molecule examples in section §2.3.

A new program *SIOCS* (Heisen & Sheldrick, 2007) builds networks of interdependent alternative conformations and optimizes them using a genetic algorithm. It is recommended that anisotropic refinement (if justified by a drop of at least 1% in the free *R*) and the modelling of alternative conformations be performed before adding the H atoms, because *SHELXL* can set them up automatically for the alternative conformations. However, one should bear in mind that a macromolecular refinement against high-resolution data is never finished, only abandoned.

3. Conclusions

It is surprising that *SHELX* has been able to maintain a dominant position in small-molecule structure determination for the last 30 years, even though computers have changed out of all recognition in this time. Probably a combination of many factors is responsible. Most of the programs were thoroughly tested and debugged for up to 10 years before being released. Fortran compilers still have to compile a great deal of 'legacy

code' and so are very compatible; the original *SHELX*-76 still compiles and runs without any changes being required. All the *SHELX* programs are very robust and general, with few special cases for the user to worry about, and a minimum number of input and output files. For example, all the programs are valid for all space groups, in conventional settings or otherwise. The reflection data may be in any order and on an arbitrary scale, and equivalent reflections may be present. The reflections are sorted, merged and systematic absences eliminated as required without any special action on the part of the user. A strict *zero dependency* rule is enforced: the programs are distributed as open source and statically linked binaries for common computer systems, and require no extra libraries, environment variables or hidden data files *etc.* The input is – in view of the complexity of the calculations being performed – relatively simple and intuitive. Perhaps most important, there is a large base of experienced users to help. Further information is available from the *SHELX* homepage at http://shelx.uni-ac.gwdg.de/SHELX/.

The author is grateful to the Fonds der Chemischen Industrie for support and to Charles Campana, William Clegg, Ernst Egert, Regine Herbst-Irmer, Peter G. Jones, Thomas R. Schneider, Isabel Usón and many other *SHELX* users for their help and encouragement. In particular, the author would like to acknowledge the crucial role that the late Durward Cruickshank's lecture notes on least-squares refinement (Cruickshank, 1970) played in the design of *SHELX*-76.

References

Allen, F. H., Johnson, O., Shields, G. P., Smith, B. R. & Towler, M. (2004). *J. Appl. Cryst.* **37**, 335–338.
Brunger, A. T. (1992). *Nature (London)*, **355**, 472–475.
Buerger, M. J. (1959). *Vector Space.* New York: Wiley.
Bunkóczi, G., Vértesy, L. & Sheldrick, G. M. (2005). *Angew. Chem.* **117**, 1364–1366.
Burla, M. C., Caliandro, R., Camalli, M., Carrozzini, B., Cascarano, G. L., De Caro, L., Giacovazzo, C., Polidori, G. & Spagna, R. (2005). *J. Appl. Cryst.* **38**, 381–388.
Caliandro, R., Carrozzini, B., Cascarano, G. L., De Caro, L., Giacovazzo, C. & Siliqi, D. (2005). *Acta Cryst.* D**61**, 556–565.
Cowtan, K. & Ten Eyck, L. F. (2000). *Acta Cryst.* D**56**, 842–856.
Cruickshank, D. W. J. (1970). *Crystallographic Computing*, edited by F. R. Ahmed, S. R. Hall & C. P. Huber, pp. 187–197. Copenhagen: Munksgaard.
Cruickshank, D. W. J. (1999). *Acta Cryst.* D**55**, 583–601.
Cuesta-Seijo, J. A., Weiss, M. S. & Sheldrick, G. M. (2006). *Acta Cryst.* D**62**, 417–424.
Debreczeni, J. È., Bunkóczi, G., Girmann, B. & Sheldrick, G. M. (2003). *Acta Cryst.* D**59**, 393–395.
Debreczeni, J. È., Bunkóczi, G., Ma, Q., Blaser, H. & Sheldrick, G. M. (2003). *Acta Cryst.* D**59**, 688–696.
Debreczeni, J. È., Girmann, B., Zeeck, A., Krätzner, R. & Sheldrick, G. M. (2003). *Acta Cryst.* D**59**, 2125–2132.
Diederichs, K. (2000). *J. Appl. Cryst.* **33**, 1154–1161.
Emsley, P. & Cowtan, K. (2004). *Acta Cryst.* D**60**, 2126–2132.
Flack, H. D. (1983). *Acta Cryst.* A**39**, 876–881.
Flack, H. D. & Schwarzenbach, D. (1988). *Acta Cryst.* A**44**, 499–506.
Frazão, C., Soares, C. M., Carrondo, M. A., Pohl, E., Dauter, Z., Wilson, K. S., Hervás, M., Navarro, J. A., De La Rose, M. A. & Sheldrick, G. M. (1995). *Structure*, **3**, 1159–1169.
Germain, G., Main, P. & Woolfson, M. M. (1970). *Acta Cryst.* B**26**, 274–285.
Heisen, B. & Sheldrick, G. M. (2007). In preparation.
Herbst-Irmer, R. & Sheldrick, G. M. (1998). *Acta Cryst.* B**54**, 443–449.
Herbst-Irmer, R. & Sheldrick, G. M. (2002). *Acta Cryst.* B**58**, 477–481.
Konnert, J. H. (1976). *Acta Cryst.* A**32**, 614–617.
Lovell, S. C., Davis, I. W., Arendall, B., de Bakker, P. I. W., Word, M., Prisant, M. G., Richardson, J. S. & Richardson, D. C. (2003). *Proteins Struct. Funct. Genet.* **50**, 437–450.
Miller, R., DeTitta, G. T., Jones, R., Langs, D. A., Weeks, C. M. & Hauptman, H. (1993). *Science*, **259**, 1430–1433.
Miller, R., Gallo, S. M., Khalak, H. G. & Weeks, C. M. (1994). *J. Appl. Cryst.* **27**, 613–621.
Motherwell, W. D. S. (1978). *PLUTO – a Program for Displaying Molecular and Crystal Structures.* CCDC, Cambridge, UK.
Müller, P., Herbst-Irmer, R., Spek, A. L., Schneider, T. R. & Sawaya, M. R. (2006). *Crystal Structure Refinement: a Crystallographer's Guide to SHELXL.* IUCr/Oxford University Press.
Nanao, M. H., Sheldrick, G. M. & Ravelli, R. B. G. (2005). *Acta Cryst.* D**61**, 1227–1237.
Pape, T. & Schneider, T. R. (2004). *J. Appl. Cryst.* **37**, 843–844.
Parisini, E., Capozzi, F., Lubini, P., Lamzin, V., Luchinat, C. & Sheldrick, G. M. (1999). *Acta Cryst.* D**55**, 1773–1784.
Rae, A. D. & Blake, A. B. (1966). *Acta Cryst.* **20**, 586.
Richardson, J. W. & Jacobson, R. A. (1987). *Patterson and Pattersons*, edited by J. P. Glusker, B. K. Patterson & M. Rossi, pp. 310–317. IUCr/Oxford University Press.
Robinson, W. T. & Sheldrick, G. M. (1988). *Crystallographic Computing 4: Techniques and New Technologies*, edited by N. W. Isaacs & M. R. Taylor, pp. 366–377. IUCr/Oxford University Press.
Rollett, J. S. (1970). *Crystallographic Computing*, edited by F. R. Ahmed, S. R. Hall & C. P. Huber, pp. 167–181. Copenhagen: Munksgaard.
Schenk, H. (1974). *Acta Cryst.* A**30**, 477–481.
Schneider, T. R., Kärcher, J., Pohl, E., Lubini, P. & Sheldrick, G. M. (2000). *Acta Cryst.* D**56**, 705–713.
Schneider, T. R. & Sheldrick, G. M. (2002). *Acta Cryst.* D**58**, 1772–1779.
Sheldrick, G. M. (1982). *Computational Crystallography*, edited by D. Sayre, pp. 506–514. Oxford: Clarendon Press.
Sheldrick, G. M. (1985*a*). *Crystallographic Computing 3: Data Collection, Structure Determination, Proteins & Databases*, edited by G. M. Sheldrick, C. Krüger & R. Goddard, pp. 175–189. IUCr/Oxford University Press.
Sheldrick, G. M. (1985*b*). *J. Mol. Struct.* **130**, 9–16.
Sheldrick, G. M. (1990). *Acta Cryst.* A**46**, 467–473.
Sheldrick, G. M. (1991). *Crystallographic Computing 5: from Chemistry to Biology*, edited by D. Moras, A. D. Podjarny & J. C. Thierry, pp. 145–157. IUCr/Oxford University Press.
Sheldrick, G. M. (1993). *Crystallographic Computing 6: a Window on Modern Crystallography*, edited by H. D. Flack, L. Párkányi & K. Simon, pp. 100–122. IUCr/Oxford University Press.
Sheldrick, G. M. (1996). *Crystallographic Computing 7: Proceedings from the Macromolecular Crystallography Computing School*, edited by P. E. Bourne & K. Watenpaugh, http://www.iucr.org/iucr-top/comm/ccom/School96/iucr.html.
Sheldrick, G. M. (1997). *Methods Enzymol.* **276**, 628–641.
Sheldrick, G. M. (1998). *Direct Methods for Solving Macromolecular Structures*, edited by S. Fortier, pp. 401–411. Dordrecht: Kluwer Academic Publishers.
Sheldrick, G. M. (2002). *Z. Kristallogr.* **217**, 644–650.
Sheldrick, G. M., Dauter, Z., Wilson, K. S., Hope, H. & Sieker, L. C. (1993). *Acta Cryst.* D**49**, 18–23.

Sheldrick, G. M. & Gould, R. G. (1995). *Acta Cryst.* B**51**, 423–431.
Sheldrick, G. M., Hauptman, H. A., Weeks, C. M., Miller, M. & Usón, I. (2001). *International Tables for Macromolecular Crystallography*, Vol. F, edited by E. Arnold & M. Rossmann, pp. 333–345. Dordrecht: Kluwer Academic Publishers.
Sheldrick, G. M. & Schneider, T. R. (1997). *Methods Enzymol.* **277**, 319–343.
Stenkamp, R. E. (2005). *Acta Cryst.* D**61**, 1599–1602.
Usón, I., Pohl, E., Schneider, T. R., Dauter, Z., Schmidt, A., Fritz, H. J. & Sheldrick, G. M. (1999). *Acta Cryst.* D**55**, 1158–1167.
Usón, I. & Sheldrick, G. M. (1999). *Curr. Opin. Struct. Biol.* **9**, 643–648.
Usón, I., Sheldrick, G. M., de la Fortelle, E., Bricogne, G., Di Marco, S., Priestle, J. P., Grüttler, M. G. & Mittl, P. R. E. (1999). *Structure*, **7**, 55–63.
Usón, I., Stevenson, C. E. M., Lawson, D. M. & Sheldrick, G. M. (2007). *Acta Cryst.* D**63**, 1069–1074.
Van der Maelen, J. F. & Sheldrick, G. M. (1996). *Anal. Quim. Int. Ed.* **92**, 7–12.
Waser, J. (1963). *Acta Cryst.* **16**, 1091–1094.
Yao, J.-X., Woolfson, M. M., Wilson, K. S. & Dodson, E. J. (2005). *Acta Cryst.* D**61**, 1465–1475.