Acta Crystallographica Section D
Biological Crystallography
ISSN 0907-4449

Interactive electron-density map interpretation: from *INTER* to *O*

T. Alwyn Jones

Department of Cell and Molecular Biology, Uppsala University, BMC, Box 596, S-751 24 Uppsala, Sweden

Correspondence e-mail: alwyn@xray.bmc.uu.se

Received 2 February 2004
Accepted 21 September 2004

A short review of the author's computer-graphics developments since 1976 is presented. Some of the major developments in these programs are reviewed and a description is provided of some of the more recent tools that can be used for electron-density map interpretation. These tools include a secondary-structure template-building system that works in conjunction with a new sequence-decoration system.

1. Introduction

Once upon a time, macromolecular crystallography was very dependent on developments in the computer industry. This dependence was related not just to computer speed but also to computer architecture, primary memory size and secondary storage. Besides having poor performance, computers were expensive, so that even large universities had only one or two machines. These were usually hidden from users, who were not trusted to interact with them directly. Lack of resources limited what could be done and affected the crystallographic algorithms themselves. For example, averaging an electron-density map is quite straightforward in a large-memory computer, but was much more difficult to manage in a small-memory 1976-vintage computer. Indeed, Bricogne's (1976) elegant solution to the problem made use of IBM's highly efficient sort program. As computer technologies developed, crystallographers were often ready to make use of them. For example, one of the first vector processor computers, the Texas ASC, was used by Hendrickson & Konnert (1980) to develop a refinement program, *PROLSQ*, which made efficient use of its capabilities.

I started working on interactive graphics in 1976 in the Computer Centre of the Max Planck Institute for Biochemistry, Martinsried, near Munich. The director, John Gassmann, had just taken delivery of the first Vector General 3400 computer-graphics system, controlled by a Digital Equipment PDP-11 computer. At the time, one could have bought two or three Ferraris for the same money, but instead we had a machine that could rotate in real time a few thousand depth-cued black-and-white vectors. The PDP-11 computer had a 16-bit architecture and limited address space. Today, the cheapest modern notebook computer from a manufacturer such as Apple is orders of magnitude faster and capable of displaying tens of thousands of coloured vectors. It also costs one hundredth of the price of the cheapest modern Ferrari, at least in Sweden.

Up to 1976, almost all macromolecular models had been built as physical wire constructions. These so-called Kendrew models had a scale of 2 cm Å^{-1} and were therefore rather large. The building blocks could be connected and screwed

tightly into place inside a frame of supporting rods. The crystallographer would build a small portion of the desired sequence and then position it within this frame. The fit to the electron density was achieved with an optical system that used a semi-silvered mirror to superimpose the model onto stacked plastic sheets of two-dimensional contoured density. Variations of such a system, called 'Fred's Folly' (Richards, 1968), existed in all protein crystallography laboratories and were slowly collecting dust and undergoing unintended conformational changes (Fig. 1). Coordinate information was converted into a computer-readable form by careful measurements taken from the wire model. Computer-based modelling programs were needed, if only to reclaim the space occupied by the wire models that were cluttering laboratories around the world. They were also needed as tools in crystallographic refinement. By 1976, a number of pioneering efforts had been made to produce more accurate models and a number of different refinement programs had been or were under development. In Martinsried, for example, Deisenhofer & Steigemann (1975) had used the real-space (RS) refinement program of *Diamond* (1971) to refine pancreatic trypsin inhibitor at 1.5 Å resolution. This consisted of multiple rounds of fitting the model to calculated maps (derived with phases calculated from the current atomic model) until the crystallographic R factor converged. The model and density were then plotted on sheets of paper and attempts were made to modify the model where appropriate. This step was difficult and time-consuming and this was what I decided to try to simplify.

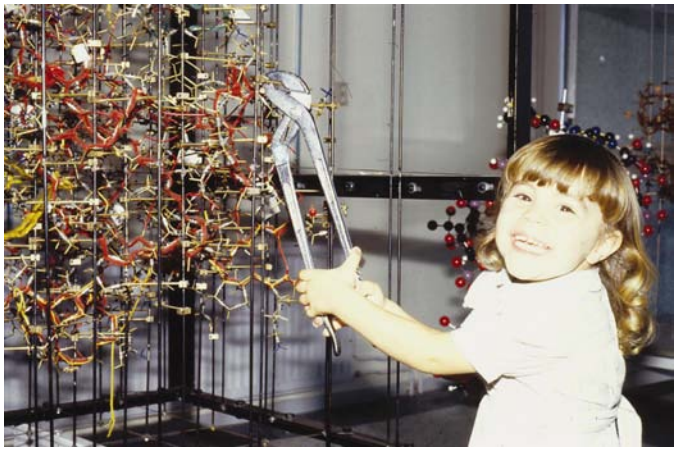


Figure 1
This is a Kendrew wire model of alcohol dehydrogenase that is about to undergo a round of rebuilding by Maelle Cambillau.

2. *INTER*, a tool in refinement

In 1978, I published a short description of a computer program, *INTER*, that was intended primarily to simplify the rebuilding stages needed during crystallographic refinement (Jones, 1978). The electron densities were calculated with Steigemann's *PROTEIN* system and then contoured in small overlapping volume elements on a large Siemens 4004 computer. This file, which could contain multiple contouring levels, and the *Diamond*-formatted coordinates file were then transferred to the graphics system over a local network, where they were saved on removable RK05 disks (capable of storing 1.5 million words). The user interacted with the graphics system using a pen/tablet pointing device and sets of dials. The pen could be used to activate a menu of commands and to identify the atoms drawn on the screen. The dials were attached directly to the VG3400 *via* an analogue-to-digital converter and were used to control the view as well as various functions. The user could display a portion of the atomic model, either as a contiguous zone of atoms or all of the atoms close to some point in space. Because of memory-space restrictions, only a few hundred atoms could be displayed at a time. More complex pictures could be generated, but these were merely vectors and could not be 'picked' by the user.

The vectors between atoms on the screen were drawn if the paired atoms were close in space. The user could change this 'screen connectivity' by making or breaking selected bonds. One set of dials allowed the user to move single atoms or groups of linked atoms around relative to the electron density background. Up to six linked torsion angles could also be changed. It was therefore possible to modify the starting model in quite general ways, hopefully improving the fit to the density (Fig. 2). Indeed, the user was free to wreak havoc upon the atomic model, disturbing any sound stereochemistry that might have existed. Provided the disturbance was not too great, sensible stereochemistry could be restored using the Hermans & McQueen (1974) regularization. This spread the build-up of stereochemical deviations from expected values over bonds, angles and fixed torsion angles. To keep atoms where the user wanted them, selected atoms would have to be fixed during regularization. This made it easy to flip a peptide, for example,

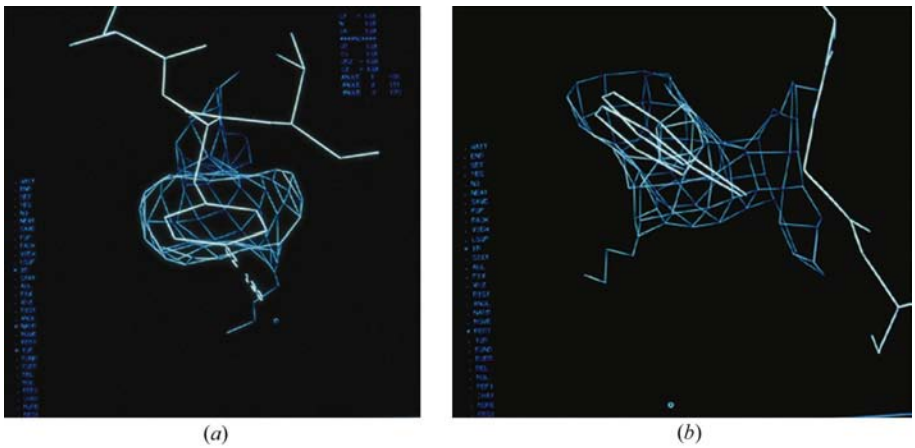


Figure 2
In these early *FRODO* slides, the ring has been fitted to the density *via* torsional rotations (*a*); a fragment was then defined and moved into the density as a rigid body (*b*).

simply by moving the O atom, fixing it and then regularizing a small zone around the peptide. Because the O atom could not move, abutting atoms would be forced to move to satisfy the stereochemical restraints. Without fixing, the O atom would have jumped back more or less to the starting position.

Only one precontoured map and only one molecule could be viewed at the same time. In regions of lower electron density, the user would have to go back to the Siemens 4004 to recontour. Such limitations were a relatively small price to pay and essentially the same rebuilding strategy is in use today. I was fortunate to be working close to Robert Huber's research group and to have Johann Deisenhofer as a user. At the time, he was refining an immunoglobulin Fc fragment at medium resolution (Deisenhofer, 1981) and his constructive feedback was always helpful. Working close to Huber's laboratory meant that once I had something useful, I had a stream of potential users with new structures to explore. Soon, I had to face the question of how to use the graphics system to replace the initial model-building stage and I explored a number of alternatives. I wrote an option that allowed one to move along the sequence, adding each residue in an α-helical conformation. Once this was fitted into the density, the next residue could be generated. Alternatively, a whole helix or strand could be generated and then interactively fitted to the density as a rigid group before adjusting the side chains. Because the graphics system could not display many vectors, the actual tracing stage (where one decides how the whole molecule folds in space) was best carried out with two-dimensional contoured mini-maps (plastic sheets plotted at a scale to produce a manageable stack of size $\sim$30 cm). This stage is critical since it requires both a global overview of the density and a detailed view at the level of peptide bumps and side chains. I therefore developed options to build secondary-structural elements that could be least-squares fitted to a set of guide points that had been derived from a mini-map. The chain between these elements could be generated from guide points (optimally three atoms per residue: C^{α}, O and a side-chain atom) to produce a first rough model (Jones, 1982), which was then carefully fitted with the interactive rebuilding tools. I spent a year in Robert Huber's group, trying out new tricks with each structure that came along, but eventually moved to Uppsala in the spring of 1979.

3. *FRODO*: widespread adoption and colour

In 1979, there were two protein crystallography groups in Uppsala, Bror Strandberg's and Carl-Ivar Brändén's, in two different universities. The new Vector General/PDP-11 and I were the first things they shared. The program had already picked up a new name, *FRODO*. The constant movement of data to and from the computers in Martinsried had resulted in a set of programs that I named after Tolkien's characters and it felt natural to name the graphics program after the hobbit. By now, three-dimensional computer graphics had become better established, although not yet widespread. That autumn (never my favourite time in Sweden), Tom Blundell invited me to Birkbeck College for two weeks, where I converted *FRODO* to run on an Evans & Sutherland Picture System 2 (E&S PS2)/PDP-11. Although they shared a common controlling computer, the graphics systems were very different. This was the first of many graphics conversions that I was to make. The following autumn, at the invitation of Michael Rossmann, I converted *FRODO* to run on an MMS-X controlled by a Texas Instruments 980B computer system. These systems had been constructed at Washington University for some American and Canadian crystallography laboratories. The different graphics and computer architectures required another major rewrite, but I now had the program running on three different computer-graphics systems. I did not push back any scientific frontiers with these conversions, but the program was now available to a larger community, I had met a lot of wonderful people and I had learned something about writing programs for alternative computer-graphics systems. Ian Tickle considerably improved my PS2 conversion at Birkbeck and Bruce Bush at Merck soon afterwards made the conversion to a VAX/E&S MPS system.

In Uppsala there were challenges to face, especially the structure determination of satellite tobacco necrosis virus that was under way in Bror Strandberg's group. Before my arrival, the structure had been solved to 4.0 Å resolution (Unge *et al.*, 1980), but the crystals diffracted to beyond 2.4 Å, with the whole virus in the asymmetric unit (60 icosahedrally related chains). Our strategy to solve the structure was to make use of non-crystallographic symmetry (NCS) as much as possible. We used NCS averaging to improve low-resolution experimental phases so that we could build a model (Liljas *et al.*, 1982). The model was used to extend the phases to higher resolution, where they were improved with NCS averaging. The model was then improved by RS refinement into the newly averaged map. The phase extension and refinements were carried out in steps out to the highest resolution of 2.4 Å (Jones & Liljas, 1984*a*). For RS refinement, I wrote a stand-alone program for the PDP-11, which split each amino acid into fragments that could then be independently optimized in the density (Jones & Liljas, 1984*b*). This distorted structure could be viewed on the graphics system, regularized and rebuilt where necessary. After a number of RS-fitting and regularization cycles, we had produced a well fitting subunit that could then be used for the next round of phase extension. I used a bricked format to store the density, with each value packed into one byte. Shortly afterwards, we replaced the PDP-11 with our first 32-bit computer, a DEC VAX 750, which essentially eliminated computer-memory problems from program development. For computer graphics, this meant I could work with entire maps instead of precomputed contours and the RS fitting could take place in *FRODO*. Although I predicted that

> The use of 32-bit computers to control the display also offers the possibility of refining a small portion of the molecule (either in real or reciprocal space) under the control of the display user, as the user continues working on the next section of molecule

(Jones, 1982), I did not develop RS refinement further, beyond extending its use to fitting rotamer conformations (Jones *et al.*, 1991). Reciprocal-space methods took over as the tools of

choice for refinement, although there were clear indications that initial models were not optimally fitting the experimental map (Mowbray *et al.*, 1999). I still argue that RS refinement should be the method of choice in virus crystallography, however. As for my move into reciprocal space, it started ten years ago but is still not available to users.

At the beginning of the 1980s, Evans & Sutherland started selling colour graphics systems that were even more expensive than the black-and-white models. I had plans for using colour computer graphics, but had to wait until 1985 when we took delivery of an E&S PS330. This family of graphics systems was totally different from the PS2 and related MPS, but a *FRODO* conversion had been made earlier by Pflugrath *et al.* (1984). With their conversion as my starting point, I made use of colour for presentation graphics (Fig. 3), allowing one to colour by atom type or amino acid or to use colour ramping according to position in the sequence or atomic temperature factor. More importantly, I wanted to use colour as a way to work effectively with a different density representation, skeletons.

4. Skeletons and databases

Early versions of *FRODO* provided a good way of working with details of an atomic model. However, if one wants to trace a structure using just computer graphics, one needs to provide some sort of overview as well as the details. The best possible overview would be the correct C^{α} trace, so how could one achieve this? Initially, users were forced to use 'mini-maps' to produce this trace, but an alternative representation of the electron density offered a better option. Greer (1974) had described a skeletonization algorithm to convert the three-dimensional array of density values into connected paths through space. His intention was to use this representation as the basis for automatically building an atomic model from just the electron density (Greer, 1976). This representation seemed to me to be ideal for providing the overview that had been lacking previously.

In my first implementation of a skeletonized density, I generated a file that was equivalent to a precontoured electron-density map. Although this provided the needed overview, the connectivity could not be changed, and it was soon obvious that this was not particularly useful. It was also clear that one needed to indicate what the skeleton represented; was it main chain, side chain, partly traced, possibly traced *etc.*? For this I needed colour computer graphics so that a user could 'paint' his/her hypothesis onto the skeleton as the trace developed (Jones & Thirup, 1986). The skeleton could then be used for the overview and the usual chicken-wire contouring of the density could be used to explore the details. By looking at these details, the user would be able to modify the skeleton where necessary, for example to make new connections in regions of weak density or remove connections where there were errors, and then use colour to update the latest folding hypothesis (Fig. 4).

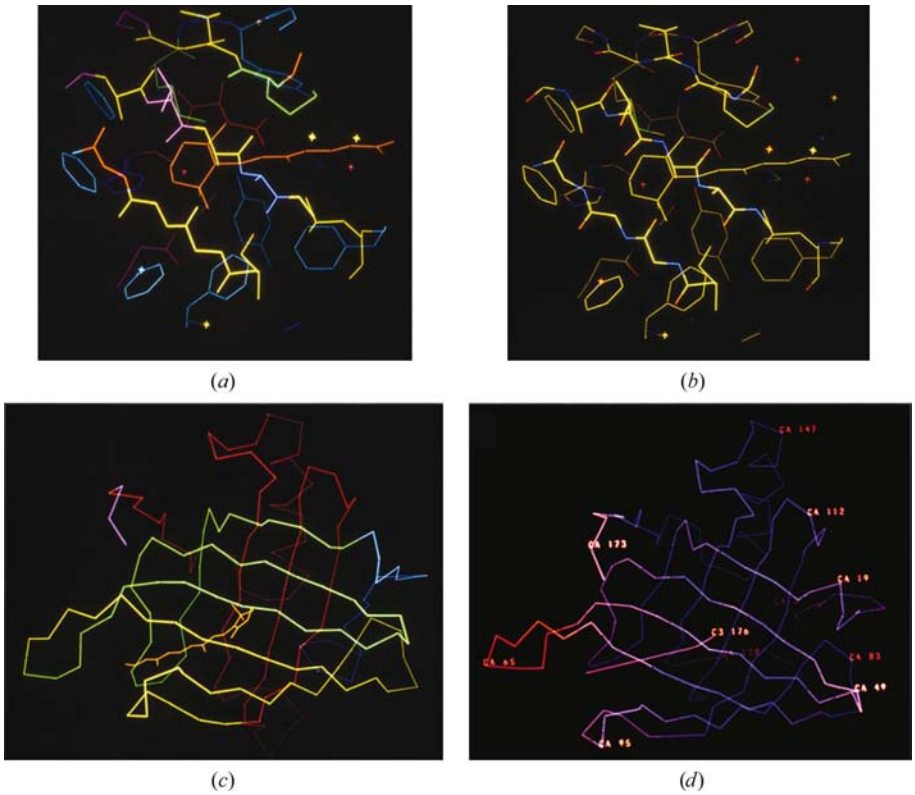


Figure 3
These are early PS330 *FRODO* illustrations, with atoms coloured according to (*a*) amino-acid type, (*b*) atom type, (*c*) position in the sequence (rainbow ramping) and (*d*) temperature factor (colour ramping from blue to red, low to high *B* factors). The structure is retinol-binding protein (Newcomer *et al.*, 1984).

One could think of a number of ways of constructing an atomic model from an edited skeleton. I decided to use the most similar pieces of existing structures. I reasoned that most crystallographers did not know the difference between a type I and II turn, but if the skeleton looked liked a particular piece of structure then it should just get selected and used to build the new model. By comparing local pieces of the skeleton with refined high-resolution structures, only stereochemically reasonable fragments would be selected (albeit with imperfect stereochemistry between the fragments). Hopefully, the carbonyl O-atom orientations from the known structure would point in the same direction as those in the new structure. During debugging of the computer code, I was pleasantly surprised by the results. I was using a small set of structures, only those that I had solved or refined in Uppsala. I found that the new retinol-binding protein (Newcomer *et al.*, 1984) that we had just solved (using hand-skeletonized density to obtain the overview!) could be reconstructed almost perfectly

from bits and pieces of just four other proteins. I found that a reverse-turn conformation that I considered to be 'new' had already existed in alcohol dehydrogenase. For obvious reasons, I called this process protein Lego modelling. In my implementation, the user selected a stretch of connected skeleton and then the C^{α} positions were determined based on skeleton branches or the distance travelled along the skeleton. These C^{α} positions were then compared with the database of structures and the 20 best fits were displayed on the screen. To make it fast, I used a $C^{\alpha}-C^{\alpha}$ distance matrix approach to select likely fragments, followed by a least-squares alignment of their C^{α} positions. The user could scan the suggestions and accept the complete main chain of the best fitting one (Jones & Thirup, 1986). A later analysis (Jones *et al.*, 1991) indeed showed that although there could be some peptide fanning, if the C^{α} backbone was close to the database structures the peptide O atom would be placed in the correct orientation. Getting the peptide plane oriented correctly is the key to building a good initial structure, but at low resolution with poorly phased maps peptide bumps are not usually apparent. Building a model therefore became a problem of selecting the right fragment rather than the right conformation. I also carried out a fragment cluster analysis but never published the results. It was disappointing to find that I already knew most of the clusters that showed a strong sequence preference. A separate fragment layer was therefore never added to the database approach.

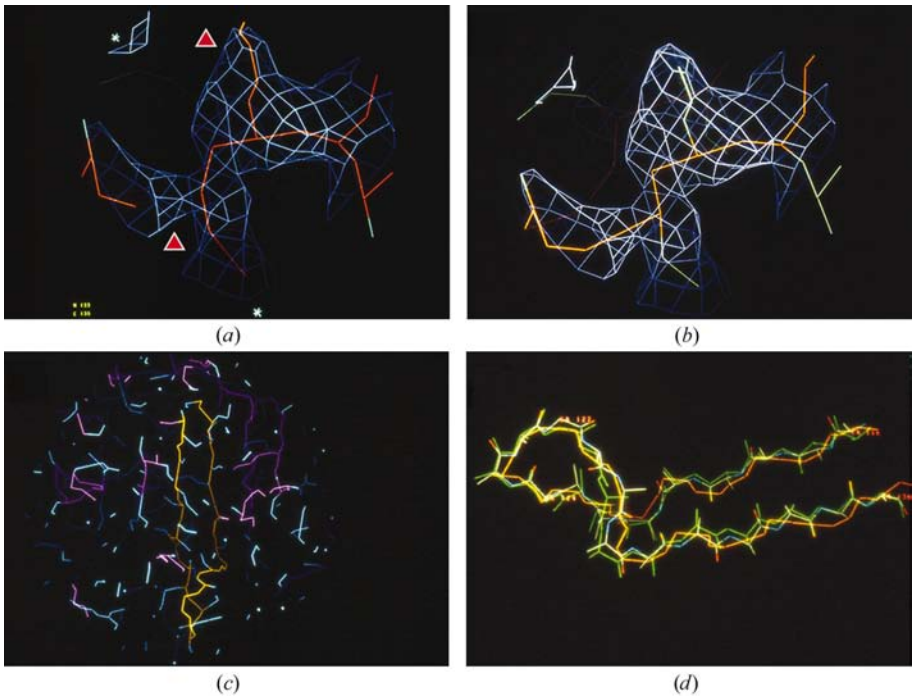


Figure 4
These PS330 *FRODO* slides illustrate the use of skeletonized density with a main-chain database. The density is the retinol-binding protein experimental map (Newcomer *et al.*, 1984) with (*a*) the unedited skeleton (triangles indicate errors in the skeleton), (*b*) a somewhat edited skeleton, (*c*) a golden-coloured partly traced portion of the skeleton and (*d*) an overlay of the traced portion with a polyalanine built from the trace in pentapeptide fragments.

5. A new start: *O*

Although I had been able to add colour and skeletons to *FRODO*, it was clear that the program was showing its age. In 1986, I started developing a new program that would give me more flexibility in what could be displayed and associated with a structure. I decided that the program would use a database to store most of the data, from molecular information to the program's own keywords and default values. Functionality would be self-contained to allow collaborative development, using the database to save relevant state values. For almost ten years, Morten Kjeldgaard collaborated with me to create the features that would be needed for a fully fledged modelling program. We each developed the things we were most interested in, meeting once or twice a year to thrash out new ideas, implement them and ensure that we kept to the '*O* way of doing things'. Most importantly, the design would allow easy porting to new graphics systems as they developed.

It was my intention to encourage the use of main-chain and side-chain databases wherever possible, *e.g.* to fit with rotamers instead of torsion angles, and to provide various goodness-of-fit indicators. By using a database, it would be possible to use and save any atomic or residue-based indicators, even those that did not yet exist. This approach and some of the indicators that we thought useful are described in Jones *et al.* (1991). These atom- or residue-based properties could also be used to select what is displayed and the atomic colour, for example. Some were available in *O* as new functions or they could be read into the program as *O* database entries (datablocks). Directly in *O*, the user could evaluate peptide orientations (Pep_flip), similarities to rotamer conformations (RSC_fit) and the fit to the density (RS_fit). The latter function could be used to assess the fit of the main chain, side chain or all atoms in the residue to the density. Originally defined as a real-space *R* factor, the command was later extended to allow the calculation of a correlation coefficient. I thought this was the first use of a density-fit indicator, but Wierenga *et al.* (1987) had already plotted the average density per residue, calculated at N, C^{α}, C main-chain atoms, of their *T. brucei* triosephosphate isomerase structure. Gerard Kleywegt, in particular, has written numerous programs that generate *O* datablocks, ranging from sequence conservation in a group of aligned sequences to combined goodness-of-fit indicators (Kleywegt & Jones, 1996).

Jones *et al.* (1991) evaluated the sensitivity of the main-chain database approach to errors in C^{α} position and experimented with building a complete structure from just C^{α} atoms

(adding the side chains by RS fitting the allowed rotamers for each residue) and then carrying out a preliminary crystallographic refinement. The use of a minimalist approach to building and refinement (especially at low resolution) had been prompted by our results on refining P2 myelin protein, a low-resolution structure (incomplete 2.8 Å data) with three molecules in the asymmetric unit (Cowan *et al.*, 1993) and my evaluation of models generated without the use of side-chain rotamers. In the case of P2 myelin, simulated-annealing refinement (Brünger & Krukowski, 1990) had produced larger differences between the three molecules than was justified from inspecting the experimental map. This led to an enjoyable collaboration with Gerard Kleywegt, where we probed the consequences of allowing too much freedom in refinement (Kleywegt & Jones, 1995; Kleywegt, 1996) and preached about using refinement protocols that were appropriate to the problem at hand (Kleywegt & Jones, 1997*b*).

To assist in the placement of sequence, I developed the *Slider* commands in 1991 [mentioned briefly by Zou & Jones (1996) and Jones & Kjeldgaard (1997)]. These commands allowed a user to guess a local portion of sequence, obtain a score of the best matches, associate the guess with a piece of structure and finally store the results in the database. Multiple guesses could be combined and evaluated interactively by the user. Zou & Jones (1996) showed that a quantitative approach would also work, even at low resolution. This approach required a well fitting polyalanine chain to act as the framework for evaluating the goodness-of-fit to the density of the 20 amino acids at each position in the chain. At each position and for each amino acid, we evaluated the fit of each rotamer, allowing the residue to pivot around the C^{α} atom. Since small residues (such as glycine, alanine and serine) always fit into the density meant for a larger residue, these were evaluated with an extended mask. For a good low-resolution experimental map (*e.g.* the averaged P2 myelin experimental map), we showed that 86% of the guesses were correct when using segment lengths of 15 residues. This fell to 67 and 35% for segments of eight and of five residues, respectively. The worst behaving segments corresponded to regions of the structure where the side chains of separated residues interacted. This version of *O* was never distributed, mainly because it lacked an elegant way of quickly fitting the polyalanine framework.

6. Using secondary-structure templates (SSTs)

It is well known that proteins are built up from secondary-structure units connected by sometimes long, but often short, segments of chain. In well phased skeletonized maps, these units are easy to spot. However, one has to be aware that at lower resolution α-helices tend to be somewhat 'stretched'. As computers became faster, we were tempted to try to recognize these secondary-structure features automatically. In one method, template convolution (*ESSENS*; Kleywegt & Jones, 1997*a*), we check each point in a map to see how well a piece of structure matches the density and then create a new map with density values corresponding to the best fit at each point. To determine the best fit, the structure is rotated through all possible orientations and the fit is evaluated at each setting with a simple scoring function. Although the rotated structure can be any structure (a nucleic acid helix, a haem group, a retinoid, an indole ring *etc.*), we normally use short five- or seven-residue α-helix and β-strand templates. Because the template is rotated about the middle C^{α} atom, the densities follow the shape of smoothed SSTs. The results for the averaged P2 myelin map are shown in Fig. 5, clearly indicating all of the secondary structure in the protein. However, because the calculation is carried out at every grid point in the map it takes some time, even with a simplified scoring function. Cowtan (1998) realized that one could carry out a faster reciprocal-space formulation and this has been distributed as part of the *CCP*4 program package. However, the real-space formulation can be speeded up by a factor of 30–100 by only evaluating the goodness-of-fit at skeleton atoms. This speed increase made it reasonable to implement the calculation directly in *O* as part of the SST-building system.

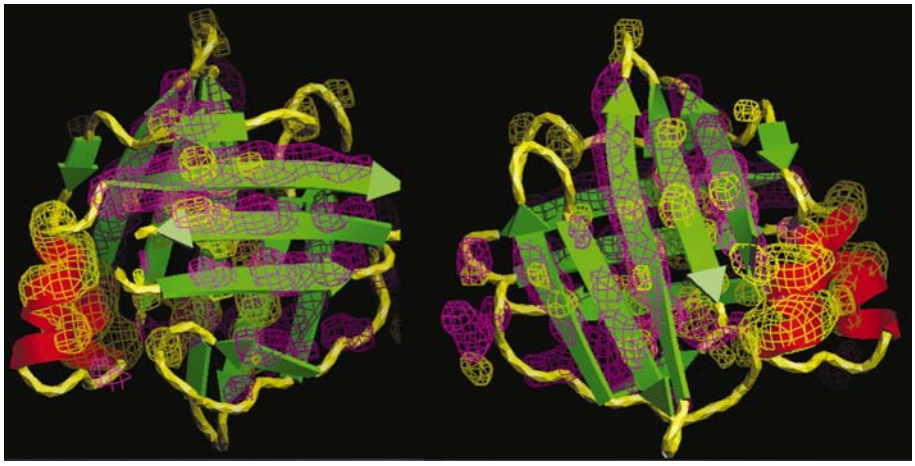


Figure 5
Two views are shown of *ESSENS* maps of the averaged P2 myelin experimental density. Yellow contours indicate the α-helix template map and purple contours indicate the map generated using a β-strand template. All helices and strands in the three-dimensional structure are apparent, even the more distorted fifth strand of the barrel.

The current release of *O* works with three different classes of molecule for map interpretation. The user's molecule has global directionality and is decorated with the molecular sequence. The TRACE molecule, on the other hand, has only local directionality (localized to either short or long sections, depending on the status of the tracing) and no sequence (a polyalanine for a protein trace and polycytosine for RNA). It is actually just a molecule in the *O* database called 'TRACE', with an extra residue property that is used to mark the residues being used. The third molecule is a skeleton molecule named 'CAT' that has neither directionality nor sequence. SSTs can be generated in

the TRACE molecule as short α-helices and β-strands (or RNA helices) with local directionality. They can be produced interactively, most simply by using the template menu (Fig. 6). Clicking on a skeleton atom causes a complete three-dimensional rotational scan of the selected SST in the selected density. If the user accepts the result, it becomes a part of the TRACE molecule. The density must first be loaded into the *FastMap* system of *O*, which allows interactive changes to contouring parameters *via* sliders. If a crystallographic asymmetric unit is provided, the user can work anywhere in space.

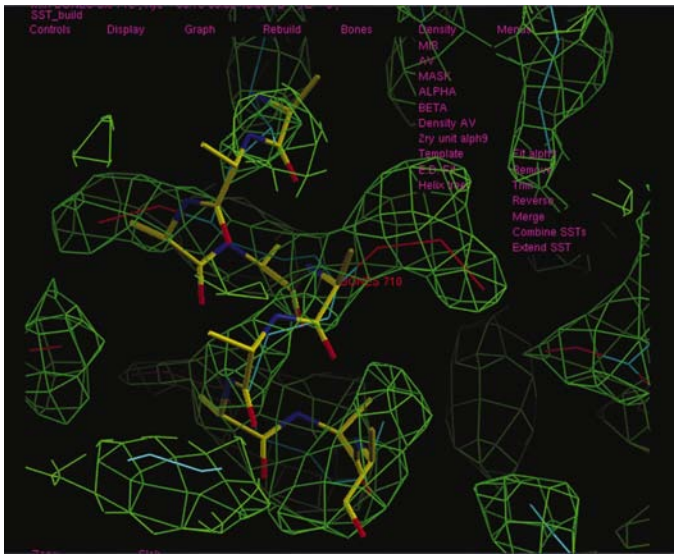


Figure 6
A nine-residue α-helix has been built into the unaveraged P2 myelin experimental map at skeleton atom 710. The SST menu is expanded and visible to the right of the centre.

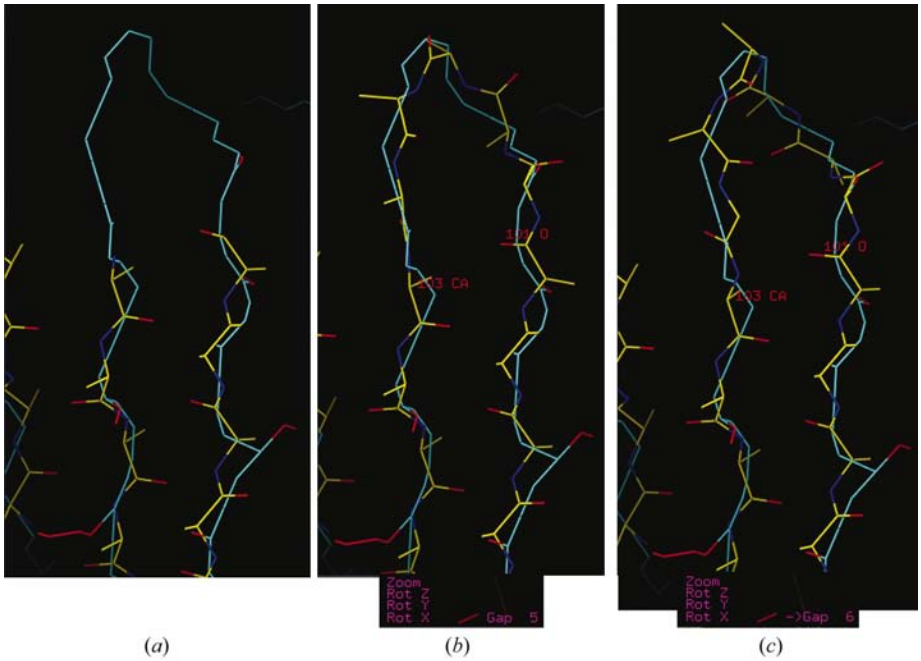


Figure 7
In (*a*), two β-strands have been generated with the Sprout commands. (*b*) They are combined into a larger unit with a suggested gap of five residues using SST_combine. In (*c*), the user has asked for a six-residue gap.

Once an SST is built, rigid-body RS refinement can further optimize the fit to the density. Both the SST and the new RS fitting options use a simple sum-function goodness-of-fit indicator, rather than the slower but more accurate functions of Jones & Liljas (1984*b*). SSTs can be deleted, trimmed, flipped, reversed, merged into the molecule being built or combined with other SSTs (Fig. 6). In the latter function, the connection between SSTs is made by following a skeleton to place the C^{α} atoms and using the main-chain database to build the new portion of chain. The user is able to change interactively the number of residues in the gap (Fig. 7), because skeletons often take 'short cuts' through reverse turns, especially at low resolution.

A helical SST is also useful for determining chain directionality, a vital part of the tracing process. At high resolution, carbonyl O-atom bumps are apparent in the density, as well as side-chain branches from the C^{α} atom. Short distances between branch points correspond to $C_i^{\alpha}-C_i$ vectors, while slightly longer distances correspond to $C_i-C_{i+1}^{\alpha}$. Provided the carbonyl O atom is located, the directionality is then defined (Greer, 1976). At low resolution, the peptide bumps are smeared out, but directionality can be determined by the helix Christmas-tree effect (Jones, 2001). In an α-helix, the $C^{\alpha}-C^{\beta}$ vector points towards the N-terminal end of the helix; if the decorating side chains are visible, the directionality is determined. Local averaging can enhance the effect; since we have main-chain coordinates, we can determine operators from the central residue to all other residues in the helix. Using these operators, it is then possible to build a locally averaged density around the central residue of the helix. If the helix is correctly oriented, the $C^{\alpha}-C^{\beta}$ atoms of this residue will fit snuggly in the density; otherwise they will not (Fig. 8). Once the directionality has been determined, the SSTs become suitable frameworks for the qualitative *Slider* commands mentioned earlier.

The SST framework can also be generated automatically. An *ESSENS*-style three-dimensional rotational search is carried out at each atom in a specified skeleton object for each desired SST template (Sprout_setup). The user can then specify how many SSTs to build (Sprout_SST) or let the program decide (Sprout_auto). Each Sprout_setup takes 5 min to run on an Apple PowerBook (1 GHz model) for an object of 725 atoms with an $\alpha 9$ template, and 3 min with a $\beta 5$ template. The other commands take a few seconds. The resulting framework can be edited as described earlier (Fig. 6). At low resolution, the SSTs will be randomly oriented with respect to the correct directionality and will therefore need to be flipped according to the local averaging results. For the P2

myelin map, both helices were correctly oriented, as were ten of the 11 strands. It may also happen that a β-SST will be misaligned so that the carbonyl groups are pointing into the side-chain density (none of the β-SSTs in the P2 myelin case). At higher resolution, the number of errors will be reduced as the atomicity of the map improves. In the experimental map of ribokinase, for example, which was used to test model-building reproducibility (Mowbray *et al.*, 1999) and phased to 2.6 Å resolution, six α-helices can be positioned automatically with the correct directionality before an error occurs.

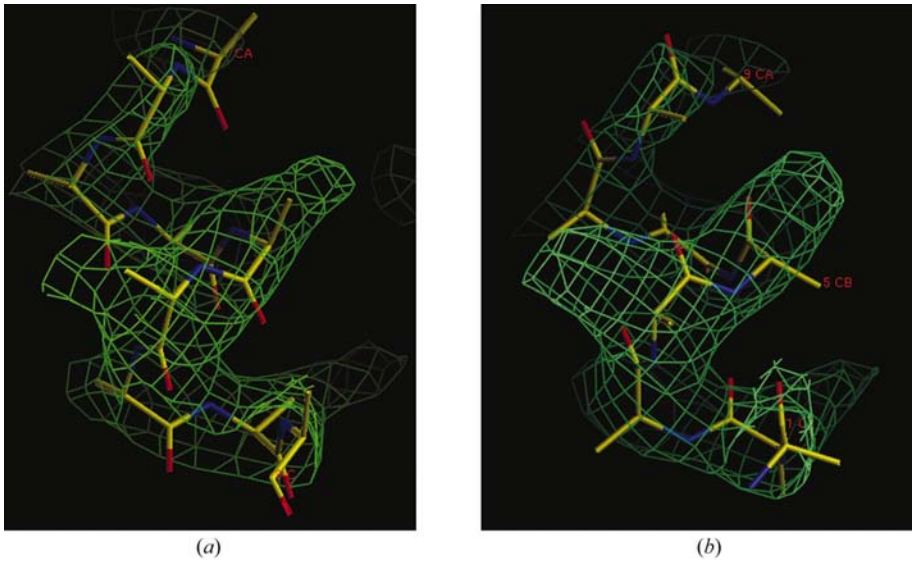
Any SST or combination of SSTs can be used to determine the qualitative fit of the sequence to the density (Decor_guess). The scoring function is simpler than that used by Zou & Jones (1996) in order to allow a more rapid evaluation. The RS rotamer-fitting commands that are used with the *FastMap* system generate a local mask around the position of the expected side chain, which is dependent on the current contouring level and the local density connectivity. Only density points within this mask are considered and the scoring function is the correlation coefficient of this mask with a mask of the best fitting rotamer, generated with a fixed 1.5 Å atomic radius. This scoring function allows one to evaluate small residues when fitting in the density meant for large residues and *vice versa*. The results of the qualitative fitting are stored in the *O* database for use with the *Slider* and *Decorate* (to be described below) systems. Since beginners often do not appreciate the size of different amino acids, there is a command (Decor_show) that overlays the best fit of each amino-acid type at a specified residue in the TRACE.

7. The CAT

The central atom trace (CAT) is a skeleton that can be generated by a number of *O* commands. These commands aim to produce as complete a trace as possible of the atomic structure. Since the CAT is also a skeleton, it can be easily edited; bonds can be made or broken, new atoms added or atoms moved. One command, Trace_grow, is interactive: starting from an SST, a skeleton is traced until there is a decision to be made. As one moves along this skeleton, a central atom trace (the C^{α} atom for proteins, the P atoms for nucleic acids) is developed depending on the mode being used. For proteins, the placement can be made at the correct $C^{\alpha}-C^{\alpha}$ distance (3.8 Å) along the skeleton, at a nearby skeleton branch point or *via* a full three-dimensional peptide-fragment rotational search (unless too far from a skeleton placement). For nucleic acids, a different option places the P atom at a local density peak along the skeleton (in the P—P range 5–7.5 Å). The tracing stops when it encounters a main-chain branch point or a break in the skeleton. The user can then modify the skeleton or the CAT and continue or start towards the C-terminal from the initial SST starting point. SST segments are absorbed as they are encountered along the skeleton.

Trace_Ca is an automatic tool that produces a CAT from a density skeleton. If the skeleton contains many breaks or spaghetti-like connections, the CAT will not be complete. Even in these cases, however, the CAT may help the user to concentrate on problem regions. The time taken varies depending on how badly connected the skeleton is; normal skeletons are processed in ~1 s, but a skeleton with many branched main-chain connections could take minutes to process. Again, SSTs are absorbed as they are met and the ultimate quality of the CAT depends on the quality of the skeleton being processed. In relatively poorly phased maps, therefore, the user will have to edit the experimental skeleton before producing the CAT. If crystallographic symmetry is specified, this tool generates a CAT that represents the asymmetric unit by combining the various segments into a compact structure.

From the CAT, a new TRACE can be generated (Sprout_CAT) in one of three ways: using the main-chain-fragment database approach (for low-resolution maps), by spinning a peptide fragment ($C^{\alpha}-CO-N-C^{\alpha}$) around each CAT vector and selecting the best fit to the density (medium-resolution case) or by carrying out a full three-dimensional rotation of the peptide fragment at each CAT atom (high-resolution case). The resulting TRACE would normally correspond to a larger part of the structure than a simple SST fragment. Therefore, it is likely to contain out-of-register errors. Such errors occur when a short stretch of sequence is built out of synchronization with the density. These errors are often localized in the structure, beginning and ending with a



(*a*) (*b*)

Figure 8
In (*a*), a nine-residue helix has been fitted to the experimental P2 myelin density with the SST system and then local averaging has been used to evaluate the directionality. In (*b*), the helix has been flipped and RS fitted and the density locally averaged. The fitting of the C^{β} of the central residue indicates that the directionality is correct in (*a*) and wrong in (*b*).

locally incorrect main-chain conformation (Jones & Kjeldgaard, 1997).

Before the TRACE is decorated with the sequence, it should be RS-fitted to the electron density. If the resolution and phasing are appropriate, this can be accomplished with a new tool (Fm_rsr_zone) that splits a segment of chain into a number of rigid fragments. For a polyalanine TRACE, each residue is split into CA—CO—N_+—CA_+ and N—CA—CB—C fragments, where N_+ and CA_+ are atoms in the next residue. Atoms that occur in more than one fragment in a segment are restrained to remain linked together to prevent excessive drift during the fitting process. In low-resolution poorly phased maps, RS-fitting may disrupt the fit of a chain to the density.

8. Sequence decoration

To decorate a TRACE with the molecular sequence, I use a dynamic programming algorithm (for a useful general description of such algorithms, see Sedgewick, 1983) to thread the known sequence onto the structural framework. At each residue in the TRACE, the goodness of fit of the 20 possible amino acids is first evaluated (Decor_guess, taking about 5 min on a PowerBook for a TRACE of ~120 residues). The dynamic programming algorithm (Decor_slide) does not carry out a Needleman & Wunsch (1970) style residue-to-residue alignment with gap penalties. Instead, a series of 'stretches' are defined, where each stretch corresponds to a short zone of residues in the TRACE, and any goodness-of-fit calculation is averaged over each residue in the stretch. It is implied that the first defined stretch comes before the second in the sequence, the second before the third *etc*. The algorithm works as follows: the last stretch in the TRACE is tested starting at all possible residues in the sequence. The score associated with each possible residue in the sequence is the average fit of the stretch when it has been decorated with the sequence starting at that particular residue. For example, if the last stretch is a fragment of five residues and the protein contained 252 residues, the score at position 123 in the sequence would be the average score of placing the residue type of 123 (say a Phe) at position 1 in this stretch, residue 124 (say a Thr) at position 2 and so on until the sequence of 127 is evaluated at position 5. Residues at the start and end of the sequence are not tested because of the finite size of each stretch, so that in this example residues 249–252 would be excluded from the calculation. The residues that are excluded at the N-terminus would correspond to the total number of residues making up the other stretches in the framework. The scores associated with the last stretch, therefore, are simply measures of how well this stretch would fit at all allowed parts of the sequence.

The score associated with the penultimate stretch is slightly different: at each allowed residue in the sequence, the score represents the sum of the score of the stretch at this position plus the best fitting thread that exists after this position in the sequence (in this case, the fit of the final stretch in the framework). This is repeated for all stretches until we reach the first one. At each step, the excluded regions at the N- and C-termini change so that when the first stretch is evaluated it is checked from the first residue in the sequence but excluded from the C-terminus by the total lengths of all following stretches. At each stage, the path through the alignment is saved for later retrieval. The time taken to generate the optimal path is ~1 s on a PowerBook.

The algorithm also allows one to fix the start of a stretch to correspond to a particular residue in the unknown structure and to specify the amino-acid type for a particular residue in a stretch. For example, the density might indicate that a certain residue in the TRACE corresponds to an aromatic side chain. For selenomethionine-substituted proteins, the ability to define the position of methionine residues on the TRACE is a particularly useful consequence of the MAD/SAD phasing method. The algorithm leaves some flexibility with regard to the maximum allowed gap size between pairs of consecutive stretches of the TRACE relative to the sequence. It also allows the exclusion of parts of the sequence from the threading.

Fortunately, the user does not have to be aware of these implementation details. The stretches are generated by default from a *Yasspa* evaluation (Kleywegt & Jones, 1997*c*) of the framework so that they correspond to α and β segments. This approach is intended to reduce the risk of register errors. The optimal path is displayed superimposed on the TRACE (Fig. 9). Where the gap size between stretches in the framework is different from the optimal path, a small '*' is drawn. The user may then want to force a particular residue in a stretch to correspond to a particular residue in the sequence (Decor_fix) or to particular amino-acid types (by editing the *O* datablock entry that is the input to the threading). The optimal path is immediately recalculated and displayed. Eventually, if the user ever finds an optimal path that is acceptable, the TRACE main chain can be merged into the unknown structure and the side chains can be RS-fitted as rotamers to the density. The gap regions between stretches will also be built if the difference between the TRACE and the sequence thread is small. During the merging process, the program keeps track of what has been built, so that in a future run the relevant sequence is excluded from the search.

9. Rebuilding during refinement

At low resolution, it is likely that the first model(s) will contain out-of-register errors. *O* provides a tool (Build_slider) to correct this sort of error easily, where the user merely redefines the position of the offending stretch in the sequence. A second input toggles how to generate the side chains, either as the main rotamer (very quickly) or RS rotamer fitted (quickly, but dependent on the length of the stretch).

The *O* command most frequently used during rebuilding is Grab_build. This combines the most widely used rebuilding tools in one window panel, including residue rotate/translate, torsions and rotamer generation. The user is free to move forwards or backwards through the sequence, generating three-dimensional density contours in the process. This command can also be used to generate secondary-structure elements, to follow a skeleton or to make a three-dimensional

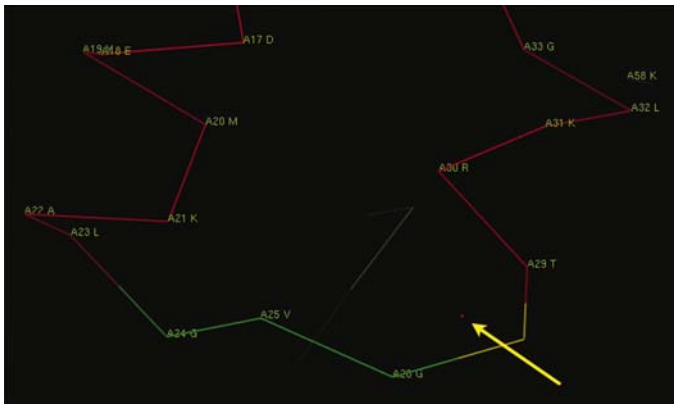


Figure 9
The result of Decor_slide dynamic threading on a Sprout_CAT-derived framework structure. The indicated '*' has been drawn because the threaded sequence and framework differ in gap size. The thread indicates a two-residue gap (residues A27 and A28), but the TRACE framework has only a single residue, coloured yellow (red and green indicate *Yasspa*-derived α-helical and β-strand segments, respectively). In this example, the thread is correct.

fit of the peptide template. The skeleton-following option replaces the older baton-building commands of the early 1990s. Coordinates for the main-chain atoms are generated by a combination of database searching and skeleton tracing. This option makes a search of the main-chain database for a five-residue fragment that matches guide points generated from the previous two residues that have been built and the preliminary placement of the next three residues along the skeleton. The central residue from the database search is then used to generate the next set of main-chain coordinates.

10. The future

I have described just some of the tools in *O* that are available to crystallographers. Many others exist, including the possibility to generate and edit masks, derive NCS operators, carry out averaging, fit a ligand *etc*. The tools described here are aimed primarily at tracing and building a new structure in an electron-density map. They can be used at any resolution and with any quality map. At high resolution, with well phased maps, these tools allow one to trace and build a new structure with little user intervention. At lower resolution and with more poorly phased maps, more is demanded of the crystallographer. The signal in the map can be enhanced to some extent by using SSTs for building, for determining chain directionality and for enhancing the usefulness of automatic tracing commands. So far, I have not developed any tools that work in reciprocal space. This will change, since I believe we spend too much time refining our structures.

As I have plodded along my road map others have roared along theirs and I should mention particularly the work of Oldfield (2001, 2002) and Levitt (2001), who have developed their own excellent interactive building programs. Researchers of course just want to get their structures solved and their papers written. The appeal of automatic tracing and rebuilding is therefore overwhelming. Some of the concepts described above have indeed found their way into such systems. The greatest success so far though has been at resolutions where one can see carbonyl bulges (*e.g.* Perrakis *et al.*, 1999). It has always been true that if you could see the carbonyl O atom you could build a good model at a computer-graphics system. Nowadays, if you can choose a good SST or fragment at the graphics you can build a reasonable model. The automated systems are not far behind in the use of this approach.

I thank my wife, Sherry Mowbray, and our children, Daniel and Elanor, for allowing me to keep my hobby, and Gerard Kleywegt and Morten Kjeldgaard for the fun we have had working together. Carl-Ivar Brändén persuaded me to move to Uppsala. He died this spring; he was a good friend and we miss him dearly. Finally, special thanks to the organizers for inviting me to the meeting and allowing me to write this review.

References

Bricogne, G. (1976). *Acta Cryst.* A**32**, 832–847.
Brünger, A. T. & Krukowski, A. (1990). *Acta Cryst.* A**46**, 585–593.
Cowan, S. W., Newcomer, M. E. & Jones, T. A. (1993). *J. Mol. Biol.* **230**, 1225–1246.
Cowtan, K. (1998). *Acta Cryst.* D**54**, 750–756.
Deisenhofer, J. (1981). *Biochemistry*, **20**, 2361–2370.
Deisenhofer, J. & Steigemann, W. (1975). *Acta Cryst.* B**31**, 238–250.
Diamond, R. (1971). *Acta Cryst.* A**27**, 436–452.
Greer, J. (1974). *J. Mol. Biol.* **82**, 279–301.
Greer, J. (1976). *J. Mol. Biol.* **100**, 427–459.
Hendrickson, W. A. & Konnert, J. H. (1980). *Computing in Crystallography*, edited by R. Diamond, S. Ramaseshan & K. Venkatesan, pp. 13.01–13.25. Bangalore: Indian Academy of Science.
Hermans, J. & McQueen, J. E. (1974). *Acta Cryst.* A**30**, 730–739.
Jones, T. A. (1978). *J. Appl. Cryst.* **11**, 268–272.
Jones, T. A. (1982). *Computational Crystallography*, edited by D. Sayre, pp. 303–317. Oxford: Clarendon Press.
Jones, T. A. (2001). *Methods in Macromolecular Crystallography*, edited by D. Turk & L. Johnson, pp. 142–147. Amsterdam: IOS Press.
Jones, T. A. & Kjeldgaard, M. (1997). *Methods Enzymol.* **277**, 173–208.
Jones, T. A. & Liljas, L. (1984*a*). *J. Mol. Biol.* **177**, 735–767.
Jones, T. A. & Liljas, L. (1984*b*). *Acta Cryst.* A**40**, 50–57.
Jones, T. A. & Thirup, S. (1986). *EMBO J.* **5**, 819–822.
Jones, T. A., Zou, J. Y., Cowan, S. W. & Kjeldgaard, M. (1991). *Acta Cryst.* A**47**, 110–119.
Kleywegt, G. J. (1996). *Acta Cryst.* D**52**, 842–857.
Kleywegt, G. J. & Jones, T. A. (1995). *Structure*, **3**, 535–540.
Kleywegt, G. J. & Jones, T. A. (1996). *Acta Cryst.* D**52**, 829–832.
Kleywegt, G. J. & Jones, T. A. (1997*a*). *Acta Cryst.* D**53**, 179–185.
Kleywegt, G. J. & Jones, T. A. (1997*b*) *Methods Enzymol.* **277**, 208–230.
Kleywegt, G. J. & Jones, T. A. (1997*c*). *Methods Enzymol.* **277**, 525–545.
Levitt, D. G. (2001). *Acta Cryst.* D**57**, 1013–1019.
Liljas, L., Unge, T., Jones, T. A., Fridborg, K., Lovgren, S., Skoglund, U. & Strandberg, B. (1982). *J. Mol. Biol.* **159**, 93–108.
Mowbray, S. L., Helgstrand, C., Sigrell, J. A., Cameron, A. D. & Jones, T. A. (1999). *Acta Cryst.* D**55**, 1309–1319.
Needleman, S. B. & Wunsch, C. D. (1970). *J. Mol. Biol.* **48**, 443–453.

Newcomer, M. E., Jones, T. A., Åqvist, J., Sundelin, J., Eriksson, U., Rask, L. & Peterson, P. A. (1984). *EMBO J.* **3**, 1451–1454.
Oldfield, T. J. (2001). *Acta Cryst.* D**57**, 82–94.
Oldfield, T. J. (2002). *Acta Cryst.* D**58**, 487–493.
Perrakis, A., Morris, R. & Lamzin, V. S. (1999). *Nature Struct. Biol.* **6**, 458–463.
Pflugrath, J. W., Saper, M. A. & Quiocho, F. A. (1984). *Methods and Applications in Crystallographic Computing*, edited by S. Hall & T. Ashida, pp. 404–407. Oxford: Clarendon Press.
Richards, F. M. (1968). *J. Mol. Biol.* **37**, 224–230.
Sedgewick, R. (1983). *Algorithms.* Reading, MA, USA: Addison–Wesley.
Unge, T., Strandberg, B., Vaara, I., Kannan, K. K., Fridborg, K., Nordman, C. E. & Lentz, P. J. Jr (1980). *Nature (London)*, **285**, 373–377.
Wierenga, R. K., Kalk, K. H. & Hol, W. G. J. (1987). *J. Mol. Biol.* **198**, 109–121.
Zou, J. Y. & Jones, T. A. (1996). *Acta Cryst.* D**52**, 833–841.