CCP4 NEWSLETTER ON PROTEIN CRYSTALLOGRAPHY

Number 34. September 1997

An informal Newsletter associated with the BBSRC Collaborative Computational Project No. 4 on Protein Crystallography.

Contents

Editor: Martyn Winn
Daresbury Laboratory, Daresbury,
Warrington, WA4 4AD, UK

CCP4 News

Martyn Winn, Adam Ralph, Alun Ashton and Sue Bailey

Daresbury Laboratory,
Daresbury,
Warrington
WA4 4AD, U.K.
ccp4@dl.ac.uk

Latest release on its way ...

Version 3.3 of the CCP4 suite will be released in the near future. In addition to a number of bug fixes and enhancements to existing programs, we plan to include the following new programs:

1. ARPP: Automated Refinement Program from Victor Lamzin.
   ARPP updates a model by removing poorly defined atoms and adding new atoms, and should be used in conjunction with a refinement program. Note that "arp" has been renamed "arpp" to avoid conflict with a unix command.
2. GETAX from Clemens Vonrhein.
   Real space searching for rotation axis of a D<n> or C<n> multimer, given an initial map and a selfrotation function solution.
3. OMIT from Bauke Dijkstra and Fred Vellieux.
   Calculate omit maps according to procedure of Bhat.
4. REVISE from Yao Jia-xing.
   Program to generate normalised anomalous scattering factor magnitudes from MAD data.
5. wulff.ps added to 'aggregated' stuff, for generating Wulff net.

In addition, there will be a new version of SCALA, which now incorporates the functionality of both ROTAVATA and AGROVATA. Documentation for most of the main programs is now distributed in html format, in addition to the formatted versions. See Alun's article in this newsletter.

Versions of the CCP4 Suite up to 3.2 have used a pseudo-PDB format for holding coordinate data. In version 3.3, this will be updated to the current PDB Version 2.1 standard. The main additions will be the ANISOU record for anisotropic U factors, and an extention to the ATOM/HETATM record to include segment and element IDs at the end of the line.

Study Weekend 1998

The next CCP4 Study Weekend is to be held in Reading on the 9th/10th of January 1998. The topic is to be databases and their use by crystallographers.

Future changes to the CCP4 working coordinate format

The macromolecular Crystallographic Information File (mmCIF) format was developed by a working group of the IUCr formed in 1990. It represents an extension of the CIF format used by small molecule crystallographers, and which is used for automatic submission to Acta Crystallographica C. mmCIF files are text files with a flexible format based around either <data_name> <data_value> pairs or a loop structure (works like a table). In particular, a wide variety of data items are supported (as defined in the mmCIF dictionary), and character data values may be lengthy and descriptive. This alleviates many of the restrictions of the traditional PDB format.

In view of the likely increasing importance of the mmCIF format, CCP4 intend to move an mmCIF-like format as the working format for coordinate data. Conversion programs will be provided to change between this working format and PDB, but the programs will no longer work directly with PDB files. Initially, we intend to use only a small subset of the full mmCIF format, which will mirror the current PDB format. Coordinate data files should not look too dissimilar from PDB files; in particular, the bulk of the file will remain as columns of atom data. As we gain experience with the format, and users become comfortable with it, we will probably increase the subset of mmCIF data items which can be used, thereby using making more use of the power of mmCIF.

The first version of the CCP4 suite to use the mmCIF-like format may appear towards the end of 1998. Releases before then will *not* include the new format. Precise details are still evolving, so watch this space. Background information on the full mmCIF format can be found at http://www.iucr.ac.uk/iucr-top/cif/mmcif/ndb/index.html.

CCP4 Looks and Leaps into the web...

Alun W. Ashton, CCP4 Daresbury
a.w.ashton@dl.ac.uk

As surfing the World Wide Web becomes easier than riding a bike, CCP4 is working hard to embrace the new technology and offer a service that both experienced users and tenderfoot newcomers can use.

As of version 3.3, CCP4 will have its documentation in HTML format to be viewed with any web browser. Although you will use the documentation installed on your own workstations it is also recommended that you occasionally visit the CCP4 WWW home pages. You will open your local copy of the documentation either by using the new command ccp4help, defined in the ccp4.setup file, or by using a File/Open File option from your browser and opening the file INDEX.html in the $CCP4/html/ directory. The reasons for using your local copy of the documentation is three fold:

1) Speed.
   Accessing your local files is still a lot faster than accessing remote files.
2) Version compatibility.
   The documentation you will have on your server will be specific to the programs you are running.
3) Browser/Server Problems.
   Because many servers assume that a text document is written in html, actual text documents will appear as one long string. This is not a problem with the program documentation (or files with the .txt extension) but the documentation is slowly being changed to include (instead of examples) links to actual runnable examples in $CEXAM. If you find that example script opened as files by your web browser appear as garbage you should check your browsers preferences to ensure it defaults to text files and not html files. It is because of this problem that all old example scripts with the .sh and .csh extensions have been changed to .exam because browsers such as Netscape would not easily allow you to view these files as formatted text files.

The CCP4 home pages contain information about the electronic mailing list, a reference list to some of the programs and information about the ftp server here at Daresbury as well as our mirror sites in the USA and Japan. The site also has a suggestion box where you can voice your concerns or suggestions to CCP4 in private! The most important thing to check is the problems page. If anyone finds an undesirable feature in any of the CCP4 programs, and we think you should know about it then this is where you will find the information.

A recent use of our web site is in archive retrieval. You can already access the most recent issues of this newsletter and now, new for 1997, you will soon be able to view the Study Weekend Proceedings in HTML. The availability of the proceedings on the web will coincide with their usual distribution. I would like to thank all the contributors to the 1997 Proceedings for their help in in making the proceedings available in this new media. If any authors of past releases of the CCP4 Study weekend proceedings still have an electronic copy of their manuscript and would like to see their contributions available on the web, sooner rather than later, please contact me! Although every possible care has been taken to ensure the accurate conversion to the HTML versions of the Proceedings I cannot guarantee that they will be 100% correct. As anyone who has ever tried to display a page containing greek character on the web will tell you, it does not always go to plan and some details may have been corrupted in the conversion process. Please let me know it you find anything which does not seem quite right! That is one good thing about displaying thing in this way - they can be corrected.

So can we see the day that the web pages will have pictures in documentations, mpg movie files with sound tracks instead of example files, JAVA scripts replacing the suite so there would be no platform dependencies and always the newest version....... probably not yet, but as the next generation of programmers enter the field there will surely be changes.

NEWS FROM THE UPPSALA SOFTWARE FACTORY - 8

Les amis d'O

Gerard J. Kleywegt
Department of Molecular Biology
Biomedical Centre, Uppsala University
Uppsala - Sweden

In this article we shall take a closer look at three of the lesser-known utility programs from Uppsala that work in conjunction with **O [1]**.

SOD

SOD **[2]** stands for "Sequences to **O** Datablocks", *i.e.* it is a program that converts sequences into information that can be used in or by **O**. The program can read individual and aligned sequences in a number of formats. It can be used to do the following:

- to generate an **O** datablock of the sequence of your protein (task INIT). This is a quick way to get this information into **O** when you are about to assign the sequence to your Ca trace. The datablock produced by SOD can be used with the *sam_init_db* command in **O** to initialise the data structures for a new protein molecule.
- to generate **O** macros with which one can quickly build a homology model or, more interestingly, a molecular replacement search model (task HOMO). In order to do this, SOD requires two aligned sequences, one of a protein whose structure is known (and available - which is not always the same, unfortunately), and one of a related protein for which you want to generate a model. SOD will use the aligned sequences to generate an **O** macro which contains mostly *Mutate* instructions (*mutate_insert* , *mutate_delete* , and *mutate_replace*). In the case of homology modelling, all residues that differ in the two sequences will be replaced by the residue type of the protein for which one wants to build a model, and deletions and insertions are included (although they will have to be modelled by the user). If one generates a molecular replacement search probe, on the other hand, residues that differ between the two sequences will be replaced by alanines, deletions are carried out, but insertions will not be made. (Other tools for generating molecular replacement search models were discussed in a previous episode in this series, available at URL: http://alpha2.bmc.uu.se/~gerard/manuals/factory_6.html.)
- to do a pairwise comparison of one sequence with one or more others (task PAIR). For each comparison an **O** datablock will be generated which contains an integer code for every residue: 0 = conserved residue, 1 = mutation, 2 = insertion in other sequence, 3 = deletion in other sequence, 4 = outside other sequence. This datablock can be used to colour the molecule, *e.g.* using the *paint_case* command; a Ca-trace will then reveal where mutations, insertions and deletions occur in the 3D structure.

- to analyse multiple aligned sequences (task MULT). This option is useful to present information regarding sequence conservation in a large family of related proteins (provided the structure of one of them is available). The following **O** datablocks will be produced (assuming that the molecule is called "M1" in **O**):
    - - M1_RESIDUE_POSSIBLE - listing all residue types encountered for every residue;
    - - M1_RESIDUE_CONSERVED - degree of conservation (%) of each residue type in the sequence;
    - - M1_RESIDUE_VARIATION - a count of the number of different residue types observed at each position;
    - - .ID_SOD - a temporary *.id_template* showing all of the above properties when you click on an atom;
    - - @M1_SOD - a macro to produce three objects from your molecule: CONS (Ca-trace colour-ramped by M1_RESIDUE_CONSERVED), VARI (Ca-trace colour-ramped by M1_RESIDUE_VARIATION), and GRAD (Ca-trace coloured in steps according to M1_RESIDUE_CONSERVED).

  Simply reading the datablock file into **O** and executing the macro will produce the three graphics objects.

ODBMAN

One of the useful features in **O** is its use of datablocks (of type real, integer, character or text) to represent information pertaining to a molecule as a whole, to each of its residues, or to each of its atoms **[3]** . Although **O** contains a number of commands to manipulate datablocks, a separate utility program (ODBMAN, for "**O** DataBlock MANipulation" **[4]**) is also available. Its options (besides trivial I/O-related ones) fall into the following categories:

- extracting information from other sources (EXtract commands). With these commands, information can be extracted from formatted files and stored as real or integer datablocks. The input can either be formatted, or field oriented (*i.e.* , containing fields separated by tabs or spaces). A separate option is available to extract information from a ProCheck **[5]** output file (residue type and name, secondary structure assignment according to the DSSP algorithm, area of the Ramachandran plot in which each residue resides, the number of bad contacts for each residue, and its H-bond energy).
- manipulating individual entries of a datablock (SEt commands). This includes options to set all entries of a datablock to a particular value, to set a consecutive stretch of entries to a particular value, to set a consecutive stretch of entries individually, and to "translate" information from other datablocks (*e.g.* , to generate an integer datablock representing secondary structure from a character datablock). Using these options, it is not too difficult to colour a protein structure in the colours of the Dutch flag, for instance.

- manipulating entire datablocks. This include options to do simple arithmetic on integer and real datablocks, to smoothen datablocks, and to modify character datablocks.
- analysis of datablocks. Options are available to list some statistics and to produce histograms of individual datablocks, and to produce line plots and scatter plots of individual datablocks or of one datablock *versus* another.

O2D

Many of the Uppsala programs (*e.g.* , ODBMAN, MOLEMAN2, LSQMAN, DATAMAN, MAPMAN) produce (ASCII) plot files in a meta-format. O2D **[6]** is a simple program to convert such plot files into other formats. Usually, this program will be used to convert plot files into PostScript files, but the program can also produce tab-delimited ASCII files, which can be read by most popular spreadsheet and graphing programs on the market, and hence used to produce more professional-looking graphs. The SGI version of this program in addition allows the user to plot data interactively in graphics windows. O2D can produce line and scatter plots, histograms and simple pie charts of 1D data, and contour plots of 2D data. In interactive mode, there are also simple facilities for integrating curves or contour plots, and to manipulate the display. The (ASCII) meta-format for both 1D and 2D plots is simple, using six-character keywords (see the manual for details). There is also a C-shell script available which will do a batch conversion of many plot files to PostScript.

AVAILABILITY

SOD, ODBMAN, and O2D are part of the X-UTIL package, which is available free of charge to academic users from *ftp://alpha2.bmc.uu.se/pub/gerard/xutil/*. Commercial users may contact GJK for more information (*mailto:gerard@xray.bmc.uu.se*). For more information about **O** , contact Alwyn Jones (*mailto:alwyn@xray.bmc.uu.se*). The O WWW site is at *http://imsb.au.dk/~mok/o/* , and the Uppsala Software Factory can be found at *http://alpha2.bmc.uu.se/~gerard/manuals/* .

REFERENCES

- **[1]** Jones, T.A., Zou, J.Y., Cowan, S.W. and Kjeldgaard, M. (1991). Improved methods for building protein models in electron density maps and the location of errors in these models. *Acta Crystallogr.* **A47** , 110-119.
- **[2]** The manual for this program is available at URL: *http://alpha2.bmc.uu.se/~gerard/manuals/sod_man.html*
- **[3]** Jones, T.A. and Kjeldgaard, M. (1997). Electron density map interpretation. *Meth. Enzymol.* **277** , in press.
- **[4]** The manual for this program is available at URL: *http://alpha2.bmc.uu.se/~gerard/manuals/odbman_man.html*
- **[5]** Laskowski, R.A., MacArthur, M.W., Moss, D.S. and Thornton, J.M. (1993). PROCHECK: a program to check the stereochemical quality of protein structures. *J. Appl. Cryst.* **26** , 283-291.
- **[6]** The manual for this program is available at URL: *http://alpha2.bmc.uu.se/~gerard/manuals/o2d_man.html*

Bulk Solvent Correction: Practical Application and Effects in Reciprocal and Real Space

Dirk Kostrewa

Pharmaceutical Research - New Technologies, F.Hoffmann-La Roche Ltd., CH-4070 Basle, Switzerland

e-mail: dirk.kostrewa@roche.com

I. Introduction

Protein crystals contain between ~30% and ~70% solvent **[1]**, most of which is disordered in the solvent channels between the protein molecules of the crystal lattice (this disordered solvent will here be denoted as bulk solvent). Thus, the electron densities of the protein molecules, with typical values of 0.43 e/A^3, are surrounded by a continuous bulk solvent electron density ranging from 0.33 e/A^3 for pure water to 0.41 e/A^3 for 4M ammonium sulphate (Figure 1).

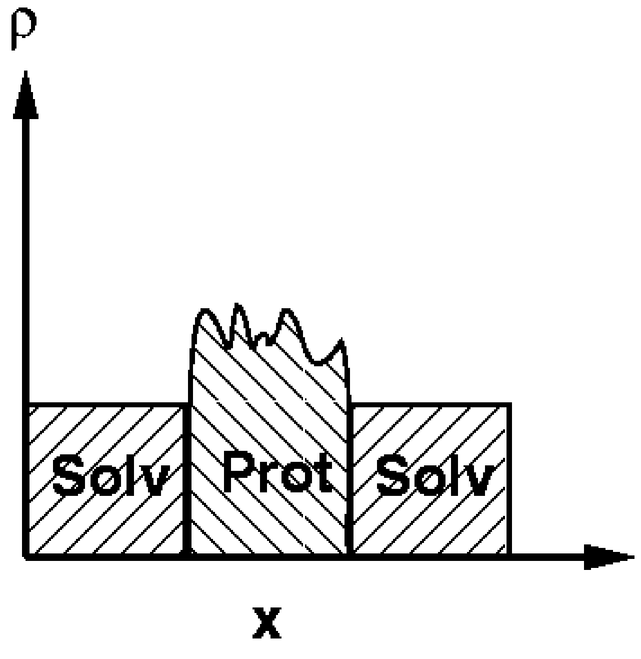


Figure 1 Schematic picture of the protein electron density in a crystal ("Prot"), surrounded by a continuous bulk solvent electron density ("Solv").

If no model for this continuous bulk solvent electron density is taken into consideration, atomic protein models are artificially placed in a "vacuum" environment, leading to a vast overestimation of the electron density contrast at the protein surface. This in turn leads to calculated structure factor amplitudes which are systematically much larger than the observed structure factor amplitudes at resolutions below ~5A (Figure 2; all example calculations were made with the structure and data of *Eco*RV complexed with cognate DNA which was crystallised from a low salt buffer, PDB entry code 1rva **[2]**; all scale factors were calculated for the resolution range 5.0-2.0A and applied to the whole resolution range 20.0-2.0A; all analyses were made in 100 equal volume shells with ~333 reflections per shell).

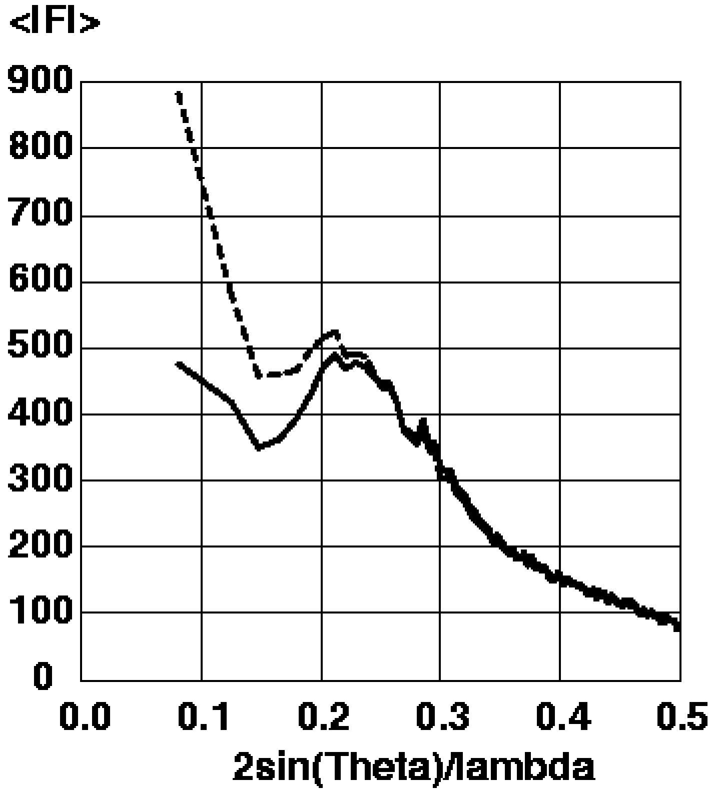


Figure 2 Magnitude of observed structure factor amplitudes (solid line) and calculated structure factor amplitudes without a bulk solvent correction (dashed line) on an arbitrary scale against resolution.

This systematic deviation between observed and calculated structure factor amplitudes leads to severe problems in scaling, in least-squares refinement with its assumption of Gaussian error distributions, and in electron density difference map calculations. In the past, it was common practice to cirumvent these problems by cutting the data at a lower resolution of, say, 6A. However, doing so creates distortions of the local electron density contrast in the protein region (for an optical example, see Kevin Cowtan's duck **[3]**). A better solution is to include an appropriate model for the bulk solvent, thus allowing the use of all data during scaling, refinement, and electron density difference map calculations. The currently available bulk solvent models will be discussed.

II. Available Bulk Solvent Models

Some definitions: F_{Prot}, calculated structure factor of the atomic protein model; F_{Solv}, calculated structure factor of the bulk solvent model; F_{Total}, calculated structure factor of the atomic protein model including a bulk solvent model.

1). The exponential scaling model

A simple bulk solvent model is the downscaling of the calculated structure factor amplitudes at low resolution to yield the total structure factor amplitude according to formula (1) **[4]**

$$|\vec{F}_{Total}|=|\vec{F}_{Prot}|-k_{sol}*|\vec{F}_{Prot}|*e^{-B_{sol}*\sin^2(\Theta)/\lambda^2} \quad (1)$$

This formula is an application of Babinet's principle, (i.e. the structure factors of a mask have the same amplitudes but opposite phases as the structure factors of the complementary mask). In other words, it is assumed that the structure factors of the bulk solvent electron density are directly proportional to the structure factors of the protein electron density with strictly opposite phases. However, this approximation is only true at resolutions lower than ~15A, using observed protein phases to estimate phase differences **[5]**.

Typical values for the two scaling parameters are ksol=0.75-0.95, where ksol reflects the ratio of the solvent electron density to the protein electron density (see Introduction), and Bsol=150A^2-250A^2, which restricts the downscaling basically to resolutions below ~ 5A. Because of its simple form, this exponential scaling bulk solvent model is implemented in most of the crystallographic refinement programs, namely REFMAC, RESTRAIN, SHELXL-93/97, TNT and BUSTER (for an overview of macromolecular refinement programs see **[6]**).

2). The mask model

In the mask bulk solvent model, the protein molecules are placed on a grid in the unit cell, and all grid points outside the protein region are filled with bulk solvent electron density. The protein boundary is determined by the sum of the atomic van-der-Waals radii and a solvent probe radius, SOLRAD. This creates a "vacuum" gap of width SOLRAD between the van-der-Waals surface of the protein and the border of the bulk solvent mask. This gap is then filled by extending the bulk solvent mask with a radius, SHRINK. The combination of the two radii SOLRAD and SHRINK produces a bulk solvent mask in close contact to the van-der-Waals surface of the protein, leaving tiny internal holes and channels "empty". The calculated structure factors of the bulk solvent electron density, scaled with a factor ksol and smoothed with a B-factor Bsol, are then vectorally added to the calculated structure factors of the protein to give the total structure factors according to formula (2)

$$\vec{F}_{Total}=\vec{F}_{Prot} + k_{sol}*\vec{F}_{Solv}*e^{-B_{sol}*\sin^2(\Theta)/\lambda^2} \quad (2)$$

This mask bulk solvent model is implemented in X-Plor 3.851 **[7]**. The two scaling parameters ksol and Bsol are determined in a least squares refinement of the total model structure factor amplitudes against the observed structure factor amplitudes in two resolution ranges. No assumption is made about the phase relationship between the protein structure factors and the bulk solvent structure factors. The recommended values of SOLRAD and SHRINK are 1.0A and 1.1A, respectively **[7]**. However, one should check the volume of solvent grid points (with MASK=1) given in the output file: best results are obtained, if this volume is ~2-4% smaller than the calculated solvent volume of the crystal **[1]** (for a complete protein model including ordered solvent molecules). If this is not the case, one can try to set SOLRAD to a value equal to or slightly bigger than the sampling grid size, and SHRINK to a value between SOLRAD and SOLRAD+0.2A. This parameter combination can produce slightly better results than SOLRAD=1.0A and SHRINK=1.1A, but it requires some additional time for testing. (The mask calculation appears to be completely insensitve to SOLRAD/SHRINK values smaller than the sampling grid size. Thus, the recommended parameter set for X-Plor 3.1 with SOLRAD=0.25A, ksol equal to the electron density of the crystallisation buffer and Bsol=50A^2 is not applicable to X-Plor 3.851.) Typical refined values for the two scaling parameters are ksol=0.3-0.4, reflecting the electron density of the crystallisation buffer, and Bsol=15A^2-40A^2, which produces a rather steep fall-off of the bulk solvent electron density with minimum overlap with the protein electron density.

III. Effects in Reciprocal Space

Figure 3 shows a comparison of observed and calculated structure factor amplitudes without a bulk solvent model and with the two available bulk solvent models.

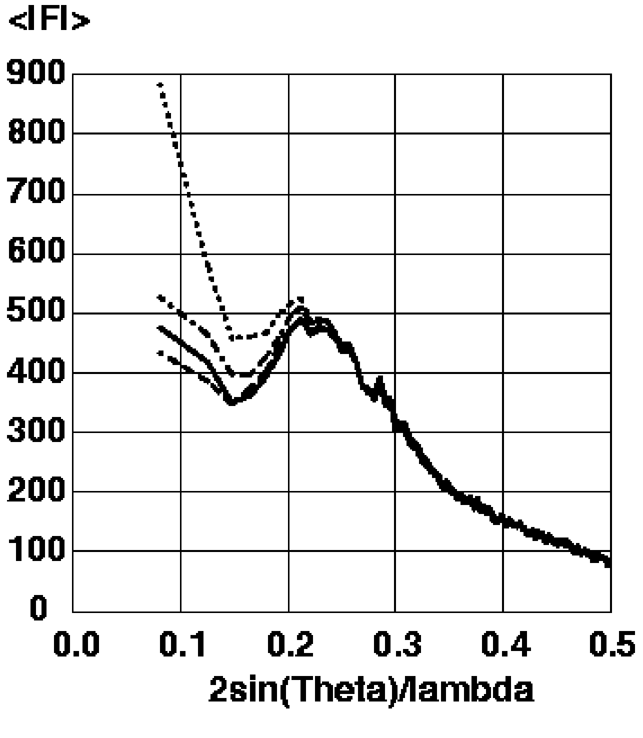


Figure 3 Magnitudes of the observed structure factor amplitudes (solid line), of the calculated structure factor amplitudes from the protein model without a bulk solvent correction (dotted line), of the calculated structure factor amplitudes from the protein model with the exponential scaling bulk solvent correction using ksol=0.74 and Bsol=189.5A^2 (dashed line), of the calculated structure factor amplitudes from the protein model with the mask bulk solvent correction using SOLRAD=1.0A, SHRINK=1.1A, ksol=0.31 and Bsol=35.2A^2 (dashed-dotted line). All amplitudes are put on an arbitrary scale.

The systematic deviation of the calculated protein structure factor amplitudes without a bulk solvent correction from the observed structure factor amplitudes is clearly visible. At the lowest resolution shell (20.0-9.0A) the calculated structure factor amplitudes are twice as large as the observed strcuture factor amplitudes. Much better correspondence between calculated and observed structure factor amplitudes can be achieved by the application of a suitable bulk solvent correction. The exponential scaling model seems to underestimate the contrast between protein and bulk solvent electron density, probably because of its intrinsic assumption of strictly opposite phases. The mask model seems to overestimate the contrast between protein and bulk solvent electron density, probably because all cavities with radii smaller than SOLRAD are left "empty".

A comparison of the bulk solvent structure factor amplitudes is shown in Figure 4.

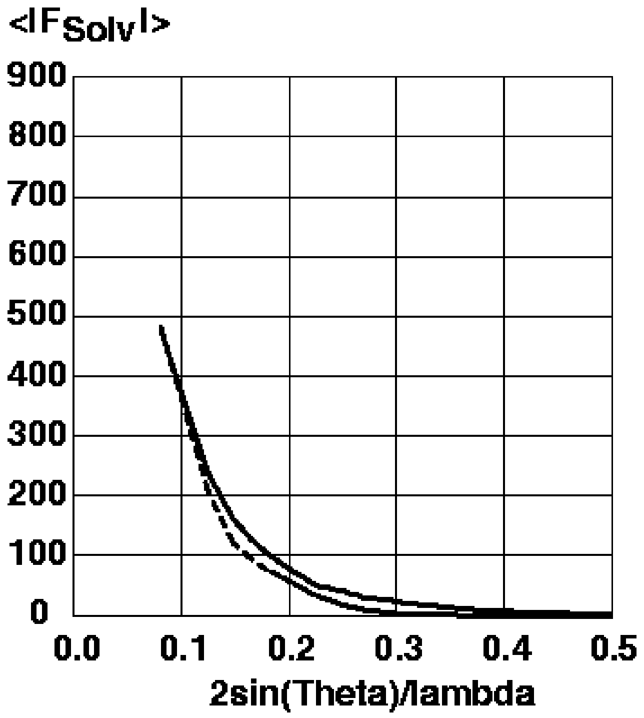


Figure 4 Magnitude of the structure factor amplitudes of the mask bulk solvent model using SOLRAD=1.0A, SHRINK=1.1A, ksol=0.31 and Bsol=35.2A^2 (solid line) and of the exponential scaling bulk solvent model (righthand term of formula (1)) using ksol=0.74 and Bsol=189.5A^2 (dashed line). The structure factor amplitudes are on the same arbitrary scale as in Figure 3.

Both bulk solvent models show structure factor amplitudes at low resolution half as large as the calculated structure factor amplitudes of the protein without a bulk solvent correction. This leads to the desired correction if the phases of the bulk solvent and the protein are opposite. In the exponential scaling bulk solvent model, this phase relationship is an intrinsic assumption. For the mask bulk solvent model the phase relationship is shown in Figure 5.

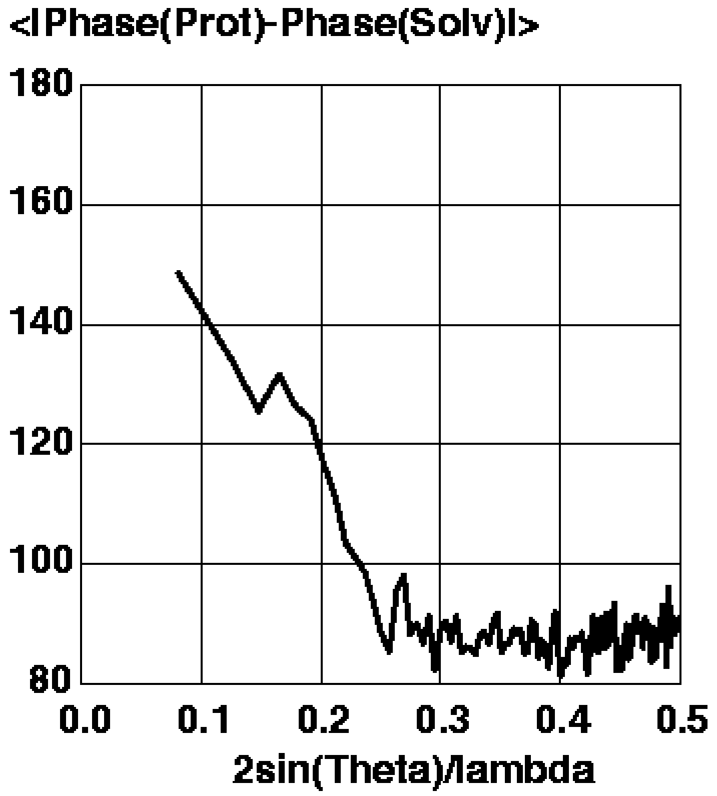


Figure 5 Phase differences between the calculated structure factors of the atomic protein model and of the mask bulk solvent model.

The phases of the atomic protein model and of the mask bulk solvent model are approximately opposite at low resolution. This phase relationship is completely lost at resolutions higher than ~4A. The calculated phase differences are very similar to the estimation of phase differences using observed protein phases **[5]**. The resulting phases of the protein model with the mask bulk solvent correction differ to the phases of the protein model without a bulk solvent correction between ~5deg at 4A to ~30deg in the lowest resolution shell, 20.0-9.0A (data not shown). The mask bulksolvent model appears to be a more realistic description of the true bulk solvent electron density than the exponential scaling bulk solvent model. This is also reflected in the R-factor as shown in Figure 6.

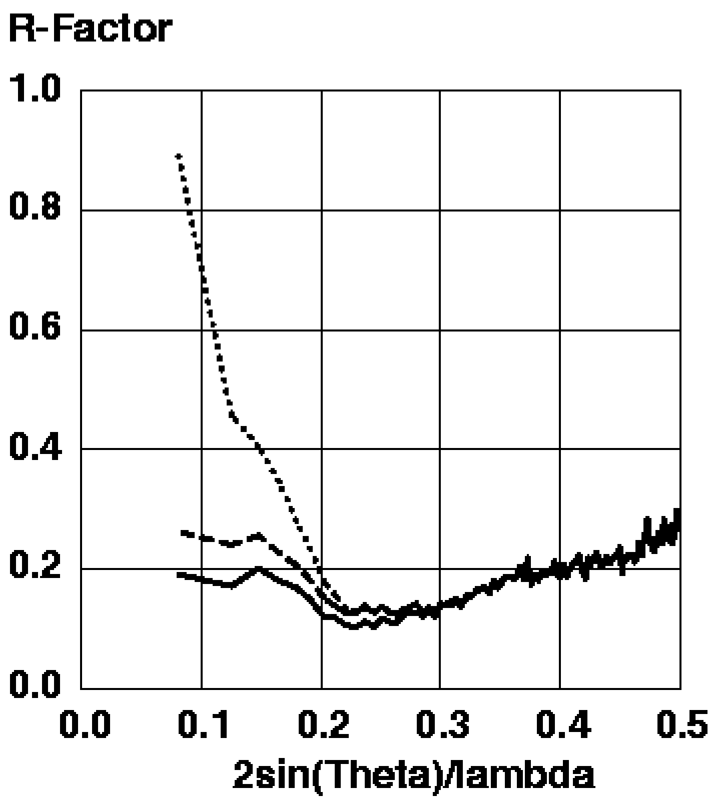


Figure 6 R-factors for the protein model without a bulk solvent correction (dotted line), for the protein model with the exponential scaling bulk solvent correction (dashed line), and for the protein model with the mask bulk solvent correction (solid line).

The mask bulk solvent correction gives a clearly better approximation to the observed structure factor amplitudes than the exponential scaling bulk solvent correction, producing R-factors which are ~5-7% lower in the low resolution range.

IV. Effects in Real Space

To give a realistic picture of the effect of the bulk solvent correction on electron density difference maps a simulated-annealing omit refinement **[8]** was done for the following amino acids (average B-factors in brackets): Tyr138 (23.5A^2), Thr139 (27.8A^2), Arg140 (44.6A^2), Val141 (39.8A^2), Ala142 (53.5A^2), and Thr143 (64.0A^2). Sigma-a weighted **[9]** omit maps with Fourier coefficients mFo-DFc were calculated without a bulk solvent correction using either data etween 6.0-2.0A or all data, with the exponential scaling bulk solvent correction using all data, and with the mask bulk solvent correction using all data. For comparison, a sigma-a weighted map with Fourier coefficients 2mFo-DFc was calculated for the same (non-omitted) amino acids of the refined structure using the mask bulk solvent correction and all data. The results are shown in Figures 7a-e. **Caution**: Omitted parts of the structure must be excluded from the XREFIN term prior to the bulk solvent mask calculation. If this is not done, a hole with the shape of the omitted parts will be cut in the bulk solvent mask, appearing as an artificial positive electron density difference map!

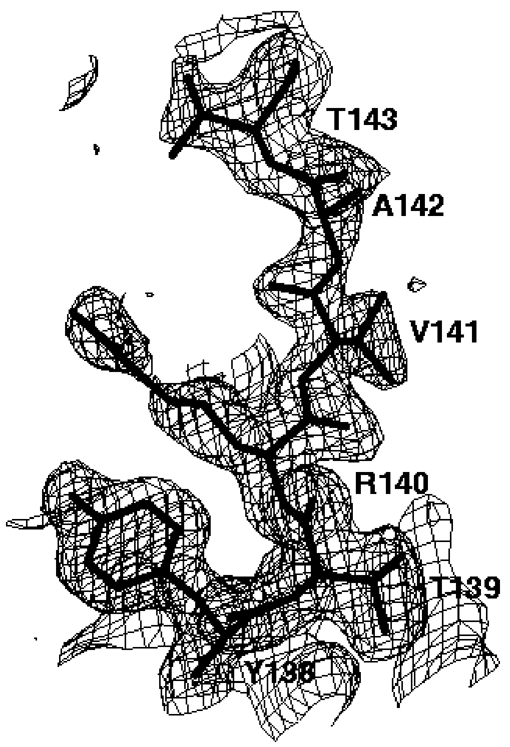


Figure 7a 2mFo-DFc electron density map of the refined structure, contoured at 0.3e/A^3. The amino acids 138-143 were not omitted. The mask bulk solvent correction was applied using all data.

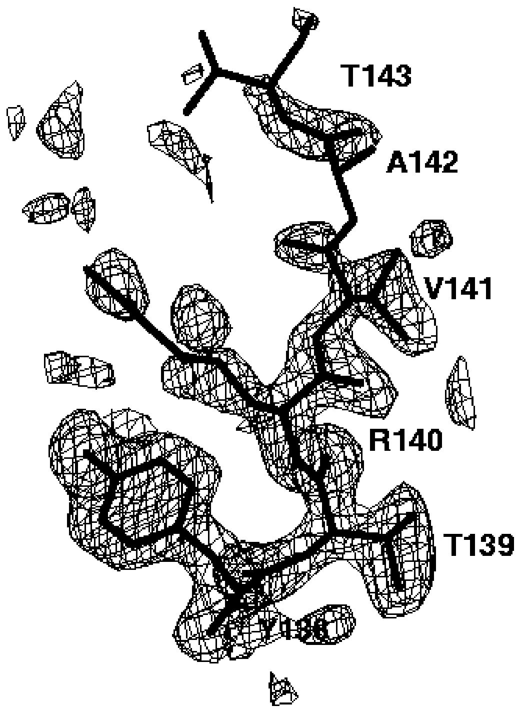


Figure 7b mFo-DFc electron density simulated-annealing omit map using data between 6.0-2.0A, contoured at 0.15e/A^3. No bulk solvent correction was applied.

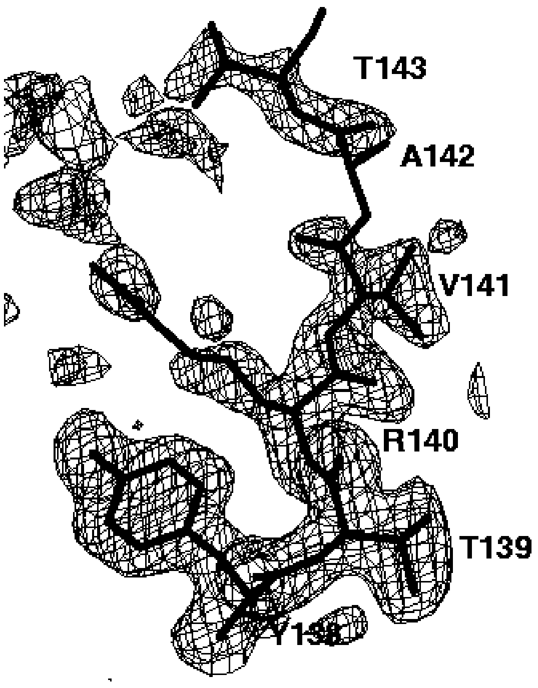


Figure 7c mFo-DFc electron density simulated-annealing omit map using all data, contoured at 0.15e/A^3. No bulk solvent correction was applied.

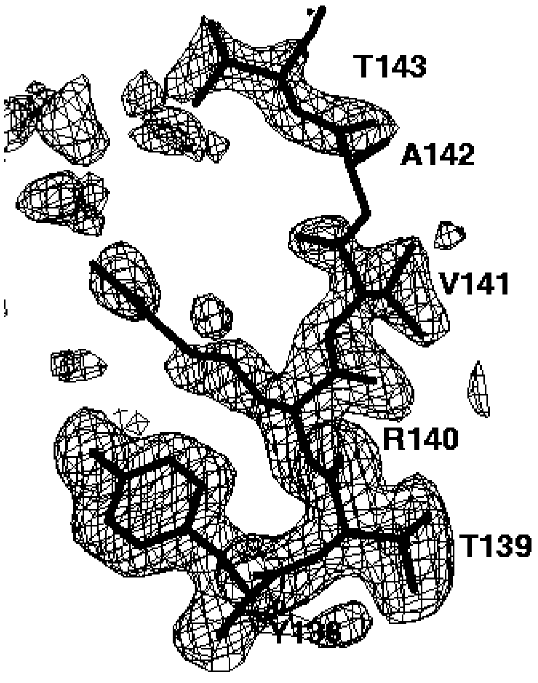


Figure 7d mFo-DFc electron density simulated-annealing omit map using all data, contoured at 0.15e/A^3. The exponential scaling bulk solvent correction was applied.

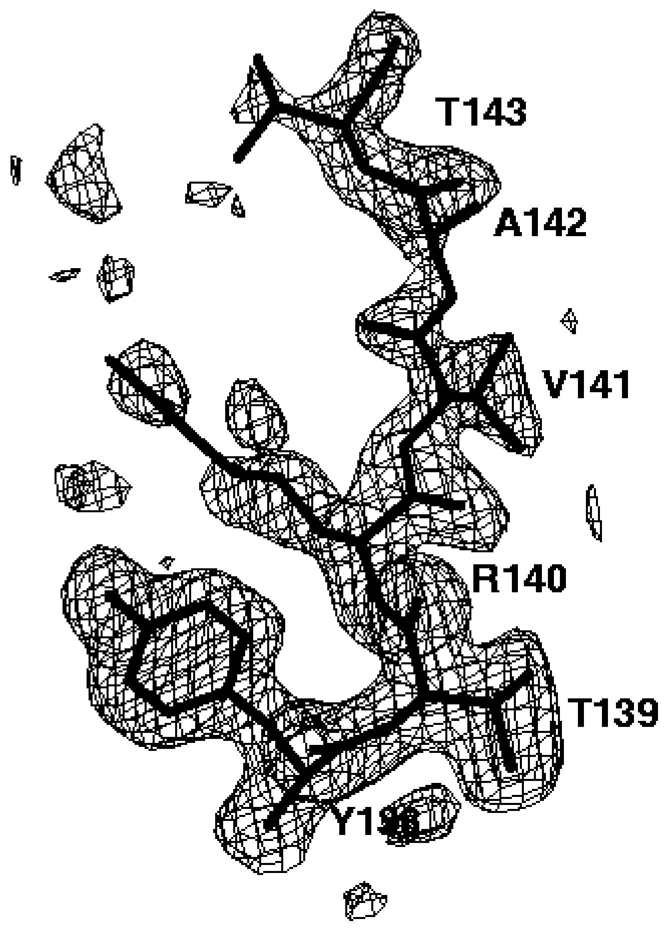


Figure 7e mFo-DFc electron density simulated-annealing omit map using all data, contoured at 0.15e/A^3. The mask bulk solvent correction was applied.

The relatively well ordered amino acids Tyr138, Thr139, Arg140 and Val141 are clearly visible in all omit maps. However, the omit maps of the poorly ordered amino acids Ala142 and Thr143 show different interpretabilities. The worst omit map is the one without a bulk solvent correction cutting all data below 6A (Figure 7b). Here, only a long peak for the main chain peptide group between Ala142 and Thr 143 is visible. This peak is not very interpretable. Lowering the contour level does not help a lot: it makes more of the missing electron density visible but also more false noise peaks appear. If the low resolution data are included (Figure 7c), the electron density of Thr143 becomes visible, but that of Ala 142 is still missing. In addition, many false peaks (on the left side of Thr143) appear, which spoil to some extend the interpretability. A slightly clearer map is obtained, if the exponential scaling bulk solvent correction is applied (Figure 7d). Still, there are a lot of false peaks visible and almost no electron density for Ala142 is visible. By far the best electron density, both with respect to main chain and side chain electron density and with respect to the absence of false peaks is obtained if the mask bulk solvent correction is applied (Figure 7e). The electron density map is now clearly interpretable and almost undistinguishable to the 2mFo-DFc map of the refined structure (Figure 7a).

V. Conclusion and Some Critical Aspects

1). Conclusion

The mask bulk solvent correction as implemented in X-Plor is the best of the currently available bulk solvent corrections, allowing the use of all data to the low resolution limit in scaling, refinement, and electron density difference map calculations. The most important benefit is the enhanced signal-to-noise ratio of electron density difference maps for missing parts of the model (for instance, soaked inhibitors, partial SIR/MIR models, partial molecular replacement models). These missing parts fall into regions filled with bulk solvent electron density which should theoretically reduce the local contrast of their electron densities. However, the overall positive effects of the bulk solvent correction on the scaling and thus on the calculation of electron density difference maps appears to be overwhelming. It might even well be, that filling the regions of missing parts with a continous electron density is a better approximation to the the true electron density than leaving a "vacuum", thus producing a better approximation to the true phases.

2). Overfitting ?

Is the application of a bulk solvent correction overfitting or "R-factor cosmetic"? For the exponential scaling bulk solvent model the picture is simple: There are only two adjustable parameters in this model compared to between several hundred to a few thousand observed reflections in the low resolution range up to ~5A. The observable-to-parameter ratio excludes the possibility of overfitting in this model. For the mask bulk solvent model, the picture is more complicated: Its boundary is determined by the positons and van-der-Waals radii of the surface protein atoms and the four adjustable parameters SOLRAD, SHRINK, ksol, and Bsol. Although the surface protein atoms are taken as observables it is not clear, how many parameters are really needed to desribe this rather complicated bulk solvent boundary. Practically, the question of overfitting was assessed by the complete Free-R method **[7]**. There, it was clearly demonstrated that the mask bulk solvent correction is a good description of the low resolution data without any sign of overfitting.

2). Better Models ?

The mask bulk solvent model is already a good model. One possible problem is that the bulk solvent structure factors are calculated from a step function. Fourier transformations of step functions have pronounced high resolution features which lead to aliasing problems if the sampling is too coarse. Given the usual 3x sampling and the relatively low smoothing B-factor, one might expect such aliasing distortions at higher resolution. A finer sampling, say 10x sampling, is prohibitive because of the rapid increase of computation time to calculate the Fourier transformation of the mask. Two solutions to this problem may be possible:

- the smoothing of the boundary of the bulk solvent electron density prior to the Fourier transformation with a suitable function (for instance, exponential function, hyperbolic tangent)

- theoretically, the mask could be put on a very fine sampling grid and the Fourier transformation could be calculated over the mask surface only **[10]**, because the whole information contents of a mask is in its boundary and not in the featureless uniform regions of protein or bulk solvent. However, to my knowledge no one has ever tried to put this idea into an computer program.

The current mask bulk solvent model assumes a flat uniform electron density distribution in the solvent channels. However, the distribution of water molecules is dependent on the type of surface atoms, leading to a pronounced first shell of higher electron density **[7]**. With the increasing accuracy of observed phases (cryo-cooling, synchrotron data, MAD phasing, better programs, such as SHARP, SOLOMON) better non-uniform bulk solvent models can be generated. Whether the additional accuracy of these models are worth the computational effort remains to be seen.

VI. References

[1] Matthews, B.W. (1968). "Solvent Content of Protein Crystals." *J. Mol. Biol.* **33**, 491-497

[2] Kostrewa, D., & Winkler, F.K. (1995). "Mg2+ Binding to the Active Site of *Eco*RV Endonuclease: A Crystallographic Study of Complexes with Substrate and Product DNA at 2A Resolution." *Biochemistry* **34**, 683-696

[3] Kevin Cowtan's World Wide Web Page:
http://www.yorvic.york.ac.uk/~cowtan/fourier/fourier.html

[4] Moews, P.C., & Kretsinger, R.H. (1975). "Refinement of the Structure of Carp Muscle Calcium-binding Parvalbumin by Model Building and Difference Fourier Analysis." *J. Mol. Biol.* **91**, 201-228

[5] Urzhumtsev, A.G., & Podjarny, A.D. (1995). " On the Problem of Solvent Modelling in Macromolecular Crystals using Diffraction Data: 1. The Low Resolution Range." *CCP4 Newsletter* **31**, 12-16

[6] Dodson, E., Moore, M., Ralph, A., & Bailey, S. (Eds.) (1996). "Macromolecular Refinement." *Proceedings of the CCP4*, **DL-CONF-96-001**

[7] Jiang, J.-S., & Bruenger, A.T. (1994). "Protein Hydration Observed by X-ray Diffraction." *J. Mol. Biol.* **243**, 100-115

[8] Hodel, A, Kim, S.-H., & Bruenger, A.T. (1992). "Model Bias in Macromolecular Crystal Structures." *Acta Cryst.* **A48**, 851-858

[9] Read, J.R. (1986). "Improved Fourier Coefficients for Maps using Phases from Partial Structures with Errors." *Acta Cryst.* **A42**, 140-149

[10] Bricogne, G. (1993). "1.3.3.3.2.5. Molecular Envelope Transformation *via* Green's Theorem.", *Int. Tables for Crsytallography*, **Volume B**, 84

The CCP4 Graphical User Interface

Liz Potterton

Chemistry Department, University of York, York Y01 4DD

lizp@yorvic.york.ac.uk

Introduction

I have been working for CCP4 since the beginning of 1997 on a new graphical user interface which should simplify running CCP4 programs and provide tools for reviewing and analysing results. There have been previous attempts at developing a user interface but none have prooved popular mostly because users found them too complicated. So we are very aware of the need to keep it simple this time. A Working Group II meeting last year produced guidelines for what was required:

- based on free, distributable software
- easy to port and maintain
- possible for someone other than the original developer to modify and extend
- user not 'locked in' to using the interface
- simple interfaces, everything in one place and not too many windows
- does not need to be comprehensive, consider the needs of a novice user first

Most of the work so far has been on an interface to run the programs - tools for reviewing and analysing results will come later.

Why Tcl/Tk?

A major issue in designing the user interface was choosing a language to work in. The ubiquity and familiarity of web browsers made some combination of HTML/Java/Javascript attractive but there seems to be two drawbacks with this system:

The client side of the system can not straightforwardly access disk so it can not read files or even list files which a file browser would need to do. We want the user interface to access files, particularly MTZ files to extract the names of column labels and information such as space group and resolution range.

It is difficult to get 'intelligently dynamic'web pages (unintelligently dynamic pages are another matter!) - that is pages which are customised in response to user input. Given the complexity of input to many CCP4 programs it is desirable to customise the interface after the user has made a key decision so they only see relevant parts of the interface with reasonable preset defaults.

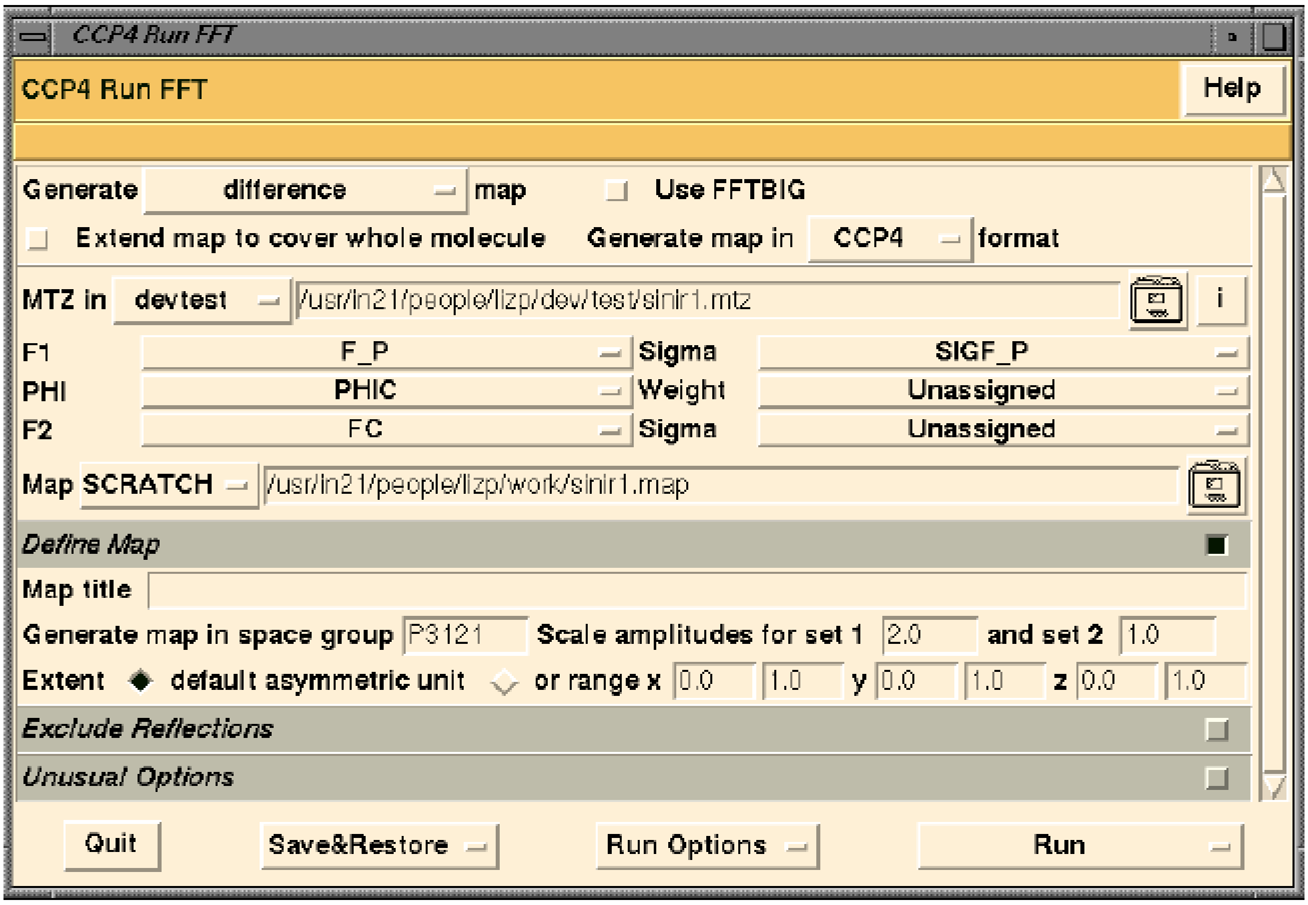
These problems may not be unsurmountable but Tcl/Tk seemed a far more attractive option as it provides all the tools we need and is straightforward to use. Tcl is a scripting language and Tk is a graphics toolbox which includes tools for 2D graph drawing which will be useful in future. The package is free and runs on many systems including UNIX, LINUX,VMS, Windows and Macs. Another advantage is that the RasMol program developed by Roger Sayle, now at Glaxo, has a Tk interface which will make it possible to set up communication between RasMol and any Tk programs.

Running CCP4 Programs - Task Windows

A task window is one window which provides the interface to, usually, one CCP4 program. Where appropriate more then one program may be accessed from one task window, for example the ***FFT*** interface to generate maps can also run the ***Extend*** program to extend a map and also jiffy programs to convert the map format.

Everything to run one task is in one window; this could quickly lead to information overload for the user but not all of the window is always visible. The window is divided vertically into folders and within each folder are related parameters. The folders containing the less popular parameters are by default closed so the contents are not visible. The user can toggle folders open and closed.

The task window for ***FFT*** is shown below.


CCP4 Run FFT

Generate difference map   Use FFTBIG
Extend map to cover whole molecule   Generate map in CCP4 format
MTZ in devtest /usr/ln21/people/lizp/dev/test/slnlr1.mtz
F1 F_P   Sigma SIGF_P
PHI PHIC   Weight Unassigned
F2 FC   Sigma Unassigned
Map SCRATCH /usr/ln21/people/lizp/work/slnlr1.map
Define Map
Map title
Generate map in space group P3121   Scale amplitudes for set 1 2.0   and set 2 1.0
Extent ◆ default asymmetric unit ◇ or range x 0.0 1.0 y 0.0 1.0 z 0.0 1.0
Exclude Reflections
Unusual Options

Here the two folders at the bottom of the display ***Exclude Reflections*** and ***Unusual Options*** are closed but could be opened by picking the toggle button on the right hand side of the folder title line. At the top of the window are the ***Protocol*** and ***File*** folders- these are not toggleable folders -they are always open. The Protocol folder contains parameters which control what is available and default values set in the rest of the interface. For example opting to draw a ***vector difference*** map rather than the current choice of ***difference*** map will change the MTZ input label fields in the File folder.

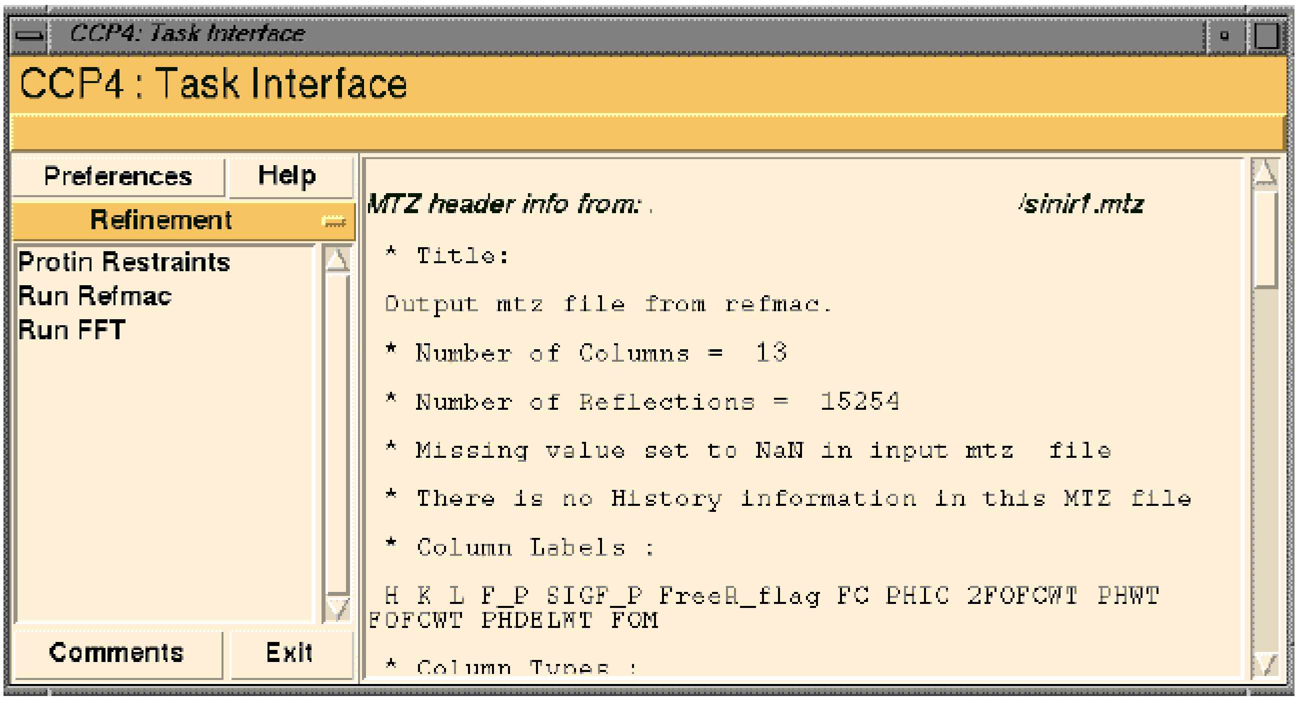
In the File folder the user can select a file - a file browser is brought up by clicking the filing cabinet icon. When the user has selects an input MTZ file it is read and useful information such as resolution range an space group are extracted and entered as the defaults in the appropriate fields. The names and types of the labelled columns are also extracted and used to set up a pop-up menu for selection of input columns.

One of the buttons at the bottom of the dialog box is an option to ***Save&Restore*** which will either save the current set parameters to file or restore previously saved parameters.

After this quick tour of the window layout you should be able to find your way around the interface. There is always the ***Help*** button which links to HTML web pages and there is context dependent help wich will bring up help on the relevant parameter if you press the < F1> key when the cursor is over an input field. Also when the cursor is over an input field the message line at the top of the window will display one line of information about the parameter.

The Main Window

The individual task windows are accessed from a main window which looks like this:



MTZ header info from: . /sinirf.mtz

```
 * Title:
 Output mtz file from refmac.
 * Number of Columns =  13
 * Number of Reflections =  15254
 * Missing value set to NaN in input mtz  file
 * There is no History information in this MTZ file
 * Column Labels :
 H K L F_P SIGF_P FreeR_flag FC PHIC 2FOFCWT PHWT
FOFCWT PHDELWT FOM
 * Column Types :
```

The contast colour field on the left (currently saying ***Refinement***) is a pop-up menu to access the various modules of the interface, each module corresponds to a stage in the structure solution process. Beneath the module name is a list of the tasks in that module, click on one of these to open the appropriate task window. The large scrolling frame on the right is used to list such things as log files or, in this case, the header information from an MTZ file generated using mtzdmp.

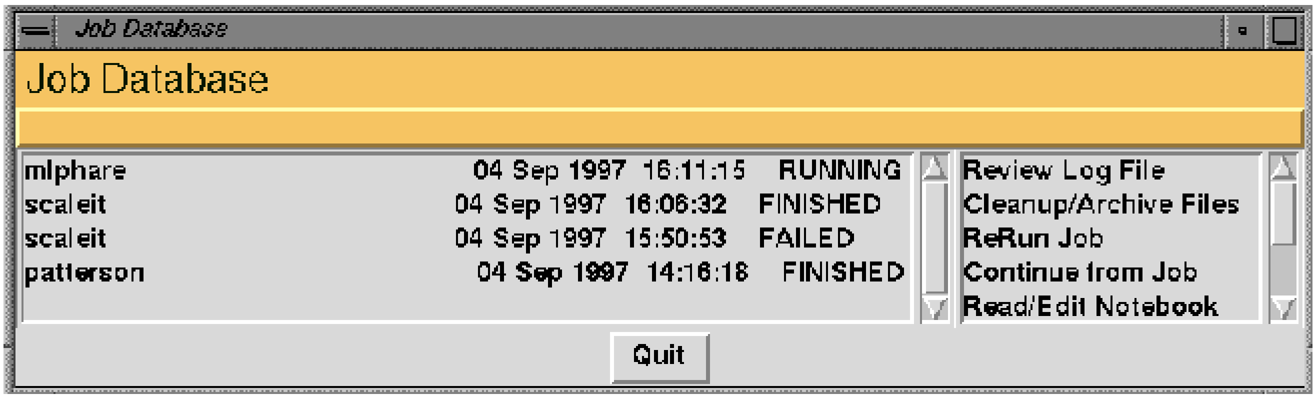
The ***Comments*** button on the bottom left will enable you to enter a message in a text editor and then mail it off, automatically, to the program developer. I am trying to encourage user feedback!

The Job Database

The interface includes some simple database functionality to keep track of the structure solution process. You should set up a different database for each structure project you are working on and the interface will allow easy switching between different project databases.

The database automatically keeps a record of every task run by saving the ***def*** file which contains the parameters used to run each task including the names of input and output files. Data files are not saved automatically but the interface will provide a simple mechanism to archive and restore data files and will keep a record of their context in the overall process.

The interface to the database is shown below. On the left of the window is a list of the jobs which are currently running or have completed with some information on their date/time of running and status. Other information such as output data file names could also be listed to help identify the job. On the right is a list of database options. The user can select a job from the list on the left and then pick a function on the right.


Job Database

Job Database

```
mlphare      04 Sep 1997  16:11:15   RUNNING
scaleit      04 Sep 1997  16:06:32   FINISHED
scaleit      04 Sep 1997  15:50:53   FAILED
patterson    04 Sep 1997  14:16:18   FINISHED
```

The functionality associated with the database includes:

 Reviewing log files and displaying graphical data in a form similar to the current xloggraph program.
 Deleting temporary filescreated while running the job.
 Archiving data files. This is saving a compressed version of the file in a 'safe' directory with a record of which job created the file.
 Rerunning a task - with the option to change some input parameters.
 A notebook which the user can edit to enter comments associated with each job.
 Interteraction with Data Harvesting tools which will simplify Data Deposition.

Future Plans

The first release of the interface will be part of the main CCP4 release in middle of 1998. This release should include interfaces to most commonly used CCP4 programs and the Job Database system. Intended for the future are:

2D Graph Plots which initially will be used to produce xloggraph style presentation of program out but which should be flexible enough for users to define their own data input and graph style. Tcl/Tk has tools for producing postscript files.

Interface to RasMol 3D molecular visualisation program to improove presentation of results and possibly also to simplify atom selection.

Plotting 2D Map Sections (and atom/vector positions) with some interactive tools this is particular useful for solving Pattersons.

MTZ File Editor and Data Analysis

Further suggestions are welcome.

Running the GUI on Different Platforms

The GUI is expected to run, as is, on any UNIX or LINUX platform. The GUI is written in Tcl/Tk which is interpreted at run time rather than compiled so the GUI does not need to be built but it will be necessary to have Tcl7.6 and Tk4.2 (or later versions) installed. This software is available free from several web sites and the GNU-style installation should be straightforward.There is a Tcl/Tk utility called ET which could enable us to bundle the Tcl/Tk executables and the GUI code to appear as one executable so it may be possible for CCP4 to provide an 'executable' for some popular platforms.

A prototype of the GUI has run on a Vax VMS system with no major problems. There are two recognised issues with porting to VMS:

the current VMS version of Tcl/Tk is not up to date and does not support a couple of useful (but not absolutely essential) functions
some GUI functionality, particularly operations such as moving and deleting files, are implemented by calls to system routines and so are machine dependent

Tcl/Tk runs on Macs and various Windows systems so, in principle, the GUI could be ported to these systems.

Just for Programmers...

..this is a brief overview of how the GUI is programmed.

The GUI program has two distinct layers - the core and the task interfaces. The core code includes:

general utilities such as file browser, database, the main window
CCP4 specific utilities for example reading MTZ file headers
a library of functions for creating task interfaces
a library of functions for use in run scripts

The basic core utilities are not CCP4 specific and could be used to interface to any program. The task interface side of the interface is intended to be simple and accessible to allow anyone to create or modify an interface to a program. For each task interface there are three files to define a task window and then run the program(s). The files are:

The Definition File

This contains a list of the names of all the parameters used in the interface , their data type and initial default values. The files used to ***Save&Restore*** the interface status are in the same format.

The Tcl File

This file describes the appearence of the task window. It is written in the Tcl scripting language but uses very few basic Tcl commands as it is based on the CCP4 interface library and should be comprehensible to someone with minimal knowlede of Tcl.

The Script File

The script file runs the program and is similar to the usual CCP4 com files. It has the advantage that, since it is written in Tcl, one file can run on all platforms (at least all the ones which support the interface!). Since these scripts do have to cope with all possible options they are not always very readable but the majority of users should have no need to look at these files. Rather than saving and/or editting a script file a user should save a definition file using the ***Save&Restore*** option.

An Example

Here is an example of what appears in each of the above files to handle the space group which appear in FFT task window shown above.

In the **fft.def** file:

```
SPACE_GROUP     _space_group              "P1"
```

The first column contains the parameter name, the second is the data type and the third is the initial default value. The data types are defined in a separate ***types.def*** file. The data types can be simple generic types such as *_logical* or *_positivereal* or more complex such as *_space_group* which can only take values which correspond to space group names or numbers read from a dictionary file. If the user enters an invalid space group name they will be warned.

In the **fft.tcl** file:

```
  CreateLine line \
      message "Space group (default as in MTZ file) (FFTSPACEGROUP)" \
      help "fftsgp" \
      label "Generate map in space group" \
      widget SPACE_GROUP
```

The backslash at the end of the Tcl line is a line continuation character so this is one command line laid out to be more readable. This calls the CCP4 interface routine ***CreateLine*** which returns a line id ***line***. There can be any number of subsequent arguments which are interpreted as a list of pairs of parameter types and parameters.

The ***message*** and ***help*** parameters define the context sensitive information which is available to the user if the cursor is in the space group input field. The message parameter is a line of text which will appear in the message line at the top of the window. The ***help*** parameter is a key to find the appropriate help text if the user picks the help function key *<F1>.*

The ***label*** and ***widget*** parameters define the text which appears in the interface and the parameter that the input field is tied to. The CreateLine function uses the information about the data type of the parameter SPACE_GROUP to determine what type of widget to draw.

Further arguments can be added to the CreateLine input to build up a more complex input line. The FFT interfacein the picture above has the scale amplitudes input fields on the same line as the spacce group input.

The **fft.run** file is created when the program is run. It is based on a template file called **fft.script** but has the additional information of parameters taken from the task window. At the top of this file are definitions of the values for all the parameters taken from the interface. For example:

```
set SPACE_GROUP P3121
```

Within the run script another temporary script file is created which is the command file read by the program. The appropriate coomand line is written to this temporary file:

```
WriteTmpFile "FFTSPACEGROUP $SPACE_GROUP"
```

The run script will run the CCP4 program(s) and when they have finished send a message to the GUI database to inform it that the job is completed.

CCP4 Refinement Workshop

Monday 1st September to Friday 5th September, University of York

<u>Organisers: Eleanor Dodson (York) and Jim Naismith (St. Andrews)</u>

Background

Working Group 1 decided that CCP4 should try to promote best practice amongst young crystallographers. The rationale being that the development in software has been so rapid that most labs do not have in-house expertise in all the major programs. It was thought that refinement techniques would be a useful starting point for such a course. Eleanor and Jim agreed to organise such a course in York.

Format of the course

We decided to split the course into tutorials and lectures. Initially we planned to accept only 20 students but we ended with 27 actually doing the tutorials. This is roughly one person per UK crystallography laboratory. The lectures were open to all and advertised via the CCP4 Bulletin Board. We actually rejected over 30 applicants. Amongst those were several from the USA. (Eleanor and Jim hereby volunteer to organise a repeat meeting in Florida preferably in February!) The timetable that was actually followed can be seen by clicking here. Some lecturers have already supplied Web formatted notes, others have promised and Jim is typing up his notes of the others. A full set of notes is planned to be available on the Web soon (see the program for details) and certainly by the end of November. An announcement will be made on the CCP4 Bulletin Board

Highlights of the course

The lecturers were all superb. David Watkin started the students off with an insight into what refinement actually means and what we actually do. Eleanor reminded us how to collect data. The importance of getting our structures right was underlined with a superb talk by Guy Dodson, who spoke about the buckling of the heme ring in hemoglobin and its biological implications. It would be fair to say Dale Tronrud stole the workshop, he combined powerful insight with a sense of humour. He got down in the F's and xyz's with the students and all professed to expert TNT users by the end. Paul Adams gave us a preview of CNS and gave a good explanation of torsional dynamics, again Paul helped the students run their scripts and iron out problems. Randy Read and Raj Pannu discussed maximum likelihood and its implementation in many software packages. Randy showed convincing evidence of the power of this technique. Both Raj and Randy helped out the CNS strugglers. Garib Murshadov told us how to run REFMAC and advertised improvements to come. Both he and Eleanor spent an afternoon and evening getting students using REFMAC like experts. Ian Tickle's presentation on RESTRAIN exposed the students to the concepts of TLS which is currently only implemented in RESTRAIN, Ian also helped students run these jobs. Isabel Unson gave a very good overview of SHELXPRO and got those with high resolution data started on this software. The theme of accuracy, precision and validation lead by Durward Cruikshank, Tom Oldfield and David Moss stimulated some of the most thought provoking discussion with Eleanor Dodson, Isabel Unson, Dale Tronrud, Jim Naismith, Ian Tickle and Randy Read all contributing to the argument. Tom Oldfield and Kevin Cowtan spoke about maps and their ideas and hints were of use to everyone, Zbigniew Dauter told us how to put in waters quickly and accurately. Kim Hendrick from EBI emphasised that if our results are to mean anything they have to be accessible to biologists. He talked about the new developments at EBI. Jim Naismith tried to explain the idiosyncrasies of the PDB, initiated the students into the mmCIF society and preached submission. Alexei Vegan and Richard Greaves wound up the meeting by outlining how to build dictionaries for those new ligands. Eleanor and Guy hosted a party on Friday evening/ Saturday morning where the students handed out presents to thank the speakers and organisers. Thoroughly the worse for wear we all departed Saturday morning.

Problems of the course

The principal complaint of the students that 27 was too many for the tutorials. This is probably true but choosing between people from different Universities was invidious and some people made late but compelling cases for inclusion. In the end we felt it better to have 27 people 80% satisfied rather than 20 people 90% satisfied.

Acknowledgements

Jim and Eleanor are enormously grateful to all those who spoke and demonstrated, without them there would have been no course. The staff at York were very helpful and made sure things went smoother than they should have done. The students were fantastic and did not complain when the organisation went off the rails. No UK student/post-doc had to pay a penny to come thanks to CCP4.