CCP4 NEWSLETTER ON PROTEIN CRYSTALLOGRAPHY

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

Number 48 **Summer 2012**

Contents

11. **Space group validation with Zanuda**
    Andrey A. Lebedev, *CCP4, Research Complex at Harwell, STFC Rutherford Appleton Laboratory, Didcot OX11 0FA, UK*
    Michail N. Isupov, *Henry Wellcome Building for Biocatalysis, College of Life and Environmental Sciences, University of Exeter, Stocker Road, Exeter EX4 4QD, UK*

12. **The xia2 manual**
    Graeme Winter, *Diamond Light Source, Harwell Science and Innovation Campus, Didcot OX11 0DE, UK*
    David Waterman, *CCP4, Research Complex at Harwell, STFC Rutherford Appleton Laboratory, Didcot OX11 0FA, UK*

13. **AMPLE - using *ab initio* modelling to tackle difficult molecular replacement cases**
    Jaclyn Bibby, *Institute of Integrative Biology, University of Liverpool, Liverpool, L69 7ZB, UK*
    Ronan Keegan, *CCP4, Research Complex at Harwell, STFC Rutherford Appleton Laboratory, Didcot OX11 0FA, UK*
    Daniel J. Rigden, *Institute of Integrative Biology, University of Liverpool, Liverpool, L69 7ZB, UK*
    Martyn Winn, *STFC Daresbury Laboratory, Warrington, WA4 4AD, UK*
    Olga Mayans, *Institute of Integrative Biology, University of Liverpool, Liverpool, L69 7ZB, UK*

Editor: David Waterman, CCP4, CSE Department, STFC Rutherford Appleton Laboratory, Didcot, Oxon, OX11 0FA, UK

Contributions are invited for the next issue of the newsletter, and should be sent to David Waterman by e-mail at david.waterman@stfc.ac.uk. HTML is preferred but other editable formats are also acceptable.

Preface

Eugene Krissinel

Research Complex at Harwell, Rutherford Appleton Laboratory, Didcot OX11 0FA, UK

Dear CCP4 Users,

This Letter begins a new series of Newsletters concerning the CCP4 Suite, following major changes that happened to the CCP4 core team in the past two years. The Project has moved from Daresbury Laboratory near Liverpool to the Research complex at Harwell, next to Diamond Light Source in the Rutherford Appleton Laboratory, south of Oxford. That's only about 170 miles away – a small distance compared to the spread of our users – but it made a considerable change. We continue to enjoy the expertise of Charles Ballard and Ronan Keegan, who served you for quite a number of previous years, and Martyn Winn gives us a helpful hand from Daresbury. These days, you may see new names on the CCP4 Helpdesk: David Waterman, Andrey Lebedev, Marcin Wojdyr and Ville Uski, who joined CCP4 from between one year to half a year ago.

CCP4 has always been a very dynamic Project, inspired by world-wide input and communications, with many changes occurring on a daily basis. The CCP4 core team is committed to providing a good service to the crystallographic community by maintaining and delivering the Suite in the most convenient way, tested and accompanied by custom support through the CCP4 Helpdesk. Over years, the Suite has grown to a considerable size. Currently, it includes some 200 programs, contributed by different authors, written in different languages and exercising a wide spectrum of personal styles. This makes the work of the CCP4 Core team very interesting, but also challenging and difficult.

Traditionally, CCP4 Newsletters accompanied CCP4 releases, as is the case now, and we usually have two releases every year. Being presented with a rapid development of our most important programs, we may introduce a quick update mechanism in the near future in order to deliver changes on a shorter, inter-release, time scale. In the 6.3.0 release, you may notice a new helper application, CCP4 Package Manager, which is a graphical CCP4 installer for Linux and Mac OS systems. We hope that this application will help us to cope with the diversity of Linux platforms and deliver platform-specific packages, such as Coot, to our users in a much more convenient way. Another new feature we are working on, is a new, cross-platform build system, which would allow a user or developer to build selected CCP4 libraries and programs on their own machines with a single command doing the whole job - from automatic checkout from CCP4 repositories (including all dependences) to final configuration and compilation. In order to assure the quality of our builds, we are setting up an automated build-and-test system for a considerable

number of various platforms and compilers. This provides the long-awaited CCP4 nightly builds, some of which are already available on CCP4 web-site, and is something that our most active developers will hopefully appreciate.

In its new location, CCP4 enjoys close links to Diamond Light Source, with many users at our doorstep and many possibilities to see our software in action and get new ideas. CCP4 has a few common projects with Diamond, and are keen on providing Diamond's MX beamlines with automated structure solution pipelines. Most certainly, this collaboration does not draw our attention away from the multi-thousand CCP4 user community elsewhere in the world, and we are equally enthusiastic and open for input and requests from anybody.

We hope that you will find the recent changes in CCP4 useful. Our Newsletters will inform you about important updates and new additions in the Suite, as well as various projects we are working on and plans that we have. Remember too, we are always looking for feedback and input from you.

Eugene Krissinel,

on behalf of the CCP4 Core team at RCaH/RAL.

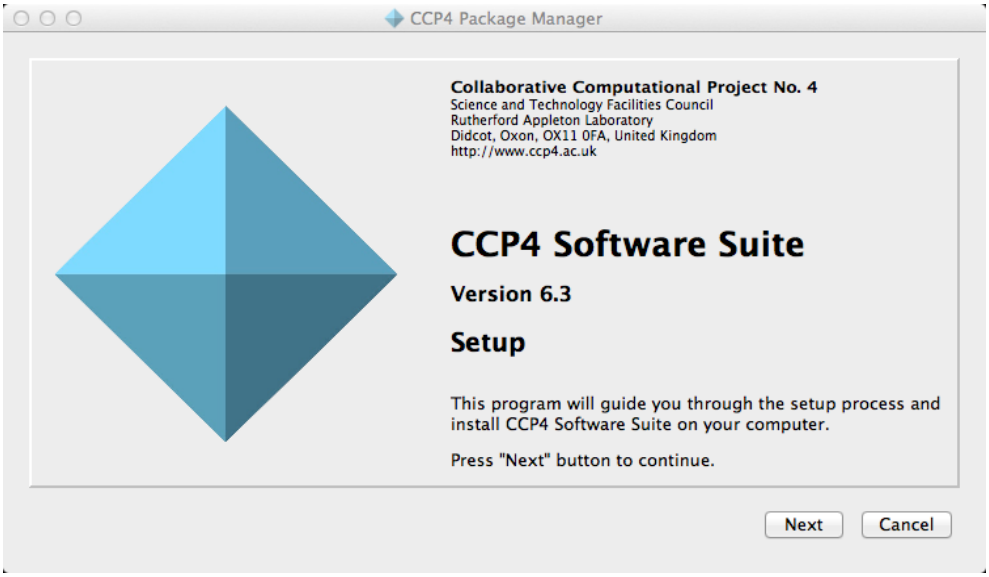
The CCP4 core team. From left to right: Andrey Lebedev, Marcin Wojdyr, Ronan Keegan, Charles Ballard, Eugene Krissinel, David Waterman and Ville Uski

What's new in CCP4

Charles Ballard

CCP4, Research Complex at Harwell, Rutherford Appleton Laboratory, Didcot OX11 0FA, UK

The new release of the CCP4 suite is version 6.3.0. This contains the usual compliment of new programs, and updates. One significant change that visitors to the download page http://www.ccp4.ac.uk/downloads will notice is the new Package Manager, shown below. This gives a better install experience for Linux and OS X.



The CCP4 Package Manager.

The old style packages and ftp site are available as alternatives. Another change is the co-release of Arp/Warp 7.3 from Hamburg. This is a big step for our academic users and those commercial sites that have an Arp/Warp license.

For CCP4 v6.3.0 “Settle”

New Programs:

aimless (Phil Evans)

> *scala* replacement for the scaling and merging of diffraction data. Handles symmetry related reflections together and computes dataset quality statistics on unmerged data.

prosmart (Rob Nichols)

> *prosmart_restrain*: external restraint generation for use in the refinement of protein structures.
>
> *prosmart_align*: alignment, superposition and scoring of protein chains.

edstats (Ian Tickle)

> electron density map statistic calculation including per residue real space correlation coefficient.

nautilus (Kevin Cowtan)

> statistical automated building of RNA/DNA in electron density, in the manner of buccaneer.

zanuda (Andrey Lebedev)

> analysis of refinement results, in particular confirmation of spacegroup.

phaser.sculptor (Gabor Bunkoczi)

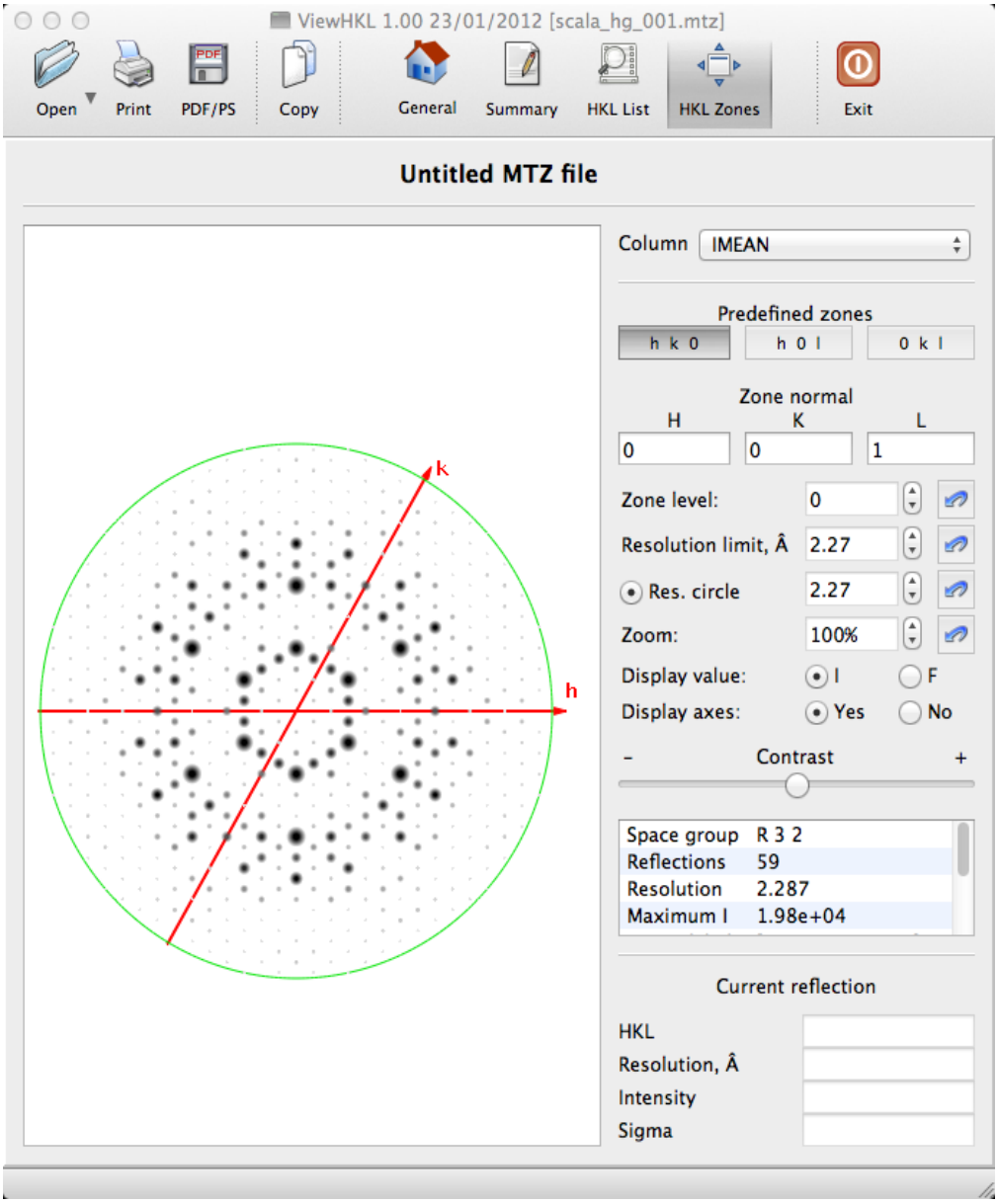
> multi-protocol search model preparation including weighted sequence similarity and surface accessibility.

gesamt (Eugene Krissinel)

> new alignment and superposition program.

ViewHKL (Eugene Krissinel)

> application for viewing the contents of reflection files, now the default mtz file viewer for the binary distributions.



Viewhkl display of the contents of an mtz file

Major Updates:

Refmac 5.7 (Garib Murshudov)

improved jelly body refinement and use of restraint information.

DNA/RNA and sugar/pucker restraints.

simultaneous refinement and density modification for SAD.

Gibbs sampling of conformational space.

estimation of errors of individual atoms, leading to better electron density.

Phaser 2.5 (Airlie McCoy, Randy Read)

correction for tNCS in MR.

imosflm/ipmosflm (Andrew Leslie, Harry Powell, Owen Johnson)

improved handling of Pilatus Images.

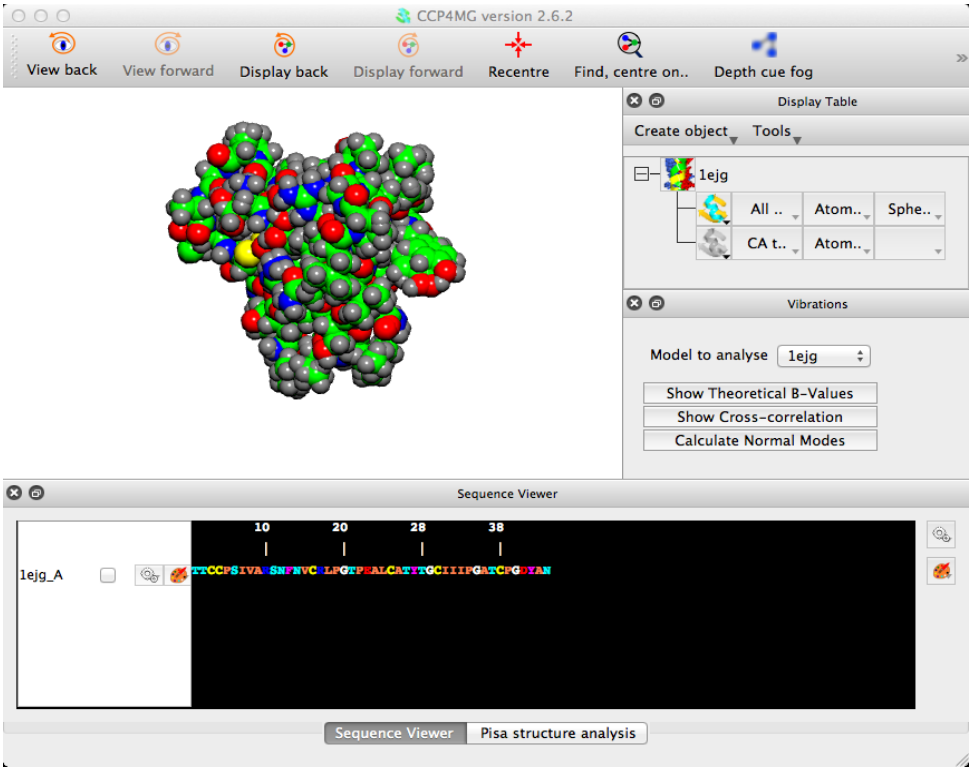
option to use aimless instead of scala in AutoScale.

PISA (Eugene Krissinel)

updated database.

CCP4MG 2.6 (Stuart McNicholas)

improved stability.



CCP4MG showing the sequence viewer tool

New Pipelines and interfaces:

Dimple (Ronan Keegan, George Pelios)

ligand density location.

Ample (Jaclyn Biddy)

ab initio molecular replacement using ROSETTA search models.

Phaser anisotropy calculation and twinning analysis.

new ccp4i interfaces giving access to the anisotropy correction and ML twinning analysis of phaser.

The distributed binaries run across all major platforms, Linux 32/64-bit, OS X 10.4+ and MS Windows. Windows users will be pleased to note that the functionality on Windows is now far closer to the Unix-like systems.

Other News

Modernisation:

CCP4 began using the venerated CVS version control system back in 1988. It is not an understatement to say that there is a lot of history in there. Since then the suite has increased in size and complexity, and in the outside world the thinking on version control design has changed markedly. Beginning earlier this year, the core team has begun the task of reorganising the suite and moving to the bazaar version control system. Bazaar is a distributed version control system, in the manner of Git and Mercurial. The ported repositories are hosted on a dedicated server (http://fg.oisin.rc-harwell.ac.uk), and are world-readable with the latest versions of the software for the very brave.

The work on the repositories is part of an update of the archiving and distribution of the suite. The other early fruits of this are the Package Manager, and an Updates Manager which will be out in 6.3.1. As a further development the suite will be undergoing further major surgery with the custom configure script being replaced by a more modern system. Much of the build will be made using Cmake. The first pieces of this can be seen in the CmakeList.txt files. Cmake is included with most Linuxes and as a download for OSX. It is an easier to use and more flexible system than the current autotools system.

As part of the move to a quicker release cycle with updates, other plans are in progress. Currently the core group have build and smoke tests running on various platforms via a buildbot herd. As a result of this, nightly builds are available “as is” in an unsupported form. The next twelve months and beyond will see fuller regression tests and even more supported systems, along with automatic bundling. This will give an alternative route to get the latest, but possibly unstable, version of the suite for those who do not want to roll their own.

The Study Weekend:

The 2012 Study Weekend was on “Data-processing” and took place in Warwick. Some 420 attendees enjoyed talks from the likes of Zbigniew Dauter and Wayne Hendrickson. The Acta D special issue should be available early in the new year. The 2013 Study Weekend will be on “Molecular Replacement” and will take place on the 3-4 January in Nottingham, UK.

As well as the Study Weekend, CCP4 sponsors and organises workshops around the world. The fifth annual CCP4-APS workshop took place in late June 2012 at the Argonne National Laboratory. This was attended by 26 students and lecturers from the US and Europe. This highly successful workshop will be running again next year, so watch the CCP4 website for news of the application dates.

Over the next 12 months there are other planned workshops in Hamburg (Germany), Fukuoka, (Japan) and Brazil.

Interactive Graphical Viewer and Browser for Reflection Data

Eugene Krissinel[a], Phil Evans[b]

[a]*CCP4, Research Complex at Harwell, Rutherford Appleton Laboratory, Didcot OX11 0FA, UK*
[b]*MRC Laboratory of Molecular Biology, Hills Road, Cambridge CB2 0QH, UK*

Reflection data analysis plays an important role in the structure solution process and is often an essential step in choosing the optimal data processing strategy. In CCP4, several programs provide for various aspects of data analysis, such as SFCHECK [1], CTRUNCATE [2], POINTLESS [3], SCALA [3], WILSON [4] and others.

Sometimes, a deeper inspection of reflection data is required, where direct access to all the content of an mtz file, both merged and unmerged, is needed. This is achieved with the help of various mtz utilities in CCP4, in particular MTZDUMP [5], and the graphical applications HKLPLOT [6] and HKLVIEW [7]. MTZDUMP provides a comprehensive inspection of reflection data, but it represents a command-prompt application and necessitates examination of extensive log files, which is inconvenient for a general user. HKLPLOT and HKLVIEW provide a graphical representation of reflection data, but they do not have all MTZDUMP capabilities and are technologically outmoded.

A new graphical application, VIEWHKL, has been developed, which aims to combine the comprehensiveness of MTZDUMP with the graphical convenience of HKLPLOT/HKLVIEW whilst bringing the latter to a modern level of graphical, user-friendly computing. VIEWHKL is written in C++ Qt (version 4.7 and higher), and makes extensive use of CCP4 Clipper library [8], where mtz-handling classes have been extended to deal also with unmerged files. The program utilizes multi-threading technologies in order to enhance user experience and keep GUI controls smoothly responsive while dealing with mtz files of significant size. VIEWHKL is fully portable to all platforms supported by Qt, and is included in CCP4 distributions for Mac OSX, Linux and Windows.

Figure 1 presents a general outlook of VIEWHKL. The program has 4 tabs for presenting different aspects of data: *General*, *Summary*, *HKL List* and *HKL Zones*, plus an additional tab named *Batches* in case of unmerged files. Upon loading a file with reflection data, header information is read first and displayed in the *General* tab. While loading some files may take as long as 10-15 seconds, the header information is displayed instantly. The tab presents data on general properties such as resolution range, cell parameters, number of reflections, and a list of data sets contained in the file with respective column labels and types.

The *Summary* tab (Figure 2) displays history records stored in the mtz header, as well as extended dataset summaries, which includes columns' value ranges, number of missed reflections, completeness, mean values and resolution range. Each row of the table corresponds to an mtz column; columns are grouped by type and shown in alternating colours.

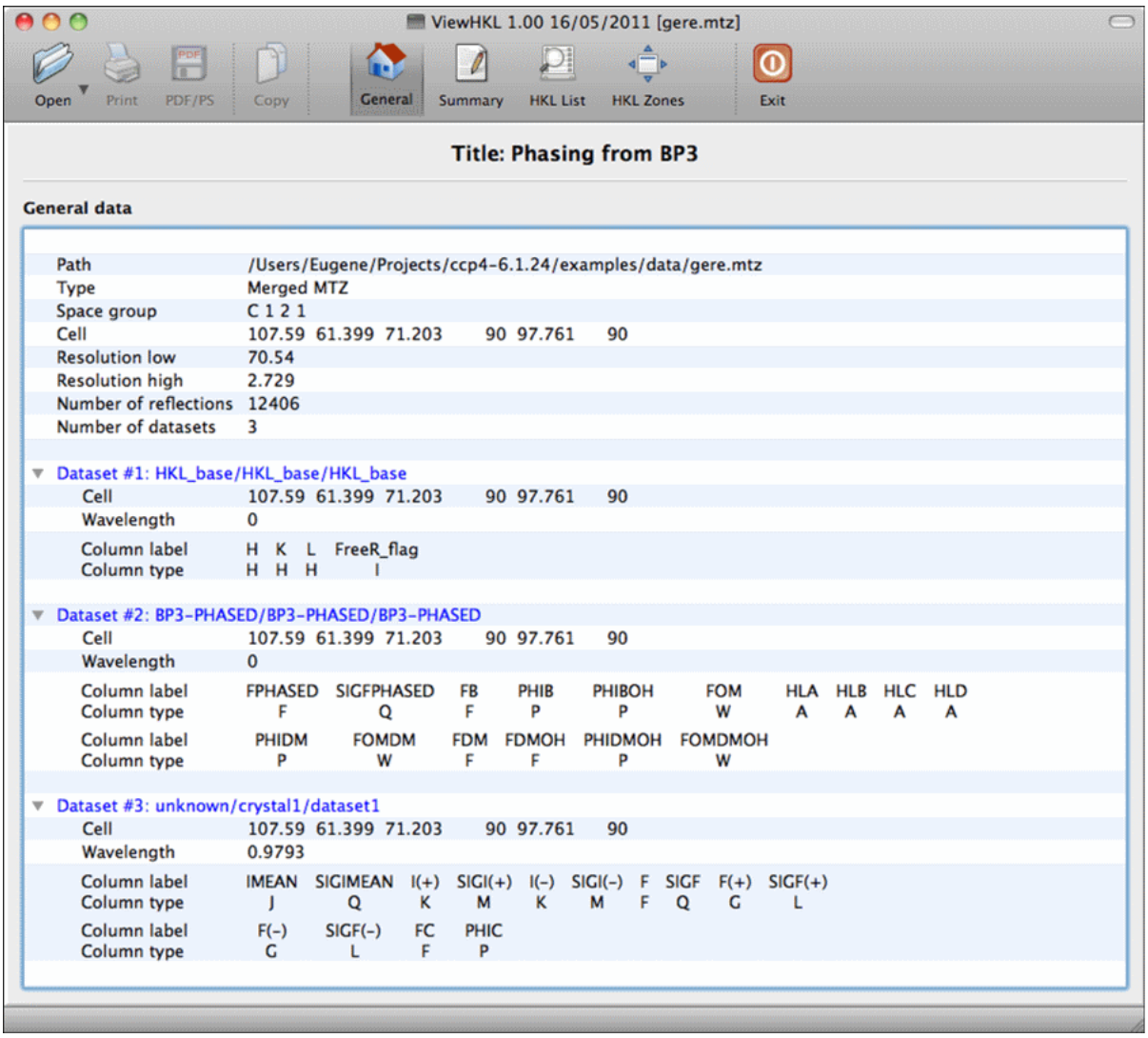
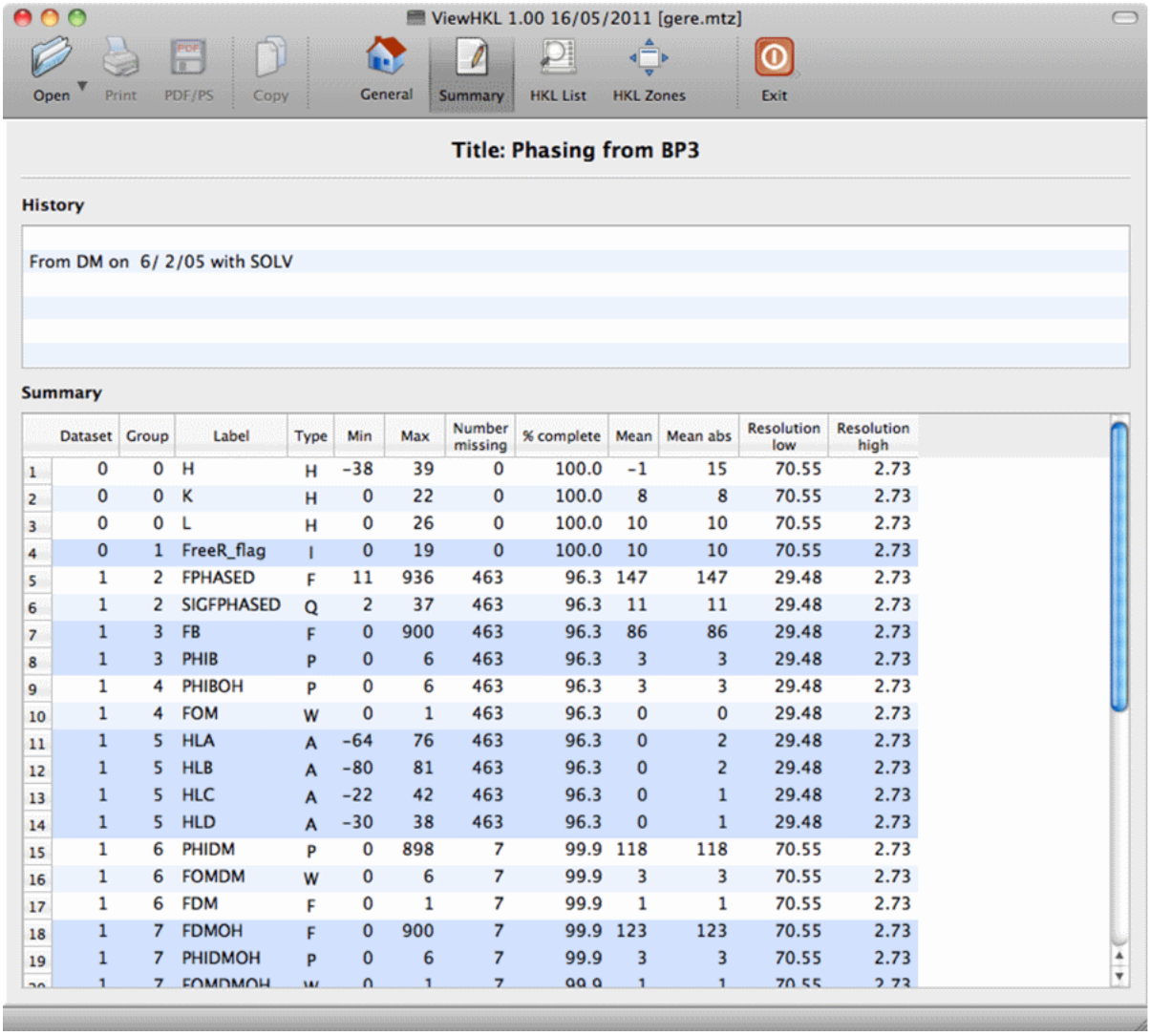


Figure 1. General outlook of VIEWHKL. The General tab presents basic header information from reflection file, and opens automatically while the file is being loaded

The reflection data browser (*HKL List* tab, Figure 3) allows for visual inspection of raw data and searches for individual reflections. Columns are grouped by column type, same as in the *Summary* tab (Fig. 2), and presented in alternating colours for easier perception. Due to the large number of reflections ($O(10^6)$), putting all of them into one table is not optimal for performance considerations. Therefore, the browser presents data in sliding “windows”, the position and size (the number of reflections) of which is chosen with two horizontal sliders at the bottom of the page. In order to look for a particular reflection, the user needs to put h,k,l indices into the corresponding input fields provided in the bottom of the page, and press Enter; the page will automatically update such that the reflection (or, if such reflection is not found, the one with closest h,k,l) is displayed in the first row.



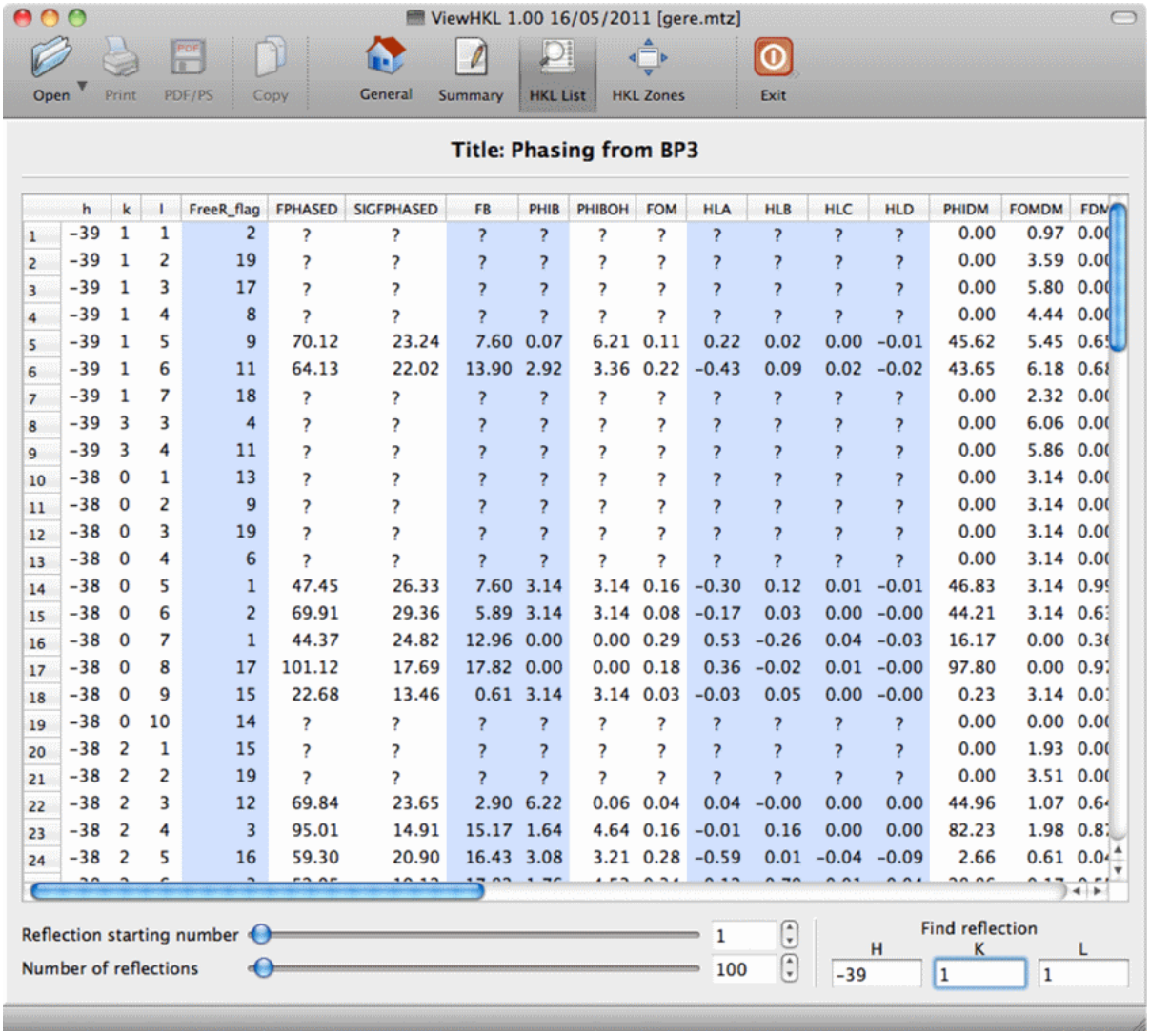
	Dataset	Group	Label	Type	Min	Max	Number missing	% complete	Mean	Mean abs	Resolution low	Resolution high
1	0	0	H	H	-38	39	0	100.0	-1	15	70.55	2.73
2	0	0	K	H	0	22	0	100.0	8	8	70.55	2.73
3	0	0	L	H	0	26	0	100.0	10	10	70.55	2.73
4	0	1	FreeR_flag	I	0	19	0	100.0	10	10	70.55	2.73
5	1	2	FPHASED	F	11	936	463	96.3	147	147	29.48	2.73
6	1	2	SIGFPHASED	Q	2	37	463	96.3	11	11	29.48	2.73
7	1	3	FB	F	0	900	463	96.3	86	86	29.48	2.73
8	1	3	PHIB	P	0	6	463	96.3	3	3	29.48	2.73
9	1	4	PHIBOH	P	0	6	463	96.3	3	3	29.48	2.73
10	1	4	FOM	W	0	1	463	96.3	0	0	29.48	2.73
11	1	5	HLA	A	-64	76	463	96.3	0	2	29.48	2.73
12	1	5	HLB	A	-80	81	463	96.3	0	2	29.48	2.73
13	1	5	HLC	A	-22	42	463	96.3	0	1	29.48	2.73
14	1	5	HLD	A	-30	38	463	96.3	0	1	29.48	2.73
15	1	6	PHIDM	P	0	898	7	99.9	118	118	70.55	2.73
16	1	6	FOMDM	W	0	6	7	99.9	3	3	70.55	2.73
17	1	6	FDM	F	0	1	7	99.9	1	1	70.55	2.73
18	1	7	FDMOH	F	0	900	7	99.9	123	123	70.55	2.73
19	1	7	PHIDMOH	P	0	6	7	99.9	3	3	70.55	2.73

Figure 2. Summary tab presents history records and extended data set summary.

The fourth tab, *HKL Zones* (Figure 4), provides graphical representation of reflection data. In order to represent 3D data on the screen, the h, k, l sphere is sliced into planes (or "zones") according to the equation

$$H \cdot h \quad + \quad K \cdot k \quad + \quad L \cdot l \quad = \quad N$$

where {*H,K,L*} is the plane's normal and *N* is an offset from the origin ("*Zone level*"). The display in the left-hand side of the page shows reflections that are found in the zone, as well as the plane axes corresponding to the particular choice of {*H,K,L*}. Reflections are shown as round spots, whose size and intensity indicate the reflection's amplitude or intensity (as chosen by the corresponding radio buttons in the right-hand side toolbar).



	h	k	l	FreeR_flag	FPHASED	SIGFPHASED	FB	PHIB	PHIBOH	FOM	HLA	HLB	HLC	HLD	PHIDM	FOMDM	FDM
1	-39	1	1	2	?	?	?	?	?	?	?	?	?	?	0.00	0.97	0.0(
2	-39	1	2	19	?	?	?	?	?	?	?	?	?	?	0.00	3.59	0.0(
3	-39	1	3	17	?	?	?	?	?	?	?	?	?	?	0.00	5.80	0.0(
4	-39	1	4	8	?	?	?	?	?	?	?	?	?	?	0.00	4.44	0.0(
5	-39	1	5	9	70.12	23.24	7.60	0.07	6.21	0.11	0.22	0.02	0.00	-0.01	45.62	5.45	0.6(
6	-39	1	6	11	64.13	22.02	13.90	2.92	3.36	0.22	-0.43	0.09	0.02	-0.02	43.65	6.18	0.6(
7	-39	1	7	18	?	?	?	?	?	?	?	?	?	?	0.00	2.32	0.0(
8	-39	3	3	4	?	?	?	?	?	?	?	?	?	?	0.00	6.06	0.0(
9	-39	3	4	11	?	?	?	?	?	?	?	?	?	?	0.00	5.86	0.0(
10	-38	0	1	13	?	?	?	?	?	?	?	?	?	?	0.00	3.14	0.0(
11	-38	0	2	9	?	?	?	?	?	?	?	?	?	?	0.00	3.14	0.0(
12	-38	0	3	19	?	?	?	?	?	?	?	?	?	?	0.00	3.14	0.0(
13	-38	0	4	6	?	?	?	?	?	?	?	?	?	?	0.00	3.14	0.0(
14	-38	0	5	1	47.45	26.33	7.60	3.14	3.14	0.16	-0.30	0.12	0.01	-0.01	46.83	3.14	0.9(
15	-38	0	6	2	69.91	29.36	5.89	3.14	3.14	0.08	-0.17	0.03	0.00	-0.00	44.21	3.14	0.6(
16	-38	0	7	1	44.37	24.82	12.96	0.00	0.00	0.29	0.53	-0.26	0.04	-0.03	16.17	0.00	0.3(
17	-38	0	8	17	101.12	17.69	17.82	0.00	0.00	0.18	0.36	-0.02	0.01	-0.00	97.80	0.00	0.9(
18	-38	0	9	15	22.68	13.46	0.61	3.14	3.14	0.03	-0.03	0.05	0.00	-0.00	0.23	3.14	0.0(
19	-38	0	10	14	?	?	?	?	?	?	?	?	?	?	0.00	0.00	0.0(
20	-38	2	1	15	?	?	?	?	?	?	?	?	?	?	0.00	1.93	0.0(
21	-38	2	2	19	?	?	?	?	?	?	?	?	?	?	0.00	3.51	0.0(
22	-38	2	3	12	69.84	23.65	2.90	6.22	0.06	0.04	0.04	-0.00	0.00	0.00	44.96	1.07	0.6(
23	-38	2	4	3	95.01	14.91	15.17	1.64	4.64	0.16	-0.01	0.16	0.00	0.00	82.23	1.98	0.8(
24	-38	2	5	16	59.30	20.90	16.43	3.08	3.21	0.28	-0.59	0.01	-0.04	-0.09	2.66	0.61	0.0(

Figure 3. HKL List tab provides hkl data browser and reflection data search function.

The uppermost control in the right-hand side toolbar (see Fig. 4) provides the choice of reflection data (group of columns) to be displayed in the page. Three fixable buttons allows for a quick switch between the “most popular” zones: *hk0* (i.e. *H=K=0, L=1*), *h0l* and *0kl*. Each choice of {*HKL*} is indicated in “Zone normal” input fields. Using these fields, a user may visualize a zone with arbitrary {*HKL*}.

The next four controls in the toolbar are for the choice of zone level *N*, the resolution limit for display and, independently, for the green circle in it (which can be switched on/off using the corresponding radio button), and for the display’s zoom. Reset buttons on the right from these controls assign default values for them. The resolution circle in the screen may be adjusted directly with the mouse (grab-and-drag), which is synchronized with the displayed value in the corresponding control.

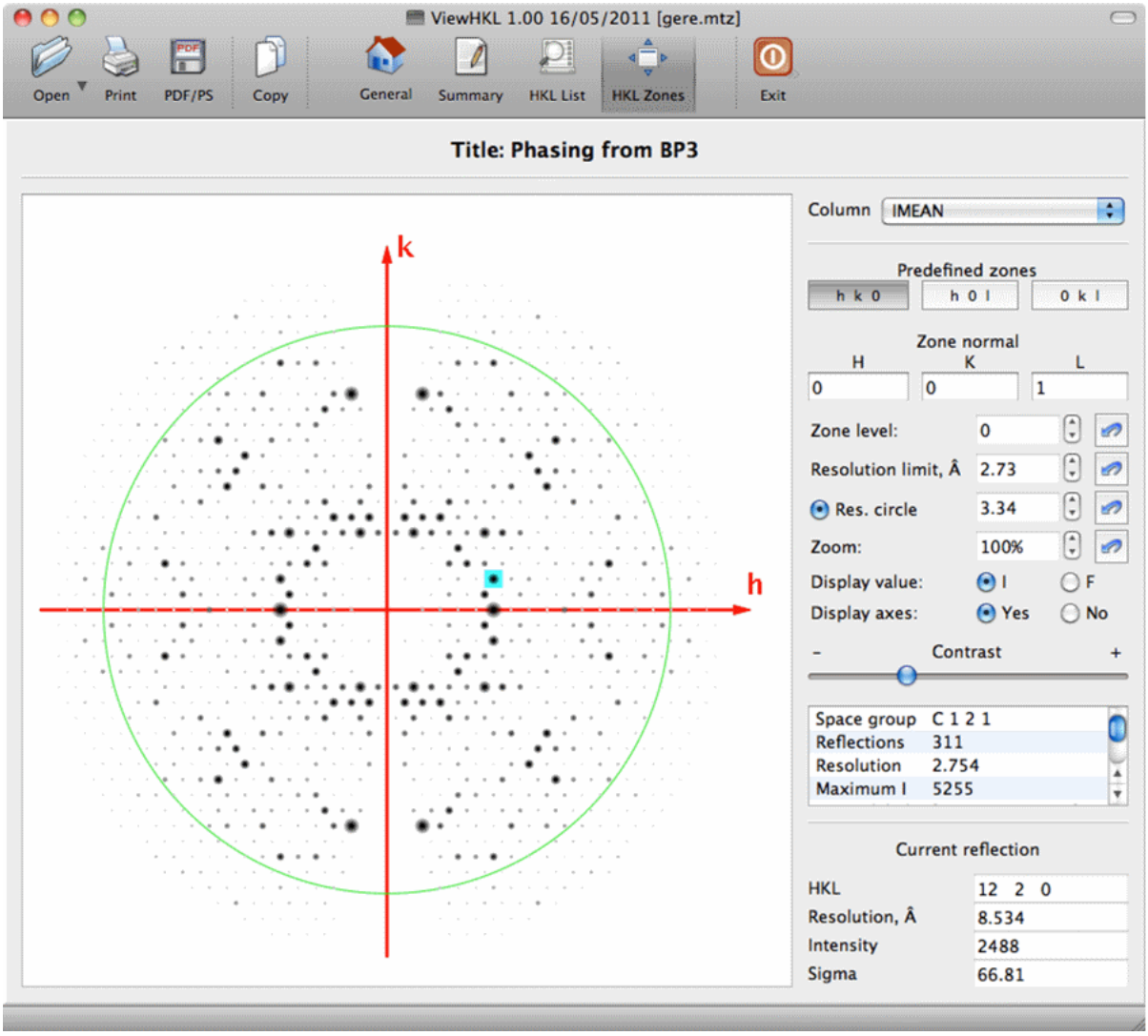


Figure 4. HKL Zones tab provides graphical representation of reflection data.

The next control element, the contrast slider, allows a user to adjust the enhancement of strong reflections to their personal taste and in dependence of particular data, in order to make the identification of the reflection pattern easier. This may also be helped sometimes by choosing amplitude, rather than intensity, for the representation of reflections in the display. The corresponding choice is provided by two radio buttons above the contrast slider.

Below the contrast slider, the right-hand side toolbar shows some basic data for the chosen zone (space group, number of reflections, resolution and intensity/amplitude limits, and cell parameters). Hovering the mouse over reflection spots in the display is accompanied by the indication of reflection parameters ({h,k,l}, resolution, intensity and sigma) in the bottom part of the toolbar, with the highlighting of the corresponding reflection in the display area (cf. Fig.4).

Starting from CCP4 version 6.3.0, VIEWHKL supersedes MTZDUMP as the default application for the inspection of MTZ files in the CCP4 GUI, and will be integrated with the next-generation GUI-2, which is currently under development.

Acknowledgement

The authors are grateful to members of CCP4 Working Group 2 and many CCP4 developers and users, who tested VIEWHKL and gave a constructive feedback, which helped debugging and design.

References

[1] A.A.Vaguine, J.Richelle, S.J.Wodak. SFCHECK: a unified set of procedure for evaluating the quality of macromolecular structure-factor data and their agreement with atomic model. Acta Cryst.(1999). D55, 191-205

[2] http://www.ccp4.ac.uk/html/ctruncate.html

[3] P.R.Evans, Scaling and assessment of data quality, Acta Cryst. D62, 72-82 (2006)

[4] http://www.ccp4.ac.uk/dist/html/wilson.html

[5] http://www.ccp4.ac.uk/dist/html/mtzdump.html

[6] http://www.ccp4.ac.uk/dist/html/hklplot.html

[7] http://www.ccp4.ac.uk/dist/html/hklview.html

[8] Cowtan K. (2003) IUCr Computing Commission Newsletter, 2, The Clipper C++ libraries for X-ray crystallography, 4-9

Enhanced Structural Alignment with GESAMT

Eugene Krissinel

CCP4, Research Complex at Harwell, Rutherford Appleton Laboratory, Didcot OX11 0FA, UK

Structural alignment of macromolecules has many applications in structure solution and analysis. Common examples include the comparison of models for molecular replacement, identification of stable domains, conformational analysis, comparative analysis of binding sites and protein function prediction.

The CCP4 Software Suite includes several programs for structural alignment and calculation of best structure superposition: LSQKAB [1], POLYPOSE [2] and SUPERPOSE (also known as Secondary Structure Matching, SSM, [3]). In addition, 3D structure alignment and superposition may be calculated with MOLREP, a program for molecular replacement [4]. These programs are not functionally equivalent to each other. E.g., LSQKAB is extremely fast and efficient but needs a manual input of matching atom pairs. POLYPOSE performs multiple alignment of a large number of structures but assumes them to be of the same length, with one-to-one correspondence between their atoms. The actual structure alignment in CCP4 (i.e. automatic computation of equivalent atom pairs) is done by SSM and MOLREP. SSM was recognized as the fastest and yet a top-quality application in the field [5]. This algorithm was designed primarily for fast searches in structural databases at the European Bioinformatics Institute (EBI), for which certain limitations were adopted. For example, SSM is applicable only to structures with at least several secondary structure elements. In addition, SSM may underperform on fragmented chains and, in certain cases, it prunes search trees if favourable for speed. MOLREP is free from SSM limitations, however, structural alignment is by far not the main option for this application, and comes as a by-product of a more general task. As a result, structural alignment in MOLREP is slow for interactive applications and database searches.

Development of GESAMT (General Efficient Structural Alignment of Macromolecular Targets) was motivated as an attempt to complement the Suite with a structural aligner, which would be comparable to SSM in performance and quality but free of SSM's limitations. In particular, applicability to incomplete structures, which do not allow for a reliable identification of secondary structure elements and may be highly fragmented, was an essential requirement.

In its essence, GESAMT may be found similar to other structural alignment algorithms, such as Combinatorial Extension (CE) [6]. At first stage, the given chain-wise structures S_1 and S_2 are represented as sequences of short overlapping fragments. Next, GESAMT superposes all fragments of S2 onto fragments of S1, and clusters the results in superposition matrix space. Finally, the largest clusters are refined and extended using iterative dynamic programming along the lines described by Gerstein and Levitt [7].

Two major points make GESAMT different from similar techniques. Firstly, clustering of short fragments is done using a global (structure-based), rather than a local (fragment-based) distance measure in superposition matrix space. Global assessment is considerably more time consuming than the local one. However, it results in a rather aggressive removal of unsuitable fragment superpositions at early stages, which results in

a dramatic decrease of the total number of primary acts of assessment. Overall, it was found that using an expensive, but aggressive assessment technique results in a surprisingly efficient “quenching” of combinatorial explosion, which other methods (such as CE) tackle with an empiric set of rules for the pruning of the search space.

The second new feature, implemented in GESAMT, is using SSM’s Q-score:

$$Q = \frac{N_{align}^2}{\left(1 + \left(rmsd / R_0\right)^2\right) N_1 N_2}$$

as a target function for the iterative dynamic programming refinement (N_{align} stands for the number of aligned residues, $N_{1/2}$ is number of residues in structures $S_{1/2}$, *rmsd* is the r.m.s.d. calculated between aligned residues at best structure superposition, and R_0 is an empiric parameter for balancing alignment length and r.m.s.d.). Over many years of experience with SSM algorithm, the Q-score was confirmed to be superior to r.m.s.d. and simple distance cut-off scores. However, the Q-score cannot be represented as a sum of effects from the alignment of particular pairs of residues, therefore it cannot be used “as-is” in a dynamic programming algorithm. Therefore, a special procedure was developed, which uses a local linearization of Q-score and converges to solution in a self-consistent manner.

Full details of the GESAMT algorithm will be published elsewhere. Below, we briefly discuss some of its main features. We give them in comparison with SSM, which is a legitimate approach here since both algorithms use the Q-score as a target function in order to identify suitable alignments.

Figure 1 presents Coverage vs. Error [8] plots for Gesamt and SSM. As may be seen, Gesamt provides better identification of similar and dissimilar structures, using SCOP’s definition. This difference is notable on the level of SCOP families at higher coverages, and holds true on fold and class levels at all coverages (sensitivities), where GESAMT gives 3 to 10 times less errors than SSM. Considerable enhancement of structure recognition power makes GESAMT a “must have” alternative to SSM (and possibly other aligners) in structural bioinformatics applications.

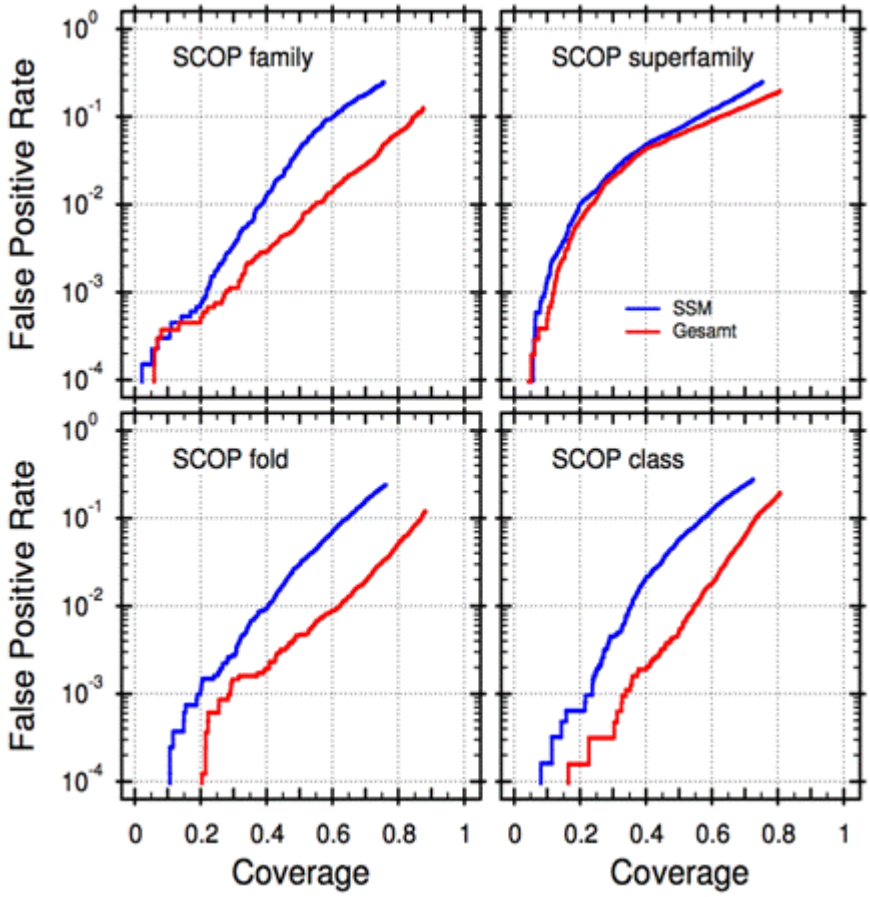


Figure 1. Sensitivity and specificity analysis for GESAMT and SSM (Coverage vs. Error [8]). The FATCAT benchmark set [9], comprising 15002 pairs of structures belonging to similar and dissimilar SCOP domains, was used to generate the curves. Neither SSM nor GESAMT were trained on this benchmark set.

Curiously, GESAMT does not offer a significant improvement of the error rate in case of SCOP superfamilies, where it is limited to a factor of 2 at selected coverages. Here, both SSM and GESAMT reach the highest level of errors comparing to other SCOP categories. The reason for these results remains unclear. In this respect, note that SCOP superfamilies are defined by a *probable* common evolutionary origin. The results may suggest that SCOP classification of superfamilies is less perfect than that of other categories. Other reasons may equally play a role such as the particular composition of the benchmark set, or indeed there could be something “special” with SSM and GESAMT algorithms, however, the latter was not confirmed by the performed investigations.

Figure 2 presents a comparative analysis of SSM and GESAMT. In the first three panels of Fig. 2, each dot represents a pair of alignments performed by SSM and GESAMT for the same protein pair. The resulting Q-scores and CPU times are used as the dot coordinates.

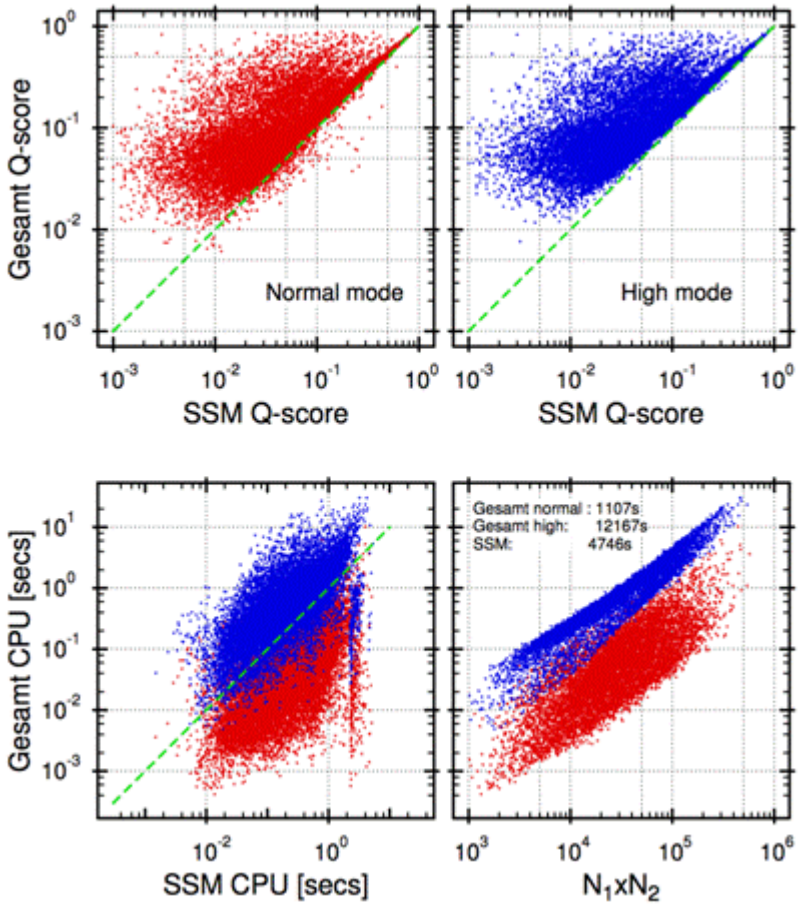


Figure 2. Comparative study of SSM and GESAMT performance on the same benchmark set as the one used in Fig. 1. Red dots correspond to GESAMT running in "Normal" mode, with parameters chosen for the optimal balance of speed and quality (as measured by Q-score). Blue dots correspond to GESAMT in "High" mode, where maximum quality is reached. Figures in low-right panel indicate gross timings (CPU clocks) for the algorithms to process all alignments in the benchmark set.

As any other algorithm of its kind, GESAMT has a few semi-empiric parameters that control the extensiveness of search in alignment field, and ultimately balance the achieved quality (as measured by Q-score) and computation time. For simplicity, these parameters have been combined in two sets, called "Normal" and "High" mode. In "Normal" mode, a reasonable balance between quality and speed is negotiated, while in "High" mode, quality considerations are ultimately preferred.

As seen from the top two panels in Fig. 2, GESAMT reaches considerably higher scores than SSM in most cases. This means GESAMT's alignments are longer at lower *rmsd*. Yet, some 5% of total alignments produced by GESAMT in "Normal" mode are lower than those achieved by SSM (cf. top-left panel in Fig. 2). These poorer alignments are attributed to the particular set of GESAMT's parameters, which allow some trade-in of quality for speed in "Normal" mode. As seen from the top-right panel in Fig. 2, these alignments improve greatly in "High" mode, where the quality of GESAMT's alignments is at worst equal to that of SSM.

The lower-left panel in Fig. 2 represents a direct comparison of GESAMT and SSM speed on the same benchmark set. As seen from the Figure, GESAMT is most often faster than SSM in "Normal mode" and most often slower than SSM in "High" mode. The timings in the lower-right panel suggest that, on average, SSM takes 0.3 secs per alignment, with "Normal mode" GESAMT 4.3 times faster, and "High mode" GESAMT 2.5 times slower

than that. These results indicate that a marginal quality decrease in “Normal” mode is accompanied by a 10-time gain in speed. It is also worth noting here, that this test is not truly indicative in respect to SSM speed. SSM was designed for mass-screening large databases, and allows for efficient precompilation of structural data. With this precompilation in force, SSM’s speed is significantly (20-30 times) faster than indicated in Fig. 2. This particular feature of the SSM algorithm cannot be used in pairwise comparison and is not engaged in CCP4’s SUPERPOSE. However, precompilation of structural data is an essential feature of the SSM web-server running at European Bioinformatics Institute (EBI) [10].

The lower-right panel in Fig. 2 presents complexity analysis for GESAMT. Theoretically, GESAMT’s complexity is estimated as $O(N_1 x N_2)$. Linear correlation between measured CPU time and the product of chain lengths is seen rather clearly in “High” mode, while “Normal” mode shows a higher extent of variation from the estimate. This is explained by the earlier mentioned fact that in “Normal” mode, GESAMT exercises greater liberty in pruning the search tree, subject to particular situation and structural features, which often results in better than theoretical complexity.

On the user side, GESAMT mimics SUPERPOSE, which means that it takes the same input and generates the same output. This was done intentionally in order to make switchover from SUPERPOSE to GESAMT as painless as possible for users and related applications.

This article must not be used as a reference for the GESAMT algorithm, and no materials/data from this communication can be used for benchmarking or any comparative studies, or referenced to, unless explicit permission is obtained from the author.

References

[1] Kabsch, W. (1976) Acta. Cryst. A32 922-923

[2] Diamond, R. (1992) Protein Sci. 1 1279-1287

[3] Krissinel, E., and Henrick, K. (2004) Acta Cryst. D60 2256-2268

[4] Vagin, A., and Teplyakov, A. (1997) J. Appl. Cryst. 30 1022-1025

[5] Kolodny, R., Koehl, P., and Levitt, M. (2005) J. Mol. Biol. 346 1173-1188

[6] Shindyalov, I.N., and Bourne, P.E. (1998) Prot. Enginrg. 11(9) 739-747

[7] Gerstein, M. and Levitt, M. (1996) Using iterative dynamic programming to obtain accurate pairwise and multiple alignments of protein structures. Proc. the 4th International Conference on Intelligent Systems for Molecular Biology, Menlo Park, Calif.: AAAI Press, 59-67

[8] Brenner S. E., Chotia C. and Hubbard T. J. P. (1998) PNAS 95, 6073-6078

[9] Ye Y. and Godzik A. (2003) Bioinformatics 19 Suppl 2 II246-II255

[10] http://www.ebi.ac.uk/pdbe/ssm .

Composite omit maps with 'comit'

K Cowtan

The 'comit' software calculates composite omit maps (Bhat, 1998) by omitting spheres of density from the asymmetric unit and reconstructing those spheres using sigma-a weighted maps to restore the omitted density. Unlike the existing omit map task, 'comit' uses refmac for its structure factor calculations, and thus makes full use of the latest features provided by 'refmac' (Murshudov et al., 1999).

The program can use one of two modes. A fast calculation using the output from refmac can generate a very good approximation to the full composite omit map with the correct choice of input MTZ columns. A slow calculation involves re-running refmac on multiple versions of the model with a few atom occupancies set to zero in each case. In slow mode the program is run twice, once to generate the input models for refmac, and once to assemble the output map from refmac. The two usage modes of 'comit' are shown in figure 1.

The slow mode, with refinement of all the omit models, should in theory lead to less bias. In practice, the fast mode produces similar results *if* the 'refmac' FC_ALL/PHIC_ALL columns are used for the initial map calculation (rather than FWT/PHWT which lead to a more biased and double-weighted map).

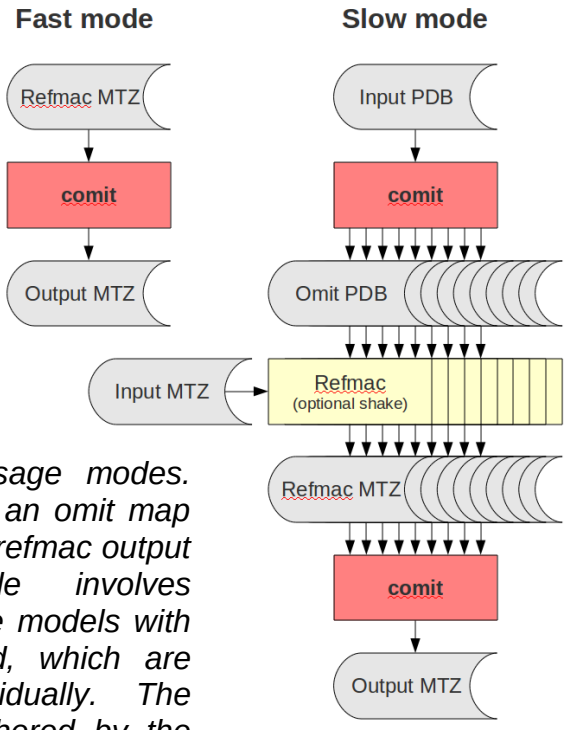


Figure 1: 'Comit' usage modes. Fast mode produces an omit map very quickly from the refmac output MTZ. Slow mode involves generation of multiple models with some atoms omitted, which are each refined individually. The results are then gathered by the program and used to compile the omit map.

The program is included in CCP4 6.3.0. Fast mode may be used through the graphical user interface. Slow mode currently requires a simple script; an example is provided in the documentation.

Citations for the use of comit should be made to the main CCP4 reference (Winn et al, Acta Cryst 2011).

References

- Bhat, T. N. (1988). J. Appl. Cryst. 21, 279-281.
- Murshudov G. N., Lebedev A., Vagin A. A., Wilson K. S. and Dodson E. J. (1999). Acta Cryst. section D55, 247-255.
- Winn M. D. et al. (2011). Acta Cryst. section D67, 235-242.

Nautilus software for automated nucleic acid building

K Cowtan

The 'nautilus' software is a new tool for automated building of RNA/DNA from electron density. It uses similar ideas to the 'buccaneer' software (Cowtan, 2008) for protein model building, but with a different and highly efficient target function for identifying nucleotide features.

The software will locate likely nucleotide features including sugars and phosphates, grow these into chains, merge overlapping chains, match the built chains to the sequence, and build the bases. The resulting structure is refined using 'refmac', and the calculation is iterated to obtain a more complete structure.

The software may be used to build nucleotide structure in experimentally phased maps, molecular replacement maps, or to add the nucleotide components to protein complexes.

The calculation consists of the following steps:

- Locate likely phosphate and sugar features in the electron density.
- Grow the features into chains.
- Merge overlapping chains.
- Join chains whose 3' and 5' ends are in close proximity.
- Rebuild the chains using fragments from the Richardson's database of well determined nucleotide structures.
- Match the chains against the known sequence.
- Build the bases and other peripheral atoms.

The calculation is notable for its speed, typically taking no more than a few tens of seconds. When combined with refinement in refmac (itself a fast package) (Murshudov et al, 1999), more than 90% of the time is spent in the refinement step. This speed is achieved through use a highly optimised 'fingerprint' for detecting structural features from the electron density values at a few highly informative points, which must have extreme density values if the feature is present in a given orientation. A fast rotation and translation search can be carried out using a method similar to the that employed by the ESSENS software (Kleywegt & Jones, 1997). The fingerprint for a sugar group is shown in figure 1.

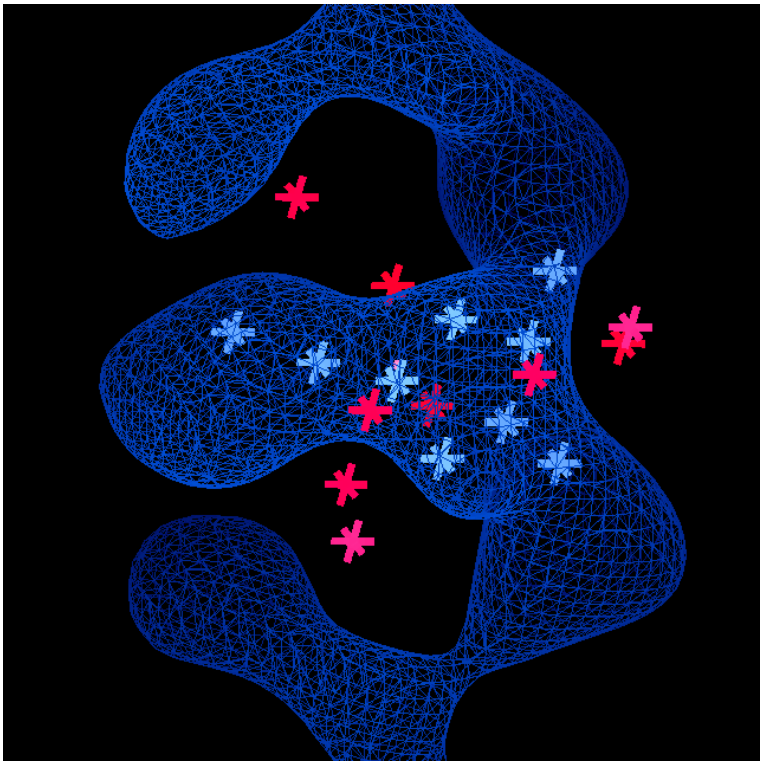


Figure 1: Nautilus sugar fingerprint. Blue crosses are locations where high density is required, red crosses are locations where low density is required.

Similar fingerprints are used to identify phosphates and to distinguish between different base types.

Version 0.3 of Nautilus is included in CCP4 version 6.3.0. The pipeline may be run through the CCP4i graphical user interface, or from the command line as either a build/refine pipeline or for a fast build only. In future an interactive version of some of the functionality will be available in the 'Coot' model building software (Emsley et al, 2010).

This article may be cited freely.

References

- Kleywegt, G. J. & Jones, T. A. (1997). Acta Cryst. D53, 179-185.
- Murshudov G. N., Lebedev A., Vagin A. A., Wilson K. S. and Dodson E. J. (1999). Acta Cryst. section D55, 247-255.
- Cowtan, K. (2008). Acta Cryst. D64, 83-89.
- Emsley, P., Lohkamp, B., Scott, W. G. & Cowtan, K. (2010). Acta Cryst. D66, 486-501.

Recent developments in the mosflm package

A.G.W. Leslie, O. Johnson and H.R. Powell
MRC Laboratory of Molecular Biology, Cambridge, UK

1. Introduction

The mosflm package for the integration of macromolecular diffraction data consists of two components, iMosflm (Battye et al., 2011) and ipmosflm (Leslie & Powell, 2007; Leslie, 2006). iMosflm (Fig. 1) is a Tcl/Tk based graphical user interface (GUI) that, via a series of panes, is designed to guide the user through the different steps in integrating a set of diffraction images. It allows inspection of the images and provides graphical feedback on the processing, for example by displaying the predicted reflection positions superposed on the diffraction image and plotting the variation in refined parameters and the standard reflection profiles. It also allows the user to change a large number of parameters that can influence the processing, to provide flexibility when dealing with particularly challenging datasets. iMosflm sends the necessary instructions to the ipmosflm background process that performs all the intensive computation. Information about refined parameters, standard profiles etc are passed from ipmosflm to iMosflm for display in the GUI. Data can be processed with ipmosflm alone by providing the necessary keyword commands, but this requires a much greater familiarity with the program than when using the iMosflm interface.

Both components are being continually developed and this article summarises the more recent developments that are available in the imminent release of ipmosflm version 7.0.9 (matched to iMosflm 1.0.7)

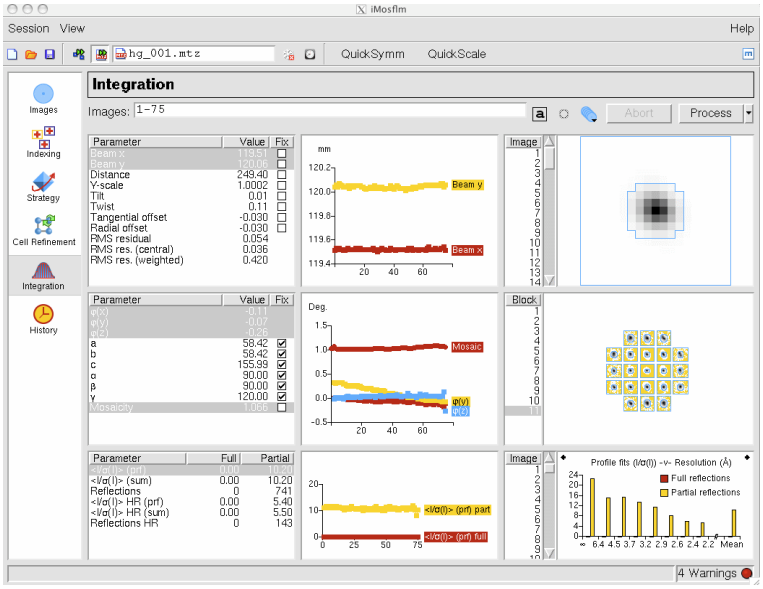


Figure 1. The Integration pane of the iMosflm graphical user interface.

2. Dealing with fine sliced, low exposure or weak diffraction images from Pilatus detectors

There were a number of issues that arose when processing Pilatus images collected with very short exposure times and very small oscillation angles, so that a significant number of pixels had values of zero. The refinement of the detector and crystal parameters were also sometimes unstable in such cases. These problems were mainly addressed in a beta release of iMosflm (version 1.0.6)/ipmosflm (version 7.0.8) in July 2011. This was a beta release because the problems were sufficiently serious to merit a new release, but there was insufficient time to carry out the usual full release testing. Further improvements in the processing of fine-sliced data have been made since that beta release.

Visualisation of the images is challenging when the exposure times are very short, with spots tending to vanish in the background noise, making it difficult to assess the quality of the diffraction. No satisfactory solution to this problem has been found yet, so in practice it is advisable to view a zoomed outer region of the image when deciding on the diffraction limit. The predicted pattern can also be difficult to see in such cases as the yellow boxes of the partial reflections become barely visible. There is now the option to change the colour of the boxes for the four different classes of reflection (full, partial, overlapped, too wide in phi) to overcome this problem (Fig. 2).

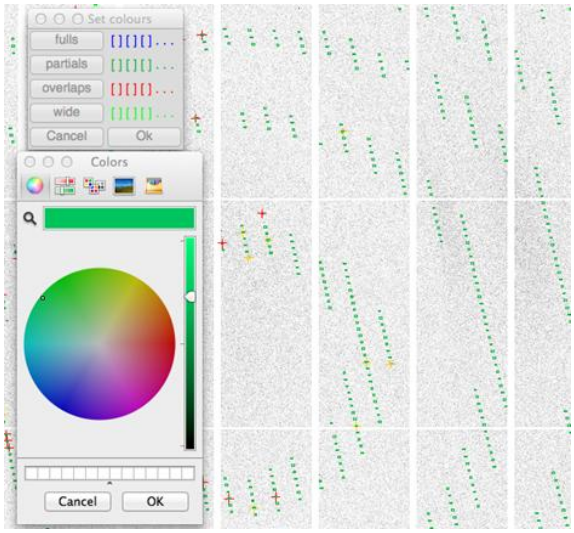


Figure 2. Colours can be chosen for the prediction boxes of each class of reflection.

Another issue arose when attempting to index diffraction patterns obtained from *in situ* samples at the Diamond Light Source. In these cases, the very small size and very low mosaicity of the crystals can result in diffraction spots that are only 1-2 pixels across. Changes to the spot finding algorithms were required to prevent these very small spots from being rejected, but once these were implemented indexing became straightforward.

In general, 3D profile fitting can offer advantages with very small oscillation angles (so-called fine phi slicing), but tests have shown that the 2D integration in mosflm provides excellent data quality even on challenging datasets (high mosaicity and weak diffraction) with oscillation angles as small as 0.1 degrees.

3. Avoiding corruption of the standard profiles by ice spots or “hot pixels”

The “standard profiles” used to evaluate the profile-fitted intensity are derived by the simple addition of all spots in a local region of the detector. As a result, it is possible for ice spots, zingers or single “hot” pixels (pixels that have an approximately constant value (that can be very large) on all images) to corrupt the standard profiles, especially if the diffraction spots are relatively weak.

An example of corruption by ice rings is shown in Fig. 3. To prevent this effect, two stages of filtering are applied when forming the standard profiles. Firstly, reflections that lie within a narrow resolution shell centred on the d-spacings for ice are omitted. The required width of the shell will depend on the strength of the ice diffraction and can be changed in iMosflm in the Processing Options. Secondly, in order to remove the influence of zingers or hot pixels, a small number of reflections that have the largest pixel intensity values are also excluded. This number can also be controlled, but defaults to 5%. As shown in Fig. 3, these steps are very effective in removing unwanted spots from the profiles.

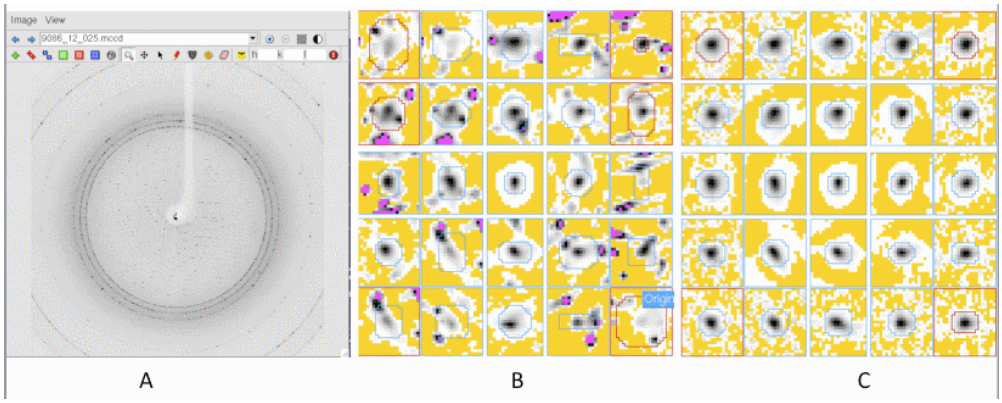


Figure 3. (A) Diffraction pattern showing strong ice rings. (B) Standard profiles without reflection filtering. (C) Standard profiles with reflection filtering.

4. An improved multi-crystal strategy option

The “Strategy” pane in iMosflm provides a straightforward means of calculating a geometrical strategy (ie a phi start and phi end) for cases where the user does not wish to collect a full 180 degree rotation, assuming that the correct Laue group is known. However, although possible in principle, it was not simple to use this option to devise a strategy when data were collected

from different crystals in different orientations (without the use of a multi-circle goniostat to allow re-orientation of the crystals).

A new strategy pane has been implemented that greatly simplifies this task (Fig.4). Briefly, the procedure is as follows. One or more reference images are read for the first crystal and indexed in the normal way. In the Strategy pane's Auto-complete menu, the user specifies how many degrees of data they expect to be able to collect from this crystal, and a start and end phi value for this crystal are calculated. Once these data have been collected, reference images for the second crystal are indexed. On entering the Strategy pane, the segment recommended (and assumed collected) for the first crystal will be displayed graphically. Running the strategy calculation again for the second crystal will then provide a start and end phi for the second crystal that will result in the highest possible completeness for crystals 1 and 2 combined. In Fig. 4, data have already been collected from two crystals (Ade12 and Ade16) and the strategy is being calculated to find the best 20 degree segment for the third crystal, Ade21.

The procedure can be repeated for further crystals. A summary of the recommended start/end phi values for each crystal are displayed at the top of the Strategy pane (Fig. 4) and this information can be saved and restored if it is necessary to exit the program. Furthermore, the phi values for each crystal can be updated graphically by manipulating the sector (wedge) displayed as a chart for the selected crystal. Thus if a particular crystal only provided 10 degrees of data instead of the expected 20 degrees, due to radiation damage, the phi end value can be updated and this revised value will be taken into account in future strategy calculations.

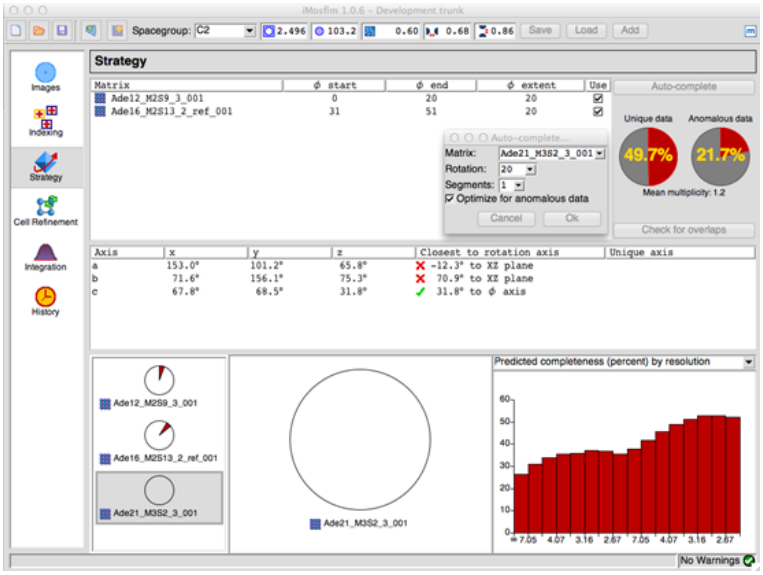


Figure 4. The new Strategy pane that simplifies strategy prediction when using multiple crystals for data collection.

5. The “mosaic blocksize” parameter

The “mosaic blocksize” parameter allows for the effect of very small mosaic blocks or domains on the dimensions of the reciprocal lattice spots and hence the reflecting range of Bragg reflections (Juers *et al*., 2007; Nave, 1998). In practice, it has the same effect as having a larger mosaic spread at low resolution than at high resolution. The effect of different mosaic blocksizes on the predicted reflections can easily be modelled in iMosflm by changing the default blocksize (100 microns) on the Images pane. As yet there is no procedure to refine this parameter, and the optimum value must be determined by running multiple processing jobs with different values and examining the merging statistics. Using too large a value will result in under-prediction of low-resolution reflections, which in turn can give higher values for Rmerge and the partial bias (FRCBIAS), while using too small a value will result in significant over-prediction at intermediate and higher resolutions, with resulting increases in Rmerge. A systematic investigation of the effect of varying the mosaic blocksize on a dataset from a crystal containing the lanthanide Praseodymium has shown that this parameter can have a significant effect on the anomalous signal at low resolution in addition to the other statistics (Table 1). In this case, a mosaic blocksize of 0.25μm gave the best anomalous correlation coefficients. Lack of time has not allowed a proper investigation of how this would affect *ab initio* structure determination (in this case the structure was determined by a combination of molecular replacement and SAD phasing), but the improvement in the anomalous correlation coefficient indicates that at least in marginal cases it is worth investigating the effect of optimizing this parameter.

Blocksize μm	Anomalous Correlation Coefficient			Rmerge			FRCBIAS		
	Overall	Low	High	Overall	Low	High	Overall	Low	High
100	0.421	0.552	0.077	0.168	0.073	0.401	-0.017	-0.039	0.043
10	0.414	0.640	0.074	0.171	0.076	0.402	-0.015	-0.043	0.064
1	0.454	0.684	0.058	0.168	0.072	0.404	-0.003	-0.044	0.087
0.75	0.454	0.660	0.075	0.170	0.071	0.414	0.001	-0.042	0.072
0.5	0.481	0.688	0.056	0.166	0.067	0.407	-0.003	-0.033	0.059
0.25	0.547	0.859	0.120	0.161	0.053	0.411	0.000	-0.013	0.041
0.15	0.445	0.797	0.046	0.183	0.050	0.499	0.001	0.001	0.060
0.1	0.459	0.833	0.030	0.186	0.052	0.439	0.009	0.017	0.052

Table 1. Statistics obtained from AIMLESS using different mosaic blocksizes in mosflm. The mid-bin resolutions for Low and High resolution bins in the Table are 7.16 Å and 3.44 Å.

6. Improved selection of the optimal indexing solution in cases of pseudosymmetry

In cases of pseudosymmetry, for example a monoclinic crystal with a β angle close to 90 degrees, or an orthorhombic crystal with very similar *a* and *b* cell dimensions, there can be an ambiguity in selecting the correct indexing solution. Previously the solution highlighted in the Indexing pane of iMosflm was selected based only on the indexing penalty. Careful analysis of a large number of reference images collected as part of the DNA project (Leslie *et al.,* 2002) demonstrated that the rms error (rmsd) in spot positions (σ(x,y) in iMosflm) can be used as a reliable indicator of the correct solution. In particular, if the rmsd for a particular solution is more than 1.3 times the rmsd of the triclinic solution, then the true symmetry is probably lower. A combination of the penalty value and the rmsd value are therefore now used to select the most likely indexing solution. At present, this is only implemented when the indexing is carried out from iMosflm, not when ipmosflm is used independently.

7. Mosaicity refinement

The correlation between cell parameters and mosaicity during post-refinement can result in the mosaicity refining to zero if the cell parameters are inaccurate. This problem has been minimized by keeping the mosaicity fixed during the initial cycle of cell refinement, and if there is a large shift in cell parameters the mosaicity is reset to its initial value for the next cycle. These changes have improved the reliability of the refinement, but there are still some circumstances in which the mosaicity can refine to either too large or too small a value, both in cell refinement and integration. This is typically associated with split diffraction spots or a combination of high mosaicity and large cell dimensions that results in adjacent reflections not being fully resolved in phi. In some cases, the only way to deal with this situation is to fix the mosaic spread at an appropriate value (estimated by visually comparing observed and calculated predictions), while in others assigning an appropriate mosaic blocksize can stabilize the mosaicity refinement. The partial bias statistic (FRCBIAS in SCALA or AIMLESS) should always be checked to see if there is evidence that the mosaic spread has been underestimated.

8. Further improvements

A variety of additional enhancements have been made. The most significant is a dramatic improvement in the speed of processing of datasets consisting of more than a few hundred images with iMosflm. Previously the rate of processing dropped dramatically after about 500 images, making the processing of fine sliced data very tedious. The rate of processing is now essentially constant when tested for over 1000 images. This improvement was present in the beta release of July 2011.

Phil Evans' program AIMLESS, a replacement for SCALA, is now the default when running the Quickscale option in iMosflm (but SCALA can be selected from the Processing Options). The data quality statistics are generally better using AIMLESS and it is typically three times faster than SCALA.

During post-refinement, there is now an option to select how the total intensity of a partially recorded reflection is divided up for use in post-refinement. Previously this was done so that approximately the same intensity was assigned to the two "parts" of the partial (corresponding to a PARTIAL value of 0.5) and this works well with "coarse sliced" data. However, for fine sliced data, the refined mosaicity tends to be too small using this approach, and a PARTIAL value of 0.25 works better, but this leads to unstable behaviour for "coarse sliced" images. Currently the default value depends on the oscillation angle and is set to 0.5 for oscillation angles greater than 0.25 degrees, 0.35 for angles greater than 0.15 degrees and 0.25 for angles less than 0.25 degrees, but can be set manually via the Processing Options.

The algorithm for estimating the mosaic spread has been improved so that it gives a more realistic estimate in cases of very high mosaicity.

Finally, some changes have been made to the estimation of the standard deviations in the intensities. In particular, the contribution of the detector error to the standard deviation (see 5.3 in Leslie, 1999) is still used for the statistics presented in iMosflm and the mosflm logfile, but it is not included in the standard deviations that are written to the output MTZ file. This contribution is now modelled by the SDADD term in SCALA or AIMLESS, which will therefore be systematically larger for data processed with the latest version of mosflm. This change was made because the contribution from this source of error was not properly modelled for partially recorded reflections and is better applied after the partial intensities have been summed. For one dataset with a weak anomalous signal that contained a large fraction of fully recorded reflections, this change made a significant difference to the success rate of the substructure determination.

9. Future developments

Significant progress has been made recently in the task of identifying and indexing multiple lattices and work is in progress to make this available in the iMosflm interface. This will be developed to allow the integration of each lattice separately, taking into account the presence of the other lattices. In the longer term, this will require a change in structure of the existing MTZ file format, to allow multiple indices to be assigned to a single intensity in order to deal with those cases where reflections from two lattices overlap.

A second topic under investigation is to speed up the integration by dividing a dataset into multiple blocks of images and integrating these in parallel, taking advantage of the fact that many machines now have multiple cpus. This has been demonstrated in principle, but two aspects require further work. Firstly, it is necessary for iMosflm to assemble the graphical output from each of the parallel jobs so that this can be presented in the GUI as if the integration were

done serially rather than in parallel. Secondly, extensive testing is required to ensure that the separate integration of many blocks of data does not have any adverse effect on data quality.

Finally, we hope to implement a “traffic light” style of representing the many warnings that mosflm can produce, where red would indicate a serious error, amber would mean that the warnings should be checked but are not serious, and green to indicate satisfactory processing. This would be linked to more detailed (and hopefully more understandable) error messages, and some progress has already been made in that respect.

Acknowledgements

We gratefully acknowledge all those who have provided useful feedback on the mosflm package, in particular Phil Evans, Graeme Winter, Olof Svensson and Frank von Delft. This work is supported by the MRC and BBSRC.

References

Battye, T.G.G., Kontogiannis, L., Johnson, O., Powell, H.R. & Leslie, A.G.W. 2011. iMosflm: a new graphical interface for diffraction image processing with MOSFLM. Acta Cryst. D67, 271-281.

Juers, D.H., Lovelace, J., Bellamy, H.D., Snell, E.H., Matthews, B.W. & Borgstahl, G.E.O. 2007. Changes to crystals of *Escherichia coli* β-galactosidase during room-temperature/low-temperature cycling and their relation to cryo-annealing. Acta Cryst D63, 1139-1153.

Leslie, A.G.W. 1999. Integration of macromolecular diffraction data. Acta Cryst. D55, 1696-1702

Leslie, A.G.W., Powell, H.R., Winter, G., Svensson, O., Spruce, D., McSweeney, S., Love, D., Kinder, S., Duke, E. and Nave, C. 2002. Automation of the collection and processing of X-ray diffraction data – a generic approach. Acta Cryst. D58, 1924-1928.

Leslie, A.G.W. 2006. The integration of macromolecular diffraction data. Acta Cryst D62, 48-57.

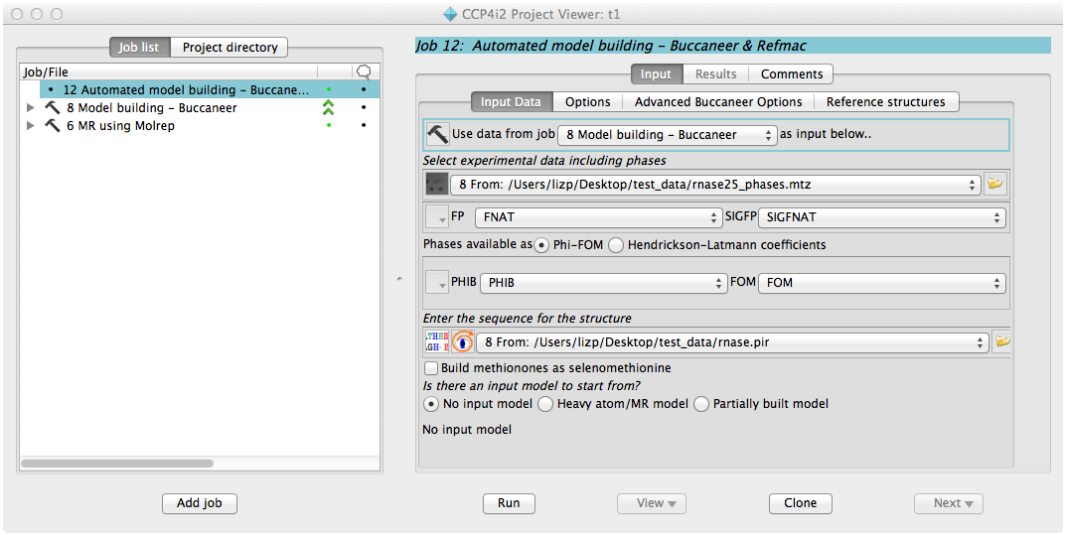
Leslie, A.G.W. & Powell, H.R. 2007. Processing diffraction data with MOSFLM. *in* Evolving Methods for Macromolecular Crystallography, Read R.J & Sussman, J.L. (eds), Springer Press, 41-51

Nave, C. 1998. A description of imperfections in protein crystals. Acta Cryst D54, 848-853.

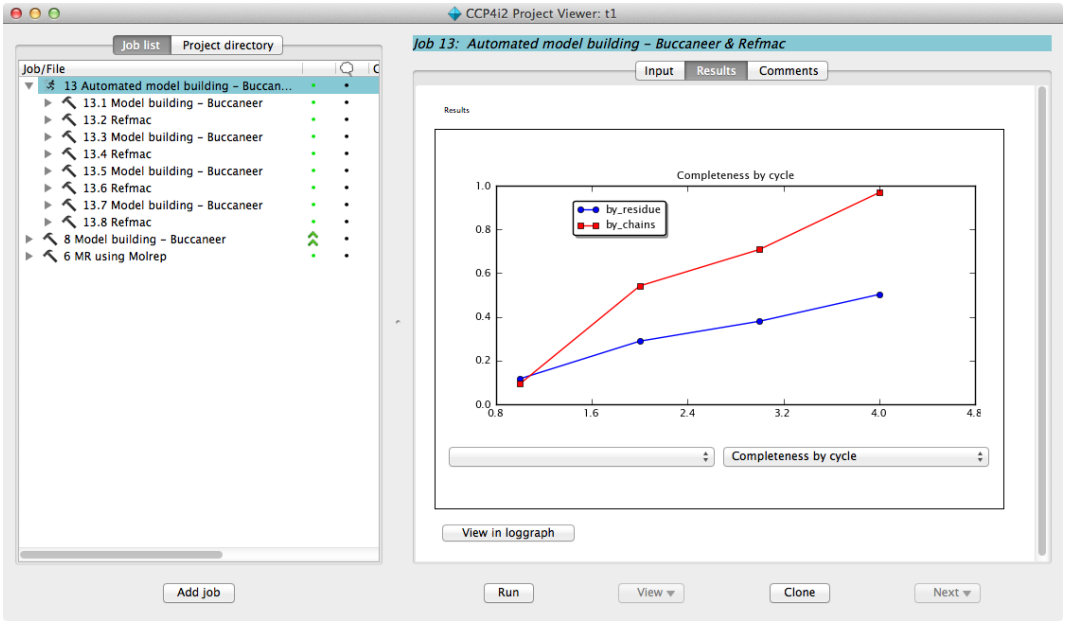
Introducing CCP4I2

Liz Potterton, March 2012

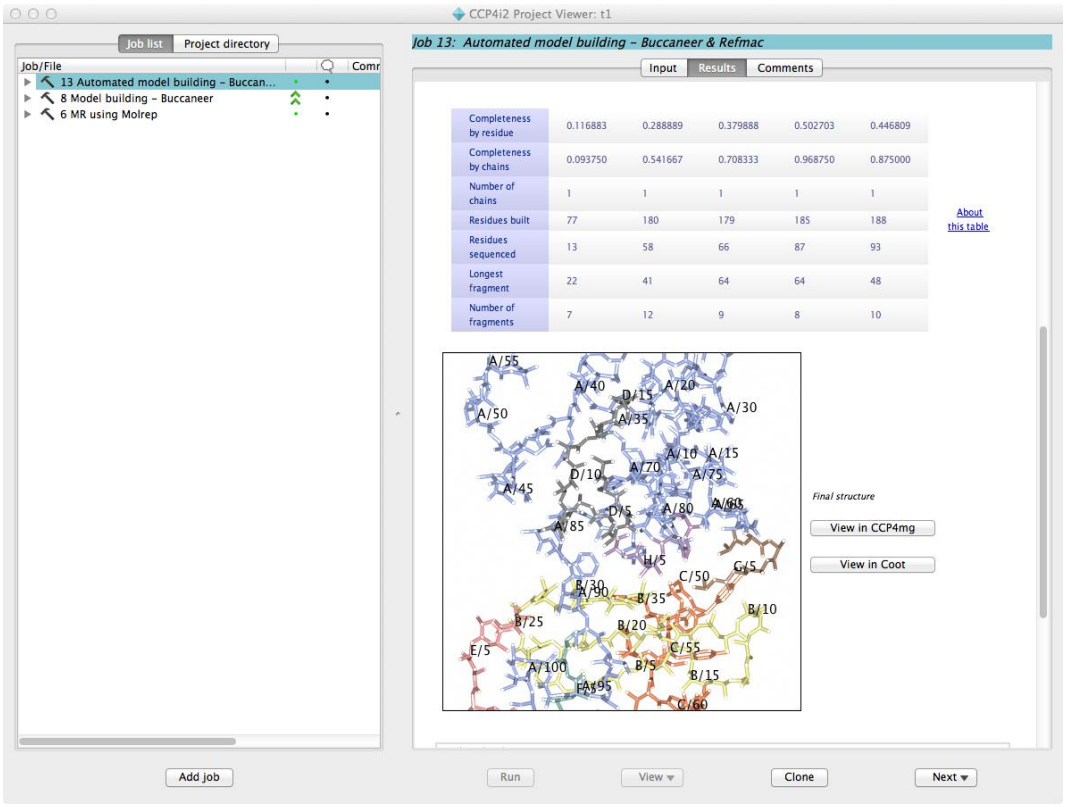
Here are a few screenshots of the new CCP4 GUI to give some idea how it will appear to crystallographers.



The main window of CCP4i2 with an overview of the jobs in the project on the left and an interface to starting an automatic model building job on the right. After this job is started the user can follow its progress.



The Buccaneer and Refmac sub-jobs are shown in the project window on the left and a job report is updated in real-time on the right. In this case the job report shows a graph of model completeness with cycle. When the job completes a fuller report is presented with graphs, tables, pictures, links and hints for what to do next.


CCP4i2 Project Viewer: t1
Job list
Project directory
Job/File
Comr
13 Automated model building – Buccan...
8 Model building – Buccaneer
6 MR using Molrep
Job 13: Automated model building – Buccaneer & Refmac
Input
Results
Comments
Completeness by residue 0.116883 0.288889 0.379888 0.502703 0.446809
Completeness by chains 0.093750 0.541667 0.708333 0.968750 0.875000
Number of chains 1 1 1 1 1
Residues built 77 180 179 185 188
Residues sequenced 13 58 66 87 93
Longest fragment 22 41 64 64 48
Number of fragments 7 12 9 8 10
About this table
Final structure
View in CCP4mg
View in Coot
Add job
Run
View
Clone
Next

CCP4i2 for Programmers

Liz Potterton

Introduction

This article will give an overview of the new CCP4i2 from the programmer's point of view and an outline of what is required to develop a new task interface. A good feature of the first Tcl-Tk based CCP4i was the ease with which developers with minimal knowledge of the Tcl language or the basic workings of CCP4i could create task interfaces. The same will be possible in the new Python based CCP4i2 and many features of the overall design of the new system will be familiar to developers who have worked with the old one.

The most important objective for the first release of the new GUI is to be easy-to-use for novice crystallographers; requiring minimal user input and providing simple job reports and clear hints on what to do next. To meet this objective a task developer does need to do some additional work in designing the best GUI and reports and providing more hints and error trapping.

As with the first CCP4i, a task in the new system requires two different sorts of script: one is a wrapper for the program and the other specifies the GUI for the task. The connection between the GUI and the scripts is a *def* file that specifies the data that appears in the GUI and is passed to the wrapper script. There was an equivalent file in the old system but the file format has changed to XML and there is a much stronger data typing in the new system.

Crystallographic Data Model and Python Data Classes

An important feature of the new system is a more rigorous data model for the crystallographic data. This is built up from basic Python classes such as *CInt* and *CList* (integer and list!) to representations of complex entities such as ensembles and rigid domains. Each class has qualifiers that provide information to enable better data validation (for example allowed value ranges) and defaults. In the core CCP4i2 there is a library of widget classes - one for each data class. So the GUI developer has a library of specialised widgets to create a task GUI and needs only to specify the layout and provide some explanatory text. The data and widget libraries are still being extended to cover new areas.

Another innovation is that the parameters must be clearly classified either as input data, output data or control parameters. The input data and output data should include the obvious input/output files but also any data specific to the crystal under study (e.g. NCS, heavy atoms etc). This classification makes it possible for the CCP4i2 system to automatically extract the input and output data for each job run and to save them to a database so the flow of data can be tracked.

The *def* file that specifies the data and parameters for a task must be organized into three containers for the three categories: *inputData*, *outputData* or *controlParameters*. Providing all the appropriate qualifiers will also help to improve the GUI. The *def* file, like all other utility files in the new system, is in XML format. There is a graphical utility, *defEd*, that developers can use to create and edit *def* files. This utility lists all the data classes (with some documentation) and also has an interface to enter the *qualifiers* for each selected class.

The *def* file specifies the parameters for each task and when a task is run (usually referred to as a *job*) the GUI creates an *input_params* file with the specific parameter values for the job.

Program Wrappers: *CPluginScript*

As with the first CCP4i each program run from the GUI must have a script wrapper that handles the input and output of that particular program. The Python base class for a wrapper is *CPluginScript* that provides functionality such as easy access to the contents of the *input_params* file and a mechanism to run programs as a separate process. It also has a method to return appropriate path names for files such as command files and log files and it is important to use this facility to ensure consistent file organization.

A task developer must sub-class *CPluginScript* but may need to do no more than provide a mechanism to write the command line and/or command file for the program. This can be done in either of two ways:

1. reimplementing *CPluginScript.makeComandAndScript()*
2. using command template file (equivalent to the *com* files for the first CCP4i)

Projects and the Database

As with the first CCP4i the user organizes work into projects. There is no fixed rule for what constitutes a project – one suggestion is that the result of a successful project is one structure for PDB submission! Projects can now be grouped into hierarchies so that it is possible to group together similar projects. Each project is associated with a project directory and within this directory each job has its own directory for input and output files and sub-jobs have sub-directories etc. The organization of directories and data files within a project is strictly controlled within CCP4i2. The destination of output files is determined automatically. So the user has no choice in naming output files but the GUI does provide tools to export files.
The database is used to keep track of projects and jobs but does not normally store crystallographic data – these remain in the PDB, MTZ and other files. The database is based on SQL and currently uses *sqlite* though it is intended that other SQL systems could be substituted in. The default arrangement is to have one database file per user with all of the users' projects stored in this

same file. Alternatives to this arrangement will be possible. There are mechanisms to allow colleagues access to your database.

From the developers perspective there is a Python module providing access to the database though implementing a task interface should not require direct database access as the database is automatically updated by the core CCP4i2 system.
But other programs will need access to the database in order to record aspects of the structure solution performed outside CCP4i2.

Job Reports

After each job the user can view a report file that should be a concise summary with information presented as graphically as possible. Reports can contain graphs, tables, pictures generated on the fly in CCP4mg, buttons to open Coot, CCP4mg or other programs and folders to show/hide more detailed information. The report is an HTML file but when it is displayed in the CCP4i2 browser it can contain specialized Qt widgets such as *Pimple* that is a Python/Qt/matplotlib-based replacement for *Loggraph,* implemented by Stuart McNicholas. We can provide other specialized widgets where necessary.

The mechanism to create job reports is an evolution from the *Baubles* system. The report is created automatically after the task has run based on a *report template file* that defines the layout of the report. The actual data that appears in the report is taken from XML files output by the program and/or the Python wrappers. Aside from needing to be XML there are no constraints on the content of the program/wrapper data files and current existing files can probably do the job. The *report template file* uses an xpath mechanism (www.w3schools.com/xpath) to specify the data to be extracted from the data files. There is a *report generator* utility that will process the *report template file* and read the necessary data files to create a report. The task programmer must provide the *report template file* and may need to add code to the program wrapper to analyse and output data for the report.

An Overview of ProSMART

Robert A. Nicholls, Marcus Fischer and Garib N. Murshudov
MRC Laboratory of Molecular Biology, Hills Road, Cambridge CB2 0QH
Email: nicholls@mrc-lmb.cam.ac.uk

"Procrustes owned two beds, one small, one large; he made short victims lie in the large bed, and the tall victims in the short one…" (Taleb, 2010)

Procrustes was a mythological Greek villain whose victims were stretched and cut in order to fit the shape of his bed. The *"Pro"* in ProSMART is due to its use of Procrustes analysis (Gower, 2010; Gower and Dijksterhuis, 2004; Catell and Hurley, 1962) for comparing local regions of structure between two protein chains. The name is fitting in this context due to manipulating the coordinates from one structure in order to optimally fit those in another, efficiently achieving a measure of local main-chain r.m.s.d. at the chosen level of structural resolution. By performing an exhaustive structural comparison of all local regions between two input structures, ProSMART is able to produce alignments by optimising the net agreement of local structures along the chain. The resulting alignment is thus independent of the global conformation of the compared chains - this is subsequently exploited for various purposes…

Contents

Introduction

ProSMART (Procrustes Structural Matching Alignment and Restraints Tool) has two main purposes: conformation-independent comparison of protein structures, and the

generation of interatomic distance restraints for subsequent use in macromolecular crystallographic refinement by REFMAC5 (Murshudov *et al.*, 2011, 1997; Nicholls *et al.*, 2012). Therefore, the tool comprises two major components:

- ProSMART ALIGN - for conformation-independent structural comparison;
- ProSMART RESTRAIN - for external restraint generation.

ProSMART is written in C++, takes one or many PDB files as input, and can be run from the command line or using a CCP4i (Potterton *et al.*, 2003) interface (see below). Mac/Linux and Windows versions are available.

This article gives a brief overview of some of the features available in ProSMART, a discussion regarding the generation of external structural restraints, an overview of the ways to run ProSMART, and the nature of the output.

Structural Comparison Features

Several features are available for performing different types of comparative structural analyses, depending on the level of structural similarity of the compared protein chains. ProSMART is particularly well suited to the comparative analysis of homologous chains in different global conformations, e.g. apo versus holo. Further to being able to achieve an alignment between similar structures, you could in principle use ProSMART to create an optimal alignment between completely dissimilar structures. The alignment achieved by ProSMART is effectively the optimal conformation-independent net agreement of local structures along the chain; alignment filtering according to local structural dissimilarity scores may be subsequently performed, if desired.

Further to achieving a conformation-independent alignment, ProSMART automatically performs identification and superposition of rigid substructures that are conserved between the compared chains.

ProSMART uses various residue-based scores for describing the dissimilarity of aligned residues' local structural environments. These scores include measures that are robust, allowing the identification of similarity in the presence of conformational change. Other scores are very sensitive, being able to detect subtle changes in the local structural environments that would otherwise be extremely hard to detect. These scores complement each other, maximising the amount of information achieved when performing such structural analyses (see Figure 2).

Another major feature of ProSMART is that the default generated output allows publication-quality illustrations to be quickly and easily achieved using the molecular graphics software CCP4mg (McNicholas *et al.*, 2011) or PyMOL (Schrödinger; DeLano, 2002), for various types of comparative structural analyses. For example, Figures 2-5 display default results from ProSMART in PyMOL, without any subsequent manipulation of superpositions or colouring (the illustrations were not tediously prepared by hand!).

The structural comparison methods used in ProSMART will be described in a future article (in the meantime, details of the methods are described by Nicholls, 2011). For the purposes of this article, we shall provide a few figures to briefly illustrate some of the functionalities that may be applied to highly homologous chain-pairs (thus would be of most relevance to cases involving external restraint generation).

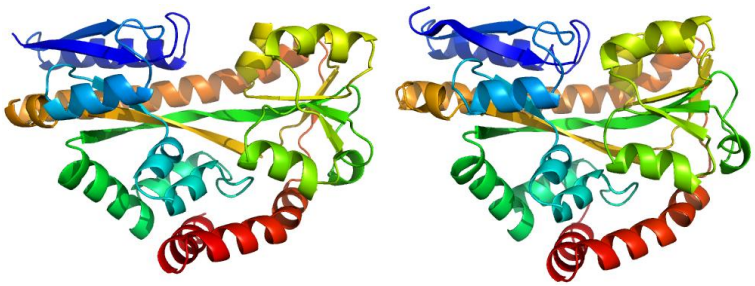


Figure 1 (stereo). Illustrations of sequence-identical chains, specifically the open (2cex_A, left) and closed (3b50_A, right) forms of the SiaP TRAP SBP (Fischer et al., 2010), rainbow-coloured along the chain.

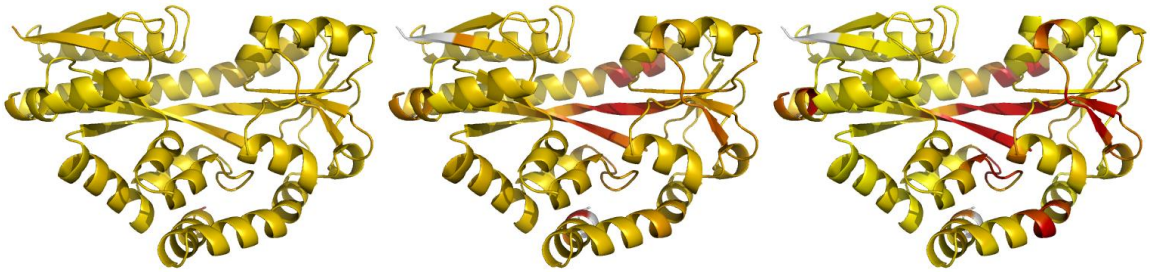


Figure 2 (stereo). Illustrations of simple results from the default ProSMART comparison of 2cex_A and 3b50_A, coloured using a gradual colour gradient according to main-chain dissimilarity scores (yellow implies similarity, red relative dissimilarity). For clarity, only the chain 2cex_A is shown (if shown, residues in 3b50_A would have been coloured the same as the corresponding residues in 2cex_A). These depictions allow quick visual identification of exactly which regions are structurally very similar, and which exhibit differences. The "minimum score" (left) is highly insensitive to global conformation - note that all residues are aligned and identified as very similar despite the global conformational change. The "central score" (middle) is more sensitive to differences in local structural environment - note that locally distorted regions such as the hinge are easily identified. The "intrafragment rotational dissimilarity score" (right) is sensitive to curvature and torsion of the local backbone - this score is useful for identifying regions that exhibit subtle backbone deformations that can be very hard to otherwise identify or quantify.

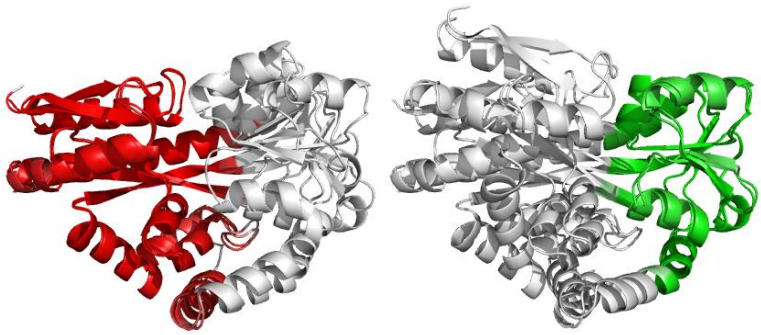


Figure 3 (stereo). Superpositions arising from the rigid substructure identification results from default ProSMART comparison of 2cex_A and 3b50_A, coloured according to cluster scores. Two rigid substructures identified, coloured red (left) and green (right), corresponding to the two domains.

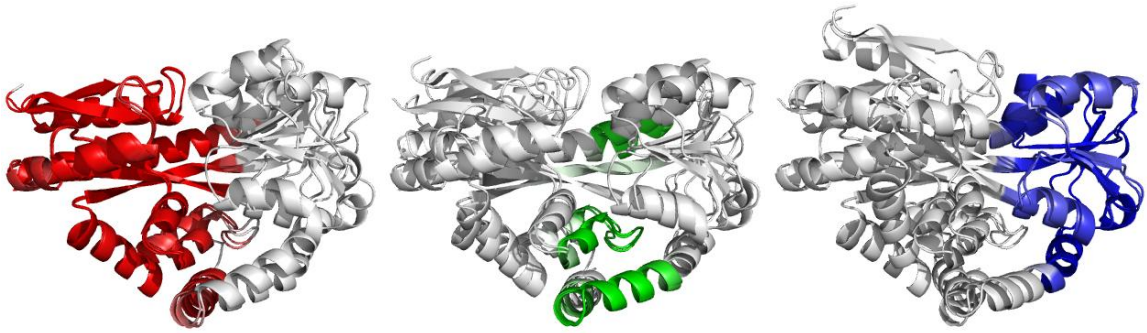


Figure 4 (stereo). Superpositions arising from the rigid substructure identification results from the ProSMART comparison of 2cex_A and 3b50_A using a fragment length of 7 residues (keyword: `-len`), coloured according to cluster scores. In contrast with Figure 3, which used the default fragment length of 9 residues, three rigid substructures are identified, coloured red (left), green (middle) and blue (right). These substructures correspond to two domains and the hinge region. This helps to illustrate the utility of performing comparative analyses at multiple levels of structural resolution.

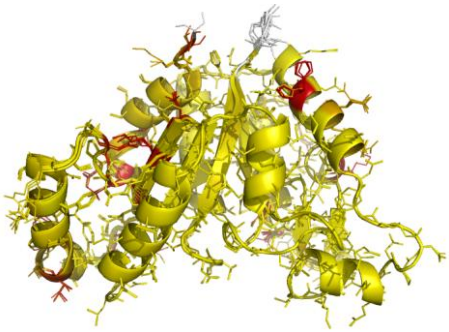


Figure 5 (stereo). Illustrations of superposed NCS-related chains in the structure of BioD with PDB code 3MLE (Porebski et al., 2012). Residues are coloured according to side-chain RMSD relative to the local coordinate frame, allowing easy visual identification of residues with side-chains in different conformations. This can be particularly useful for cross-validation during various stages of the refinement process (e.g. by identifying changes in side-chain conformation before/after refinement, and identifying which side-chains are/aren't pulled towards reference structures after application of external side-chain restraints).

External Restraints for use in Macromolecular Crystallographic Refinement

Information from well-refined higher-resolution structures may be used to improve reliability of low-resolution structures during refinement, provided that the reference structure is sufficiently similar to the target. When using such external information, there are various questions one should ask that might affect restraint generation, for example:

- How structurally similar is the external reference structure to the target structure being refined?
- Are the structures identical in sequence?
- Is the reference structure high-resolution and well refined? Should it be re-refined?
- Does it make sense to restrain side-chain atoms?
- If only main-chain restraints are to be used, is the backbone of the reference structure sufficiently well refined, compared with that of the target structure, in order for the target structure to benefit from the use of external restraints?
- If there is not a suitable reference structure, is it possible to generate generic restraints instead (e.g. for alpha-helices)?

Here, we discuss how ProSMART can be used to generate interatomic distance restraints for subsequent use in refinement by REFMAC5. Details of the methods used, along with examples of application, are described in an upcoming article (Nicholls *et al.*, 2012).

Choice of external reference structure(s)

External reference structures would usually be identical or close homologues, although in theory any structures could be used. There are no hard-coded limits on sequence identity, although intuitively only sufficiently similar structures should be used as external information. However, ProSMART is general and flexible, and provides the ability to allow user-input beyond the realms of common sense!

It is possible to use ProSMART to generate restraints using multiple reference structures. If multiple homologues are provided then REFMAC automatically selects the regions that are most consistent with the existing structure, only using the restraints that are closest to the current interatomic distances. This is done separately for each interatomic restraint. This means that if many structures are used as references, only the structure(s) most similar to the target should actually affect refinement, in theory. Nevertheless, we do not suggest that blindly using all homologues is a good strategy - manual consideration and common sense should always prevail! Carefully selecting one or a selection of homologous structures that are highly conserved in local structure would generally be a better approach, at present.

Note that the target and reference structures should be conserved in *local* structure - conservation of overall fold at the global level is neither necessary nor sufficient. To clarify, the conformation-independent approach of ProSMART means that sensible external restraints may be generated even if the target and reference structures adopt different global conformations (e.g. apo and holo forms). Indeed, the external restraints generated by ProSMART always operate locally, and thus they should not enforce global rigidity.

If a reference structure contains multiple (e.g. NCS-related / multimeric) chains in the PDB file then ProSMART will generate restraints for all target chains using information from all reference chains, by default. This may be desirable in some cases, for reasons outlined above. However, it is also possible to avoid this, and instead generate restraints only for the chain considered to be the best match to the target chain, in terms of net local structural similarity (keyword: `-restrain_best`).

It is important not to forget that reference structures may contain errors - the naïve application of restraints from such structures may cause errors to propagate into the target structure during refinement. Consequently, it is recommended to inspect reference structures, and possibly also consider their re-refinement before use, e.g. using PDB_REDO (Joosten *et al.*, 2009).

Selection of suitable structural information

It is recommended to perform a comparative structural analysis between target and reference structures prior to refinement (i.e. viewing the results from ProSMART ALIGN). This allows the user to visualise and quantify local (dis)similarities between the structures and thus make better-informed decisions regarding the local structural similarity of their presumed homologue to the target, prior to the application of the external restraints. It is also recommended for such comparative analyses to be performed following refinement, allowing the user to visualise and quantify the effect of the external restraints on local structure. In tandem with inspection of the electron density, this would help in deciding whether the external restraints were constructive (thus should be kept) or destructive (thus should be removed or replaced) in each local region, and also help in deciding the values of REFMAC5 external restraint weighting parameters (see below).

Ideally, the target and reference structures should be manually inspected, and the decision should be made as to whether restraints should or should not be generated for all regions. Challenging structures may require special attention, however, this should be deemed worth the effort when the alternative is to produce a better diffracting crystal. If there are some regions that are actually different between the two structures, and it is decided that external restraints should not be generated for these regions, then the restraints corresponding to these residues/regions can be removed (keywords: `-restrain` and `-restrain_rm`). Additionally, it is possible to specify that restraints should only be generated for regions that are sufficiently conserved, in terms of local main-chain (keyword: `-cutoff`) and/or side-chain (keyword: `-side_cutoff`) similarity. By default, restraints are generated for all aligned portions of structure, regardless of local structural conservation - the user must decide whether dissimilarity thresholds are appropriate in the particular case. Alternatively, PDB files may be separately filtered in a way deemed appropriate for

subsequent restraint generation, either manually or using a tool such as Tim Grüne's mrprep (Grüne, 2012).

Note that restraints can be generated only for main-chain chain atoms (keyword: `-main`), or for both main-chain and side-chain atoms (keyword: `-side`).

Important REFMAC5 parameters

At present, caution is due when optimising certain parameters in REFMAC5, most notably the external restraints weight (keyword: `EXTERNAL WEIGHT SCALE`) and Geman-McClure parameter (which down-weights outliers, keyword: `EXTERNAL WEIGHT GMWT`), in order to successfully apply external restraints during refinement. For more information, see Nicholls *et al.* (2012). Information on how to provide REFMAC5 with such keywords can be found here and here; they can also be specified using the CCP4i REFMAC5 GUI (see below).

Appropriate values for these parameters will vary for each case, and will depend on many factors. Such factors may include properties of the target structure's refinement in the absence of external restraints (e.g. X-ray resolution, quality of electron density, geometry weight, number of NCS-restrained chains) and also properties of the external structural information, such as the quality of the external information (i.e. reference structures), and the type of external restraints (e.g. main-chain, side-chain, generic SS H-bond or fragment-based restraints), etc.

As an aside, it is important to make sure that sufficient cycles of REFMAC5 are executed when using external restraints - the external restraints will seem to have little effect if running only 5 refinement cycles. Something like 20-30 cycles may be required, and even more if also using jelly-body restraints (Murshudov *et al.*, 2011).

Generic fragment and secondary-structure-based restraints

Further to generating restraints using homologous protein structures, ProSMART allows the generation of restraints to specific *n*-residue structural fragments (keyword: `-lib` to use all fragments present in the local library). Such generic fragment-based restraints can be used to help the structure better-adopt a desired conformation, for example, using an ideal helical fragment to help stabilise the formation of a helix. The implemented approach is generalised, allowing any structures to be used as reference fragments in principle (e.g. the user may provide their own fragment coordinate files). At present, ProSMART comes with fragments representing an ideal alpha-helix (keyword: `-helix`) and a representative beta-strand (keyword: `-strand`), which can be used to generate in-sequence quasi-secondary-structure restraints. Note that these restraints are not considered true secondary-structure restraints, since they restrain all atom-pairs in the local structural environment, rather than just those considered to be hydrogen-bonded (also, note that the residue alignment is not assigned using a traditional method such as DSSP). This method is considered powerful, since it does not require the low-resolution structure to be sufficiently well modelled to assign secondary-structure using hydrogen-bonding patterns. For example, you may want to use helix restraints from

ProSMART to force a particular troublesome helix to maintain a reasonable helical conformation during early/intermediate stages of model building/refinement.

Further to existing functionalities, generic restraints representing specific atomic interactions, notably hydrogen bonds (e.g. in alpha-helices, across beta-sheets), and also external restraints for DNA/RNA from homologous structures, will be available imminently in an upcoming version of ProSMART (which should be available by the time you read this!).

How to Run ProSMART

Please contact the authors for any assistance or information (see here).

Command line

Basic instructions on how to run ProSMART using a command line interface can be found here.

Typing `prosmart` will give a list of all keywords in that version. A list of keywords with more detailed descriptions can be found here. Note that keywords can be specified in an external text file (keyword: `-f`), which can be useful if the arguments list gets too long.

Instructions on how to use a command line interface to run REFMAC5 with external restraints generated by ProSMART can be found here.

CCP4i REFMAC5 GUI

An updated version of the REFMAC5 CCP4i GUI, which includes the ability to automatically run ProSMART for external restraint generation, has been developed by Martyn Winn at STFC Daresbury. The updated interface is currently available, and will be distributed in the next release of the CCP4 suite (version 6.3). The REFMAC5 GUI allows you to specify for ProSMART to be automatically run (using mainly default settings) prior to running REFMAC5 with the generated external restraints. Note that this new feature will be hidden if ProSMART is not installed.

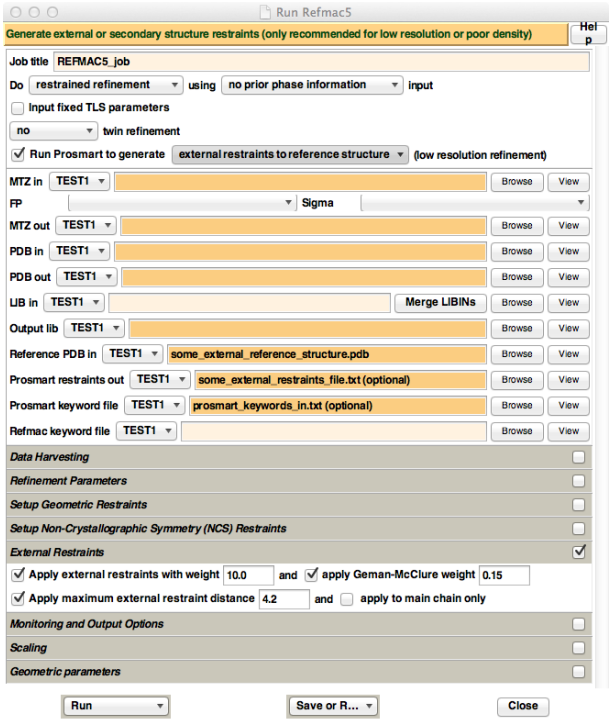


Figure 6. Appearance of the new REFMAC5 CCP4i GUI, when ProSMART is installed. For illustrative purposes only, text has been inputted into the "Reference PDB in", "Prosmart restraints out", and "Prosmart keyword file" fields - these are the three fields that appear when enabling the "Run Prosmart to generate" option. Also, the "External Restraints" tab has been opened, which contains options that control the way REFMAC5 uses the external restraints generated by ProSMART.

To automatically generate and use ProSMART external restraints, make sure that the *"Run Prosmart to generate"* button is ticked. This will cause three input file options to appear: *"Reference PDB in"* (analogous to the `-p2` keyword), *"Prosmart restraints out"*, and *"Prosmart keyword file"* (analogous to the `-f` keyword). For simple execution, only the *"Reference PDB in"* has to be specified. For more advanced functionality, create a simple text file containing the desired ProSMART keywords, and pass this file to the GUI in the *"Prosmart keyword file"* field.

The REFMAC5 GUI has four options controlling behaviour of the external restraints during refinement, namely the external restraints weight, Geman-McClure parameter, maximum external restraints distance, and whether or not side-chain atoms are to be restrained in addition to main-chain atoms. These options can be found in the *"External Restraints"* tab in the GUI. Unless there is a very good reason for doing otherwise (or just want to experiment!), it is highly recommended to enable *"Apply maximum external restraint distance"*, and set it to 4.2 (which is a magic number that tends to always be approximately optimal). Other parameters should be experimented with; see Nicholls *et al.* (2012) for more information.

As an aside, it may be advisable to select *"Run&View Com File"* instead of *"Run"* from the bottom-left drop-down box. This will display the command (with keywords) used to run REFMAC5 before actual execution - this will confirm whether REFMAC5 is being run as intended!

How to ensure that the external restraints are being used by REFMAC5 during refinement

To ensure that CCP4i is running ProSMART and REFMAC5 as intended, it is recommended to inspect the output log file to see exactly what command was used to run ProSMART, and confirm that the ProSMART job completed successfully. The log file can be accessed from the main CCP4i GUI by double-clicking the appropriate job (or alternatively selecting *"View Files from Job"* then *"View Log File"*). For example, at the top of this log file you should see something like this:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
* Information from CCP4Interface script
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

*** Starting Prosmart to determine restraints to external structure ***
 Using command: prosmart -p1 "some_pdb.pdb" -o "some_location" -side -p2
"another_pdb.pdb"
 Writing results to directory "some_directory"

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

This is then followed by the main ProSMART log. Upon successful completion, the following lines (or similar) will be displayed after the ProSMART log:

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
* Information from CCP4Interface script
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

 *** Prosmart finished ***
  Copying file of restraints some_file.txt to another_file.txt

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

To confirm that the external restraints are being used, locate the restraints table in the log file (just before refinement cycle 1), which will look something like this:

```
---------------------------------------
                        Standard  External       All
               Bonds:       7711     22212      29923
              Angles:      13788         0      13788
             Chirals:        706         0      706
              Planes:       1372         0      1372
            Torsions:       3296         0      3296
---------------------------------------
```

Usage of the external restraints is confirmed by the fact that the number of external bonds is non-zero (and generally quite large - in this case, 22212).

As of REFMAC5 version 5.7.0022, more output regarding external restraints will be available in the log file, providing information about the input parameters and options used. This information will look something like this (located above the standard restraints table):

```
------------------------------------------------------------------------------
 External restraints group   :            1
 External restraints file    :input_keywords
 Fail if one of the atoms involved in the restraints is missing in the pdb file
 Use restraints for all defined atoms
 Ignore restraints if abs(dmod-drest) >    50.000000     *sigma
 Ignore retraints if input dist >   1.00000003E+32
 Weight scale sigmas         :   1.0000000
 Weight min sigma            :   0.0000000
 Weight max sigma            :   100.00000
 GM parameter                :  0.10000000
 Number of distances         :       10809
 Number of angles            :           0
 Number of torsions          :           0
 Number of planes            :           0
 Number of chirals           :           0
 Number of intervals         :           0
------------------------------------------------------------------------------
```

How to use the REFMAC5 GUI with a pre-generated ProSMART restraints file

It is often desirable to run ProSMART separately (e.g. from the command line, or the ProSMART GUI) instead of running ProSMART automatically using the REFMAC5 GUI. In this case, it is necessary to provide the REFMAC5 GUI with the external restraints file generated by ProSMART. This restraints file should be provided to the GUI in the *"Refmac keyword file"* field. Note that, in this case, the *"Run Prosmart to generate"* button should not be ticked.

For more control over how REFMAC5 deals with the external restraints, it is advised to create a REFMAC5 external keywords file that specifies the location of the ProSMART restraints file, and then pass this keywords file to the GUI using the *"Refmac keyword file"* field (latest versions of REFMAC5 only). To do this, create a simple `.txt` file (note: must be plain text, not rich text) that contains the commands to tell REFMAC5 to use the ProSMART restraints file. For example, this file may simply contain the line:

```
@my_prosmart_restraints_file.txt
```

where `my_prosmart_restraints_file.txt` is the name/location of the restraints file generated by ProSMART. The `@` symbol specifies for REFMAC5 to parse the `my_prosmart_restraints_file.txt` file and use any external restraints found.

In practical application, it is necessary to tell REFMAC5 how to deal with these restraints, e.g. what weighting parameters to use. This can be achieved by specifying `EXTERNAL` keywords. For example, the options enabled in the *"External Restraints"* tab in the GUI shown in Figure 6 would be specified:

```
EXTERNAL DMAX 4.2
EXTERNAL WEIGHT SCALE 10
EXTERNAL WEIGHT GMWT 0.15
@my_prosmart_restraints_file.txt
```

Note that any `EXTERNAL` keywords must be specified *above* the `@my_prosmart_restraints_file.txt` line.

In addition, the `EXTERNAL USE MAIN` keyword can be used to discard any side-chain restraints that may be present in the external restraints file (this equivalent to the *"apply to main chain only"* option in the REFMAC5 GUI). Of course, regardless of whether this keyword is specified, side-chain atoms will only be restrained if you tell ProSMART to generate side-chain restraints (keyword: `-side`). The `EXTERNAL USE ALL` keyword may be specified to ensure that all external restraints present in the ProSMART restraints file are used.

As of REFMAC5 version 5.7.0022, an additional keyword `EXTERNAL USE HBOND` may be specified, which only uses external restraints that may form hydrogen bonds (i.e. restrained atom-pair are donor and acceptor); all other restraints present in the input file will be ignored. Note that only one of `EXTERNAL USE MAIN`, `EXTERNAL USE ALL`, and `EXTERNAL USE HBOND` may be specified (per external restraints file).

For clarification of the format of REFMAC5 keywords files, here is an example keyword file.

CCP4i ProSMART GUI

A ProSMART GUI for CCP4i is available from the Murshudov Group website (under Software→ProSMART→Download ProSMART). Installation instructions can be found here. This GUI provides a user-friendly alternative to running ProSMART from the command line, providing near-comprehensive functionality for both ProSMART ALIGN (structural comparison) and ProSMART RESTRAIN (external restraint generation) features. Note that restraints generated using this GUI can be passed to the REFMAC5 GUI via an external keywords file, as described above.

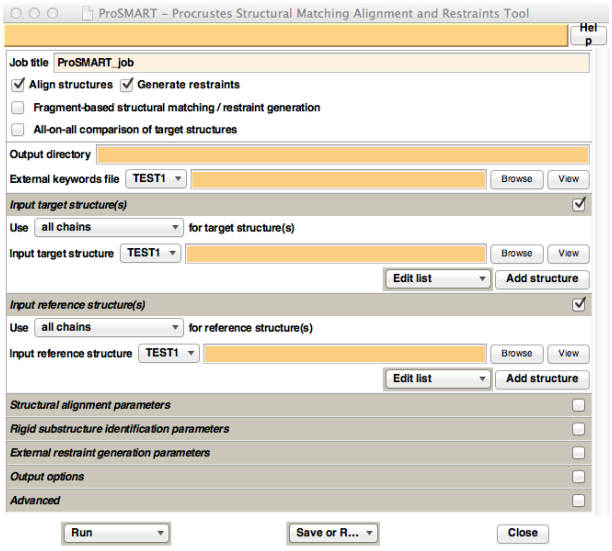


Figure 7. Appearance of the ProSMART CCP4i GUI.

ProSMART Output

After running ProSMART, there are two major sources of general results information: the main ProSMART log file, and the ProSMART HTML output page.

The main ProSMART log file indicates what options were used (e.g. whether or not side chain restraints were generated), which chain-pairs were considered, and whether the job finished successfully. In the CCP4i GUIs, this log file can be accessed by double-clicking the appropriate job (or alternatively selecting *"View Files from Job"* then *"View Log File"*). If using a command line interface, this information is printed to screen.

The ProSMART HTML output page allows navigation of all major results files, provides an easy way of viewing the log files from all pairwise executions of ProSMART ALIGN and ProSMART RESTRAIN, and provides a list of the program options used (which can be useful for future reference). The HTML results page may be accessed from the ProSMART CCP4i GUI under *"View Files from Job"*. Otherwise, this results file may be found in the ProSMART output directory (which by default would be here: `ProSMART_Output/ProSMART_Results.html` relative to the current working directory).

The major output files generated by ProSMART that can currently be viewed from the HTML output page include:

- Residue-based alignment, including various local conformation-independent structural dissimilarity scores;
- Global alignment score matrices, which include measures that are independent of global conformation;
- Transformations required to superpose structures, based on both global alignment and rigid substructure identification features;
- PDB-format files containing the structures in the coordinate frames of the (potentially various) superpositions;
- Colour scripts, used for visualisation of residue-based dissimilarity scores and rigid substructure belongingness;
- Interatomic distance restraint files, both for individual chain-pairs, and concatenated for all chains.

The output superposed PDB files may be viewed using PyMOL (Schrödinger; DeLano, 2002). The output PyMOL colour scripts may be used to colour residues according to their various dissimilarity scores (including those illustrated in Figures 2-5).

The latest release of CCP4mg (McNicholas *et al.*, 2011) includes an experimental ProSMART analysis feature that allows results from a ProSMART execution to be loaded for a powerful interactive illustration of the comparative analyses. Unlike with PyMOL, CCP4mg does not use/require ProSMART to provide superposed PDB files and colour scripts in order to achieve the desired effect. The ProSMART transformation files are used to superpose both global alignments and any identified

rigid substructures; these structures may be coloured according to their residue-based dissimilarity scores, with colour gradients altered in real-time using a slider.

Availability and Contact Information

ProSMART will be distributed as part of the CCP4 suite (Winn *et al.*, 2011) in the upcoming release (version 6.3). Latest versions, along with simple installation instructions and documentation, are always available from the Murshudov Group website: http://www2.mrc-lmb.cam.ac.uk/groups/murshudov/ (under Software→ProSMART).

For more information, please contact: nicholls@mrc-lmb.cam.ac.uk. Any comments or questions are always welcome!

This article may be freely cited and referenced, although it is preferred that references to restraint generation using ProSMART be made to Nicholls *et al.*, 2012.

Acknowledgements

We would like to thank Martyn Winn for developing the REFMAC5 GUI, Stuart McNicholas for working on the ProSMART analysis features in CCP4mg, Marcin Wojdyr and Charles Ballard for providing advice on software and installation issues and working on the integration of ProSMART into CCP4, and CCP4 for support and distribution. We would also like to thank our colleagues for interesting discussions, and the many users who have provided useful feedback, resulting in greatly improved functionality.

This work was supported by the Medical Research Council (grant number: MC US A025 0104). Part of this work was carried out whilst the authors were at the Structural Biology Laboratory, Department of Chemistry, University of York, during which time RAN was funded by a BBSRC Ph.D. Studentship, MF was funded by a Wild Fund Scholarship and a BBSRC Ph.D. Studentship, and GNM was funded by the Wellcome Trust.

References

Catell, R.B. and Hurley, J.R. (1962) The Procrustes program in producing direct rotation to test a hypothesized factor structure. Behavioural Science 7, 258-262.

DeLano, W.L. (2002) The PyMOL Molecular Graphics System.

Fischer M., Zhang Q.Y., Hubbard, R.E. and Thomas G.H. (2010) Caught in a TRAP: substrate-binding proteins in secondary transport. Trends in Microbiology 18(10), 471-478.

Gower, J.C. (2010) Procrustes methods. Wiley Interdisciplinary Reviews: Computational Statistics 2(4), 503-508.

Gower, J.C. and Dijksterhuis, G.B. (2004) Procrustes problems. Oxford University Press, USA.

Grüne, T. (2012) mrprep - PDB preparation tool for use with ProSmart or for Molecular Replacement. http://shelx.uni-ac.gwdg.de/~tg/research/programs/mrprep/

Joosten, R.P., Salzemann, J., Bloch, V., Stockinger, H., Berglund, A.C., Blanchet, C., Bongcam-Rudloff, E., Combet, C., Da Costa, A.L., Deleage, G., Diarena, M., Fabbretti, R., Fettahi, G., Flegel, V., Gisel, A., Kasam, V., Kervinen, T., Korpelainen, E., Mattila, K., Pagni, M., Reichstadt, M., Breton, V., Tickle, I.J., Vriend, G. (2009) PDB_REDO: automated re-refinement of X-ray structure models in the PDB. Journal of Applied Crystallography 42(3), 376-384.

McNicholas, S., Potterton, E., Wilson, K. S. and Noble, M. E. M. (2011) Presenting your structures: the CCP4mg molecular-graphics software. Acta Crystallographica D67, 386--394.

Murshudov G.N., Skubak P., Lebedev A.A., Pannu N.S., Steiner R.A., Nicholls R.A., Winn M.D., Long F. and Vagin A.A. (2011) REFMAC5 for the Refinement of Macromolecular Crystal Structures. Acta Crystallographica D67, 355-367.

Murshudov G.N., Vagin A.A. and Dodson E.J. (1997) Refinement of Macromolecular Structures by the Maximum-Likelihood Method. Acta Crystallographica D53, 240-255.

Nicholls R.A. (2011) Conformation-Independent Comparison of Protein Structures. University of York (thesis). http://etheses.whiterose.ac.uk/2120/.

Nicholls R.A., Long F. and Murshudov G.N. (2012) Low resolution refinement tools in REFMAC5. Acta Crystallographica D68, 404-417.

Porebski P.J., Klimecka M., Chruszcz M., Nicholls R.A., Murzyn K., Cuff M.E., Xu X., Cymborowski M., Murshudov G.N., Savchenko A., Edwards A. and Minor W. (2012) Structural characterization of Helicobacter pylori dethiobiotin synthetase reveals differences between family members. FEBS Journal.

Potterton E., Briggs P., Turkenburg M. and Dodson E. (2003) A graphical user interface to the CCP4 program suite. Acta Crystallographica D59, 1131-1137.

Schrödinger, LLC. The PyMOL Molecular Graphics System, Version 1.5.0.1.

Taleb, N.N. (2010) The bed of Procrustes: philosophical and practical aphorisms. Random House.

Winn, M.D., Ballard, C.C., Cowtan, K.D., Dodson, E.J., Emsley, P., Evans, P.R., Keegan, R.M., Krissinel, E.B., Leslie, A.G.W., McCoy, A., McNicholas, S.J., Murshudov, G.N., Pannu, N.S., Potterton, E.A., Powell, H.R., Read, R.J., Vagin, A. and Wilson, K.S. (2011) Overview of the CCP4 suite and current developments. Acta Crystallographica D67, 235-242.

Space group validation with *Zanuda*

Andrey A. Lebedev [1], Michail N. Isupov[2]
[1]CCP4, STFC Rutherford Appleton Laboratory, Research Complex at Harwell, Harwell Science & Innovation Campus, Didcot OX11 0FA, England
[2]Henry Wellcome Building for Biocatalysis, College of Life and Environmental Sciences, University of Exeter, Stocker Road, Exeter EX4 4QD, England

Introduction

The presence of pseudosymmetry and especially its interplay with twinning may lead to an incorrect space group assignment. Moreover, if the pseudosymmetry is very close to an exact crystallographic symmetry, the structure can be solved and partially refined in the wrong space group. Typically in such false structures all or some of the pseudosymmetry operations are treated as crystallographic symmetry operations and vice versa. Such misassignment is not uncommon when the structure is solved by molecular replacement (MR) and it only becomes apparent, when the R-free ceases to decrease at about 35% or even at a higher value, and no further model rebuilding and refinement can improve it. At this point the electron density map remains imperfect (breaks in the main chain electron density, poor solvent peaks) while does not suggest any particular ways of model improvement.

The program *Zanuda* presented in this article was developed to automate the validation of space group assignment in such circumstances. In addition, the program can be used to restore the correct space group in structures which were intentionally solved in low symmetry space groups including *P*1. The validation is based on the results of a series of refinements in space groups, which are compatible with the observed unit cell parameters. Two assumptions are made in this method. Firstly, the pseudosymmetry operations, if any, are close enough to exact symmetry operations and, therefore, refinement converges to the global minimum when wrong symmetry constraints are removed and correct constraints are imposed. Secondly, the errors in individual macromolecules do not hinder the difference between pseudosymmetry and crystallographic symmetry, *i.e.* the model is already refined well enough (R-free around or below 40%). However it is not assumed that this refinement has been performed in the true space group.

Program usage

Zanuda is a Python script wrapping *Refmac* [1], several CCP4 [2] programs for handling MTZ-files and one purpose-written *FORTRAN* program for analysis of the pseudosymmetry in the input model and conversion of the model and data into possible space groups. *Zanuda* is included in CCP4 release 6.3.0. Its CCP4I task window (Fig. 1) can be opened from Validation & Deposition folder of the CCP4I GUI (task name "Validate space group") or from Program List folder (task name "Zanuda"). *Zanuda* summary file (Fig. 2) is explained further in the text. Originally the program was designed for YSBL server [3], where it runs in the default mode.

The program reads an input model and experimental data from files in PDB and MTZ formats, respectively. Both files are mandatory and must refer to the same space group and unit cell parameters. The input experimental data are to be presented as the observed structure amplitudes (not as intensities). Readability test is performed using *Refmac* in the mode of map coefficient calculation (zero cycles of restrained refinement). If the input files have not passed this test, the program stops and a user is prompted to correct or replace the input files and make sure that *Refmac* can read them.

The program has two modes. In the default mode it refines a series of models using *Refmac* and selects a model with highest symmetry from the ones with best refinement statistics. The program output includes this model in PDB format and a corresponding MTZ file from *Refmac* containing experimental data and map coefficients. Important, the transformed data in the output MTZ file are generated from already merged input data and should not be used at late stages of refinement; by no means can they be used for the PDB deposition. For these two purposes the experimental data are to be reprocessed accordingly.

One best structure is selected in this default automatic mode, while all the intermediate files are removed. However a refinement protocol and selection criterion universally suitable for all structures do not exist. Therefore, in the second mode of *Zanuda* no refinements are performed; instead all the transformed models and data are stored in a directory defined by a user. In the task interface, this mode can be activated using the drop-down list with the choice between "REFINE..." and "SAVE...". The second mode could be useful when, for example, refinement statistics for two structures are very similar and the automatic choice of one of them cannot be reliable. The initial models and transformed data for these two structures can be quickly obtained using this mode and then a more careful refinement and model rebuilding can be performed manually by a user.

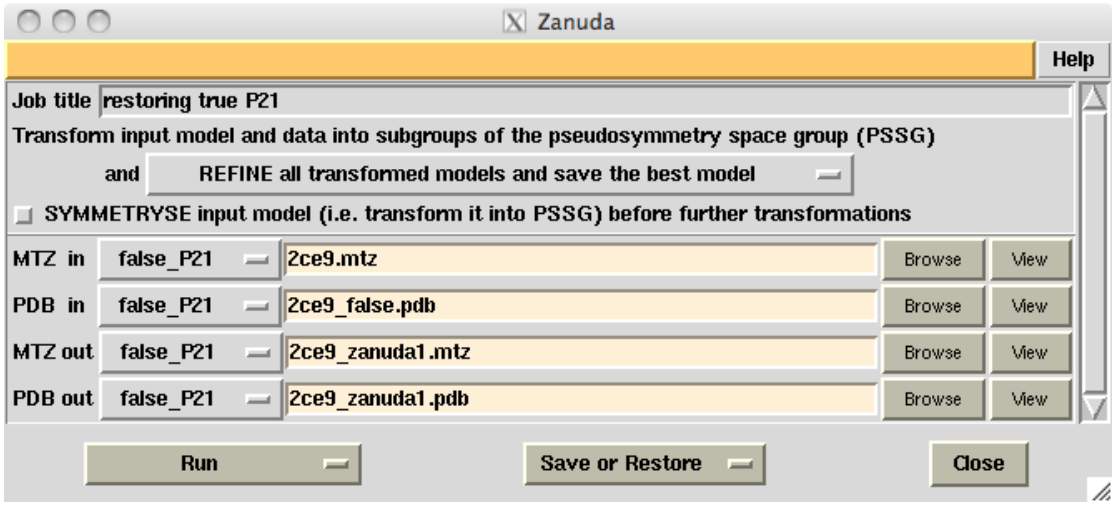


Figure 1. CCP4I interface for Zanuda task with default settings.

```
coordinates       2ce9_false.pdb
data              2ce9.mtz
readability test  passed (Refmac_5.7.0024)
resolution        2.600
spacegroup        P 1 21 1
cell              107.810 56.478 126.642 90.00 112.68 90.00

Step 1.
R-factors for the starting model.
Transformation into a supergroup.

-------------------------------------------------------------------
| Subgroup | Spacegroup | R.m.s.d. |   Refinement in tested group   |
|          |            | from the |--------------------------------|
|   Ref    |            | starting |  Rigid   |     Restrained      |
|          |            | model, A |----------|---------------------|
|          |            |          |    R     |    R     |  R-free  |
|----------|------------|----------|----------|----------|----------|
| >>   2   | P 1 21 1   |  0.0004  |    --    |  0.3023  |  0.3392  |
|      5   | P 1 21 1   |  0.2498  |    --    |    --    |    --    |
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

Step 2.
Refinements in subgroups.
There are 3 subgroups to test.

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
| >>   2   | P 1 21 1   |  0.0004  |    --    |  0.3023  |  0.3392  |
-------------------------------------------------------------------
|      1   | P 1        |  0.1288  |  0.2786  |  0.2747  |  0.3528  |
|      2   | P 1 21 1   |  0.1306  |  0.2782  |  0.2746  |  0.3554  |
|      4   | P 1 21 1   |  0.4668  |  0.2754  |  0.2283  |  0.3044  |
-------------------------------------------------------------------
| <<   4   | P 1 21 1   |  0.4668  |  0.2754  |  0.2283  |  0.3044  |
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

Step 3.
Refinement of the best model.
Candidate symmetry elements are added one by one.

^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
| >>   4   | P 1 21 1   |  0.4668  |  0.2754  |  0.2283  |  0.3044  |
-------------------------------------------------------------------
|      1   | P 1        |  0.4601  |  0.2831  |  0.2285  |  0.3029  |
|      4   | P 1 21 1   |  0.4790  |    --    |  0.2261  |  0.3046  |
-------------------------------------------------------------------
| <<   4   | P 1 21 1   |  0.4790  |    --    |  0.2261  |  0.3046  |
-------------------------------------------------------------------

R-factor in the original subgroup is NOT the best.
The original spacegroup assignment seems to be incorrect.
```

Figure 2. Correction of the space group assignment for the false origin structure generated from the PDB structure 2ce9.

This figure shows the summary file of Zanuda (with timestamps excluded). (Step 1) *The input structure (subgroup 2) was transformed into the PSSG (subgroup 5) to calculate the r.m.s.d. of CA-atoms between the initial and symmetrised structures.* (Step 2) *The input structure was refined in candidate subgroups and* (Step 3) *transformed into the correct space group (subgroup 4). The input and output for a given step are marked by ">>" and "<<", respectively. All shown subgroups have equivalent unit cells except for the PSSG, which has the parameter a halved.*

Pseudosymmetry

The space group of the crystal contains all the symmetry operations that map the crystal structure on itself. Similarly one can define a space group that, in addition, contains all the operations that perform approximate mapping, in a sense that the atomic coordinates need small adjustments to make the overlap between the structure and its copy exact. Such approximate operations are called

pseudosymmetry operations and the extended space group will be further referred to as a pseudosymmetry space group (PSSG).

Noteworthy, the non-crystallographic symmetry (NCS) and pseudosymmetry are different concepts. An NCS operation is local and is defined by the best overlap of two NCS-related molecules after applying the NCS operation to one of them. On the contrary the pseudosymmetry operation is global and is defined by the best match between the entire crystal and its transformed copy. Thus the NCS operation and pseudosymmetry operation relating the same two molecules are in general different operations and may coincide only in special cases.

In structures with one molecule per asymmetric unit (AU) there is no pseudosymmetry and PSSG coincides with the space group of the crystal. In many cases of NCS, as, for example, in crystals with five molecules per AU, the global mapping of the crystal on itself cannot be defined even formally and PSSG remains equal to the crystal space group. In addition, *Zanuda* imposes the upper limit of 3 Å for the C-α r.m.s.d. between the structure and its copy generated by an additional global operation. Global operations with larger values of r.m.s.d. are ignored as they are unlikely to be misinterpreted.

False origin structures

Let us consider a structure with space group symmetry $P2_1$ and pseudotranslation vector **a**/2 (Fig. 3, Table 1). Its PSSG is a $P2_1$ space group with the basis of lattice vectors (**a**/2, **b**, **c**) (Fig. 3*a,b*). There are two $P2_1$ subgroups of the PSSG, both having the basis (**a**, **b**, **c**) compatible with the experimentally observed unit cell parameters. Let the first of these two subgroups be the true space group of the crystal structure (Fig. 3*c,d*). Then the second one is associated with the false origin structure in which pseudosymmetry axes are treated as crystallographic axes and vice versa (Fig. 3*e,f*). The two structures are different because different subsets of atoms are related by crystallographic symmetry, or, in other words, different symmetry constraints were implicitly imposed on the structures during refinement.

A coordinate file corresponding to the false structure (Fig. 3*e,f*) can easily be generated manually from the coordinates of the true structure (Fig. 3*c,d*) using, for example, the program *Lsqkab* from the CCP4 suite. At the start, the AU in the first structure may need to be redefined to include molecules related by pseudotranslation rather than by the rotation about a pseudosymmetry axis. Such an AU is convenient because it is also an AU in the second structure. Then the shift **a**/4 is applied to all atoms. (In a general case, the transformation from one subgroup to another is more complicated and includes, along with transformation of coordinates, extension of the AU or merging of several AUs together with averaging coordinates of related atoms.)

As a result of our transformation, the pseudosymmetry axes are treated as crystallographic axes (Fig. 3*f*). In the new model the relative positions of symmetry-related molecules (Fig. 3*e*), which are implicitly generated during refinement, are different from their relative positions in the true structure (Fig. 3*c*). Obvious

consequences of such an error include bad refinement statistics and distorted electron density, usually with several breaks along the main chain.

False origin solutions are sometimes generated, when a structure with pseudotranslation is solved by MR. A real case of a false origin problem for a monoclinic structure was discussed in [4]. A more sophisticated example was encountered during MR structure solution of the GAF domain of CodY protein, PDB code 2gx5 [5]. The structure belonged to $P4_322$ space group and had a pseudotranslation **c**/2. Therefore, three false origin structures were possible, one belonging to $P4_322$ space group and two belonging to $P4_122$. Another interesting example was reported in [6]. In this case the complete $P3_1$ structure was solved only after a false origin MR solution of $P3_121$ substructure had been corrected.

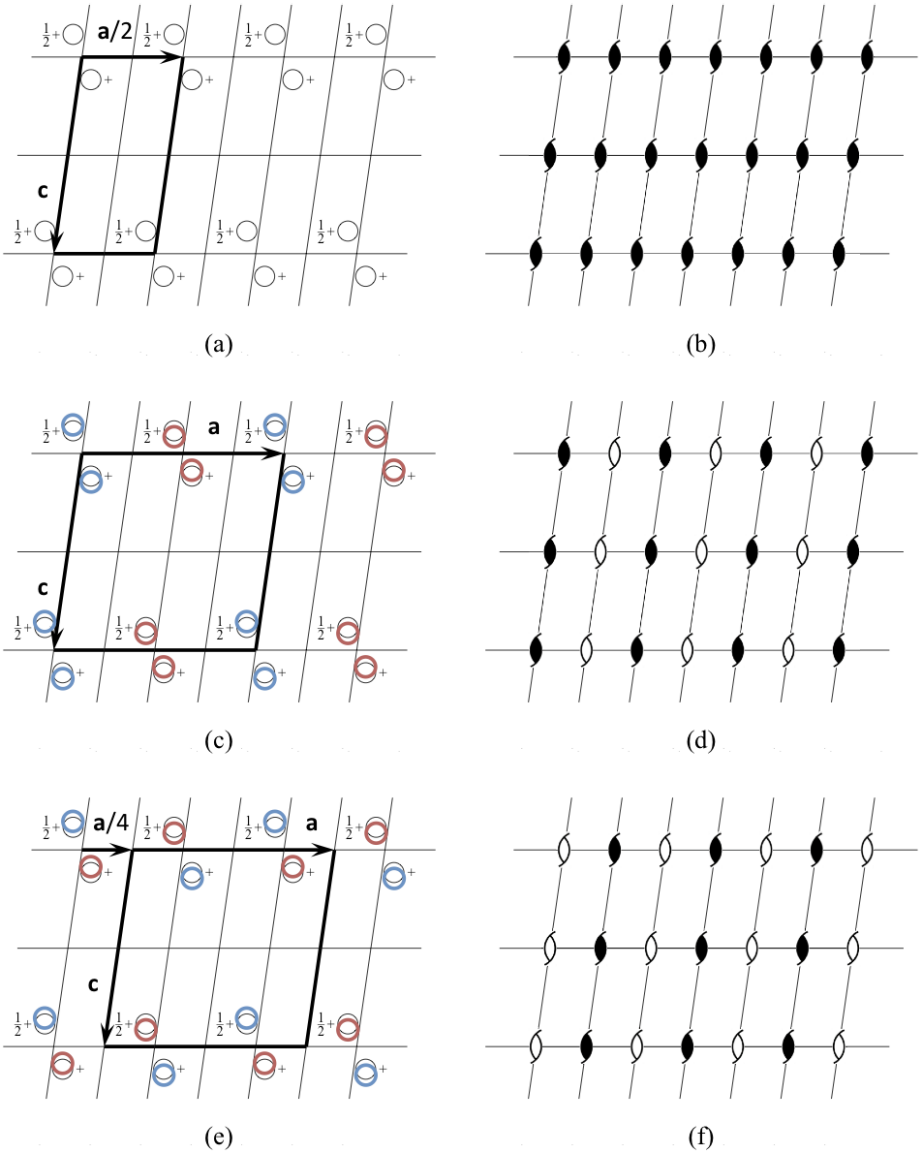


*Figure 3. Pseudotranslation **a**/2 in a $P2_1$ space group.*

Let the approximate structure (a, b)*, in which the pseudotranslation **a**/2 acts as crystallographic translation, be known. This structure belongs to space group $P2_1$ with the basis of lattice vectors (**a**/2, **b**, **c**). This leaves two possibilities for the true*

structure, (c, d) *and* (e, f)*. Both structures belong to the space group $P2_1$ with the basis of lattice vectors (**a**, **b**, **c**). The difference between the two structures is that crystallographic two-fold axes that are shown as filled shapes in* (d) *are treated as pseudosymmetry axes, which are shown as open shapes in* (f)*, and vice versa. Accordingly, relative positions of symmetry related atoms, displayed as circles of the same colour in* (c) *and* (e)*, are different. If, for example,* (c, d) *represent the true structure, then* (e, f) *represent an associated false origin structure. The standard crystallographic origin in $P2_1$ is located on the crystallographic two-fold axis, and, therefore, the crystallographic coordinates of corresponding atoms in the two structures differ by approximately **a**/4. In particular, this ambiguity may cause problems with structure solution using molecular replacement. Dependent on whether the first copy of search model is found at its true position or is displaced by **a**/4 from the true position, the molecular replacement will result in either the correct or the false origin solution.*

Subgroup	SG	Basis	Origin	Fig. 3
1	$P\,1$	**(a, b, c)**	0	–
2	$P\,1\,2_1\,1$	**(a, b, c)**	**a**/4	(*e*, *f*)
3	$P\,1$	**(a**/2, **b, c)**	0	–
4	$P\,1\,2_1\,1$	**(a, b, c)**	0	(*c*, *d*)
5	$P\,1\,2_1\,1$	**(a**/2, **b, c)**	0	(*a*, *b*)

*Table 1. Subgroups of the PSSG for a $P2_1$ structure with the pseudotranslation **a**/2.*

Subgroup reference number used in summary file in Fig. 2 (Subgroup)*, space group Hermann-Mauguin symbol* (SG)*, basis of lattice vectors* (Basis)*, position of the standard origin relative to the standard origin in true structure* (Origin) *and references to the panels of Fig. 3 are shown for five subgroups of PSSG including PSSG itself. Among an infinite number of subgroups these subgroups have either smallest unit cells (3 and 5), or the basis of lattice vectors compatible with the experimentally observed unit cell parameters (1, 2 and 4).*

Example

The structure with PDB code 2ce9 [7] has the combination of symmetry and pseudosymmetry as shown in Fig. 3. The false origin structure can be generated from the true PDB structure as explained in the previous section, or using *Zanuda* in the no-refinement mode (selection "SAVE ..." in the mode list). The model generated using *Zanuda* was further refined with strong geometrical restraints and then used as an input for the demonstrative *Zanuda* run presented in Figs. 1 and 2.

Summary file (Fig. 2) contains a description of the input (including the confirmation that the input files have passed the readability test) and three tables corresponding to three steps of *Zanuda* protocol. Each table row corresponds to an atomic model. A reference number of a subgroup of PSGG to which the model belongs is given in the first column. The first column may also contain a symbol denoting the role of the model for this step, with ">>" and "<<" standing for input and output models,

respectively. The space group Hermann-Mauguin symbol for the subgroup is shown in the second column. The reference numbers and space group symbols for relevant subgroups are also listed in Table 1.

The third column shows C-α r.m.s.d. of a given model from the input model. This is a global deviation between two infinite crystal structures, not between two AUs or two molecules. For example, this number for the final model is small, of order of 0.1 Å, if the initial and final subgroups are the same. On the contrary, if the initial model was substantially incorrect (pseudosymmetry operations were treated as crystallographic operations), then the r.m.s.d. for the final model will be larger, typically 0.5 to 2 Å.

The last three columns present R-work after rigid body refinement, and R-work and R-free after restrained refinement. However, at the Step 1 of the protocol no refinement is performed and the only two R-factors shown are for the modified input model (see the next paragraph). Also, rigid body refinement at Step 3 is performed only for the *P*1 model and remaining rigid-body R-factor columns are void. In the no-refinement mode all the R-factor columns are void.

At Step 1, the PSSG is determined, the input model is modified and transformed into PSSG. The modification involves the removal of solvent and of those residues, which have no match in at least one of the pseudosymmetry related chains. The first row in the table corresponds to the modified model in the original subgroup. The second row corresponds to the model transformed into PSSG. In the default mode discussed in our example, this transformed model is not used further on and is shown for information only. R.m.s.d. for this structure characterises the deviation of the pseudosymmetry in the input structure from the exact crystallographic symmetry. Limiting value here is 3 Å; larger deviations of pseudosymmetry operations from exact symmetry do not usually hinder the space group assignment and are not included in PSSG.

Step 2 involves independent refinements in those subgroups of the PSSG, which have the basis of lattice vectors compatible with observed unit cell parameters. In our example this criterion was satisfied for one *P*1 subgroup (subgroup 1 in Fig. 2 and Table 1) and two $P2_1$ subgroups (subgroups 2 and 4). After this step it was already quite clear that the subgroup 4 was the correct one. The structure refined in this subgroup deviated from the input model significantly more (r.m.s.d. 0.47 Å) than structures in subgroups 1 and 2 (r.m.s.d. 0.13 Å), and it was a change in the right direction as indicated by substantially lower R-free for this model (30.4%) compared to the other two (35.3% and 35.5%). One could have expected that refinement in *P*1 (subgroup 1) would also be able to improve the model because there are no rotational-symmetry constraints in this space group. However, this has not happened because the previous refinement in an incorrect subgroup (during the input model preparation) pushed the model into a wrong local minimum from which the model can not escape in the course of refinement in *P*1.

The model with lowest R-free is passed to Step 3, where it is expanded to *P*1 (subgroup 1), refined, and then symmetry operations are added one by one, with a round of refinement after each addition. In our case this procedure only confirms that the subgroup 4 is the right answer, but in general comparison of R-free factors after these refinements provides a reasonable criterion for final subgroup selection. In fact

this procedure alone would have been sufficient, provided the refinement in *P*1 always converge to a correct minimum. However this is not always the case – as demonstrated by our example – and series of refinements in subgroups at Step 2 give more chance of a successful structure correction.

In more detail, the following actions are performed at the Step 3. Firstly, rigid body and restrained refinement are carried out for the input model extended to the space group *P*1 (subgroup 1). This is followed by one or more cycles of increasing the symmetry. At each cycle an attempt is made to find such a symmetry operation from PSSG that (i) hasn't been used in previous cycles, and (ii) does not result in changes of primitive unit cell parameters - so the pseudotranslation operations are never added. If several non-equivalent operations are available, the program chooses the one, which gives a minimal r.m.s.d. between the previous refined structure and its copy generated using the operation under consideration. If found, such a symmetry operation and the space group obtained in the previous cycle define a new space group with higher symmetry; the refined model from the previous cycle is transformed into the new space group and refined again. Only if the new operation is found and refinement in the new space group has lead to a decrease in R-free or to its increase by no more than 2%, *Zanuda* moves to a new cycle and the described procedure is repeated. Otherwise it terminates and outputs the result from the previous cycle.

In our example two such cycles were completed. At the first cycle, a screw two-fold rotation operation was added resulting in the space group $P2_1$ (subgroup 4). After refinement in this space group, the increase in R-free was substantially less than the threshold value, so the second cycle started. However, the remaining symmetry operations included pseudotranslations and rotations about the screw two-fold pseudosymmetry axes, and each of them would extend the subgroup 4 to the complete PSSG (subgroup 5) with a smaller unit cell. Therefore, the second cycle was terminated and the structure refined in subgroup 4 was accepted as a final result (Fig. 2).

Other options

If a symmetry operation is missing in the space group from which the model is transformed but is present in the space group to which the model is transformed, then coordinates of atoms related by this operation are averaged and atoms are merged. Of course no such averaging happens for the models representing the initial space group and its subgroups including *P*1. Therefore these models could remain strongly biased towards incorrect symmetry and refinement alone may be unable to pull these models out of the local minimum associated with the incorrect symmetry. This is exactly what has been observed in our example during *P*1 refinement at step 2.

Preliminary transformation of the model into PSSG (check box "SYMMETRISE ...") helps the model escape from such local minima. In addition, with this approach all the refinements at Step 2 start, in effect, from the same crystal structure, which allows more strict comparison of subsequent refinements. If this option were used in our example, then the correct symmetry would have been restored after refinements in both the correct space group $P2_1$ (subgroup 4) and space group *P*1 (subgroup 1).

Preliminary transformation into PSSG helps reduce the bias of the model towards incorrect symmetry, but still does not guarantee a success. Probably a more efficient method would be to start from the structure solved in *P*1 by MR. Then the true space group can be recovered using *Zanuda*. (The option "SYMMETRISE ..." should not be used here). The drawback of such approach is that it might prove difficult if not impossible to solve the structure in *P*1 by MR using the original search model. An attempt to sidestep this difficulty by using the search model refined in the initial, incorrect subgroup, may again result in an incorrect structure.

References

[1] Murshudov, G. N., Skubák, P., Lebedev, A. A., Pannu, N. S., Steiner, R. A., Nicholls, R. A., Winn, M. D., Long, F. & Vagin, A. A. (2011). *Acta Cryst.*, **D**67, 355–367.

[2] Winn, M. D. et al. (2011). *Acta Cryst.*, **D**67, 235–242.

[3] http://www.ysbl.york.ac.uk/YSBLPrograms/

[4] Isupov, M. N. & Lebedev, A. A. (2008). *Acta Cryst.*, D**64**, 90–98.

[5] Levdikov, V. M., Blagova, E., Colledge, V. L., Lebedev, A. A., Williamson, D. C., Sonenshein, A. L. & Wilkinson, A. J. (2009). *J. Mol. Biol.*, **390**, 1007–1018.

[6] Watson, A. A., Lebedev, A. A., Hall, B. A., Fenton-May, A., Vagin, A. A., Dejnirattisai, W., Felce, J., Mongkolsapaya, J., Screaton, G. R., Murshudov, G. N. & O'Callaghan, C. A. (2011). *J. Biol. Chem.*, **286**, 24208–24218.

[7] Jennings, B. H., Pickles, L. M., Wainwright, S. M., Roe, S. M., Pearl, L. H. & Ish-Horowicz, D. (2006). *Mol. Cell*, **22**, 645–655

The *xia2* manual

Graeme Winter and David Waterman

June 26, 2012

Contents

1 Quick Start Guide

If you don't like reading manuals and just want to get started, try:

```
xia2 -2d /here/are/my/images
```

or

```
xia2 -3d /here/are/my/images
```

(remembering of course -atom X if you want anomalous pairs separating in scaling.) If this appears to do something sensible then you may well be home and dry. Some critical options:

- -atom X:
  Tell xia2 to separate anomalous pairs i.e. $I(+) \neq I(-)$ in scaling.
- -2d:
  Tell xia2 to use MOSFLM and SCALA.
- -3d:
  Tell xia2 to use XDS and XSCALE.
- -3dii:
  Tell xia2 to use XDS and XSCALE, indexing with peaks found from all images.

If this doesn't hit the spot, you'll need to read the rest of the document.

2 Introduction

In a nutshell, *xia2* is an expert system to perform X-ray diffraction data processing on *your* behalf, using *your* software with little or no input from *you*. It will correctly handle multi-pass, multi-wavelength data sets as described later but crucially it is not a data processing package. Specifically, if you use *xia2* in published work please include the references for the programs it has used, which are printed at the end of the output.

The system was initially written to support remote access to synchrotron facilities, however it may prove useful to anyone using MX, for example:

- assisting new or novice users,
- giving a second opinion to experts,
- assisting busy users to allow them to focus on problem cases, or
- providing reproducible processing.

The last of these may be most useful for users in a pharmacutical setting, or people wishing to test or benchmark equipment, for example beamline scientists. In all cases however the usage of the program is the same.

3 Background

Users of macromolecular crystallography (MX) are well served in terms of data reduction software, with packages such as HKL2000, Mosflm[1], XDS[2] and d*TREK often available and commonly used. In the main, however, these programs require that the user makes sensible decisions about the data analysis to ensure that a useful result is reached. This manual describes a package, *xia2*, which makes use of some of the aforementioned software to reduce diffraction data automatically from images to scaled intensities and structure factor amplitudes, with no user input.

In 2005, when the *xia2* project was initiated as part of the UK BBSRC e-Science project e-HTPX, multi-core machines were just becoming common, detectors were getting faster and synchrotron beamlines were becoming brighter. Against this background the downstream analysis (e.g. structure solution and refinement) was streamlined and the level of expertise needed to use MX as a technique was reducing. At the same time mature software packages such as Mosflm, Scala[3], CCP4[4] and XDS were available and a new synchrotron facility was being built in the UK. The ground

[1] A.G.W. Leslie, Acta Cryst. (2006) D62, 48-57
[2] W. Kabsch, Acta Cryst. (2010) D66, 125-132
[3] P. Evans, Acta Cryst. (2006) D62, 72-82
[4] CCP4, Acta Cryst. (1994) D50, 760-763

was therefore fertile for for the development of automated data reduction tools. Most crucially, however, the author was told that this was impossible and a waste of time - sufficient motivation for anyone.

4 Acknowledgements

Without the trusted and capable packages Mosflm, CCP4, Scala and XDS it would clearly be impossible to develop *xia2*. The author would therefore like to thank Andrew Leslie, Harry Powell, Phil Evans, Wolfgang Kabsch and Kay Diederichs for their assistance in using their programs and modifications they have made. In addition, more recent developments such as Labelit [5], Pointless[6] and CCTBX[7] have made the development of *xia2* much more straightforward and the end product more reliable. The author would therefore like to additionally thank Nick Sauter and Ralf Grosse-Kunstleve for their help.

Development of a package such as this is impossible without test data, for which the author would like to thank numerous users, particularly the Joint Center for Structural Genomics, for publishing the majority of their raw diffraction data.

During the course of *xia2* development the project has been supported by the UK BBSRC through the e-HTPX project, the EU Framework 6 through the BioXHit project and most recently by Diamond Light Source. The software itself is open source, distributed under a BSD licence, but relies on the user having correctly configured and licenced the necessary data analysis software, the details of which will be discussed shortly.

[5] N.K. Sauter et al. J. Appl. Cryst. (2004) 37, 399-409

[6] P. Evans, Acta Cryst. (2006) D62, 72-82

[7] R.W. Grosse-Kunstleve et al. J. Appl. Cryst. (2002) 35, 126-136

5 Using xia2

As mentioned in the quick start section, to get started simply run:

```
xia2 -2d /here/are/my/images
```

or

```
xia2 -3d /here/are/my/images
```

The program is used from the command-line; there is no GUI. The four most important command-line options are as follows:

Option	Usage
-atom X	tell xia2 to separate anomalous pairs i.e. $I(+) \neq I(-)$ in scaling
-2d	tell xia2 to use MOSFLM and SCALA
-3d	tell xia2 to use XDS and XSCALE
-3dii	tell xia2 to use XDS and XSCALE, indexing with peaks found from all images

These specify in the broadest possible terms to the program the manner in which you would like the processing performed. The program will then read all of the image headers found in `/here/are/my/data` to organise the data, first into sweeps, then into wavelengths, before assigning all of these wavelengths to a crystal.

The data from the experiment is understood as follows. The SWEEP, which corresponds to one "scan," is the basic unit of indexing and integration. These are contained by WAVELENGTH objects which correspond to CCP4 MTZ datasets, and will ultimately have unique Miller indices. For example, a low and high dose pass will be merged together. A CRYSTAL however contains all of the data from the experiment and is the basic unit of data for scaling. This description of the experiment is written automatically to an instruction file, an example of which is shown in Figure 1

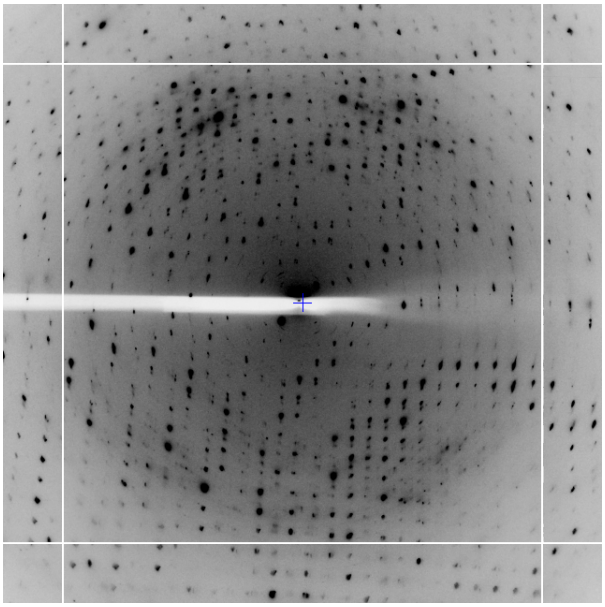
6 Introductory example

The most straightforward way to discuss the operation of the program is through demonstrations with real examples. The first of these is a dataset from a DNA / ligand complex recorded at Diamond Light Source as part of ongoing research. The structure includes barium which may be used for phasing, and the data were recorded as a single sweep. As may be seen from Figure 2, the quality of diffraction was not ideal, and radiation damage was an issue. Initially the data were processed with

```
BEGIN PROJECT AUTOMATIC
BEGIN CRYSTAL DEFAULT

BEGIN HA_INFO
ATOM Ba
END HA_INFO

BEGIN WAVELENGTH SAD
WAVELENGTH 0.979500
END WAVELENGTH SAD

BEGIN SWEEP SWEEP1
WAVELENGTH SAD
DIRECTORY /dls/i02/data/2011/mx1234-5
IMAGE K5_M1S3_3_001.img
START_END 1 450
END SWEEP SWEEP1

END CRYSTAL DEFAULT
END PROJECT AUTOMATIC
```

Figure 1: The input file to the program, which is generated automatically, shows how the input data are understood. This may be adjusted and the program rerun, which will be covered in more detail later in the manual.

Figure 2: Illustration of the central region of a diffraction pattern from the example data set.



```
xia2 -3d -atom Ba /here/are/my/data
```

giving the merging statistics shown in Table 1. From these it is clear that there is something wrong: it is very unusual to have near atomic resolution diffraction with $\sim 10\%$ $R_{\rm merge}$ in the low resolution bin. The most likely reasons are incorrect assignment of the pointgroup and radiation damage - the latter of which is clear from the analysis of $R_{\rm merge}$ as a function of image number (Figure 3 left.) A development option is now available (-3da rather than -3d) which will run Aimless in the place of Scala for merging, and which gives the cumulative completeness as a function of frame number, as shown in Figure 3 right. From this it is clear that the data were essentially complete after approximately 200 frames, though the low resolution completeness is poor.

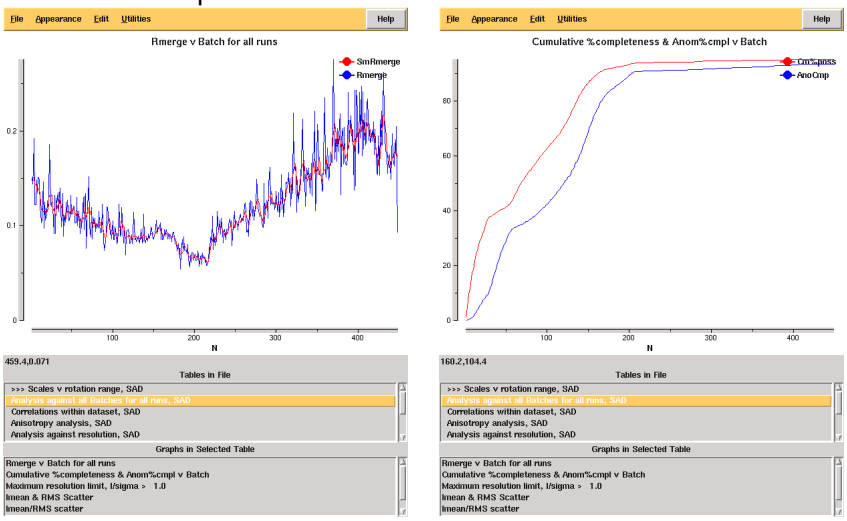
6.1 Modifying input

From the example it would seem sensible to investigate processing only the first 200 of the 450 images. While it is usual to limit the batch range in scaling when processing the data manually, *xia2* is not set up to work like this as decisions made for the full data set (e.g. scaling model to use) may differ from those for the subset - we therefore need to rerun the whole *xia2* job after modifying the input. All that is necessary is to adjust the image

Table 1: Merging stats for processing of the full example data set.

High resolution limit	1.25	6.45	1.25
Low resolution limit	18.85	18.85	1.27
Completeness	95.2	60.1	70.2
Multiplicity	12.2	8.4	4.8
I/sigma	12.3	18.5	2.6
Rmerge	0.113	0.096	0.564
Rmeas(I)	0.129	0.118	0.633
Rmeas(I+/-)	0.121	0.105	0.679
Rpim(I)	0.034	0.038	0.267
Rpim(I+/-)	0.043	0.041	0.368
Wilson B factor	12.131		
Anomalous completeness	93.3	52.6	58.0
Anomalous multiplicity	6.4	5.0	2.0
Anomalous correlation	0.544	0.791	-0.297
Anomalous slope	1.085	0.000	0.000
Total observations	118588	529	1634
Total unique	9749	63	337

Figure 3: Merging statistics and completeness as a function of frame number for the example data.



```
BEGIN PROJECT AUTOMATIC
BEGIN CRYSTAL DEFAULT

BEGIN HA_INFO
ATOM Ba
END HA_INFO

BEGIN WAVELENGTH SAD
WAVELENGTH 0.979500
END WAVELENGTH SAD

BEGIN SWEEP SWEEP1
WAVELENGTH SAD
DIRECTORY /dls/i02/data/2011/mx1234-5
IMAGE K5_M1S3_3_001.img
START_END 1 200 ! THIS WAS 450
END SWEEP SWEEP1

END CRYSTAL DEFAULT
END PROJECT AUTOMATIC
```

Figure 4: The modified input file to the program, showing the change to START_END.

range (START_END) to get the modified input file shown in Figure 4 and rerun as

```
xia2 -3d -xinfo modified.xinfo
```

giving the results shown in Table 2. These are clearly much more internally consistent and give nice results from experimental phasing though with very poor low resolution completeness. At the same time we may wish to adjust the resolution limits to give more complete data in the outer shell, which may be achieved by adding a RESOLUTION instruction to either the SWEEP or WAVELENGTH block.

7 Program Output

As the program runs the key results are written to the screen and recorded in the file `xia2.txt`. This includes everything you should read and includes appropriate citations for the programs that *xia2* has used on your behalf. There is also a file `xia2-debug.txt` which should be send to xia2.support@gmail.com in the event of program failure. There are also two sensibly named directories, `LogFiles` and `DataFiles`, which will be discussed shortly.

Table 2: Merging stats for the first 200 frames of the example data set.

High resolution limit	1.22	6.34	1.22
Low resolution limit	19.62	19.62	1.24
Completeness	86.9	49.1	37.8
Multiplicity	5.3	4.9	1.7
I/sigma	20.1	37.0	2.3
Rmerge	0.036	0.020	0.355
Rmeas(I)	0.060	0.038	0.448
Rmeas(I+/-)	0.043	0.023	0.491
Rpim(I)	0.023	0.014	0.297
Rpim(I+/-)	0.022	0.011	0.339
Wilson B factor	10.70		
Anomalous completeness	77.7	41.0	18.3
Anomalous multiplicity	2.7	3.5	0.5
Anomalous correlation	0.779	0.931	0.000
Anomalous slope	1.553	0.000	0.000
Total observations	50875	272	342
Total unique	9552	55	199

7.1 xia2.txt

By design, the program output from *xia2* includes only the information that is critical to read, as will be shown for a 450 image Pilatus 2M data set recorded from a thaumatin crystal. The results from indexing are displayed as lattice / unit cell:

```
------------------- Autoindexing SWEEP1 --------------------
All possible indexing solutions:
tP  57.60  57.60 149.51  90.00  90.00  90.00
oC  81.45  81.46 149.51  90.00  90.00  90.00
oP  57.59  57.60 149.50  90.00  90.00  90.00
mC  81.46  81.45 149.50  90.00  89.95  90.00
mP  57.60  57.59 149.53  90.00  89.93  90.00
aP  57.59  57.61 149.52  89.93  89.99  89.99
Indexing solution:
tP  57.60  57.60 149.51  90.00  90.00  90.00
```

where in each case the solution with the lowest penalty is displayed. The results of integration are displayed as one character per image - which allows the overall behaviour of the data to be understood at a glance. While mostly 'o' is usually a good indication of satisfactory processing, '%' are not unusual, along with '.' for weaker data. If the output consists of mostly 'O' then it may be helpful to record a low dose data set. The output includes a convenient legend, and looks like the following:

```
-------------------- Integrating SWEEP1 --------------------
Processed batches 1 to 450
Weighted RMSD: 0.26 (0.09)
Integration status per image (60/record):
oooooooooooooooooooooooooooooooooo.oooooooooooooooooooooo
oooooooooooooooooooooo.oooooooooooooooooo.ooooooo.oooooooo
ooo.o.ooooooo.ooooooooooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
ooooooooooooooooooo.ooooooooooooooooooooooooooooooooooooo
oooooooooooooooooooooooooooooooooooooooooooooooooo.oooooooo
oooooooooo.oooooooooo..ooo.oooooooooooooooooooo.ooooooooooooo
ooooooooooooooooooo.oooooooo.
"o" => good        "%" => ok        "!" => bad rmsd
"O" => overloaded  "#" => many bad  "." => blank
"@" => abandoned
Mosaic spread: 0.140 < 0.189 < 0.290
```

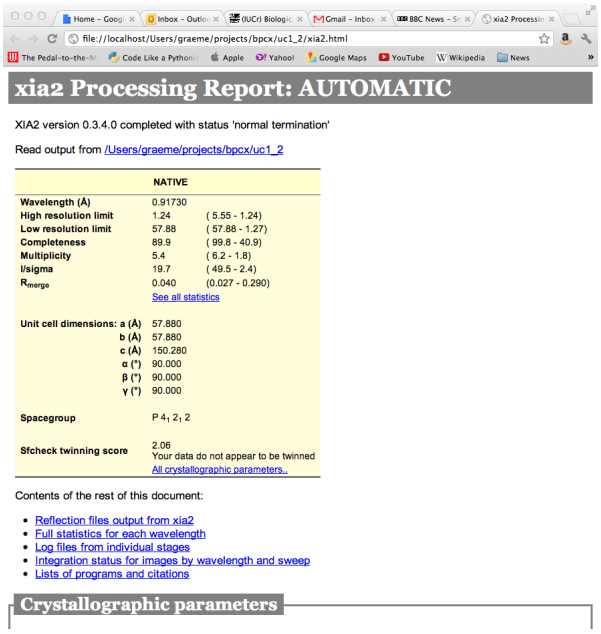
7.2 HTML pages

If `xia2html` has been run there is a nicely formatted html version of this report, which includes graphical representation of some of the log file output from e.g. Scala. Loading up `xia2.html` will give (hopefully self documenting) results as shown in Figure 5. If you have manually run xia2, immediately running `xia2html` in the same directory will generate this.

8 Commonly used program options

There are a number of program options used on a daily basis in *xia2*, which are:

Figure 5: Illustration of xia2html output.



xia2 Processing Report: AUTOMATIC

XIA2 version 0.3.4.0 completed with status 'normal termination'

Read output from /Users/graeme/projects/bpcx/uc1_2

	NATIVE	
Wavelength (Å)	0.91730	
High resolution limit	1.24	(5.55 - 1.24)
Low resolution limit	57.88	(57.88 - 1.27)
Completeness	89.9	(99.8 - 40.9)
Multiplicity	5.4	(6.2 - 1.8)
I/sigma	19.7	(49.5 - 2.4)
R_{merge}	0.040	(0.027 - 0.290)
	See all statistics	
Unit cell dimensions: a (Å)	57.880	
b (Å)	57.880	
c (Å)	150.280	
α (°)	90.000	
β (°)	90.000	
γ (°)	90.000	
Spacegroup	$P 4_1 2_1 2$	
Sfcheck twinning score	2.06 Your data do not appear to be twinned	
	All crystallographic parameters..	

Contents of the rest of this document:

- Reflection files output from xia2
- Full statistics for each wavelength
- Log files from individual stages
- Integration status for images by wavelength and sweep
- Lists of programs and citations

Crystallographic parameters

-atom X	tell *xia2* to separate anomalous pairs i.e. $I(+) \neq I(-)$ in scaling
-2d	tell *xia2* to use MOSFLM and SCALA
-3d	tell *xia2* to use XDS and XSCALE
-3dii	tell *xia2* to use XDS and XSCALE, indexing with peaks found from all images
-2da	tell *xia2* to use MOSFLM and AIMLESS
-3da	tell *xia2* to use XDS and XSCALE, merging with AIMLESS
-3daii	tell *xia2* to use XDS and XSCALE, merging with AIMLESS, indexing with peaks found from all images
-xinfo modified.xinfo	use specific input file
-image /path/to/an/image.img	process specific scan
-spacegroup spacegroup_name	set the spacegroup, e.g. P21
-cell a,b,c,α,β,γ	set the cell constants
-small_molecule	process in manner more suited to small molecule data

Options running Aimless are able to cope with an extremely large number of images - i.e. many thousands, useful when trying to merge data from a number of crystals each with a large number of images, though time consuming!

8.1 Resolution limits

The subject of resolution limits is one often raised - by default in *xia2* they are:

- Merged $\frac{I}{\sigma_I} > 2$
- Unmerged $\frac{I}{\sigma_I} > 1$

However you can override these with -misigma, -isigma.

9 Installing xia2

xia2 depends critically on having CCP4 and CCTBX available. However to get access to the full functionality you will also need XDS and Phenix (which includes Labelit and CCTBX.) Therefore for a "standard" *xia2* installation I would recommend:

- Install CCP4 include updated versions of Pointless and Aimless from ftp://ftp.mrc-lmb.cam.ac.uk/pub/pre
- Download XDS from http://xds.mpimf-heidelberg.mpg.de/ and add this to your path[8]
- Download PHENIX from http://www.phenix-online.org and be sure to source the setup for this *after* CCP4
- Download *xia2* from http://xia2.sf.net and tweak the setup file to reflect where it's installed

By and large, if these instruction are followed you should end up with a happy xia2 installation. If you find any problems it's always worth checking the blog (http://xia2.blogspot.com) or sending an email to xia2.support@gmail.com.

[8] To use -xparallel you will need to fiddle with forkintegrate in the XDS distribution

10 Comments

A question often asked is “which options work best” to which the answer is always “it depends!” This is primarily because the outcome of the analysis depends more on the quality of the data than anything else. However I would always try for yourself and get a feel for how the program works for your data - running both -2d and -3d will simply require more computing time / disk space rather than more effort, so it is certainly worthwhile. For small molecule data though -3dii -small_molecule is a good mix. Also -3d often works better for very finely sliced data.

This manual may be cited freely, however it is preferred that references to the use of *xia2* be made to Winter, G. Journal of Applied Crystallography (2010) 43, 186-190. Please also remember that the programs that *xia2* has used must be correctly cited.

11 Getting xia2

- Blog: xia2.blogspot.com
- Code: xia2.sf.net
- List: xia2-list@lists.sourceforge.net

AMPLE – using *ab initio* modelling to tackle difficult molecular replacement cases

Jaclyn Bibby[1], Ronan Keegan[2], Olga Mayans[1], Martyn Winn[3], Daniel J. Rigden[1]

[1]Institute of Integrative Biology, University of Liverpool, Liverpool L69 7ZB, UK
[2]RCaH, STFC Rutherford Appleton Laboratory, Chilton OX110FA, UK
[3]STFC Daresbury Laboratory, Warrington WA44AD, UK

AMPLE is a joint software development by the University of Liverpool and CCP4. The project's main aim is to assess the suitability of using cheaply obtained *ab initio* models as search models in molecular replacement. An additional goal of the project is to make *AMPLE* into an automated software pipeline tool that can be made available to CCP4 users. This new tool can be used to generate candidate search models for use in molecular replacement problems where success cannot be achieved using homologous structures from other sources or in cases where no obvious homologous structure is available.

Ab initio structure prediction is the prediction of a target structure fold based purely on its sequence information. Some examples of *ab initio* protein structure prediction software are *ROSETTA* (http://www.rosettacommons.org/) and *QUARK* (http://zhanglab.ccmb.med.umich.edu/QUARK/). The first step these programs typically perform is to generate thousands of what are known as "decoy" models. These are rough predictions of the target structure created by a fragment assembly process using libraries of fragments determined for overlapping ranges of the target. Fragments are selected from the PDB based on sequence similarity and consistency with predicted secondary structure. These decoys are clustered based on tertiary structure to give an idea of the most likely fold for the target. A large cluster of similarly folded decoys is indicative of a correct prediction. This first step is relatively quick (~1 minute of CPU time per model). The next step is to add side chains followed by the refinement of the most likely folds under more realistic physics-based force fields. This step can require significantly more computing power and can take up to 100 CPU days to complete. *AMPLE*'s approach is to use *ROSETTA* to perform the first of these steps (decoy assembly and clustering) and to derive from the "cheaply obtained" clustered decoys a set of suitable search models for use in molecular replacement. This method was shown to work in a significant fraction of test cases in a pilot project (see Rigden et al. 2008). The program is currently configured to work with *ROSETTA* but models can be input from other sources manually such as those generated using *QUARK*.

Figure 1: Cluster of decoy models

A spin-off benefit of using these clustered decoys to generate search models is that they naturally lend themselves to the production of “ensemble” search models. Ensembles have been shown to make excellent molecular replacement search models and on some occasions can give a correctly positioned solution where any of the constituent search models that make up the ensemble fail. The general procedure for the processing of decoys into search models is described in detail elsewhere (Acta Cryst. D, currently in submission) but the end result is a set of several hundred ensemble models derived from the initial clustered decoys. Several processes are applied to the clusters to give a broad range of ensembles including many degrees of truncation based on a variance and RMS difference score between the decoys and three different degrees of side chain addition (polyalanine, most reliable side chains and all side chains)

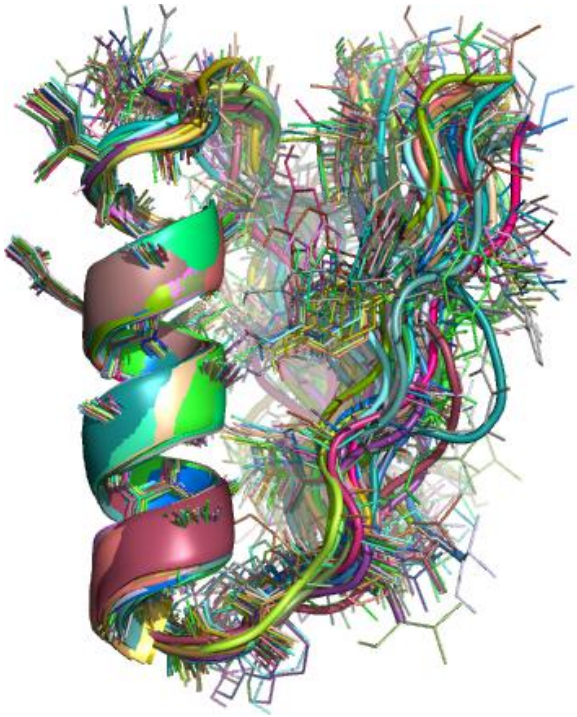


Figure 2: Example ensemble model

MrBUMP (Keegan and Winn, 2008) from the CCP4 suite is then used as the engine for processing all of these search models in molecular replacement. It automates the passing of the models to *PHASER* (McCoy et al., 2007) and *MOLREP* (Vagin and Teplyakov, 1997) and will refine the positioned models output from these programs using *REFMAC*5 (Murshudov et al., 1997) to give an initial estimate of whether or not they are a correct solution. An optional follow-on step, given sufficient resolution (2.2Å or better), of using the phase improvement and C_α-tracing options in the latest *SHELXE* (Sheldrick, 2008) can be performed to give a clear answer as to whether or not MR has been successful. A CC value for the partially traced structure against the native data of 25% or more is a solid indication of success and provides a suitable starting model for model completion.

A comprehensive test of *AMPLE* was carried out using a set of 295 structures from the PDB. The set was made up of structures with maximum sequence length of 120 residues and resolution of 2.2Å or better. Current limitations mean that it is less reliable for larger structures however it has been found to work for a structure as big as 240 residues. A mix of all-*α*, all-*β*, and mixed *α* – *β* secondary structures were selected. In the fragment generation step only fragments from non-homologous structures were used to generate decoys. Results showed that a correct solution could be found in 43% of cases. Success was heavily dependent on secondary structure type. For all-*α* targets, the success rate rose to 79% whereas all-*β* targets were difficult to solve and the success rate for these cases was just 3%.

A beta test release version of *AMPLE* is included in the latest release of the CCP4 software suite (version 6.3.0) with a ccp4i interface for ease of use. Several non-

CCP4 packages are needed to make it work (including *ROSETTA*). Details of what dependencies are needed and how to run the program are available in the ccp4 wiki page for *AMPLE* (www.ccp4wiki.org).

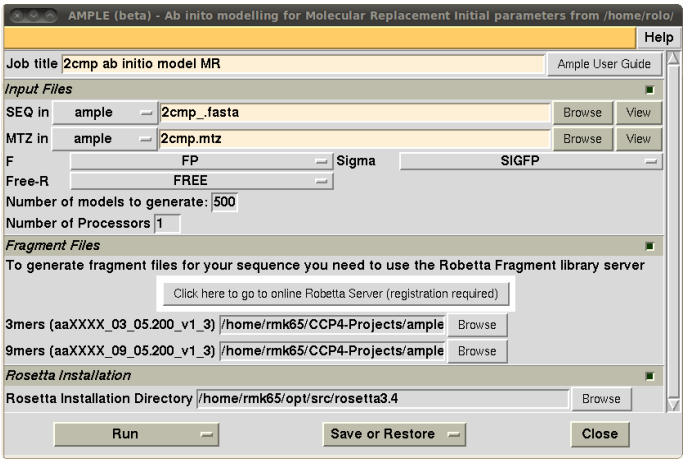


Figure 3: the ccp4i interface for AMPLE

References

Rigden, D.J., Keegan, R.M. and Winn, M.D. “Molecular replacement using *ab initio* polyalanine models generated with ROSETTA” (2008) *Acta Cryst.* D**64**, 1288-1291

Keegan, R.M. and Winn, M.D. “*MrBUMP*: an automated pipeline for molecular replacement” (2008) *Acta Cryst*. D**64**, 119-124

McCoy, A.J., Grosse-Kunstleve, R.W., Adams, P.D., Winn, M.D., Storoni, L.C., Read, R.J. “Phaser crystallographic software” (2007) *J. Appl. Cryst.* **40**, 658-674

Vagin, A. and Teplyakov, A. “MOLREP: an automated program for molecular replacement” (1997) *J. Appl. Cryst.* **30**, 1022-1025

Murshudov, G.N., Vagin, A.A. and Dodson, E.J. “Refinement of Macromolecular Structures by the Maximum-Likelihood Method” (1997) *Acta Cryst*. D**53**, 240-255

Sheldrick, G.M. "A short history of SHELX" (2008). *Acta Cryst*. A**64**, 112-122