



The *PHENIX* graphical interface

Nathaniel Echols

NEchols@lbl.gov

help@phenix-online.org

<http://www.phenix-online.org>

The PHENIX Project

Lawrence Berkeley Laboratory

Paul Adams, Pavel Afonine, Nat Echols, Richard Gildea, Ralf Grosse-Kunstleve, Jeff Headd, Nigel Moriarty, Nicholas Sauter, Peter Zwart



Los Alamos National Laboratory

Tom Terwilliger, Li-Wei Hung



Randy Read, Airlie McCoy, Gabor Bunkoczi, Rob Oeffner

Cambridge University



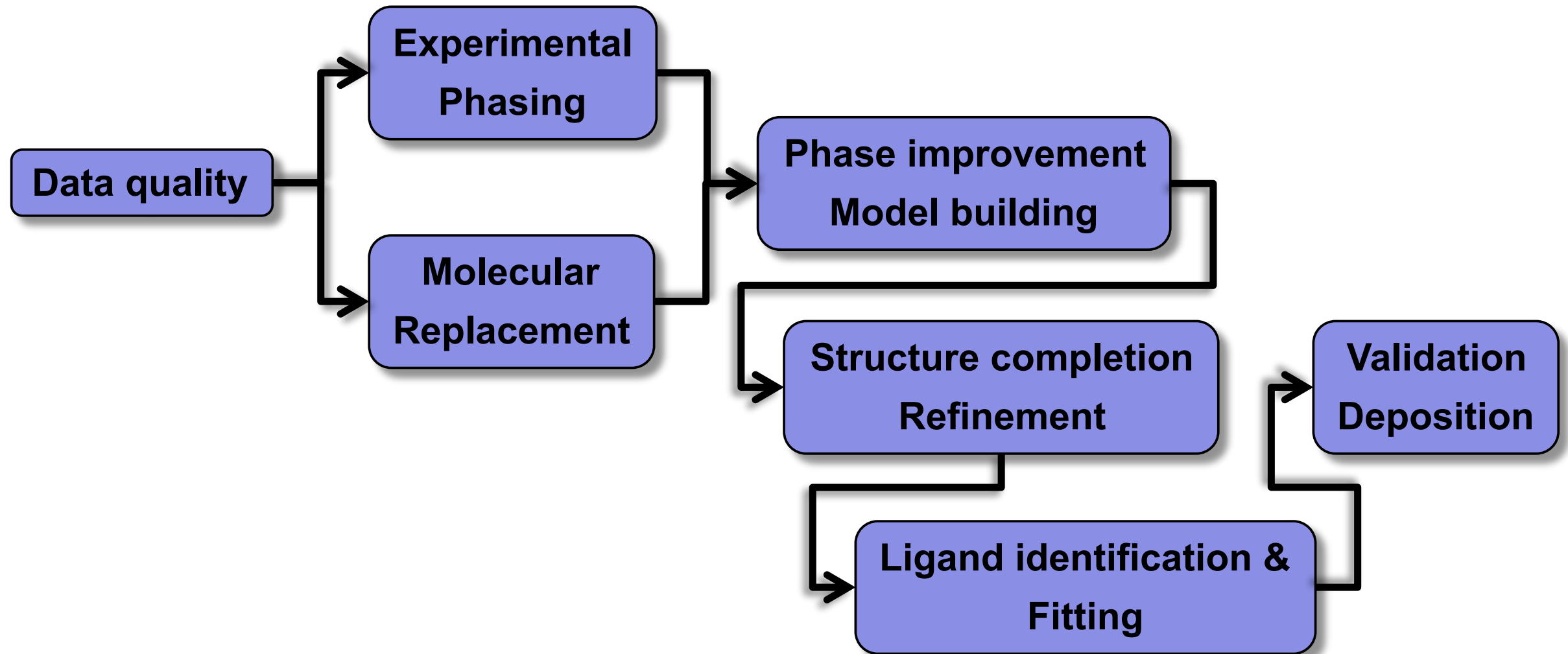
Duke University

Jane & David Richardson, Vincent Chen, Swati Jain, Gary Kapral, Chris Williams, Bryan Arendall, Bradley Hintze

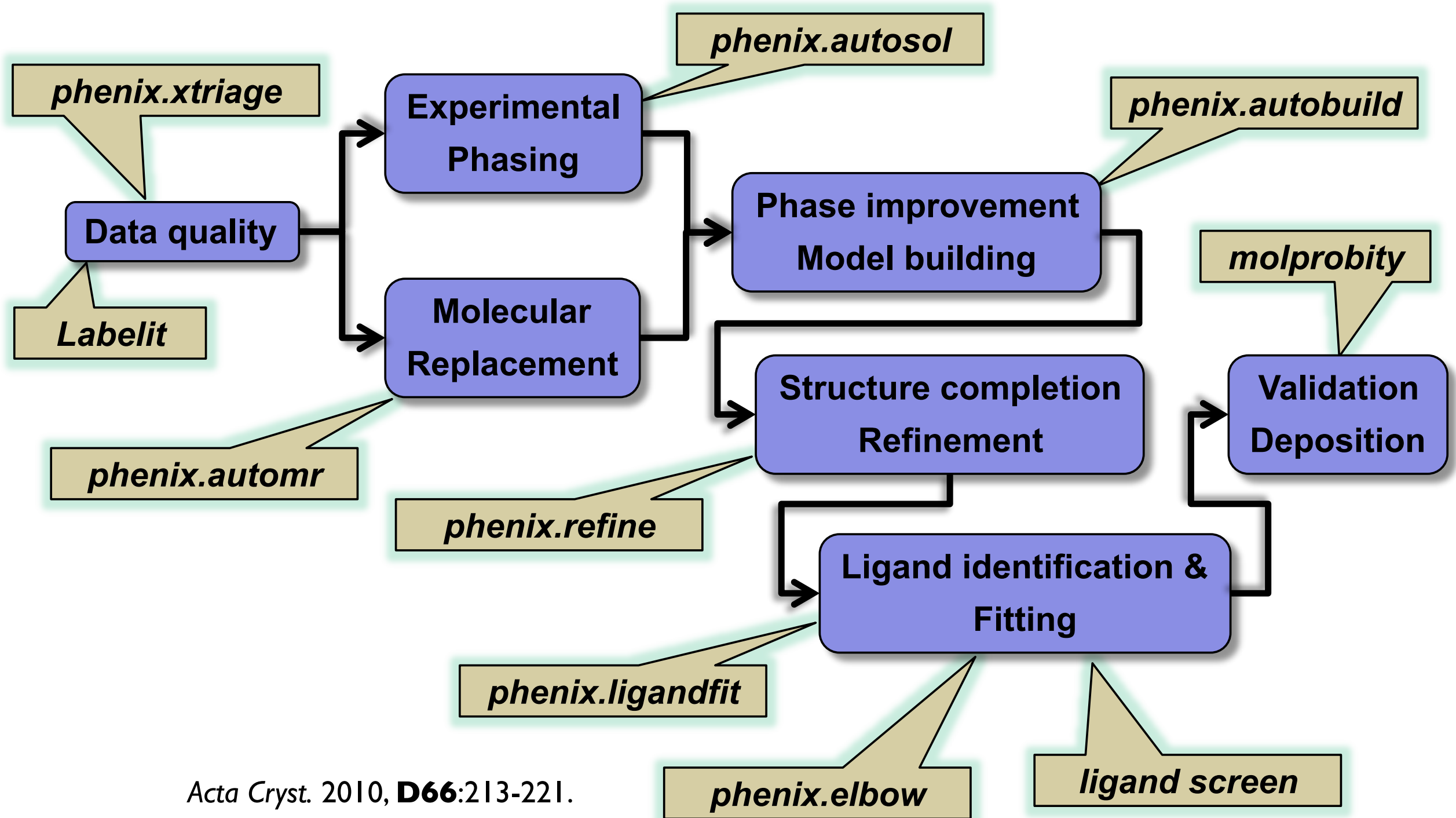


*An NIH/NIGMS funded
Program Project*

Automation of Structure Solution



Automation of Structure Solution



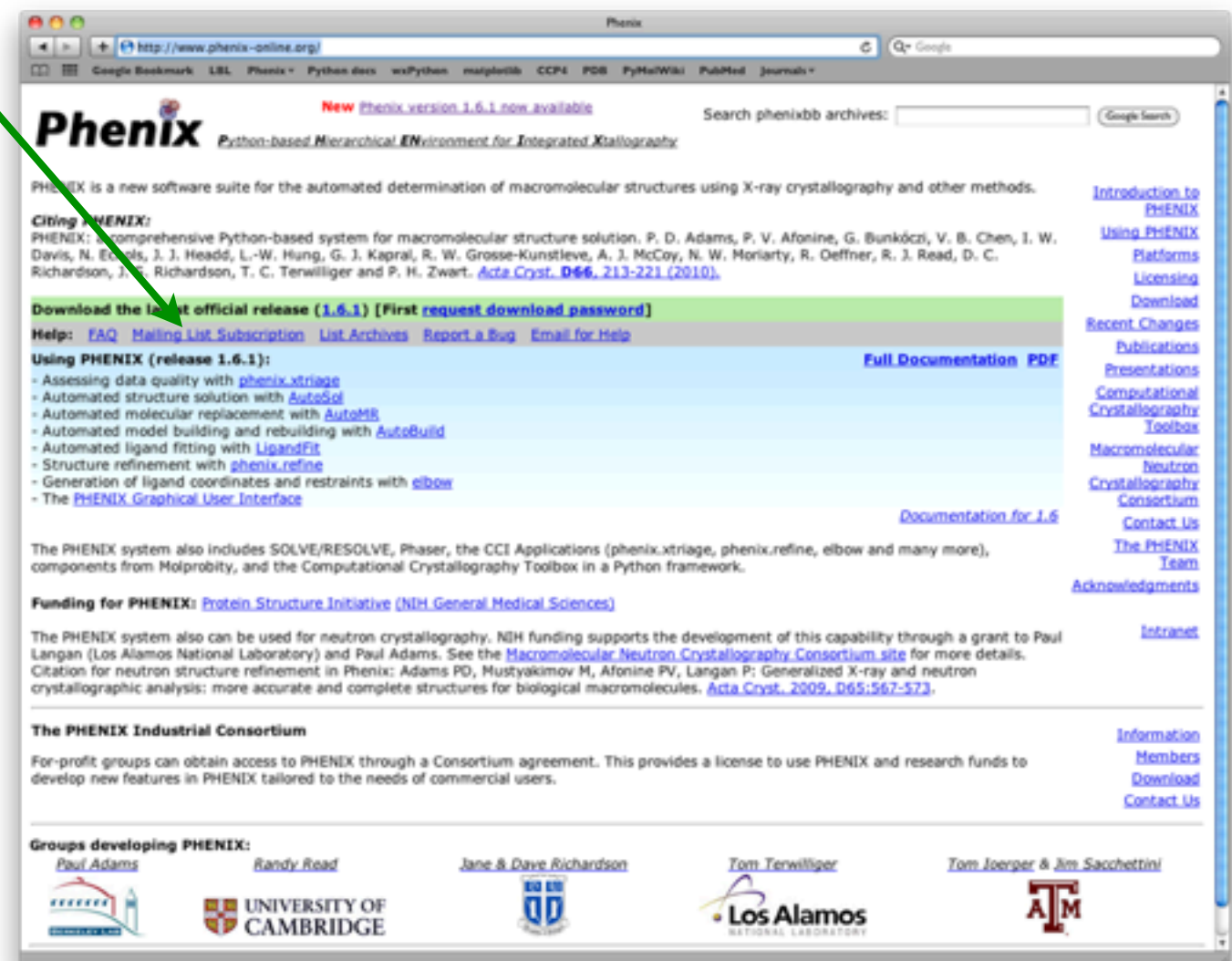
Acta Cryst. 2010, **D66**:213-221.

Why Automation?

- Can speed up the process and can help reduce errors
- Software can try more possibilities than we are typically willing to bother with
- Makes difficult cases more feasible for experts
- Routine structure solution cases are accessible to a wider group of (structural) biologists
- Multiple trials or use of different parameters can be used to estimate uncertainties
- What is required:
 - Software carrying out individual steps
 - Integration between the steps (collaboration between developers)
 - Algorithms to decide which is best from a list of possible results
 - The computer has to make the decisions
 - Strategies for structure determination and decision-making

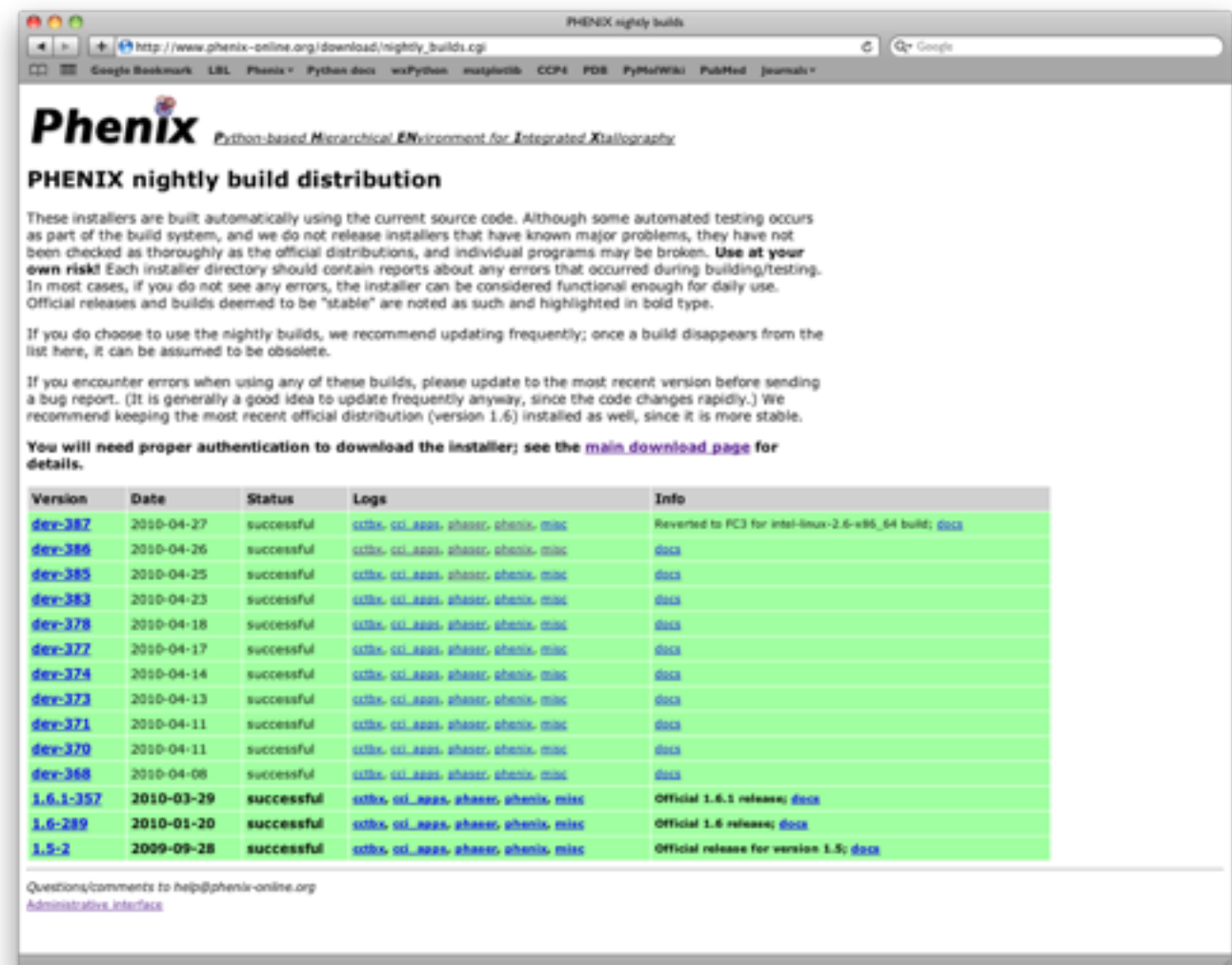
PHENIX resources online

- help@phenix-online.org: user support
- bugs@phenix-online.org: bug reports
- phenixbb@phenix-online.org: message board (subscribers only)
- Regular stable releases, and “nightly” builds
- Supported on:
 - Linux (RedHat, Fedora)
 - Mac OSX
 - Windows (in progress)
- Extensive documentation



Obtaining PHENIX

- Free to academic users; simple online registration required (*please use your academic email address!*)
- Regular official releases (typically 2-8 months)
- Nightly builds
- Regular releases
- Supported on:
 - Linux (RedHat, Fedora)
 - Mac OSX
- Extensive documentation



The screenshot shows a web browser window titled "PHENIX nightly builds" with the URL http://www.phenix-online.org/download/nightly_builds.cgi. The page header includes the Phenix logo and the tagline "Python-based Hierarchical Environment for Integrated X-ray Crystallography". The main heading is "PHENIX nightly build distribution". Below this, there is a paragraph of text explaining that the installers are built automatically and that users should use them at their own risk. A table follows, listing various builds with columns for Version, Date, Status, Logs, and Info. The table shows several nightly builds (dev-387 to dev-368) and three official releases (1.6.1-357, 1.6-289, and 1.5-2). At the bottom of the page, there is a link for "Questions/comments to help@phenix-online.org" and a link to the "Administrative interface".

Version	Date	Status	Logs	Info
dev-387	2010-04-27	successful	utilx, util_xeas, phaser, phenix, misc	Reverted to FC3 for intel-linux-2.6-x86_64 build; docs
dev-386	2010-04-26	successful	utilx, util_xeas, phaser, phenix, misc	docs
dev-385	2010-04-25	successful	utilx, util_xeas, phaser, phenix, misc	docs
dev-383	2010-04-23	successful	utilx, util_xeas, phaser, phenix, misc	docs
dev-379	2010-04-18	successful	utilx, util_xeas, phaser, phenix, misc	docs
dev-377	2010-04-17	successful	utilx, util_xeas, phaser, phenix, misc	docs
dev-374	2010-04-14	successful	utilx, util_xeas, phaser, phenix, misc	docs
dev-373	2010-04-13	successful	utilx, util_xeas, phaser, phenix, misc	docs
dev-371	2010-04-11	successful	utilx, util_xeas, phaser, phenix, misc	docs
dev-370	2010-04-11	successful	utilx, util_xeas, phaser, phenix, misc	docs
dev-368	2010-04-08	successful	utilx, util_xeas, phaser, phenix, misc	docs
1.6.1-357	2010-03-29	successful	utilx, util_xeas, phaser, phenix, misc	Official 1.6.1 release; docs
1.6-289	2010-01-20	successful	utilx, util_xeas, phaser, phenix, misc	Official 1.6 release; docs
1.5-2	2009-09-28	successful	utilx, util_xeas, phaser, phenix, misc	Official release for version 1.5; docs

http://www.phenix-online.org/download/nightly_builds.cgi

Command line tools

- **Very simple and intuitive syntax**

- **Data validation**

```
phenix.xtrriage porin_fp.mtz
```

- **Automated structure solution**

```
phenix.autosol data=peak.sca seq_file=nsf-d2.seq
```

- **Automated model building**

```
phenix.autobuild data=scale.mtz model=mr.pdb  
seq_file=correct.seq
```

- **Automated ligand fitting**

```
phenix.ligandfit data=nsf-d2.mtz model=noligand.pdb  
ligand=atp.pdb
```

- **Structure refinement**

```
phenix.refine nsf-d2.mtz nsf.pdb
```

- **Building ligand coordinates and restraints**

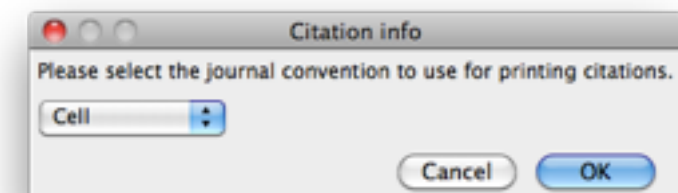
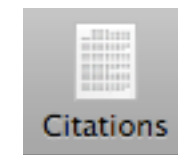
```
phenix.elbow --smiles="C12CC3CC(C2)CC(C1)C3"
```


Why a GUI?

- Condense and summarize output
 - *Human beings make poor text parsers*
 - Many results are inherently graphical - better to plot data than show a table
- Higher-level automation:
 - Simplify transitions between programs, and automatically pick relevant files
 - Suggest appropriate next steps (or run them immediately)
- Track and organize results for a project

PHENIX GUI: Major features and design goals

- Automatic interface generation based on underlying configuration files
- All features available in command line versions should be present in the GUI as well
- Integration with Coot and PyMOL for nearly all programs
- Validation GUI directly controls graphics windows
- Graphical presentation of current progress (where appropriate) and results
- Drag-and-drop of files (from desktop or between windows) supported in most interfaces
- Visual atom selections (mainly for phenix.refine)
- Customization of program behavior and project details
- Simple transitions between programs: start AutoBuild directly from AutoSol, etc.
- Run processes either directly in GUI or independently (“detached”)
- Track and display appropriate citations for programs used
- Automatic bug reports for Python errors - sent directly to me



Central interface

- Project management and application list

Simple utilities
(including download
from PDB)

The screenshot displays the PHENIX central interface. At the top, there is a menu bar with icons for Quit, Preferences, Help, Citations, Reload last job, Coot, PyMOL, KING, and Other tools. Below the menu bar, there are two tabs: 'Actions' and 'Job history'. The main content area is divided into two columns. The left column is titled 'Projects' and contains a table of project information. The right column is titled 'Reflection tools' and lists various utilities. At the bottom, there is a 'Current directory' field and a 'Project' field.

Many user-adjustable settings in here

Project-specific settings (including default files)

ID	Last modified	# of jobs	R-free
FXa	Sep 19 2011 05:57 ...	5	---
rnase-s_nat	Sep 19 2011 03:27 ...	28	0.2508
p9-sad	Sep 19 2011 03:10 ...	30	0.2820
scratch	Sep 19 2011 02:24 ...	56	0.0043
PknB_D76A	Mar 16 2011 07:17 ...	31	0.2982
yghZ	Mar 13 2011 05:56 ...	9	---
porin-twin_nat	Jan 17 2011 03:08 pm	3	---
pka-compare_nat	Oct 13 2010 11:37 ...	3	0.2533
beta-blip	Oct 13 2010 09:41 ...	15	---
nsf-d2-ligand_nat	Oct 13 2010 09:35 ...	1	---
PKA	Sep 01 2010 09:44 ...	2	---
pGI	Mar 11 2010 07:37 ...	7	0.2284

Reflection tools

- Xtrriage**
Analysis of data quality and crystal defects
- Reflection file editor**
Utility for merging and converting reflections
- Calculate F(model)**
Utility for generating structure factors from a PDB file
- Import CIF structure factors**
Convert data deposited in PDB to MTZ format
- French & Wilson data correction**
Statistics-based handling of negative intensities
- Model-based phases**
Utility to obtain PHI, FOM, and Hendrickson-Lattman coefficients from data and PDB file
- 3D data viewer**
OpenGL visualization of reflection data in reciprocal space
- 2D data viewer**
Visualization of reflection data in slices through reciprocal space

Model tools

- PDB Tools**
Utility for simple modifications of PDB files, including geometry regularization
- Combine PDB files**

Current directory: /private/var/tmp Browse...

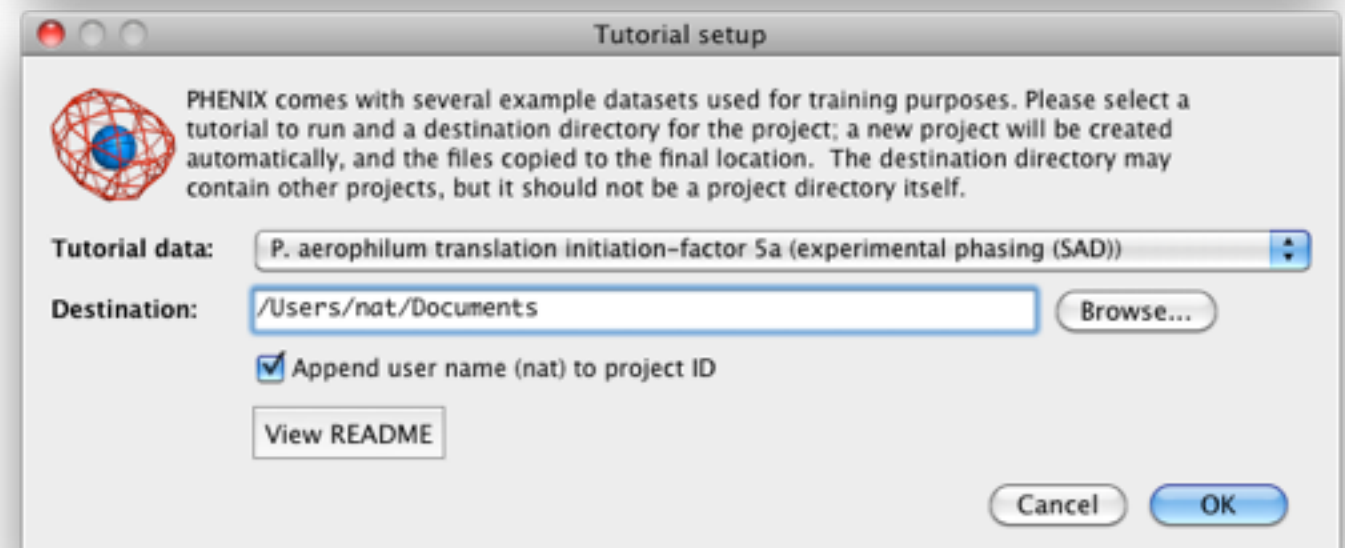
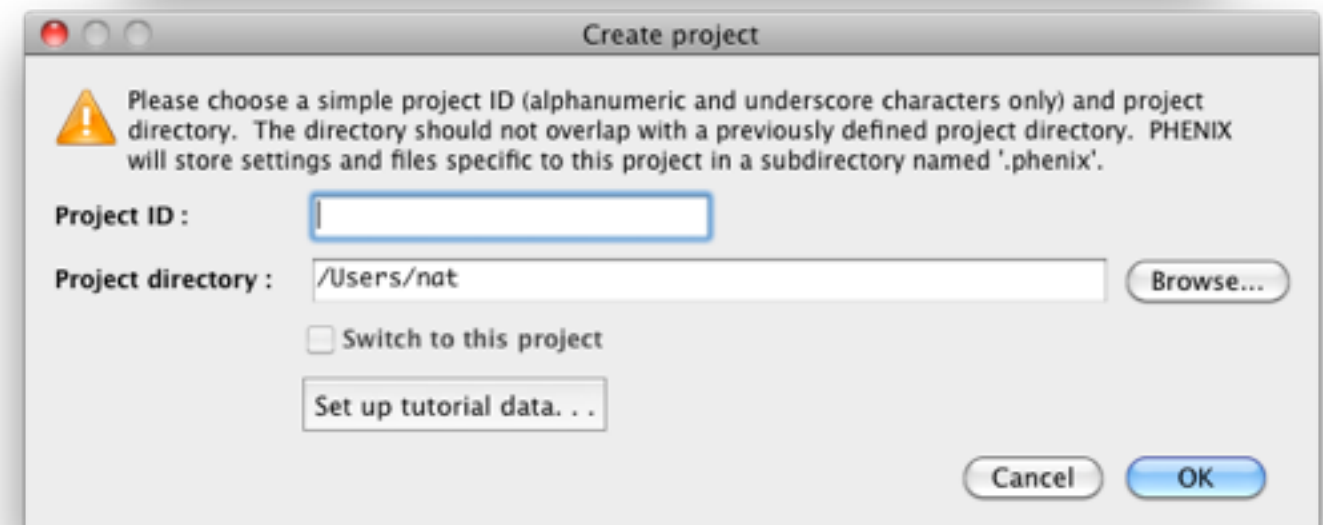
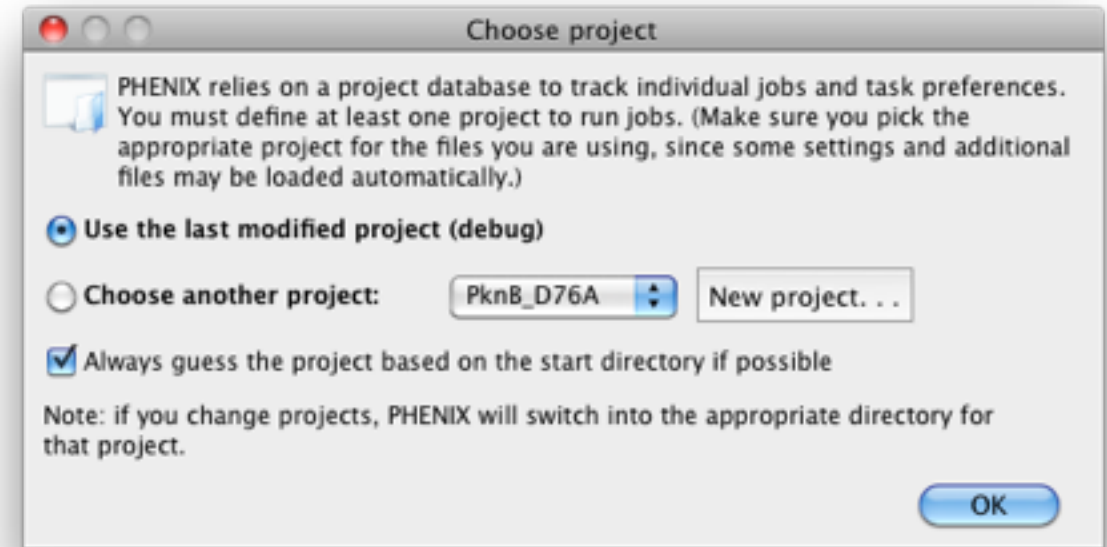
PHENIX version dev-683 Project: scratch

Project management

- Projects are mainly used to track related jobs and store results
- Some project-specific settings available (X-ray data, hydrogen addition)
- Creating a project in a directory will add a subdirectory “.phenix” to store internal data
- Tutorial setup with sample data now integrated with project management

Things to avoid:

- Making your home directory a project directory
- Nesting project directories
- Moving project directories - use “Move project” in the Projects menu for this



Project history

Display can be limited to specific programs with this menu

PHENIX home

Quit Preferences Help Citations Reload last job Coot PyMOL KING Other tools

Actions Job history

Jobs for project scratch

Show only: [dropdown] Hide failed or aborted jobs

Sort by: job ID [dropdown] Show flagged jobs only

ID	Program	Date	Job title or directory	R-free
38	phenix.reflection_file_editor	Sep 07 2010 12:53 pm	---	---
39	phenix.cif_as_mtz	Oct 11 2010 11:51 am	---	---
40	phenix.validate	Oct 11 2010 11:55 am	---	0.2249
41	phenix.xtriage	Nov 23 2010 02:24 pm	---	---
42	phenix.validate	Nov 23 2010 02:31 pm	---	0.34
43	phenix.pdbtools	Nov 27 2010 04:42 pm	---	---
44	phenix.xtriage	Mar 16 2011 07:20 pm	[phenix.xtriage]	---
45	phenix.refine	Mar 16 2011 08:44 pm	[phenix.refine]	---
46	phenix.refine	Mar 16 2011 08:48 pm	[phenix.refine]	---
47	phenix.refine	Mar 19 2011 06:38 pm	[phenix.refine]	0.1638
48	phenix.fmodel	Mar 21 2011 12:21 am	fake Fs for cyclic peptide	---
49	phenix.refine	Mar 21 2011 12:31 am	[phenix.refine]	0.0043
50	phenix.refine	Mar 24 2011 09:40 pm	[phenix.refine]	---
51	phenix.refine	Mar 25 2011 11:40 am	[phenix.refine]	0.1638
52	phenix.refine	Mar 25 2011 03:57 pm	[phenix.refine]	0.1622
53	phenix.refine	Mar 25 2011 04:01 pm	[phenix.refine]	0.1611
54	phenix.xtriage	Apr 13 2011 08:19 pm	[phenix.xtriage]	---
55	phenix.refine	May 02 2011 10:28 am	[phenix.refine]	0.1656
56	phenix.xtriage	Sep 19 2011 02:24 pm	[phenix.xtriage]	---

Restore job Flag/unflag job Mark as failed Delete Show details

Current directory: /private/var/tmp Browse...

PHENIX version dev-683 Project: scratch

This icon indicates that the job failed (for any reason)

This icon indicates an aborted job

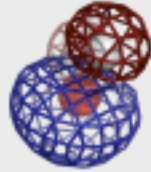
“Flagging” a job marks it as important

This button opens a summary panel (next slide)

Project history: job summaries

- Overview of input and output files, and statistics

Project 'scratch': job #55

 **ID: 55 Refine** Run on May 02 2011 10:28 am

Title: [phenix.refine]

Directory: /private/var/tmp/Refine_55

Config file: .phenix/project_data/refine_55.eff

Statistics:

R-work:	0.1857	R-free:	0.1656
RMSbonds:	0.007	RMSangles:	0.917

Input files: (SP = /var/tmp)

File name	Parameter
<input type="button" value="🔍"/> /private/var/tmp/1yjp.mtz	Reflections file, File with R(free...
<input type="button" value="🔍"/> /private/var/tmp/1yjp.pdb	Input model

Output files:

File name	Contents
<input type="button" value="🔍"/> scratch_refine_55.eff	Effective parameters for this run
<input type="button" value="🔍"/> scratch_refine_55.geo	Geometry restraints before refi...
<input type="button" value="🔍"/> scratch_refine_55.log	phenix.refine log file
<input type="button" value="🔍"/> scratch_refine_55.mtz	Map coefficients for Coot
<input type="button" value="🔍"/> scratch_refine_55.pdb	Refined model

Project directory layout

- Inside each project directory:

.phenix/

project.phil

basic project info

job_history.phil

job history (without details)

tmp/

folder for temporary files

defaults/

various default files (if defined)

project_data/

data for individual jobs

job_[X].phil

record of input/output files and statistics

refine_1.eff

job runtime files (including

refine_1.log

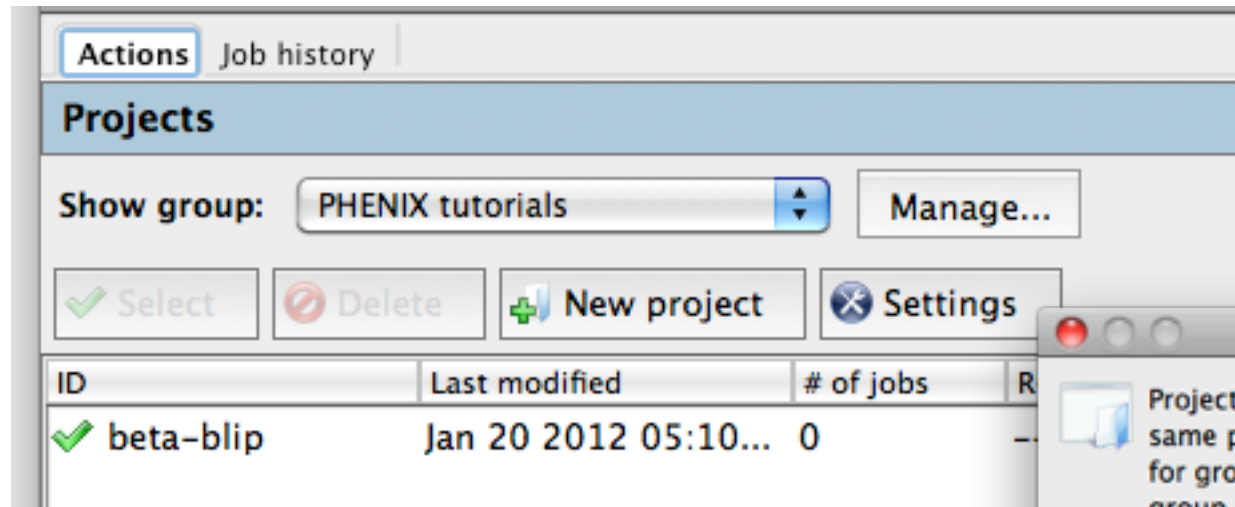
log and saved result)

refine_1.pkl

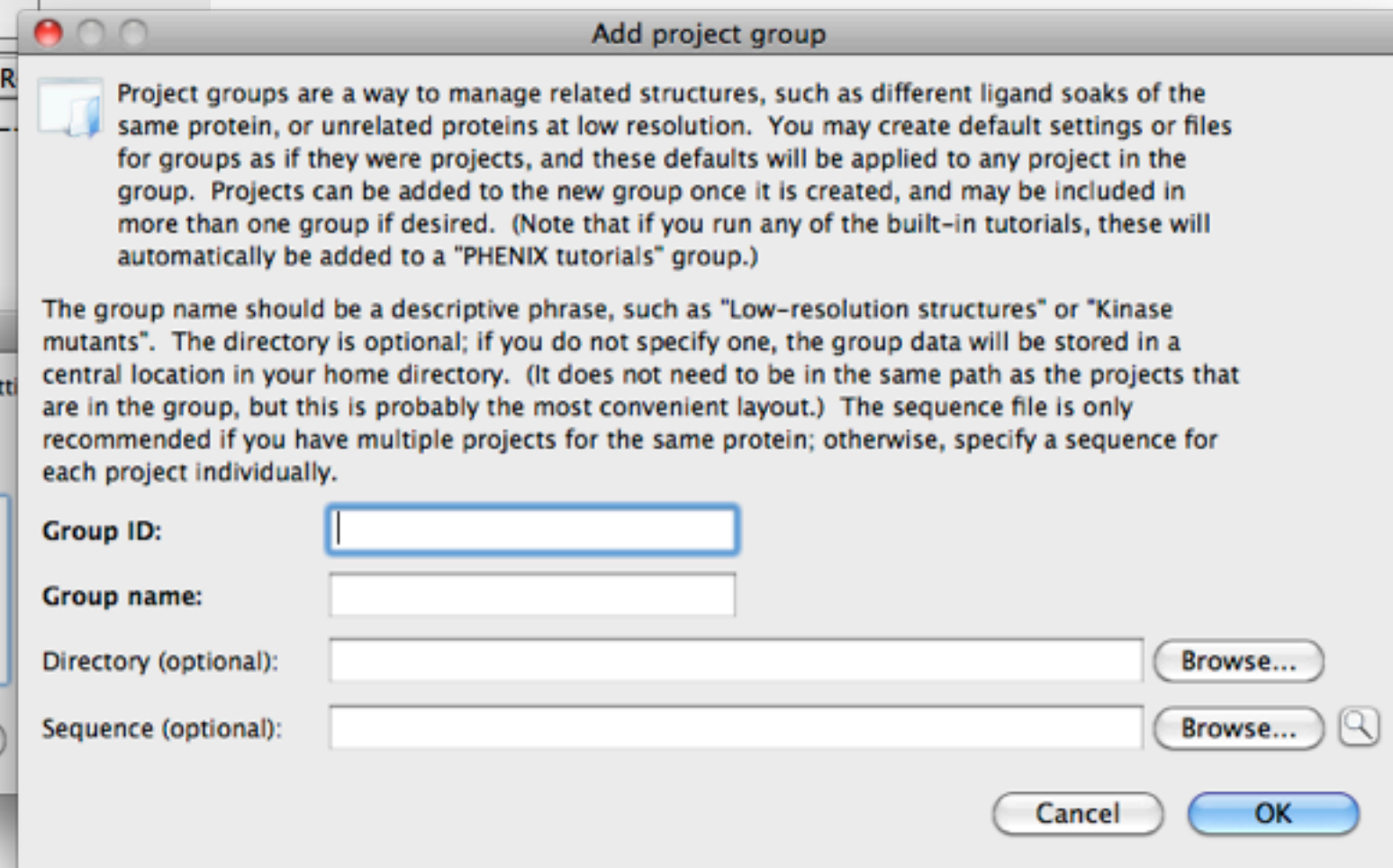
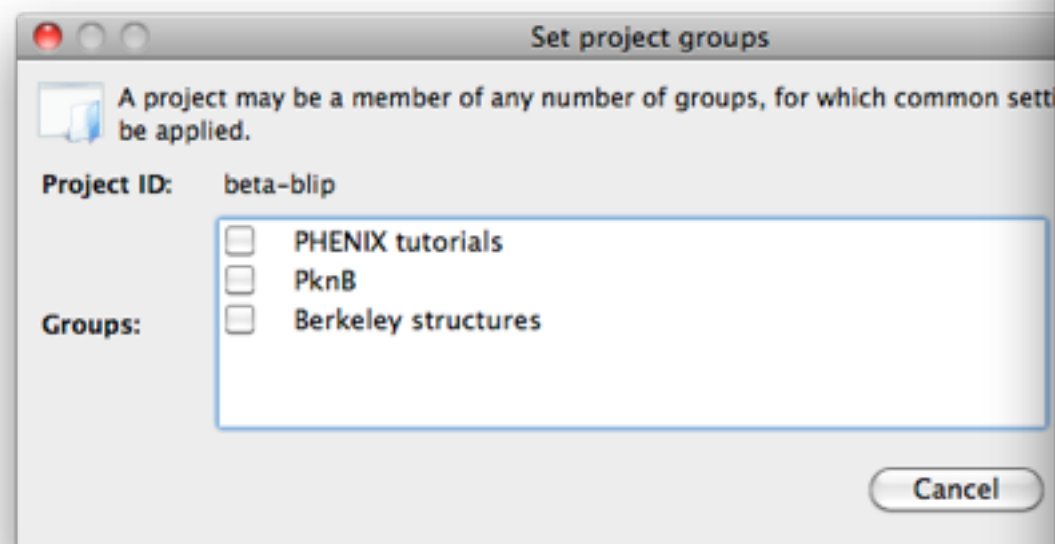
- You should not need to modify any of these files

New feature (January 2012): project groups

- An additional sorting layer, primarily for managing common files and settings - mostly* optional

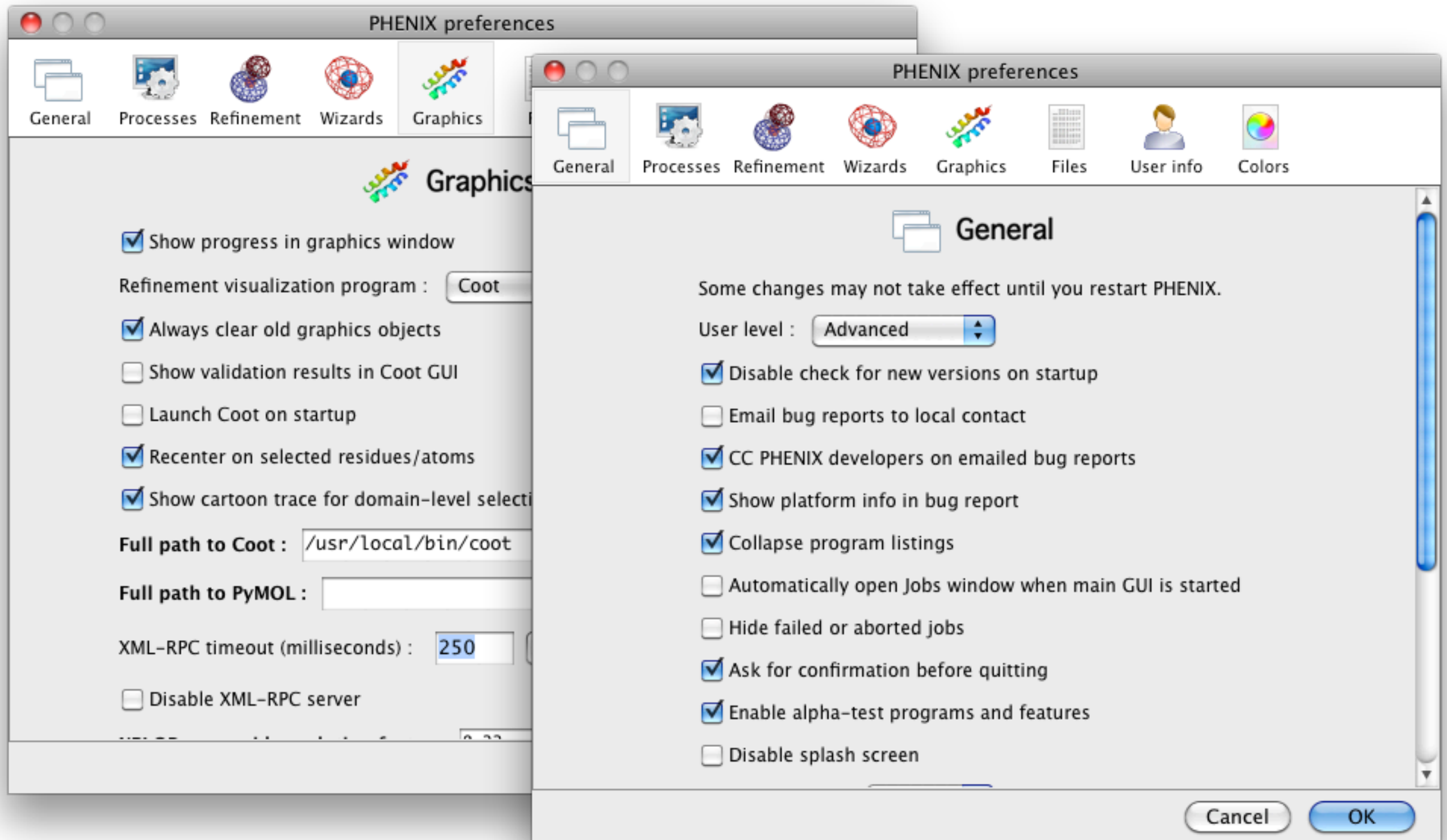


**Tutorials are automatically added to a "PHENIX tutorials" group - any other project will not be part of a group unless explicitly specified*



User preferences

- Settings for overall behavior and individual programs





Overview of available programs


- Central interface (“phenix” command)
- AutoSol, AutoMR, AutoBuild, and LigandFit wizards
- phenix.refine (and associated utilities)
- Xtrriage - comprehensive assessment of data quality
- Phaser - advanced interface for MR and SAD
- Validation - most of Molprobit, and more
- Several map-related interfaces
- Reflection file editor - combine files, create or extend R-free flags
- REEL - graphical restraints editor (*Nigel Moriarty*)
- 40+ programs currently available, more coming soon


Overview of available programs


Molecular replacement


 **AutoMR (simple interface)**
Automated molecular replacement with Phaser – search for a single ensemble

 **AutoMR (advanced interface)**
Automated molecular replacement with Phaser – supports multiple ensembles and components


 **Phaser-MR (simple interface) [alpha]**
Automated molecular replacement with Phaser – search for a single ensemble


 **Phaser-MR**
Maximum-likelihood molecular replacement


 **MRage – automated pipeline [alpha]**
Integrated model identification, preparation, and parallel MR search

 **MR-Rosetta (beta)**
AutoMR combined with Rosetta model improvement and AutoBuild for difficult structures


 **Sculptor**
Modify a molecular replacement search model


 **Sculptor – Coot interface**
Extension to Coot GUI for running Sculptor interactively


 **Ensembler**
Create ensemble of models for molecular replacement

 **Parallel Phaser [alpha]**
Evaluate many different search models independently across multiple processors


Experimental phasing

 **AutoSol**
Automated experimental phasing with model-building


 **Hybrid Substructure Search**
Dual-space identification of heavy-atom sites


 **Phaser-EP**
Maximum-likelihood SAD experimental phasing


Model building

 **AutoBuild**
Automated model-building and refinement

 **Phase and build**
Faster auto-building combined with density modification











 **Find Helices and Strands**
Fast chain tracing

 **Fit Loops**
Fast placement of missing loops in electron density







 **Morph model**
Model improvement for poor molecular replacement solutions

Overview of available programs





Maps

-  **Calculate maps**
Utility for creating likelihood-weighted maps using model phases, output as MTZ file or CCP4-format maps
-  **FFT map coefficients**
Generate map file(s) (in CCP4 or XPLOR format) from an MTZ file containing one or more sets of map coefficients
-  **AutoBuild – create omit map**
Simulated annealing and iterative-build omit maps using the AutoBuild wizard
-  **Density modification [alpha]**
Simple density modification by solvent flipping
-  **RESOLVE density modification**
Simple interface for running density modification only using AutoBuild and RESOLVE
-  **Isomorphous difference map**
Create Fo-Fo map from isomorphous datasets
-  **Superpose maps**
Superpose two PDB files and transform the associated map coefficients to the new orientation
-  **Cut out density**
Extract an arbitrary user-defined region from map coefficients file
-  **Multi-crystal averaging**
Density modification with multi-crystal averaging of maps
-  **Find difference map peaks and holes**
Identify local maxima and minima in mFo-DFc map (and anomalous map if available) and flag waters with excess density

Ligands









-  **LigandFit**
Fitting of ligands into electron density
-  **eLBOW**
electronic Ligand Building and Optimization Workbench: ligand restraints generation and optimization
-  **eLBOW (wizard-style) [alpha]**
Simplified interface to eLBOW
-  **REEL**
Ligand restraints viewer and editor
-  **Ligand identification**
Automated ligand search using database of 200 most frequent ligands, or user input
-  **Guided ligand replacement**
Ligand fitting based on an existing protein-ligand complex

Refinement







-  **phenix.refine**
Automated X-ray and neutron refinement
-  **DEN refinement [alpha]**
Deformable elastic network refinement using simulated annealing, for low-resolution and molecular replacement structures
-  **ReadySet**
Utility for preparing PDB files for refinement – automatically generate restraints and add hydrogens
-  **Covalent ligand setup [alpha]**
Generate link restraints for various covalent ligands, such as carbohydrate groups

Overview of available programs





Reflection tools

-  **Xtrriage**
Analysis of data quality and crystal defects
-  **Reflection file editor**
Utility for merging and converting reflections
-  **Calculate F(model)**
Utility for generating structure factors from a PDB file
-  **Import CIF structure factors**
Convert data deposited in PDB to MTZ format
-  **French & Wilson data correction**
Statistics-based handling of negative intensities
-  **Model-based phases**
Utility to obtain PHI, FOM, and Hendrickson-Lattman coefficients from data and PDB file
-  **3D data viewer**
OpenGL visualization of reflection data in reciprocal space
-  **2D data viewer**
Visualization of reflection data in slices through reciprocal space




Model tools

-  **PDB Tools**
Utility for simple modifications of PDB files, including geometry regularization
-  **Combine PDB files**
Merge a model split across multiple files, with automatic chain renumbering and clash check
-  **Superpose PDB files**
Simple structure alignment program
-  **Find NCS operators**
Identify non-crystallographic symmetry in model, heavy-atom sites, or electron density map
-  **Apply NCS operators**
Transform a molecule by NCS matrices to generate complete structure
-  **Add conformations [alpha]**
Add alternate conformations in bulk, for an entire model or user-defined atom selection

Utilities

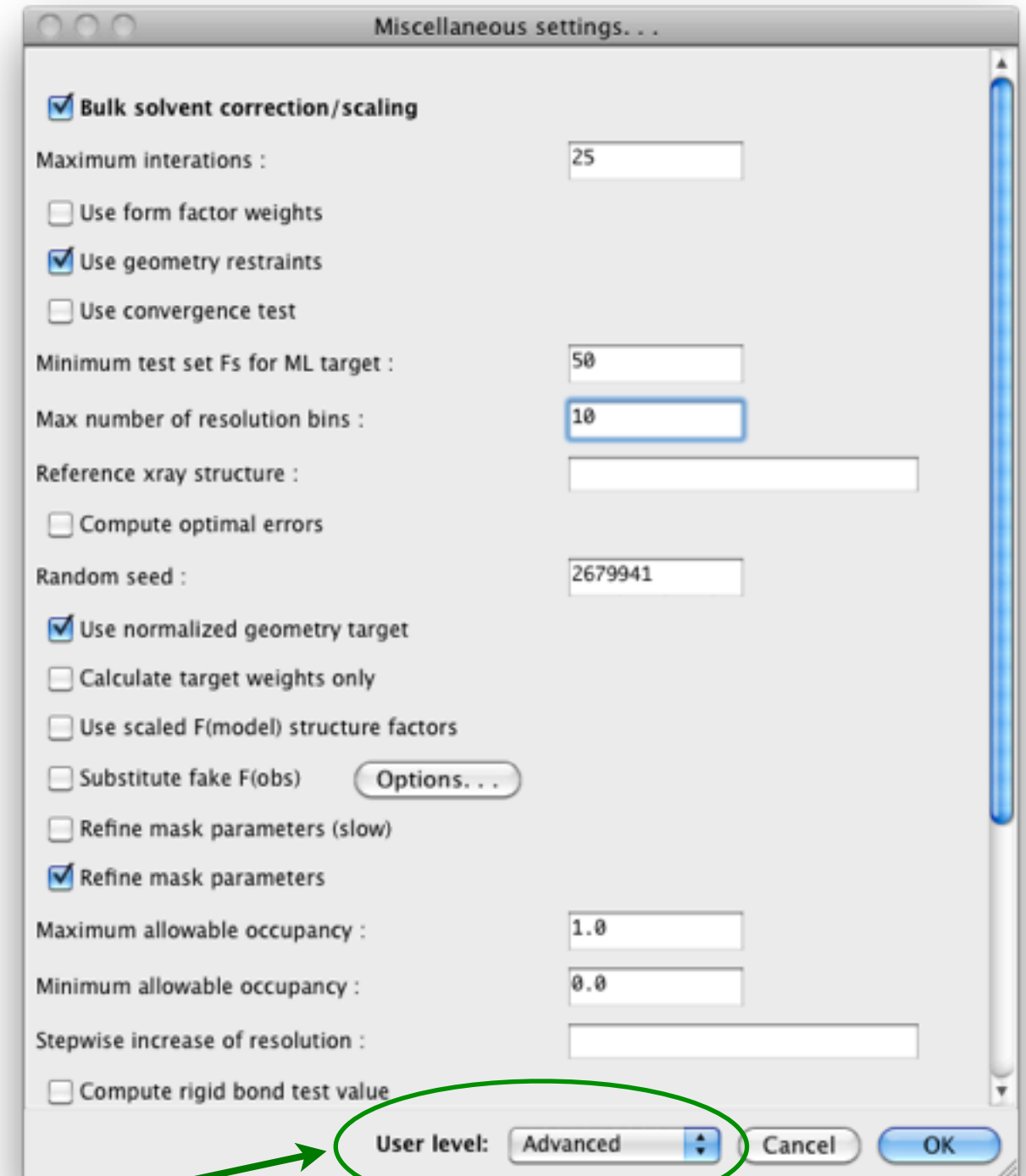
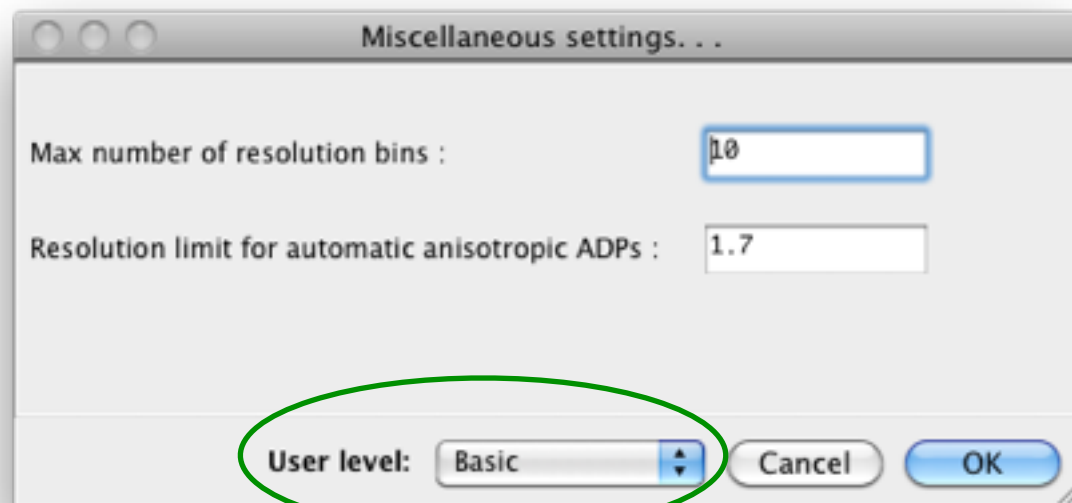
-  **Calculate map correlations**
Simple calculation of CC between two sets of map coefficients in MTZ format, accounting for origin shifts
-  **Calculate model-map correlations**
Simple calculation of CC between a PDB file and map coefficients in MTZ format, accounting for origin shifts
-  **Generate "Table 1" for journal**
Extraction of final model statistics for publication, including data processing logfiles
-  **Diffraction image viewer**
Simple viewer for detector images, similar to ADXV

Validation

-  **Comprehensive validation**
Model quality assessment, including real-space correlation and geometry inspection using Molprobity tools
-  **POLYGON**
Graphical comparison of validation statistics and the PDB
-  **Structure comparison**
Identify differences between multiple structures of the same protein, using multiple criteria

“Expert level” control in dialog windows

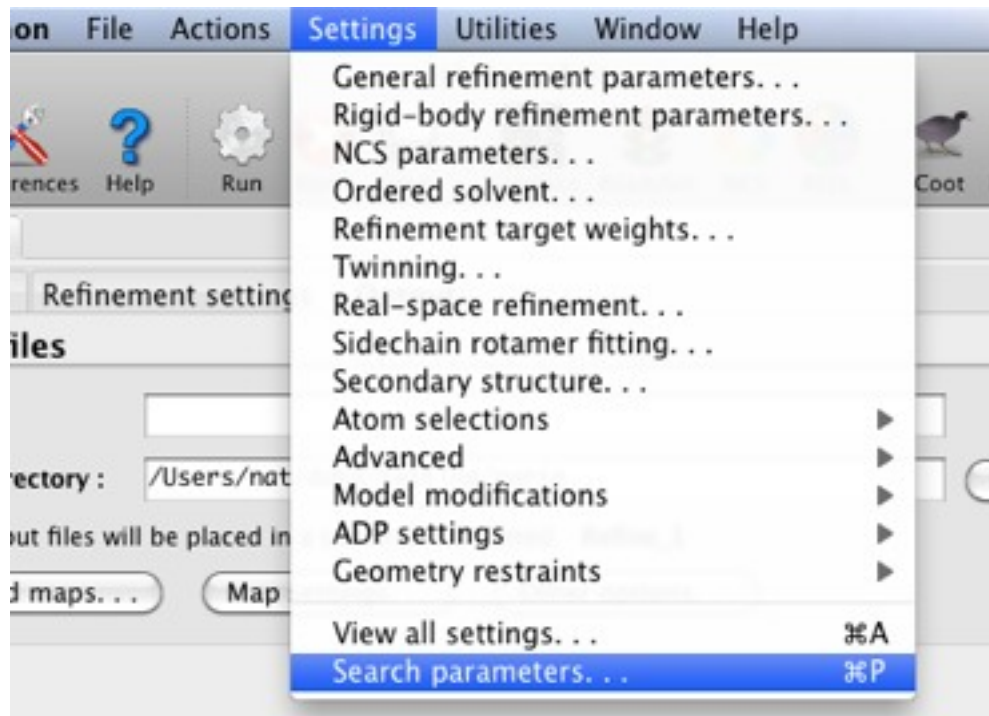
Configuration interfaces can be dynamically adjusted to show only the most basic/popular options, or more detail:



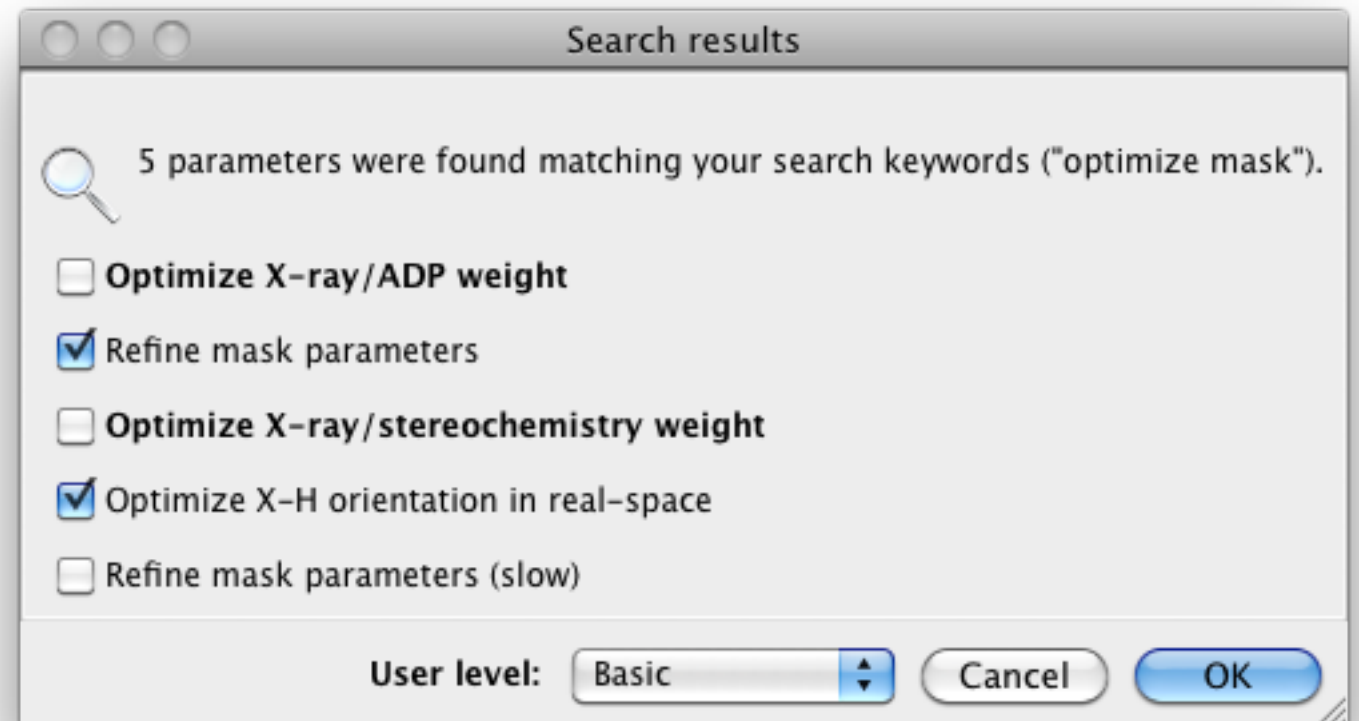
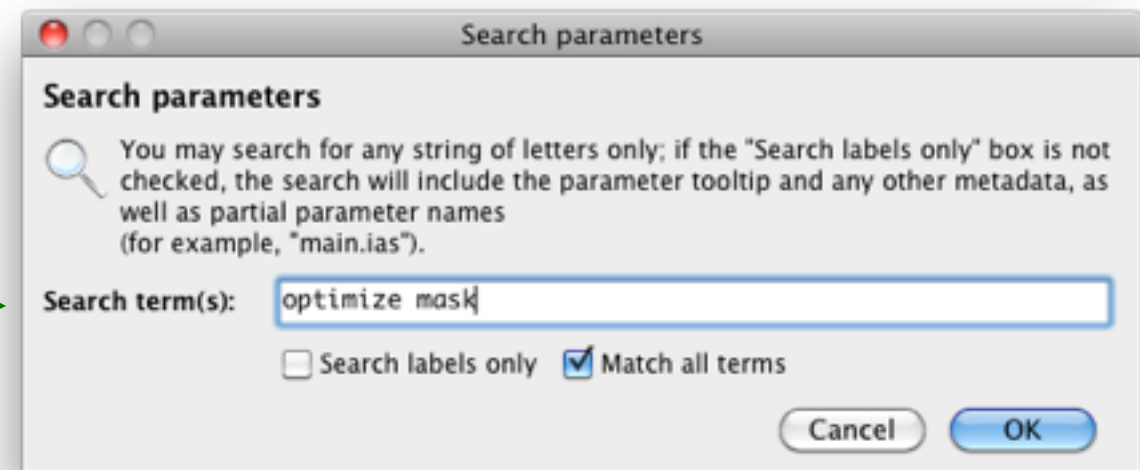
Changing the “user level” shows or hides advanced controls; the default level can be set in the Preferences

Keyword search for parameters

Most documentation covers the command-line parameters; the corresponding GUI controls may be easiest to find with the search

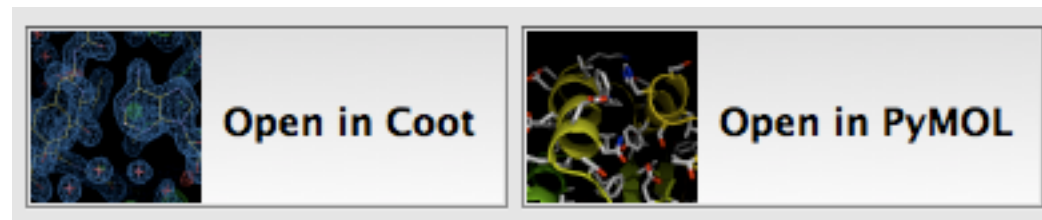


It helps to be as specific as possible: searching for “mask” alone finds 19 parameters, “optimize mask” (shown) finds 5, and “optimize_mask” (a specific parameter name) finds 2.

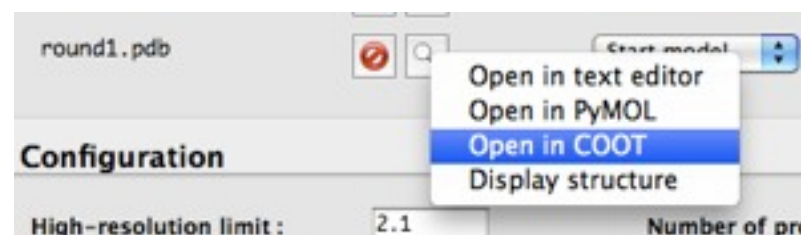


Coot/PyMOL integration

- Coot must have Python support (default on Mac)
- Specific paths to executables usually required on Linux
- *Preferences->Graphics->Full path to Coot [...PyMOL]*
- Most results can be opened directly in graphics apps



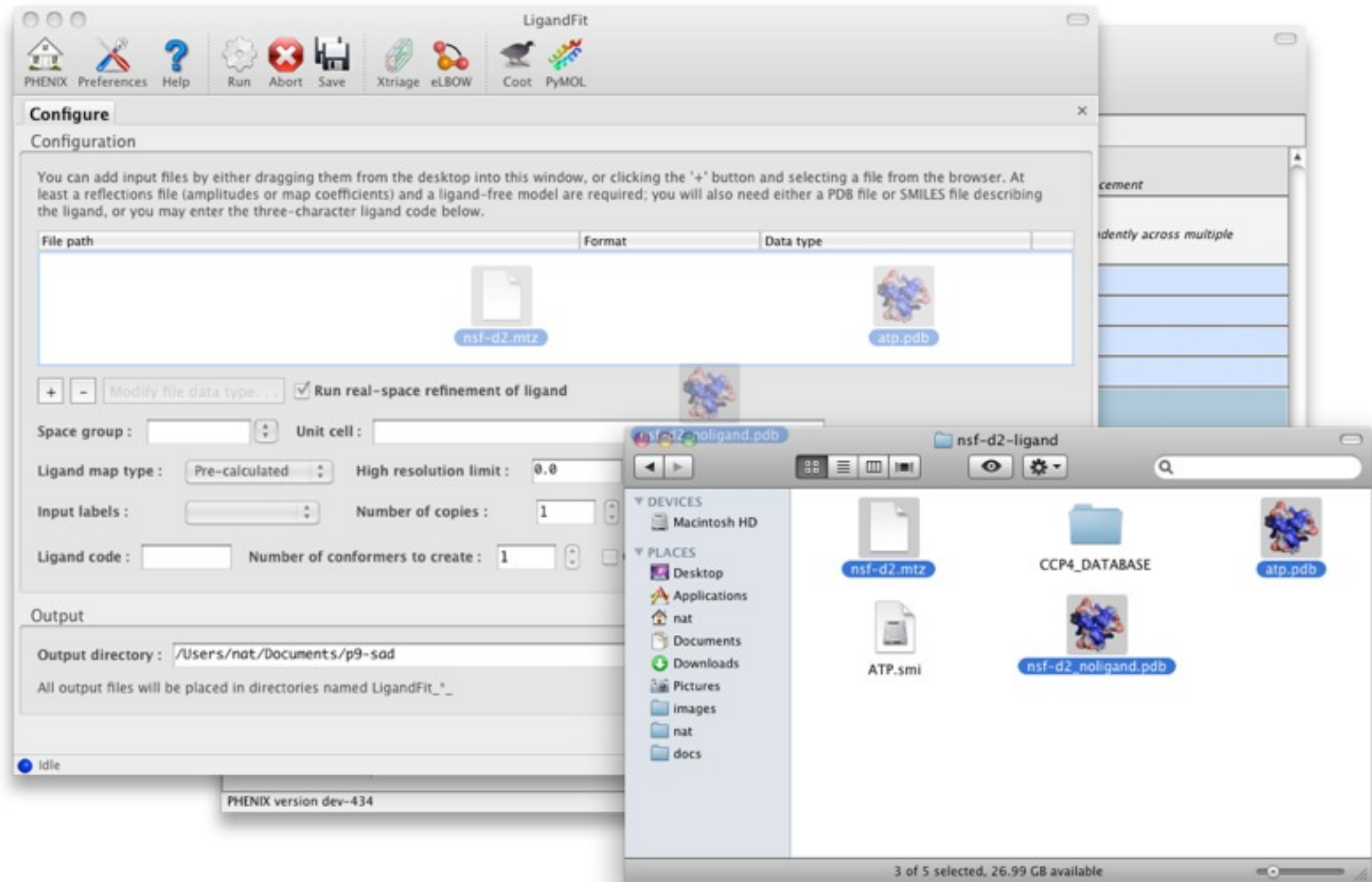
- Any PDB file listed in GUI can also be opened



- AutoSol, AutoBuild, and phenix.refine will update Coot continuously while running

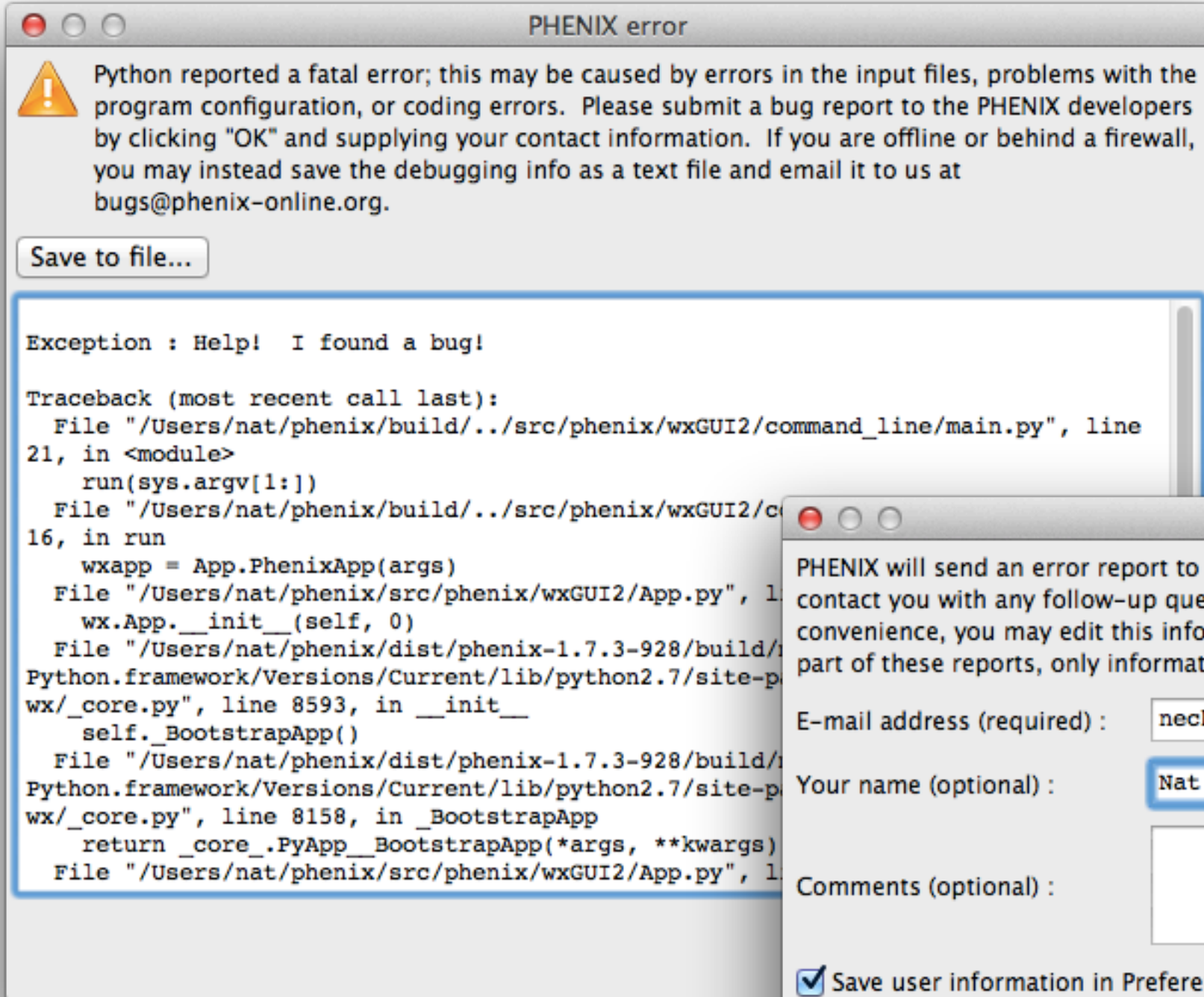
File management

- Bulk file input possible on both Mac + Linux



Automatic bug reports

- Sent via email to Phenix developers - please submit!

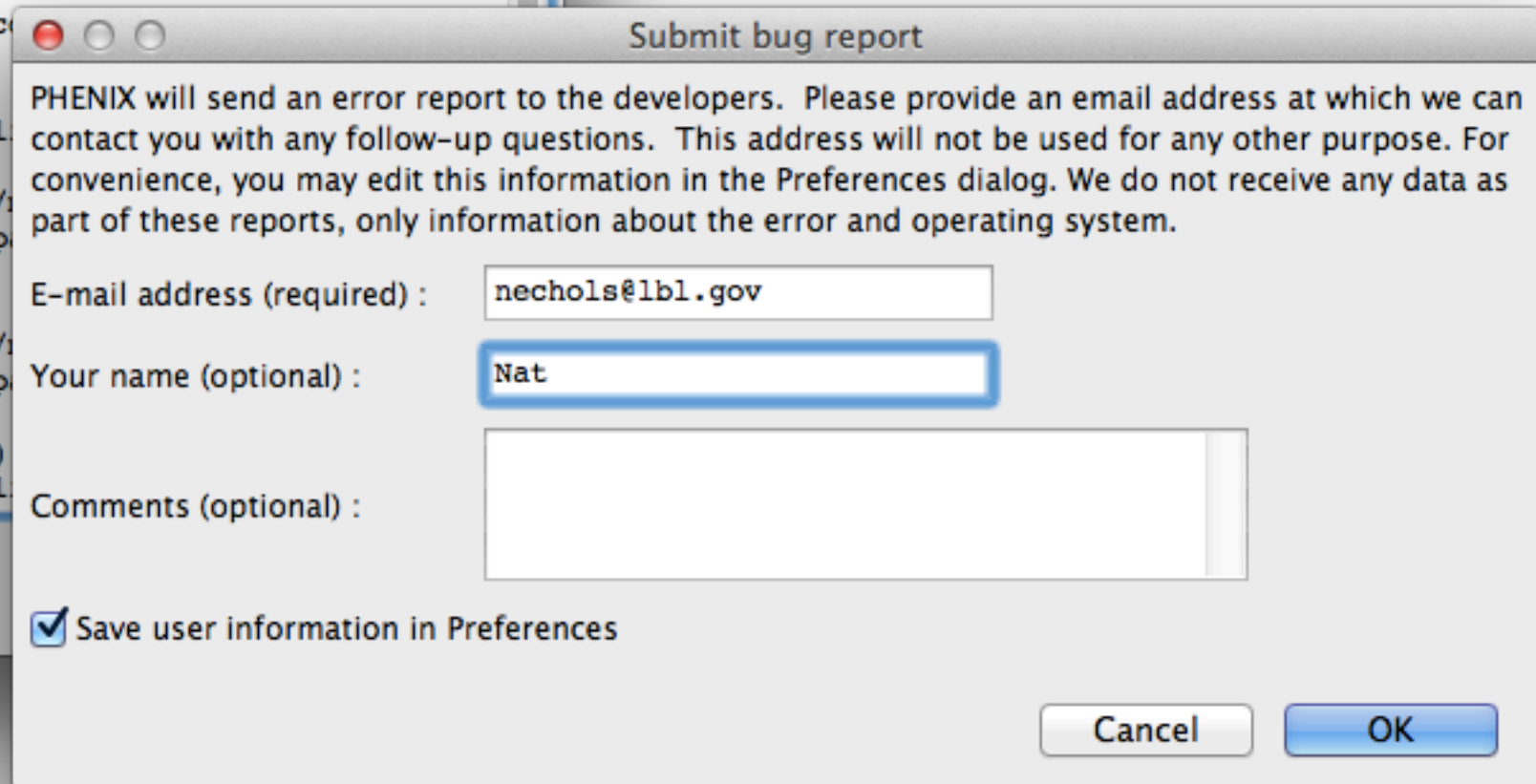


PHENIX error

Python reported a fatal error; this may be caused by errors in the input files, problems with the program configuration, or coding errors. Please submit a bug report to the PHENIX developers by clicking "OK" and supplying your contact information. If you are offline or behind a firewall, you may instead save the debugging info as a text file and email it to us at bugs@phenix-online.org.

Save to file...

```
Exception : Help! I found a bug!  
Traceback (most recent call last):  
  File "/Users/nat/phenix/build/./src/phenix/wxGUI2/command_line/main.py", line  
21, in <module>  
    run(sys.argv[1:])  
  File "/Users/nat/phenix/build/./src/phenix/wxGUI2/c  
16, in run  
    wxapp = App.PhenixApp(args)  
  File "/Users/nat/phenix/src/phenix/wxGUI2/App.py", line  
    wx.App.__init__(self, 0)  
  File "/Users/nat/phenix/dist/phenix-1.7.3-928/build/  
Python.framework/Versions/Current/lib/python2.7/site-p  
wx/_core.py", line 8593, in __init__  
    self._BootstrapApp()  
  File "/Users/nat/phenix/dist/phenix-1.7.3-928/build/  
Python.framework/Versions/Current/lib/python2.7/site-p  
wx/_core.py", line 8158, in _BootstrapApp  
    return _core.PyApp_BootstrapApp(*args, **kwargs)  
  File "/Users/nat/phenix/src/phenix/wxGUI2/App.py", line
```



Submit bug report

PHENIX will send an error report to the developers. Please provide an email address at which we can contact you with any follow-up questions. This address will not be used for any other purpose. For convenience, you may edit this information in the Preferences dialog. We do not receive any data as part of these reports, only information about the error and operating system.

E-mail address (required) :

Your name (optional) :

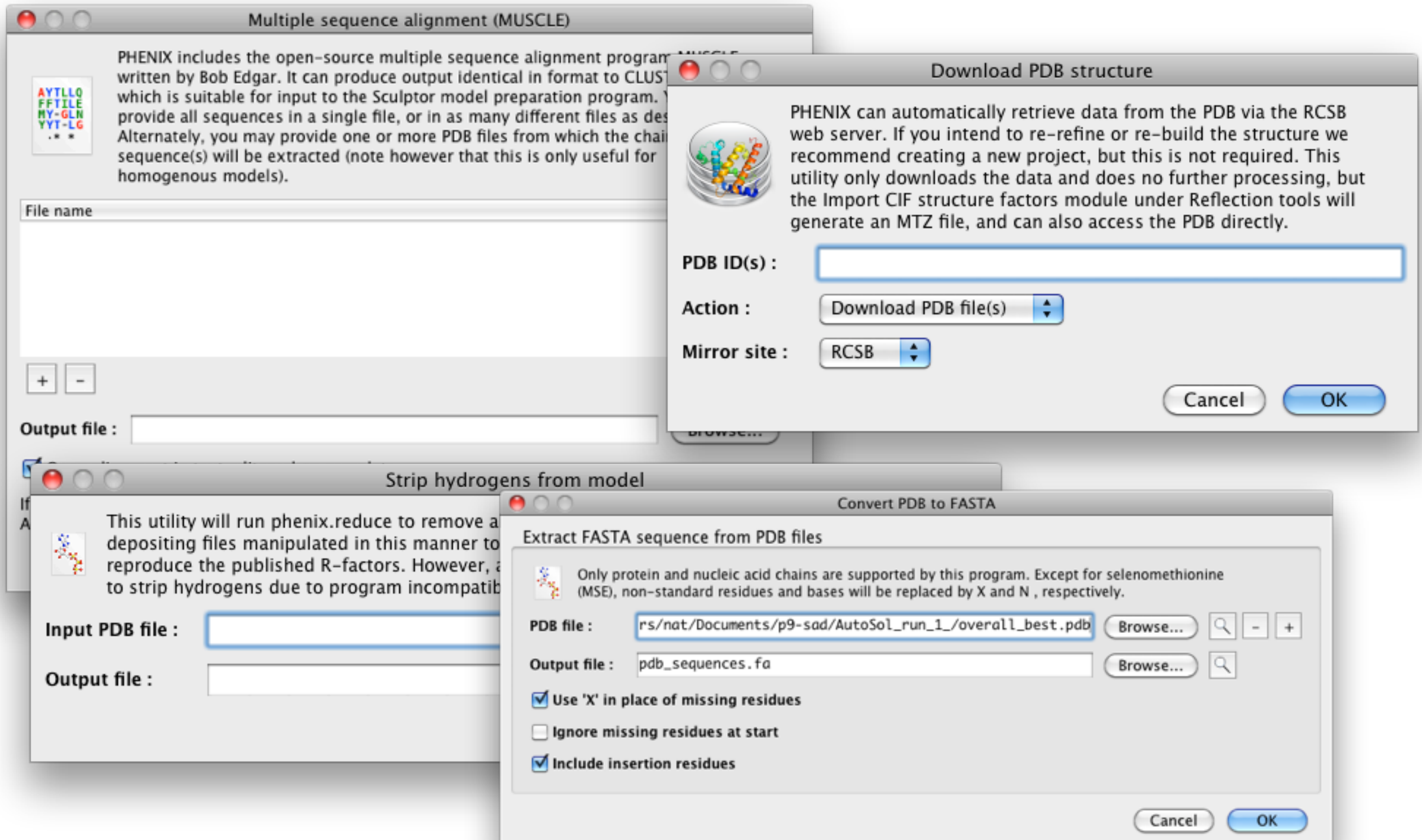
Comments (optional) :

Save user information in Preferences

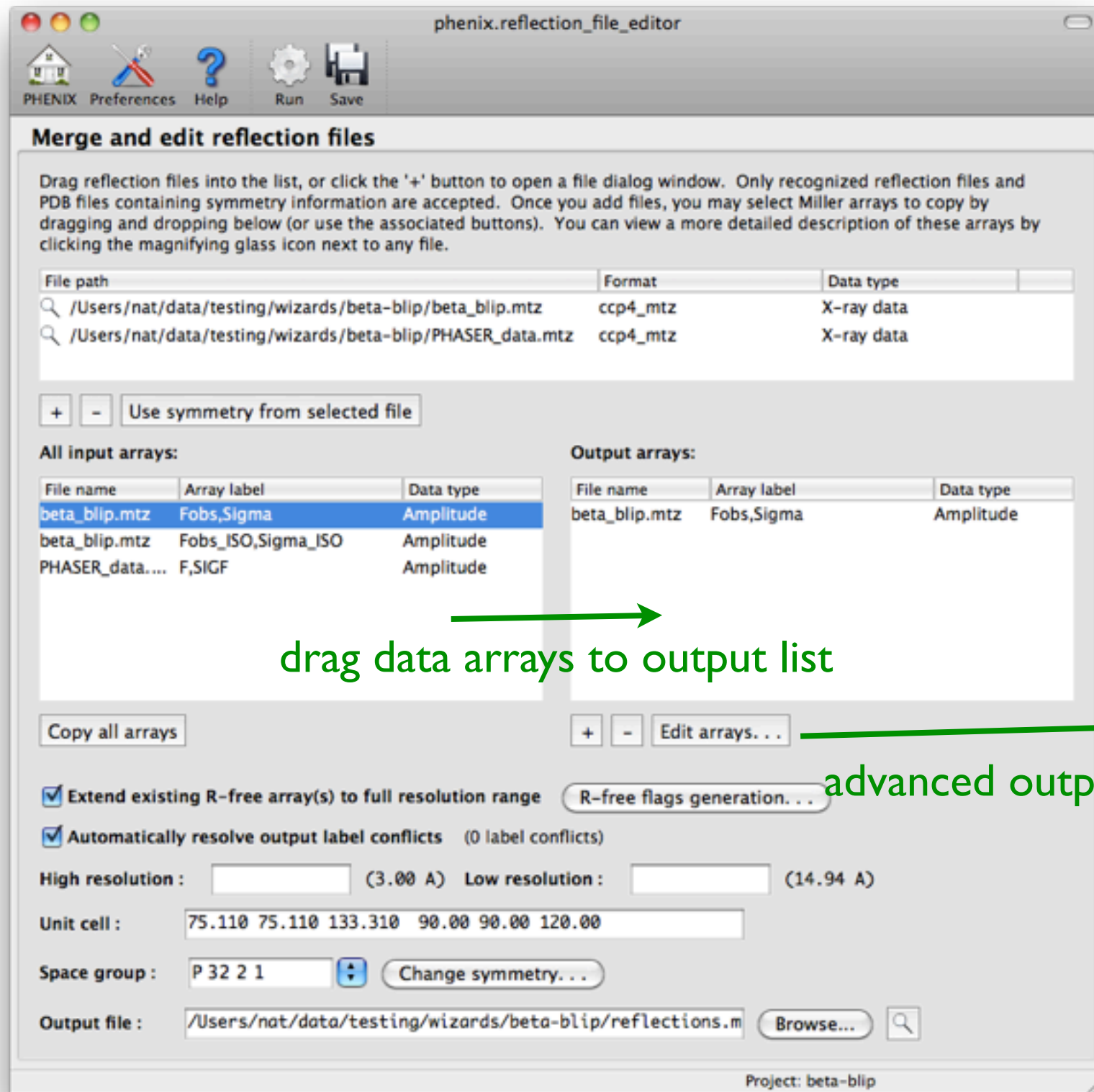
Cancel OK

Utility functions (“Other tools” in main GUI)

- Convenient access to very small and fast programs



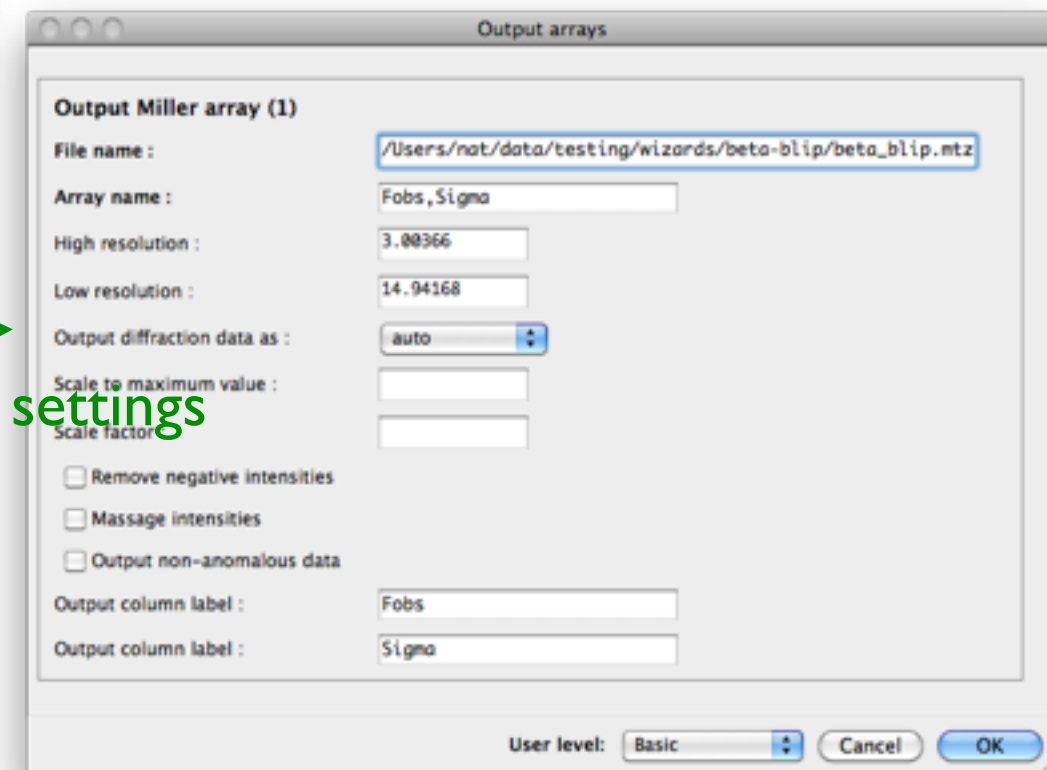
Reflection file editor



drag data arrays to output list

advanced output settings

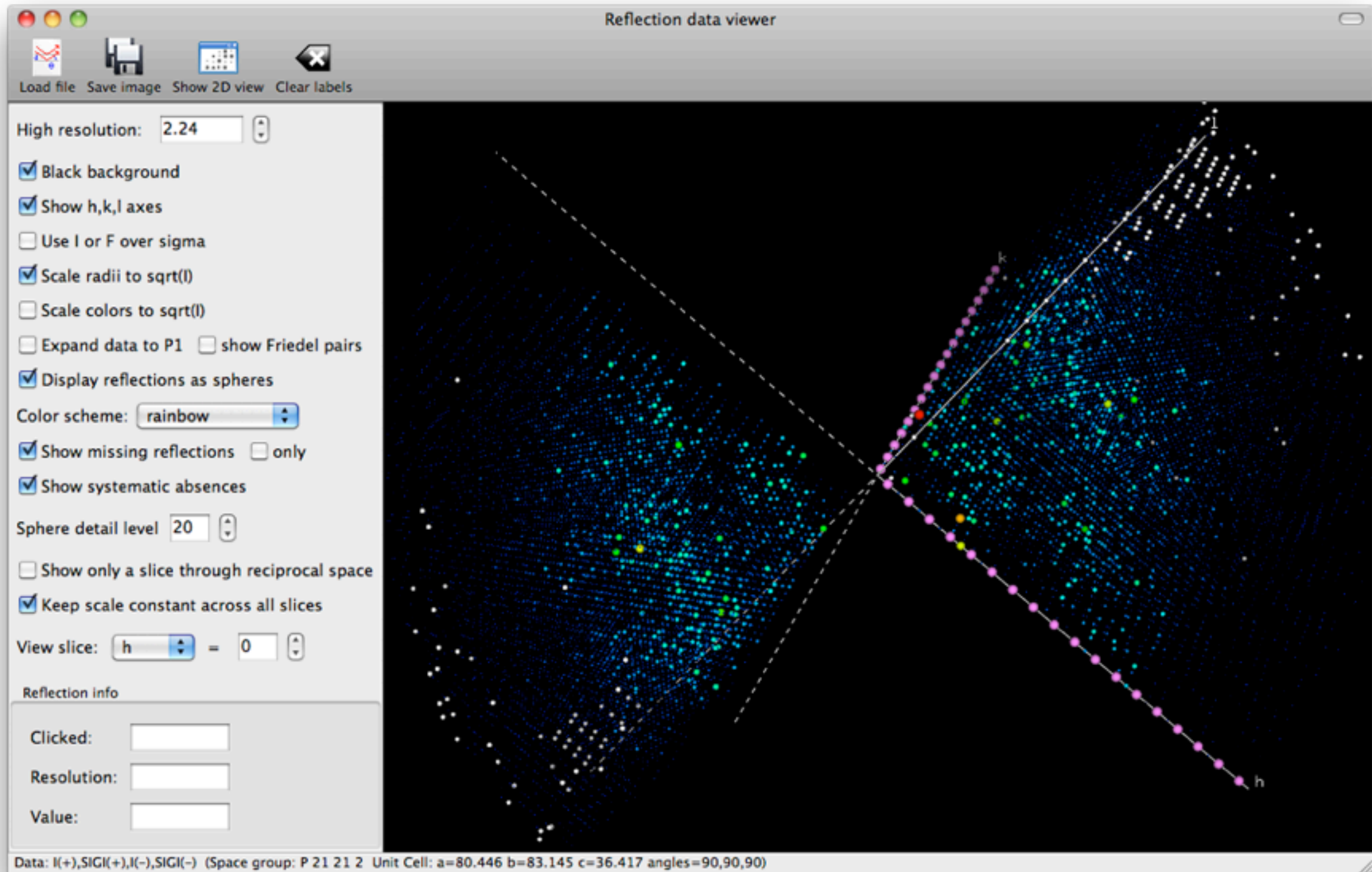
Combine and manipulate reflection files in any format, output as MTZ. Capable of extending old R-free sets, and generating new sets as thin shells (for refinement in presence of NCS). *For use with fully processed data only - reflections will be merged and h,k,l indices altered as required.*



(All functionality is also available on the command line as `iotbx.reflection_file_editor`, but we recommend using the GUI for this unless you are scripting an automation pipeline.)

phenix.data_viewer: visualizing reflections in 3D

Useful for identifying pathologies and other dataset properties



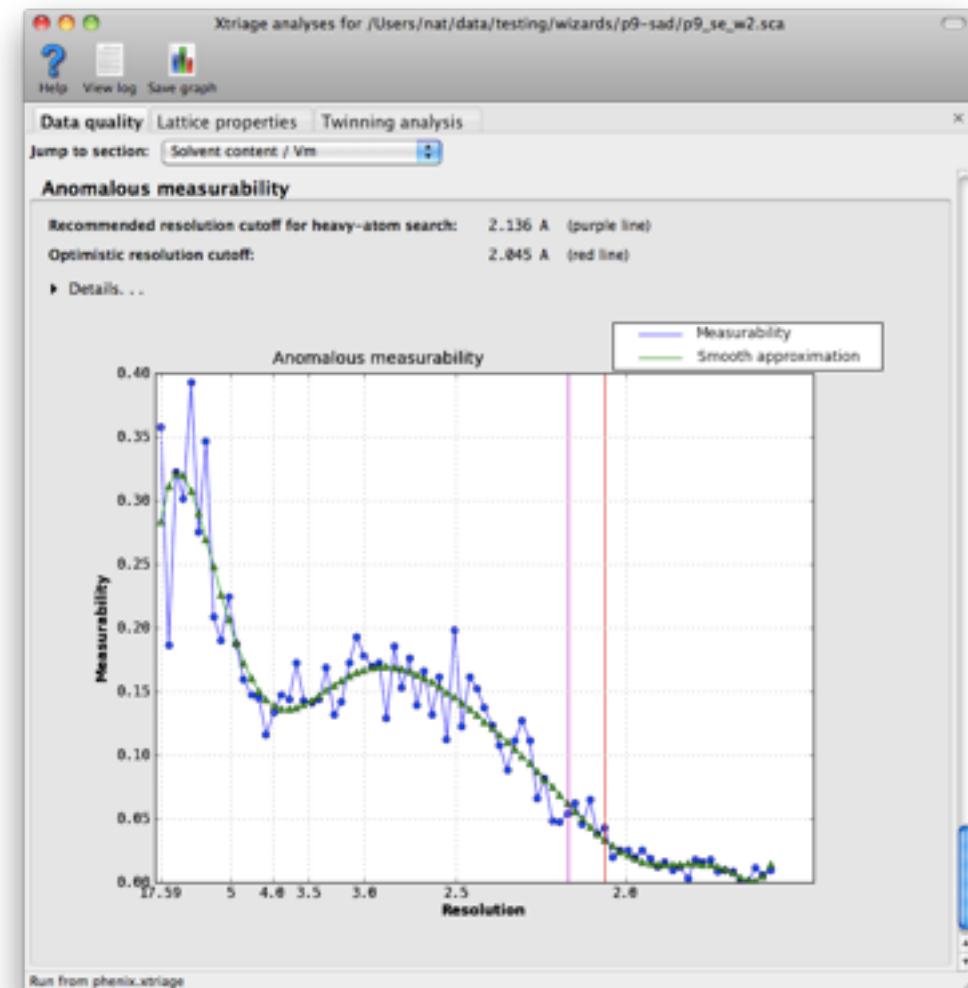
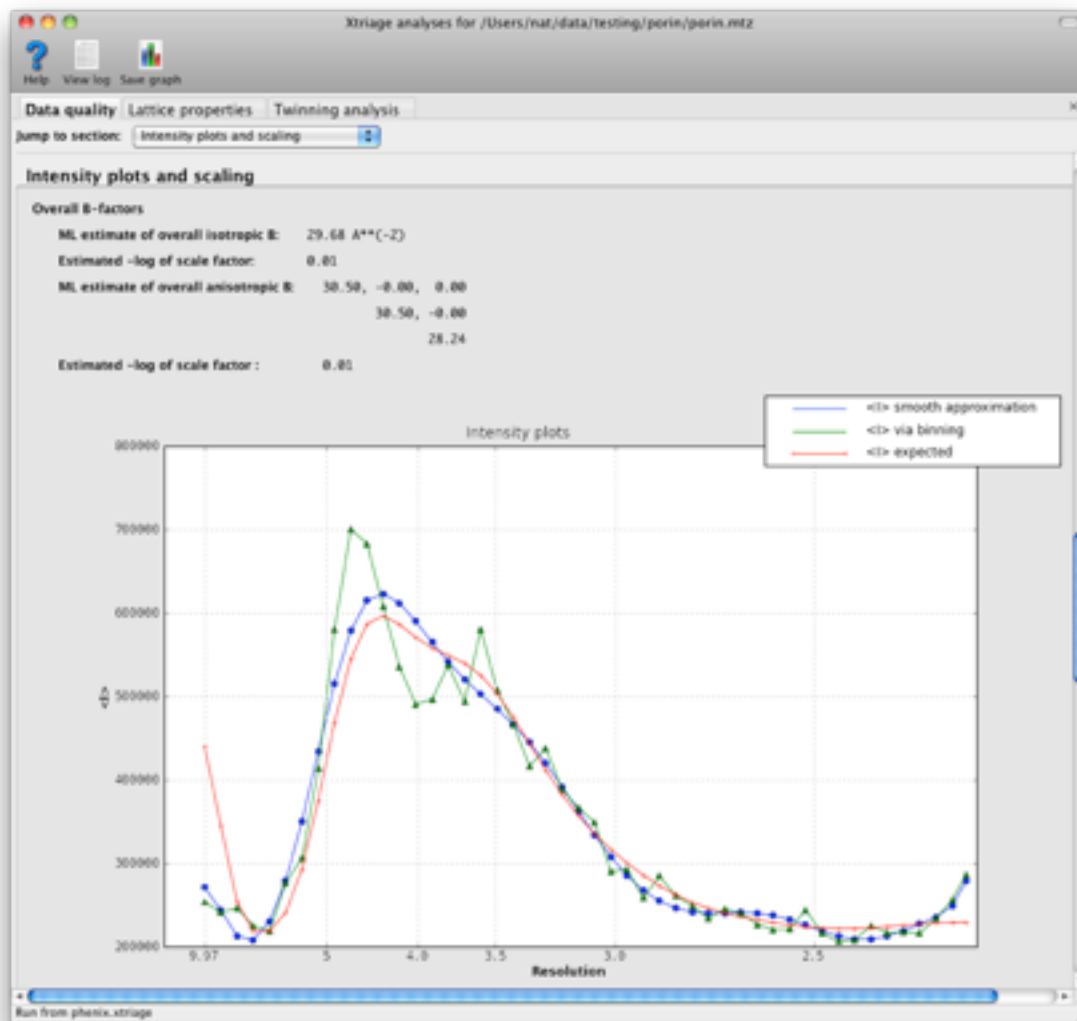
Anomalous data in P21212, showing missing reflections (white) and systematic absences (violet)

Data analysis with *phenix.xtrriage*

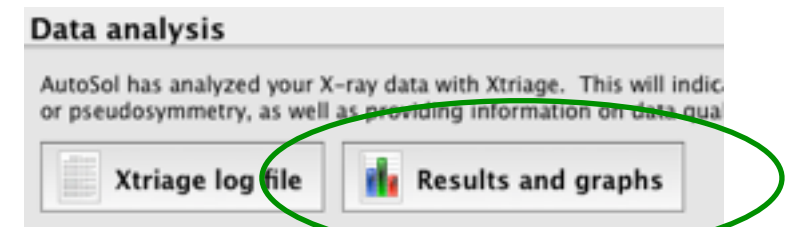
- Analysis of signal-to-noise, data quality, Wilson plot, translational NCS, twinning, symmetry issues, and more

Wilson plot and B-factors for a typical protein crystal dataset (\$PHENIX/examples/porin-twin)

Anomalous signal vs. resolution for an excellent SeMet dataset (\$PHENIX/examples/p9-sad)



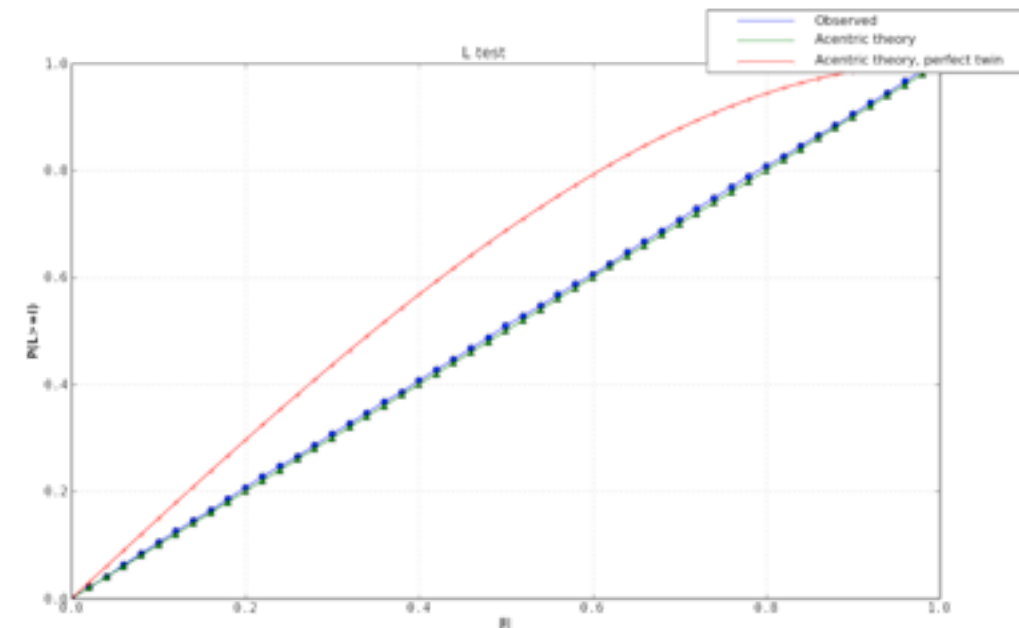
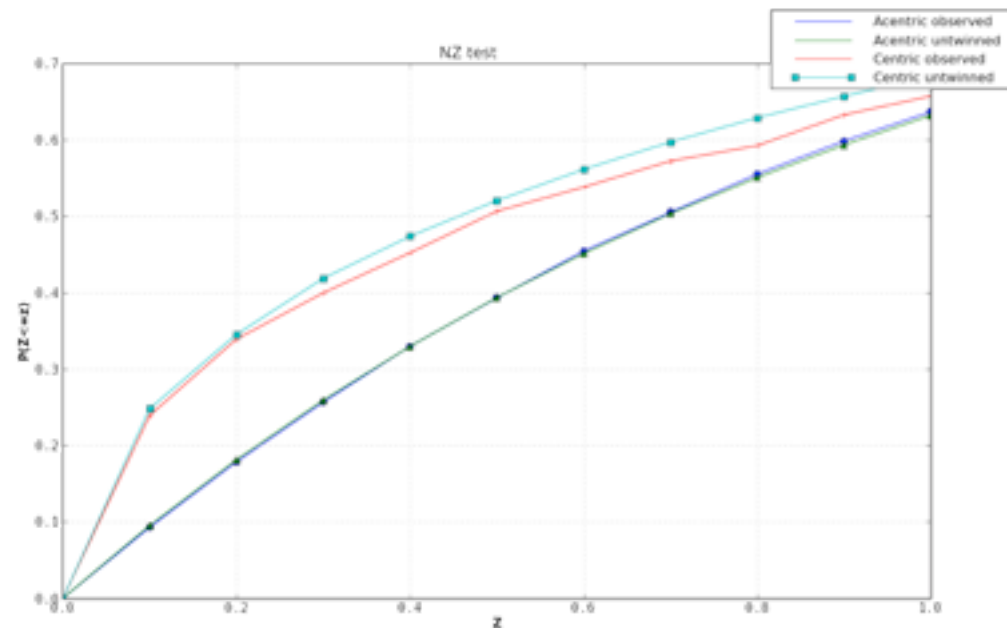
AutoSol and AutoBuild run Xtrriage almost immediately, and results can be viewed from those GUIs. However, it may save time and effort to run Xtrriage yourself first.



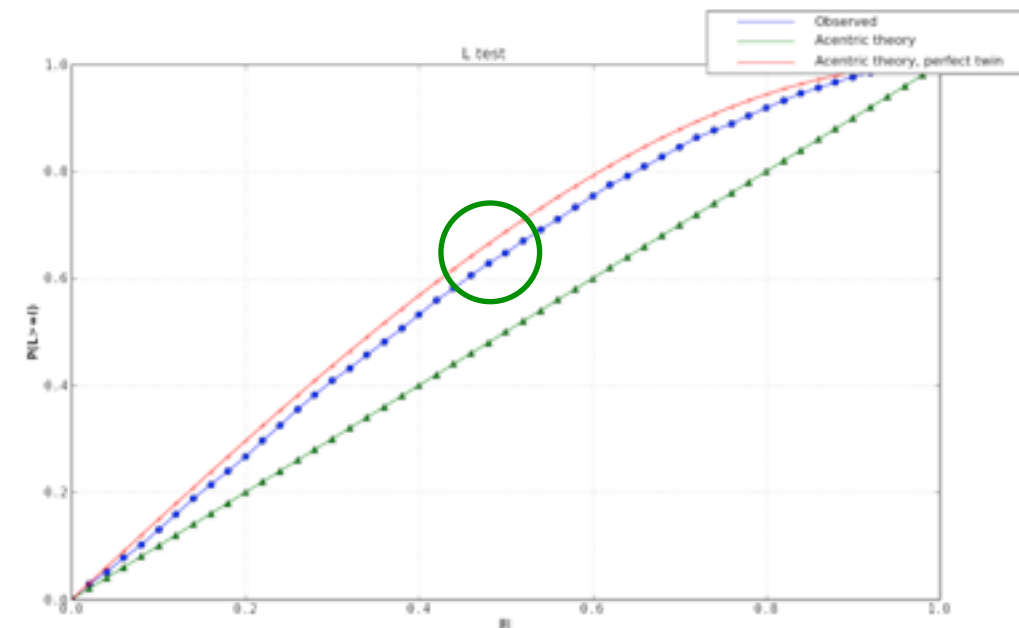
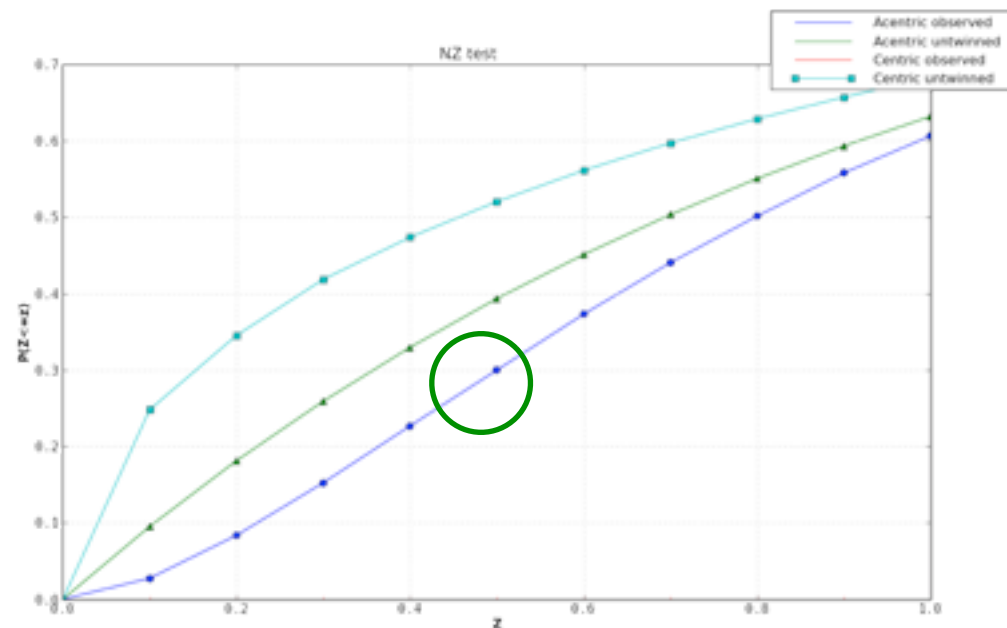
Identifying twinned structures in Xtrriage

Twinning can't be detected by looking at diffraction images, but it changes the distribution of intensity values in predictable ways

Good data (p9-sad example): observed intensity distributions are close to expected values



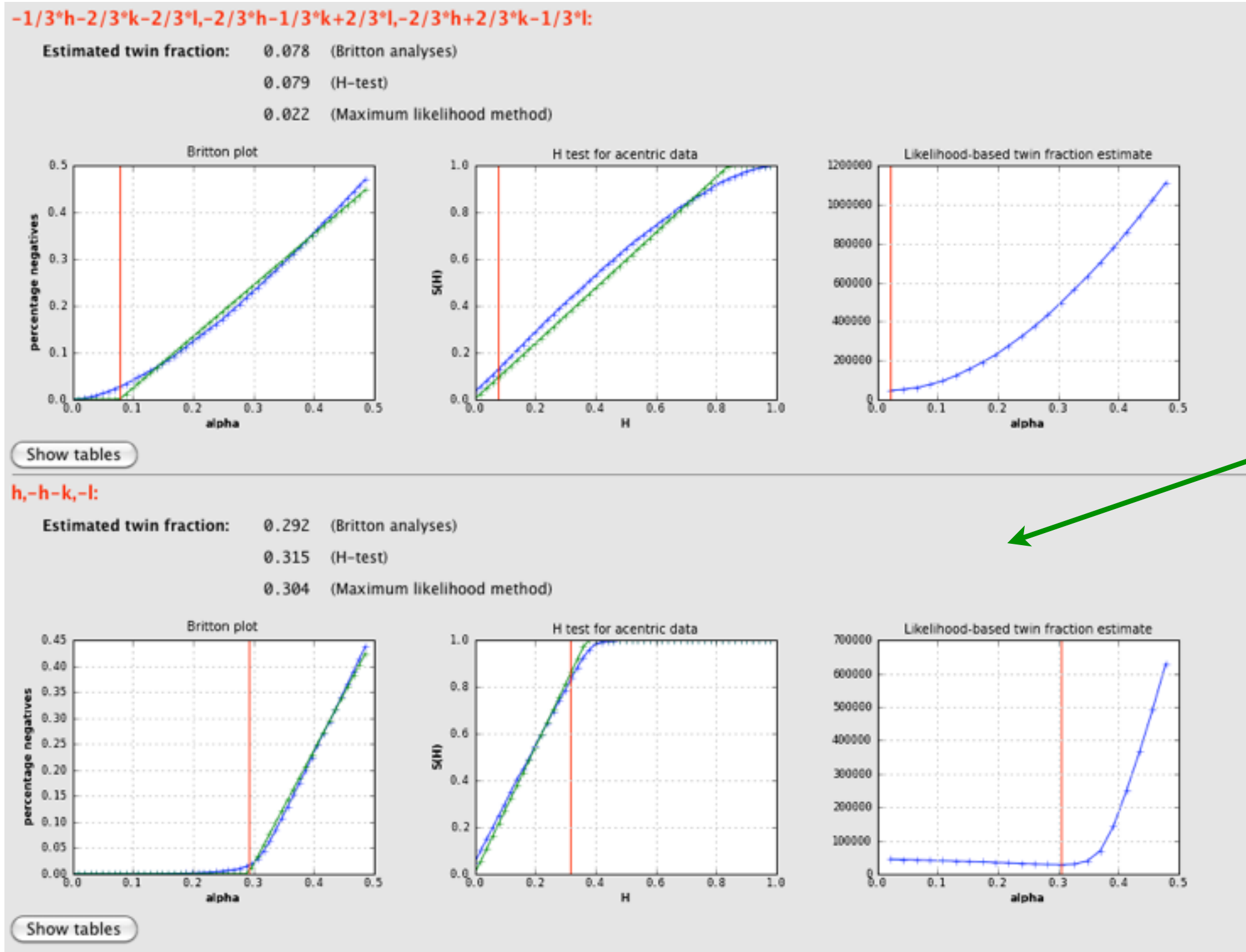
Twinned data (porin-twin example): NZ test curve is sigmoidal, L test curve is shifted upwards



Intensity distributions can also be affected by pseudotranslation (especially NZ test); make sure you look at all of the evidence for twinning!

Identifying twinned structures in Xtrriage

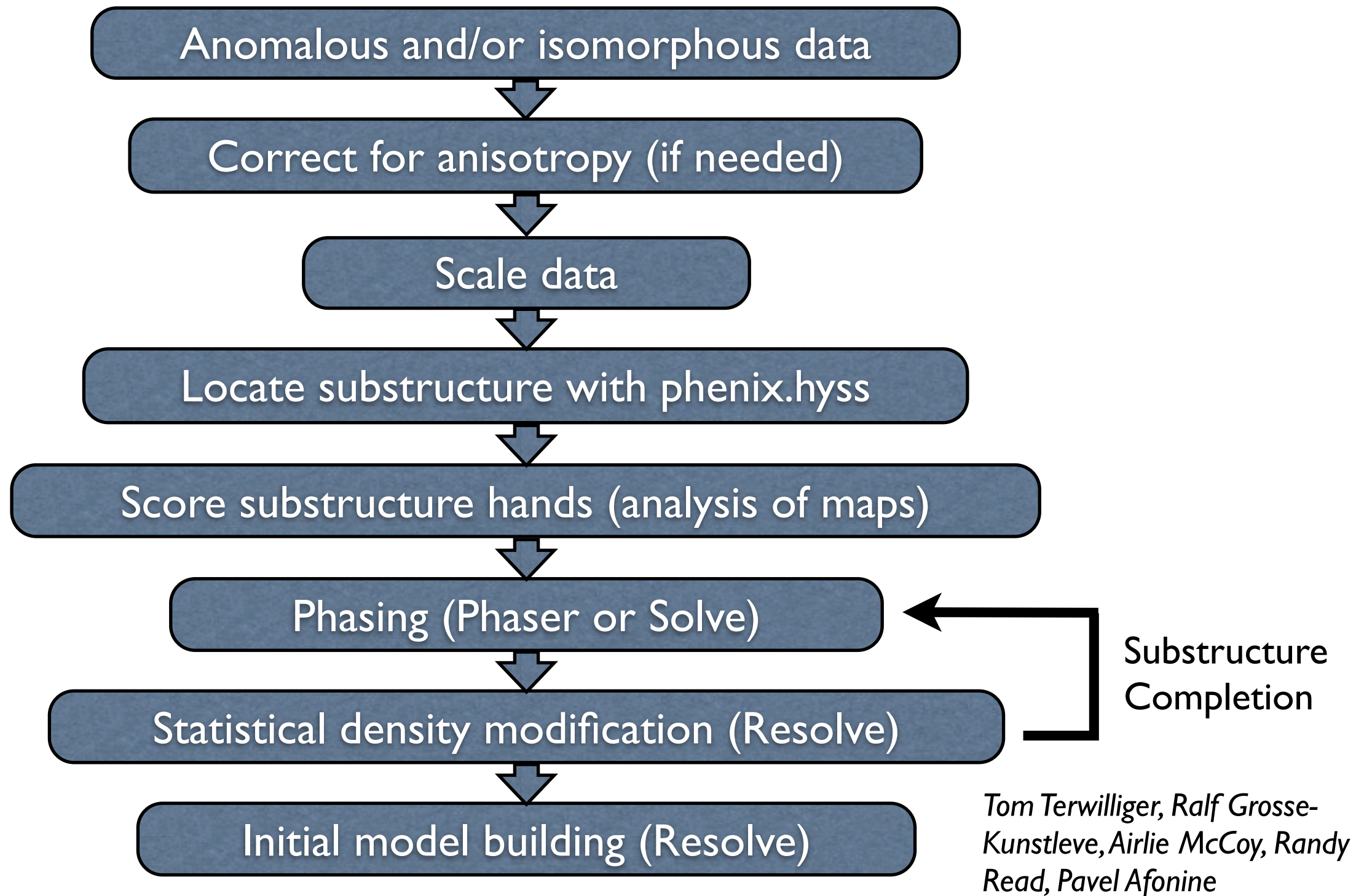
The twin fraction for all possible twin laws will be estimated; usually one of these is obviously different



Two twin laws from the porin-twin example are shown; in this case $h,-h-k,-l$ is the actual twin law for this crystal. This can be used in phenix.refine, which will determine the true twin fraction based on the refined model.

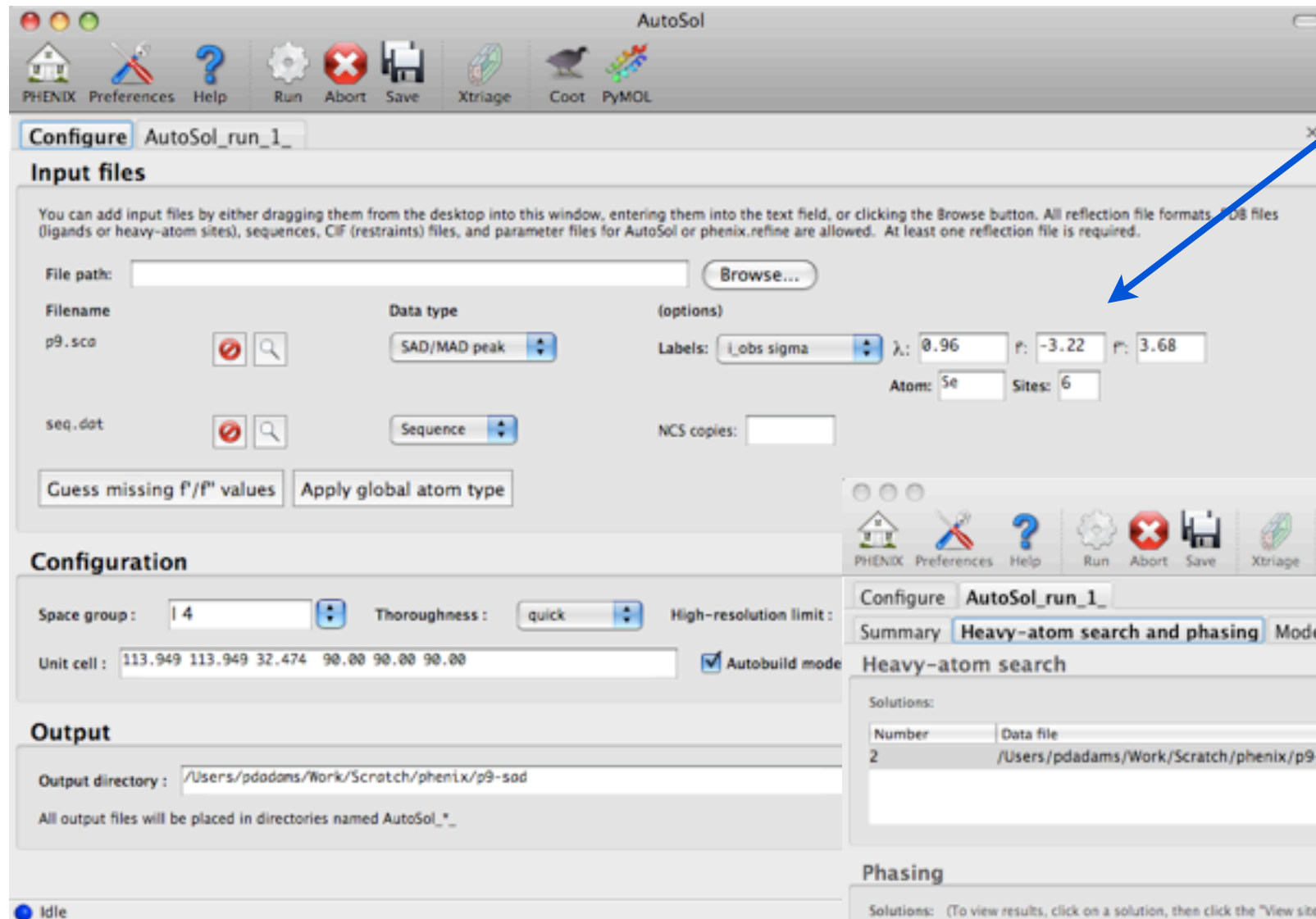
The validation GUI (or phenix.model_vs_data) will also try to determine if your structure is twinned based on the R-factors with and without a twin law.

AutoSol: an experimental phasing pipeline



Terwilliger et al: Decision-making in structure solution using Bayesian estimates of map quality: the PHENIX AutoSol wizard. Acta Cryst. 2009, D65:582-601.

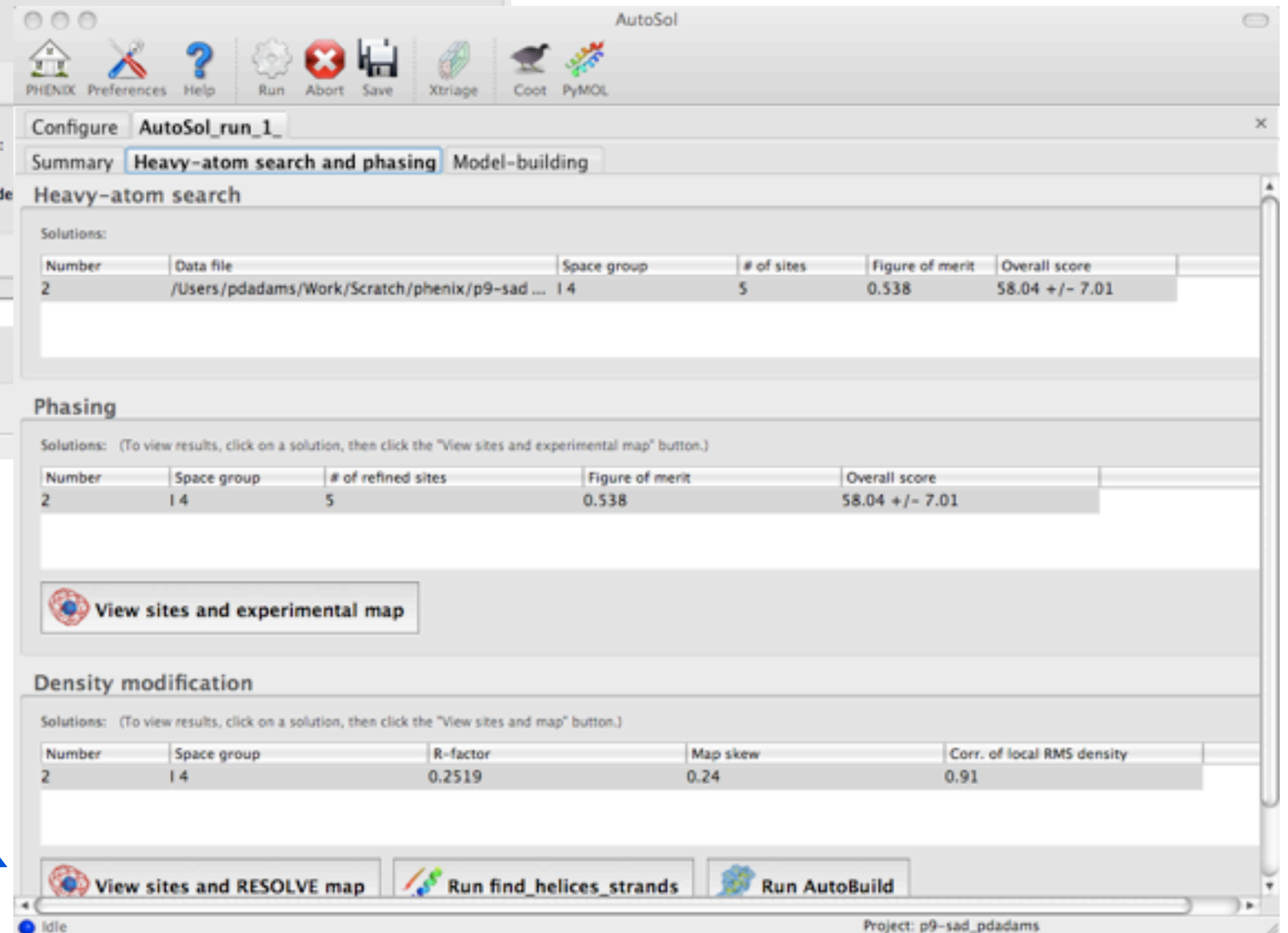
The AutoSol graphical interface



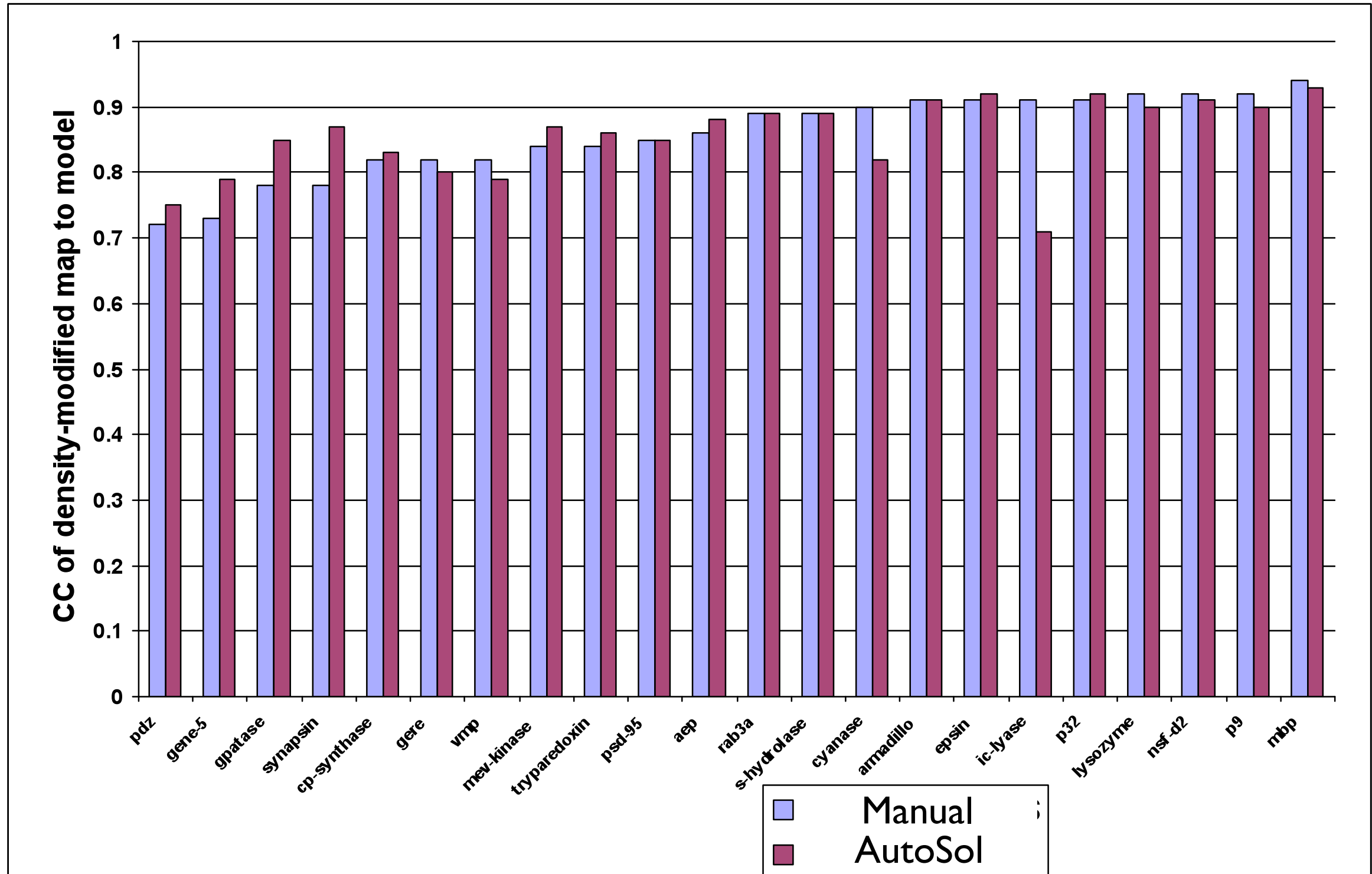
Files added by dragging from desktop into the window

Very little input required for simple SAD experiments, but multiple datasets and methods are also supported

All phasing results (sites and maps) linked to building programs and external graphics windows



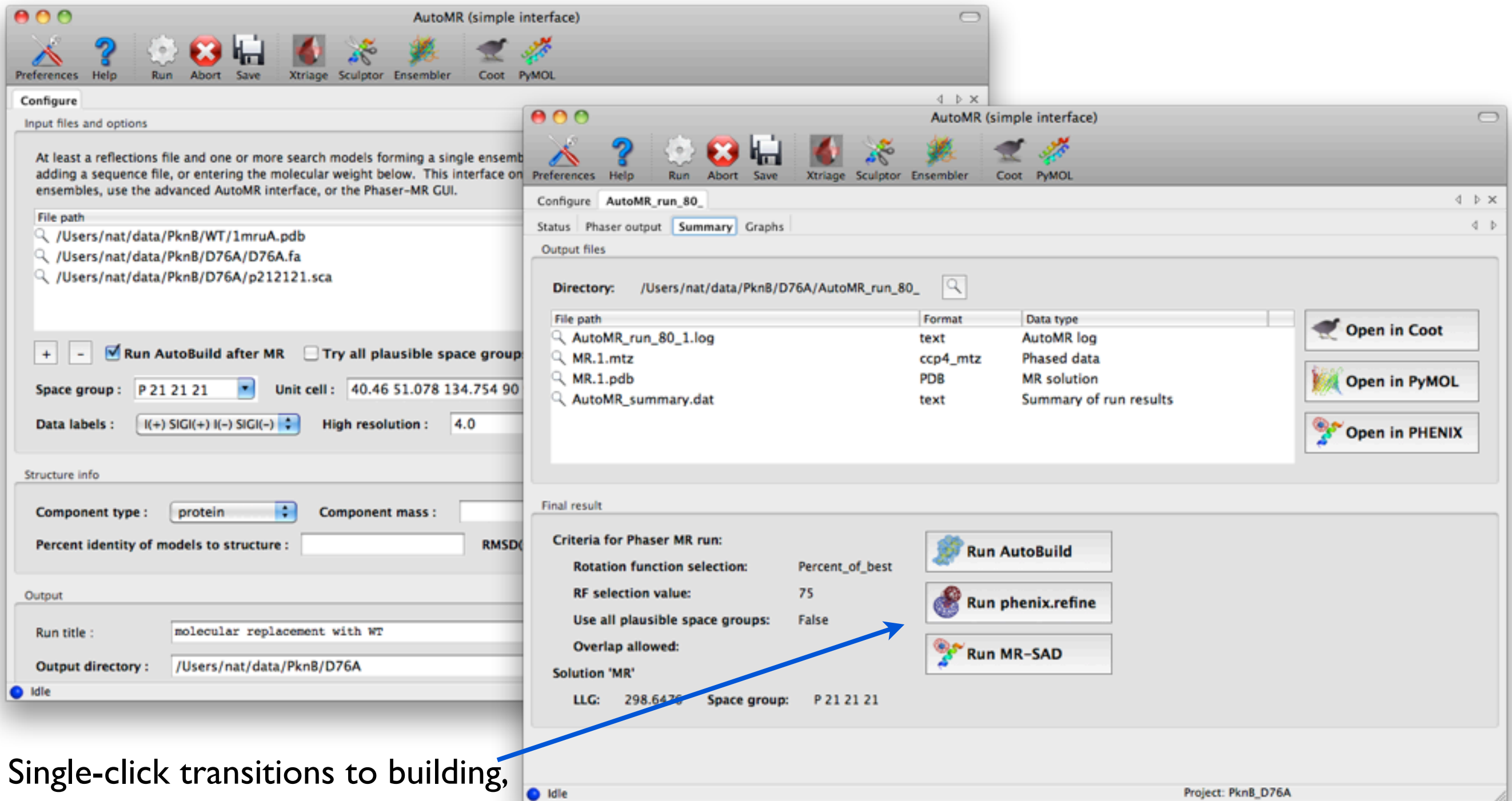
How Competitive is Automated Solution?



Tom Terwilliger, Paul Adams

AutoMR: Phaser made easy

- Streamlined setup of ensembles and composition



Single-click transitions to building, refinement, MR-SAD GUIs

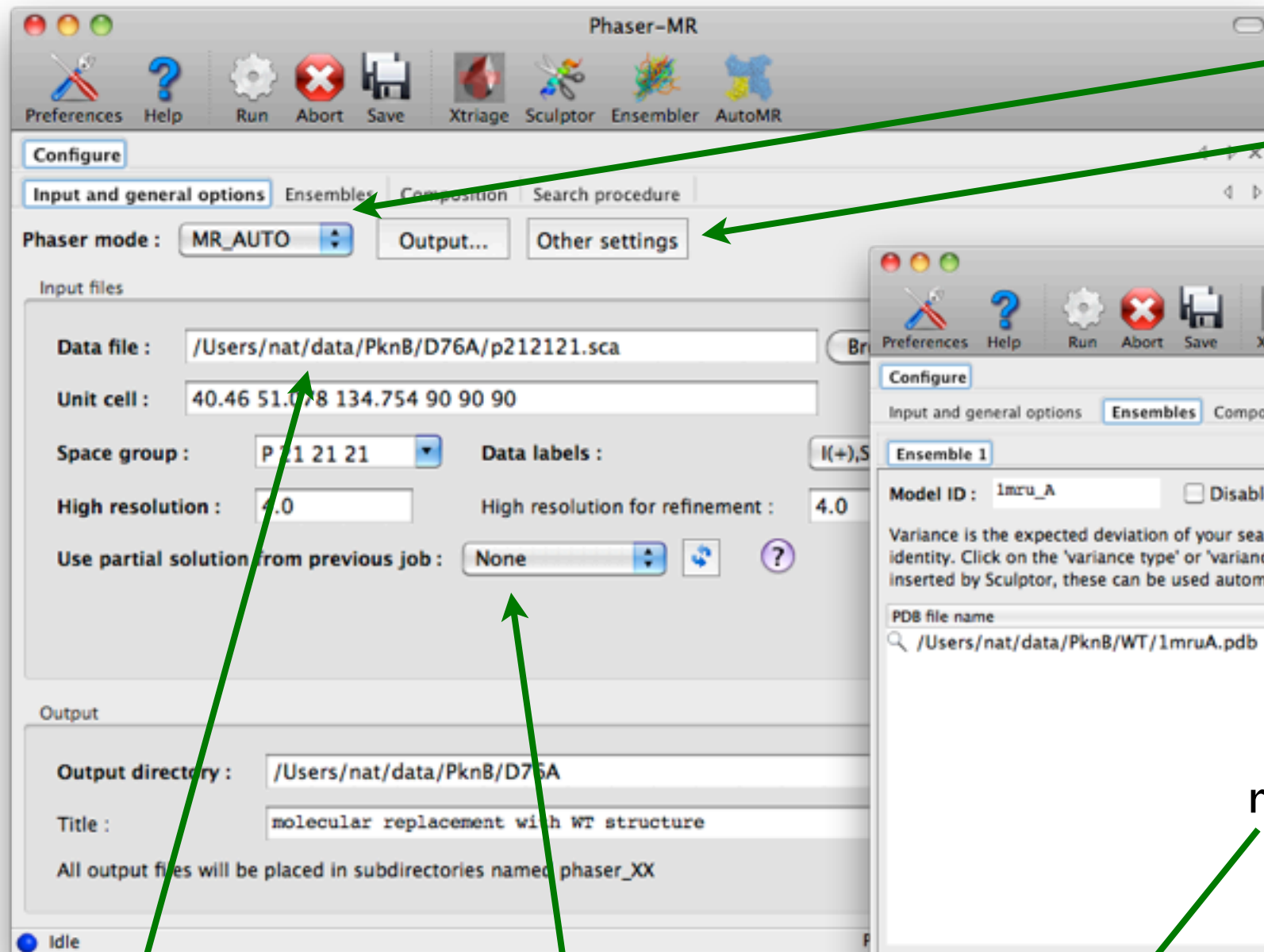
Phaser: Airlie McCoy, Gabor Bunkoczi, Rob Oeffner, Randy Read; AutoMR: Tom Terwilliger

Phaser-MR for advanced users and difficult cases

- Includes all features of command-line program

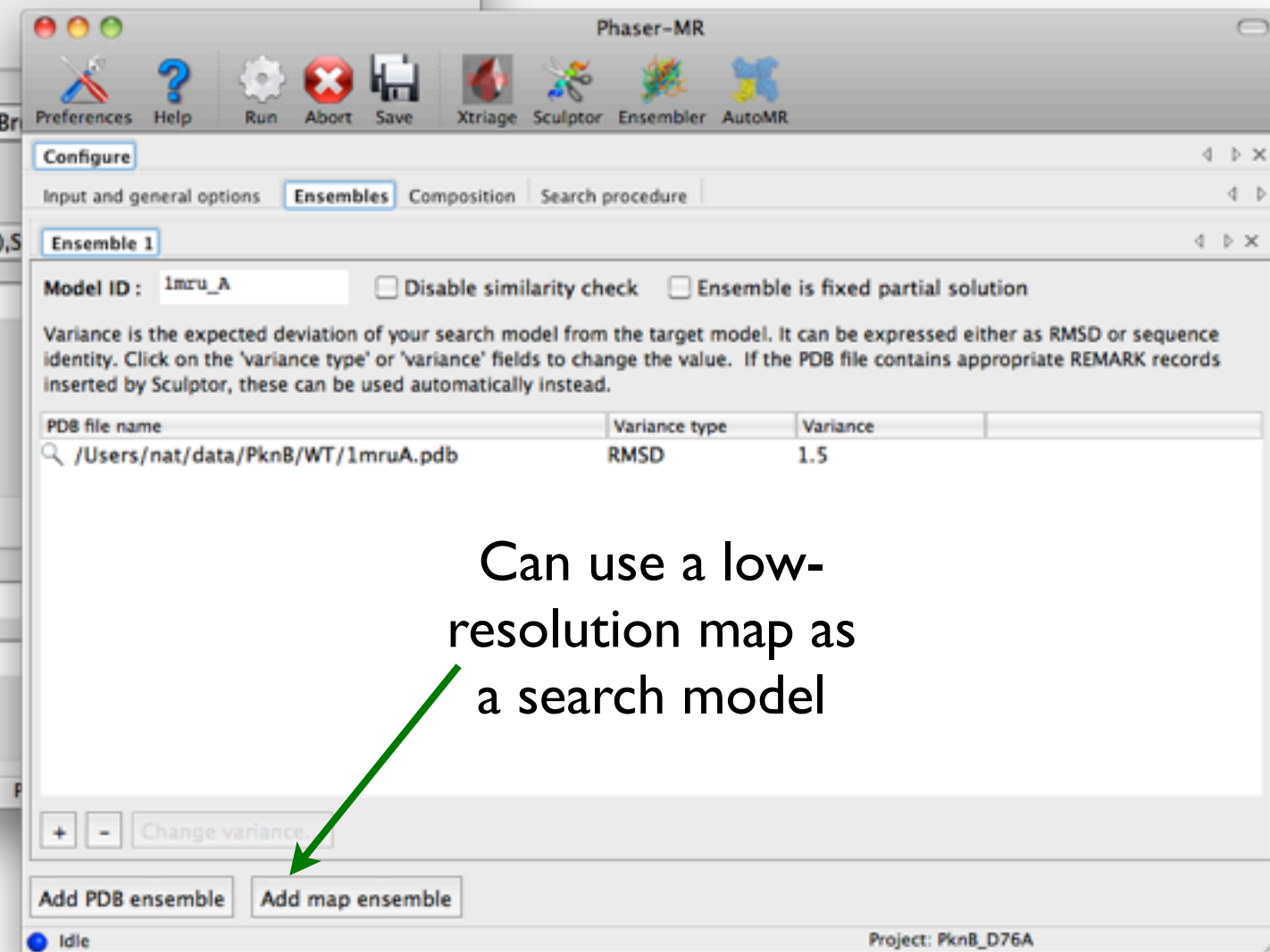
Supports all MR modes (automatic or manual)

Most keywords found here



Any reflection file format permitted

One-click re-use of partial solutions from past runs

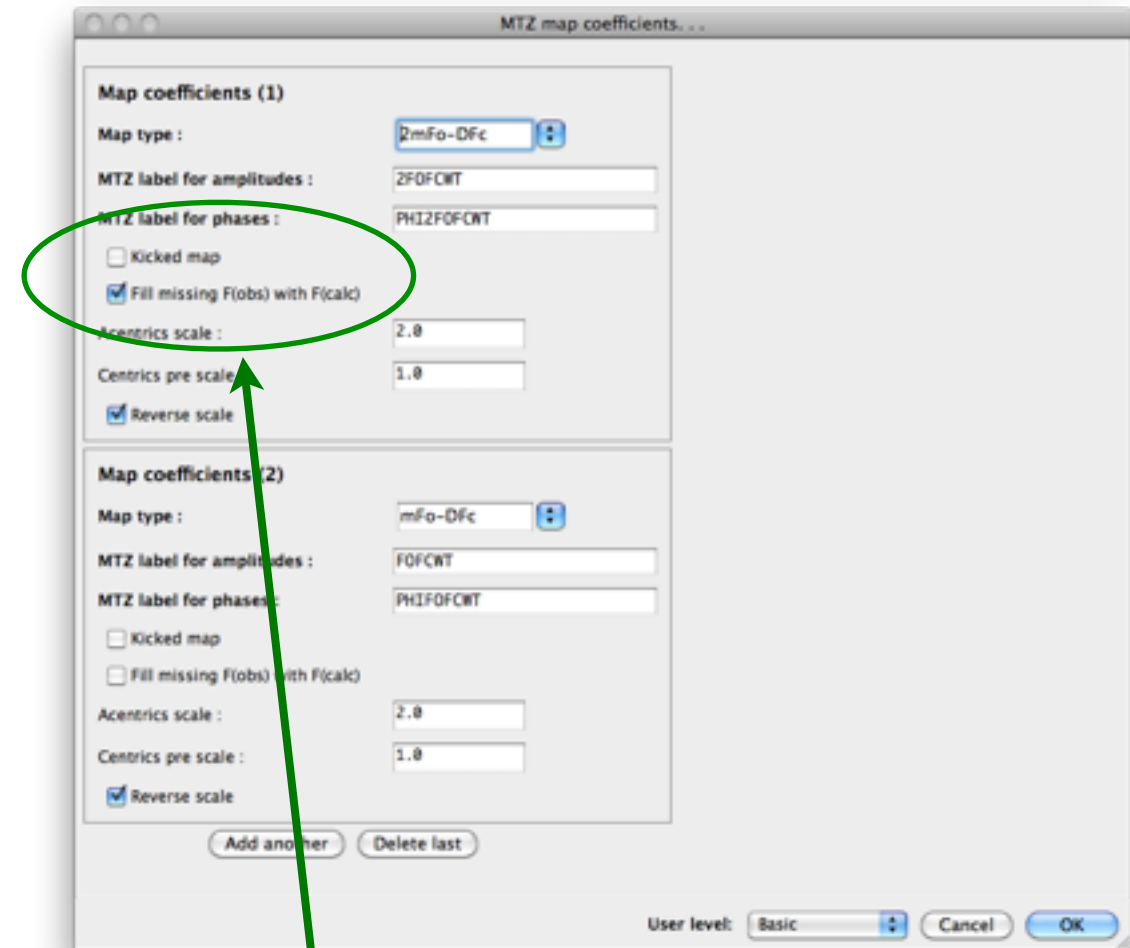
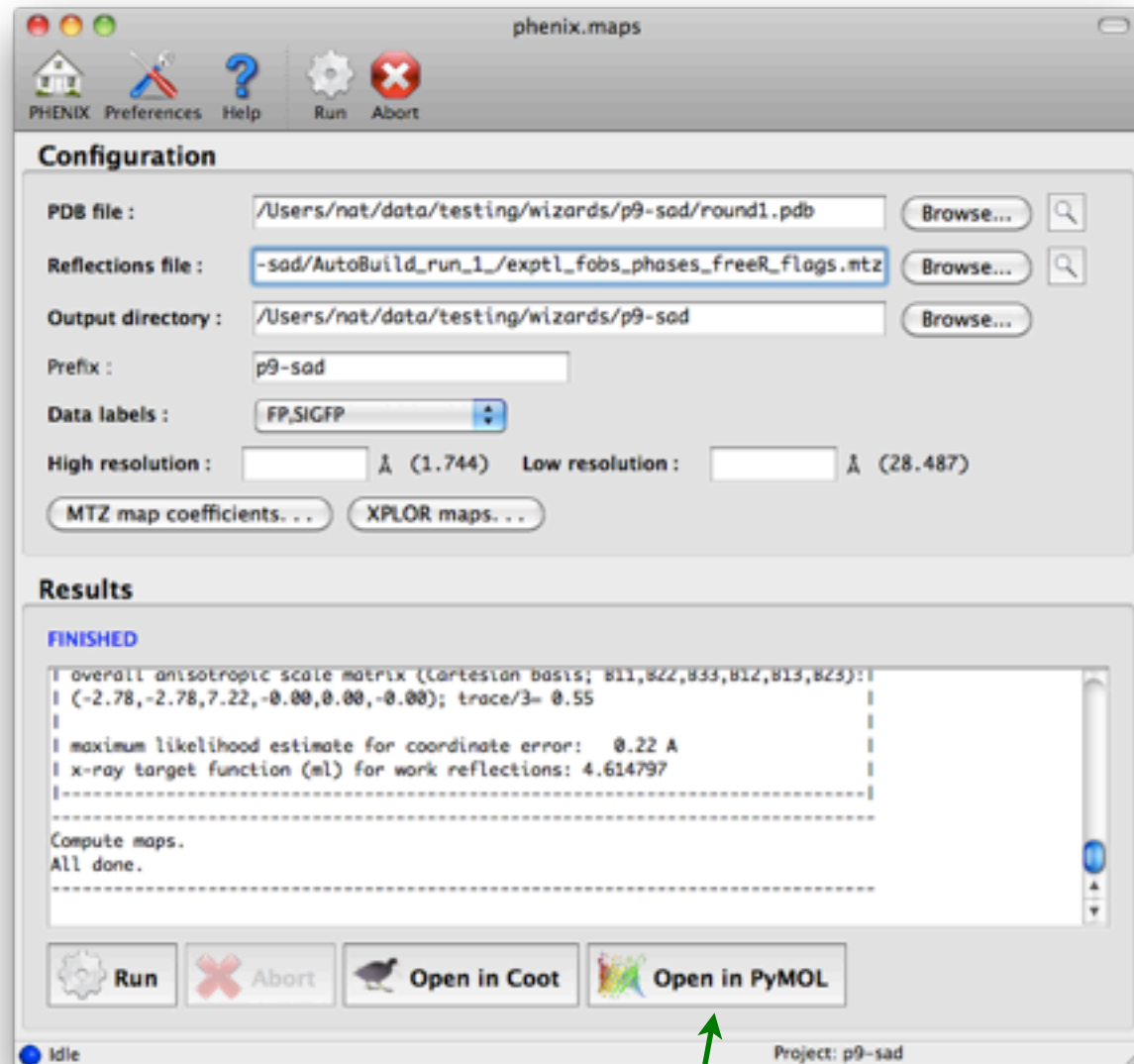


Can use a low-resolution map as a search model

Phaser: Airlie McCoy, Gabor Bunkoczi, Rob Oeffner, Randy Read

phenix.maps GUI

Very simple interface for creating simple maps (including anomalous difference maps) in MTZ or XPLOR format*



“kicked” map: removes bias by averaging maps calculated with shaken coordinates (*Praenikar et al. 2009 Acta Cryst. D65:921*)

Fill missing F(obs) with F(calc): often improves 2mFo-DFc maps, but watch out for bias! (phenix.refine and Refmac both do this)

*To save disk space, Phenix does not write XPLOR or CCP4 maps by default; however, most programs in the GUI will convert MTZ map coefficients to CCP4 format when you click the “Open in PyMOL” button.

phenix.refine: graphical extensions

- Combines with *phenix.ready_set* for adding hydrogen/deuterium and generating restraints - not fully automatic yet

The image shows a screenshot of the phenix.refine graphical user interface. The main window is titled 'phenix.refine' and has a menu bar with 'Preferences', 'Help', 'Run', 'Abort', 'Save', 'Graphics', 'ReadySet', 'NCS', 'TLS', and 'Xtrriage'. Below the menu bar is a toolbar with icons for these functions. The 'ReadySet' window is open, showing a configuration panel with the following options:

- Add hydrogens to model if absent
- Add deuteriums to solvent molecules
- Convert all possible sites to deuterium
- Optimize ligand geometry
- Remove waters from model
- Add hydrogens to solvent molecules
- Convert exchangeable sites to deuterium
- Generate ligand restraints
- Metal ion coordination restraints
- Output edits determined by LINK records

The 'Results' section at the bottom of the ReadySet window has a 'Show run info' button. A red arrow points from the 'ReadySet' window to the 'phenix.refine ready_set output files' dialog box. The dialog box contains the following text:

ReadySet is done processing the PDB files. One or more output files have been created; please choose which you wish to use for refinement. We strongly recommend that you review them first to make sure that any changes or instructions are appropriate for your structure!

- Processed PDB file**
The structure has been checked to ensure compatibility with PHENIX, e.g. consistent use of residue and atom names, and hydrogen or deuterium atoms have been added in the appropriate positions if requested.
- Metal coordination restraints**
Restrictions for the coordination geometry of metal ligands have been generated; these will be automatically added to the main phenix.refine parameter file.
- Ligand restraints file**
The CIF file contains the restraints needed to refine any unknown ligands found in your structure.
- Apply restraints to all future refinements in this project**

The dialog box has 'Cancel' and 'OK' buttons at the bottom right. A blue arrow points from the text 'Automatic re-use of parameters in subsequent refinement jobs' to the 'Apply restraints to all future refinements in this project' checkbox.

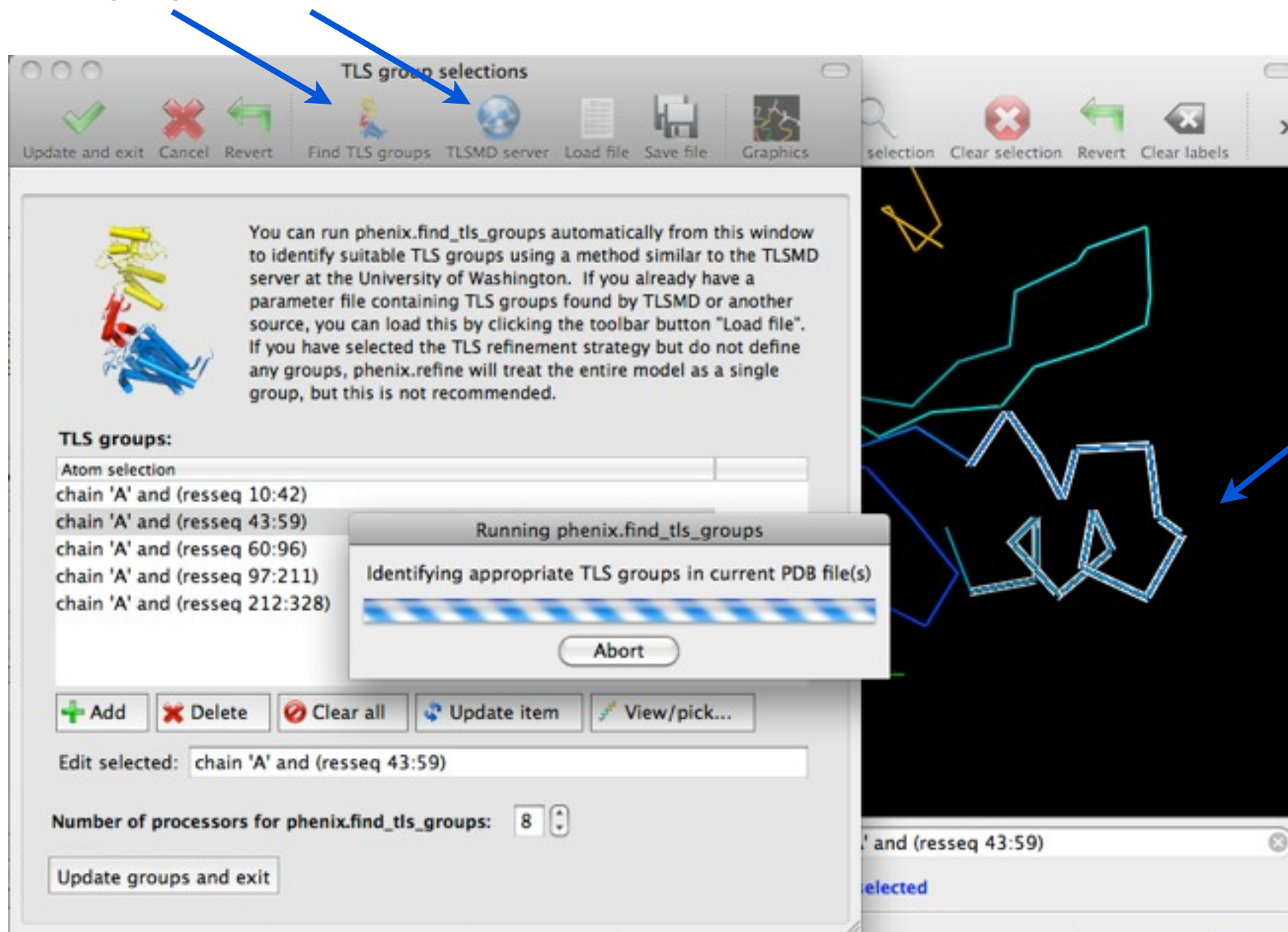
Automatic re-use of parameters
in subsequent refinement jobs

phenix.refine: Pavel Afonine et al.; *phenix.ready_set*: Nigel Moriarty

phenix.refine: graphical extensions

- *phenix.find_tls_groups*: highly parallel automatic TLS setup (similar to TLSMD), available as interactive component

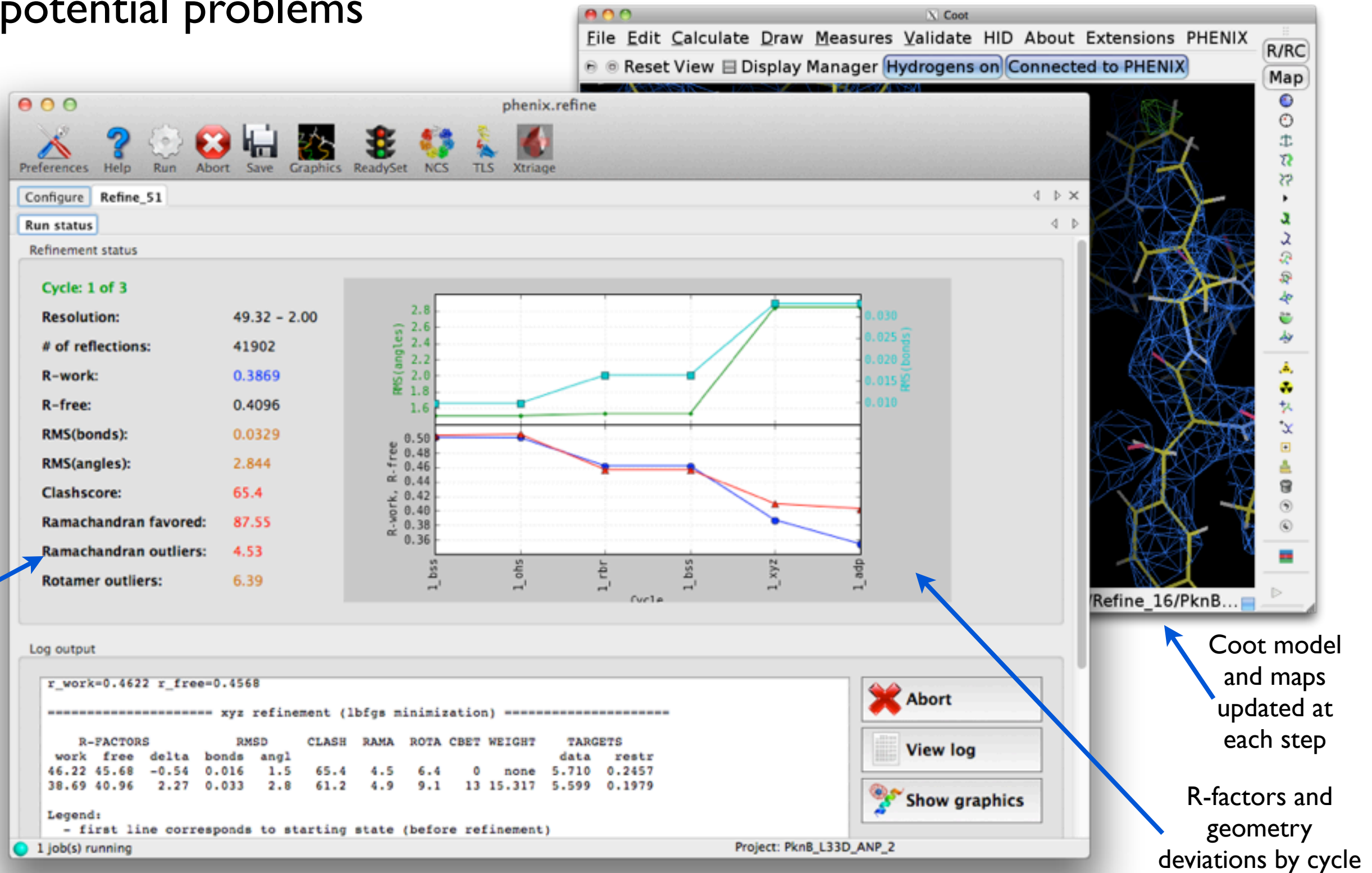
Buttons to launch *find_tls_groups* or TLSMD web server



phenix.find_tls_groups: Pavel Afonine

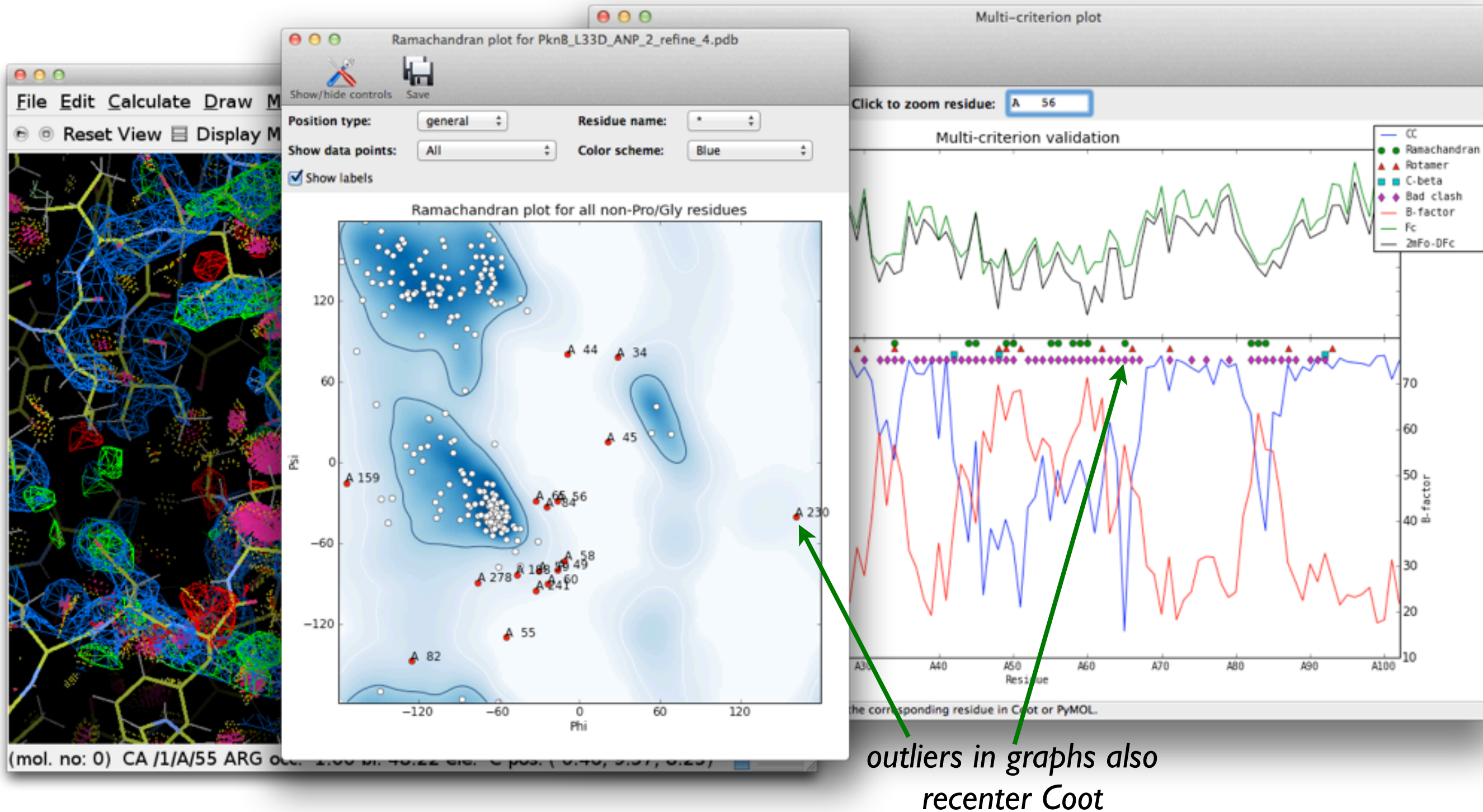
Integrating refinement and validation

- Constant feedback during refinement enables immediate detection of potential problems



Visualizing validation problems

- Outlier lists recenter Coot view; Probe dots automatically loaded



Advanced validation tools

- Combines Molprobit with phenix.model_vs_data; run automatically after phenix.refine

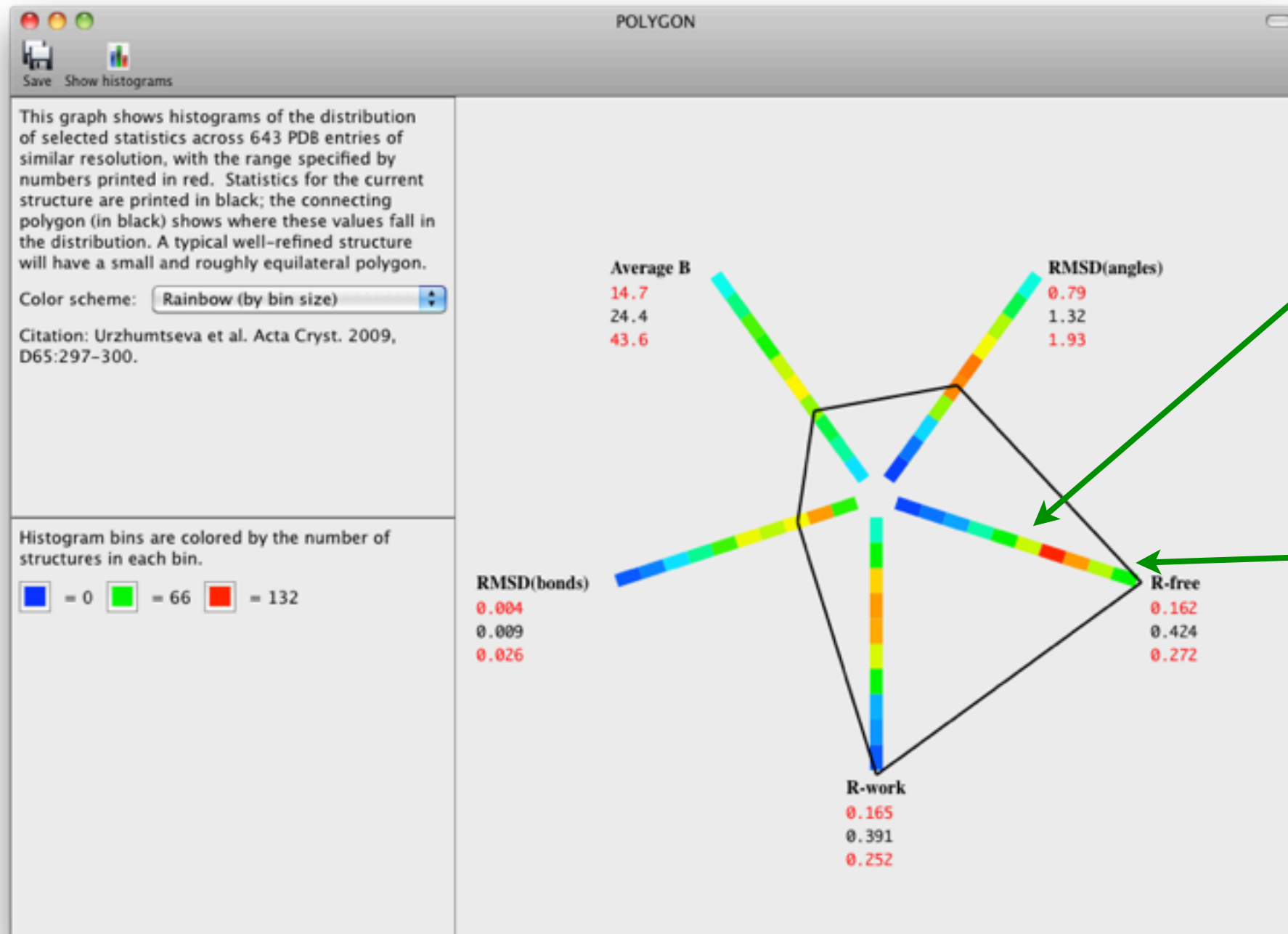
The image displays three overlapping windows from the MolProbity and KiNG software suite. The top-left window shows the MolProbity interface with the 'Clashes' tab selected, displaying 'All-atom contact analysis' and 'Bad contacts from PROBE: 45 overlapping atoms'. The middle window shows the 'Validation summary' for 'PknB_L33D_ANP_2_refine_4.pdb' with the following statistics:

Metric	Value	Goal
Ramachandran outliers	7.5%	< 0.2%
Rotamer outliers	7.3%	1%
Clashscore	85.28	

The bottom-right window shows the KiNG 2.18 interface displaying a 3D molecular model of the protein structure with various validation metrics overlaid. The 'KINEMAGE #1' panel on the right lists various validation checks, including 'mainchain', 'Calphas', 'sidechain', 'H's', 'water', 'chain A', 'Rota outliers', 'Rama outliers', 'angle dev', 'length dev', 'Cbeta dev', 'chain S', 'vdw contact', 'small overlap', 'bad overlap', 'H-bonds', 'McMc contacts', 'ScSc contacts', 'McSc contacts', 'Hets contacts', and 'dots'.

POLYGON

- Graphical comparison of statistics versus the PDB



Colored bars are one-dimensional histograms showing distribution of values for structures at similar resolution

The black polygon shows where the statistics for the user's structure fall in each histogram

The structure used to generate this figure has good geometry relative to the PDB, but very poor R-factors.

POLYGON: Ludmilla Urzhumtseva, Pavel Afonine, Sacha Urzhumtsev;
Urzhumtseva et al. (2009) Acta Cryst. D65:297-300.

Parallel validation of multiple structures

- Identifies points of difference between structures of the same protein, with optional map superpositioning

The screenshot displays the Coot software interface. On the left, a 'Parallel structure comparison' window shows a grid of rotamer data for various residues across different protein chains. The grid is color-coded: green for the most common rotamer, yellow for minority, and red for outliers. A blue arrow points from the text 'Comparison of sidechain rotamers across all chains (green = most common, yellow = minority, red = outlier)' to the grid. Below the grid, a 'Coot controls' window lists the models being compared, with checkboxes for each chain. On the right, the main Coot window shows a 3D molecular model with multiple chains superimposed in different colors (blue, green, yellow, red) to highlight differences. A status bar at the bottom right indicates 'Structure comparison' and provides a brief description: 'Identify differences between multiple structures of the same protein, using multiple criteria'.

	19 LEU	21 LYS	23 LYS	24 GLU	25 ASP	27 LEU	28 LYS	29 LYS	31 GLU
3fhi.pdb:A	---	---	OUTLIER	tt0	m-20	mt	---	mttm	---
3dnd.pdb:A	OUTLIER	OUTLIER	tttt	tt0	m-20	OUTLIER	mtmm	mttt	mm-40
1l3r.pdb:E	---	---	tttp	---	m-20	mt	---	mttt	mm-40
3fjq.pdb:E	---	tmtm?	tttt	mt-10	m-20	mt	mttt	mttt	mm-40
1syk.pdb:A	n?	mttt	tptm	tp10	t70	OUTLIER	mptt	mttt	tp10
1syk.pdb:B	n?	mttt	tptm	tp10	t70	OUTLIER	mptt	mttt	mt-10
3dne.pdb:A	OUTLIER	mptt	ttpt	mt-10	m-20	OUTLIER	mtmm	mttt	mm-40

Comparison of sidechain rotamers across all chains (green = most common, yellow = minority, red = outlier)

Double-clicking any cell in the grid zooms Coot/PyMOL

Coot controls

Load models and maps Fetch modified structures

The chains being compared, and any associated maps, have been superposed on the first chain in the list. If you want to modify any of the models after modifying them in Coot, you must click the button labeled 'Fetch modified structures' to reassemble the original PDB file with modifications in the proper orientation.

Models:

- 3fhi.pdb:chain A
- 3dnd.pdb:chain A
- 1l3r.pdb:chain E
- 3fjq.pdb:chain E
- 1syk.pdb:chain A
- 1syk.pdb:chain B
- 3dne.pdb:chain A

Check or uncheck a file name/chain ID to show or hide the model and any associated map in Coot.

Structure comparison
Identify differences between multiple structures of the same protein, using multiple criteria

(Collaboration with Herb Klei, BMS)

Works in progress and future plans

- **Improved Windows support**
- Fully automated molecular replacement
- Simplified GUI for eLBOW (ligand restraints)
- *LABELIT* GUI (indexing of diffraction images)
- You can preview new developments by checking “Enable alpha-test programs and features” in the preferences
- *Suggestions? Email NEchols@lbl.gov*

Acknowledgments

- **Lawrence Berkeley Laboratory**

- Paul Adams, Pavel Afonine, Richard Gildea, Ralf Grosse-Kunstleve, Jeff Headd, Nigel Moriarty, Nicholas Sauter, Peter Zwart

- **Los Alamos National Laboratory**

- Tom Terwilliger, Li-Wei Hung

- **Cambridge University**

- Randy Read, Airlie McCoy, Laurent Storoni, Gabor Bunkoczi, Robert Oeffner

- **Duke University**

- Jane Richardson & David Richardson, Ian Davis, Vincent Chen, Chris Williams, Bryan Arendall, Laura Murray, Gary Kapral, Swati Jain, Bradley Hintze

- **Others**

- Alexandre Urzhumtsev
- Luc Bourhis
- Herb Klei
- Garib Murshudov & Alexi Vagin
- Paul Emsley, Kevin Cowtan, Bernhard Lohkamp, William Scott, Charles Ballard
- Warren DeLano
- David Abrahams
- PHENIX Testers & Users: Brent Appleton, Joel Bard, Scott Classen, Ben Eisenbraun, James Fraser, Felix Frolow, Christine Gee, Miguel Ortiz-Lombardia, Blaine Mooers, Bob Nolte, Engin Ozkan, Daniil Prigozhin, Miles Pufall, Richard Rymer, Edward Snell, Eugene Valkov, Erik Vogan, Frank von Delft, Andre White, and many more

- **Funding:**

- NIH/NIGMS: *P01GM063210, P50GM062412, P01GM064692, R01GM071939*
- PHENIX Industrial Consortium
- Lawrence Berkeley Laboratory

- **Authors of open-source software packages:**

wxPython, matplotlib, numpy, ksDSSP, MUSCLE, PULCHRA

