PHENIX and the CCTBX

Paul Adams

Computational Crystallography Initiative &
Berkeley Center for Structural Biology,
Physical Biosciences Division,
Lawrence Berkeley National Laboratory
The *PHENIX* Project

- **Goal:** develop new crystallographic software for automated structure determination, using modern algorithms and advances in computer science.

- The new system has to be comprehensive and flexible enough to automate structure solution, but allow user interaction, input and control.

- Automated crystallography at medium to low-resolution (3Å):
  - Maximum Likelihood methods (Phasing, Molecular replacement, Density modification, Refinement)
  - Pattern matching for automated model building
  - Simulated annealing refinement

- Flexibility in design to allow the incorporation of other biophysical techniques. Single particle cryo-EM image reconstruction (NIH P01 funding), and neutron crystallography (NIH R01, Langan & Adams).

http://www.phenix-online.org

**Funding:** *NIH Program Project (NIGMS, PSI), Director - Paul Adams*
Automated Structure Solution - *PHENIX*

Data collection

Data processing

*PHENIX*

ML molecular replacement

ML density modification

ML simulated annealing

Automated model building

Validation

Heavy atom location

ML phasing

LLG maps

Deposition

PDB

The \textit{PHENIX} project

A collaboration between several groups

- **Computational Crystallography Initiative (LBNL)**
  - Paul Adams, Ralf Grosse-Kunstleve, Pavel Afonine
  - Nigel Moriarty, Nicholas Sauter, Peter Zwart

- **Los Alamos National Lab (LANL)**
  - Tom Terwilliger, Li-Wei Hung, Thiru Radhakannan

- **Cambridge University**
  - Randy Read, Airlie McCoy, Laurent Storoni
  - Hamsapriye

- **Texas A&M University**
  - Tom Ioerger, Jim Sacchettini, Kreshna Gopal, Tod Romo
  - Reetal Pai, Erik McKee

- **Duke University**
  - Jane Richardson, David Richardson, Ian Davis
PHENIX – Design Considerations

• Interpreted language interface
  – Fully featured (i.e. not csh)
  – Direct access to crystallographic algorithms
  – Transparent syntax (i.e. not PERL)
  – Informed by experience developing CNS
  – c.f. Mathematica, matlab, web programming with PERL, visual basic, etc.

• Different interfaces:
  – Higher-level GUI(s)
  – Command-line

• Long term maintainability
• Integration of different components
• Flexibility (extension to other methods)
Choice of Implementation Language(s)

- Maintainability
- Reusability
- Modularity

Python
- Very high-level programming
- Easy to use (dynamic typing)
- Fast development cycle (no compilation)
- Too slow for certain tasks

C++
- High-level or medium-level programming
- Many arcane details (strong static typing)
- Largely automatic dynamic memory management (templates)
- Much faster than Python
- With enough attention, performance even rivals that of FORTRAN
• Built around the Python scripting language: efficient development of new ideas


• Different user interfaces are required for novice users, expert users, structural genomics centers, and real-time data analysis.
Performance

- Computer Science wisdom:
  - Typically 90% of the time is spent in 10% of the code
- Similar to idea behind vector computers:
  - Python = Scalar Unit
  - C++ = Vector Unit
- Loading the vector unit, with Python

```python
miller_indices = flex.miller_index()
for h in xrange(100):
    for k in xrange(100):
        for l in xrange(100):
            miller_indices.append((h,k,l))
```
- Calculating epsilons, with C++

```python
space_group = sgtbx.space_group_info("P 41 21 2").group()
epsilons = space_group.epsilon(miller_indices)
```

Computing 1 million epsilons takes only 0.65 seconds
The Computational Crystallography Toolbox

• Is an evolving C++ library of fundamental algorithms for computational crystallography (and more)

• Contains a variety of tools:
  – General crystallographic (cctbx, iotbx)
  – Macromolecular crystallographic (mmtbx)
  – General scientific (scitbx)
  – Installation (libtbx)

• Crystallography general – not specific to macromolecules

• Open source project at Sourceforge (http://cctbx.sourceforge.net)
  – ensures continued availability
  – rapid development
  – easy contribution by all developers

def to_mtz():
    from iotbx import reflection_file_reader
    import os

    reflection_file = reflection_file_reader.any_reflection_file(
        file_name=os.path.expandvars("$CNS_SOLVE/doc/html/tutorial/data/pen/scale.hkl"))

    from cctbx import crystal
    crystal_symmetry = crystal.symmetry(unit_cell=(97.37, 46.64, 65.47, 90, 115.4, 90),
                                        space_group_symbol="C2")

    miller_arrays = reflection_file.as_miller_arrays(
                        crystal_symmetry=crystal_symmetry)

    from iotbx import mtz

    mtz_dataset = None
    for miller_array in miller_arrays:
        if (mtz_dataset is None):
            mtz_dataset = miller_array.as_mtz_dataset(
                        column_root_label=miller_array.info().labels[0])
        else:
            mtz_dataset.add_miller_array(
                        miller_array=miller_array,
                        column_root_label=miller_array.info().labels[0])

    mtz_object = mtz_dataset.mtz_object()
    mtz_object.show_summary()
    mtz_object.write("pen_data.mtz")
to_mtz output

Title: /home/programs/cns_solve_1.1/doc/html/tutorial/data/pen/scale.hkl:F_PH

Space group symbol from file: C2

Number of crystals: 2
Number of Miller indices: 6735
Resolution range: 22.0198 2.79124

History:
Crystal 1:

Crystal 2:
  Name: crystal
  Project: project
  Id: 1
  Unit cell: (97.37, 46.64, 65.47, 90, 115.4, 90)
Number of datasets: 1
Dataset 1:
  Name: dataset
  Id: 1
  Wavelength: 1
  Number of columns: 9
  Column number, label, number of valid values, type:
  1 H  6735/6735=100.00% H: index h,k,l
  2 K  6735/6735=100.00% H: index h,k,l
  3 L  6735/6735=100.00% H: index h,k,l
  4 F_PHGA  6735/6735=100.00% F: amplitude
  5 SIGF_PHGA 6735/6735=100.00% Q: standard deviation
  6 F_KUOF  6735/6735=100.00% F: amplitude
  7 SIGF_KUOF 6735/6735=100.00% Q: standard deviation
  8 F_NAT  6735/6735=100.00% F: amplitude
  9 SIGF_NAT 6735/6735=100.00% Q: standard deviation
## CCI web services

### Computational Crystallography Toolbox

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explore symmetry</td>
<td>Obtain detailed information about a space group</td>
</tr>
<tr>
<td>Alternative space group settings</td>
<td>Browse alternative space group settings for a given space group</td>
</tr>
<tr>
<td>Structure factor calculation</td>
<td>Calculate structure factors from a set of atoms</td>
</tr>
<tr>
<td>SHELX LATT and SYMM cards</td>
<td>Generate SHELX LATT and SYMM cards for a given space group</td>
</tr>
<tr>
<td>Wyckoff positions</td>
<td>Assignment of Wyckoff positions to atoms in a given unit cell</td>
</tr>
<tr>
<td>Fractionalization and orthogonalization of coordinates</td>
<td>Convert coordinates from/to Fractional or Orthogonal coordinates</td>
</tr>
<tr>
<td>Change space group setting</td>
<td>Convert coordinates from one space group setting to another</td>
</tr>
<tr>
<td>Change the hand of a crystal structure</td>
<td>Change the hand of a set of coordinates (useful in heavy atom location)</td>
</tr>
<tr>
<td>Euclidean Model Matching (EMMA)</td>
<td>Superimpose two sets of coordinates taking into account different origin choices and polar axes (useful for comparing potential heavy atom solutions)</td>
</tr>
<tr>
<td>LABLEUT</td>
<td>Autoindex a pair of macromolecular diffraction images</td>
</tr>
</tbody>
</table>
Leveraging Third-party Tools

- Minimize the duplication of effort
- Maximize functionality with limited resources

- Core components:
  - Python (freely available, open source scripting language)
  - Boost.Python (part of the open source Boost C++ project)

- Graphical User Interface:
  - wxPython (all GUI development at the Python level, no C/C++)
  - PyMOL (making use of an open source visualization tool)

- Data handling for external formats:
  - CCP4 libraries for MTZ (use the code written by the CCP4 developers)

- Build tools:
  - Scons (Python based replacement for make)

- Code management:
  - CVS (widely used code versioning system) – transition to SVN soon
Command-line tools

Provide highly automated, easy-to-use tools

- **Data statistics:**
  - `iotbx.reflection_statistics <reflections>` …
  - `mmtbx.xtriage <reflections>` …

- **Heavy atom location:**
  - `phenix.hyss <reflections> <#-sites> <element>`

- **Site comparison:**
  - `iotbx.emma <coordinates> <coordinates>`

- **Refinement:**
  - `phenix.refine <reflections> <pdb-file(s)> <lib-file(s)>`
Structure Refinement Protocol

Bulk-solvent & Anisotropic scaling
- Parameters for maximum-likelihood target
- Ordered solvent (water) modeling
- Target weights calculation
- Rigid body refinement
- TLS refinement
- Simulated annealing refinement
- Coordinate & ADP refinement

% phenix.refine model.pdb data.hkl
% phenix.refine model.pdb data.hkl simulated_annealing=True
% phenix.refine model.pdb data.hkl parameters_file

Neutrons: joint refinement against X-ray and neutron data implemented

--- Paul Adams  Physical Biosciences Division ---
Refinement with Neutron Data

• Neutron

\%
phenix.refine model.pdb data_n.hkl remove_hydrogens=false
scattering_table=neutron

→ Currently phenix.refine does not build hydrogens; they should be built externally and they should comply with Monomer Library.

• Joint (X-ray + neutron)

\%
phenix.refine model.pdb data_x.hkl remove_hydrogens=false
neutron_data.file_name=data_n.hkl

→ No need to have “equivalent” data sets (resolution, reflection-to-reflection correspondence)
Test:

- start PDB model (1GKT, endothiapepsin complexed with transition state analogue inhibitor h261)
- shake model (1.5 Å mean error)
- use all available data (no sigma or resolution cutoffs)

<table>
<thead>
<tr>
<th>Refinement mode</th>
<th>( R_{\text{WORK}} )</th>
<th>( R_{\text{FREE}} )</th>
<th>Phase error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xray start:</td>
<td>55.5</td>
<td>55.4</td>
<td>78.8</td>
</tr>
<tr>
<td>final:</td>
<td>27.0</td>
<td>30.9</td>
<td>31.5</td>
</tr>
<tr>
<td>Neutron start:</td>
<td>49.1</td>
<td>48.9</td>
<td>76.5</td>
</tr>
<tr>
<td>final:</td>
<td>25.5</td>
<td>33.9</td>
<td>38.5</td>
</tr>
<tr>
<td>Joint final (xray):</td>
<td>22.1</td>
<td>25.9</td>
<td>26.1</td>
</tr>
<tr>
<td>final (neutron):</td>
<td>25.2</td>
<td>29.3</td>
<td>30.9</td>
</tr>
</tbody>
</table>
Automation with Strategies

- The path through the network is determined by the outcome of each task (a very natural approach)
- Strategies are constructed and modified graphically (visual programming)
- Development is made possible for non-programmers
- This general approach facilitates the integration of a wide range of tools and techniques
PHENIX Wizards

- Guides the user through a complex procedure
- Makes decisions based on current information
- Prompt for user input when required
- AutoSol – hkl to initial model
  - combines HySS, SOLVE and RESOLVE
- AutoMR – automated maximum likelihood molecular replacement with PHASER
- AutoBuild – iterative model building with RESOLVE and phenix.refine
- LigandFit – automated ligand fitting using RESOLVE

T. Terwilliger, Los Alamos National Laboratory
Nigel Moriarty, Lawrence Berkeley Laboratory
Laurent Storoni, Cambridge University
PHENIX Availability

- New beta release in July 2006
- Supported on:
  - Linux (developed on Redhat, Fedora)
  - In the near future:
    - Mac OSX, Windows
    - Tru64 Alpha
    - Irix SGI

- Freely available to academic (non-profit) groups
- Web download (http://www.phenix-online.org/)
- Consortium for commercial users

- The cctbx is open source and available to all
  - Provides C++ and Python interfaces to crystallographic algorithms
  - cctbx.sourceforge.net
Acknowledgments

• Computational Crystallography Initiative
  – Pavel Afonine (phenix.refine)
  – Ralf Grosse-Kunstleve (cctbx, phenix.refine, HySS, …)
  – Nigel Moriarty (PHENIX GUI, elbow)
  – Nick Sauter (iotbx, labelit)
  – Peter Zwart (mmtbx.xtriage)

• Los Alamos National Laboratory
  – Tom Terwilliger
  – Li-Wei Hung

• Cambridge University
  – Randy Read
  – Airlie McCoy
  – Laurent Storoni

• Texas A&M University
  – Tom Ioerger
  – Jim Sacchettini
  – Erik McKee

• Others
  – CCP4 developers (MTZ library)
  – Alexei Vagin & Garib Murshudov (Monomer Library)
  – Kevin Cowtan, York, (Reciprocal space asymmetric units and CLIPPER)
  – David Abrahams (principal Boost.Python author)

• Funding:
  – LBNL (DE-AC03-76SF00098)
  – NIH/NIGMS (P01GM063210)
  – NIH/NIGMS (P01GM064692)
  – PHENIX Industrial Consortium