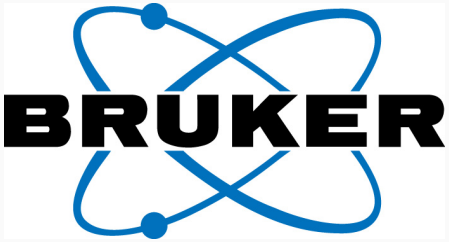


CCTBX: AN INTRODUCTION

Not quite for dummies

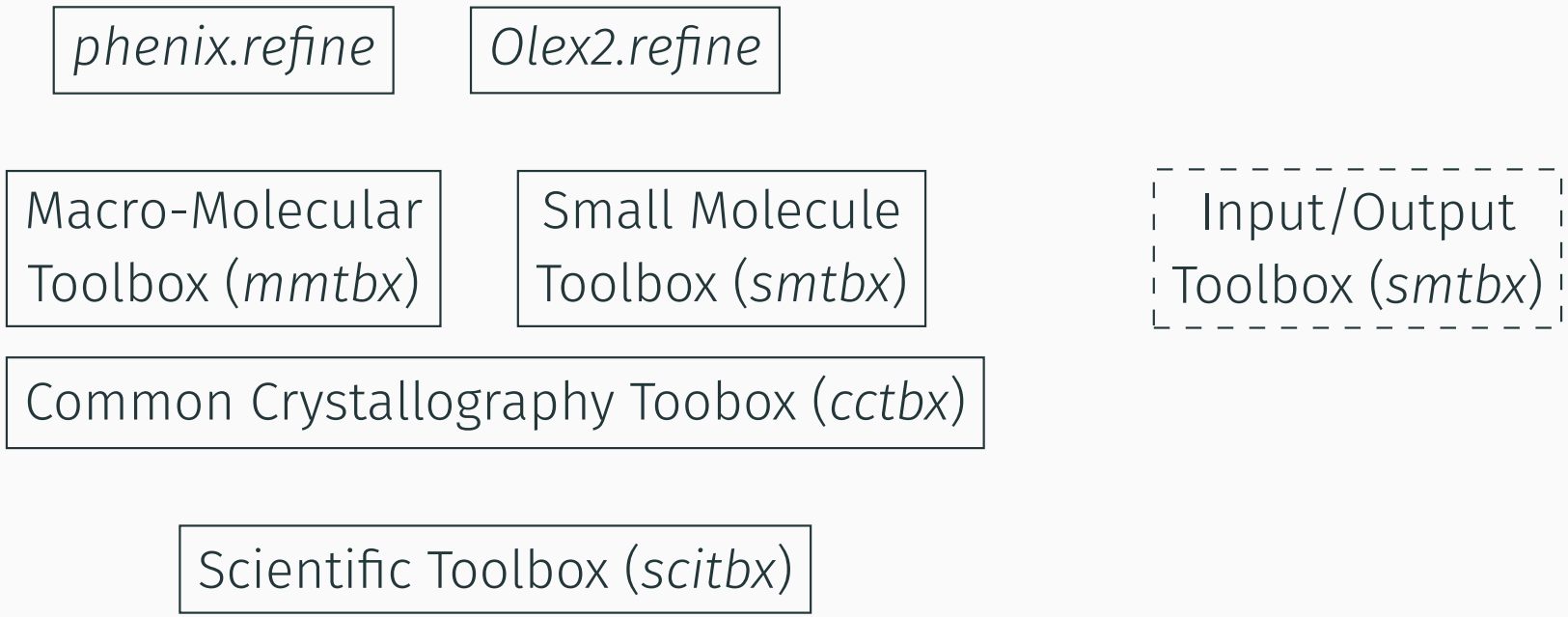
Luc J. Bourhis

August 21, 2015

Durham University, UK and Bruker AXS

THE BIG PICTURE

Modular design to minimise dependencies

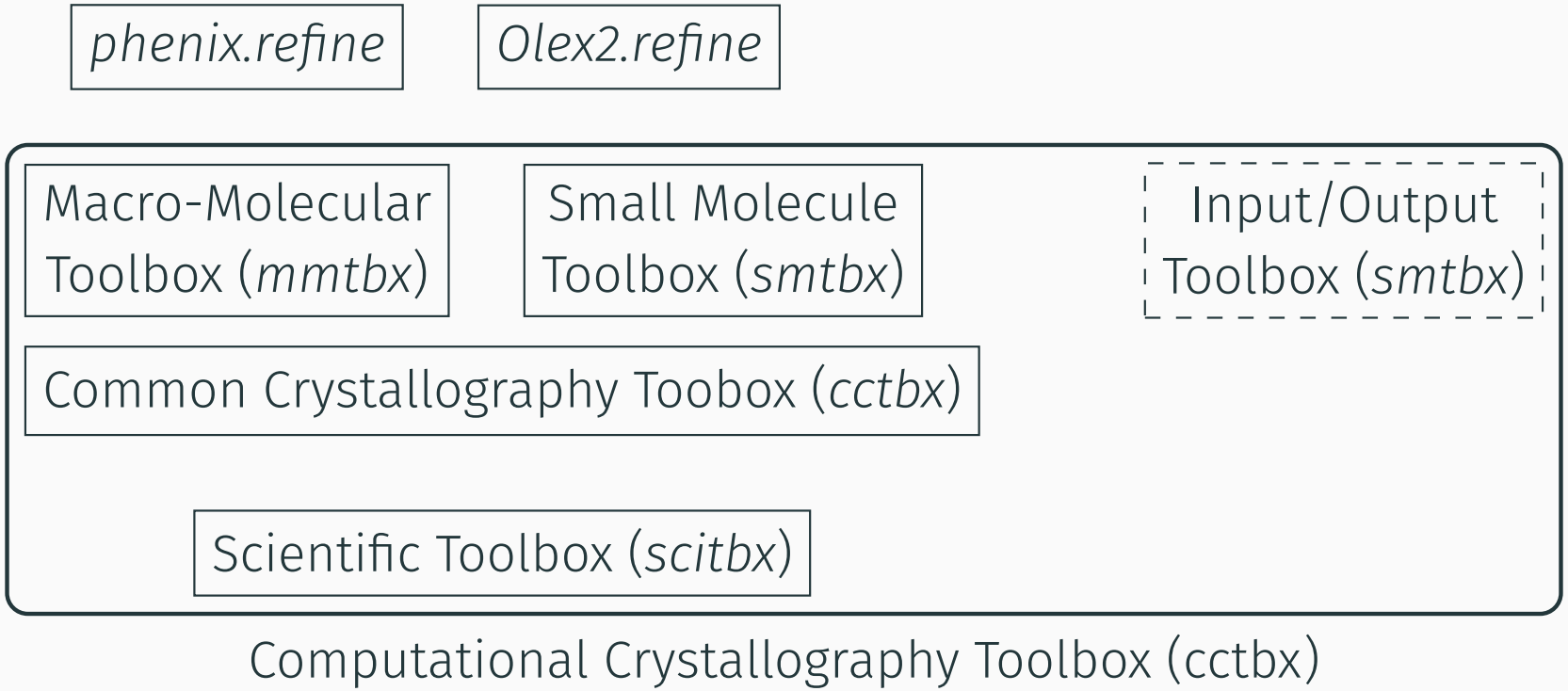


In Python:

```
from cctbx import sgtbx
```

THE BIG PICTURE

Modular design to minimise dependencies



In Python:

```
from cctbx import sgtbx
```

DOWNLOADS

- This tutorial: slides, code and data
  `https://github.com/luc-j-bourhis/rovinj-2015-cctbx-tutorial`
- Source and binary packages:
  `http://cci.lbl.gov/cctbx_build/`
  Not very up-to-date anymore but it will do for this tutorial!
  - Linux: choose CentOS 6.2 (64-bit) even if you don't wear a red hat
    `source cctbx_build/setpaths.sh`
  - Windows 7 (64-bit) or Windows XP (32-bit)
    `cctbx_build/setpaths.bat`
  - `cctbx.python my_script.py`
- Source code repository (version control):
  - Currently hosted on Sourceforge
  - Plans to move to Github soon-ish

MODULES

From the foundations to the roofs:

- Python $\leftrightarrow$ C++ bridge
  - Python is slow but convenient
  - C++ is fast but cumbersome
  - Boost.Python is a bridge between them and boost_adaptbx rules it
- scitbx: mathematics, non-crystallographic physics (rigid body)
  - n-dimensional arrays, bridge with NumPy
  - linear algebra, special functions, non-linear least-squares
- cctbx: common crystallography
  - structure factors computation, and their derivatives
  - space group toolbox
  - Fourier transforms
- smtbx: constraints
  - structure factors computation and derivatives

MODULES

Based on cctbx:

- rstbx: indexing and integration of diffraction images
- xfel: X-ray free-electron lasers
- crys3d, gltbx: tools to write GUI displaying 3D objects (structures, Fourier maps)
  - $\longrightarrow$ Phenix

Matrices in pure Python:

```
>>> from scitbx.matrix import rec, col
>>> A = rec(elems=(-1,0,0, 0,-1,0, 0,0,-1), n=(3,3))
>>> b = col((3,4,5))
>>> C = A * b
>>> print C
matrix.rec(elems=(-3, -4, -5), n=(3,1))
>>> abs(C)
7.0710678118654755
```

Matrix written by row: $\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$ Use only for few operations on small matrices.

Particularly useful in crystallography

```
>>> from scitbx.matrix import col, row, rt
>>> rt(((-1,0,0, 0,-1,0, 0,0,-1),
       (7,8,9)))*col((3,4,5))
matrix.rec(elems=(4, 4, 4), n=(3,1))
>>> dihedral_angle([(-1,-1,1), (0,0,0),
                    (1,0,0), (1,1,1)], deg=True)
-90.0
```

Used a lot.

AND NOW, LADIES AND GENTLEMENT, THE MAIN FEATURE!

Let's triple the unit cell of a structure.

Why, oh, why?

Phase transition: unit cell before the transition $(a, b, c, 90, 90, 90)$, and after $(3a, b, 3c, 90, 90, 90)$.

To compare the two structures, it is useful to put the first one in a tripled unit cell, keeping the same symmetries.