DIALS: command line usage

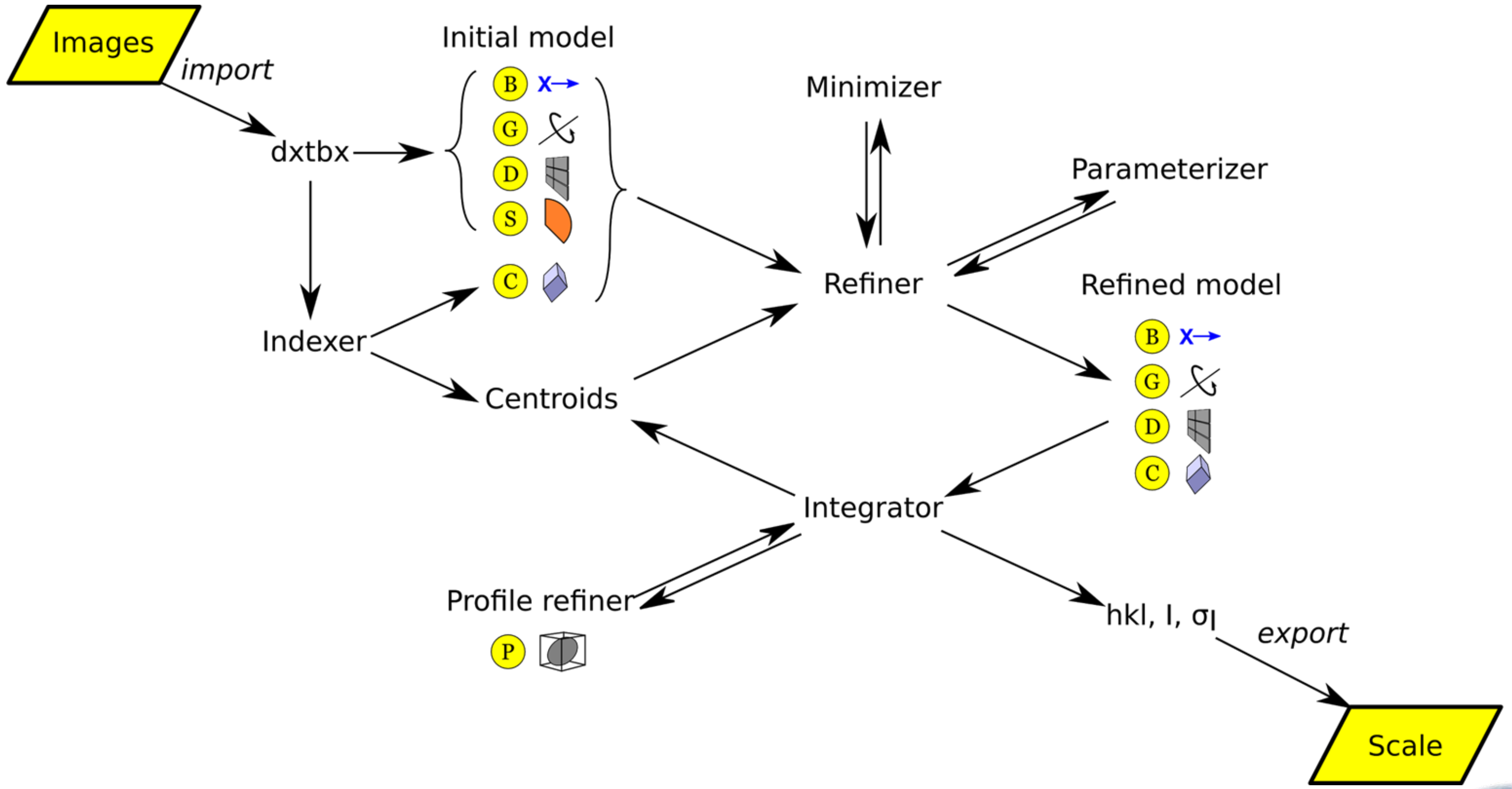
James Parkhurst

ECM DIALS workshop August 2015

DIALS principles

- **Modular construction**: allows more than one algorithm for a given task
- **Toolkit**: use the tools as you like, not only one way
- **Extensible**: it is relatively simple to add new methods & new instruments
- **Correct**: model the experiment as closely as possible, including full detector corrections

DIALS overview



Main DIALS programs

- dials.import
- **dials.find_spots**
- **dials.index**
- dials.refine_bravais_settings
- **dials.refine**
- **dials.integrate**
- dials.export_mtz
- (POINTLESS/AIMLESS)

DIALS on the command line

```
$ dials.import ${data_directory}/th_8_2_0*.cbf
$ dials.find_spots datablock.json nproc=8
$ dials.index datablock.json strong.pickle
$ dials.refine_bravais_settings experiments.json indexed.pickle
$ dials.reindex indexed.pickle change_of_basis_op=a,b,c
$ dials.refine bravais_setting_9.json reindexed_reflections.pickle \
  outlier.algorithm=tukey use_all_reflections=true \
  scan_varying=true output.experiments=refined_experiments.json
$ dials.integrate refined_experiments.json refined.pickle \
  outlier.algorithm=null nproc=4
$ dials.export_mtz integrated.pickle refined_experiments.json
   hklout=integrated.mtz
$ pointless hklin integrated.mtz hklout sorted.mtz > pointless.log
$ aimless hklin sorted.mtz hklout scaled.mtz > aimless.log << eof
  resolution 1.3
  anomalous off
eof
$ ctruncate -hklin scaled.mtz -hklout truncated.mtz \
  -colin '/*/*/[IMEAN,SIGIMEAN]' > ctruncate.log
```

Who needs a GUI?

General usage

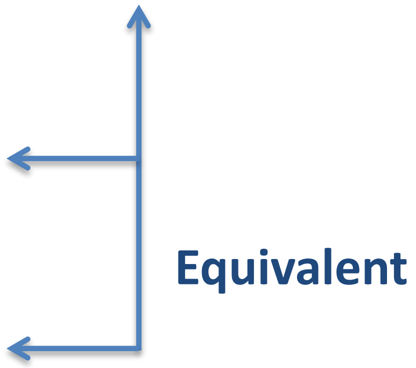
All input uses phil parameters

```
$ dials.integrate experiments.json indexed.pickle

$ dials.integrate \
  input.experiments=experiments.json \
  input.reflections=indexed.pickle

$ dials.integrate params.phil

$ cat params.phil
input.experiments=experiments.json
input.reflections=indexed.pickle
```



N.B. only datablock/experiment JSON files, reflection table PICKLE files and PHIL parameter files can be specified as positional arguments

Help

To view some help

```
$ dials.integrate –h
Usage: dials.integrate [options] experiment.json

Options:
  -h, --help            show this help message and exit
  -c, --show-config     Show the configuration parameters.
  -a ATTRIBUTES_LEVEL, --attributes-level=ATTRIBUTES_LEVEL
                        Set the attributes level for showing
configuration
                        parameters
  -e EXPERT_LEVEL, --expert-level=EXPERT_LEVEL
                        Set the expert level for showing configuration
                        parameters
  -v                    Set the verbosity
```

Configuration

To view some configuration parameters

```
$ dials.integrate –c
Showing configuration parameters with:
  attributes_level = 0
  expert_level = 0

integration {
  background {
    algorithm = *simple null
  }
  intensity {
    algorithm = sum *fitrs
  }
}
input {
  experiments = None
  reflections = None
}

...
```

Any option can be set on the command line or within a *.phil file by passing the file on the command line.

More detailed output can be produced by setting the expert and attribute levels (–e and –a respectively).

For brevity, only a subset of the available parameters are shown.

dials.import

Import the image data to use within DIALS. Analyse the metadata for each image to determine relationships between sets of images (e.g. sweeps and stills). Write a datablock JSON file containing experimental geometry.

Input	**Output**
Path to images files	datablock.json

Examples

```
$ dials.import /path/to/images_*.cbf
$ dials.import template=/path/to/images_####.cbf
```

dials.find_spots

Find strong pixels on a sequence of images. Combine pixels into spots using 3D connected component labelling. Write a strong.pickle file containing the list of found spots.

Input	Output
datablock.json	strong.pickle

Examples

```
$ dials.find_spots /path/to/images_*.cbf
$ dials.find_spots datablock.json
```

dials.index

Perform autoindexing on the strong spots in the input pickle file. Output an experiment list JSON file containing the experimental geometry and crystal model. Also output an indexed.pickle file containing the indexed strong spots.

Input	**Output**
datablock.json strong.pickle	experiments.json indexed.pickle

Examples

```
$ dials.index datablock.json strong.pickle
$ dials.index datablock.json strong.pickle \
   unit_cell=37,79,79,90,90,90 \
   space_group=P43212
```

dials.refine

Refine diffraction geometry of input experiments against indexed reflections. For rotation scans the model may be either static or scan varying. Output files containing refined experiments and spots.

Input	**Output**
experiments.json indexed.pickle	refined_experiments.json refined.pickle

Examples

```
$ dials.refine experiments.json indexed.pickle
$ dials.refine experiments.json indexed.pickle \
   scan_varying=True
```

dials.integrate

Predict the positions of reflections on the diffraction images and integrate them. The input list of strong indexed spots is used to compute the bounding box parameters for each reflection. Output a pickle file with predicted reflections and their intensities and sigmas

Input	**Output**
refined_experiments.json refined.pickle	integrated.pickle

Examples

```
$ dials.integrate refined_experiments.json \
   refined.pickle
```

dials.export_mtz

Export the results of dials processing as an unmerged MTZ file, reading for input into downstream programs such as pointless and aimless.

Input	**Output**
refined_experiments.json integrated.pickle	hklout.mtz

Examples

```
$ dials.export_mtz refined_experiments.json \
   integrated.pickle \
   hklout=hklout.mtz
```

Summary

- Command line is the main DIALS user interface
- Output from previous program is input into the next
- Configuration is done using cctbx PHIL parameters on the command line or via an input “*.phil” file

Thanks!

DIALS
Diffraction Integration for Advanced Light Sources