Vukica Srajer July 23 2002

# LaueView basics

## Table of contents

1. Setting up LaueView 3

2. Starting up 4
- 2.1 LaueView main menu 4
- 2.2 Basic LaueView input/output files 5
- 2.3 Read and display an image, create crystal file and default file 7
  - 2.3.1 Read an image 7
  - 2.3.2 Display an image 7
  - 2.3.3 Save the default file for this image 8
  - 2.3.4 Exit LaueView 8
- 2.4 Read and display an image when you already have a default file 8
- 2.5 Create a crystal information file 8
- 2.6 Read an existing crystal information file 9

3. Interactive indexing of an image and geometry refinement 10
- 3.1 Preparing the default file for indexing 10
- 3.2 Beam position refinement 11
  - 3.2.1 Pick beam position 11
  - 3.2.2 Pick spots for ellipses 11
  - 3.2.3 Refine ellipses 12
  - 3.2.4 Refine ellipses and beam position 13
- 3.3 Indexing 14
- 3.4 Geometry refinement 16

4. Processing a Laue data set 19
- 4.1 Set-up the default file of the indexed/refined image for data set processing: change/setup filenames and directories (important) 19
- 4.2 Create and modify default files for all images 20
  - default.mtf 20
  - default_modify.mtf 21
- 4.3 Geometry refinement 21
  - 4.3.1 Find spots for geometry refinement 23
    - findspot.mtf 23
  - 4.3.2 Geometry refinement 24
    - refine.mtf 24
    - refinedef.mtf 25
    - refine_prog.mtf 25
- 4.4 Creating a set file (important) 28
  - set.mtf 28
- 4.5 Profile fitting 28

| | |
|---|---|
| spoverlap_rdb.mtf | 29 |
| selectsam.mtf | 29 |
| sampling.mtf | 31 |
| rejectsam.mtf | 31 |
| 4.6 Integration | 32 |
| integration_rdb.mtf | 32 |
| rejectsht_bg.mtf | 33 |
| rejectsht_1x1.mtf | 33 |
| 4.7 Scaling | 34 |
| scale.mtf | 35 |
| Check wavelength normalization curve | 37 |
| rejectfct.mtf | 38 |
| 4.8 Merging | 38 |
| sht2fct1.mtf | 38 |
| sht2fct2.mtf | 39 |
| apply.mtf | 39 |
| 4.6 Deconvolution of energy overlaps | 40 |
| deconvolution.mtf | 40 |
| 4.7 Finalizing | 41 |
| final.mtf | 41 |
| rejecthkl.mtf | 42 |
| Check data completeness | 42 |
| Appendix 1: Rejecting outliers: reject*.mtf scripts | 44 |

## 1. Setting up LaueView

- Runs only on SGIs
- Add to the .cshrc file:

```
# set LaueView envirment
setenv CRYSTALNAME crystal_name           # sets up the crystal info filename
                                            (file crystal_name.xtl has to exist)
setenv CRYSTALINFO ~/xtal_info            # sets up dir path for the crystal info file
setenv WORK /home/vukica/data_set_1       # sets up working directory (where the
                                            images are)
setenv LAUEVIEWHOME /usr/local/LaueView   # sets up dir path to LaueView program
setenv SCREENX 1100                       # sets up image display size
setenv SCREENY 1100
```

- Turn off “Auto windows placement” on your SGI (in Desktop/Customize/Windows).

- Known limitations:
    - o Only 100 images/data set can be scaled together.
    - o Filenames have to have less than 10 characters.

- Make sure that calls to LaueView program are consistent in all C-shell script files (*.mtf files). You should either specify LaueView (were LaueView is aliased or linked to a particular version of the program) or specify a particular version (such as LaueView2.5b, LaueView3.1b…) directly.

- Usage flowcharts: flowchart.byRen.ps, flowchart.byYang.ps (modified by VSarajer)

- Other manuals: manual.byRavelli,ps, hard copy of Adachi LaueView manual

- References:

1. Srajer V., Crosson S., Schmidt M., Key J., Schotte F., Anderson S., Perman B., Ren, Z., Teng T-.Y., Bourgeois D., Wulff M., and Moffat K. *Extraction of Accurate Structure Factor Amplitudes from Laue Data: Wavelength Normalization with Wiggler and Undulator X-Ray Sources*. **J. Synchrotron Rad. 7: 236-244 (2000).**
2. Ren Z., Bourgeois D., Helliwell J.R., Moffat K., Srajer V., and Stoddard B.L. *Laue Crystallography: Coming of Age*. **J. Synchrotron Rad. 6: 891-917 (1999).**
3. Yang X., Ren Z., and Moffat K. *Structure Refinement Against Synchrotron Laue Data: Strategies for Data Collection and Reduction*. **Acta Cryst. D 54:367-77 (1998).**
4. Ren Z., Ng K., Borgstahl G.E.O., Getzoff E.D., and Moffat K. *Quantitative Analysis of Time-Resolved Laue Diffraction Patterns*. **J. Appl. Cryst. 29:246-260 (1996).**
5. Ren Z., and Moffat K. *Deconvolution of Energy Overlaps in Laue Diffraction*. **J. Appl. Cryst. 28:482-493 (1995).**
6. Ren Z., and Moffat K. *Quantitative Analysis of Synchrotron Laue Diffraction Patterns in Macromolecular Crystallography*. **J. Appl. Cryst. 28:461-481 (1995).**

## 2. Starting up

### 2.1 LaueView main menu

| | |
|---|---|
| F(ile) | *# read/write files* |
| V(iew) | *# display image* |
| P(ick) | *# pick spots: beam center, ellipse spots for beam center refinement, nodals for indexing* |
| E(llipse) | *# refine ellipses and beam center* |
| I(ndex) | *# indexing and geometry refinement* |
| (i)N(tegration) | *# integration – rarely used interactively* |
| S(cale) | *# scaling – rarely used interactively* |
| L(aueSim) | *# simulation – use for checking completeness of the data* |
| T(ime-resolved) | *# rarely used* |
| Q(uit) | *# quit (quit or stop can by type anywhere in the program to exit)* |

- To choose an item from a menu it is enough to type one letter (the one that is not in parenthesis).
- Menu names are typically not case sensitive (do not have to be upper case).
- <CR> is typically defined as the most often used item on the menu (as indicated)
- All main menu items have several submenus. LaueView:menu:submenu:subsubmenu is displayed on the line where you enter your choice.
- Not all submenus are unique – sometimes several paths from different main menu items point to the same submenu.

  Example: Index/Crystal and LaueSim/Xtal lead to the same submenu. Or: default file can be saved from several places like F(ile)/Def(ault) or I(ndex)/R(efine)/D(efault).
- Pay attention to units when entering values for parameters: not always same units for the same parameter.

  Example: Index/Refine/Manual/Distance: distance should be in mm. Typing distance anywhere in the program (low case !) will require distance in pixels.
- Several parameters can be set anywhere within the program:

| | |
|---|---|
| o fil(e) | *# to setup filenames* |
| o dir(ectory) | *# to setup directory names* |
| o wav(elength) | *# to setup wavelength range for the X-ray spectrum* |
| o res(olution) | *# to setup crystal resolution range* |
| o dis(tance) | *# to setup crystal-to-detector distance* |
| o rev(iew) | *# to refresh the image display* |
| o su(perimpose) | *# to superimpose observed and predicted diffraction pattern after indexing (this actually doesn't work from the Index/Refine/Goniometer menu)* |

These commands, however, have to be typed in low case!!!

## 2.2 Basic LaueView input/output files

- Crystal information file:
  - Input
  - Contains cell parameters and space group info (crystal_name.xtl). Cell parameters and spece group have to be known for Laue data processing,
  - Can be created using LaueView (see below).

- Default files:
  - Output
  - A default file filename.def contains all information that has been entered up to the point of saving the default file. This allows one to resume with processing of an image after exiting LaueView without repeating all previous steps. Save this file often to prevent repeating steps in the initial stages of interactive indexing and refining of the first image!!! This initial default file is later copied to all other default files and they are used in processing of a data set.

- X-ray spectrum file (xrayspectrum.lam):
  - An X-ray spectrum (actually the wavelength normalization curve) is actually derived from the data, during the scaling process. However, an initial X-ray spectrum is needed as a starting point. If there is no information from the beamline where experiment was done on the shape of the spectrum, at least limits $\lambda_{min}$ and $\lambda_{max}$ should be known. In that case create an xrayspectrum.lam file as:

    $\lambda_{min}$ 0

    $\lambda_{min}$+0.01 1

    $\lambda_{max}$-0.01 1

    $\lambda_{max}$ 0

    (box function, $\lambda_{min}$ and $\lambda_{max}$ are in Å)

    If shape of the spectrum is roughly known, create an ASCII file with two columns of numbers: wavelength (Å) and relative intensity, based on he known spectrum.

- C shell script files *.mtf:
  - Needed for batch processing a Laue data set, after an interactive indexing and geometry refinement of one image.
  - Contain LaueView commands what would have been typed in an interactive processing mode.
  - Sample files are provided with the program.
  - The header of each file describes what the file does, what are input and output files and how to run the file.
  - Typically the top part of the script has to be edited (as indicated) to be customized for a particular Laue data set.
  - Make sure all modified mtf script files in your directory are executable. If not, you might be running the wrong ones from the LaueView directory.
  - Make sure the correct crystal is specified in all script files. Or you can use properly set global CRYSTALNAME variable from your .cshrc.

  - One mtf file that is used but not called directly is scalesigma.mtf. It is called by final.mtf.
  - Flow charts of the usage of these script files to process a Laue data set are also distributed with the program (flowchart.byRen.ps, flowchart.byYang.ps).

- An example of the processed data is also distributed with the program.

## 2.3 Read and display an image, create crystal file and default file

### 2.3.1 Read an image

```
LaueView
dir(ectory)            # to enter image directory/path
0                      # enter 0 to type a dir name
1                      # select dir 1 (directory where the images are)
dir_name               # enter directory name
fil(e)                 # to enter image filename
0                      # enter 0 to type a filename
1                      # select filename 1 (image fileneme)
filename               # enter filename
F(ile)                 # go to File menu
E(xternal image)
1(6-bit
R(ecords               # enter image size in number of Xpixels
W(ords/record          # enter image size in number of Y pixels
P(ixel size            # enter x,y pixel size (in mm)
Q(uit)
R(ead)                 # read the file
Y(es)
```

- fil(e) and (F(ile) or f(ile)) are two different commands. “fil(e)” and “dir(ectory)” (low case) can be typed anywhere in the program and allow you to set up file and directory names (see below: creating default file). “F(ile)” or “f(ile)” is the first entry of the main LaueView menu that is used to read and write input/output files.

### 2.3.2 Display an image

Continuing from 2.3.1:

```
V(iew)                 # go to View menu
V(iew)                 # view the image
F(ull)                 # make sure the full image mode is on
```

- Use View/Modify to change the highest and lowest level (H(ighest level), L(owest level)), color mode (C(olor mode)), invert colors (N(egative))…
- Use View/Window to zoom in:

```
V(iew)
W(indow)
O(pen window)
                       # left click in the window; a small red box will appear;
```

| | |
|---|---|
| | *move it with the mouse to position the upper left corner where you want it; left click, hold and move mouse to adjust the size of the box (new window) as desired; click the middle mouse button to exit* |
| V(iew) | |

- Toggle View/Full to ON to restore the full image size:

| | |
|---|---|
| V(iew) | |
| F(ull) | *# toggle Full from OFF (was OFF after zooming in) to ON* |
| V(iew) | |
| Q(uit) | *# go to Main menu* |

### 2.3.3 Save the default file for this image

| | |
|---|---|
| F(ile) | *# go to File menu* |
| D(efault) | |
| W(rite) | |
| Y(es) | |

### 2.3.4 Exit LaueView

Q(uit) or S(top) from anywhere
Y(es) or N(o) to save or not default file, R to resume

## 2.4 Read and display an image when you already have a default file

| | |
|---|---|
| LaueView filename | *# load default file filename.def(do not type def extension)* |
| F(ile) | |
| E(xternal image) | |
| R(ead) | |
| Y(es) | |

## 2.5 Create a crystal information file

| | |
|---|---|
| LaueView | |
| I(ndex) | *# go to Index menu* |
| C(rystal) | |
| S(pace group) | |
| space_group_number | *# enter space group number* |
| (cell) P(arameters) | |
| a,b,c, A, B, G | *# enter cell parameters (can also enter them one by one using a, b, c, ALPHA, BETA, GAMMA in this menu)* |

```
N(ame)
N(ame)
crystal_name              # enter the name for the crystal (crystal file will be crystal_name.xtl)
S(ave)
Q(uit)
Q(uit)                    # go to Main menu
```

## 2.6 Read an existing crystal information file

```
LaueView
I(ndex)                   # go to Index menu
C(rystal)
N(ame)
N(ame)
crystal_name              # enter the name for the crystal (crystal file will be crystal_name.xtl)
L(oad)
Q(uit)
Q(uit)                    # go to Main menu
```

## 3. Interactive indexing of an image and geometry refinement

### 3.1 Preparing the default file for indexing

- Follow steps in 2.3 to create a default file for the image that will be indexed (we will refer to it as first image although it does not have to be the first image of the data set).
- Load crystal information from the crystal_name.xtl file, set φ angle, crystal-to-detector distance and initial estimates for wavelength range and resolution:

```
LaueView filename          # load default file filename.def
I(ndex)                    # go to Index menu
C(rystal)                  # load the crystal info file (if information is not already in the def file)
N(ame)
N(ame)
crystal_name
L(oad)
Q(uit)
G(oniometer)               # setup/change goniometer phi, chi, omega angles
P(hi)                      # change phi angle for the first image to match the nominal value (or
                           leave at 0 for this first image and set phi angles for all other images
                           relative to 0 )
phi_angle
Q(uit)
R(efine)                   # go to Index/Refine menu to set distance
M(anual)
D(istance)                 # set distance (in mm)
distance (in mm)
Q(uit)
wav(elength)               # set wavelength range
w_min w_max
res(olution)               # set resolution range
r_min r_max
Q(uit)
D(efault file)             # save the default file!!!
W(rite)
Y(es)
Y(es)
```

- Most of these parameters can be also entered from LaueSim menu but using this menu may change image display parameters. If you use the LaueSim menu, make sure the View/Full image is ON before you save the default file and exit.
- Wavelength range ($\lambda_{min}$, $\lambda_{max}$) comes from the X-ray spectrum information from the beamline where the experiment was conducted. It should be the same as used to create the initial X-ray spectrum xrayspectrum.lam.

- Resolution range is set based on your knowledge of your crystal diffraction at this time.
- After indexing of the image, one can adjust the wavelength range and resolution range if needed, by inspecting the match between predictions and observations.
- Best values for the wavelength and resolution limits (so-called soft limits) will eventually come from data processing (wavelength normalization curve for wavelength limits and $I/\sigma_I$ vs resolution plot for the resolution limit). It is highly advisable to repeat the processing with these new values.
- Best and fastest way to process a Laue data set is to initially process 3-4 images only through geometry refinement, integration and scaling. This will already provide a decent wavelength normalization curve, a good estimate for crystal resolution, as well as a good estimate of other data processing parameters. Use these parameters to process the entire data sets (takes much longer).

## 3.2 Beam position refinement

- The X-ray beam position in Laue data processing has to be determined precisely before indexing. Laue patterns provide an easy way to determine and refine the beam position as it is located at the intersection of all the ellipses in the pattern.
- Procedure of refining beam position involves:
  1. picking beam position by visual inspection of ellipses;
  2. picking spots for several intersecting ellipses (3-4) that define well the beam position (not all tangential to each other!);
  3. picking ellipses and refining them against picked spots (they will be pinned to the initial beam position);
  4. refining both ellipses and the beam position.

### 3.2.1 Pick beam position

```
LaueView filename          # load default file filename.def
F(ile)                     # read the image
E(xternal image)
R(ead)
Y(es)
V(iew)                     # display the image
V(iew)
Q(uit)
P(ick)                     # go to Pick menu
C(enter)                   # pick beam position (follow instructions on the screen)
Q(uit)                     # go to Main menu
```

### 3.2.2 Pick spots for ellipses

Continue from 3.2.1:

| | |
|---|---|
| P(ick) | *# go to Pick menu* |
| S(tore in a slot) | *# pick and store* |
| P(ick) | *# pick spots for the first ellipse* |
| 1 | *# enter the slot number (1) where picked spots for the first ellipse will be stored (1-10); this opens up a small window to show magnified region where the cursor is, left click to drop the window on the screen; left click on the image, hold and move cursor to a spot on the ellipse you selected – you will also see the spot magnified in the small window; release the left button to pick that spot; repeat for 10-15 spots evenly distributed on the ellipse; avoid saturated spots; click on the middle mouse button when done* |
| P(ick) | *# pick spots for the second ellipse* |
| 2 | *# slot 2 for the second ellipse; pick spots as described above* |
| P(ick) | *# pick spots for the third ellipse* |
| 3 | *# slot 3 for the third ellipse; pick spots as described above* |
| Q(uit) | |
| Q(uit) | *# go to Main menu* |

- To guide you in picking spots for an ellipse you can pick an ellipse first:

| | |
|---|---|
| P(ick) | |
| E(llipse) | *left click and hold – this will display faint, dotted blue ellipse pinned at the beam position; while holding the left button move the ellipse center and make the ellipse larger or smaller to best match one of the observed ellipses (by visual inspection); release the left button to pick the ellipse; ellipse in now displayed as solid blue line; click on the middle button to exit; continue with picking spots along the marked ellipse (as described above)* |

### 3.2.3 Refine ellipses

Continue from 3.2.2:

| | |
|---|---|
| E(llipse) | *# go to Ellipse menu* |
| A(ngles only) | *# refine ellipses only, pinned to the visually chosen beam position* |
| # | *# to enter ellipse slot* |
| 1 | *# enter ellipse slot* |
| E(llipse) | *# to pick the ellipse on the screen;* |

| | |
|---|---|
| | *picking ellipse on the screen initializes the ellipse parameters – center, orientation, axes lengths (parameters are stored in the same slot as the spots for the ellipse);* *left click and hold – this will display faint, dotted blue ellipse pinned at the beam position;* *while holding the left button move the ellipse center and make the ellipse larger or smaller to best fit over the observed ellipse spots (by visual inspection);* *release the left button to pick the ellipse; ellipse in now displayed as solid blue line;* *click on the middle button to exit* |
| # | |
| 2 | |
| E(llipse) | |
| | *# pick the ellipse for slot 2 as described above* |
| # | |
| 3 | |
| E(llipse) | |
| | *# pick the ellipse for slot 3 as described above* |
| R(efine) | *# refine ellipses (more than 3 ellipses can be picked and refined refined but 3 are typically used for beam position refinement);* *refinement changes the initial ellipse parameters (set by picking the ellipses on the screen) to fit the observed spots;* *while refining, ellipses are shown as blinking dotted blue lines, after refinement as solid lines;* *refined ellipse parameters are stored in the same slot where the ellipse spots are* |
| Q(uit) | |

## 3.2.4 Refine ellipses and beam position

Continue from 3.2.3:

| | |
|---|---|
| C(enter) | *# refine ellipses and beam position (in the Ellipse menu)* |
| # | *# to enter ellipse slot* |
| 1 | *# enter ellipse slot for the first ellipse* |
| S(lot) | *# get the ellipse parameters from the slot* |
| # | |
| 2 | *# enter ellipse slot for the second ellipse* |
| S(lot) | *# get the ellipse parameters from the slot* |
| # | |
| 3 | *# enter ellipse slot for the third ellipse* |
| S(lot) | *# get the ellipse parameters from the slot* |
| R(efine) | *# refine all 3 ellipses and the beam position* |
| Y(es) | *# update beam position* |
| Q(uit) | |
| Q(uit) | *# go to main menu* |

- And do not forget to save the default file!!!

```
F(ile)                          # go to File menu
D(efault)
W(rite)
Y(es)
Y(es)
Q(uit)
Q(uit)                          # go to Main menu
```

- You can save the ellipses as you may need them later for indexing (see 3.3). In fact, all slots where you stored spots (for ellipses or other purpose) can be saved in a filename.spt file. To create a filename.spt file:

```
F(ile)
S(pot)
W(rite)
Y/N)
Q(uit)
```

### 3.3 Indexing

- To index an image one can use nodals or ellipses. If using ellipses one can use those that were picked and refined for beam position refinement or one can pick (and refine) and use other ellipses.
- Typical problems with indexing involve wrong beam position, distance or detector pixel size.
- To index an image:

```
LaueView filename               # load default file filename.def
F(ile)                          # read the image
E(xternal image)
R(ead)
Y(es)
V(iew)                          # display the image
V(iew)
Q(uit)
P(ick)                          # go to Pick menu
S(tore in slot)
P(ick)                          # pick nodals
5                               # store nodals in slot 5 (slots 1-10 can be used);
                                pick nodals same way as you picked ellipse spots;
                                3-4 nodals are enough, more can crash the program;
                                pick prominent nodals distributed across the image if possible and as
                                remote from the center as possible
```

| | |
|---|---|
| Q(uit) | |
| Q(uit) | *# go to the Main menu* |
| I(ndex) | *# go to Index menu* |
| M(ethod) | *# choose indexing method* |
| N(odal) | *# select nodal (or ellipse)* |
| Q(uit) | |
| I(ndex) | *# index* |
| *5* | *# enter slot where nodals are stored (or several slot numbers where refined ellipses are stored and quit to start indexing); during indexing a number of matches are found and the best one is picked by the program* |
| su(perimpose) | *# check indexing: color coded predictions (red – long wavelength, blue – short wavelength) will be superimposed on the observed pattern* |
| Q(uit) | *# go to Main menu* |

- And do not forget to save the default file!!!

| | |
|---|---|
| F(ile) | *# go to File menu* |
| D(efault) | |
| W(rite) | |
| Y(es) | |
| Y(es) | |
| Q(uit) | |
| Q(uit) | *# go to Main menu* |

- Indexing is right if the observed pattern is matched, although could be somewhat rotated.
- If not indexed right, re-index using different set of nodals or try various combinations of ellipses (make sure the ellipses are refined - same way as described in 3.2.2 and 3.2.3). Indexing with ellipses seems to work more often than with nodals.
- If still have problems, double check distance and pixel size. Also make sure you are not using low T cell parameters while processing room T Laue data!
- If still have problems, double check (re-do) beam position refinement (3.2.2, 3.2.3 and 3.2.4).
- If still have problems – choose another image with more prominent nodals.
- The program finds typically several matches (2-3 to 50 or more!) when indexing and picks the best one. Matches are not all independent – several (many) could be very similar. If there is a problem and indexing appears difficult, one can select other matches and check if one of those is right (happens occasionally). To check, after indexing and while in the Index menu:

| | |
|---|---|
| A(ccept) | |
| 10 | *# number of the solution (match) you want to accept* |
| su(perimpose) | |

- After successful indexing one can check if the initial wavelength and resolution limits were reasonable. Vary these limits and compare predictions and observations. Adjust the limits to minimize number of unmatched observations (no prediction) or unmatched predictions (no observation).

## 3.4 Geometry refinement

- After successful indexing of an image, geometry needs to be refined to better match predictions and observations. Good prediction/observation match ensures a more successful integration later.
- Parameters that are refined are listed in Index/Refine/Parameters. It is typically necessary to refine all parameters for each frame, even when parameters do not physically vary from frame to frame (like distance and detector parameters).
- Three parameters, however, have to stay fixed during refinement of each frame as they are not independent from other parameters: one of cell lengths, one pixel size (horizontal or vertical) and one of detector tilt angles.
- Parameters can be turned ON and OFF during geometry refinement. Only crystal orientation angles (1,2,3) are turned ON initially. Other parameters should be turned ON gradually during the refinement.
- One needs to choose a "box size" (Index/Refine/Box). The meaning of the box size: only when a prediction is within the radius=box_size from the corresponding observation, they are used in the refinement. However, if there are several observations or several predictions within this radius, none are taken into account.
- Box size has to be large initially (15-20) if the indexed pattern is significantly off (0.1 – 0.2°) from the observed one. Using large box size will reduce the number of spots initially used for the refinement (there will be several observations or predictions within the radius=box_size). When predictions and observations for these initially used reflections are brought closer by the refinement, the box size has to be reduced to pick up more spots for further refinement.
- An error indicator window can be used to monitor progress of the refinement. The window displays how much each prediction is off in x and y from the corresponding observation. A circle is also shown with the radius=box_size. Reflections used for refinement are within this circle and are shown in red (green are the ones that are not used). As the refinement progresses, the number of red spots will increase and they will get concentrated at the center.
- Numerical indicators for progress of the refinement: 1) number of matched spots and 2) final residual (in pixels; the overall residual between predictions and observations). The number of matched spots will increase and final residual will decrease. For a very good geometry refinement, final residual is <0.5, for an acceptable refinement =1. The number of matched spots should be high (close to max) even at small box size.

- Standard procedure for geometry refinement:
  - Start with a large box (10-15), turn on 1,2,3, X and Y, refine.
  - Check error plot, number of matched spots and final residual. Keep reducing the box size without turning ON other parameters as long as the number of matched

spots is not being reduced (the red cluster of spots stays within the circle in the error plot).
- o When it becomes impossible to reduce box size further without losing matched spots, turn ON one or two other parameters.
- o Repeat last two steps until the box size is reduced to about 1 pixel while at the same time number of matched spots stays high (close to max, as this number will be increasing up to some point while at the end, for very small box size, will start decreasing somewhat).
- o Typical order of turning parameters ON:
  1,2,3,X,Y
  R,S
  D,H
  E,F
  Cell parameters
- o Cell parameters should be turned ON only at the very end of the refinement, at box size <3, if possible, to minimize an artificial change and fluctuation of cell parameters from frame to frame. Typically, cell parameters have to be turned ON to improve geometry refinement.

- It is not necessary to use the full resolution range for the refinement. One typically starts with lower resolution range and extends it later in the refinement.

- To refine load the default file of the indexed image, read the image, display and:

| | |
|---|---|
| I(ndex) | # go to Index menu |
| R(efine) | |
| S(pot) | # pick spots to be used for refinement |
| F(ind) | |
| S(igmacut) | # select spots with relatively high I/$\sigma_I$ (sigmacut) (5-10 if possible to find enough of those spots) |
| 10 | |
| H(ow many) | # select number of spots for geometry refinement, 200-500 typically |
| F(ind) | |
| 11 | # enter the slot number for spots, should be >10<br># picked spots will be marked by green crosses on the screen; if too many spots are picked up at the edge of the detector you can get rid of them by trimming the image. This is done by zooming in: use window open option in View menu (see **2.3.2**) and select the part of the image you want to use for finding spots for refinement) |
| Q(uit) | |
| P(arameters) | # toggle ON/OFF parameters for refinement |
| X | |
| Y | |
| Q(uit) | |
| E(rror) | # error plot |
| F(unction) | # toggle plotting ON |

| | |
|---|---|
| Q(uit) | |
| B(ox) | |
| 15 | # type box size |
| R(efine) | # refine;<br>error window pops up; left click to drop it on the screen;<br>check error plot, number of matched spots and drop between initial and final residual |
| Y(es) | # accept refinement cycle (or not) |

- continue either by going from B(ox) (reducing box size) or from P(arameters) (turning ON other parameters) depending on the outcome of this first refinement cycle, as described above.
- Record the way you reduced the box size and turned parameters ON during the geometry refinement since you want to apply same protocol for other images in the same data set.

- Do not forget to save the default file!!!

# 4. Processing a Laue data set

- As mentioned earlier, Laue data processing is somewhat iterative process due to the fact that best values for resolution limit and wavelength limits ($\lambda_{min}$ and $\lambda_{max}$) come from processing data itself. A second pass is then usually needed with these optimized parameters as input parameters for more optimal data processing. However, processing a Laue data set of 50-100 images is relatively slow – it typically requires a few days, as it is limited by several computationally intensive steps (like geometry refinement and integration).

- It is therefore highly advisable to initially process 3-4 images only and carry processing through geometry refinement, integration and scaling. This will provide a decent wavelength normalization curve, a good estimate for crystal resolution, as well as a good estimate of other data processing parameters. These parameters should then be used as input in processing of the entire data sets. Make sure to overestimate highest resolution for this initial processing of 1-3 images as this is necessary to determine the actual resolution limit from the data (see 4.6).

- Important: Please review section **2.2** on basic LaueView input/output files. You will be using C shell script files *mtf for batch processing of a data set.

## 4.1 Set-up the default file of the indexed/refined image for data set processing: change/setup filenames and directories

- Setup filenames and directory names in the first image default file. It is important that this step is done correctly at the beginning of a data set processing. This first default file will be copied to all other default files.

- Three filenames and three directory names need to be set:

Filename 1: image filename
Filename 2: same as 1 (image filename)
Filename 3: dataset name (name for the entire data set)
Directory 1: path/dir where the images are (default is set by WORK in the .cshrc file)
Directory 2: path/dir where the analysis files will be (typically ./)
Directory 3: path/dir where the crystal info file is (default is set by CRYSTALINFO in the .cshrc)

- To set filenames and directories:

```
LaueView filename                    # load default file for filename.def
file
0
1
filename
file
```

```
0
2
filename
file
0
3
ds-name
dir
0
1
where-the-images-are
dir
0
2
where-analysis-is-done
dir
0
3
where-the-crystal-info-file-is
F(ile)                                  # save default file!
D(efault)
W(rite)
Y(es)
Y(es)
Q(uit)
Q(uit)
```

### 4.2 Create and modify default files for all images

#### default.mtf

- This script copies the default file of the indexed/refined image and creates the default files for all other images. Only filenames are changed, φ settings are not.
- Input files: *.def
- Out put files: modified *.def
- File modifications:
    - Edit:

    ```
    set firstimage = d_001          # your indexed image filename
    ```

    - Change image filename list.

- Save these original *.def files in a safe place as you may need them later in the original form. They will be overwritten during the automated geometry refinement even when the refinement is not successful.

**default_modify.mtf**

- Run this script if you want to modify the existing default files. Should be used after default.mtf and before refinedef.mtf, if refinedef.mtf is used for geometry refinement.
- It will reload the original cell parameters from crystal_name.xtl file, set distance and pixel size to nominal values, and set-up proper φ angles for all frames.
- Input files: *.def
- Output files: modified *.def
- File modifications:

  - Edit:

```
set crystal        = mbco
set dis            = 150
set pixel_h        = 0.079
set pixel_v        = 0.079
set dataset_name   = mb1          # third filename in the default files
```

  - Change image filename list.
  - Change phi_angle list.

## 4.3 Geometry refinement

- Geometry refinement for a data set is done in a similar way as for the initial indexed image: spots are selected and refinement protocol should be the same as for the initial image.
- When refining a data set, there are several possibilities:

  1. Always start with the same initially indexed/refined image.
     - Use: **refine.mtf** or **refinedef.mtf**.
     - **refine.mtf :** resets distance and cell parameters, sets up φ angles, refines
     - **refinedef.mtf :** refines only. Run default_modify.mtf before running refinedef.mtf to reset distance, pixel size, cell parameters and to set up φ angles.
  2. Start refinement of each frame with parameters from the default file of the preceding refined frame.
     - Use: **refine_prog.mtf** (does not reset any parameters)

- Resetting parameters before each frame is refined minimizes propagation of error. You can, of course, customize the refinement scripts to reset parameters you want.

- You may encounter problems in carrying automated geometry refinement for an entire data set if frame-to-frame angular error (difference between nominal and actual crystal orientation angles φ, ω, χ) is = 0.1-0.2°. In order to determine if the automated refinement will work and which refine* mtf to use, check how well predicted is an image that is somewhat remote from the indexed image (10 images away or so). To check:

```
LaueView filename            # load default file for the indexed image filename.def ;
                             this will load all refined parameters including the crystal
                             orientation matrix
file
0
1
filename2                    # change filename to the name of remote image
F(ile)
E(xternal)
R(ead)                       # read the remote image
Y(/N)
V(iew)
V(iew)                       # view the remote image
Q(uit)
I(index)
G(oniometer)                 # set proper j   for the remote image
P(hi)
remote_image_phi
Q(uit)
su(perimpose)                # superimpose prediction pattern
```

If you find that predictions for the remote image are too far from the observations to be refined (try refining it interactively as you refined the indexed image, see 3.4), you will not be able to use refine.mtf.

In this case:

1. Try using refine_prog.mtf rather then refine.mtf.
2. Try following refinement steps 4-6 in flowChart.byRen.
3. Use following (somewhat tedious) protocol:
    - After default.mtf, run default_modify.mtf to reset parameters you want and to setup φ angles.
    - Interactively load frames, adjust φ, ω, χ angles by checking predictions vs observations by su(perimpose) (do not forget to get out of Index/Goniometer menu to Index menu to superimpose), save default files.
    - Sequence to load/display file and get you to Index/Goniometer manu (can store it in a file sequence_1 and call in while in LaueView main menu by @sequence_1):

f
e
r
y
v
v
q
i
g

- Sequence (sequence_2) to save the default file after an adjustment of angles (from Index menu):
  q
  f
  d
  w
  y
  y
  q
  q
  q
  q
- Run refinedef.mtf

### 4.3.1 Find spots for geometry refinement

**findspot.mtf**

- Finding spots for geometry refinement.
- Input files: *.def
- Output files: *.spt (spots stored in slot #11)
- File modifications:
  - o Edit (select sigmacut and numberofspots to pick several hundreds of spots for refinement):

```
set boxsize          = 9
set sigmacut         = 8          # spots with I/sI > 8  (or any chosen sigmacut)
                                  will be selected
set numberofspots    = 800
set slot             = 11
```

  - o Change image filename list.

- findspot.log file: search for "spots found" to check how many spots were selected

Example log file output:

```
639 spots found by I/sigma(I) > or =   8.000000
crossimg: less than        800 spots found
639 spots found and stored in slot #         11
minimum & maximum I/sigma(I):   8.242032       10.93876
```

### 4.3.2 Geometry refinement

**refine.mtf**

- Geometry refinement for all files in the data set.
- Input files: *.def
  *.spt
- Output files: modified *.def
- File modifications:
  - Edit:

```
set highest_resolution1        = 2.5
set highest_resolution2        = 2.0
set  lowest_resolution         = 100
set  longest_wavelength        = 1.55
set shortest_wavelength        = 0.73
set slot                       = 11
set final_tolerance            = 1.e-4
set crystalname                = mbco
set distance                   = 150
```

  - Set reference image:

```
set reference_image            = d_001
```

  - Change image filename list.
  - Change phi_angle list

- refine.log file: search for the end of refinement for each image, for example for “box size (pixel):   1.000000”

Make sure that all frames refined well!!!

Example:

```
# of matched spots:      691
```

```
box size (pixel):          1.500000
initial residual (pixel):  0.2583284
final residual (pixel)     0.2578535
```

**refinedef.mtf**

- Geometry refinement.
- Input files: *.def
  *.spt
- Output files: modified *.def
- File modifications:
  - Edit:

```
set crystal                  = mbco
set highest_resolution1      = 2.5
set highest_resolution2      = 2.0
set   lowest_resolution      = 100
set  longest_wavelength      = 1.55
set shortest_wavelength      = 0.73
set slot                     = 11
set final_tolerance          = 1.e-4
```

  - Change image filename list.

- Check refinedef.log (same as for refine.log).

**refine_prog.mtf**

- Geometry refinement.
- Input files: *.def
  *.spt
- Output files: modified *.def
- File modifications:
  - Edit:

```
set highest_resolution1      = 2.5
set highest_resolution2      = 2.0
set   lowest_resolution      = 100
set  longest_wavelength      = 1.55
set shortest_wavelength      = 0.73
set slot                     = 11
set final_tolerance          = 1.e-4
```

- Set first reference image:

  set first_reference = d_001

- Change image filename list
- Change phi angle list

- Check refine_prog.log (same as for refine.log).

**For all refine*.mtf scripts:** check and modify this part (refinement protocol) if necessary for the refinement of your data set

```
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
N(one)
X
Y
Q(uit)
B(ox)
10
R(efine)
Y/N)
B(ox)
9
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
1
2
3
H
Q(uit)
B(ox)
8
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
R
S
Q(uit)
B(ox)
6
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
E
```

```
Q(uit)
B(ox)
5
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
N(one)
D
Q(uit)
B(ox)
4
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
a
c
Q(uit)
B(ox)
3
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
A
B
G
Q(uit)
B(ox)
2
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
F
Q(uit)
B(ox)
1
R(efine)
Y/N)
resolution
$highest_resolution2 $lowest_resolution
B(ox)
0.8
R(efine)
Y/N)
T(olerance)
$final_tolerance
$final_tolerance
```

B(ox)
0.7
R(efine)
Y/N)

## 4.4 Creating a set file

**set.mtf**

- Create a ds-name.set file (ds-name=data set name). This file contains list of images in the data set with a pattern number assigned to each frame. It is important to create this file before processing of the data set!!! Otherwise scaling will fail.
- Input files: *.def
- Output files: ds-name.set
- File modifications:

    - Edit:

      set dataset_name = mb1 # *should match the third filename in the default files*

    - Change image filename list.

## 4.5 Profile fitting

- In this part of data processing:
    - overlapped reflections are predicted and marked (spoverlap_rdb.mtf) for given resolution and wavelength limits and sp_radius input parameter (see below)
    - strong, non-overlapping reflections are selected for analytical profile fitting (selectsam.mtf)
    - profile fitting is performed (sampling.mtf**)**
    - outliers are rejected (rejectsam.mtf)
- Prediction of overlapped spots is based on sp_radius parameter: Spots overlap if the distance between their centers is smaller than sp_radius. The overlap clearly depends on wavelength and resolution limits.
- Chose same sp_radius for both spoverlap_rdb.mtf and selectsam.mtf.
- sp_radius for sampling.mtf is the integration box size and should match the spot size (diameter of the spot!) typical for the diffraction patern (zoom in and count pixels/spot). Same sp_radius has to be used for integration later (integration_rdb.mtf).
- The sp_radius used for sampling.mtf, however, could be larger than the sp_radius used for spoverlap_rdb.mtf/selectsam.mtf. This will most likely be the case for very streaky diffraction patterns when choosing the spot size as sp_radius for selectsam.mtf will fail to select enough non-overlaping spots for profile fitting.
- Choose resolution limit as best or conservative estimate. Same limits for spoverlap_rdb.mtf and selectsam.mtf

- Make sure wavelength limits match the limits in the X-ray spectrum file (input for spoverlap_rdb.mtf). Same limits for spoverlap_rdb.mtf and selectsam.mtf
- Maximize number of spots selected by selectsam.mtf (as profile fitting is the best integration and should be done for as many spots as possible) by adjusting sigma cut but make sure < 1000 spots are selected for each frame.

## spoverlap_rdb.mtf

- Mark overlapping reflections.
- Input files: *.def
  ds-name.set
  X-ray-spectrum.lam
- Output files: *.lnk
- File modifications:
  - Edit:

```
set  highest_resolution        = 1.6
set   lowest_resolution        = 100
set  longest_wavelength        = 1.55
set shortest_wavelength        = 0.73
set xray_spectrum              =  x-ray-spectrum
set sp_radius                  = 15
set dataset                    = mb1
```

  - Change image filename list.

- spoverlap_rgb.log file: search for "reflections checked"

  Example log file output:

```
 2146   reflections checked
   797   reflections spatially overlapping with another
```

~20% overlap is typical, can be larger for streaky patterns

## selectsam.mtf

- Select spots for profile fitting.
- Input files: *.def, *.lnk, *.spt, ds-name.set
- Output files: *.spt (slot #12)
- File modifications:
  - Edit:

```
set  highest_resolution        = 1.6
set   lowest_resolution        = 100
set  longest_wavelength        = 1.55
set shortest_wavelength        = 0.73
set leftx                      = 1        # not necessary but you can crop horizontally
                                          (left/right) to avoid edge regions
set rightx                     = 2048
set topy                       = 1        # not necessary but can crop vertically
                                          (top/bottom) to avoid edge regions
set bottomy                    = 2048
set slot_4_selected_sam        = 12
set sp_radius                  = 15
set sigmacut                   = 3        # spots with I/sI > 3 (or any chosen sigmacut)
                                          will be selected
set prediction_error           = 2
set dataset                    = mb1
```

o Change image filename list.

- selectsam.log file: search for “spots checked”

  Example log file output::

```
 2133    spots checked
  797    spatial overlap
    0    no diffraction
    0    out of detector
    0    out of displaying image (no checking)
    0    out of view port
    0    on edge of loaded image
   37    off predicted
    2    overloaded
    0    not integrated
    0    negative or 0 sigma
  692    sigma cutoff
  611    spots selected in slot #        12
```

Make sure < 1000 spots are selected for each image!
To display selected spots:

LaueView filename
F(ile)
E(xternal)
R(ead)
F(ile)
S(pot)
R(ead)
Q(uit)

```
V(iew)
V(iew)
Q(uit)
P(ick)
S(tore)
M(ark)
S(lot)
12
U(nits)    # toggle to mm
M(ark)
```

This will mark spots selected by selectsam.mtf with green crosses.

**sampling.mtf**

- Profile fitting.
- Input files: *.def
  *.lnk
  *.spt
  ds-name.set
- Output files: *.sam
- File modifications:

  - Edit:

```
set  highest_resolution      = 1.6
set   lowest_resolution      = 100
set  longest_wavelength      = 1.55
set shortest_wavelength      = 0.73
set slot_4_selected_sam      = 12
set sp_radius                = 15
set a                        = 0.20
set b                        = 0.15
set dataset                  = mb1
```

  - Change image filename list.

**rejectsam.mtf**

- Interactive rejection of outliers.
- Check Appendix 1 for general info on rejection mtf files.
- Input files: *.sam
- Output files: *.sam
- File modifications:

  - Change image filename list.

- Reject reflections at the tails of the following pairs of histograms: 3/4, 10/11, 12/13, 14/15.
- Typically 5-10% rejected (total).

## 4.6 Integration

- Integration should be done in two cycles:
    1. Deliberately overestimate highest resolution (1.5? if the best estimate is 2Å) to determine the actual resolution limit from the data. This method of determining crystal diffraction limit, however, assumes that the detector was close enough during the data collection so that resolution is limited by the diffraction limit and not by the detector size. Diffraction limit is determined from the plot of $I/\sigma_I$ vs resolution (see Ren and Moffat, J. Appl. Cryst. 28, 1995, 461-481).
    2. Using the diffraction limit from 1. redo integration.

- Alternatively, resolution limit can be judged from the initial processing of 1-3 images. In that case only one integration cycle can be applied to the entire data set.

- One integration cycle consists of:
    - spoverlap_rdb.mtf: marks overlapped spots (for given resolution and wavelength limits and sp_radius)
    - integration_rdb.mtf: integration (one parameter fit that scales the profiles derived in sampling.mtf to fit intensities of all other predicted spots)

- Make sure that sp_radius in both cycles is the same for spoverlap_rdb.mtf and integration_rdb.mtf AND same as it was for sampling.mtf.

### spoverlap_rdb.mtf

- Same as described above.

### integration_rdb.mtf

- Integration of all reflections. Profiles from sampling.mtf applied and scaled to match observations (one parameter fit).
- Input files: *.def
  *.lnk
  *.sam
  ds-name.set
- Output files: *.sht
- File modifications:

    - Edit:

```
set  highest_resolution        = 1.6
set   lowest_resolution        = 100
set  longest_wavelength        = 1.55
set shortest_wavelength        = 0.73
set xray_spectrum              = X-ray-spectrum
set sp_radius                  = 15
set min_sam                    = 20
set dataset                    = mb1
```

  - Change image filename list.

**rejectsht_bg.mtf**

- Basic automatic rejection of outliers: I<1 and $I/\sigma_I < 0.01$ is rejected.
- This file also combines individual *.sht files into one sht file: ds-name.sht.
- Check Appendix 1 for general info on rejection mtf files.
- Input files: *.sht
- Output files: ds-name.sht
- File modifications:

  - Edit:

```
set dataset                    = mb1
set crystal                    = mbco
set x_min                      = 1
set x_max                      = 2048
set y_min                      = 1
set y_max                      = 2048
set sigmacut                   = 0.01
set longest_wavelength         = 1.55
set shortest_wavelength        = 0.73
```

  - Change image filename list.

- Check % of rejected spots.

**rejectsht_1x1.mtf**

- Interactive rejection of outliers, following the automatic one.
- Also, plot $I/\sigma_I$ vs resolution to determine the resolution limit in the first cycle of integration or during the initial processing of 1-3 images. $I/\sigma_I$ will decay with resolution until noise level is reached when it will flatten out. Crystal diffraction limit is determined as the break point in this plot (from decay to flat).
- Check Appendix 1 for general info on rejection mtf files.
- Input files: ds-name.sht
- Output files: ds-name.sht

- Call: **rejectsht_1x1.mtf ds-name tmp**
- File modifications:

    - Edit:

    set crystal = mbco

- To run rejectsht_1x1.mtf type: rejectsht_1x1.mtf data-set-file temp-file.
- Reject mainly poorly measured reflections with $I/\sigma_I$<0.2-0.4 (plot 13/15). $\sigma_I$ is the residual from the profile fit (not derived from multiple measurements or signal/background). Also check other columns like and reject outliers: 9/16 (h/h-bg), 12 (bg), 14 (sigma)…
- Do not forget to "mv tmp.sht data-set-file.sht" after exiting. Otherwise rejections will not be accepted.

## 4.7 Scaling

- Scaling of Laue data involves several cycles of scaling and rejection where in addition to frame-to-frame scale (F) and temperature (B) factors, wavelength normalization curve has to be derived. For more on Laue scaling check: Ren and Moffat, J. Appl. Cryst. 28, 1995, 461-481.
- Typical scaling/rejecting cycles for a complete data set :

    1. wavelength normalization only, 64 Chebyshev polynomial terms
       + reject <1% (only bad outliers) (start with 16 polynomial terms if processing 2-3 frames only)

    2. wavelength normalization (64 terms) + Fs
       + reject ~5%

    3. wavelength normalization (128 terms for undulator spectra) + Fs
       + major rejection (5-10%) (use 32 polynomial terms if processing 2-3 frames)

    4. same as 3.
       + reject if needed

    5. wavelength normalization + Bs

- Always finish with scaling rather than rejecting step.

- After each cycle of scaling check scaling log for unweighted and weighted $R_{merge}$ (for $F^2$). Large discrepancy between the two factors indicates that rejection is not sufficient enough (since when poorly measured reflections are weighted down, $R_{merge}$ improves). The $R_{merge}$ has to be < 20% even if it means rejecting a larger % of data. For a very good data set $R_{merge}$ =10-12% while completeness and redundancy is good and completeness in the last resolution shell is 30-50%.

- The major reduction in $R_{merge}$ occurs typically only during the first two cycles of scaling. For further reduction, poor data has to be rejected.

**scale.mtf**

- Scaling of data.
- Input files: ds-name.sht (for the first cycle),
  ds-name.fct (for all other cycles),
  ds-name.set,
  reference_image.def (default file of the image used as reference: F=1, B=0),
  ds-name.sca (output file of scaling.mtf, contains all scaling parameters; used automatically as an input file for scaling.mtf if exists)
- Output files: ds-name.sca
  ds-name.fct
- File modifications:

  - Edit and keep fixed for all scaling cycles:

```
set  highest_resolution      = 1.6
set   lowest_resolution      = 100
set  longest_wavelength      = 1.55
set shortest_wavelength      = 0.73
set referencewavelength      = 1.09       # chose wavelength with high X-ray intensity ,
                                          close to the peak intensity if possible (not
                                          necessary)
set dataset                  = mb1
set crystal                  = mbco
set reference_pattern        = d_001
```

  - Set for the first scaling cycle:

```
set input_data_type          = sht        # use ds-name.sht as input file
#set input_data_type         = fct
#set wavelength_range        = long
#set wavelength_range        = short
set wavelength_range         = ()
```

  - For all other scaling cycles:

```
#set input_data_type         = sht
set input_data_type          = fct        # use ds-name.fct as input file
#set wavelength_range        = long
#set wavelength_range        = short
set wavelength_range         = ()
```

  - Turn ON/OFF (1/0) the following parameters for various scaling cycles

```
 #1 = on; 0 = off
set weighting               = 1      # keep always ON (1)
set Lorentz                 = 1      # keep always ON (1)
set refine_polarization     = 0      # keep always OFF (0)
set refine_lambda_curve     = 1      # keep always ON (1)
set Chebyshev               = 64     # start with 64 in the first cycle, 128 for all other
                                     cycles
set isotropic_scaling       = 0      # OFF in the first cycle, turn ON in second and 3
set anisotropic_1           = 0      # keep OFF
set anisotropic_2           = 0      # keep OFF
set isotropic_B             = 0      # OFF in first few, turn ON at the end
set anisotropic_B           = 0      # keep OFF
```

- scale.log: search for the last "R-", check R and wR to determine rejection and next scaling step. Also check scaling parameters (listed in the log file).

  Example:

```
          R-factor:    12.78981    % sum | <F2(H)> - F2(H) | / sum F2
    delta R-factor:    2.0030141E-04
                       0.1568558    %
 weighted R-factor:    10.44685    % sum w*|<F2(H)>-F2(H)| / sum w*F2
    delta R-factor:    -9.8846853E-05
                       -9.4529368E-02%

current Chebyshev polynomial coefficients:
      order   coefficient
         1   1.049148
         2   1.268510
         3 -0.8905163
         4  0.6140925
                .
                .
                .

current isotropic scale-factors:
   pattern #   scale-factors
         1   1.000000
         2  0.9578620
         3   1.048997
         4  0.8885068
         5   1.285046
         6   1.525580
         7   1.555987
         8   1.440777
```

```
            9   1.266563
           10   1.164129

   current isotropic temperature-factors:
      pattern #   temperature-factors
            1  0.0000000E+00
            2 -4.8709000E-04
            3 -7.3173586E-03
            4 -1.8322733E-03
            5  1.0832351E-03
            6  5.1792352E-03
            7  1.3621163E-02
            8  9.1744522E-03
            9  4.6615120E-02
           10 -3.4104023E-02
```

- To check the wavelength normalization curve:

```
LaueView filename                    # load def file for the reference image
fil(e)
3
F(ile)
(s)C(ale)
R(ead)
Y(es)
Q(uit)
S(cale)
C(urrent)
P(lot)
X(-ray spectrum)
P(lot)
```

Or save the curve (see below) and plot using your favorite plotting program.

- To save wavelength normalization curve:

```
LaueView filename                    # load def file for the reference image
fil(e)
3
F(ile)
(s)C(ale)
R(ead)
Y(es)
X(-ray spectrum)
W(rite)
Q(uit)                               # if do not want to change wavelength interval type quit
```

| | |
|---|---|
| Q(uit) | # if do not want to change scale type quit |
| Y(es) | # yes will save spectrum in ds-name.lam file; to give it a different file name type: no, file, 0, 4, new_filename, quit, yes |

**rejectfct.mtf**

- Use to reject between scaling cycles.
- Input files: ds-name.fct
- Output files: ds-name.fct (modified)
- Call: **rejectfct.mtf ds-name**
- File modification:
  - Edit:

    set crystal = mbco

- Plot and reject: 28, 29, 30, 31
- Plot them all against wavelength (7) – they should be flat when data is scaled (could be a lot of scatter at wavelengths with low intensity); ripples or curved shape suggest that wavelength normalization is not completed.

## 4.8 Merging

- Merging of scaled data is done by **apply.mtf**. If run after the last cycle of scaling, data rejected during scaling will not be merged.
- If one wants to recover data rejected during scaling and merge them with the rest of the data, or apply before merging rejection criteria different than those used during scaling, or recover observations with redundancy=1, **sht2fct1.mtf** or **sht2fct2.mtf** can be used (**sht2fct1.mtf** recovers redundancy=1 observations while **sht2fct2.mtf** does not) before **apply.mtf**.

**sht2fct1.mtf**

- Recover all reflections from the ds-name.sht file, including redundancy=1 reflections.
- Input files: reference_image.def
  ds-name.set
  ds-name.sht
  ds-name.sca
- Output files: ds-name.fct
- File modifications:
  - Edit:

```
set crystal          = mbco
set dataset          = mb1
set reference_pattern          = d_001
set  highest_resolution        = 1.6
set   lowest_resolution        = 100
set  longest_wavelength        = 1.55
set shortest_wavelength        = 0.73
set referencewavelength        = 1.09
```

**sht2fct2.mtf**

- Same as sht2fct1.mtf except that redundancy=1 observation are not included.

**apply.mtf**

- Merge observations.
- Input files: reference_image.def
  ds-name.set
  ds-name.fct
  ds-name.sca
- Output files: ds-name.sin.hkl
- File modifications:

  - Edit (same inputs as for scaling!):

```
set  highest_resolution        = 1.6
set  longest_wavelength        = 1.55
set shortest_wavelength        = 0.73
set referencewavelength        = 1.09
set crystal                    = mbco
set dataset                    = mb1
set reference_pattern          = d_001
```

- apply.log: searh for “R-“ several times.

Example:

```
           R-factor:          12.78981    % sum | <F2(H)> - F2(H) | / sum F2
  weighted R-factor:          10.44685    % sum w*|<F2(H)>-F2(H)| / sum w*F2

unweighted R-factor on |F|:          7.038818    %
  weighted R-factor on |F|:          4.906307    %
  R-factor calculated from:          19417 reflections
        total measurments:          93732
          total reflections:          19417
        overall redundancy:          4.827316
```

R-factors on |F| as function of resolution

| from (A) | to (A) | # of measurments | unweighted R (%) | weighted R (%) |
|---|---|---|---|---|
| 100.000 | 3.200 | 25249 | 6.877 | 4.602 |
| 3.200 | 2.540 | 26413 | 7.289 | 4.708 |
| 2.540 | 2.219 | 17114 | 6.959 | 5.310 |
| 2.219 | 2.016 | 10647 | 6.611 | 5.120 |
| 2.016 | 1.871 | 6792 | 7.238 | 5.711 |
| 1.871 | 1.761 | 3871 | 7.945 | 6.321 |
| 1.761 | 1.673 | 2303 | 8.262 | 6.688 |
| 1.673 | 1.600 | 1343 | 8.028 | 6.351 |

R-factors on |F| as function of wavelength

| from (A) | to (A) | # of measurments | unweighted R (%) | weighted R (%) |
|---|---|---|---|---|
| 1.020 | 1.086 | 14084 | 7.445 | 5.324 |
| 1.086 | 1.152 | 34862 | 5.798 | 4.496 |
| 1.152 | 1.219 | 17590 | 6.356 | 4.998 |
| 1.219 | 1.285 | 10902 | 7.166 | 5.476 |
| 1.285 | 1.351 | 6892 | 8.072 | 6.298 |
| 1.351 | 1.418 | 4750 | 8.682 | 6.943 |
| 1.418 | 1.484 | 3102 | 8.788 | 7.226 |
| 1.484 | 1.550 | 1550 | 9.605 | 8.164 |

R-factors on |F| as function of integrated intensity

| from | to | # of measurments | unweighted R (%) | weighted R (%) |
|---|---|---|---|---|
| 25. | 79. | 6346 | 12.605 | 12.373 |
| 79. | 254. | 23512 | 10.085 | 8.553 |
| 254. | 811. | 28960 | 7.718 | 6.467 |
| 811. | 2595. | 20751 | 6.123 | 5.094 |
| 2595. | 8302. | 9513 | 5.108 | 4.157 |
| 8302. | 26561. | 3492 | 4.907 | 3.846 |
| 26561. | 84977. | 1045 | 5.008 | 4.089 |
| 84977. | 271862. | 113 | 4.433 | 3.961 |

## 4.9 Deconvolution of energy overlaps

**deconvolution.mtf**

- Deconvolution of energy overlaps. This script uses individual *.sht files for integration of multiples. Rejection criteria should be the same as for singles (rejectsht_1x1.mtf). If you have not recorded these criteria just run rejectsht_1x1.mtf again (without rejecting anything) to check the limits on I, I/$\sigma_I$, $\sigma_I$, height, height-bgr, background and other parameters (see below).
- Input files: *.def
  *.sht
  ds-name.set

ds-name.sca

- Output files: ds-name.har.hkl
- File modifications:
    - Edit (same criteria as for rejectsht_1x1.mtf):

```
set crystal                  =mbco
set dataset                  = mb1
set reference_pattern        = d_001
set  highest_resolution      = 1.6
set  longest_wavelength      = 1.55
set shortest_wavelength      = 0.73
set x_min                    = 1
set x_max                    = 2048
set y_min                    = 1
set y_max                    = 2048
set sigmacut                 = 0.3
set sigmamax                 = 44000
set sigmamin                 = 70
set i_min                    = 20
set height_max               = 32600
set height_min               = 40
set height_bgd_max           = 32400
set height_bgd_min           = 3
set bragg_max                = 0.68
set bragg_min                = 0
set spatial_overlap_min      = 0
set background_max           = 350
set background_min           = 40
```

## 4.10 Finalizing

- Finalizing involves calculating true $\sigma_F$ for singles, scaling $\sigma_F$ of multiples to those of singles, and combining singles and multiples. **final.mtf** is used. **final.mtf** will call **scalesigma.mtf.** Make sure you edit **scalesigma.mtf** to enter proper crystal name or set up the CRYSTALNAME variable correctly in your .cshrc file.
- Reflections can be rejected if necessary after **final.mtf** by **rejecthkl.mtf** (typically $F/\sigma_F < 1$, small %). Run **final.mtf** again after **rejecthkl.mtf**.

### final.mtf

- Input files: ds-name.sin.hkl
  ds-name.har.hkl
- Output files: ds-name.sin.hkl (modified)
  ds-name.har.hkl (modified)

ds-name.hkl (merged singles and multiples)

- Call: **final.mtf ds-name**
- File modifications:
    - Edit:

```
set crystal            = mbco
```

**rejecthkl.mtf**

- Input files: ds-name.sin.hkl
  ds-name.har.hkl
- Output files: ds-name.sin.hkl (modified)
  ds-name.har.hkl (modified)
- Call: **rejecthkl.mtf ds-name.sin ds-name.har**
- File modifications:
    - Edit:

```
set CRYSTALNAME = hbi_m37v
```

Repeat **final.mtf** after **rejecthkl.mtf** and save final statistics displayed on the screen.

- Check data completeness:

```
LaueView filename      # load in ref image def file
L(aueSim)
X(tal)                 # load crystal parameters from the crystal_info file (or skip if want to keep
                       crystal parameters from the def file)
N(ame)
N(ame)
crystal_name
L(oad)
dir                    # set up dir path and file name for the hkl file
2
fil
0
4
ds-name.hkl            # type hkl file name with the hkl extension
Q(uit)
H(kl)
L(oad)
R(ead)                 # read the hkl file
Y(es)
Q(uit)
```

```
C(ompleteness)
R(esolution)          # set resolution limits
100 1.6
B(in)                 # set number of resolution bins
8
C(ompleteness)        # display completeness
S(hell)               # toggle from shell to accumulative completeness
C(ompleteness)        # display accumulative completeness
```

## Appendix 1: Rejecting outliers

- Following mtf files use LauePlot for interactive rejection of outliers based on scatter plots and histograms (scatter plot in shown in red and histogram in green):
    - rejectsam.mtf (follows sampling.mtf)
    - rejectsht_bg.mtf (follows integration_rdb.mtf; non-interactive)
    - rejectsht_1x1.mtf (follows rejectsht_bg.mtf)
    - rejectfct.mtf (between scaling cycles)
    - rejecthkl.mtf (follows final.mtf)

- To reject (rejectfct.mtf example):

```
rejectfct.mtf ds-name
<CR>                    # skip title; a display window appears
<CR>                    # a list of columns is displayed, chose one for x and one for y axis
7 31                    # column 7 (wavelength) as x and 31 as y, for example
<CR>
S(et)
x-min x-max             # enter new values for x-min and x-max to cut out outliers; check the
                        histogram shown in green when deciding what to cut (cut the tails of
                        the distribution)
y-min y-max             # enter new values for y-min and y-max to cut out outliers
R(eject)
C(olumn)
31
<CR>                    # total rejection % is shown as well as % for each column
<CR>                    # to get a list of columns again
.
.
.
O(utput)                # to save modified ds-name.fct file
S(top)                  # to exit
```

- The rejected % for individual columns could add up to more than 100% as same reflections could fall into several rejection criteria.