LaueView2.4 Manual

Contents

Chapter 1

Version History

Program history:

?? ?? New version LaueView2.1.
Sep 96 Profile fitting ploblem fixed.
Oct 96 Indexing problem $P3_121/P3_112$ fixed.

Manual history:

Oct 96 First version manual based on E-mails between Zhong Ren, Mia Raves and Raimond Ravelli

Chapter 2

Introduction

We started to use *LaueView* for the data processing of Laue frames as an alternative for the *CCP4 Laue Software Program Suite*. Using the *CCP4* programs, we obtained data from a small number of Laue frames on acetylcholinesterase [ref?], that were used to produce interpretable electron density maps. However, the quality of the maps were inferior to maps obtained with data from monochromatic experiments. Another data set processed using the *CCP4* software concerned a small molecule. Although the data set was rather incomplete, the structure could be solved *ab initio* using *SHELXS*.

Our main problems with using the *CCP4 Laue Software Program Suite* were the high R-factors (20-30molecule data. The protein diffracted rather weak, which could explain the high R-factors, but the small molecule diffraction patterns were very nice (high I/sig). Optimization of the *CCP4* data processing using some jiffies and the program *SCALEPACK* to derive frame scaling and temperature factors externally and (important!) to reject individual outliers, improved the statistics somewhat, although not drastically. Although we realize that more experienced users might have obtained better hklIsigI's from our data, we decided just to redo the whole processing using another program, *LaueView*.

LaueView has first been described in Ren, Z. & Moffat, L. (1996). *J. Appl. Cryst.* **16**, 461-481. The deconvolution is explained in Ren, Z. & Moffat, L. (1996). *J. Appl. Cryst.* **16**, 482-493. There are a few important differences with the *CCP4* programs:

- The prediction of the spots is done using soft wavelength and resolution limits. Using a first approximation of the X-ray spectrum and Wilson statistics for the $|F|^2$ versus the resolution, all spots predicted to have an intensity less than a certain threshold are not treated at all.

- The integration is done by fitting an emprical function on the spots. The parameters that describe this function are determined using a few hunderds good, non-overlapping spots, and vary smoothly over the whole frame. For this purpose, the frame has been divided into a huge number of sections, compared to only 9 or 17 sections used by the *INTLAUE* that comes with the *CCP4* package.

- Chebyshev polynomials are used to describe the wavelength normalisation curve, which allows this curve to have sharp features like the Pt-absorption edge of a focussing mirror.
- The normalisation curve is refined interactively, and after each step outliers, defined using different criteria, can be rejected. Once the normalisation curve has been established well enough, some of the previously rejected reflections may be included again.
- Harmonic overlapping reflections can be deconvoluted and included into the final data set.
- The program includes a strategy option as well, which can be helpfull to design the most optimal data collection strategy given the unit cell, wavelength range, diffraction limit and crystal orientation.

This manual has been written based on E-mails between Zhong Ren, Mia Raves and Raimond Ravelli, as a result of the processing of an ESRF data set on acetylcholinesterase. During processing, a number of bugs have been found in the program. The user should be aware of the fact the bugs always can occur with (such a) extensive program.

The manual contains all the script files needed for the different stages of processing. Comments are added in the introductions of each stage of the data reduction.

Chapter 3

How to run LaueView

3.1 How to obtain the program

LaueView is free for academic and teaching purposes and can be obtained after sending an E-mail to Zhong Ren (renz@vishnu.uchicago.edu). He will ask you to sign an agreement and help to install and run the program.

3.2 System Requirements

The *LaueView* program only runs on SiliconGraphics computers. At least IRIX 5.3 should be running and a lot of memory (64 Mb?) and swap space (50 Mb?) should be available.

3.3 Starting LaueView

Define the following environment variables in your **.cshrc** file:

```
setenv LAUEVIEWHOME  '/mnt2/ks/raves/LaueView'
setenv WORK          '/mnt2/ks/ravelli/s999'
setenv CRYSTALNAME   's999'
setenv CRYSTALINFO   '/mnt2/ks/ravelli/s999'
setenv SCREENX       '1280'
setenv SCREENY       '1024'
setenv TMPDIR        '.'
```

The program can be runned either interactively or using some scripts (the **.mtf** files). If the program is started interactively, it will give a header, the environment variables and, if a default file was specified, the default parameters. This is followed by a menu:

```
        ? F(ile)
          V(iew)
          P(ick)
          E(llipse)
          I(ndex)
       (i)N(tegration)
          S(cale)
          L(aueSim)
          T(ime-resolved)
          Q(uit)
```

Each chapter of this manual will show how to go through some of the options interactively, and gives the script to use the program in batch mode for each of the different stages of data reduction. The next section explains how to display a frame interactively.

3.4 How to display a Laue Diffraction Image

To display a a diffraction pattern, two things has to be done: the frame has to be read in the right format, and the frame has to displayed in a way the user likes.

To read a frame, go to the LAUEVIEW:FILE:EXTERNAL_IMAGE: submenu. This menu looks like:

```
          ? E(nraf nonius fast area-detector: off)
            F(uji imaging plate           : on )
            K(odak storage phosphor       : off)
            O(ptronics photoscanner       : off)
            1(6-bit                       : off)
     (full) I(mage                        : on )
            L(ogrithmi                    : off)
            L(inear                       : on )
            U(pside down                  : off)
            G(aussian filter              : off)
            R(ead)
            Q(uit)
            <CR>(read)
```

Choose the right image format. The program can deal with the following frame-formats:

- ESRF-ID9 CCD camera. To specify the format of these frames go to LAUEVIEW:FILE:EXTERNAL_IMAGE:16-BIT: and give R(ECORDS) 1152, W(ORDS/RECORD) 1242, P(IXEL SIZE) 0.115 0.115 and S(ATURATION) 65535. Go back to the LAUEVIEW:FILE:EXTERNAL_IMAGE: menu by using Q(UIT) and turn the K(ODAK STORAGE PHOSPHOR) option to ON.

- FUJI imaging plate. Make sure that F(UJI IMAGING PLATE) is turned on. L(INEAR) should be turned ON, L(OGRITHMI) OFF.
- ... The *Enraf nonius fast area-detector*, the *Kodak storage phosphor* and the *Optronics photoscanner* still have to be described.

Now read the frame. Give, if necessary, the correct D(IRECTORY), F(ILENAME), M(IDDLENAME), and E(XTENSION). The four different options of the D(IRECTORY) and F(ILENAME) can be saved in a **.def** file. The program will tell whether the frame was read correctly or not.

To display the frame, go to the LAUEVIEW:LAUEVIEW submenu. This menu looks like:

```
? F(ull image: off)
  C(olor mode: heat)
  V(iew)
  R(eview)
  M(odify)
  W(indow)
  S(noop)
  G(nomonic)
  Q(uit)
  <CR>(view)
```

With V(IEW) or R(EVIEW) the image is actually displayed. The complete command REVIEW can always be given within the program. Adjust the H(IGHEST LEVEL) and the L(OWEST LEVEL) using the M(ODIFY) submenu if the colours are not scaled within the right pixel values (e.g. whole image is red or white). The colour mode can be changed using the C(OLOR MODE) submenu: we mainly have used H(EAT) and G(RAY-LEVEL). After changing the drawning options, always give V(IEW) or REVIEW to update the frame. This is also necessary after windows had been moved on the graphical display of the frame: the IRIX window manager will not update the frame by itself.

To zoom into a part of the frame, type WINDOW anywhere within the program. Place a box with the left mouse button on the frame and adjust the size of the box by dragging the middle mouse button. After a click with the middle mouse button within the box, the program will zoom into the selected part of the frame. To get the whole frame back again, use the F(ULL IMAGE) and V(IEW) options or type FULL anywhere within the program.

Chapter 4

Indexing and Refinement

4.1 Introduction

...

4.2 Manual Indexing

The indexing of laueview consists of the determination of the crystal orientation. Since the length of each reciprocal lattice vector that corresponds to each spot is unknown, a direct construction of the reciprocal space as is done by monochromatic indexing programs, is difficult. We have to make advance of the fact that nodals (low indices h k l and often multiples) are easily recognized in the Laue pattern. Comparing the angles between the corresponding reciprocal rays with a list of angles between the main reciprocal rays found in a simulated reciprocal lattice of the unit cell in a standard setting, gives us the relative orientation of the crystal. Since we are only working with angles, the direct beam position and crystal-detector distance have to be known very accurately.

Do the following things to prepare the indexing:

- Give the crystal-to-detector distance by typing **distance** anywhere.
- The unit cell and spacegroup need to be set correctly. One can do this in the LAUEVIEW:LAUESIM:XTAL: submenu.
- A starting beam center has to be found. Pick this manually from the screen using LAUEVIEW:PICK:CENTER.
- Refine the beam center

To refine the beam center, make use of the fact that all the conics will intersect eachother in the direct beam center. Pick some (10) spots on one ellipse using LaueView:pick:store_in_slot: and store them in a slot, say 1. Repeat this for at least 2 more conics. Store them in different slots, say 2, 3, etc. Refine the (theta/alpha) A(ngles only) of these ellipses using the LaueView:ellipse:refine: submenu:

```
0th ellipse
     ? (slot) #(:           1)
              C(enter)
    (pick an) E(llipse)
  (data from) S(lot)
              R(efine)
              Q(uit)
              <CR>(quit)
```

Select the first ellipse (slot) # 1, and click on the center of the ellipse after giving the (pick an) E(llipse) command. Click a few times on the left mouse button if you cannot get the right ellipse directly. Click on the middle mouse button to accept the ellipse. The program will give the ellipse parameters. Repeat the same procedure for ellipse 2, 3, ...

Now we can refine the ellipse parameters using R(efine). The program will give initial chi2's and final chi2's for each ellipse. The final chi2's should become close to 1. Q(uit) the LaueView:ellipse:refine submenu and go to the following refinement submenu ((angles &) C(enter):

4.3 Manual Refinement

4.4 default.mtf

This script uses as an input a manually well refined reference default. As an output, ***.def** files are generated for all the images that are on the image name list. See for an example of a **.def** file the *appendix*.

```
#!/bin/csh -f
#
# LaueView2.1 motif default.mtf
#
# This motif generates a set of default files for 1-spot Laue pattern.
# The first image must be trimmed properly, and the data set name needs to be
# set in the first image's default file.  These infomation will be spread out
# all over the data set.
#
```

```
# input:  \$firstimage.def
# output: directory2/filename2.def
#
# To run this motif in background, type
# default.mtf &

set firstimage = ache1

rm    default.log
touch default.log

# please replace image name list.
foreach image ( \
            ache1  ache2  ache3  ache4  ache5  ache6  \
            ache7  ache8  ache9  ache10 ache11 ache12 \
            ache13 ache14 ache15 ache16 ache17 ache18 \
            ache19 ache20 ache21 ache22 ache23 ache24 \
              )
   if (\$image != \$firstimage) then
      set pattern = \$image

      cat << endofinput > default.inp
F(ile)
D(efault)
directory
2
filename
0
1
\$image
filename
0
2
\$pattern
W(rite)
Y/N)
STOP
endofinput

      rm \$pattern.def
      LaueView \$firstimage << endofinput >> default.log
@default
endofinput
```

```
   endif
end

rm default.inp
exit
```

4.5 findspot.mtf

This script does a peak search on the frames. The spots will be stored in slot #11 in the output **.spt** files. As an input, the script need properly trimmed **.def** files. Check in the logfile **findspot.log** the number of spots actually found.

```
#!/bin/csh -f

# LaueView2.1 motif findspot.mtf
#
# This motif finds spots using a pattern recognition program.
# These spot coordinates will be used in the geometry refinement.
# This motif is for 1-spot pattern only.
# Spots found will be stored in slot 11, but you can change it if necessary.
# This motif will remove the old file \$pattern.spt.
#
# input:  \$pattern.def, directory1/filename1.img
# output: directory2/filename1.spt
#
# To run this motif in background, type:
# nohup findspot.mtf &

# please set these numbers
set how_nice_i_am = 20
set boxsize       = 11
set sigmacut      = 20
set numberofspots = 900
set slot          = 11

cat << endofinput > findspot.inp
nice
\$how_nice_i_am
F(ile)
E(xternal image)
directory
```

```
1
filename
1
R(ead)
N(o)
E(xtension)
sfc
Q(uit)
Y/N)
I(ndex)
R(efine)
S(pot)
F(ind)
B(ox)
\$boxsize
S(igma cut)
\$sigmacut
H(ow many)
\$numberofspots
F(ind)
\$slot
Q(uit)
Q(uit)
Q(uit)
F(ile)
S(pot)
directory
2
filename
1
W(rite)
Y/N)
STOP
endofinput

rm    findspot.log
touch findspot.log

# please replace image name list.
foreach image ( \
            ache1  ache2  ache3  ache4  ache5  ache6  \
            ache7  ache8  ache9  ache10 ache11 ache12 \
            ache13 ache14 ache15 ache16 ache17 ache18 \
            ache19 ache20 ache21 ache22 ache23 ache24 \
              )
   set pattern = \$image
   rm \$pattern.spt
   LaueView \$pattern << endofinput >> findspot.log
@findspot
endofinput

end

rm findspot.inp
exit

# to run this motif in background, type
# nohup findspot.mtf &
```

4.6 refine.mtf

This script indexes and refines all the frames specified in the **laueimage** array, using a well refined reference frame and the information about the spindle angles for every frame. The input **.def** files will be overwritten by the program as an output: be prepared to have backups in case something goes wrong. *LaueView* will match predicted spot postions with spots written previously to the **.spt** files (slot #11). A measured peak is considered to be indexed if the distance between a prediction and the actual spot postion is less then a certain box size. The **refine.mtf** script starts with a box size of 15, and decreases this slowly to 2 (pixel units). This final box size should be circa 3 times the refined root-mean-squared distance between the predicted and measured spot positions. Check in the **refine.log** the "# of matched spots".

```
#!/bin/csh -f
#
# LaueView2.1 motif refine.mtf
#
# Geometry refinement, including orientation, a, c, alpha, beta, gamma, center,
# distance, tilt angles, pixel size, bulge effect etc.
# Refine a reference image interactively then run this motif to get a set of
# images well refined.
# This motif is for 1-spot pattern only.
# This motif also does a detector edge testing that helps re-determine the
# center better.  The section can be commented out if the detector edges are
# not straight lines.
```

```
#
# !!!!!!! THIS MOTIF WILL OVERWRITE THE OLD DEFAULT FILE !!!!!!!
#
# input:  \$pattern.def, directory1/filename1.img, directory2/filename1.spt,
#         and directory2/\$reference_image.def
# output: directory2/filename1.def
#
# To run this motif in background, type
# nohup refine.mtf &

limit coredumpsize 0

# please set these values:
set how_nice_i_am       = 20
# Resolution and wavelength range for geometry refinement only.
# Don't set highest resolution to real diffraction limit.
set highest_resolution1 = 3.3
set highest_resolution2 = 3.0
set   lowest_resolution = 100
set  longest_wavelength = 1.5
set shortest_wavelength = 0.5
set slot                = 11
set final_tolerance     = 1.e-4
set crystalname         = ache
set distance            = 300
#   12345678901234567890

rm refine.log
touch refine.log

# Please replace image name list and set the reference image whose geometry
# refinement is already well done.
set reference_image     = ache1
set laueimage = ( \
                   ache1  ache2  ache3  ache4  ache5  ache6  \
                   ache7  ache8  ache9  ache10 ache11 ache12 \
                   ache13 ache14 ache15 ache16 ache17 ache18 \
                   ache19 ache20 ache21 ache22 ache23 ache24 \
                )

# Please replace phi angles for each image.
set phi       = ( \
                   0      -5.0   -10.0  -15.0  -20.0  -25.0  \
                    -30.0  -35.0  -40.0  -45.0  -50.0  -55.0  \
                    -60.0  -65.0  -70.0  -75.0  -80.0  -85.0  \
                    -90.0  -95.0  -100.0 -105.0 -110.0 -115.0 \
                 )

@ number = 1
foreach image (\$laueimage)
   if (\$image != \$reference_image) then
      set pattern   = \$laueimage[\$number]
      set phi_angle = \$phi[\$number]

      cat << endofinput > refine.inp
nice
\$how_nice_i_am
F(ile)
E(xternal image)
I(mage)
R(ead)
Y/N)
F(ile)
S(pot)
directory
2
filename
1
R(ead)
Y/N)
D(efault)
directory
2
filename
0
4
\$reference_image
R(ead)
Y/N)
Y/N)
Q(uit)
Q(uit)
resolution
\$highest_resolution1 \$lowest_resolution
wavelength
\$longest_wavelength \$shortest_wavelength
```

```
I(ndex)
C(rystal)
N(ame)
N(ame)
\$crystalname
L(oad)
Q(uit)
G(oniometer)
P(hi)
\$phi_angle
Q(uit)
R(efine)
S(pot)
S(lot)
\$slot
(G(edge)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<edge_testing
(directory<<<<<<<<<<<<<<<<<<<<<<<<<<<edge_testing
(2<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<edge_testing
(filename<<<<<<<<<<<<<<<<<<<<<<<<<<<<edge_testing
(2<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<edge_testing
(C(enter)<<<<<<<<<<<<<<<<<<<<<<<<<<<<edge_testing
(Y/N)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<edge_testing
(Q(uit)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<edge_testing
M(anual)
D(istance)
\$distance
T(ilt angles)
0 0 0
Q(uit)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
N(one)
(1
(2
(3
(a
(b
(c
(A(lpha)
(B(eta)
(G(amma)
(D(istance)
X
Y
```

```
(H(orizontal pixel size)
(V(ertical pixel size)
(R
(S
(T<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<nomally_detector_tilt_angle_T_is_kept_off
Q(uit)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
15
R(efine)
Y/N)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
14
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
N(one)
1
2
3
(a
(b
(c
(A(lpha)
(B(eta)
(G(amma)
(D(istance)
X
Y
(H(orizontal pixel size)
(V(ertical pixel size)
(R
(S
(T<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<nomally_detector_tilt_angle_T_is_kept_off
Q(uit)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
13
R(efine)
Y/N)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
12
R(efine)
Y/N)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
```

```
11
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
N(one)
1
2
3
a
(b
c
(A(lpha)
(B(eta)
(G(amma)
(D(istance)
X
Y
H(orizontal pixel size)
(V(ertical pixel size)
(R
(S
(T<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<nomally_detector_tilt_angle_T_is_kept_off
Q(uit)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
10
R(efine)
Y/N)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
8
R(efine)
Y/N)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
6
resolution
\$highest_resolution2 \$lowest_resolution
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
N(one)
1
2
3
a
```

```
(b
c
A(lpha)
B(eta)
G(amma)
D(istance)
X
Y
H(orizontal pixel size)
(V(ertical pixel size)
R
S
(T<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<nomally_detector_tilt_angle_T_is_kept_off
Q(uit)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
5
T(olerance)
\$final_tolerance
\$final_tolerance
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
N(one)
1
2
3
a
(b
c
A(lpha)
B(eta)
G(amma)
D(istance)
X
Y
H(orizontal pixel size)
(V(ertical pixel size)
R
S
(T<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<nomally_detector_tilt_angle_T_is_kept_off
E(bulge-square)
Q(uit)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
4
R(efine)
Y/N)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
3
R(efine)
Y/N)
P(arameters)<<<<<<<<<<<<<<<<<<<<<<<<this_is_the_beginning_of_a_cycle
N(one)
1
2
3
a
(b
c
A(lpha)
B(eta)
G(amma)
D(istance)
X
Y
H(orizontal pixel size)
(V(ertical pixel size)
R
S
(T<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<nomally_detector_tilt_angle_T_is_kept_off
E(bulge-square)
F(bulge-cubic)
Q(uit)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
2.0
R(efine)
Y/N)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
3
R(efine)
Y/N)
B(ox)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<set_box_size
2
R(efine)
Y/N)
D(efault file)
directory
```

```
2
filename
0
1
\$image
filename
0
2
\$pattern
W(rite)
Y/N)
Y/N)
STOP
endofinput

      touch \$pattern.def
      LaueView \$pattern << endofinput >> refine.log
@refine
endofinput

   endif
   set reference_image = \$laueimage[\$number]
   @ number++
end

rm refine.inp
exit

# To run this motif in background, type
# nohup refine.mtf &
```

Chapter 5

Profile Fitting

5.1 Introduction

. . .

5.2 Manual Control

. . .

5.3 set.mtf

This script generates one set file. An example of a set file is given in the *appendix*. It contains the number of frames, and the pattern name, pattern #, and direct beam center of each frame. This construction was made to allow the program to process time-resolved multiple-spot Laue diffraction patterns. As an input, the program uses all the **.def** files.

```
#!/bin/csh -f
#
# LaueView2.1 motif set.mtf
#
# This motif generates a .set file.  Run this motif after geometric refinement.
#
# input:  a set of .def files
# output: \$dataset_name.set
#
# To run this motif in background, type
# set.mtf &

limit coredumpsize 0
```

```
# please edit the dataset_name.
set dataset_name = ache

rm \$dataset_name.set
cat << endofinput > \$dataset_name.set
0
endofinput
rm set.log
touch set.log
rm laueview.def

# please replace the pattern names list.
foreach image ( \
            ache1  ache2  ache3  ache4  ache5  ache6  \
            ache7  ache8  ache9  ache10 ache11 ache12 \
            ache13 ache14 ache15 ache16 ache17 ache18 \
            ache19 ache20 ache21 ache22 ache23 ache24 \
              )
   set pattern = \$image

   cat << endofinput > set.inp
F(ile)
T(set)
filename
0
3
\$dataset_name
L(oad)
Y/N)
A(dd)
filename
0
1
\$pattern
D(efault file)
Y/N)

filename
3
S(ave)
Y/N)
Y/N)
```

```
STOP
endofinput

   LaueView << endofinput >> set.log
@set
endofinput

end

rm set.inp
exit

# To run this motif in background, type
# set.mtf &
```

5.4 spoverlap_rdb.mtf

This script predicts which spots are spatially overlapping with each other. The principle is to select the best spots and extract profiles from them. Best spots are of low and medium resolution, stimulated by the central part of the spectrum, no spatial overlaps.

The prediction is done resolution-dependent (LaueView:integration:spatial:Bandpass), which means that based on Wilson statistics and an input wavelangth normalisation curve, *LaueView* will calculate for each spot within the hard wavelength and resolution limits whether the reflection will probably have any significant intensity or not. Only those reflections that are predicted to have significant intensity will be treated further. The input wavelangth normalisation curve should be given as a **synchrotron.lam** file, and contains just some prior knowledge about the X-ray spectrum. This does not have to be perfect. The worst spectrum could be a step function right on the wavelength range

The minimum distance between two non-overlapping spots can be specified using the variable **sp_radius** and should be given in pixel units.

As an input, the script needs the **.def** files, the **.lam** and the **.set** file. New files, **.lnk**, will be written as the output files. They contain information about the predicted reflections with overlap information(?).

Search in the **spoverlap_rdb.log** file for the number of "reflections checked" and the number of "reflections spatially overlapping with another". There should be a reasonable large number of reflections left that are not overlapping with each other.

```
#!/bin/csh -f
#
```

```
# LaueView2.1 motif spoverlap_rdb.mtf
#
# Checking spatial overlap
# This motif is for 1-spot patterns only.
# This motif uses resolution-dependent bandpass.
#
# input:  \$pattern.def, directory2/filename3.set
# output: directory2/filename1.lnk
#
# To run this motif in background, type
# nohup spoverlap_rdb.mtf &

limit coredumpsize 0

# Please set these values:
set how_nice_i_am       = 20

set  highest_resolution = 3.5
set   lowest_resolution = 100
set  longest_wavelength = 1.00
set shortest_wavelength = 0.50
set xray_spectrum       = esrf
set sp_radius           = 13
#   12345678901234567890

cat << endofinput > spoverlap_rdb.inp
nice
\$how_nice_i_am
resolution
\$highest_resolution \$lowest_resolution
wavelength
\$longest_wavelength \$shortest_wavelength
F(ile)
T(set)
directory
2
filename
3
L(oad)
Y/N)
Q(uit)
X(-ray spectrum)
directory
```

```
2
filename
0
4
\$xray_spectrum
R(ead)
\$longest_wavelength \$shortest_wavelength
Y/N)
N)
Q(uit)
N(tegration)
S(patial)
R(radius)
\$sp_radius
B(andpass
print
S(patial)
C(lear)/A(ppend)?
directory
2
filename
1
W(rite)
Y/N)
STOP
endofinput

rm spoverlap_rdb.log1
touch spoverlap_rdb.log1

# please replace file name list
foreach pattern ( \
            ache1  ache2  ache3  ache4  ache5  ache6  \
            ache7  ache8  ache9  ache10 ache11 ache12 \
            ache13 ache14 ache15 ache16 ache17 ache18 \
            ache19 ache20 ache21 ache22 ache23 ache24 \
                )
   set image = \$pattern
   rm \$image.lnk
   LaueView \$pattern << endofinput >> spoverlap_rdb.log1
@spoverlap_rdb
endofinput
```

```
end

rm spoverlap_rdb.inp
exit

# To run this motif in background, type
# nohup spoverlap_rdb.mtf &
```

Please notice the input in spoverlap_rdb.mtf:

```
set  highest_resolution = 3.5
set   lowest_resolution = 100
set  longest_wavelength = 1.00
set shortest_wavelength = 0.50
set xray_spectrum       = esrf
set sp_radius           = 13
```

The resolution and spatial overlap radius is deliberately under-estimated. This makes the difference between reflections checked and spatial overlaps large enough, around 1000 for the acetylcholinesterase example.

5.5 selectsam.mtf

This script selects a subset from the non-overlapping spots to be used for profile fitting. Non-overlapping spots are rejected if they are predicted to lie outside a window, if they are below a certain I/sig, overloaded, negative or 0 sigma, etc.

The **sigmacut**, the minimum and maximum values for the X and Y positions (**leftx rightx topy bottomy**) and the maximum difference between the observed spot position and the predicted spot position in pixel units (**prediction_error**) can be specified.

As input, the **.def**, **.img**, **.spt**, and **.lnk** files are used. As an output, the **.spt** files are updated. Slot #12 will contain the subset of selected non-overlapping reflections.

Search in the logfile **selectsam.log** for "spots selected" to check whether enough spots were selected. Around 500 spots were selected in our example.

```
#!/bin/csh -f
#
# LaueView2.1 motif selectsam.mtf
#
# Setecting sample reflections
# This motif is for 1-spot pattern only.
# This motif modifies the existing .spt file.
```

```
#
# input:  \$pattern.def, directory1/filename1.img, directory2/filename1.spt,
#         directory2/filename3.set, directory2/filename1.lnk
# output: directory2/filename1.spt
#
# To run this motif in background, type
# nohup selectsam.mtf &

limit coredumpsize 0

# please set these values:
set how_nice_i_am       = 39

set  highest_resolution = 3.5
set   lowest_resolution = 100
set  longest_wavelength = 1.00
set shortest_wavelength = 0.50
set leftx               = 40
set rightx              = 1190
set topy                = 50
set bottomy             = 1130
set slot_4_selected_sam = 12
set sp_radius           = 13
set sigmacut            = 10
set prediction_error    = 3
#   12345678901234567890

cat << endofinput > selectsam.inp
nice
\$how_nice_i_am
resolution
\$highest_resolution \$lowest_resolution
wavelength
\$longest_wavelength \$shortest_wavelength
F(ile)
E(xternal image)
I(mage)
R(ead)
Y(es)
F(ile)
S(pot)
directory
2
```

```
filename
1
R(ead)
Y/N)
T(set)
directory
2
filename
3
L(oad)
Y/N)
Q(uit)
Q(uit)
P(ick)
S(pot)
D(elete)
\$slot_4_selected_sam
Q(uit)
Q(uit)
L(aueSim)
M(onitor)
V(iewport)
\$leftx \$topy \$rightx \$bottomy
Q(uit)
Q(uit)
N(tegration)
S(patial)
R(radius)
\$sp_radius
directory
2
filename
1
L(oad)
Y/N)
Q(uit)
2(d)
S(lot)
F(ind spot)
S(igmacut)
\$sigmacut
E(rror)
\$prediction_error
```

```
#
\$slot_4_selected_sam
F(ind)
Q(uit)
Q(uit)
Q(uit)
Q(uit)
F(ile)
S(pot)
directory
2
filename
1
W(rite)
Y/N)
Y/N)
STOP
endofinput

rm selectsam.log
touch selectsam.log

foreach pattern ( \
            ache1  ache2  ache3  ache4  ache5  ache6  \
            ache7  ache8  ache9  ache10 ache11 ache12 \
            ache13 ache14 ache15 ache16 ache17 ache18 \
            ache19 ache20 ache21 ache22 ache23 ache24 \
                )
   set image = \$pattern
   touch \$image.spt
   LaueView \$pattern << endofinput >> selectsam.log
@selectsam
endofinput

end

rm selectsam.inp
exit

# To run this motif in background, type
# nohup selectsam.mtf &
```

To display the spots stored in slot #12 on top of the displayed image:

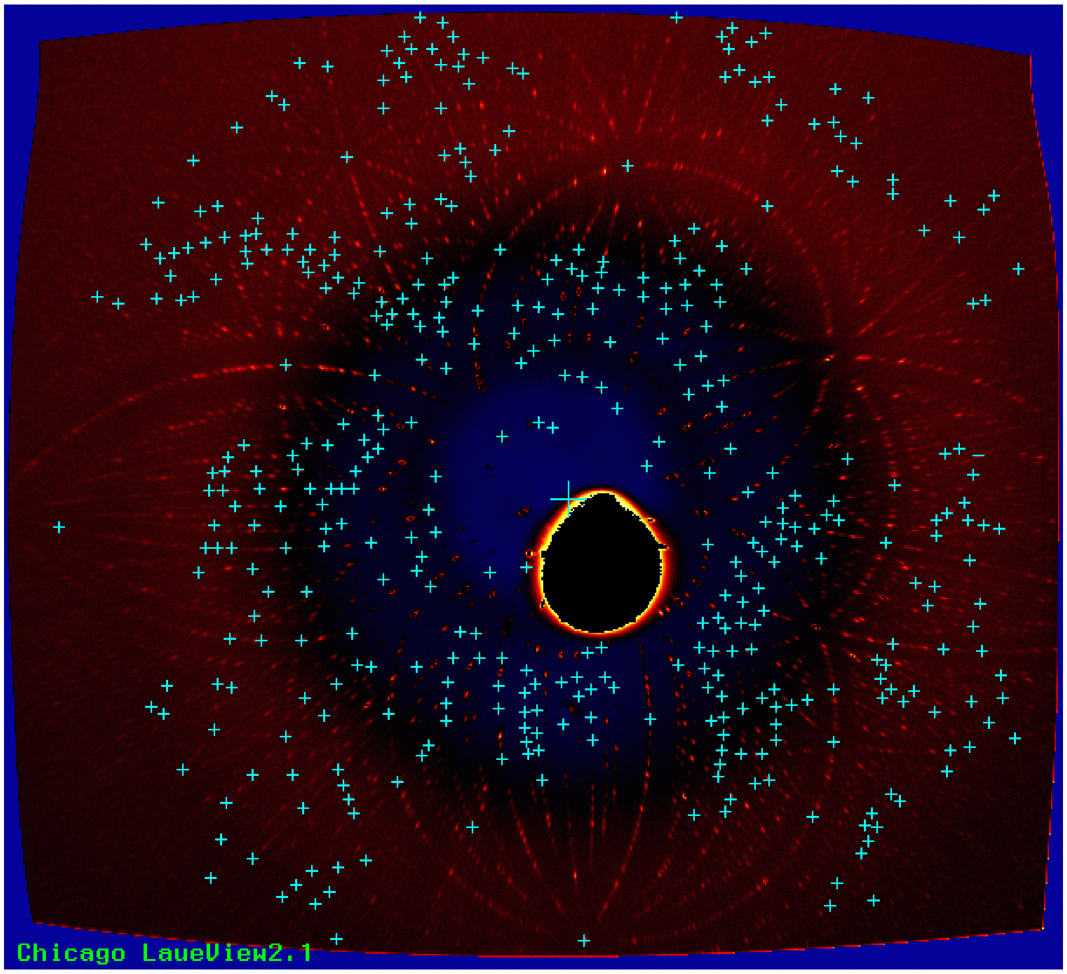

Figure 5.1: Selected spots for sampling profiles

- read and display the Laue frame,
- read the spot file (LAUEVIEW:FILE:SPOT:),
- go to the LAUEVIEW:PICK:STORE_IN_SLOT:MARK_ON_SCREEN: submenu, specify the correct S(LOT), change, if necessary, the U(NIT), and M(ARK) the spots on the screen.

The following figure gives an example of the non-overlapping spots to be used for profile fitting. The viewport could be made smaller for this example, and the spots are also not very well distributed over the whole frame.

5.6 sampling.mtf

This script will perform a profile fitting on the selected spots that are stored in slot #12.

n which the subset from the non-overlapping spots was stored by **select-sam.mtf** (#12). To start with correct values for the long (**a**) and short axes (**b**) of the integration profile, check the refinement of the profiles for a number of spots manually using the LaueView:integration:2d: submenu:

```
        ? M(anual)
          P(arameters)
          C(ontral)
   (clic)K
          S(lot)
          I(ntegration)
          R(eview)
          Q(uit)
          <CR>(quit)
```

The correct starting values for **a** and **b** as well as starting values for the other parameters that are used to descibe the spot profile, can be given in the M(anual) submenu.

In the P(arameters) submenu one can specify which parameters should be refined. A spot profile is defined by:

$$\begin{aligned} P &= pk * p(x,y) + p_x * (x-h) + p_y * (y-k) + bg \\ p(x,y) &= \exp\Big(-\Big\{\left[(x-h+d_x)*\cos(\phi+\epsilon)+(y-k+d_y)*\sin(\phi+\epsilon)\right]^2/A^2\Big\}^{ga} \\ &\quad -\Big\{\left[-(x-h+d_x)*\sin(\phi+\epsilon)+(y-k+d_y)*\cos(\phi+\epsilon)\right]^2/B^2\Big\}^{gb}\Big) \end{aligned}$$

where,

$$\begin{aligned} A &= a + s_a(x-h) + t_a(y-k) \\ B &= b + s_b(x-h) + t_b(y-k) \end{aligned}$$

Which of these parameters will be refined or not is controled by the P(arameters) submenu:

- **P(eak intensity:)** pk
- **B(ackground level:)** bg
- **(background) S(lope:)** p_x and p_b
- **(spatial) O(verlap:)** ? Not recommended

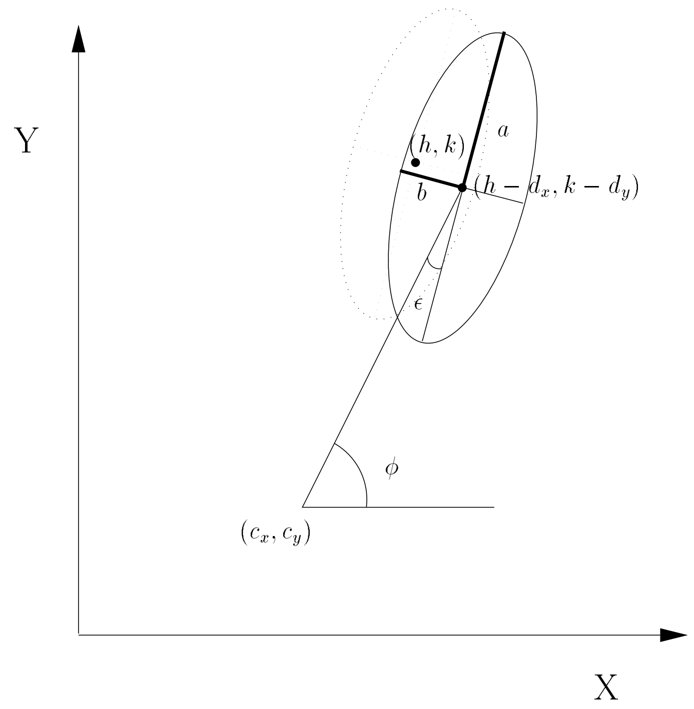

Figure 5.2: Parameters describing a spot profile

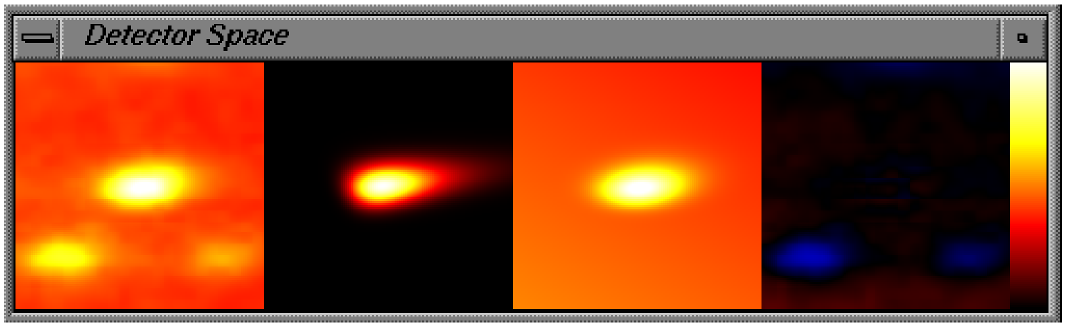

Figure 5.3: Checking individual integration profiles

- **R(adial streaky:)** a and b
- **A(bnormal spot:)** s_a, t_a, s_b and t_b
- **N(on-radial:)** ϵ
- **(prediction) E(rror)** d_x and d_y
- **(non-)G(aussian:)** ga and gb

Once good starting values were given and the right parameters were selected to be refined, one can (CLICK)K on a spot to refine the corresponding profile. A new window will appear with left (i) the original spot, then (ii) the standard integration spot, (iii) the refined profile, and (iv) the residual. The residual should be as small as possible.

As input, the .def, .img, .set, and .spt files are used. As an output, **.sam** files are produced containing the profile fitting parameters.

Check the number of lines in the .sam files (using unix command 'wc -l') to get an idea about the number of profiles derived for each frame.

```
#!/bin/csh -f
#
# LaueView2.1 motif sampling.mtf
#
# This motif fits sample profiles for a set of 1-spot pattern Laue images.
# This motif does not use profile mask.
#
# input:  \$pattern.def, directory1/filename1.img, directory2/filename1.spt,
#         directory2/filename3.set
# output: directory2/filename2.sam
#
```

```
# To run this motif in background, type
# nohup sampling.mtf &

limit coredumpsize 0

# please set these values:
set how_nice_i_am       = 39

set  highest_resolution = 3.5
set   lowest_resolution = 100
set  longest_wavelength = 1.00
set shortest_wavelength = 0.50

set slot_4_selected_sam = 12
set sp_radius           = 13
set a                   = 0.35
set b                   = 0.12
#   12345678901234567890
# The following two parameters are not used.
set maskcutoff          = 0.002

cat << endofinput > sampling.inp
nice
\$how_nice_i_am
resolution
\$highest_resolution \$lowest_resolution
wavelength
\$longest_wavelength \$shortest_wavelength
F(ile)
E(xternal image)
I(mage)
R(ead)
Y(es)
F(ile)
T(set)
directory
2
filename
3
L(oad)
Y/N)
Q(uit)
S(pot)
```

```
directory
2
file
1
R(ead)
Y/N)
Q(uit)
N(tegration)
2(d mosaic model)
M(anual)
1 (a)
\$a
2 (b)
\$b
Q(uit)
C(ontral)
R(adius
\$sp_radius
(C(utoff)
(\$maskcutoff
K(profile mask)<<<<<<<<<<<<<<<<<<<<<<<turn_off_profile_mask
G(auss)<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<use_Gauss-Jordan_elimination
Q(uit)
P(arameters)
S(lope           : off)
(O(verlap         : off)<<<<<<<<<<<<<<normally_keep_this_off
R(adial streaky  : off)
A(bnormal spot   : off)
N(on-radial      : off)
E(rror           : off)
(G(aussian        : off)
Q(uit)
S(lot)
O(n screen: on)
directory
2
filename
2
S(ample)
Y/N)
\$slot_4_selected_sam
STOP
endofinput
```

```
rm sampling.log
touch sampling.log

foreach pattern ( \
            ache1  ache2  ache3  ache4  ache5  ache6  \
            ache7  ache8  ache9  ache10 ache11 ache12 \
            ache13 ache14 ache15 ache16 ache17 ache18 \
            ache19 ache20 ache21 ache22 ache23 ache24 \
                )
   set image = \$pattern
   rm \$pattern.sam
   LaueView \$pattern << endofinput >> sampling.log
@sampling
endofinput
end

rm sampling.inp
exit

# To run this motif in background, type
# To run this motif in background, type
# nohup sampling.mtf &
```

5.7 rejectsam.mtf

The outliers in the sampling profiles can be rejected interactively using **rejectsam.mtf**. This is the first time we use LauePlot, a utility program. LauePlot displays 2d and 1d distributions of an ascii data file, rejects data, splits file, etc.
...

```
#!/bin/csh -f
#
# LaueView2.1 motif rejectsam.mtf
#
# Sample rejection
#
# input:  *.sam
# output: *.sam
#
# To run this motif interactively, type
# rejectsam.mtf
```

```
limit coredumpsize 0

set samfiles = ()
set samfilestmp = ()

# please replace filename list
set image = ( \
            ache1  ache2  ache3  ache4  ache5  ache6  \
            ache7  ache8  ache9  ache10 ache11 ache12 \
            ache13 ache14 ache15 ache16 ache17 ache18 \
            ache19 ache20 ache21 ache22 ache23 ache24 \
            )
foreach file (\$image)
   set samfiles    = (\$samfiles    \$file.sam)
   set samfilestmp = (\$samfilestmp \$file.tmp.sam)
end

cat << endofinput > sam.lab
lin x
lin y
lin half-long-axis-a
lin half-short-axis-b
lin abnormal-xa
lin abnormal-ya
lin abnormal-xb
lin abnormal-yb
lin non-radial-correction
lin off-prediction-x
lin off-prediction-y
lin slope-x
lin slope-y
lin non-Gaussian-correction-a
lin non-Gaussian-correction-b
lin chi-square
endofinput

#foreach file (\$image)
#   grep -v '*' \$file.sam > tmp.sam
#   mv tmp.sam \$file.sam
#end

rm \$samfilestmp
```

```
LauePlot2.1 \$samfiles -l sam.lab -c 3 4 -1wX 0 -ro \$samfilestmp -t
foreach file (\$image)
   mv \$file.tmp.sam \$file.sam
end

exit

# To run this motif interactively, type
# rejectsam.mtf
```

Chapter 6

Integration

6.1 Introduction

...

6.2 spoverlap_rdb2.mtf

We have to redetermine which spots are overlapping with eachother, using the 'correct' hard thresholds. So the only difference with the previous script **spoverlap_rdb.mtf** are the values for the **highest_resolution**, **lowest_resolution**, **longest_wavelength**, **shortest_wavelength** and **sp_radius**.

The right hard thresholds are not known *a priori*. We might have to run **spoverlap_rdb2.mtf** and **integration_rdb.mtf** a few times using different limits, and determine the border between signal and noise by inspecting plots like I/σ versus wavelength and/or resolution.

```
#!/bin/csh -f
#
# LaueView2.1 motif spoverlap_rdb.mtf
#
# Checking spatial overlap
# This motif is for 1-spot patterns only.
# This motif uses resolution-dependent bandpass.
#
# input:  \$pattern.def, directory2/filename3.set
# output: directory2/filename1.lnk
#
# To run this motif in background, type
# nohup spoverlap_rdb.mtf &
```

```
limit coredumpsize 0

# Please set these values:
set how_nice_i_am       = 20

set  highest_resolution = 2.7
set   lowest_resolution = 100
set  longest_wavelength = 1.40
set shortest_wavelength = 0.40
set xray_spectrum       = esrf
set sp_radius           = 13
#   12345678901234567890

cat << endofinput > spoverlap_rdb.inp
nice
\$how_nice_i_am
resolution
\$highest_resolution \$lowest_resolution
wavelength
\$longest_wavelength \$shortest_wavelength
F(ile)
T(set)
directory
2
filename
3
L(oad)
Y/N)
Q(uit)
X(-ray spectrum)
directory
2
filename
0
4
\$xray_spectrum
R(ead)
\$longest_wavelength \$shortest_wavelength
Y/N)
N)
Q(uit)
N(tegration)
S(patial)
```

```
R(radius)
\$sp_radius
B(andpass
print
S(patial)
C(lear)/A(ppend)?
directory
2
filename
1
W(rite)
Y/N)
STOP
endofinput

rm spoverlap_rdb.log1
touch spoverlap_rdb.log1

# please replace file name list
foreach pattern ( \
            ache1  ache2  ache3  ache4  ache5  ache6  \
            ache7  ache8  ache9  ache10 ache11 ache12 \
            ache13 ache14 ache15 ache16 ache17 ache18 \
            ache19 ache20 ache21 ache22 ache23 ache24 \
                )
   set image = \$pattern
   rm \$image.lnk
   LaueView \$pattern << endofinput >> spoverlap_rdb.log1
@spoverlap_rdb
endofinput

end

rm spoverlap_rdb.inp
exit

# To run this motif in background, type
# nohup spoverlap_rdb.mtf &
```

6.3 integration_rdb.mtf

This script will perform the actual integration . . . .

```
#!/bin/csh -f
#
# LaueView2.1 motif integration_rdb.mtf
#
# This motif integrates a set of 1-spot pattern Laue images.
# This motif does not use profile mask.
#
# input:  \$pattern.def, directory1/filename1.img, directory2/filename1.lnk,
#         directory2/filename3.set, directory2/filename1.sam
# output: directory2/filename2.sht
#
# To run this motif in background, type
# nohup integration_rdb.mtf &

limit coredumpsize 0

# please set these values:
set how_nice_i_am       = 20

set  highest_resolution = 2.7
set   lowest_resolution = 100
set  longest_wavelength = 1.40
set shortest_wavelength = 0.45
set xray_spectrum       = esrf
set sp_radius           = 13
set min_sam             = 20
#   12345678901234567890
# The following parameter is not used
set maskcutoff          = 0.01

cat << endofinput > integration_rdb.inp
nice
\$how_nice_i_am
resolution
\$highest_resolution \$lowest_resolution
wavelength
\$longest_wavelength \$shortest_wavelength
F(ile)
E(xternal image)
I(mage)
R(ead)
Y(es)
F(ile)
```

```
T(set)
directory
2
filename
3
L(oad)
Y/N)
Q(uit)
X(-ray spectrum)
directory
2
filename
0
4
\$xray_spectrum
R(ead)
\$longest_wavelength \$shortest_wavelength
Y/N)
N)
Q(uit)
N(tegration)
S(patial)
directory
2
filename
2
L(oad)
Y/N)
Q(uit)
2(d mosaic model)
S(lot)
B(in)<<<<<<<<<<<<<<<<<<<<call_bin_to_set_smaller_bin
#
\$min_sam
D(radius)
\$sp_radius
directory
2
filename
2
R(ead sample)
Y/N)
I(ntegration)
```

```
Q(uit)
C(ontral)
B(andpass
Q(uit)
I(ntegration)
S(creen)
R(adius)
\$sp_radius
(C(utoff)
(\$maskcutoff
K(profile mask)<<<<<<<<<<<<call_to_turn_it_off
directory
2
filename
2
I(ntegration)
Y/N)
STOP
endofinput

rm integration_rdb.log
touch integration_rdb.log

foreach pattern ( \
            ache1  ache2  ache3  ache4  ache5  ache6  \
            ache7  ache8  ache9  ache10 ache11 ache12 \
            ache13 ache14 ache15 ache16 ache17 ache18 \
            ache19 ache20 ache21 ache22 ache23 ache24 \
                )
   set image = \$pattern
   touch \$pattern.sam
   rm \$pattern.sht
   LaueView \$pattern << endofinput >> integration_rdb.log
@integration_rdb
endofinput
   grep -v na \$pattern.sht > tmp.sht
   mv tmp.sht \$pattern.sht
#   grep -v '*' tmp.sht > \$pattern.sht
end

rm integration_rdb.inp
rm tmp.sht
exit
```

```
# To run this motif in background, type
# nohup integration_rdb.mtf &
```

6.4 rejectsht_bg.mtf

The following script will combine all ***.sht** files into one **dataset.sht** and reject some spots automatically, based on the minimal and maximum values of the spot positions X and Y (**x_min**, **x_max**, **y_min**, and **y_max**), wavelength (**longest_wavelength** and **shortest_wavelength**) and I/σ ratio (**sigmacut**).

```
#!/bin/csh -f
#
# LaueView2.1 motif rejectsht_bg.mtf
#
# Data rejection after integration
# This motif filters a set of .sht files.  It only rejects several fields
# automatically.  Further rejection needs to be done manually, by
# rejectsht_1x1.mtf for example.  This motif also sorts the output file
# \$dataset.sht according to crystal symmetry.
# Anomalous scattering effect can be turned on or off.
#
# SEE ALSO rejectsht_bgall.mtf, rejectsht_1x1.mtf
#
# input:  *.sht
# output: \$dataset.sht
#
# To run this motif in background:
# nohup rejectsht_bg.mtf &

limit coredumpsize 0

# please set these:
set dataset            = ache
set crystal            = ache

set x_min              = 40
set x_max              = 1190
set y_min              = 50
set y_max              = 1130
set sigmacut           = 0.1
set longest_wavelength  = 1.2
set shortest_wavelength = 0.45
#   12345678901234567890

set shtfiles = ()
set shtfilestmp = ()

# please replace filename list
set image = ( \
            ache1  ache2  ache3  ache4  ache5  \
            ache6  ache7  ache8  ache9  ache10 \
            ache11 ache12 ache13 ache14 ache15 \
            ache16 ache17 ache18 ache19 ache20 \
            ache21 ache22 ache23 ache24 \
            )
foreach file (\$image)
#   grep -v NaN \$file.sht > tmp.sht
#   mv tmp.sht \$file.sht
   set shtfiles    = (\$shtfiles    \$file.sht)
   set shtfilestmp = (\$shtfilestmp \$file.tmp.sht)
end

set dataset = \$dataset.sht
set datasettmp = \$dataset.tmp

cat << endofinput > sht.lab
lin h
lin k
lin l
lin pattern #
lin multiplicity
lin spatial overlap
lin x (pixel)
lin y (pixel)
log height (count)
res resolution (A)
lin wavelength (A)
lin background (count)
log intensity
log sigma(I)
opl 13 / 14
opl 9 - 12
ope 11 / 10
err error code
```

```
12 13 14 11 15 8 16 17 1 27 28 21 7 6 31 30 29
endofinput

rm \$datasettmp

LauePlot2.1 \$shtfiles -l sht.lab -c 13 14  +u -rjo \$datasettmp -d -i \$crystal \
         << endofinput >& rejectsht_bg.log
S(et error bits)
1.e30 1
1.e30 1
P(lot)
7 8
S(et error bits)
\$x_min \$x_max
\$y_min \$y_max
P(lot)
11 15
S(et error bits)
\$longest_wavelength \$shortest_wavelength
\$sigmacut 1.e30
R(eject)
B(it#)
1
3
7
15
16
17
21
28
31
0
O(utput)
stop
endofinput

rm \$dataset
sort -T. +0n \$datasettmp -o \$datasettmp
LauePlot2.1 \$datasettmp -duo \$dataset << endofinput >>& rejectsht_bg.log
O(utput)
stop
endofinput
```

```
wc \$dataset >>& rejectsht_bg.log

rm \$datasettmp

exit

# To run this motif in background, type
# nohup rejectsht_bg.mtf &
```

6.5 rejectsht_1x1.mtf

Filter the single **.sht** file interactively. ...

```
#!/bin/csh -f
#
# LaueView2.1 motif rejectsht.mtf
#
# Data rejection
# This motif filters a single .sht file interactively.
# This motif has no sorting functions.
#
# SEE ALSO rejectsht_bg.mtf, rejectsht_bgall.mtf
#
# input:  \$1.sht
# output: \$2.sht
#
# To run this motif interactively, type
# rejectsht_1x1.mtf input output

limit coredumpsize 0


# please set:
set crystal = ache

if (\$\#argv < 2) then
   echo "Usage: rejectsht.mtf input output"
   exit 1
endif
set dataset = \$1.sht

cat << endofinput > sht.lab
```

```
lin h
lin k
lin l
lin pattern #
lin multiplicity
lin spatial overlap
lin x (pixel)
lin y (pixel)
log height (count)
res resolution (A)
lin wavelength (A)
lin background (count)
log intensity
log sigma(I)
opl 13 / 14
opl 9 - 12
ope 11 / 10
err error code
12 13 14 11 15 8 16 17 1 27 28 21 7 6 31 30 29
endofinput

LauePlot2.1 \$dataset -l sht.lab -c 13 14 -1wX 0 -rjo \$2.sht -t -i \$crystal
echo to accept the result, please mv \$2.sht \$dataset

exit

# To run this motif interactively, type
# rejectsht_1x1.mtf input output
```

Chapter 7

Deriving the Wavelength Normalisation Curve

7.1 Introduction

...

7.2 Manual Control

...

7.3 scale.mtf

This script is used to determine and refine the general scale factor $f_{general}$:

$$f_{general} = f_L f_P f_\lambda f_{isoS} f_{anisoS} f_{isoB} f_{anisoB} f_A f_U f_O, \tag{7.1}$$

where

- f_L is Lorentz factor ($f_L = \sin^2\theta$) *Is there a print error in equation 10 of article J.Appl.Cryst.28, p.468?*
- f_P is the Polarization factor ($f_P = 2/(1+\cos^2 2\theta - \tau\cos 2\varphi \sin^2 2\theta)$)
- f_λ is wavelength-normalization curve written as a Chebyshev-polynomal:

$$f_\lambda = \zeta + \exp\left\{\sum_{j=1}^{n_\lambda} c_j \left[\cos(j cos^{-1}\lambda') - \cos(j cos^{-1}\lambda'_r)\right]\right\}, \tag{7.2}$$

where λ' is the normalized wavelength,

$$\lambda' = \left[\lambda - \frac{1}{2}(\lambda_{max}+\lambda_{min})\right] / \frac{1}{2}(\lambda_{max}-\lambda_{min}) \tag{7.3}$$

and λ'_r is a normalized reference wavelength; when $\lambda = \lambda_r$, $f_\lambda = 1 + \zeta$, where ζ is a small positive number (say 10^{-10}

- f_{isoS} is an isotropic scale factor for each frame ($f_{isoS} = \exp s$)
- f_{anisoS} is an anisotropic scale factor for each frame
- f_{isoB} is an isotropic temperature factor for each frame ($f_{isoB} = \exp(-B \sin^2 \theta/\lambda^2)$)
- f_{anisoB} is an anisotropic temperature factor for each frame
- f_{isoD} is an isotropic radiation damage correction factor. It is another resolution-dependent scale factor in addition to the temperature factors
- f_{anisoD} is an anisotropic radiation damage correction factor
- f_A is a general absorption correction
- f_U is a detector spatial-nonuniformity correction
- f_O is a detector nonlinearity correction

...

```
#!/bin/csh -f
#
# LaueView2.1 motif scale.mtf
#
# Laue data scaling
#
# input:  \$reference_pattern.def, directory2/filename3.set,
#         directory2/filename3.\$wavelength_range.sca,
#         directory2/filename3.\$wavelength_range.sht or
#         directory2/filename3.\$wavelength_range.fct
# output: directory2/filename3.\$wavelength_range.fct
#
# To run this motif in background, type
# nohup scale.mtf &

limit coredumpsize 0

# please set these parameters:
set       how_nice_i_am = 39
set  highest_resolution = 2.7
set   lowest_resolution = 100
set  longest_wavelength = 1.4
set shortest_wavelength = 0.4
set referencewavelength = 0.7

set dataset              = ache
set crystal              = ache

#set input_data_type      = sht
set input_data_type      = fct
#set wavelength_range     = long
#set wavelength_range     = short
set wavelength_range     = ()

set tolerance            = 1.e-5
set SVD_tolerance        = 1.e-5
set max_iteration        = 100
set LM_constant          = 0.001
set lamda_rate           = 10
set reference_pattern    = ache1
set x_overall            = 24
set x_individual         = 24
set a_overall            = 8
set a_individual         = 8
#   12345678901234567890

if (\$input_data_type == sht) set inputdatatype = T
if (\$input_data_type == fct) set inputdatatype = F

cat << endofinput > scale.inp
nice
\$how_nice_i_am
resolution
\$highest_resolution \$lowest_resolution
wavelength
\$longest_wavelength \$shortest_wavelength
F(ile)
T(set)
L(oad)
Y/N)
Q(uit)
Q(uit)
L(aueSim)
X(tal)
N(ame)
```

```
N(ame)
\$crystal
L(oad)
Q(uit)
Q(uit)
S(cale)
L(oad)
O(rt: hkl values)
directory
2
filename
3
middlename
0
1
\$wavelength_range
\$inputdatatype
Y/N)
S(ca)
Y/N)
C(ommon)
Q(uit)
T(control)
T(olerance)
\$tolerance
I(teration)
\$max_iteration
S(VD tolerance
\$SVD_tolerance
C(onstant)
\$LM_constant
L(amda rate
\$lamda_rate
W(eighting)<<<<<<<<<<<<<<<<<<<<<<<weighting
Q(uit)
M(anual)
I(nput)
R(eference wavelength)
\$referencewavelength
Q(uit)
P(arameters)
M(reference pattern name
\$reference_pattern
```

```
Z(lorentz)<<<<<<<<<<<<<<<<<<<<<<<<selete_refinement_parameters
P(olarization)
(C(alculate)
R(efine)
Q(uit)
X(-ray spectrum normalization)
X(-ray spectrum normalization)
P(olynomial order)
\$x_overall
\$x_individual
Q(uit)
(F(actor)
(I(sotropic)
(F(actor)
(1(-Anisotropic)
(2(-Anisotropic)
(T(emperature factor)
(I(sotropic)
(T(emperature factor)
(A(nisotropic)
(G(eneral absorbtion)
(O(verall)
(A(bsorbtion order)
(\$a_overall
(G(eneral absorbtion)
(I(ndividual)
(A(bsorbtion order)
(\$a_individual
(0( absorbtion constant term)
(3( absorbtion cubic term)
(4( absorbtion quartic term)
(W(eight)
Q(uit)
S(cale)
C(urrent)
L(ist)
P(olarization)
X(-ray)
F(actor)
T(emperature
(G(eneral absorbtion)
Q(uit)
Q(uit)
```

```
W(rite
directory
2
filename
3
middlename
0
1
\$wavelength_range.tmp
S(ca
Y/N)
F(ct
W(rite)
Y/N)
STOP
endofinput

if (\$wavelength_range == '') then
   if (\$inputdatatype == T) touch \$dataset.sht
   if (\$inputdatatype == F) touch \$dataset.fct
   rm \$dataset.tmp.sca
   rm \$dataset.tmp.fct
else
   if (\$inputdatatype == T) touch \$dataset.\$wavelength_range.sht
   if (\$inputdatatype == F) touch \$dataset.\$wavelength_range.fct
   rm \$dataset.\$wavelength_range.tmp.sca
   rm \$dataset.\$wavelength_range.tmp.fct
endif

LaueView \$reference_pattern << endofinput >& scale.log
@scale
endofinput

if (\$wavelength_range == '') then
   mv \$dataset.tmp.sca \$dataset.sca
   mv \$dataset.tmp.fct \$dataset.fct
else
   mv \$dataset.\$wavelength_range.tmp.sca \$dataset.\$wavelength_range.sca
   mv \$dataset.\$wavelength_range.tmp.fct \$dataset.\$wavelength_range.fct
endif

rm scale.inp
exit
```

```
# To run this motif in background, type
# nohup scale.mtf &
```

7.4 rejectfct.mtf

Data rejection after scaling ...

```
#!/bin/csh -f
#
# LaueView2.1 motif rejectfct.mtf
#
# Data rejection after scaling
#
# input:  \$dataset.fct
# output: \$dataset.fct
#
# To run this motif interactively, type
# rejectfct.mtf fct_file

if (\$\#argv < 1) then
   echo "Usage: rejectfct.mtf fct_file"
   echo "Usage: file name implies the extension .fct"
   echo "Usage: xtal_info to be loaded from \$CRYSTALINFO/\$CRYSTALNAME.xtl"
   exit 1
endif

set dataset = \$1
set crystal = \$CRYSTALNAME

cat << endofinput > fct.lab
lin h
lin k
lin l
lin pattern #
lin x (pixel)
lin y (pixel)
lin wavelength (A)
log intensity I
log sigma(I)
log F2
log sigma(F2)
log <F2>
log sigma<F2>
lin F2-<F2>
log f(general)
lin f(Lorentz)
lin f(polarization)
log f(wavelength)
log f(scale)
lin f(temperature)
lin f(absorbtion)
lin f(nonuniformity)
lin f(nonlinearity)
lin f(radiation-damage)
opl 8 / 9
opl 10 / 11
opl 12 / 13
ope 14 / 10
ope 14 / 11
ope 14 / 12
ope 14 / 13
opr 1 2 3
endofinput

rm \$dataset.tmp.fct
LauePlot2.1 \$dataset.fct -l fct.lab -c 7 31 -1wX 0 -ro \$dataset.tmp.fct -t -i \$crys
mv \$dataset.tmp.fct \$dataset.fct
exit

# To run this motif interactively, type
# rejectfct.mtf fct_file
```

Chapter 8

Wavelength Normalisation

8.1 Introduction

...

8.2 Manual Control

...

8.3 sht2fct1.mtf or sht2fct2.mtf

```
#!/bin/csh -f
#
# LaueView2.1 motif sht2fct1.mtf
#
# Copy a .sht file to a .fct format.  The observations of redundancy=1 are
# included in the output.  This motif helps retrieving observations as many as
# possible.  However, the quality of redundancy-1 observations is unknown
# unless an external reference is available.  This motif also calculates the
# <F2> and the error etc. for evaluating the data quality of redundante data.
# LauePlot could be run to reject those reflections which have large error.
# The input .sht file must be properly sorted by symmetry, e.g. by
# rejectsht_bg.mtf or rejectsht_bgall.mtf.
# If one doesn't want to include redundancy-1 observations, the other motif
# sht2fct2.mtf should be run.
#
# input:  \$reference_pattern.def, directory2/filename3.set,
#         directory2/filename3.\$wavelength_range.sht, and
#         directory2/filename3.\$wavelength_range.sca
# output: directory2/filename3.\$wavelength_range.fct
#
# SEE ALSO rejectsht_bg.mtf, rejectsht_bgall.mtf, sht2fct2.mtf
#
# To run this motif in background, type
# nohup sht2fct1.mtf &

limit coredumpsize 0

if (\$#argv > 0) then
   echo "Usage: nohup sht2fct1.mtf &"
   exit 1
endif

# please set these parameters:
set how_nice_i_am       = 39
set crystal             = ache
set dataset             = ache
set reference_pattern   = ache1
set  highest_resolution = 2.7
set   lowest_resolution = 100
set  longest_wavelength = 1.4
set shortest_wavelength = 0.45
set referencewavelength = 0.7
set wavelength_range    = ()
#set wavelength_range    = long
#set wavelength_range    = short
#   12345678901234567890

#####################################################
# please check Lorentz factor and polarization factor
#####################################################

cat << endofinput > sht2fct1.inp
nice
\$how_nice_i_am
resolution
\$highest_resolution \$lowest_resolution
wavelength
\$longest_wavelength \$shortest_wavelength
F(ile)
T(set)
L(oad)
```

```
Y/N)
Q(uit)
Q(uit)
L(aueSim)
X(tal)
N(ame)
N(ame)
\$crystal
L(oad)
Q(uit)
Q(uit)
S(cale)
P(arameters)
Z(lorentz)<<<<<<<<<<<<<<<<<<<<<<Please_turn_on_or_off_Lorentz_factor
P(olarization)
C(alculate)<<<<<<<<<<<<<<<<<<<<<<Please_turn_on_or_off_polarization_factor
Q(uit)
Q(uit)
L(oad)
1(-redundancy: excluded)
directory
2
filename
3
middlename
0
1
\$wavelength_range
S(ca)
Y/N)
directory
2
filename
3
T(sht)
Y/N)
Y/N)
STOP
endofinput

if (\$wavelength_range == '') then
   rm    \$dataset.fct
   touch \$dataset.sht
   touch \$dataset.sca
else
   rm    \$dataset.\$wavelength_range.fct
   touch \$dataset.\$wavelength_range.sht
   touch \$dataset.\$wavelength_range.sca
endif

LaueView2.4 \$reference_pattern << endofinput >& sht2fct1.log
@sht2fct1
endofinput

rm sht2fct1.inp

exit

# To run this motif in background, type
# nohup sht2fct1.mtf &
```

8.4 rejectfct.mtf

8.5 apply.mtf

Appendix A

File Formats

A.1 Introduction

Some of the files used and or produced by *LaueView* are described overhere.

A.2 The .def file

The **.def** file contains all the values of the parameters that describe a certain Laue diffraction frame. It is needed for each diffraction pattern. The file looks like (line numbers are added):

```
 1  ache10
 2  ache10
 3  ache
 4
 5  /mnt2/ks/ravelli/data/esrf/edr/sfc/
 6  /mnt2/ks/ravelli/data/esrf/edr/
 7
 8
 9      2608.505       668.6285       604.3580
10  S
11            1152        1242
12               1           1
13     0.0000000E+00  0.0000000E+00           6
14           20000        5000
15     0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
16     0.0000000E+00  0.0000000E+00
17     0.0000000E+00            3   3.000000
18               5
19               1           1        1242        1152
```

```
   20             0           0           0           0
   21     0.0000000E+00  0.0000000E+00  0.0000000E+00  0.0000000E+00
   22     0.0000000E+00  0.0000000E+00
   23           152
   24      113.0000       112.7253       136.8326       89.97274       89.90104
   25      120.0523
   26     0.1149928      0.1150000
   27     0.4736209     -0.3193767      0.8207816      0.7243161      0.6714280
   28    -0.1566956     -0.5010499      0.6687201      0.5493313
   29     0.4736205      0.7243146     -0.5010493     -0.3193762      0.6714253
   30     0.6687176      0.8207811     -0.1566957      0.5493304
   31     -45.00000       90.00000      0.0000000E+00  0.0000000E+00
   32      1.500000      0.5000000
   33      3.000000       100.0000
   34    -0.8138283      0.1752279     -9.5462508E-02
   35     0.0000000E+00  0.0000000E+00
   36     0.0000000E+00  0.0000000E+00
   37     0.0000000E+00  0.0000000E+00
   38     0.0000000E+00  0.0000000E+00
   39     1.0633278E-04 -2.1477763E-06
```

The meaning of these values are given for each line number:

```
line  1:  file name 1 - pattern name.
line  2:  file name 2 - image name.
line  3:  file name 3 - data set name.
line  4:  file name 4 - anything.
line  5:  directory 1 - image file directory.
line  6:  directory 2 - working directory.
line  7:  directory 3 - xtal_info directory.
line  8:  directory 4 - anything.
line  9:  xtal-to-film distance and direct beam center in pixel coordinates.
line 10:  Unknown. Bravais lattice type?
line 11:  Frame file dimensions.
line 12:  Active records.
line 13:  Peak and background cut off and a third, unknown value.
line 14:  Highest and lowest level used to view the frame.
line 15 and 16:  Unknown.
line 17:  Unknown.
line 18:  Conventional box size.
line 19:  Window.
line 20:  Unknown.
line 21 and 22:  Unknown.
line 23:  Space group number.
line 24 and 25:  Cell parameters.
line 26:  Pixel size in both directions.
line 27 and 28:  Misseting matrix.
line 29 and 30:  Alignment matrix.
line 31:  Goniometer parameters omega, chi, phi, and 2-theta.
line 32:  Wavelength range.
line 33:  Resolution range.
line 34:  Detector tilt angles.
line 35:  Left edge x=a+by,a,b.
line 36:  Right edge x=a+by,a,b.
line 37:  Top edge x=a+by,a,b.
line 38:  Bottom edge x=a+by,a,b.
line 39:  Detector bulge constant.
```

A.3 The crystalname.set file

The **.set** file is generated by the **set.mtf** script. The file looks like this:

```
          24
acheed7_1            1       76.832       69.459
acheed7_2            2       76.848       69.452
acheed7_3            3       76.869       69.475
acheed7_4            4       76.853       69.479
acheed7_5            5       76.873       69.477
acheed7_6            6       76.822       69.473
acheed7_7            7       76.888       69.472
acheed7_8            8       76.846       69.457
acheed7_9            9       76.854       69.463
acheed7_10          10       76.822       69.453
acheed7_11          11       76.850       69.461
acheed7_12          12       76.847       69.461
acheed7_13          13       76.835       69.457
acheed7_14          14       76.829       69.469
acheed7_15          15       76.820       69.461
acheed7_16          16       76.816       69.453
acheed7_17          17       76.834       69.468
acheed7_18          18       76.816       69.456
acheed7_19          19       76.841       69.447
acheed7_20          20       76.925       69.450
acheed7_21          21       76.866       69.459
acheed7_22          22       76.877       69.449
```

```
acheed7_23          23       76.860       69.438
acheed7_24          24       76.890       69.446
```

The first column gives the pattern name, the second the pattern number, the third and the fourth the direct-beam center.

A.4 The synchrotron.lam file

The **synchrotron.lam** file contains an initial wavelength normalisation curve. It is used in the **spoverlap_rdb.mtf** and the **integration_rdb.mtf** scripts where the resolution-dependent bandpass is used to exclude reflections that do lie within the hard resolution and wavelength limits, but that are predicted to have very low intensities. If the lambda curve is completely unknown *a priori*, a simple block function between the wavelength limits can be used as well.

The **synchrotron.lam** file looks like this:

```
   0.4500000     9.0157814E-02
   0.4600000     0.1296453
   0.4700000     0.2105847
   0.4800000     0.3193447
   0.4899999     0.4004577
   0.4999999     0.4642419
   0.5099999     0.5297875
   0.5199999     0.5896280
   0.5299999     0.6365602
   0.5399999     0.6767832
   0.5499999     0.7180305
   0.5599999     0.7605418
   0.5699999     0.8004765
   0.5799999     0.8363830
   0.5899999     0.8702731
   0.5999998     0.9039699
   0.6099998     0.9362506
   0.6199998     0.9636853
   0.6299998     0.9836515
   0.6399998     0.9962929
   0.6499998      1.003817
   0.6599998      1.008365
   0.6699998      1.010620
   0.6799998      1.010093
   0.6899998      1.006366
   0.6999997      1.000000
   0.7099997     0.9923855
```

```
0.7199997     0.9848578
0.7299997     0.9779431
0.7399997     0.9712996
0.7499997     0.9642444
0.7599997     0.9562894
0.7699997     0.9471823
0.7799997     0.9364715
0.7899997     0.9231195
0.7999997     0.9057576
0.8099996     0.8836795
0.8199996     0.8579286
0.8299996     0.8315052
0.8399996     0.8082380
0.8499996     0.7908411
0.8599996     0.7792141
0.8699996     0.7699530
0.8799996     0.7576487
0.8899996     0.7378434
0.8999996     0.7101108
0.9099995     0.6788694
0.9199995     0.6509661
0.9299995     0.6317207
0.9399995     0.6218646
0.9499995     0.6166440
0.9599995     0.6076710
0.9699995     0.5876286
0.9799995     0.5554471
0.9899995     0.5173220
0.9999995     0.4821676
 1.010000     0.4557652
 1.020000     0.4380587
 1.029999     0.4244319
 1.039999     0.4098251
 1.049999     0.3930181
 1.059999     0.3769192
 1.069999     0.3637963
 1.079999     0.3502454
 1.089999     0.3287708
 1.099999     0.2982436
 1.109999     0.2714364
 1.119999     0.2659723
```

The first column gives the wavelength, the second the intensity.

A.5 The .lnk file

The **.lnk** file is produced by **spoverlap_rdb.mtf** and contains information about which spots are overlapping with which spots.

It looks like this:

```
   -28     17     35      1     11    378    750      0      0      0      0      0
   -28     19     34      1     17    423     28     77    758    925      0      0
   -28     20     34      1     58    453     30     79   1695      0      0      0
   -28     21     33      1     29    468     32     81      0      0      0      0
   -28     22     32      1      1    487     34     83    151      0      0      0
   -28     22     33      1     73    494     35     84    152    240    346    470
   -28     23     32      1     46    512     36     86    154    609      0      0
   -28     24     31      1     21    531     38     88    156    243      0      0
   -28     24     32      1     92    534     39     89   2104   1702    352    476
.................................
................
```

The first three columns represents hkl, the fourth column is the pattern number. If you find that the pattern number is equal to 0 or none of those in your **.set** file, there is an error!

The fifth and sixth column are the predicted spot positions X and Y in pixel coordinates, the fifteenth and sixteenth column give these positions in mm's. Column number seven till fourteen give information about the overlaps(?).

A.6 The .sam file

The **.sam** file contain integration profile parameters. These parameters are described in **sampling.mtf**.

One line in this file looks like:

```
    160.3920    286.2922    0.449264E+00  0.235475E+00 -0.688638E-01
  0.224186E+00  0.111742E+00  0.341956E-01 -0.363558E-01 -0.210948E-01
  0.472350E-02 -0.211654E+02 -0.156448E+01  0.100000E+01  0.100000E+01
  0.100705E+08
```

It gives the values of x, y (mm), a, b (mm), s_a, t_a, s_b, t_b, ϵ (rad), d_x, d_y (mm), p_x, p_y, ga, gb, χ^2.

A.7 The .sht file

The **.sht** file is produced by **integration_rdb.mtf** and contains information about the integrated intensities of each spot.

It looks like this:

image format	#Records	#Words/rec	Pixel size X Y		Saturation
Enraf nonius fast area-detector	512	512	?	?	?
Fuji imaging plate	2500	2048	0.100	0.100	?
Kodak storage phosphor	?	?	?	?	?
Optronics photoscanner	?	?	?	?	?
ESRF-ID9 CCD detector	1152	1242	0.115	0.115	65535

```
 -1  -1  -3  17   1  1.4950  613.86  624.19  3015 35.486340  0.792843  1855.67  0.33
  1   1   4  20   1  1.4950  736.23  575.62  2969 29.275257  0.821955  1834.57  0.38
  1   1   5  22   1  1.4670  735.53  571.13  5245 24.636526  0.702056  2026.16  0.92
 -2   1  -1  14   1  1.4950  632.94  579.74  5167 52.230186  0.868890  1956.43  0.89
 -1   2   7  22   1  0.9234  740.39  521.09  3436 18.480007  0.775899  2355.70  0.23
 -1   2   9  23   1  0.7465  749.75  512.31 14403 14.688395  0.689223  3450.41  0.19
 -1   2  12  24   1  1.4950  742.20  522.49  4874 11.179519  0.470983  2113.76  0.42
 -1   2  13  24   1  0.7492  794.28  464.88  8227 10.349585  0.743117  3248.02  0.94
  0  -2  -3  15   1  1.4950  624.73  640.07  3854 33.376400  0.722045  2711.32  0.19
.................................
.................
```

The first three columns represents *hkl*, the fourth column is the pattern number. The X and Y positions are given in column 7 and 8 in pixel coordinates. Column 10 is the resolution corresponding to the reflection, column 11 gives the wavelength.

What do the other columns mean? Which one is I, which is σ?

A.8 Surported images

The following image formats has been tested with *LaueView*.