dtintegrate
dtprofit

SBC01-????-00

January 9, 1996

Prepared by: J.W. Pflugrath
Molecular Structure Corporation
3200 Research Forest Drive
The Woodlands, Texas 77381
(713) 363-1033
(713) 364-3628 FAX
jwp@msc.com

Prepared for: Contract No. 943072401
Mary Westbrook
Argonne National Laboratory
9700 South Cass Avenue
Argonne, Illinois 60439
(708) 252-8914
(708) 252-4021 FAX
westbrook@anlel.el.anl.gov

Reviewed by:

Approved by:

TABLE OF CONTENTS

1 Scope and Purpose

This document describes how to use the dtintegrate and dtprofit modules of d*TREK. Together dtintegrate and dtprofit integrate a scan or series of single crystal X-ray diffraction images such as those created by the dtcollect module of d*TREK to produce a reflection list that contains the Miller index (hkl), estimated intensity and standard deviation for the Bragg reflections that appear in the scan. Dtintegrate uses the same procedures as dtpredict to predict reflection positions and as tCrefine to refine crystal, detector and source properties. The documentation for these modules should be consulted for more information.

dtintegrate reads the individual images of a scan and collects three-dimensional profiles from predicted reflection positions. Every few images (set by the `-batch` command line argument), the differences between observed and calculated reflection positions are minimized. The reflection profiles are written to a file named `reflnprofiles.profit`. dtprofit reads this file and performs 3D profiile analysis with the method of Kabsch (1988) and writes a reflection list file called `dtprofit.ref`.

dtintegrate gets its initial information about the detector, source and crystal properties from the header of an image file. This file is usually the output of the tCrefine module. This information is needed to predict reflection positions. The default is to integrate all reflections predicted in the scan, so that additional input is not required for integration.

1.1 Definitions and Abbreviations

The definitions described in the dtcollect, tCrefine, and dtpredict documents are appropriate for dtintegrate.

2 Background

In order to integrate the intensities of Bragg reflections in a single crystal diffraction experiment, their positions in images must be predicted accurately. This can be done provided information about the source, the crystal, the crystal goniometer, the detector, the detector goniometer, and the rotation is known. For the predictions to be accurate, the crystal, source and detector properties must be refined by minimizing the differences between observed and predicted reflection positions. In order to accurately predict reflections for images collected after the start of a scan, properties of the crystal, detector, and source should be refined periodically throughtout integration. One can assume the detector and source properties do not change over the course of a scan data collection and therefore not refine them. However, experience shows that the initial values for the detector and source may not be the most valid, so it is best to refine them just as one would refine crystal properties. Crystal properties can change due to radiation damage and incorrectly mounted crystals (i.e., the crystal can dry out or slip which changes the unit cell parameters and the crystal orientation).

Integration thus proceeds by looping through the images in a scan. For each image:

A. If this image is the last one in a 'batch' of images, then refine the crystal, detector and source properties based on the observed positions of reflections.

B. The reflections which appear in that image are predicted and the positions of the full reflections are re-predicted.

C. For each predicted reflection:

   1. If this is the first time this reflection is predicted, then allocate a 3D array to hold a set of pixels that would completely encompass the reflection. The array should be large enough to include pixels for a local background determination and allow for small shifts of the predicted reflection position due to crystal slippage and inaccurate prediction.

   2. Next copy a window of pixels around the reflection from the image to the 3D array and check if the 3D array is completed.

   3. If the array is full, then pre-process the reflection. It is corrected for non-uniformity of response and the observed reflection centroid is determined if the peak is strong enough. Mark the reflection as full.

   4. Then go on to the next reflection.

D. If this image is the last one in a 'batch' of images, then process the 'full' reflections. This means determine a local background, decide in a preliminary analysis which pixels are in background regions, which are in the peak region and which are in neither region (i.e. belong to the peak of another reflection). Process the array into 3D profiles. Write these profiles to disk for subsequent profile analysis.

E. Then go on to the next image.

When all images have been read in and processed, the result is a reflection list with preliminary integrated intensities and a file containing reflection profiles. dtprofit is then run to do the actual profile fitting. In profile fitting, one makes the valid assumption that reflections have the same spot boundary and the same (except for a constant scale factor) intensity distribution within the boundary. In other words, reflections have the same expected profiles except each is scaled by a different constant. Thus if a reference profile is determined, then a least-squares fit between the observed profile and the reference profile can be made and the scale factor determined. If the reference profile is scaled such that its integrated intensity is 1, then the scale factor between observed and reference profile is equal to the estimated intensity of the reflection. For more information on profile fitting please consult the following references:

Diamond, R. (1969) *Acta Cryst.* **A25**, 43-55. Profile Analysis in Single Crystal Diffractometry.

Ford, G.C. (1974) *J. Appl. Cryst.* **7**, 555-564. Intensity Determination by Profile Fitting Applied to Precession Photographs.

Rossmann, M.G. (1979) *J. Appl. Cryst.* **12**, 225-238. Processing Oscillation Diffraction Data for Very Large Unit Cells with an Automatic Convolution Technique and Profile Fitting.

Rossmann, M.G., Leslie, A.G.W., Abdel-Meguid, S.S. and Tsukihara, T. (1979) *J. Appl. Cryst.* **12**, 570-581. Processing and Post-Refinement of Oscillation Camera Data.

Kabsch, W. (1988) *J. Appl. Cryst.* **21**, 916-924. Evaluation of Single-Crystal X-ray Diffraction Data from a Position-Sensitive Detector.

Observed intensities are corrected for the Lorentz and polarization factors calculated by the prediction algorithm. They are not otherwise scaled, however. A subsequent step is needed to merge, scale and average the reflections before they are used in any crystallographic calculations.

3 Initial input image

dtintegrate reads the header of an input image specified on the command line to get the initial values for most of the objects that it manipulates. These include the source, crystal, detector, crystal goniometer, and scan properties. In order to begin its work, dtintegrate does not read the binary pixel data in the input image which may be even be missing (i.e. the image is dimensioned 0 by 0). If the detector spatial distortion and non-uniformity of response information in a header requires additional images to be read in, then dtintegrate does so automatically. A typical image header is shown below. The keywords required by dtintegrate are the same as those required by dtpredict, so please refer to that documentation for their descriptions. The close relationship between dtintegrate and dtpredict is obvious once you realize that the routines of dtpredict are used by dtintegrate to predict reflections.

```
{
HEADER_BYTES= 2048;
TYPE=mad;
SIZE1=0;
SIZE2=0;
CRYSTAL_GONIO_VALUES=0.0 0.0 0.0;
COMMENT=Header edited by dtheaderedit;
SCAN_TITLE=start end inc time nOsc nDark nDup nDlim nDC nDCup;
SCAN_ROTATION=0.0 12.0 0.2 4 0 1 0 100 1 0;
SCAN_ROTATION_AXIS_NAME=omega;
SCAN_ROTATION_VECTOR=1.0 0.0 0.0;
SCAN_TEMPLATE=lyso3.####.ss;
```

```
SCAN_SEQ_INFO=0 2 0;
ROTATION=0.0 0.2 0.2 4 0 1 0 100 1 0;
ROTATION_VECTOR=1.0 0.0 0.0;
ROTATION_AXIS_NAME=omega;
CRYSTAL_UNIT_CELL=88.00 88.00 104. 90.00 90.0 120.0;
CRYSTAL_MOSAICSPREAD=0.145;
CRYSTAL_DESCRIPTION=Test of dtpredict;
CRYSTAL_SPACEGROUP=175;
CRYSTAL_REFINE_FLAGS=0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
CRYSTAL_ORIENT_ANGLES=152.3 5.0 152.9;
CRYSTAL_ORIENT_VECTORS=1 0 0 0 1 0 0 0 1;
SOURCE_VECTORS=0.0 0.0 1.0  0 1 0 1 0 0;
SOURCE_POLARZ=0.5 1.0 0.0 0.0;
SOURCE_SPECTRAL_DISPERSION=0.0002 0.0002;
SOURCE_SIZE=0.0 0.0 0.0 0.0;
SOURCE_CROSSFIRE=0.0 0.0 0.0 0.0;
SOURCE_WAVELENGTH=1 1.54178;
SOURCE_REFINE_FLAGS=0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
DETECTOR_NUMBER=1;
DETECTOR_NAMES=D0_;
D0_NONUNF_TYPE=Dark_only;
D0_NONUNF_INFO=nonunf.img dark.img;
D0_SPATIAL_DISTORTION_TYPE=Interp_spatial;
D0_SPATIAL_DISTORTION_INFO=distor;
D0_DETECTOR_DIMENSIONS=512 512;
D0_DETECTOR_SIZE=50.0 50.0;
D0_DETECTOR_VECTORS=1 0 0 0 1 0;
D0_DETECTOR_DESCRIPTION=ANL-SBC gold detector single module;
D0_DETECTOR_REFINE_FLAGS=0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0;
D0_GONIO_NUM_VALUES=6;
D0_GONIO_NAMES=RotZ RotX RotY TransX TransY TransZ;
D0_GONIO_UNITS=deg deg deg mm mm mm;
D0_GONIO_VECTORS=0 0 1 1 0 0 0 1 0 1 0 0 0 1 0 0 0 -1;
D0_GONIO_VALUES=1.0 2.0 0.0 -.5 -.8 102.3;
D0_GONIO_DESCRIPTION=A simple detector goniostat;
CRYSTAL_GONIO_NUM_VALUES=3;
CRYSTAL_GONIO_NAMES=Omega Chi Phi;
CRYSTAL_GONIO_UNITS=deg deg deg;
CRYSTAL_GONIO_VECTORS=1 0 0 0 1 0 1 0 0;
CRYSTAL_GONIO_DESCRIPTION=A 3-circle eulerian crystal goniostat;
FILENAME=hex3.img;
BYTE_ORDER=big_endian;
Data_type=short int;
COMPRESSION=None;
}
```

Other command line options allow one set parameters specific to dtintegrate. These are described in the section 4.

4 Running dtintegrate

After dtintegrate has been installed and placed in your PATH, just enter dtintegrate along with command line options to run it. It parses the command line arguments, reads any required input images, spatial distortion files, non-uniformity files and writes a reflection profile file if reflections are integrated. Messages are written to stdout and stderr as required. The syntax for running dtintegrate is:

```
dtintegrate [-scan scanfile][ -reso rmin rmax |
                              -batch num_images_per_batch |
                              -seq  start_seq_num end_seq_num |
                              -pad num_pad_images
                              -window fast_size slow_size
                              -profsize profile_size1 profile_size2
                              -nonunf file_with_nonunf_info_in_header
                              ]
```

```
dtprofit
```

4.1 Command line options

Although dtintegrate will run without any command line options if scan information is found in the header of a file called `scan.img` in the current working directory, most situations will benefit from the use of one or more command line options described below. dtprofit has no command line options.

Command line options: Description and default

`-scan scanfile`

The header of image `scanfile` (no default name) is read for the keywords described in section 3 to set up the experiment properties required for predicting reflections. The keywords `SCAN_ROTATION`, `SCAN_ROTATION_VECTOR` are used to specify the rotation properties. If a `-scan` option is not specified, then dtintegrate uses '`-scan scan.img`' as the default.

`-reso fReso1 fReso2`

A resolution range in Ångstroms which overrides the default resolution range determined from the source wavelength and the detector(s) position. In order not miss any reflections on the image, do not use this option. If `fReso1` is 0.0 or less, then the resolution range is also determined from the source wavelength and detector(s) position.

`-seq Start_seq_num End_seq_num`

This overrides the default sequence of images that will be integrated as determined from the `SCAN_ROTATION` keyword. `Start_seq_num` is the sequence number of the first image to be integrated while `End_seq_num` is the sequence number of the last image to be integrated. The default is to integrate every image specified by the `SCAN_ROTATION` keyword in th scan file. Use this command line option to choose a specific sequence of contiquous images in a scan.

`-batch num_images_per_batch |`

This overrides the number of images (default=5) that will be processed before a refinement is carried out and reflection 3D profiles generated. It wastes time to refine too often, unless the crystal properties are changing drastically. It is also unwise to have the batch size very large as the 3D reflection profiles are kept in core memory until after a refinement is done so that accurate centroids for them can be re-predicted based on the observed centroids.

`-pad num_pad_images`

This overrides the number of images (default=0) that will be added *before and after* the predicted number of images that a reflection needs for its 3D profile. The extra images can be useful for background determination and to allow for crystal slippage. The actual number of images used for a reflection profile depends on the effective mosaic spread and reflection position (Lorentz factor).

`-window fast_size slow_size`

This overrides the nominal size of the window of pixels (default=21 by 21 pixels) in the image that will be accumulated in the 3D reflection profile. The actual size for any given reflection may be adjusted internally for spatial distortion and oblique incidence. The sizes need to be large enough to include a local background. The sizes can be so large that neighboring reflections are included in the area. These neighbors will be excluded automatically from the background determination.

`-profsize profile_size1 profile_size2`

This overrides the size of the Kabsch-style 3D profile (default=0.5, 0.3) for reflections. The first parameter describes the degrees subtended by the reflection profile in a plane perpendicular to the scattered beam wavevector, while the second parameter describes the degrees subtended by the reflection profile along a vector from the origin of the Ewald sphere and bisecting the reciprocal lattice vector. The first parameter is mainly a function of the reflection size on the detector and the crystal to detector distance. It can be approximated by arctan(window size / crystal-to-detector distance). The second parameter is mainly a function of the effective mosaic spread of the crystal, but in no case should be less than the rotation width of a single image.

`-nonunf file_with_nonunf_info_in_header`

Specifies an alternate file to retrieve non-uniformity information from. The default info comes from the `-scan` file. This option overrides the information in there. Note that this does not *directly* specify the non-uniformity file to use. The *header* of this file specifies the information to use which in turn may point to another file (which can be the same file).

4.2 Examples

```
dtintegrate -scan refine.head -seq 0 10
dtprofit
```

The above command line reads image `refine.head` for scan information and overrides the image sequence range found in that image with a new range of 0 to 10. Any non-uniformity correction information is found in the header of image `refine.head`. Results are written to the file `reflnprofiles.profit` which are then read by the dtprofit program to create the file `dtprofit.ref`.

```
dtintegrate -scan refine.head -reso 3.5 50 -pad 2 -batch 3 -window 25 25
dtprofit
```

The first command reads image `refine.head` for scan information and overrides the default resolution value of the entire detector surface with a resolution range of 3.5 to 50 Ångstroms. Each reflection 3D profile is padded by 2 images and is 25 by 25 pixels in size. Refinement is carried out after every 3 images.

4.3 Results

The command:

```
dtintegrate -scan newrefine.head -reso 10. 3.5 -pad 2 \
            -window 25 25 -batch 3 -seq 1 44
dtprofit
```

produced the following output which is explained below.

```
1   File newrefine.head successfully opened.
2   Cnonunf::nInitValues called
3   D0_NONUNF_TYPE: >>None<<


4   Command line string: >>-reso<<
```

```
Command line string: >>-pad<<
Command line string: >>-window<<
Command line string: >>-batch<<
Command line string: >>-seq<<
Integrate object listing:
Pad: 2
Window size: 25, 25
Profile size: 0.5, 0.3
Scan seqrange: 1, 44
Images/batch: 3
Resolution: 10, 3.5
dtintegrate: 3D method used
Cnonunf::nInitValues called
D0_NONUNF_TYPE: >>None<<

File /u6/jwp/DATA/NUCL4/nucl4001.osc successfully opened.
Header of file /u6/jwp/DATA/NUCL4/nucl4001.osc successfully read.
Image is 1900 by 1900 pixels.
Data_type in header is unsigned short int.
Compression_type is RAXIS.
Byte_order is big_endian.

IMAGE #: 0 Name: /u6/jwp/DATA/NUCL4/nucl4001.osc

Rotation list:
Start: 24
End: 24.25
Increment: 0.25
Time: 60
Predicting reflns...
Done predicting reflns...
Processing reflns ...
there are 597 in the list.
Number new, active, full, donew, dones:
206, 206, 0, 0, 0
Reflections deleted: 391

File /u6/jwp/DATA/NUCL4/nucl4002.osc successfully opened.
Header of file /u6/jwp/DATA/NUCL4/nucl4002.osc successfully read.
Image is 1900 by 1900 pixels.
Data_type in header is unsigned short int.
Compression_type is RAXIS.
Byte_order is big_endian.

IMAGE #: 1 Name: /u6/jwp/DATA/NUCL4/nucl4002.osc

Rotation list:
Start: 24.25
End: 24.5
Increment: 0.25
Time: 60
Predicting reflns...
Done predicting reflns...
Processing reflns ...
there are 803 in the list.
Number new, active, full, donew, dones:
72, 278, 0, 0, 0
Reflections deleted: 525

File /u6/jwp/DATA/NUCL4/nucl4003.osc successfully opened.
Header of file /u6/jwp/DATA/NUCL4/nucl4003.osc successfully read.
Image is 1900 by 1900 pixels.
Data_type in header is unsigned short int.
Compression_type is RAXIS.
Byte_order is big_endian.

IMAGE #: 2 Name: /u6/jwp/DATA/NUCL4/nucl4003.osc
```

```
Rotation list:
Start: 24.5
End: 24.75
Increment: 0.25
Time: 60
Predicting reflns...
Done predicting reflns...
Processing reflns ...
there are 858 in the list.
Number new, active, full, donew, dones:
56, 334, 0, 0, 0
Reflections deleted: 524
... end of batch ...

File /u6/jwp/DATA/NUCL4/nucl4004.osc successfully opened.
Header of file /u6/jwp/DATA/NUCL4/nucl4004.osc successfully read.
Image is 1900 by 1900 pixels.
Data_type in header is unsigned short int.
Compression_type is RAXIS.
Byte_order is big_endian.

IMAGE #: 3 Name: /u6/jwp/DATA/NUCL4/nucl4004.osc

Rotation list:
Start: 24.75
End: 25
Increment: 0.25
Time: 60
Predicting reflns...
Done predicting reflns...
Processing reflns ...
there are 1139 in the list.
Number new, active, full, donew, dones:
80, 414, 0, 0, 0
Integrate: start...
Strong: 0, Weak: 0
Integrate: ...done.
Reflections deleted: 725

File /u6/jwp/DATA/NUCL4/nucl4005.osc successfully opened.
Header of file /u6/jwp/DATA/NUCL4/nucl4005.osc successfully read.
Image is 1900 by 1900 pixels.
Data_type in header is unsigned short int.
Compression_type is RAXIS.
Byte_order is big_endian.

IMAGE #: 4 Name: /u6/jwp/DATA/NUCL4/nucl4005.osc

Rotation list:
Start: 25
End: 25.25
Increment: 0.25
Time: 60
Predicting reflns...
Done predicting reflns...
Processing reflns ...
there are 1018 in the list.
Number new, active, full, donew, dones:
70, 484, 20, 0, 20
Reflections deleted: 534

File /u6/jwp/DATA/NUCL4/nucl4006.osc successfully opened.
Header of file /u6/jwp/DATA/NUCL4/nucl4006.osc successfully read.
Image is 1900 by 1900 pixels.
Data_type in header is unsigned short int.
Compression_type is RAXIS.
Byte_order is big_endian.
```

```
IMAGE #: 5 Name: /u6/jwp/DATA/NUCL4/nucl4006.osc

Rotation list:
Start: 25.25
End: 25.5
Increment: 0.25
Time: 60
Predicting reflns...
Done predicting reflns...
Processing reflns ...
there are 1086 in the list.
Number new, active, full, donew, dones:
70, 534, 40, 0, 60
Reflections deleted: 532
... end of batch ...
Refine listing:

Crystal listing:

Unit cell lengths: 73.551, 68.265, 104.777
Unit cell  angles: 90, 90, 90
Unit cell  volume: 526081

Spacegroup number: 16
name: P222
Num. equiv. posns: 4
Equival. position  1:
     0,      0           0
     1,      0           0
     0,      1           0

Equival. position  2:
     0,      0           0
    -1,      0           0
     0,      1           0

Equival. position  3:
     0,      0           0
    -1,      0           0
     0,     -1           0

Equival. position  4:
     0,      0           0
     1,      0           0
     0,     -1           0

Orientation angles: 172.744, -29.3, 15.008
Mosaic Spread: 0.6
Description: Test of d*TREK modules

CRYSTAL_Goniometer listing:

Description: A 3-circle eulerian crystal goniostat
Number of values: 3

Value 0 name: Omega
units: deg
Vector: 1, 0, 0
Datum value: -24
Current value: Unknown

Value 1 name: Chi
units: deg
Vector: 0, 0, 1
Datum value: 0
Current value: Unknown

Value 2 name: Phi
```

```
units: deg
Vector: 1, 0, 0
Datum value: 0
Current value: Unknown


Source listing:

Single wavelength: 1.54178
Direction vector: 0, 0, 1
Spectral Dispersion: 0.0002, 0.0002
Crossfire: 0, 0, 0, 0
Polarization: 0.5, 1, 0, 0
Size: 0, 0, 0, 0
Intensity: 0
Refinement flags: Not yet available

D0_Detector listing:

Pixel dimensions: 1900, 1900
Nominal size in mm: 193.23, 199.5
Description: R-AXIS IIC at MSC
Rotation: 0, 0, 0
Translation: -0.142, 0.197, -96.346
Fast pixel vector: 0, 1, 0
Slow pixel vector: -1, 0, 0

D0_Goniometer listing:

Description: A simple detector goniostat
Number of values: 6

Value 0 name: RotZ
units: deg
Vector: 0, 0, 1
Datum value: 1.234
Current value: Unknown

Value 1 name: RotX/Swing
units: deg
Vector: 1, 0, 0
Datum value: 0
Current value: Unknown

Value 2 name: RotY
units: deg
Vector: 0, 1, 0
Datum value: 0
Current value: Unknown

Value 3 name: TransX
units: mm
Vector: 1, 0, 0
Datum value: -0.142
Current value: Unknown

Value 4 name: TransY
units: mm
Vector: 0, 1, 0
Datum value: 0.197
Current value: Unknown

Value 5 name: TransZ/Dist
units: mm
Vector: 0, 0, -1
Datum value: 96.346
Current value: Unknown
```

```
DetResolution min: 154178
DetResolution max: 1.65533

Internal loop:        0
Reflections accepted: 60
Reflections rejected: 0

Internal loop:        1
Reflections accepted: 60
Reflections rejected: 0

Internal loop:        2
Reflections accepted: 60
Reflections rejected: 0

Internal loop:        3
Reflections accepted: 60
Reflections rejected: 0

Internal loop:        4
Reflections accepted: 60
Reflections rejected: 0

Refinement residuals
rmsResid (mm)  = 0.130414
rmsResid (Deg) = 0.0344216

Crystal
a, b, c:    73.395     68.225    104.852
Sigmas:     0.0646     0.0110     0.0252
Shifts:     0.0000     0.0000     0.0001
alpha, beta, gamma:    90.000     90.000     90.000
Sigmas:     0.0000     0.0000     0.0000
Shifts:     0.0000     0.0000     0.0000
Rot1, Rot2, Rot3:   172.748    -29.297     14.914
Sigmas:     0.0036     0.0100     0.0726
Shifts:     0.0000    -0.0002    -0.0249
Mosaicity:     0.600
Sigma:     0.0000
Shift:     0.0000

Detector:  0
Trans1, Trans2, Trans3:    -0.008      0.158     96.996
Sigmas:     0.1148     0.0459     0.1999
Shifts:     0.0000     0.0000     0.0210
Rot1, Rot2, Rot3:     0.848      0.000      0.000
Sigmas:     0.0837     0.0000     0.0000
Shifts:    -0.0125     0.0000     0.0000

Source
Rot1, Rot2, Wavelength:     0.000      0.000      1.5418
Sigmas:     0.0000     0.0000     0.0000
Shifts:     0.0000     0.0000     0.0000

File intrefine.head successfully opened.

File /u6/jwp/DATA/NUCL4/nucl4007.osc successfully opened.
Header of file /u6/jwp/DATA/NUCL4/nucl4007.osc successfully read.
Image is 1900 by 1900 pixels.
Data_type in header is unsigned short int.
Compression_type is RAXIS.
Byte_order is big_endian.

IMAGE #: 6 Name: /u6/jwp/DATA/NUCL4/nucl4007.osc

Rotation list:
Start: 25.5
End: 25.75
```

```
Increment: 0.25
Time: 60
Predicting reflns...
Done predicting reflns...
Processing reflns ...
there are 1369 in the list.
Number new, active, full, donew, dones:
74, 568, 58, 0, 118
Integrate: start...
Strong: 118, Weak: 0
Integrate: ...done.
Reflections deleted: 859

File /u6/jwp/DATA/NUCL4/nucl4008.osc successfully opened.
Header of file /u6/jwp/DATA/NUCL4/nucl4008.osc successfully read.
Image is 1900 by 1900 pixels.
Data_type in header is unsigned short int.
Compression_type is RAXIS.
Byte_order is big_endian.

IMAGE #: 7 Name: /u6/jwp/DATA/NUCL4/nucl4008.osc

Rotation list:
Start: 25.75
End: 26
Increment: 0.25
Time: 60
Predicting reflns...
Done predicting reflns...
Processing reflns ...
there are 1110 in the list.
Number new, active, full, donew, dones:
63, 573, 64, 0, 64
Reflections deleted: 537

File /u6/jwp/DATA/NUCL4/nucl4009.osc successfully opened.

[... many lines deleted ...]

IMAGE #: 42 Name: /u6/jwp/DATA/NUCL4/nucl4043.osc

Rotation list:
Start: 34.5
End: 34.75
Increment: 0.25
Time: 60
Predicting reflns...
Done predicting reflns...
Processing reflns ...
there are 1502 in the list.
Number new, active, full, donew, dones:
69, 603, 71, 0, 215
Integrate: start...
Strong: 215, Weak: 0
Integrate: ...done.
Reflections deleted: 970

File /u6/jwp/DATA/NUCL4/nucl4044.osc successfully opened.
Header of file /u6/jwp/DATA/NUCL4/nucl4044.osc successfully read.
Image is 1900 by 1900 pixels.
Data_type in header is unsigned short int.
Compression_type is RAXIS.
Byte_order is big_endian.

IMAGE #: 43 Name: /u6/jwp/DATA/NUCL4/nucl4044.osc

Rotation list:
Start: 34.75
```

```
346 End: 35
347 Increment: 0.25
348 Time: 60
349 Predicting reflns...
350 Done predicting reflns...
351 Processing reflns ...
352 there are 1150 in the list.
353 Number new, active, full, donew, dones:
354 77, 609, 68, 0, 68
355 Reflections deleted: 541
```

Explanation

Line 1 — dtintegrate is started and file `newrefine.head` is read.

Lines 2-3 — The non-uniformity object is initialized based on the information in `newrefine.head`. In this case, there is no non-uniformity to be applied.

Lines 4-8 — Command line options are parsed.

Lines 9-18 — Parameters used by dtintegrate are listed. The values are either the default values or the ones overridden on the command line.

Lines 19-24 — The first image in the scan is read in and information from its header is displayed.

Line 25 — The first image is numbered 0 and its name is listed.

Lines 26-30 — The rotation information from the image is listed. This includes the rotation start, end, increment and time.

Lines 31-34 — Reflections which appear on the image are predicted.

Lines 35-37 — Reflections which appear on the image are processed in a first pass. There were 206 new 3D arrays (shoeboxes) allocated and data from the image moved into them (`new`). There were a total of 206 active reflections (`active`) open in this image. No shoeboxes were filled. (`full`). Since there were no full shoeboxes, none were processed either as weak (`donew`) or strong (`dones`) reflections. Just remember that if you have all weak reflections, then you will not be able to do profile fitting, as you have no strong reflections to compute reference profiles from.

Line 55 — There were 56 new reflections and 334 active reflections in image 2.

Line 116 Finally some shoeboxes were filled up. There were 20 full shoeboxes and there were all preprocessed as strong.

Lines 138-276 This is the refinement loop. Even though `-batch 3` was specified on the command line, there were no reflections available for refinement until the second batch of images was finished.

Consult the tCrefine documentation for an explanation of all of these lines.

Lines 251-252 The root mean square difference between predicted and observed reflections centroids is listed.

Line 277 dtintegrate always writes the refinement results to the file `intrefine.head`.

Line 278 Integration resumes with reading in image 6 with sequence number 7.

Line 319 Many lines were deleted because they are repetitive.

Line 342 The last image with sequence number 44 is read in and processed.

The file `dtintegrate.ref` with the background-peak-background integration of the reflections was written. It has a standard d*TREK reflection file format. Below is a truncated listing of the file produced by the above example.

```
5 14 0
nH
nK
nL
nDetector_number
nNonunf_flag
fIntensity
fSigmaI
fCalc_pixel0
fCalc_pixel1
fCalc_rot_mid
fCalc_rot_width
fCalc_polarz
fCalc_lorentz
fCalc_oblique
fResolution
fObs_pixel0
fObs_pixel1
fObs_rot_mid
fObs_rot_width
    3  -17   -7 0 0 85954.8 602.587 1359.54 1010 24.5148 0.614138 0.924924 2.58067
1.08823 3.82877 1365.76 1010.82 24.5198 -999
   -3   10   -3 0 0 364503 435.747 734.539 974.991 24.9057 0.607173 0.972394 4.25585
1.02933 6.45861 731.83 975.897 24.9105 -999
```

```
     3   18   -1 0 0 660363 780.311 565.905 823.491 24.5262 0.63487 0.926624 2.6104
   1.09269 3.74323 559.694 821.784 24.5122 -999
     9  -16   -2 0 0 41327.2 634.775 1364.66 863.393 24.8027 0.613915 0.922555 2.54091
   1.09131 3.76893 1370.44 862.447 24.7832 -999
     7  -13   -1 0 0 22344.9 387.538 1278.05 881.164 24.3121 0.612952 0.949135 3.13527
   1.05716 4.68836 1282.75 880.649 24.3156 -999
```

The output of the dtprofit program follows.

```
1   !***ERROR in DSKBR, reading error.
2   !***ERROR in DSKBR, reading error.
3   Number of reflection profiles read in: 2625
4   Sum (Pi**2/Vi) = 1001.34
5   File D0000.krefer successfully opened.
6   Success writing file D0000.krefer!

7   Sigma Range   Avg.Corr.  Num.Refln.
8    0... 3         -0.18       26
9    3... 6          0.17       11
10   6... 9         0.164       13
11   9...12         0.218       14
12  12...15         0.255       14
13  15...18         0.301       17
14  18...21         0.376       15
15  21...24         0.315       11
16  24...27         0.324        8
17  27...30         0.842     2496

18  Corr.Range   Num.Refln.
19    0...0.1          57
20  0.1...0.2          28
21  0.2...0.3          39
22  0.3...0.4          53
23  0.4...0.5          63
24  0.5...0.6         135
25  0.6...0.7         199
26  0.7...0.8         253
27  0.8...0.9         339
28  0.9...  1        1459
```

Line 1 The file `reflnprofiles.profit` generated by dtintegrate was read. Reflection profiles were processed until an end-of-file was reached. Reference profiles were calculated.

Line 2 The profiles file was read a second time until an end-of-file was reached. Observed profiles were analyzed with the Kabsch method.

Line 3 There were 2625 reflection profiles in the input file.

Line 4 The sum of the square of the variance weighted reference profiles is 1001.34. This number has no special meaning.

Lines 5-6 A d*TREK image file was written with the overall reference profile in it. This image file can be viewed with dtdisplay as a diagnostic tool.

Lines 7-17 The correlation of the observed profiles to the reference profile in 10 different ranges of I/sigmaI is given. The lower ranges should not have a very high correlation, while the higher ranges should have a correlation above 0.8. These correlations confirm that the assumption that the reflections have the same profiles.

Lines 18-28 The number of reflections in 10 different correlation ranges is given. This also confirms the underlying assumptions of profile analysis.

The reflection results of dtprofit are written to the file `dtprofit.ref` which is a standard d*TREK reflection file format. Below is a truncated example of this file. The fields `fIntensity` and `fSigmaI` contain the profile analyzed reflection intensity. The fields `fOtherInt` and `fOtherSig` contain the background-peak-background intensity estimated by dtintegrate. All intensities and sigmas have been corrected for both the Lorentz and polarization factors which is also included in the field `fLPfactor` as the multiplier.

```
4 12 0
nH
nK
nL
nBadFlag
fIntensity
fSigmaI
fOtherInt
fOtherSig
fObs_pixel0
fObs_pixel1
fObs_rot_mid
fCalc_pixel0
fCalc_pixel1
fCalc_rot_mid
fLPfactor
fCorrelation
     3  -17   -7 0 86174.8 274.128 85954.8 602.587 1365.76 1010.82 24.5198 1359.54 1010 24.5148 0.418949 1.00256
    -3   10   -3 0 365861 322.356 364503 435.747 731.83 975.897 24.9105 734.539 974.991 24.9057 0.241641 1.00373
     3   18   -1 0 644888 574.647 660363 780.311 559.694 821.784 24.5122 565.905 823.491 24.5262 0.413418 0.976566
     9  -16   -2 0 35310.3 229.748 41327.2 634.775 1370.44 862.447 24.7832 1364.66 863.393 24.8027 0.426598 0.854408
     7  -13   -1 0 21440.3 151.118 22344.9 387.538 1282.75 880.649 24.3156 1278.05 881.164 24.3121 0.336045 0.959517
```

4.4 Overcoming difficulties

Since dtintegrate gets all of its input information from an image header, difficulties may arise when the header is incomplete. Be sure that the detector spatial distortion information and non-uniformity information is correct and the files exist in the current directory or the filenames in the header include the path or directory.

The output of dtintegrate clearly shows when any files have not been read properly. Examine the output to ensure that the initial source, crystal, goniometers, detector and rotation properties are correct. If these are incorrect, then the image header is incorrect and should be modified.

In these cases, add or correct the information (use dtheaderedit, see section 5 of the dtpredict manual) and run dtintegrate again.

Consult the dtpredict and tCrefine documentation for help with problems with predicting reflections and refining crystal, detector and source properties.

5 File formats

Image file formats are discussed in the calibrate and dtpredict documentation. An example of an image header is given in Section 3. The `reflnprofiles.profit` file is binary (stream linefeed) and contains the following information:

Incident source vector 3 binary floating point numbers of 4 bytes each
Number of grid points in profiles: grid1, grid2, grid3
3 binary integer numbers of 4 bytes each
Size in degrees of profiles 3 binary floating point numbers of 4 bytes each
Value to flag a bad pixel 1 binary floating point number of 4 bytes
Then for each reflection until the end-of-file,
the following structure,

```
typedef struct _tagProfitRec
{
  int   nH;
  int   nK;
  int   nL;
  float fIntensity;
  float fSigma;
  float fOtherInt;
  float fOtherSig;
  float fBackgAvg;
  float fBackgSig;
  float fBackgNumPx;
  float fWeight;
  float fCalcPx0;
  float fCalcPx1;
  float fCalcRotMid;
  float fObsPx0;
  float fObsPx1;
  float fObsRotMid;
```

```
  float a3fScatt[3];
  float fRecipLP;
} tagProfitRec;
```

followed by the floating point profile (grid1 * grid2 * grid3 * 4 bytes) and the floating point variance in the profile (grid1 * grid2 *grid3 * 4 bytes).